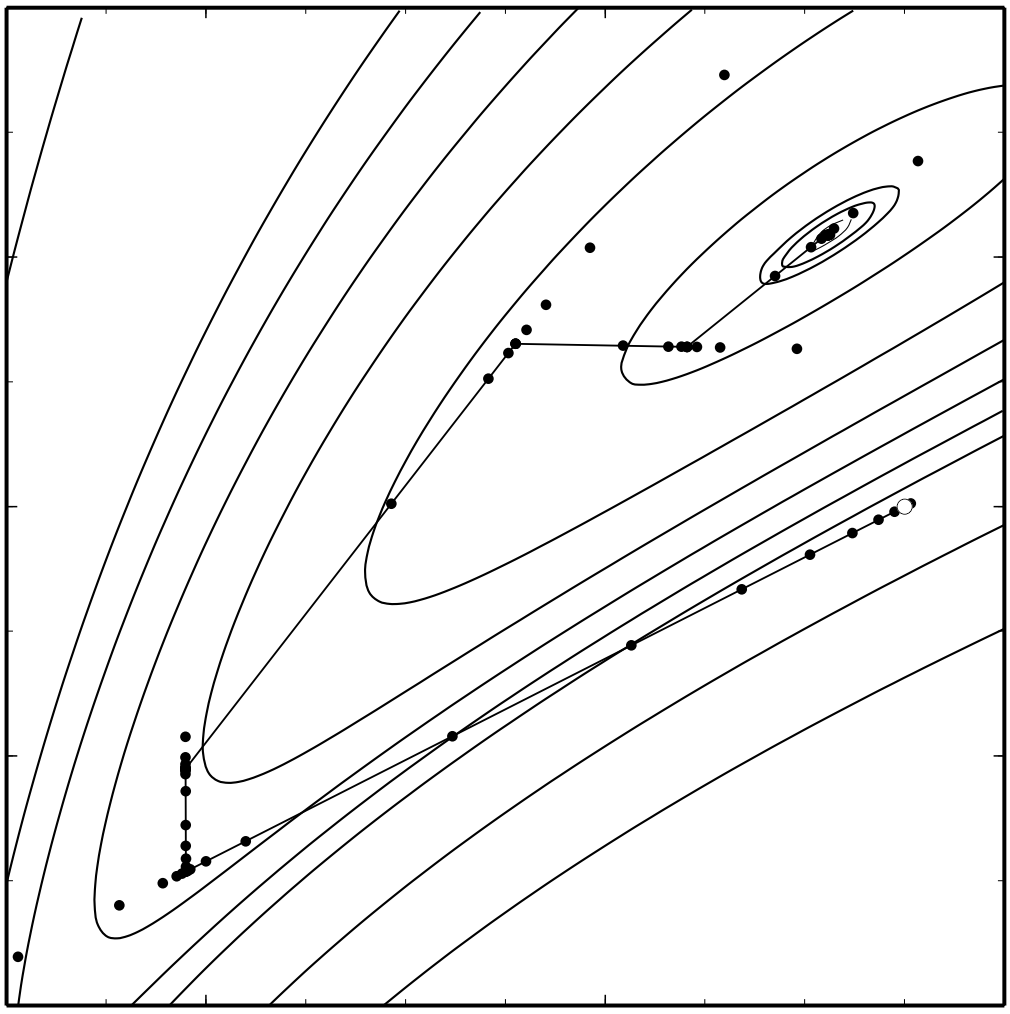
V. Blobel / E.Lohrmann

Statistische und numerische Methoden der Datenanalyse



HAMBURG – 2012

Statistische und numerische Methoden der Datenanalyse

Volker Blobel

Institut für Experimentalphysik
Universität Hamburg

Erich Lohrmann

Institut für Experimentalphysik
Universität Hamburg
und
Deutsches Elektronensynchrotron Desy

HAMBURG – 2012

Volker Blobel, Institut für Experimentalphysik, Universität Hamburg,
Luruper Chaussee 149, D – 22 761 Hamburg

Erich Lohrmann, Deutsches Elektronensynchrotron Desy,
Notkestraße 85, D – 22 607 Hamburg

Die **Buch-Ausgabe**
wurde im Jahre 1998 vom Verlag Teubner Stuttgart; Leipzig
in der Reihe der Teubner Studienbücher publiziert.

Online-Ausgabe erstellt am 21. Mai 2012.

ISBN 978-3-935702-66-9 (e-Buch) <http://www-library.desy.de/elbook.html>

Vorwort zur Buch-Ausgabe

Der Umfang des Datenmaterials in Wissenschaft und Technik nimmt immer schneller zu; seine Auswertung und Beurteilung erweisen sich zunehmend als die eigentliche Schwierigkeit bei vielen wichtigen Problemen. Dem steht auf der anderen Seite ein seit Jahren ungebrochenes Anwachsen der Rechnerleistung und die zunehmende Verfügbarkeit mächtiger Algorithmen gegenüber, aber es ist oft nicht einfach, von diesen Hilfsmitteln den richtigen und professionellen Gebrauch zu machen. Dieses Buch, entstanden aus der Praxis der Verarbeitung großer Datenmengen, will eine Einführung und Hilfe auf diesem Gebiet geben.

Viele der Probleme sind statistischer Natur. Hier ist es sprichwörtlich leicht, Fehler zu machen. Deshalb sind der Erklärung und der kritischen Durchleuchtung statistischer Zusammenhänge auch im Hinblick auf die Praxis ein angemessener Raum gewidmet und ebenso den Monte Carlo-Methoden, welche heute einen verhältnismäßig einfachen Zugang zu vielen statistischen Problemen bieten.

Werkzeuge für die Organisation und Strukturierung großer Datenmengen bilden ein weiteres wichtiges Thema. Dazu gehören auch effiziente Verfahren zum Sortieren und Suchen, welche oft Teil größerer Algorithmen sind.

Die Verarbeitung großer Datenmengen hat oft die Extraktion verhältnismäßig weniger Parameter zum Ziel. Hier sind Verfahren wie die Methoden der kleinsten Quadrate und der Maximum-Likelihood wichtig, in Verbindung mit effektiven Optimierungsalgorithmen. Ein weiteres Problem, welches oft unterschätzt wird, ist die Rekonstruktion ursprünglicher Verteilungen aus fehlerbehafteten Messungen durch Entfaltung.

Mit der Verfügbarkeit mathematischer Bibliotheken für Matrixoperationen können viele Probleme elegant in Matrixschreibweise formuliert und auf Rechnern gelöst werden. Deswegen werden auch die einfachen Grundlagen der Matrixalgebra behandelt.

Das Buch führt in diese Dinge ein, es ist gedacht für Naturwissenschaftler und Ingenieure, die mit Problemen der Datenverarbeitung befaßt sind. Neben Algorithmen dienen zahlreiche einfache Beispiele der Verdeutlichung.

Um der Unsitte entgegenzuwirken, mehr oder weniger gut verstandene Rezepte schwarzeschachtelartig anzuwenden, wird auf Erklärungen und Beweise innerhalb eines vernünftigen Rahmens großer Wert gelegt, ebenso auf eine kurze Einführung des Informationsbegriffs.

Programmcodes von im Text erwähnten Algorithmen sind, sofern sie einen solchen Umfang haben, daß Kopieren von Hand nicht ratsam ist, im Internet zugänglich gemacht (`http://www.desy.de/~blobel`). Sie sind von den Autoren benutzt und überprüft worden, trotzdem kann für ihre Korrektheit und für die Abwesenheit von Nebeneffekten keine Garantie übernommen werden. Kurze Programme sind meist in Fortran, in einigen Fällen in C formuliert. Sie sollen zur Präzisierung der Algorithmen dienen. Sie sind kurz genug, so daß sie jeder in seine Lieblingssprache umschreiben kann.

Vielen Kollegen, vor allem Dr. F. Gutbrod und Dr. D. Haidt sind wir dankbar für wertvolle Diskussionen, Informationen und Anregungen. Frau A. Blobel und Frau U. Rehder danken wir für ihre Hilfe bei der Erstellung des Manuskripts, Herrn Dr. P. Spuhler vom Teubner-Verlag danken wir für die gute Zusammenarbeit.

Hamburg, im Juli 1998 V. Blobel und E. Lohrmann

Vorwort zur online-Ausgabe

Nachdem das Buch im Handel nicht mehr erhältlich sein wird, möchten wir es in dieser Form zugänglich machen.
Verglichen mit der ersten Auflage haben wir einige Ergänzungen und Verbesserungen vorgenommen und Literaturhinweise ergänzt.
Da sich seit der ersten Ausgabe die objekt-orientierte Programmierung weitgehend durchgesetzt hat, haben wir die Fortran-Programme aus dem laufenden Text entfernt und im Anhang untergebracht. Da kann sie jeder in seine Lieblingssprache umschreiben.

Hamburg, im April 2012 V. Blobel und E. Lohrmann

Inhaltsverzeichnis

1 Datenbehandlung und Programmierung

1.1 Information

Jede Art von Datenverarbeitung erfordert Methoden, Daten zu repräsentieren, zu codieren, und zu übertragen. Im Mittelpunkt steht dabei der Informationsbegriff. Durch Datenübertragung kann Information an einen Empfänger übermittelt und der Empfänger zu einer Handlung veranlaßt werden.

Zur Übertragung von Information dienen Daten, die als Folge von Symbolen aufgefaßt werden können. Ein Fernsehbild kann zum Beispiel durch eine Folge von Binärzahlen als Symbolen repräsentiert werden. Ein geschriebener Text kann durch eine Folge von alphabetischen und numerischen Symbolen (Zeichen) repräsentiert werden, unter Hinzunahme von Interpunktions- und Leerzeichen. Die Menge aller Symbole, die in einer Nachricht vorkommen können, heißt ihr *Alphabet*. Eine *Nachricht* ist eine Sequenz von Symbolen aus einem Alphabet.

Beispiele für Alphabete:

1. die Buchstaben des deutschen Alphabets a, b, c, ... z;
2. alle Buchstaben des deutschen Alphabets, zusätzlich die Ziffern $0-9$, Interpunktionszeichen, Sonderzeichen, Leerzeichen und Zeilenende-Markierung;
3. alle Wörter der deutschen Sprache;
4. die zwei Binärzahlen 0, 1;
5. die zwei Buchstaben A, B;
6. Buchstaben-Gruppen wie zum Beispiel AB, AAB, ABB, WORT, ...;
7. Binärzahlen wie zum Beispiel 0, 10, 110, 111, ...

Man möchte nun zu einer formalen Definition der durch eine Nachricht übertragenen Informationsmenge kommen. Diese Definition sollte qualitativ mit dem intuitiven Verständnis von Information übereinstimmen. Intuitiv assoziiert man einen großen Informationsgehalt mit einer überraschenden Nachricht, *man hat sie nicht erwartet*. Zum Beispiel enthält die Nachricht 'es wird morgen in der Saharawüste regnen' mehr Information als die Nachricht 'es wird morgen in Hamburg regnen', da die erste Nachricht überraschender und unwahrscheinlicher ist als die zweite.

Eine formale Definition der Informationsmenge basiert daher auf dem Begriff der Wahrscheinlichkeit insofern, als eine unwahrscheinliche Nachricht mehr Information enthält als eine wahrscheinliche.

Definition: Die Einheit der Informationsmenge erlaubt die Unterscheidung zwischen zwei gleichwahrscheinlichen Möglichkeiten. Diese Einheit wird BIT[1] genannt.

Um ein bestimmtes Symbol aus N gleichwahrscheinlichen Symbolen auszuwählen, teilt man die Menge von N Symbolen in zwei gleich große Mengen von je $N/2$ Symbolen; um zu entscheiden, welcher Hälfte das gesuchte Symbol angehört, benötigt man 1 BIT an Information. Dieser Prozeß der Teilung wird wiederholt, bis das gesuchte Symbol isoliert ist. Es werden $\lg N$ solcher Schritte benötigt ($\lg N = \log_2 N$), und dementsprechend insgesamt $\lg N$ BITs an Information. Die Wahrscheinlichkeit P_N eines jeden der N gleichwahrscheinlichen Symbole ist (siehe Kapitel 4.2)

$$P_N = \frac{1}{N} \, .$$

Die benötigte Informationsmenge, um ein gegebenes Symbol auszuwählen, ist also

$$H = \lg N = -\lg P_N \qquad [\mathrm{BIT}] \, .$$

[1] BIT ist hier in Großbuchstaben gesetzt, um es von dem Wort *bit* (*binary digit*, 0 oder 1) zu unterscheiden.

Für den allgemeinen Fall betrachtet man eine Folge von M Symbolen. Diese Symbole werden einem Alphabet entnommen; die Wahrscheinlichkeit für das Vorkommen des Symbols i des Alphabets in dieser Folge sei P_i, $i = 1, 2, \ldots N_s$, wobei N_s die Zahl der Symbole des Alphabets ist.

Das Vorkommen eines Symbols i in einer Nachricht überträgt die Informationsmenge $-\lg P_i$; insgesamt gibt es $N_i = P_i \cdot M$ solcher Symbole, ihr Informationsgehalt ist $-N_i \lg P_i = -MP_i \cdot \lg P_i$. Summiert man über alle N_s Symbole des Alphabets und teilt durch die Anzahl M der Symbole der Nachricht, so erhält man H, die **mittlere Informationsmenge/Symbol**:

$$H = -\sum_{i=1}^{N_s} P_i \lg P_i \qquad [\text{BIT}] \,. \tag{1.1}$$

Diese wichtige Formel gilt nur, wenn die Symbole der Folge statistisch unabhängig sind (siehe Kapitel 4.2). Diese Bedingung ist nicht immer erfüllt, zum Beispiel wenn die Symbole die Buchstaben des deutschen Alphabets sind und die Folge Worte einer Sprache darstellt. Offensichtlich sind die Buchstaben des Alphabets in den Worten einer Sprache korreliert, zum Beispiel folgt fast immer ein u auf ein q.

Als einen wichtigen Spezialfall betrachtet man das Alphabet der Binärzeichen (bits) 0 und 1. Die Wahrscheinlichkeit der 0 sei $P(0) = p$, dann ist die Wahrscheinlichkeit der 1 gegeben durch $P(1) = 1 - p$, und die mittlere Informationsmenge/bit in einer Kette binärer Zeichen 0 und 1 ergibt sich zu

$$H_2 = -p \lg p - (1-p) \lg(1-p) = S(p) \,. \tag{1.2}$$

Abbildung 1.1 zeigt die Funktion $S(p)$. Sie hat ein Maximum bei $p = 1/2$ und ist null für $p = 0$ und $p = 1$. Eine Folge, die ausschließlich aus den Zeichen 0 oder 1 besteht, hat daher keinen Informationsgehalt; ein Code mit guter Effizienz der Informationsübertragung/bit sollte im Mittel eine gleiche Anzahl der bits 0 und 1 haben.

Entropie. Man betrachtet ein System aus N gleichen und wechselwirkenden Teilchen, zum Beispiel Gasmolekülen. Die Anzahl von Teilchen mit Energie E_i sei $n_i = N \cdot P_i$. Die Anzahl von Möglichkeiten W, die N Teilchen in M Bereichen mit Energie E_i $(i = 1, 2, \ldots M)$ anzuordnen, ist

$$W = N! / \prod_{i=1}^{M} n_i! \,.$$

Die *Entropie* des Systems ist gegeben durch $S = k \ln W$; dabei ist k die Boltzmann-Konstante. Mit der Stirling-Formel in ihrer einfachen Form $x! \cong (x/\mathrm{e})^x$ ergibt sich

$$\ln W = -N \sum P_i \ln P_i \,,$$

unter Benutzung von $\sum_i P_i = 1$ und $\sum_i n_i = N$, und daher

$$S = -kN \sum P_i \ln P_i \,.$$

Diese Formel hat eine formale Ähnlichkeit mit Gleichung (1.1); daher werden Informationsmenge und Entropie manchmal als Synonyme benutzt.

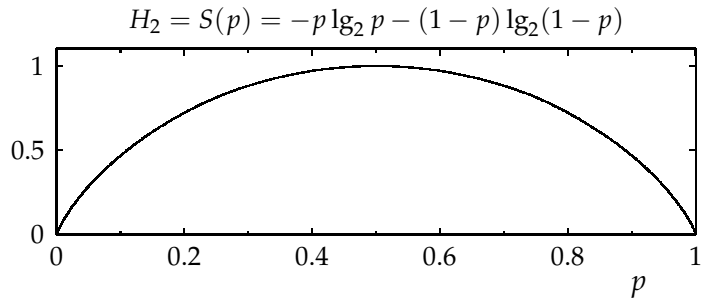


Abbildung 1.1: Die mittlere Informationsmenge pro bit, $S(p)$, in einer Kette von Binärziffern (0 und 1) als Funktion von p, der Wahrscheinlichkeit des Vorkommens von Bit 0.

1.2 Codierung

Codierung bedeutet, daß die Symbole einer Nachricht anhand einer eineindeutigen Beziehung von einem Alphabet in die Symbole eines anderen Alphabets übersetzt werden. Ein wichtiger Spezialfall ist die Codierung in binäre Zahlen (0, 1, 10, 11, 100, ...). Diese führt auf eine Kette der Ziffern 0 und 1. Wenn die Binärzahlen nicht eine feste Länge haben, muß man in der Lage sein, Anfang und Ende einer jeden Binärzahl zu bestimmen, d.h. die Symbole des Alphabets müssen die *Präfixeigenschaft* besitzen: Kein Binärsymbol darf der Anfang eines anderen Binärsymbols sein.

Die folgende Tabelle zeigt zum Beispiel vier Symbole eines Alphabets, die in einen Binärcode mit Präfixeigenschaft transformiert wurden:

Alphabet 1	Alphabet 2 (binär)
AA	0
AB	10
BA	110
BB	111

Codierungstheorem. Man betrachtet eine Nachrichtenquelle mit einem mittleren Informationsgehalt von $\langle H \rangle = R$ BITs/Nachricht. Es ist möglich, eine Folge von Nachrichten in einem Binärcode zu codieren, so daß im Mittel weniger als $R + \epsilon$ (mit $\epsilon > 0$) bits/Nachricht benötigt werden. Es ist keine Codierung mit weniger als R bits/Nachricht möglich.

Beispiel 1.1 Codierung von Sprache.

Tabelle 1.1 zeigt die relative Häufigkeit, mit der die Buchstaben des Alphabets in der deutschen Sprache vorkommen. Mit Gleichung (1.1) berechnet sich der mittlere Informationsgehalt eines Buchstabens zu

$$\langle H \rangle = - \sum_{\text{alle Buchstaben}} P_i \lg P_i = 4.1 \text{ BIT/Buchstabe} .$$

Da 30 Symbole in Tabelle 1.1 vorhanden sind, könnten alle einfach als 5-bit-Binärzahlen codiert werden. Tatsächlich benutzte Codierungen umfassen auch Ziffern und Spezialzeichen; sie benötigen daher mehr als 5 bits. Beispiele sind BCD (6 bits), EBCDIC and ASCII (8 bits). Tabelle 1.2 zeigt den Standard ASCII Zeichensatz. Für eine realistische Bestimmung des mittleren

Symbol	P_i	Code	Symbol	P_i	Code
-	0.151	000	O	0.018	111001
E	0.147	001	B	0.016	111010
N	0.088	010	Z	0.014	111011
R	0.068	0110	W	0.014	111100
I	0.064	0111	F	0.014	1111010
S	0.054	1000	K	0.009	1111011
T	0.047	1001	V	0.007	1111100
D	0.044	1010	Ü	0.006	1111101
H	0.043	10110	P	0.005	1111110
A	0.043	10111	Ä	0.005	11111110
U	0.032	11000	Ö	0.003	111111110
L	0.029	11001	J	0.002	1111111110
C	0.027	11010	Y	0.0002	11111111110
G	0.027	11011	Q	0.0001	111111111110
M	0.021	111000	X	0.0001	111111111111

Tabelle 1.1: Relative Häufigkeit P_i der Buchstaben des Alphabets in der deutschen Sprache. Ebenfalls dargestellt ist der mittels der Huffman-Methode (siehe Beispiel 1.2) konstruierte Code.

Informationsgehalts von tatsächlichem Text müßte man von den Symbolen in dieser Tabelle und ihrer Häufigkeit ausgehen.

Die Codierung kann verbessert werden, wenn man kurze Binärzahlen einsetzt, um die häufigsten Buchstaben zu codieren; für die selteneren werden dementsprechend längere Binärzahlen verwendet.

Ein optimaler Code, wie er in Tabelle 1.1 dargestellt ist, benutzt im Mittel nur 4.3 bits/Buchstabe. Aber dies ist noch nicht das Ende der möglichen Verbesserungen. Offensichtlich sind die Buchstaben in den Wörtern korreliert. Man könnte versuchen, jedes Wort einer Sprache durch eine Binärzahl zu codieren. Wenn man die Wörter der deutschen Sprache nach der Häufigkeit der Benutzung ordnet, erhält man eine Liste wie *die, der, und, zu, in, ein, an, den, auf, das, ...*. Zipfs empirisches Gesetz besagt, daß die Wahrscheinlichkeit des Auftretens des Wortes an der Position i der obigen Liste sich annähernd ergibt zu

$$P_i = 0.1/i \ .$$

Die Anzahl der Wörter N in einer Sprache ist nach diesem Gesetz gegeben durch die Gleichung

$$\sum_{i=1}^{N} 0.1/i = 1 \ ,$$

was auf $N = 12367$ führt. Natürlich ist Zipfs Gesetz nur eine Näherung. Der mittlere Informationsgehalt eines Wortes ist in dieser Näherung

$$\langle H_W \rangle = \sum_{i=1}^{12367} (0.1/i) \lg (10i) = 9.7 \text{ BIT/Wort} \ .$$

Da im Mittel 5.6 Buchstaben auf ein Wort kommen, erhält man $9.7/5.6 = 1.7$ BIT/Buchstabe, und entsprechend dem Codierungstheorem sollte es im Prinzip möglich sein, im Mittel nur 1.7 bits/Buchstabe einzusetzen, um Text zu codieren. Natürlich würde dies ein Alphabet von mehr als 12000 Symbolen und eine optimale Codierung voraussetzen.

Huffman-Codierung. Bei dieser Methode handelt es sich um eine optimale Codierung der Symbole eines Alphabets durch Binärzahlen. Die Methode stellt sicher, daß sie die Präfixeigenschaft besitzen. Der Algorithmus ist der folgende:

1. Man erstelle eine Liste aller Symbole des Alphabets, nach absteigender Auftrittswahrscheinlichkeit geordnet.
2. Das letzte Symbol der Liste erhält eine 1, das vorletzte eine 0 als letzte Ziffer des Binärcodes.
3. Man verbinde die letzten beiden Symbole und addiere ihre Wahrscheinlichkeiten.
4. Man verschiebe sie an den richtigen Platz in der geordneten Liste.
5. Man fahre fort bei 2.

Beispiel 1.2 Huffman-Codierung.

Die folgende Tabelle zeigt die Konstruktion eines Huffman-Codes für ein Alphabet von fünf Buchstaben mit einer gegebenen Wahrscheinlichkeit P_i. Die letzte Spalte zeigt das Endergebnis.

	P_i			P_i			P_i					endgültiger Code
A	0.66		A	0.66		A	0.66		A	0	A	0
B	0.16		B	0.16		CDE	0.18	0	BCDE	1	B	11
C	0.15		C	0.15	0	B	0.16	1			C	100
D	0.02	0	DE	0.03	1						D	1010
E	0.01	1									E	1011

Datenkomprimierung. Die Codierung von Daten in kompakter Form wird häufig eingesetzt. Zum Beispiel kann die Menge der zur Übertragung und Speicherung von Farbbildern in digitaler Form benötigten Daten stark reduziert werden durch geschickt gewählte Codierungsverfahren. Ein ähnliches Verfahren kann eingesetzt werden, um Musik auf CDs zu komprimieren. Datenkomprimierungsalgorithmen sind heutzutage auf vielen Computersystemen verfügbar.

Der Grad, bis zu dem Datenkompression durch einen Binärcode erreicht werden kann, wird anhand des Begriffes der *Code-Redundanz* R gemessen. Diese ist gegeben durch

$$R = B - H\,.$$

Dabei ist H der mittlere Informationsgehalt/Symbol, bestimmt durch Gleichung (1.1) und B die mittlere Anzahl von bits/Symbol des binären Codes. Gelegentlich ist eine gewisse Redundanz nützlich (siehe Kapitel 1.3). Religiöse Texte und Reden zum Beispiel besitzen häufig eine große Redundanz.

Der in Tabelle 1.2 aufgeführte ASCII -Zeichensatz definiert eine 1-zu-1-Abbildung von Zeichen [ch] (inklusive Spezialzeichen) nach 8-bit-Ganzzahlen [int], für die angegebenen Zeichen werden allerdings nur 7 bits benötigt. Daneben gibt es noch andere Konventionen. Das kann z.B. zur Konsequenz haben, dass ein Passwort, das Sonderzeichen entält, nicht von allen Tastaturen gleich wiedergegeben wird.

1.3 Informationsübertragung

Ein Datenübertragungskanal ist charakterisiert durch die Anzahl von bits (0 oder 1), die er pro Sekunde übertragen kann. Diese Größe wird gemessen in Baud: 1 Baud (Bd) = 1 bit/s an übertragenen Daten. Aus dem Codierungstheorem folgt, daß die Übertragungsrate in Bd eine obere Grenze für die Informationsmenge (in BIT) ist, die pro Sekunde über diesen Datenkanal übertragen werden kann. Diese maximale Übertragungsrate heißt *Kapazität*. Die Übertragungsrate, die für Kommunikationsnetzwerke angegeben wird, ist wieder nur eine obere Grenze für

int	ch	int	ch	int	ch	int	ch	int	ch	int	ch	int	ch	int	ch
0		1		2		3		4		5		6		7	
8		9		10		11		12		13		14		15	
16		17		18		19		20		21		22		23	
24		25		26		27		28		29		30		31	
32	blk	33	!	34	"	35	#	36	$	37	%	38	&	39	’
40	(	41	)	42	*	43	+	44	,	45	-	46	.	47	/
48	0	49	1	50	2	51	3	52	4	53	5	54	6	55	7
56	8	57	9	58	:	59	;	60	¡	61	=	62	¿	63	?
64	@	65	A	66	B	67	C	68	D	69	E	70	F	71	G
72	H	73	I	74	J	75	K	76	L	77	M	78	N	79	O
80	P	81	Q	82	R	83	S	84	T	85	U	86	V	87	W
88	X	89	Y	90	Z	91	[	92	\	93	]	94	ˆ	95	_
96	‘	97	a	98	b	99	c	100	d	101	e	102	f	103	g
104	h	105	i	106	j	107	k	108	l	109	m	110	n	111	o
112	p	113	q	114	r	115	s	116	t	117	u	118	v	119	w
120	x	121	y	122	z	123	{	124	|	125	}	126	˜	127	

Tabelle 1.2: Der ASCII-Zeichensatz. Das Leerzeichen (blank) hat Code 32.

die Menge nützlicher Binärdaten, die das Netzwerk tatsächlich übertragen kann, da zusätzlich Daten der Netzwerkverwaltung und -steuerung, Protokollinformationen und anderes mehr hinzukommen. Außerdem, wie beim wirklichen Verkehr, können Netzwerke blockieren, wenn das Verkehrsaufkommen sich der maximalen Kapazität nähert.

Man betrachtet einen idealen, fehlerfreien Kanal mit einer Kapazität C. Unter Verwendung eines optimalen Codes kann die Informationsmenge C mit $N \approx C$ bits übertragen werden. Ein wirklicher Kanal ist nicht fehlerfrei, sondern besitzt eine Wahrscheinlichkeit p, ein bit (0 oder 1) fehlerhaft zu übertragen (in vielen modernen Installationen ist p sehr klein, $p \ll 10^{-10}$). Diese Daten können im Prinzip durch gleichzeitige Übermittlung eines Stromes von Korrekturbits korrigiert werden. Eine 1 würde bedeuten, daß das bit korrekt ist, ansonsten würde eine 0 übertragen (in der Praxis würde man eine bessere Methode wählen, um die gleiche Information mit weniger bits zu übermitteln). Die Menge der in den Korrekturbits vorhandenen Information ergibt sich zu

$$H_{\text{corr}} = N\,(-p \lg p - (1-p)\lg(1-p)) = N \cdot S(p)\ ,$$

und die Kapazität des fehlerbehafteten Kanals ist daher

$$C' = C - H_{\text{corr}} \cong C\,(1 - S(p))$$

mit $N \sim C$.

Fehlererkennung/Fehlerkorrektur. Es gibt eine Vielzahl von fehlererkennenden/fehlerkorrigierenden Codes auf dem Markt. Eine simple Fehler-Erkennungsmethode ist der Paritätstest: Ein Paritätsbit wird an jedes binäre Wort angehängt; das Paritätsbit wird zu 0 oder 1 gewählt, so daß die Anzahl der 1-bits im Wort gerade ist. Das ist Konvention; man kann auch ungerade wählen. Dieser Test kann einen Fehler in einem Wort finden, aber nicht korrigieren.

Fortgeschrittene Verfahren können Fehler erkennen und korrigieren (zum Beispiel Hamming-Code). Eine einfache, wirksame, aber unökonomische Prüfmethode ist die Reflexion: Die Nachricht wird zum Sender zurückgesandt und dort mit der ursprünglichen Nachricht verglichen.

1.4 Analogsignale – Abtasttheorem

Das Whittaker-Shannonsche Abtasttheorem liefert die Anzahl von Abtastpunkten, die benötigt wird, um ein analoges (kontinuierliches) Signal ohne Informationsverlust zu digitalisieren. Um

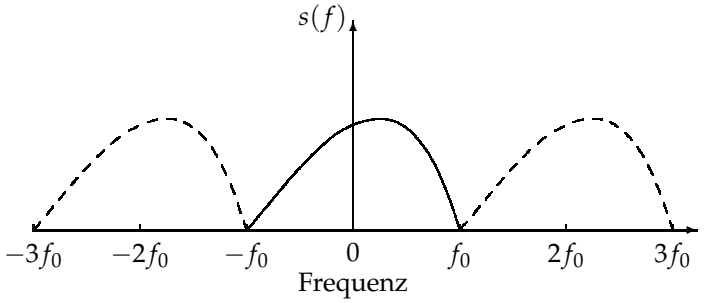


Abbildung 1.2: Die Fouriertransformierte $s(f)$.

es herzuleiten, betrachtet man ein analoges Signal $u(t)$; die unabhängige Variable t bezeichne der Einfachheit halber die Zeit. Das Signal kann durch ein Fourier-Integral dargestellt werden

$$u(t) = \int_{-\infty}^{+\infty} s(f) e^{2\pi i f t} \, \mathrm{d}f \tag{1.3}$$

mit $f =$ Frequenz, $s(f) =$ Fouriertransformierte. Nun wird in allen Anwendungen das Fourier-Spektrum bei einer oberen Frequenz f_0 abgeschnitten; f_0 heißt die Bandbreite des Signals. Daher kann man schreiben

$$u(t) = \int_{-f_0}^{+f_0} s(f) e^{2\pi i f t} \, \mathrm{d}f \, . \tag{1.4}$$

Die Fouriertransformierte $s(f)$ ist in Abbildung 1.2 dargestellt. Per definitionem erstreckt sie sich von $-f_0$ bis $+f_0$. Da das Fourier-Integral (1.4) von $u(t)$ sich nur zwischen den Grenzen $\pm f_0$ erstreckt, kann $s(f)$ außerhalb dieser Grenzen periodisch fortgesetzt werden, wie in Abbildung 1.2 angedeutet, ohne $u(t)$ zu verändern. Die neue Funktion, $s'(f)$, ist in f periodisch mit einer Periode $2f_0$. Zwischen $-f_0$ und $+f_0$ ist sie mit $s(f)$ identisch, daher gilt

$$u(t) = \int_{-f_0}^{+f_0} s'(f) e^{2\pi i f t} \mathrm{d}f. \tag{1.5}$$

Nun kann $s'(f)$ aufgrund der Periodizität in f als eine Fourier-Reihe mit Periode $2f_0$ geschrieben werden

$$s'(f) = \sum_{n=-\infty}^{+\infty} a_n e^{-2\pi i n f / 2f_0} \tag{1.6}$$

mit den Fourier-Koeffizienten

$$a_n = \frac{1}{2f_0} \int_{-f_0}^{+f_0} s'(f) e^{2\pi i n f / 2f_0} \, \mathrm{d}f. \tag{1.7}$$

Man betrachtet nun den Spezialfall $t = n/2f_0$ ($n = 0, 1, 2, \ldots$). In Gleichung (1.5) eingesetzt, ergibt sich

$$u\left(\frac{n}{2f_0}\right) = \int_{-f_0}^{+f_0} s'(f) e^{2\pi i n f / 2f_0} \, \mathrm{d}f = 2f_0 \, a_n$$

durch Vergleich mit Gleichung (1.7).

Die Fourier-Koeffizienten a_n stehen daher in direkter Beziehung zu den Werten $u(n/2f_0)$ des Signals, und die Fouriertransformierte kann geschrieben werden als

$$s'(f) = \frac{1}{2f_0} \sum_{n=-\infty}^{+\infty} u\left(\frac{n}{2f_0}\right) e^{-2\pi i n f/2f_0} . \tag{1.8}$$

Die Funktion $s'(f)$ definiert das Signal eindeutig anhand von Gleichung (1.5), daher kann die Signalfunktion eindeutig beschrieben werden durch die Werte $u(n/2f_0)$ zu den Abtastzeiten

$$t_n = \frac{n}{2f_0}, \qquad n = 0, 1, 2, \ldots. \tag{1.9}$$

Daraus folgt das **Abtasttheorem**: *Die Funktion $u(t)$ ist eindeutig und ohne Informationsverlust darstellbar durch Abtastungen in Zeitintervallen $\Delta t = 1/2f_0$.*

Kapazität eines analogen Datenübertragungskanals. Was ist die größtmögliche Übertragungsrate von Analoginformationen für einen gegebenen Übertragungskanal? Eine einfache Antwort kann durch eine Näherung gefunden werden.

Man betrachtet einen Übertragungskanal mit Bandbreite f_0, zum Beispiel ein Koaxialkabel mit einem Treiber an einem Ende und einem ADC am anderen Ende. Die im analogen Signal enthaltene Information kann durch $2f_0$-maliges Abtasten pro Sekunde extrahiert werden (Gleichung (1.9)). Wie präzise kann das Signal gemessen werden? Das hängt vom Rauschen ab. Mit der Bezeichnung $S =$ Signalleistung und $N =$ Rauschleistung kann die Amplitude des Signals in Schritten von $\sqrt{N}$ gemessen werden; insgesamt können $m = \sqrt{(N+S)/N}$ Schritte gebildet werden. Diese Information kann in $\lg m = 1/2 \lg(1 + S/N)$ bits/s codiert werden. Die Kapazität des Kanals ist daher

$$C \leq 2f_0 \cdot 1/2 \lg\,(1 + S/N) = f_0 \cdot \lg(1 + S/N)\ \text{BIT/s} . \tag{1.10}$$

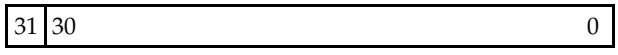
Im Prinzip kann C durch große Signal- bzw. geringe Rauschleistung vergrößert werden, aber das unterliegt offensichtlichen praktischen Grenzen.

Kann Information ohne Energieübertragung übermittelt werden, d.h. mit $S = 0$? Dies würde $N = 0$ bedingen. In der klassischen Physik ist keine verschwindende Rauschleistung erlaubt, da der absolute Nullpunkt nicht erreicht werden kann. In der Quantenmechanik ist die Situation aufgrund von Vakuumfluktuationen nicht besser.

1.5 Repräsentation numerischer Daten

Numerische Daten sind in Binärform codiert, zum Beispiel $1_{10} = 1_2$, $2_{10} = 10_2$, $3_{10} = 11_2$, $4_{10} = 100_2$, $5_{10} = 101_2, \ldots$

Ganzzahlrepräsentation.
Hierfür gibt es in der Praxis ein Reihe verschiedener Konventionen.
Das bit in der ersten (linken) Stelle in einem Wort ist in einigen Repräsentationen das Vorzeichenbit (0 = positiv, 1 = negativ). Eine häufig verwendete Repräsentation negativer Ganzzahlen ist das *2-Komplement*: 2-Komplement = 1-Komplement + 1, wobei man das 1-Komplement durch Ersetzen von 0 durch 1 und 1 durch 0 in der Binärrepräsentation erhält. Die Differenz $a - b$ zweier positiver Zahlen a und b ist dann die Summe von a und dem 2-Komplement von b: $a - b = a +$ (2-Komplement von b).



Ein 32-bit Wort

In der IEEE-Norm sind lange Ganzzahlen (long integer) in einem 32-bit-Wort gespeichert, unter Verwendung des 2-Komplements für negative Ganzzahlen. Daher ist die erste Position das Vorzeichen-bit. Der abgedeckte Zahlenbereich erstreckt sich von $-2^{31} \to 2^{31} - 1$.

Berechnungen mit Ganzzahlarithmetik erfordern große Sorgfalt, da die Ergebnisse von Berechnungen innerhalb des abgedeckten Zahlenbereichs liegen müssen *inklusive* aller Zwischenergebnisse.

Gleitkommazahlen.
Auch hierfür gibt es viele verschiedene Repräsentationen. Beispiele: Eine reelle Zahl Z wird in folgender Binärform gespeichert:

$$Z = (1 - 2b_s) \cdot F \cdot 2^E \,,$$

wobei F ein binärer Bruch < 1 ist, also $F = (b_1 \cdot 2^{-1} + b_2 \cdot 2^{-2} + \cdots)$; $b_1, b_2, b_3, \cdots$ sind binäre bits (also 0 oder 1), und b_s ist das Vorzeichenbit. Normalerweise ist F so normiert, daß $|F| \geq 1/2$ ist, d.h. für das erste bit gilt $b_1 = 1$; einige Rechner lassen es daher weg. E ist der Exponent.

In der IEEE-Norm wird eine Gleitkommazahl (mit einfacher Genauigkeit) in einem 32-bit-Wort gespeichert: 1 bit für das Vorzeichen des Bruchs, 8 bit für den Exponenten, 23 bit für den Bruch. Um den Exponenten als positive Zahl darzustellen, wird zu E die Zahl 127 hinzuaddiert. Der darstellbare numerische Bereich ist daher

$$\begin{array}{rcr} -3.4 \cdot 10^{38} & \text{bis} & -1.17 \cdot 10^{-38} \\ 1.17 \cdot 10^{-38} & \text{bis} & 3.4 \cdot 10^{38} \end{array}$$

'Null' wird durch eine 0 in jedem bit-Feld dargestellt. Man beachte, daß es einen Bereich ganz kleiner Zahlen gibt, die (mit der Ausnahme der 0) nicht dargestellt werden können, und zwar $-1.17 \cdot 10^{-38}$ bis $1.17 \cdot 10^{-38}$.

Gleitkommazahldarstellung mit einfacher Genauigkeit in einem 32-bit-Wort.

Probleme beim Rechnen mit Gleitkommazahlen. Eine Gleitkommazahl mit einem Bruch von zum Beispiel 23 bit hat eine Genauigkeit, die etwa 7 Dezimalstellen entspricht. Für viele Anwendungen kann dies nicht ausreichend sein, und doppelte Genauigkeit wird erforderlich. Bei doppelter Genauigkeit wird eine Zahl in zwei Rechnerworten gespeichert, dadurch werden Genauigkeit und abgedeckter Zahlenbereich stark vergrößert. Falls dies immer noch nicht ausreicht, kann auf einigen Rechnern noch größere Genauigkeit verwendet werden. Abhängig von der Architektur des Rechners ist doppelte Genauigkeit nicht unbedingt langsamer als einfache Genauigkeit, aber natürlich wird mehr Speicherplatz für die Zahlen benötigt.

Eine oft unterschätzte Fehlerquelle sind Rundungsfehler. Man bedenke, daß ein typischer Rechner pro Sekunde viele Millionen Rundungsfehler machen kann. Wenn die am wenigsten signifikanten Ziffern einer Zahl im Verlauf einer Rechnung fortgelassen werden müssen, gibt es zwei Möglichkeiten:

$$\begin{array}{lll} \text{abhacken} & \text{zum Beispiel} & 1001 \to 100 \\ & & 1010 \to 101 \\ \text{runden} & \text{zum Beispiel} & 1001 \to 101 \\ & & 1010 \to 101 \quad . \end{array}$$

Es geht die Sage von jemandem, der *runden* in *abhacken* in dem Assemblercode für Banktransaktionen änderte. Die durchschnittliche Differenz von 0.005 DM/Transaktion transferierte er auf sein Privatkonto. Diese alte Sage sollte heute nicht mehr in die Tat umsetzbar sein.

Zahlen wie 0.1 oder $1/3$ können in Binärdarstellung nicht exakt repräsentiert werden; es handelt sich um unendliche periodische binäre Brüche, zum Beispiel $0.1 = 0.000\overline{1100}\cdots$.

Man sollte bei Vergleichen stets vorsichtig sein, wenn man Gleitkommaarithmetik verwendet. Was ist beispielsweise der Wert von $Z = 10 \cdot 0.1 - 1$? Dies ist abhängig von der Systemarchitektur. In einigen Fällen mag allerdings tatsächlich $Z = 0$ herauskommen.
Ein ähnliches Beispiel ist das Programmstück

```
A = 0.1
B = A
IF (A.EQ.B) THEN
```

Die Gleichheit wird unter Umständen nicht erkannt, wenn A und B in Registern unterschiedlicher Wortlänge gehalten werden.
Vorsicht sollte man auch bei der Subtraktion zweier Zahlen fast gleicher Größe walten lassen, zum Beispiel $1 - \cos\epsilon$ für $\epsilon \ll 1$.

Beispiel 1.3 Lösungen der quadratischen Gleichung $ax^2 + bx + c = 0$.

Quadratische Gleichungen haben für $b^2 \geq 4ac$ zwei reelle Lösungen. Eine Lösung der Gleichung für den Fall $b > 0$ ist $x_1 = (-b - \sqrt{b^2 - 4ac})/2a$. Die andere Lösung x_2 sollte der Beziehung $x_1 \cdot x_2 = c/a$ entnommen werden und *nicht* $x_2 = (-b + \sqrt{b^2 - 4ac})/2a$. Mathematisch sind diese Lösungen natürlich äquivalent, aber letztere kann zu numerischen Problemen aufgrund von Rundungsfehlern führen, wenn $4ac \ll b^2$ ist.

Beispiel 1.4 Berechnung kritischer Ausdrücke.

Man betrachtet den Ausdruck

$$f = 333.75b^6 + a^2(11a^2b^2 - b^6 - 121b^4 - 2) + 5.5\,b^8 + \frac{a}{2b}$$

mit $a = 77617.0$ and $b = 33096.0$. Die korrekte Lösung ist $f = -0.827396\cdots$ und nicht $f = 1.1726\cdots$. Es gibt Algorithmen, die Rundungsfehler kontrollieren; dabei wird das Ergebnis zwischen zwei Grenzen eingeschlossen.

Beispiel 1.5 Berechnung einer großen Zahl von Summanden.

Die Addition bzw. Subtraktion einer Vielzahl von Termen sollte mit Vorsicht behandelt werden. Man betrachtet zum Beispiel

$$\ln 2 = \lim_{m\to\infty} \sum_{n=1}^{m} \frac{(-1)^{n+1}}{n}\ .$$

Bricht man die Summe nach den ersten m Termen ab, so ist leicht zu sehen, daß sich dann ein Fehler $< 1/m$ ergibt. Wenn m sehr groß ist, wird das numerische Ergebnis dadurch beeinflußt, ob man die Summe mit $n = 1$ oder von hinten, mit $n = m$, beginnt. Es erweist sich allgemein als nützlich, die Berechnung so zu führen, daß kleine Zahlen zuerst addiert werden.

1.6 Programmorganisation

Das Schreiben guter Programme wird von vielen als eine Kunst betrachtet. Wenn eine Aufgabe sehr groß ist und von vielen Mitarbeitern bewältigt werden muß, kann man sich nicht auf Kunst allein verlassen. Für diese Probleme existieren Methoden, Strategien und Werkzeuge, die zum Einsatz kommen sollten, und die nicht in diesem Buch behandelt werden.

Für kleine Ein-Mann-Probleme gibt es eine Reihe einfacher Regeln, die das Leben erleichtern können; einige werden im folgenden beschrieben. Ein Programm sollte folgenden Anforderungen genügen:

- *Korrekt* – hierzu werden Testphasen benötigt (siehe Kapitel 1.7).
- *Zuverlässig* – Eingaben sollten auf Vollständigkeit, Plausibilität usw. untersucht und in der Ausgabe wiederholt werden. Warnungen und Fehlermeldungen sollten sinnvoll sein, zum Beispiel „Falscher Index: LL“ und *nicht* „Fehler XY25B“, oder auch nur „Fehler“!
- *Leicht lesbar* – Aussagekräftige Variablennamen sollten benutzt werden, keine einbuchstabigen Bezeichnungen, insbesondere *nicht* für globale Variable. Der Programmkopf sollte eine Beschreibung des Programms, seiner Ein- und Ausgaben, des Autors und des Datums der letzten Änderung enthalten. Kommentare sollten benutzt werden nach der Daumenregel: Im Mittel ein Kommentar auf fünf Anweisungen.
- *Wartungsfreundlich* – Strukturierte Programmierung sollte eingesetzt und keine numerischen Konstanten wie 125.8 im Code verwendet werden; Ausnahmen sind 0, 1.
- *Portierungsfreundlich* – Es sollten keine Programmiertricks verwendet werden, die nur mit bestimmten Compilern oder besonderer Computer-Hardware funktionieren.
- *Effizient* – Unbedachter Einsatz von Speicherplatz ist kostenträchtig und kann eine Berechnung stark verlangsamen, zum Beispiel durch ineffizienten Einsatz des Caches. Der Quelltext sollte nicht auf Kosten der Lesbarkeit optimiert werden, dies kann man getrost dem Compiler überlassen.
- *Einsatz von Subroutinen und Rekursion* – Man beachte, daß Subroutinen und Rekursion beim Aufruf einen Verwaltungsaufwand verursachen.

Dienstbibliotheken. Die meisten Rechner-Installationen umfassen Bibliotheken von Programmen zur Erledigung verschiedenster mathematischer oder algorithmischer Dienste, wie Sortieren, Zufallszahlenerzeugung, nicht-elementare mathematische Funktionen, Manipulation von Matrizen, Statistik usw. Diese Bibliotheks-tools sollten eingesetzt werden, wo immer dies möglich und sinnvoll ist. Was ist sinnvoll? Viele tools, die häufig allgemeiner gehalten sind als das bearbeitete Problem erfordert, können einen großen Verwaltungsaufwand, zum Beispiel durch Rufen von Subroutinen, und schwierige Anforderungen an das System verursachen und damit überhöhte Anforderungen an Rechenzeit und Speicherplatz. Zum Beispiel sollte man nicht zur Berechnung des Mittelwerts dreier Meßwerte des Luftdrucks ein großes Statistikpaket aufrufen.
Diese tools sind Programme und haben deshalb Fehler, wie im nächsten Abschnitt erklärt wird. Man benutze deshalb niemals ein tool, ohne es vorher getestet zu haben. Selbst wenn es fehlerfrei ist, könnte man seine Funktionsweise mißverstanden haben, oder es könnte ein Spezialfall vorliegen, für den das tool nicht gut geeignet ist, zum Beispiel wenn alle numerischen Eingabewerte nur aus Nullen bestehen. Ein weiteres Beispiel ist die Sortierung einer bereits sortierten Liste, was einige Sortierprogramme sehr ineffizient bewerkstelligen. *Daher sollte ein tool niemals ohne vorhergehenden Test eingesetzt werden.* Hier greifen die Prinzipien der Programmprüfung. Die Ausgabe des tools sollte mit ungültigen, extremen, ungewöhnlichen oder falschen Eingabewerten geprüft werden. Das tool sollte mit einer Anzahl von Testfällen konfrontiert werden, für die die Antworten bereits bekannt sind. Die vorliegende spezielle Anwendung mag ein Spezialfall sein, der nicht adäquat von dem tool behandelt wird. Ein Test mit einer entspre-

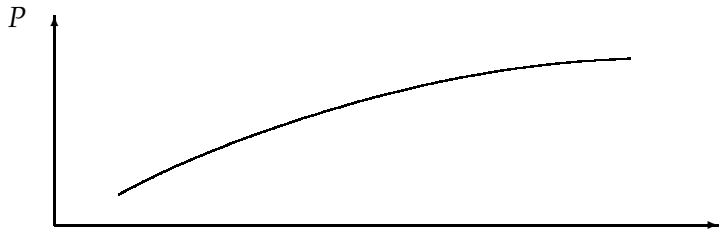


Abbildung 1.3: Abhängigkeit der Wahrscheinlichkeit P, weitere Fehler zu finden, von der Anzahl der bereits gefundenen Fehler.

chend vereinfachten und exakt lösbaren Version des Problems sollte durchgeführt werden.

Objekt-orientierte Programmierung: Dies ist eine sehr effektive Methode, bewährt vor allem beim Schreiben grosser Programme. Sie erfüllt dabei wichtige Bedingungen, wie einfache Möglichkeiten für Änderungen und Programmpflege, wohldefinierte Schnittstellen und Wiederverwendbarkeit von Routinen. Sie hat drei Grundprinzipien:

1. Encapsulation: Das Problem wird in Programm-Moduln (Objekte) mit wohldefinierter Funktionalität und wohldefinierter Schnittstelle strukturiert.
2. Nachrichten (messages): Informationsaustausch und Kontrolle der Objekte erfolgt durch Nachrichten.
3. Struktur (inheritance): Es gibt eine Hierarchie unter den Objekten mit der Möglichkeit, Daten und Funktionalität gemeinsam zu nutzen.

Objekt-orientierte Programmierung wird von Sprachen wie C++ oder Java unterstützt.

1.7 Programmprüfung

Jedes Programm enthält Fehler.

Das Prüfen von Programmen ist sehr wichtig und sollte einen angemessenen Bruchteil der Entwicklungszeit eines Programms in Anspruch nehmen. Dies führt zum Hauptsatz der Programmprüfung: **Ziel einer Programmprüfung ist es, Fehler zu finden**. Das Ziel ist *nicht*, zu *beweisen*, daß das Programm korrekt funktioniert [55].

Allgemeine Regeln

- Es gibt ein *blinder Fleck-Phänomen* gegenüber den eigenen Fehlern, daher ist die Hilfe Dritter bei der Prüfung eigener Programme wünschenswert.
- Die meisten Betriebssysteme umfassen Prüfwerkzeuge (debug tools). Diese sollten eingesetzt werden.
- Man versuche stets eine Vorhersage des Ausgangs einer Prüfung.
- Das Ergebnis jeder Prüfung sollte sorgfältig untersucht werden.
- Prüfungen und ihr Ergebnis sollten dokumentiert werden.
- Eine Routine, in der ein Fehler entdeckt wird, enthält wahrscheinlich noch weitere (siehe Abbildung 1.3).

Der Grund ist die (90-10)-Regel des täglichen Lebens, formuliert wie folgt: *90% der Fehler sind in 10% der Routinen*.

Die Programmprüfung erfolgt in den folgenden Schritten:

- Man prüft individuelle Module/Routinen.
- Man prüft das vollständige System. Manchmal ist es vorteilhaft, die ersten beiden Schritte zu faktorisieren, indem man das komplette System zuerst mit einem *Skelett* von vereinfachten Routinen testet, bevor man zusätzliche Komplikationen anhand der vollständigen Routinen einführt.
- Man prüft die Installation und Funktion in einem langen Testeinsatz.
- Man entscheidet, den Test zu beenden.

Beim Prüfen einzelner Routinen erweist es sich je nach der Güte der Prüf-Werkzeuge als nützlich, einige oder alle der folgenden Punkte zu überprüfen:

1. Sind alle Variablen definiert?
2. Sind die Indizes von Feldern innerhalb ihrer Grenzen?
3. Zeigt jeder Zeiger auf eine gültige Speicherzelle?
4. Stimmen Parameterlisten in aufrufenden/aufgerufenen Unterroutinen überein (Anzahl, Datentypen)?
5. Sind alle Variablen korrekt initialisiert?
6. Um 1 verzählt? (Notfalls Finger zum Zählen einsetzen).
7. Über-/Unterlauf in Zwischenergebnissen; kann ein Nenner $= 0$ werden?
8. Man überprüfe alle logischen Ausdrücke; sie sind besonders anfällig für Fehler.
9. Sind die Abbruchbedingungen für Schleifen und Programme korrekt?
10. Rechtschreibfehler, zum Beispiel 1 statt I oder l, O statt 0.
11. Unbeabsichtigte Benutzung globaler Variablennamen für verschiedene Variablen – man verwende keine kurzen, nichtssagenden Namen.
12. Man untersuche Eingabedaten auf Plausibilität, Vollständigkeit, Typ.
13. Man suche nach möglichen Nebenwirkungen.

Moderne Programmiersprachen wie C, C++ oder Java übernehmen einige dieser Überprüfungen automatisch.

Beispiel 1.6 Nebenwirkungen durch globale Variable.

b sei eine globale Variable.

```
function  f(x)
begin     b: = x + 1; f: = b;   end
function  g(x)
begin     b: = x + 2; g: = b;   end
```

Man nehme nun an, daß vor dem Aufruf der beiden Funktionen $b = 0$ ist; dann erhält man

$$\begin{array}{rcccc} \text{f(b)} + \text{g(b)} & = & 1 + 3 & = & 4 \\ \text{aber g(b)} + \text{f(b)} & = & 2 + 3 & = & 5 \end{array} .$$

Beispiel 1.7 Prüfen eines Programms.

Man nehme an, es sei ein Brief an alle Angestellten einer Firma zu versenden:

„Lieber X, Sie sind seit NN Jahren in dieser Firma, und Sie haben noch MM Jahre bis zu Ihrer Pensionierung. Mit freundlichen Grüßen, . . .“.

Lösung:
Das Programm liest drei Zahlen aus Dateien:
N_1 = Einstellungsjahr des Angestellten in der Firma, N_2 = gegenwärtiges Jahr,

$N_3 =$ Jahr der Pensionierung entsprechend den Firmenrichtlinien. Es berechnet NN $= N_2 - N_1$, MM $= N_3 - N_2$, setzt dies in den Brieftext ein und druckt.

Dieses Programm wäre wertlos, da es die Eingabedaten nicht auf Inkonsistenzen, Fehler und Spezialfälle untersucht. Um ein realistisches Programm zu prüfen, könnte man ihm die folgenden Beispielfälle vorsetzen:

- drei gültige Zahlen N_1, N_2, N_3
- eine oder mehrere der Zahlen sind keine Ganzzahlen
- eine oder mehrere der Zahlen sind keine gültigen Jahreszahlen oder überhaupt keine Zahlen, zum Beispiel 1789; 21; 2001; 01; ax
- NN oder MM sind negativ oder Null
- NN oder MM sind größer als eine Grenze, zum Beispiel 50
- eine oder mehrere der Zahlen fehlen in der Eingabe
- man mische verschiedene Konventionen für die Jahreszahl, zum Beispiel $N_1 = 1975$, $N_2 = 96$, $N_3 = 00$ (für 2000).

Literatur zur Programmprüfung: [55, 54, 11].

2 Algorithmen und Datenstrukturen

2.1 Algorithmen und ihre Analyse

Algorithmen. Eine Problemlösung, die als Rechnerprogramm formuliert werden kann, wird hier als *Algorithmus* bezeichnet. Große und schwierige Probleme werden dabei in Teilprobleme aufgeteilt, und auf dieser Ebene werden möglichst weitgehend Standardalgorithmen zur Problemlösung einbezogen. Die Auswahl optimaler Algorithmen spielt dabei naturgemäß eine große Rolle. Beispiele für Teilprobleme, die mit Standardalgorithmen gelöst werden können:

- Suche nach einem Namen in einer Liste;
- Sortieren einer Liste nach Schlüsselworten in aufsteigender Ordnung.

Solche Probleme können oft innerhalb größerer Aufgaben in der Datenauswertung auftreten. Ein Beispiel für eine solche Aufgabe ist die Bestimmung des *Medians* einer großen Stichprobe: Nach einem Sortierschritt auf die Daten ist die Bestimmung des Medians (der 50% Quantile) einfach.

Kriterien für die Güte von Algorithmen. Grundlegende Kriterien sind natürlich *Korrektheit* und *Effizienz.* Weiterhin sind aber auch *Klarheit* und *Einfachheit* wichtig, für numerische Algorithmen die *Genauigkeit* des Resultats. Die Anforderungen an *Rechenzeit* und *Speicherplatz* sind von großer praktischer Bedeutung. Manchmal muß man das Verhältnis zwischen Speicherplatz und Rechenzeit optimieren: Ein schnellerer Algorithmus erfordert unter Umständen mehr Speicherplatz und umgekehrt.

Klassifikation von Algorithmen nach ihrem Rechenzeitbedarf. Ein Algorithmus für eine bestimmte Problemlösung enthält in der Regel einen Parameter n, der die *Größe* des Problems beschreibt. Der Parameter n kann zum Beispiel die Anzahl der numerischen Daten oder allgemeiner der Datenelemente sein, welche der Algorithmus verarbeiten muß. Die Rechenzeit oder die Zahl der Rechenschritte wird in einer charakteristischen Weise von n abhängen, zum Beispiel

$$f(n) \propto n, \quad \log n, \quad n \log n, \quad n^2, \quad n^3, \quad 2^n \ ,$$

wie in Abbildung 2.1 zu sehen. Wichtig ist vor allem das Verhalten für große Werte von n. Um dieses Verhalten zu charakterisieren, wird die O-Notation benutzt. Man schreibt $f = O(g)$ (*f wächst nicht schneller als g*), wenn das Verhältnis $f(n)/g(n)$ für große Werte von n durch eine Konstante beschränkt ist. f und g sind von gleicher Größenordnung, falls $f = O(g)$ und $g = O(f)$ (*f und g wachsen gleich schnell*). Als Größenordnung einer Funktion $f(n)$ wird eine möglichst einfache Funktion von gleicher Größenordnung wie f bezeichnet, zum Beispiel ist $f(n) = n(n+1)/2$ von der Größenordnung n^2.

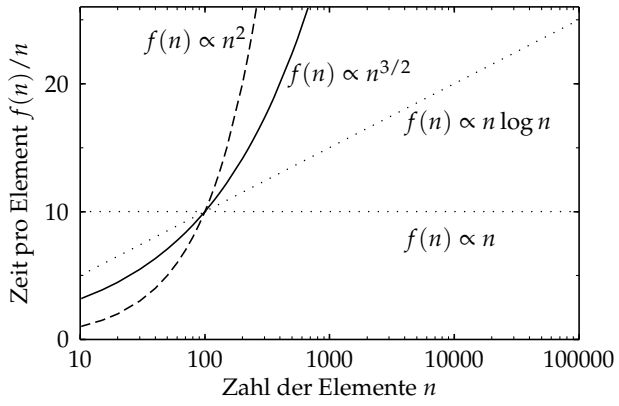


Abbildung 2.1: Zeit pro Element (in willkürlichen Einheiten) bei verschiedenen Abhängigkeiten von $f(n)$: $\propto n$, $\propto n \log n$, $\propto n^{3/2}$ und $\propto n^2$.

Anwachsen von Funktionen. Typische Abhängigkeiten sind:

$\mathbf{f(n) = 1}$ Die Rechenzeit ist konstant, unabhängig von n. Dies ist optimal.

$\mathbf{f(n) = n}$ Die Abhängigkeit der Rechenzeit von n ist linear, wenn für jede der n Eingabedaten ungefähr dieselbe Rechenzeit aufgewendet wird.

$\mathbf{f(n) = n^2}$ Solche Algorithmen taugen nur für Probleme mit relativ kleinem n. Diese Abhängigkeit entsteht typischerweise, falls eine doppelte Schleife über alle Paare von Elementen ausgeführt wird.

$\mathbf{f(n) = \log n}$ Ein Algorithmus mit *logarithmischer* Abhängigkeit wird mit wachsendem n kaum langsamer. Eine Verdoppelung von n erhöht die Rechenzeit jeweils um einen konstanten Betrag. Eine solche Abhängigkeit ist oft das Resultat einer (rekursiven) Aufteilung des Problems in kleinere (Beispiel binäre Suche).

$\mathbf{f(n) = n \log n}$ Diese Abhängigkeit entsteht, wenn die vorhergegangene Methode auf alle n Elemente angewandt wird. Sie ist typisch für gute Sortieralgorithmen.

$\mathbf{f(n) = 2^n}$ Eine solche Abhängigkeit kann als Resultat von Anwendung roher Gewalt entstehen. Sie ist hinsichtlich des Rechenzeitbedarfs sehr problematisch.

Eine logarithmische Abhängigkeit ist ein erstrebenswertes Ziel, wobei es auf die Basis des Logarithmus nicht ankommt. Oft tritt die Basis 2 auf; die folgende Notation wird für die Basis 2 (Zweierlogarithmus) bzw. e (natürlicher Logarithmus) benutzt:

$$\log_2 n = \lg n \qquad \log_e n = \ln n \; .$$

Oft wird die ganze Zahl $\lceil \lg n \rceil$ benötigt, für die gilt

$$n \leq 2^{\lceil \lg n \rceil} < 2n \; .$$

$\lceil x \rceil$ ist die kleinste ganze Zahl, die größer oder gleich x ist, und $\lfloor x \rfloor$ ist die größte ganze Zahl, die kleiner oder gleich x ist.

Beim Vergleich zweier Algorithmen für dieselbe Aufgabe sollte man bei der Zahl der Operationen auch den Faktor vor der n-Abhängigkeit betrachten; bei nicht zu großen Werten von n kann ein n^2-Algorithmus schneller als ein $n \log n$-Algorithmus sein.

Manchmal ist es zu einfach, nur einen einzigen Parameter n als Maß für den Aufwand zu betrachten. Zum Beispiel kann der Umfang der auszugebenden Daten durchaus ein zusätzliches Kriterium sein.

Durchschnittlicher und ungünstigster Fall. Die Zahl der Operationen eines Algorithmus kann neben der Größe n auch von der speziellen Art der Eingabedaten abhängen. Es ist dann für den Vergleich von Algorithmen wichtig, neben dem durchschnittlichen Fall (*average case*) auch den ungünstigsten Fall (*worst case*) zu betrachten. Das Verhalten in diesen Fällen wird beschrieben durch Funktionen $A(n)$ bzw. $W(n)$, welche die Zahl der Operationen bzw. die Rechenzeit angeben. Beispielsweise sind bestimmte Sortieralgorithmen besonders langsam, wenn die Eingabe in verkehrter Reihenfolge sortiert ist; ein bekannter Algorithmus ist sogar besonders langsam, wenn die Eingabe bereits richtig sortiert ist. Offensichtlich ist Vorsicht bei der Benutzung von Algorithmen mit ausgeprägtem Verhalten im ungünstigsten Fall geboten.

Ein anderer Aspekt ist das Verhalten des Algorithmus *im Mittel*, beschrieben durch $A(n)$. Dazu muß man Annahmen über die Wahrscheinlichkeit p_i des Auftretens bestimmter Eingabedaten vom Typ i machen. Man erhält

$$A(n) = \sum_i p_i \, T_i(n) \, ,$$

wobei $T_i(n)$ der Zeitaufwand für Fall i ist. Oft ist es ziemlich offensichtlich, wie die Wahrscheinlichkeiten p_i zu wählen sind.

Zwei Beispiele mögen das vorhergehende erläutern.

Wenn $\boldsymbol{A}$ und $\boldsymbol{B}$ reelle $n \times n$ Matrizen sind und das Matrixprodukt $\boldsymbol{C} = \boldsymbol{AB}$ berechnet werden soll, dann ist die Zahl der Multiplikationen bei einer üblichen Implementation in allen Fällen von der Ordnung n^3. Es ist also $A(n) = W(n) = n^3$ (mittlerer = ungünstigster Fall). Nun gibt es sehr komplexe Algorithmen für die Matrixmultiplikation, die lediglich $n^{2.81}$ Multiplikationen benötigen [58], sie erfordern jedoch, bedingt durch ihre Komplexität, einen zusätzlichen Aufwand und bieten in der Praxis meist keine Vorteile.

Für Sortieralgorithmen kann man die Zahl der Vergleichsoperationen zählen, die zum Sortieren von n Zahlen erforderlich sind. Für das Sortierverfahren durch Einfügen zum Beispiel sind es $n^2/2$ Vergleiche im ungünstigsten Fall (Eingabedaten verkehrt herum sortiert) und $n^2/4$ Vergleiche im Mittel (vollständig unsortierter Fall). Falls die Daten *fast schon sortiert* sind, nähert sich der Algorithmus einer linearen n-Abhängigkeit an. Der Algorithmus ist also gut für kleine Datenmengen, besonders wenn sie schon vorsortiert sind, und er wird langsam für große Datenmengen.

2.2 Datenstrukturen

2.2.1 Datenfelder

Eine natürliche und häufige Datenstruktur ist das ein- oder mehrdimensionale Datenfeld (array) konstanter Größe mit indizierten Elementen. Dabei wird eine Menge von Daten desselben Typs unter einem Namen, zum Beispiel A, zusammengefaßt und in aufeinanderfolgenden Plätzen im Speicher abgelegt. Der Zugriff auf bestimmte Elemente des Datenfeldes erfolgt über Indizes, zum Beispiel $A_{2,6,4}$. Datenfelder sind Grundbausteine für die Formulierung und Implementation vieler *numerischer Algorithmen.*

Ein eindimensionales Feld oder *Vektor* ist eine lineare Anordnung von Elementen, die durch *eine* ganze Zahl indiziert werden. Ein zweidimensionales Feld oder *Matrix* ist eine lineare Anordnung von Vektoren, jeder mit derselben festen Zahl von Elementen (zweidimensionale Felder sind also rechteckig). Zwei Indizes werden benötigt, um die Spalte und die Zeile der Matrix zu spezifizieren. Eine Verallgemeinerung ist das *n-dimensionale Datenfeld*, bei der jedes Element durch einen Satz von n ganzen Zahlen $j_1, j_2, \ldots j_n$ indiziert wird. Ein n-dimensionales Datenfeld ist eine lineare Anordnung von $(n-1)$-dimensionalen Datenfeldern.

Ein n-dimensionales Feld A wird deklariert durch $A(k_1, k_2, \ldots k_n)$, wobei die Indizes üblicherweise mit 0 (in C) oder 1 (in FORTRAN) beginnen und in Schritten von 1 bis k_j gehen. Man kann auch eine allgemeinere Zählung zulassen und die Indizes von i_j bis k_j laufen lassen. Die Feld-Deklaration wird dann folgendermaßen geschrieben:

$$A(i_1 : k_1, i_2 : k_2, \ldots i_n : k_n) \quad \text{mit} \quad i_j \leq \text{Index}_j \leq k_j \quad \text{für} \quad j = 1, 2, \ldots n \; .$$

Die Zahl der Elemente in der obenstehenden Deklaration von A ist

$$M = \prod_{j=1}^{n} (k_j - i_j + 1) \; .$$

Der Platz im Speicher wird reserviert durch einen Vektor B, deklariert durch $B(1 : M)$. Für die Abbildung der Elemente von A auf den Speicherbereich B gibt es zwei Konventionen: Die *inverse lexikalische Ordnung* (spaltenweise Speicherung) wird in Fortran benutzt. Das Element

$A_{j_1 j_2 \ldots j_n}$ wird gespeichert im q-ten Element von B, wobei $B(q) := A(j_1, j_2, \ldots j_n)$ mit

$$q = 1 + (j_1 - i_1) + \sum_{s=2}^{n} \left((j_s - i_s) \prod_{l=1}^{s-1} (k_l - i_l + 1) \right) \qquad \text{(spaltenweise)} .$$

Die *lexikalische Ordnung* (zeilenweise Speicherung) wird in Sprachen wie C benutzt. Die entsprechende Speicherplatzadresse ist

$$q = 1 + \sum_{s=1}^{n-1} \left((j_s - i_s) \cdot \prod_{l=s+1}^{n} (k_l - i_l + 1) \right) + (j_n - i_n) \qquad \text{(zeilenweise)} .$$

Diese Gleichungen zeigen, daß bei der Adressierung mehrdimensionaler Datenfelder ein gewisser Rechenaufwand für die Indexrechnung erforderlich ist. Zum Beispiel erfordert der Befehl `A(3,7,4,6) = 0.0` in Fortran sechs Multiplikationen und 20 Additionen/Subtraktionen. Stets treten Produktterme der Form

$$\prod_{l=s+1}^{n} (k_l - i_l + 1)$$

innerhalb der Adreßberechnung auf, und sie müssen mindestens einmal berechnet werden; manchmal werden sie abgespeichert, um künftige Adreßberechnungen zu beschleunigen.

Trotz dieses Aufwands ist der Zugriff auf die Elemente eines Datenfeldes im Prinzip sehr einfach wegen der *Linearität*: Die Speicheradresse hängt *linear* vom Index ab, zum Beispiel für die Elemente einer bestimmten Zeile oder die Diagonalelemente einer $n \times n$ Matrix (konstantes Inkrement von $n + 1$).

Man könnte denken, daß diese Speicherschemata bloße Konvention sind und ohne Bedeutung für die Effizienz. Dies stimmt jedoch nicht generell, wie das Beispiel einer doppelten Schleife über die 1000×1000 Elemente `A(I,J)` der zweidimensionalen Matrix `A` zeigt. Je nach der Reihenfolge der Schleifen über `I` bzw. `J` werden die Elemente entweder in aufeinanderfolgenden Speicherplätzen oder in den Plätzen 1, 1001, 2001 . . . 2, 1002, 2002 . . . adressiert. Falls diese Plätze im cache-Speicher gehalten werden, kann dies häufiges Umspeichern mit entsprechendem Zeitverlust bedeuten.

Strukturen. Die Elemente, die zu einem Datenfeld zusammengefaßt werden, müssen von gleichem Typ sein. In der Praxis braucht man auch Objekte, die aus verschiedenen Datentypen gebildet werden können. Deshalb gibt es zum Beispiel in C die *Struktur* als ein Objekt von Variablen unterschiedlichen Typs, die unter einem Namen zusammengefaßt werden. Datenfelder können als Elemente Strukturen enthalten.

2.2.2 Abstrakte Datentypen

Algorithmen greifen auf Datenstrukturen zu. Algorithmen und Datenstrukturen werden mit Vorteil durch Operationen auf die Daten beschrieben statt durch die Details der Implementation. Eine Datenstruktur wird dann als *abstrakter Datentyp* bezeichnet. Wichtige Datentypen sind Datenfelder (arrays), verkettete Listen, Stapel (stacks), Warteschlangen (queues), Graphen und Bäume.

Die Benutzung abstrakter Datentypen ist entscheidend wichtig für die Entwicklung großer Programme. Als Grundregel gilt, daß nichts außer der Definition der Datenstruktur und des Algorithmus, der auf die Daten zugreift, auf die innere Struktur Bezug nehmen darf. Dadurch erzielt man einfache Schnittstellen zwischen den Algorithmen, den Datenstrukturen und den Anwenderprogrammen.

Unterteilungen von Datenmengen. Große Datenmengen müssen in kleinere Einheiten unterteilt werden, damit sie handhabbar bleiben. Als Analogie kann man sich ein Buch vorstellen, welches in Kapitel, die Kapitel in Abschnitte und diese in Paragraphen und weiter in Sätze unterteilt sind.

Unter den vielen Möglichkeiten der Unterteilung von Datenmengen wird hier die folgende vorgestellt: Die strukturierte Datenmenge heißt *Datei* (file). Sie ist unterteilt in *Datensätze* (records). Jeder Datensatz kann in *Datenfelder* (fields) unterteilt sein. Man kann noch feiner unterteilen: Man kann auf die Ebene einzelner Zeichen (characters) und dann bits gehen. Eine mögliche hierarchische Einteilung ist in der nebenstehenden Skizze gezeigt.

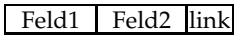
Datei (file)
Block
Datensatz (record)
Segment
Feld
Zeichen (character)
bit

Der Name Datei wird oft für Daten auf externen Speichermedien reserviert. Andere Namen für Datenmengen sind *Datenblöcke* und *Segmente*.

Zeiger (pointer). Sie dienen der Adressierung in Datenstrukturen. Zeiger stellen in C einen elementaren Datentyp dar. Im Gegensatz zur *Adresse* sind sie variabel, sie lassen sich neu initialisieren, und man kann mit Zeigern rechnen. Ein Zeiger kann mit einer beliebigen Adresse geladen werden. Der Adreßwert 0 wird meist als Anfangswert geladen und zeigt, daß der Zeiger (noch) nicht korrekt gesetzt wurde. Der Wert -1 wird zur Anzeige einer ungültigen Adresse benutzt.

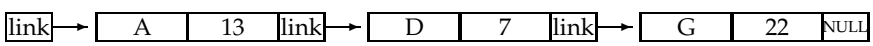
Datenfelder mit fester Länge werden fest vereinbart und erlauben es nicht, flexibel auf Speicherbedarf zu reagieren, der sich erst bei der Ausführung eines Programms ergibt. Eine *dynamische Speicherverwaltung* dagegen erlaubt es, den Platz für Objekte erst unmittelbar vor ihrer Verwendung zur Verfügung zu stellen. In C und in anderen Programmiersprachen gibt es Funktionen zur Anforderung von Speicherplatz aus dem vom Betriebssystem verwalteten Systemspeicher. Man kann auch ein großes fest vereinbartes Datenfeld mit Hilfe von Indizes als Zeigern dynamisch unterteilen.

Gekettete Listen. Der Aufbau von geketteten Listen (linked lists) erlaubt den flexiblen Umgang mit Datenmengen; eine Liste wird aus einer beliebig veränderbaren Anzahl von Strukturen gebildet. Mit jedem Element der Liste ist ein Zeiger vereinbart, der auf das folgende Element der Liste mit gleichem Aufbau weist. Dadurch läßt sich eine beliebig lange Kette bilden. Die einzelnen Glieder einer Kette bezeichnet man als Knoten (node). Ein Knoten mit zwei Feldern, welche die Daten enthalten und einem Zeiger (link) sieht wie folgt aus:



In der einfach-geketteten Liste (single linked list) enthalten die Knoten einen einzigen Zeiger, der jeweils auf den nachfolgenden Knoten verweist. Zusätzlich gibt es einen Zeiger, der auf den ersten Knoten der Liste verweist.

Eine verkettete Liste mit drei Knoten ist zum Beispiel



Standard-Operationen sind das Anfügen eines weiteren Knotens am Anfang, am Ende oder an einer anderen Position, und das Entfernen von Knoten; diese Operationen erfordern nur das Verändern von Zeigern. Der Vorteil verketteter Listen verglichen mit fest vereinbarten Datenfeldern besteht in der größeren Flexibilität, zum Beispiel falls die Elemente verschiedene Größe haben, oder falls Elemente eingefügt, entfernt oder modifiziert werden sollen. Der Nachteil ist der Zeitaufwand, um ein bestimmtes Element anzusprechen.

Ein *Knoten* enthält spezifische Informationen, wie zum Beispiel einen *Namen*, und einen Zeiger zum nächsten Knoten (link). In Diagrammen von Datenstrukturen werden Zeiger als Pfeile dargestellt. Der Wert des Zeigers ist ein Index bzw. die Adresse eines Knotens. Der Zeiger des letzten Knotens muß das Ende der Struktur anzeigen. Der *null*-Zeiger, bezeichnet mit NULL, zeigt an, daß *kein* Knoten folgt. Um die Liste zu beginnen, benötigt man einen speziellen Knoten, welcher *Anker* (head) heißt, und der auf den ersten Knoten der Liste zeigt.

Doppelt-gekettete Listen. In der doppelt-geketteten Liste gibt es neben dem Zeiger auf den Nachfolger *(next)* auch einen weiteren Zeiger auf den Vorgänger *(prev)*. Dies kann das Aufsuchen von Knoten sehr beschleunigen. Entsprechend gibt es neben dem Zeiger auf den ersten Knoten der Liste einen weiteren Zeiger auf den *letzten* Knoten der Liste. In einer Variante zeigt der *next*-Zeiger des letzten Knotens wieder auf den ersten Knoten und der *prev*-Zeiger des ersten Knotens wieder auf den letzten Knoten. Dadurch entsteht ein geschlossener Ring, der spezielle Probleme einfach gestaltet.

Elementare Operationen mit Listen sind:

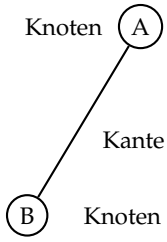
1. ein neues Element einfügen,
2. ein Element entfernen,
3. ein Element mit einem bestimmten Schlüssel suchen,
4. ein Element suchen und modifizieren,
5. eine Liste in zwei verkettete Listen aufteilen,
6. eine Liste sortieren.

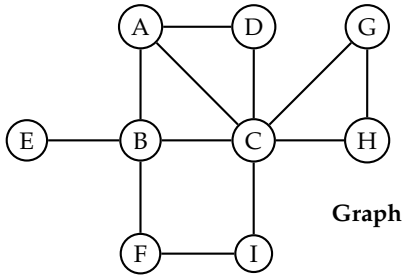
Wenn ein Sortierschlüssels in den Knoten vorliegt, kann die Liste sortiert werden; dies erleichtert Suchoperationen und das Einfügen neuer Knoten. Ein Umordnen der Elemente ist einfach: Drei Zeiger müssen geändert werden. Im Falle eines indizierten Datenfeldes müßten dagegen viele Elemente umgespeichert werden.

Stapel und Warteschlangen. Spezielle Listen erfordern nur eine begrenzte Zahl möglicher Operationen. Bei dem *Stapel* (stack) gibt es zwei Operationen: Ein neues Element wird durch Einsetzen am Anfang hinzugefügt (push), und ein Element wird am Anfang entfernt (pop). Man nennt einen Stapel auch einen LIFO-Speicher (last in, first out). Stapel sind einfach durch verkettete Listen realisierbar. Bestimmte Computersprachen benutzen Stapel für Berechnungen; jede Operation erhält ihre Argumente aus dem Stapel, und das Resultat wird wieder auf dem Stapel abgelegt.

Auch in der *Warteschlange* (queue) genannten Liste gibt es zwei Operationen: Elemente werden am *Anfang* eingesetzt und vom *Ende* entfernt. Bei dieser Liste, auch FIFO-Speicher genannt (first in, first out), werden die Elemente in der Eingangsreihenfolge abgearbeitet.

Graphen. Ein *Graph* besteht aus einer Menge von *Knoten* und aus einer Menge von *Kanten* (Zweigen), die jeweils zwei Knoten verbinden. Geometrisch kann ein Graph durch Punkte oder Kreise veranschaulicht werden, die durch Linien verknüpft werden. Durch Graphen kann in einer Datenorganisation die Beziehung zwischen Objekten beschrieben werden. Die Objekte oder Knoten können Namen tragen und enthalten Daten. Graphen heißen *gerichtet*, wenn die Kanten aus Pfeilen bestehen, die eine Richtung angeben.



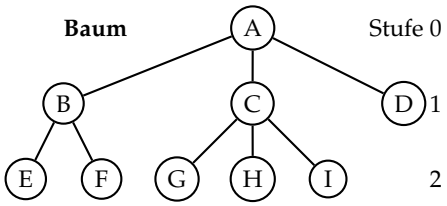


Eine *Kette* (oder Pfad) ist ein geschlossener Kantenzug oder eine ununterbrochene Folge von Kanten, wobei keine Kante mehr als einmal durchlaufen wird. Ein *Zyklus* ist eine Pfad von einem Knoten auf sich selbst zurück. Ein *zusammenhängender* Graph liegt vor, wenn sich je zwei verschiedene Knoten durch mindestens eine Kette miteinander verbinden lassen.

Bäume. Ein *Baum* (tree) ist ein Graph, bei dem für je zwei seiner Knoten genau eine Kette existiert, die diese beiden Knoten miteinander verbindet (kein Zyklus). Wird zu einem Baum eine weitere Kante hinzugefügt, entsteht notwendigerweise ein Zyklus. Der *spanning tree* eines Graphen ist der Subgraph, der alle Knoten enthält, jedoch von den Kanten nur soviele, wie notwendig sind, um einen Baum zu bilden (siehe auch Seite 34).

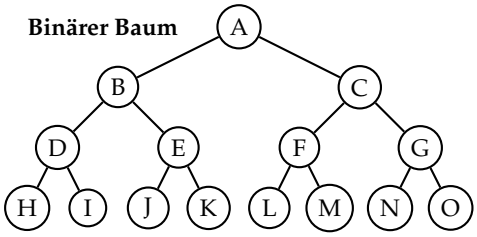
Die nachfolgende Abbildung zeigt einen Baum. Ein ausgezeichneter Knoten ist die *Wurzel* A auf der Stufe 0; es gibt genau eine Kette zwischen der Wurzel und jedem der Knoten.

Umgekehrt wie in der Natur ist die Wurzel oben. Direkt mit der Wurzel verbundene Knoten sind auf der Stufe 1 usw. Knoten ohne Zweige zur nächsten Stufe heißen *Endknoten* oder *Blätter*.



Wird die Wurzel eines Baums entfernt, so zerfällt der Baum in unzusammenhängende Unterbäume, die die gleiche Eigenschaft wie Bäume haben. Mehrere nichtzusammenhängende Bäume oder Unterbäume werden auch *Wald* genannt.

Bei einem *binären Baum* hat jeder Knoten höchstens zwei Unterbäume. Beispiel für einen binären Baum ist der Stammbaum des Menschen, bei dem jeder Mensch, als Knoten dargestellt, ein Elternpaar hat. In einem *geordneten Baum* gibt es auf jeder Stufe eine festgelegte Ordnung zwischen den Knoten. Beispiele für geordnete binäre Bäume sind die *heap*-Struktur (Seite 27) und der Suchbaum (Seite 30).



2.3 Sortieren

Es mögen n Elemente gegeben sein, wobei jedes Element ein einzelnes Wort, ein Vektor oder ein Datensatz (record) sein kann, und wo jedes Element einen *Sortierschlüssel* t_i, $i = 1, 2, \ldots n$ hat. Das Sortierproblem besteht darin, die Elemente in eine bestimmte Ordnung zu bringen, so daß zwischen jedem Paar t_i, t_{i+1} eine bestimmte Beziehung besteht. Der Normalfall ist Sortieren in aufsteigender Reihenfolge der Sortierschlüssel:

$$t_i \leq t_{i+1} \qquad i = 1, 2, \ldots (n-1) \qquad \text{aufsteigende Ordnung .}$$

Dieser Fall wird im folgenden betrachtet. Außerdem wird nur das Sortieren von Daten betrachtet, die sich im Speicher (also nicht auf externen Speichermedien) befinden.

Die Grundoperationen sind das *Vergleichen* zweier Sortierschlüssel und das *Vertauschen* von Elementen. Falls die Elemente lange Datensätze sind, ist die letztere Operation sehr aufwendig. In diesem Fall empfiehlt sich eine Datenstruktur mit Zeigern, so daß man nur die Zeiger umsetzen muß. Der Aufwand eines Sortieralgorithmus bemißt sich nach der Zahl der Vergleichs- und Vertauschoperationen $A(n)$ im Normalfall und der entsprechenden Zahl $W(n)$ im ungünstigsten Fall. Ein wichtiges Kriterium ist der zusätzliche Speicherbedarf; einige Algorithmen kommen ohne aus. Ein weiteres Kriterium ist Stabilität: Ein Sortieralgorithmus heißt *stabil*, wenn die Reihenfolge identischer Sortierschlüssel erhalten bleibt.

Es werden zunächst einige Sortieralgorithmen vorgestellt für den einfachen Fall einer Menge reeller Zahlen t_i, die geordnet werden soll.

Sortieren durch Auswahl. Der Algorithmus ist naheliegend: Man sucht das kleinste Element und tauscht es mit dem Element auf Platz eins, dann das zweitkleinste und tauscht es mit dem Element auf Platz zwei usw.

Eine Analyse zeigt, daß die Zahl der Vergleiche $A(n)$ mit der Zahl n der zu sortierenden Elemente ansteigt wie $A(n) = W(n) = (n-1) + (n-2) \ldots + 2 + 1 = n(n-1)/2 \approx n^2/2$, die Abhängigkeit von n ist also quadratisch. Für große Werte von n ($n > 10$) wird diese Methode sehr zeitaufwendig, falls das Sortierprogramm sehr häufig aufgerufen wird, und man sollte dann ein anderes benutzen. Eine positive Eigenschaft ist, daß der ungünstigste Fall und der mittlere Fall etwa denselben Aufwand erfordern. Auch ist die Zahl der Austauschoperationen maximal gleich n; die Abhängigkeit von n ist damit linear, während sie sonst meist quadratisch ist.

Sortieren durch Einfügen. Diese Methode ist einfach und deshalb wichtig, weil sie Teil eines schnelleren Algorithmus ist (Shell's Sortiermethode). Die Idee ist die folgende: Angenommen, die ersten Elemente seien schon sortiert. Dann besteht der nächste Schritt darin, die Länge des sortierten Stücks zu vergrößern, indem man das nächste Element am richtigen Platz einsetzt. Das folgende Programmstück tut dies.

```
      DO I=2,N              ! sort by insert
        TT=T(I)             ! new element
        DO J=I-1,1,-1       ! compare in reverse order
          IF(T(J).LE.TT) GOTO 10
          T(J+1)=T(J)       ! move one place up
        END DO
        J=0                 ! new element is smallest so far
10      T(J+1)=T(I)         ! insert new element at position J+1
      END DO
```

Falls `TT=T(I)` das nächste einzusetzende Element ist, müssen alle bereits sortierten Elemente um einen Platz hochgeschoben werden, bis ein kleinerer oder gleich großer Sortierschlüssel gefunden ist. Dann wird das Element mit `TT` in die Lücke eingesetzt. Diese Methode wird auch von gewieften Kartenspielern verwendet.

Als Beispiel wird der Algorithmus auf eine Menge von 10 Buchstaben angewandt. Gezeigt ist in einer Tabelle die Anfangsordnung und die Anordnung nach jeder Schleife, die den Austausch von Elementen erforderte; vertauschte Buchstaben sind fett gedruckt.

Die Zahl der Austauschoperationen erscheint ziemlich groß; dies liegt daran, daß die Elemente jeweils nur um einen Platz bewegt werden, zum Beispiel wird der Buchstabe E viermal um eine Position bewegt.

Index	Schlüssel	1	2	3	4	5	6
1	A	A	A	A	A	A	A
2	G	**E**	E	**D**	**B**	B	B
3	E	**G**	**F**	**E**	**D**	**C**	C
4	F		**G**	**F**	**E**	**D**	D
5	H		H	**G**	**F**	**E**	E
6	D			**H**	**G**	**F**	F
7	B				**H**	**G**	G
8	J				J	**H**	H
9	C					**J**	**I**
10	I						**J**

Bei der Analyse des Algorithmus zählt man die Vergleiche. Für jeden Index i gibt es maximal $(i-1)$ Vergleiche. Summiert über alle i gibt dies eine obere Grenze $W(N)$ für den ungünstigsten Fall. Falls die Daten verkehrt herum sortiert sind, hat man in der Tat diesen ungünstigsten Fall

$$W(n) = \sum_{i=2}^{n}(i-1) = \frac{n(n-1)}{2} \; .$$

Dieses Resultat ist dasselbe wie das mittlere Verhalten beim Auswahlsortieren. Die Zahl der Vergleiche im durchschnittlichen Fall ist jedoch nur $A(n) \approx n^2/4$, was nicht so leicht zu sehen ist. Angenommen, alle Permutationen der Sortierschlüssel sind gleich wahrscheinlich, dann ist die Zahl der Vergleiche für jeden Wert des Index i

$$\sum_{j=1}^{i-1} \frac{1}{i} j + \frac{1}{i}(i-1) = \frac{i+1}{2} - \frac{1}{i} \; .$$

Addiert man alles auf, so erhält man

$$A(n) = \sum_{i=2}^{n} \left(\frac{i+1}{2} - \frac{1}{i} \right) = \frac{n^2}{4} + \frac{3n}{4} - 1 - \sum_{i=2}^{n} \frac{1}{i} \; .$$

Für große Werte von n erhält man $A(n) \approx n^2/4$, einen nur halb so großen Wert wie für das Sortieren durch Auswahl. Meist werden Elemente nur um einen Platz bewegt. Dies kann, wie unten gezeigt, durch Shell's Sortieralgorithmus verbessert werden.

Shell's Sortieralgorithmus. Dieser Algorithmus, erfunden von D. Shell [72], ist nur wenig komplizierter als Sortieren durch Einfügen, und er ist schnell. In dem Verfahren wird ein Inkrement m festgelegt. Zu jedem Inkrement m gibt es m Folgen, die durch die Elemente mit Index-Inkrement m gebildet werden, und die zum Beispiel durch Einfügen sortiert werden. Man erhält damit m unabhängig sortierte Folgen, die verflochten sind. Vollständiges Sortieren wird durch eine Folge von Inkrementen erreicht mit dem letzten Inkrement $m = 1$.

Das Beispiel in der Tabelle zeigt das Sortieren von 10 Buchstaben, wobei nacheinander die Inkremente $m = 4$ und $m = 1$ benutzt werden. Bei $m = 4$ gibt es vier separate Gruppen von Buchstaben (zwei mit 3 und zwei mit 2 Elementen); nur drei Austauschoperationen sind in diesem Schritt erforderlich. Die weiteren Spalten zeigen das Sortieren mit Inkrement $m = 1$; die Zahl der Vergleichs- und Austauschoperationen ist viel kleiner als beim Sortieren durch Einfügen.

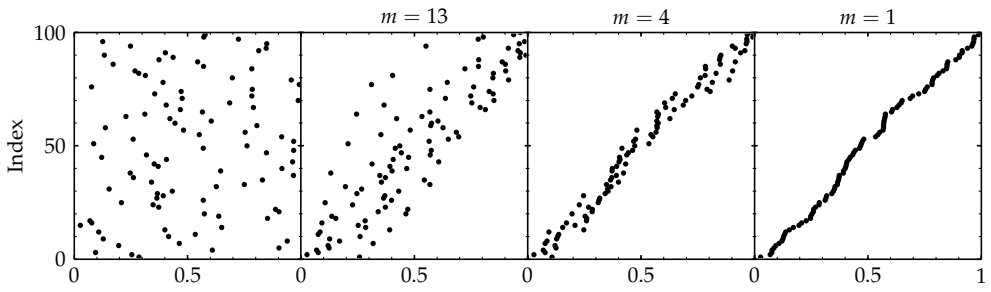


Abbildung 2.2: Sortieren von 100 Zufallszahlen zwischen 0 und 1.0 (Abszisse) mit dem Shell-Algorithmus. Von links nach rechts die ursprüngliche Reihenfolge und die Reihenfolge nach dem Sortieren mit dem Inkrement $m = 13$, $m = 4$ und $m = 1$.

Index	Schlüssel	(1)	(2)	(3)	(4)	$m = 4$	1	2	3	4	5
1	A	A				A	A	A	A	A	A
2	G		G			D	**B**	B	B	B	B
3	E			E		B	**D**	**C**	C	C	C
4	F				F	F	F	**D**	D	D	D
5	H	H				C		**F**	**E**	E	E
6	D		D			G		G	**F**	F	F
7	B			B		E			**G**	G	G
8	J				J	J			J	**H**	H
9	C	C				H				**J**	**I**
10	I		I			I					**J**

Die Zahl der Vergleiche in diesem Algorithmus hängt von der Folge der Inkremente m ab. Die genaue Analyse des Algorithmus ist schwierig, und die optimale Wahl ist nicht bekannt, jedoch sind einige Fälle gut studiert. Empirisch findet man, daß die Folge $m = \ldots, 1093, 364, 121, 40, 13, 4, 1$ gute Resultate gibt. Diese Inkremente folgen, beginnend mit $m_1 = 1$, aus der Relation

$$m_{s+1} = 3m_s + 1 \; .$$

Man begint das Sortieren mit $s = t$, wobei t die kleinste Zahl mit $m_{t+2} \geq n$ ist. Die nächsten Inkremente findet man nach der Formel $m_s = (m_{s+1} - 1)/3$. Die Abbildung 2.2 zeigt die verschiedenen Phasen beim Sortieren von 100 Zufallszahlen.

Die theoretischen Studien von Shell's Algorithmus sind zwar nicht abgeschlossen, jedoch kann man zeigen, daß nie mehr als $n^{3/2}$ Vergleiche nötig sind. Dies ist asymptotisch schlechter als optimale Strategien, welche wie $n \lg n$ gehen. Ein Test mit Zufallszahlen zeigt jedoch, daß selbst für $n = 10^4$ Shell's Methode kaum langsamer ist als $n \lg n$-Verfahren.

Die untenstehende Tabelle vergleicht verschiedene Algorithmen. Sie enthält auch das sogenannte bubble sort. Bei dieser Methode geht man durch alle Elemente durch; benachbarte Elemente werden vertauscht, falls notwendig, und dies wird wiederholt, bis bei einem Durchgang keine Vertauschoperation mehr nötig ist. Wie man aus der Tabelle sieht, ist dieses Verfahren vergleichsweise ungünstig.

	Zahl der Vergleiche	Zahl der Vertauschungen
Sortieren durch Auswahl	$n^2/2$	n
Sortieren duch Einfügen	$n^2/4$	$n^2/8$
bubble sort	$n^2/2$	$n^2/2$
Shell's Algorithmus	$\approx n^{3/2}$	-

Sortieren großer Datensätze. Bisher wurde nur der Fall eines Vektors von n Zahlen oder Sortierschlüsseln t_i, $i = 1, 2, \ldots n$ diskutiert. In der Praxis hat man dagegen oft n Datensätze A_i, $i = 1, 2, \ldots n$ zu sortieren, wobei jeder Datensatz A_i einen Sortierschlüssel t_i enthält. Vertauschoperationen für die Datensätze sind im allgemeinen sehr aufwendig. Falls die Tabelle $A(1:N)$ aus großen Datensätzen besteht, ist es daher sinnvoll, einen Indexvektor einzuführen, welcher auf die Sortierschlüssel t_i der Datensätze zeigt. Der Indexvektor hat die Elemente I_j, $j = 1, 2, \ldots n$. Die Algorithmen werden dann so abgeändert, daß sie auf den Indexvektor zugreifen und Elemente des Indexvektors statt ganzer Datensätze vertauschen. Das Resultat ist ein Algorithmus, der den Indexvektor sortiert, so daß der kleinste Schlüssel t_i den Index I_1, der nächste Schlüssel den Index I_2 hat usw.

Das im Anhang aufgeführte Fortran-Unterprogramm `ISORTS` ist eine auf diese Weise modifizierte Implementation von Shell's Sortieralgorithmus. Die Prozedur `ISORTS` kann auch benutzt werden, um einen Vektor `A(1:N)` von einfachen Zahlen zu sortieren. In diesem Fall setzt man `IDX(I)=I` für alle `I`. Für spezielle Anwendungen (reelle Zahlen statt ganzer Zahlen, mehrfacher Schlüssel) kann das Unterprogram leicht abgeändert werden.

Vertausch-Strategie. Oft reicht es, den Indexvektor zu sortieren und die Datensätze nicht zu bewegen, zum Beispiel kann das Ausdrucken in sortierter Reihenfolge direkt, gesteuert durch den Indexvektor, geschehen. Manchmal muß man aber doch die Datensätze selbst in die richtige Reihenfolge bringen.

Wenn die Länge aller Datensätze gleich ist, kann dies durch Vertauschen erfolgen. Es erfordert einen Hilfsspeicher (aux) von der Länge eines Datensatzes, ist mit Hilfe des sortierten Indexvektors möglich und erfordert etwa n Vertauschoperationen. Dies wird in dem untenstehenden Pseudo-Programm gezeigt. Das Kopieren des Datensatzes von Position `K` nach Position `J` wird durch ein Unterprogramm `MOVETO` ausgeführt. Dem Hilfsspeicher wird dabei die Position 0 zugewiesen. Eine Schleife über alle Indizes `I` wird ausgeführt. Falls `IDX(I)=I`, ist der Datensatz schon am richtigen Platz. Sonst wird der Datensatz `A(I)` auf Platz `I` im dem Hilfsspeicher gespeichert. Dann ist die Position `I` frei und der Datensatz `A(IDX(I))` kann dort gespeichert werden. Damit wird der Platz von `IDX(I)` frei, usw.

```
      DO I=1,N              ! loop to move into sorting position
        IF(IDX(I).NE.I) THEN
          CALL MOVETO(I,0)      ! move from position I to aux
          K=I
10        J=K
          K=IDX(J)
          IDX(J)=J
          IF(K.NE.I) THEN
            CALL MOVETO(K,J)    ! move from position K to position J
            GOTO 10
          END IF
          CALL MOVETO(0,J)      ! move from aux to position J
        END IF
      END DO
```

Rangtabelle. Der *Rang* ist die Position eines Elements *nach* dem Sortieren. Der i-te Eintrag in der Rangtabelle gibt den Rang des i-ten Elements in der ursprünglichen Anordnung, und die Tabelle geht von 1 bis N. Eine Rangtabelle `IRANK(1:N)` kann aus der Indextabelle `IDX(1:N)` konstruiert werden, wie das untenstehende Programmstück zeigt.

```
      INTEGER IDX(1:N), IRANK(1:N)
      DO I=1,N
```

```
  IRANK(IDX(I)) = I
END DO
```

Die nebenstehende Tabelle zeigt in einem einfachen Beispiel die ursprüngliche und die sortierte Tabelle der Sortierschlüssel, zusammen mit dem Indexvektor und der Rangtabelle.

	1	2	3	4	5	6
Schlüssel, unsortiert	30	16	90	9	2	77
Indextabelle	5	4	2	1	6	3
Rangtabelle	4	3	6	2	1	5
Schlüssel, sortiert	2	9	16	30	77	90

Quicksort. Die besten Sortieralgorithmen erfordern einige $n \lg n$ Grundoperationen. Gebräuchlich sind zwei Algorithmen mit kleinen Konstanten vor der $n \log n$-Abhängigkeit.

Ein sehr guter Sortieralgorithmus für große Werte von n ist Quicksort [32], von C.A.R. Hoare 1962 erfunden, der sich als rekursives Verfahren definieren läßt. Aus der Menge der Sortierschlüssel wird ein Element ausgewählt, genannt Pivotelement; die anderen Elemente werden in zwei Untermengen aufgeteilt: Eine Menge mit Schlüsseln kleiner als das Pivotelement, und eine Menge mit Schlüsseln größer oder gleich dem Pivotelement. Dieses Verfahren wird dann jeweils auf die beiden Untermengen angewandt. Eine Untermenge von weniger als zwei Elementen wird nicht mehr unterteilt.

Das Standard-Buch *Programmieren mit C* von Kernighan und Ritchie [41] enthält eine einfache Version von Quicksort, die ein gutes Beispiel für die Verwendung der Rekursion ist. Die dreimal vorkommende Tauschoperation ist in einer eigenen Funktion `swap` definiert. Die Standard-C-Bibliothek enthält eine Version von `qsort`, die Objekte von beliebigem Typ sortieren kann.

```
/* qsort: sort v[left]...v[right] in ascending order */
void qsort(int v[], int left, int right)
{
    int i, last;
    void swap(int v[], int i, int j);
    if(left >= right) /* nothing to do if the vector */
        return; /* has less than two elements */
    swap(v, left, (left + right)/2); /* move selected element */
    last = left;                     /* to v[0] */
    for (i = left+1; i <= right; i++)  /* split */
        if (v[i] < v[left])
            swap(v, ++last, i);
    swap(v, left, last);          /* get back selected element */
    qsort(v, left, last-1);
    qsort(v, last+1, right);
}

/* swap: exchange v[i] and v[j] */
void swap(int v[], int i, int j)
{
    int temp;
    temp = v[i];
    v[i] = v[j];
    v[j] = temp;
}
```

In dieser einfachen Implementation wird als Pivotelement jeweils das *mittlere* Element benutzt. Das Verfahren funktioniert optimal, wenn die Teilmengen jeweils etwa gleich groß sind, und dann ist die durchschnittliche Zahl der Vergleiche $2n \ln n$. Die Abbildung 2.3 veranschaulicht

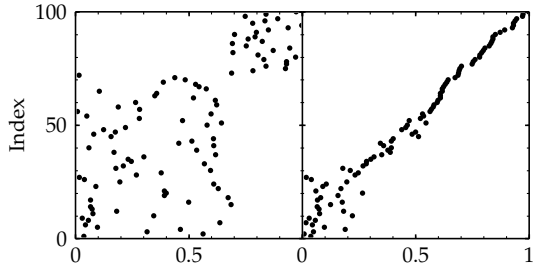


Abbildung 2.3: Sortieren von 100 Zufallszahlen mit dem Quicksort-Algorithmus, links nach der ersten Teilung und rechts in einem teilsortierten Zwischenzustand.

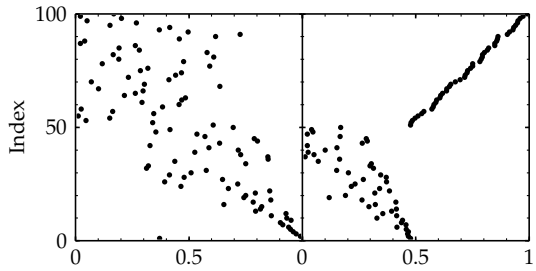


Abbildung 2.4: Sortieren von 100 Zufallszahlen mit dem Heapsort-Algorithmus, links der Zustand nach Definition des Heaps und rechts in einem Zwischenzustand, in dem ein Teil bereits sortiert ist.

den Sortiervorgang. Quicksort braucht einen Hilfsspeicher: Ein Stapel (stack) der durchschnittlichen Länge $2 \lg n$ wird für die rekursiven Aufrufe benötigt. Obwohl die Formulierung mit Rekursion sehr elegant aussieht, ist es in der Praxis effizienter, den Stapelspeicher direkt zu programmieren. Viele Varianten existieren, zum Beispiel kann es vorteilhaft sein, für kurze ($n \leq 7$) Untermengen eine der einfachen Sortiermethoden anzuwenden.

Ein Problem ist, daß die Rechenzeit im ungünstigsten Fall sehr lang sein kann. In diesem Fall ist Quicksort ein n^2-Algorithmus, und benötigt außerdem einen sehr großen Hilfsspeicher der Größe n. Für bestimmte Implementationen gehört eine bereits sortierte Tabelle zum ungünstigsten Fall.

Es gibt daher Implementationen, die diese ungünstigen Fälle vermeiden sollen. Man nennt den Quicksort-Algorithmus *fragil*; eine sorgfältig für ein bestimmtes Problem ausgewählte Version ist vermutlich die beste Lösung; in anderen Fällen können jedoch Probleme auftreten. Eine einfache Implementation des Algorithmus (in Fortran) findet sich im Anhang; der Aufruf ist identisch mit dem Aufruf bei dem Unterprogramm `ISORTS`.

Heapsort. Der Heapsort-Algorithmus, erfunden von J. Williams [82], hat ebenfalls ein $n \lg n$-Verhalten. Selbst im ungünstigsten Fall ist er nur 20% langsamer als im Mittel. Außerdem benötigt er keinen Zwischenspeicher.

Der Heapsort-Algorithmus beruht auf einer besonderen, *heap (Haufen)* genannten Datenstruktur. Dies ist eine Menge von n Zahlen t_i, $i = 1, 2, \ldots n$, die der folgenden Beziehung gehorcht:

$$t_{\lfloor j/2 \rfloor} \geq t_j \quad \text{für} \quad 1 \leq \lfloor j/2 \rfloor < j \leq n \tag{2.1}$$

(bei der Division $\lfloor j/2 \rfloor$ wird $j/2$ zur nächsten ganzen Zahl abgerundet). Das Element t_1 hat dadurch stets den maximalen Wert.

Wie das nachfolgende Diagramm für den Fall von zehn Knoten $t_1 \ldots t_{10}$ zeigt, ist ein *heap* der Spezialfall eines binären Baums. Die Wurzel ist Element t_1, die zwei nächsten Knoten sind t_2 und t_3, die vier folgenden Knoten sind $t_4 \ldots t_7$, usw.

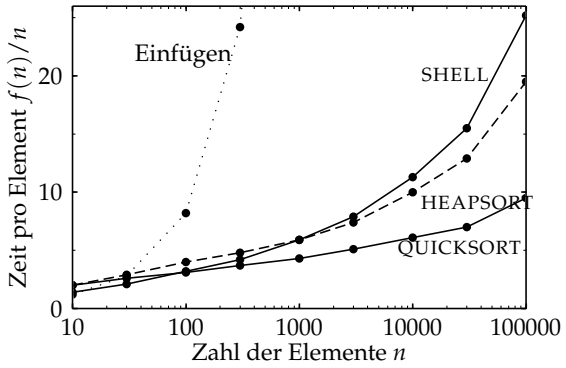
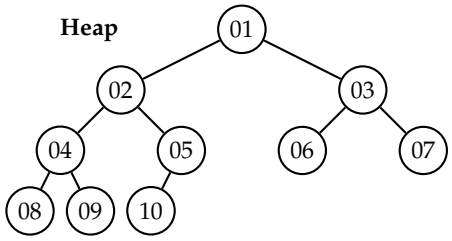


Abbildung 2.5: Rechenzeit pro Element (in willkürlichen Einheiten) beim Sortieren von Zufallszahlen mit verschiedenen Algorithmen.

Bei der Erweiterung eines heaps werden neue Knoten in der untersten Ebene von links nach rechts zugefügt; dies erfordert unter Umständen ein Umordnen von Elementen, um die Beziehung (2.1) wieder herzustellen.



Der Heapsort-Algorithmus arbeitet in zwei Phasen. Zunächst werden die Sortierschlüssel in einen heap umgeordnet. Dann erfolgt die eigentliche Sortierung vom Ende her, indem jeweils der Sortierschlüssel von der Wurzel (der größte verbleibende Schlüssel) aus dem heap entfernt wird; für die noch im heap verbliebenen Elemente wird durch Umordnen die heap-Eigenschaft wieder hergestellt. Die Abbildung 2.4 veranschaulicht den Sortiervorgang. Das im Anhang aufgeführte Fortran-Unterprogramm `FSORTH` ist eine durch die Einführung des Indexvektors weiterentwickelte Version einer Routine bei [58]. Der Aufruf ist identisch mit dem Aufruf bei dem Unterprogramm `ISORTS`, jedoch sind hier die Sortierschlüssel vom Typ `REAL`.

Vergleich der Algorithmen. In der Abbildung 2.5 wird die Rechenzeit pro Element für das Sortieren von $n = 10$ bis $n = 10^5$ Elementen für verschiedene Algorithmen auf einem bestimmten Prozessor verglichen. Die Elemente sind zwischen 0 und 1 gleichmäßig verteilte Zufallszahlen. Bei kleinen Werten bis $n = 100$ ist Shell's Algorithmus am schnellsten, bei größeren Werten von n dagegen Quicksort. Verglichen mit dem n^2-proportionalen einfachen Sortieren durch Einfügen sind die Unterschiede zwischen den drei Algorithmen Shell, Quicksort und Heapsort bis $n = 10^5$ eher gering. Durch kleine Veränderungen an den Programmen, auf anderen Prozessoren oder durch Verwendung optimierender Compiler können sich leicht Veränderungen um Faktoren der Größenordnung 2 ergeben.

Die Algorithmen Heapsort und Shell, die beide nicht fragil sind, sind in der Praxis zu empfehlen, außer wenn eine sehr gut angepaßte Version von Quicksort zur Verfügung steht.

Vereinigen von Listen (merge). Zwei sortierte Listen A und B mit jeweils n und m Elementen sollen zu einer sortierten Liste C mit $n + m$ Elementen vereinigt werden. Der folgende Algorithmus leistet dies. Beginnend mit den Listen A und B vergleicht er die jeweils ersten Sortierschlüssel; der jeweils kleinere wird nach C gebracht. Falls A oder B leer sind, wird der Rest nach C gebracht.

```
indexA=1;    indexB=1;    indexC=1;
while indexA =< n and indexB =< m do
   if A[indexA] < B[indexB]
      then   C[indexC] = A(indexA]; indexA = indexA + 1
      else   C[indexC] = B(indexA]; indexB = indexB + 1
```

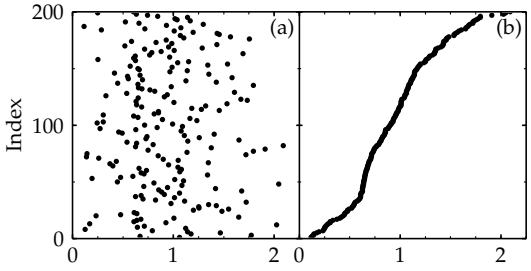


Abbildung 2.6: (a) Unsortierte Daten der Stichprobe vom Umfang 200 und (b) sortierte Daten. Auf der Abszisse ist der Wert der Variablen x aufgetragen.

```
    end if
    indexC = indexC + 1
end while

if indexB > m
    then  move A[indexA]...A[n]   to  C[indexC]...C[n+m]
    else  move B[indexB]...B[m]   to  C[indexC]...C[n+m]
end if
```

Man kann leicht sehen, daß dieser Algorithmus im ungünstigsten Fall nur $n + m - 1$ Vergleiche erfordert.

Mergesort. Der obenstehende Algorithmus kann zum vollständigen Sortieren benutzt werden. Dazu wird die zu sortierende Liste rekursiv immer weiter in je zwei etwa gleich große Teile geteilt, die Teillisten auf der untersten Ebene sortiert und dann alle Teillisten wieder vereinigt. Dieser Mergesort-Algorithmus hat selbst im ungünstigsten Fall ein Verhalten entsprechend $W(n) = n \lg n - n$, benötigt jedoch zusätzlichen Arbeitsspeicher.

Beispiel 2.1 Analyse einer Stichprobe mittels Sortieren.

Für eine Stichprobe von Zahlen x soll untersucht werden, ob die Zahlen sich bei einem oder mehreren bestimmten Werten häufen. Die Abbildung 2.6 zeigt links die Werte x_i aufgetragen gegen den Index i. Eine Häufung um bestimmte x-Werte ist nicht zu erkennen.

Die Abbildung 2.6 zeigt rechts die gleichen Daten nach dem Sortieren. Aus diesen sortierten Daten kann die Dichte $d(x)$ der Daten auf der x-Skala als Funktion von x leicht bestimmt werden, zum Beispiel nach der Formel

$$d(x_i) = \frac{2m}{x_{i+m} - x_{i-m}}$$

mit einem bestimmten Inkrement m. Die so definierte Dichte ist in der Abbildung 2.7 gegen x aufgetragen für das Inkrement $m = 7$; man erkennt deutlich ein scharfes Maximum bei etwa $x = 0.65$. Eine ähnliche Analyse kann auch mit Hilfe eines Histogramms der Stichprobe durchgeführt werden. Die dargestellte auf sortierten Daten basierende Methode vermeidet jedoch die evtl. vergröbernde Intervalleinteilung des Histogramms und ist auch auf Daten anwendbar, die in einem sehr weiten Bereich streuen.

2.4 Suchen

Suchoperationen treten oft innerhalb größerer Datenanalysen auf. Die Aufgabe ist, ein bestimmtes Element in einer Tabelle oder Liste zu finden. Der Algorithmus sollte als Ergebnis den Index des betreffenden Elements liefern oder null, falls das Element nicht gefunden wurde. Ein ähnliches Problem tritt beim Interpolieren in Funktions-Tabellen auf. In manchen Anwendungen ist die Länge n der Tabelle sehr groß und ein schneller Algorithmus wichtig. Die

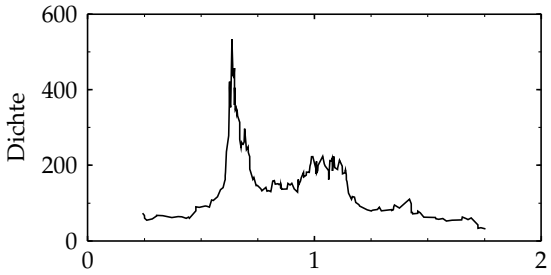


Abbildung 2.7: Aus den sortierten Daten der Stichprobe berechnete Dichte der Verteilung als Funktion der Variablen x.

Zeitabhängigkeit ist $A(n) = n$ für einfache Algorithmen, $A(n) = \lg n$ für eine binäre Suche und $A(n) =$ konstant für den hash-Algorithmus (berechenbare Speicheradressen).

Sequentielles Suchen. Das einfachste Verfahren ist eine Schleife über alle n Elemente der Tabelle `ITAB(1:N)`. Man sucht dasjenige Element, welches mit `ITEM` identisch ist. Der Wert des Index `I` mit `ITEM=ITAB(I)` wird bestimmt; falls kein solches Element gefunden werden kann, wird `I=0` gesetzt.

Wenn die Suche erfolglos ist, benötigt der Algorithmus n Vergleiche, also $W(n) = n$. Wenn das Element in der Liste ist, dann ist bei einer zufälligen Verteilung der Elemente die mittlere Zahl von Vergleichen etwa $A(n) = (n+1)/2$. Genau genommen benötigt man zusätzliche Instruktionen wegen der Abfrage auf Ende der Schleife. Mit einer kleinen Modifizierung der einfachen Schleife kann dieser zusätzliche Aufwand hier (und in ähnlichen Fällen mit einer oft durchlaufenen inneren Schleife) vermieden werden:

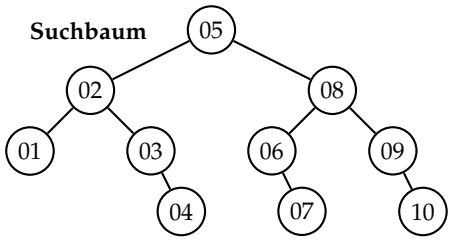
```
      ITAB(N+1)=ITEM        ! insert ITEM into table
      I=0
10    I=I+1                 ! loop
      IF(ITAB(I).NE.ITEM) GOTO 10
      IF(I.GT.N) I=0        ! not in table - I=0
```

Der Trick besteht darin, daß man `ITEM` als Element mit Index $n+1$ in die Tabelle einfügt. Damit ist man sicher, wenigstens ein Element zu finden, und braucht nie auf das Ende der Tabelle abzufragen. Falls gewisse Schlüssel häufiger vorkommen als andere, kann man sie an den Anfang der Tabelle bringen und so das Verfahren beschleunigen.

Falls die Tabelle sortiert ist, ist eine Verbesserung auf $n/2$ für den Fall erfolgloser Suche möglich, denn dann kann bei der sequentiellen Suche im Mittel nach der Hälfte der Vergleiche abgebrochen werden. Allerdings ist bei sortierten Listen sogar die schnelle binäre Suche möglich. Wenn neue Elemente in die Tabelle aufgenommen werden sollen, muß dies so geschehen, daß die Tabelle weiterhin sortiert bleibt. Dazu muß man unter Umständen viele Elemente umspeichern, was zeitaufwendig ist. Hier ist eine verkettete Liste vorteilhafter.

Binäre Suche. Diese schnelle Suchmethode erfordert, daß die Liste sortiert ist. Sie benutzt nie mehr als $\lceil \lg n \rceil + 1$ Vergleiche. Das Verfahren ist einfach und folgt dem Prinzip *teile und herrsche*. Um ein bestimmtes Element `ITEM` zu finden, wird es mit dem Element in der Mitte der Tabelle (Index $\lfloor (n+1)/2 \rfloor$) verglichen. Ist `ITEM` kleiner als dieses Element, dann muß es in der ersten Hälfte der Tabelle sein. Dies wird rekursiv angewandt, wobei sich bei jedem Schritt die Länge der Tabelle halbiert.

Die binäre Suche kann mit einem binären Suchbaum veranschaulicht werden, in dem die Knoten einen Schlüssel t_i tragen und so geordnet sind, daß bei jedem Unterbaum der linke Unterbaum nur Schlüssel enthält, die kleiner als t_i sind, und der rechte Unterbaum nur Schlüssel, die größer als t_i sind. In der Skizze rechts sind die Indizes für den Fall von zehn Schlüsseln gezeigt.



In der im Anhang aufgeführten Fortran-Funktion `LEFT` wird die binäre Suche angewandt, um in einer geordneten Liste t_i, $i = 1, 2, \ldots n$ das Intervall $[t_\ell, t_{\ell+1})$ zu finden, welches einen vorgegebenen Wert x einschließt. Das Ergebnis der Funktion, der Index ℓ mit $t_\ell \le x < t_{\ell+1}$, wird zum Beispiel bei der Funktionsinterpolation in Tabellen und für B-Splines (Kapitel 10.2) gebraucht. Statt der Rekursion werden in der Funktion zwei Marken benutzt, um die Untertabellen zu markieren. Die zwei Marken `L` und `R` werden mit den Werten `0` und `N+1` initialisiert, welche auf die fiktiven Elemente `TAB(0)` und `TAB(N+1)` zeigen, deren Wert zu $-\infty$ und $+\infty$ angenommen wird. Das Programm behandelt auch den Fall mehrfacher Tabelleneinträge, wie es bei der Anwendung zur Interpolation vorkommen kann. Die Suche in der Funktion `LEFT` und das Ergebnis sind in der Abbildung 2.8 für den Fall $n = 10$ mit der Vergleichsreihenfolge t_5, t_8, t_6, t_7 veranschaulicht. Die waagerechten Pfeile geben den Bereich der Werte der Funktion `LEFT` an (die linke Intervallgrenze ist eingeschlossen, die rechte nicht). Wegen $t_2 = t_3$ ist ein doppelter Knoten vorhanden.

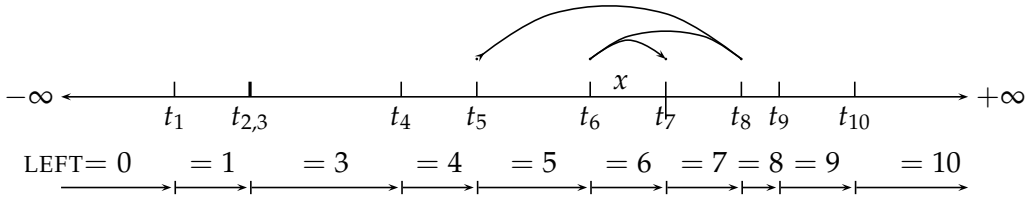


Abbildung 2.8: Binäre Suche nach dem Intervall mit $t_\ell \le x < t_{\ell+1}$.

Berechenbare Speicheradressen (Hashing). Falls die Schlüsselworte disjunkte ganze Zahlen zwischen 0 und $p - 1$ sind, dann kann jedes Element in einer Tabelle der Länge p mit dem Schlüsselwort als Index gespeichert werden. Das Element kann dann ohne Suchen direkt adressiert werden. Dies ist die bei weitem beste Möglichkeit, für große Werte von p benötigt man jedoch sehr viel Speicherplatz.

Berechenbare Speicheradressen (hashing) sind eine Verallgemeinerung dieser einfachen Methode. Sie werden zum Beispiel für die Verwaltung von Symboltabellen in einem Makroprozessor oder Übersetzungsprogramm verwendet. Zunächst wird mit Hilfe einer geeignet gewählten *hash-Funktion* das Schlüsselwort in einen Index umgerechnet, der als Adresse in der Tabelle dient. Die hash-Funktion sollte unterschiedliche Schlüssel in Indizes $0 \ldots (p-1)$ umrechnen, wobei es möglichst wenig Fälle geben sollte, wo mehr als ein Schlüssel zum selben Index führt, sogenannte Kollisionen. Diese Kollisionen müssen aufgelöst werden; zum Beispiel können alle zum selben Index führenden Schlüssel in einer verketteten Liste angeordnet werden. Praktisch wird beim hash-Algorithmus eine lange Liste in p kurze Unterlisten aufgespalten, die sequentiell organisiert werden können. Die Suche ist schnell für ein entsprechend großes p. Man muß gegebenenfalls bei der Auswahl des Algorithmus die Anforderungen an Rechenzeit gegen Speicherplatzbedarf abwägen.

Hash-Funktion. Wenn der Schlüssel eine ganze Zahl m ist, kann die hash-Funktion direkt durch die modulo-Funktion m modulo p realisiert werden, die als Ergebnis eine Zahl zwischen

0 und $(p-1)$ ergibt; als Zahl p wird bevorzugt eine Primzahl benutzt. Oft ist der Schlüssel ein aus mehreren Zeichen bestehender Name; ein Zeichen wird üblicherweise durch eine Zahl c_i zwischen 0 und 255 repräsentiert. Durch die Formel

$$m = \sum_i c_i \cdot 31^{i-1}$$

wird eine Zeichenkette in eine ganze Zahl m umgewandelt, die dann mit der modulo-Funktion in eine Zahl zwischen 0 und $(p-1)$ umgewandelt wird. Die Zahl 31 führt zu einer *Durchmischung* der Zeichen und damit zu einer geringen Wahrscheinlichkeit für Kollisionen.

2.5 Bereichsuche (range search)

Eine weitere Form von Suchoperationen, genannt *range searching*, tritt bei Datenbank-Abfragen und zum Beispiel auch bei bestimmten Hypothesentests (Kapitel 9.3.3) auf. Dabei ist die Aufgabe, unter einer eventuell großen Zahl von Daten alle Elemente zu finden, die in einem bestimmten Größenbereich fallen. Beispiel sind Punkte in einer Ebene, von denen alle in einem gegebenen Rechteck liegenden Punkte auszuwählen sind.

Ist die Gesamtzahl N von Daten nicht zu groß, kann man eine einfache sequentielle Suche durchführen, die bei M Abfragen eine Rechenleistung proportional zu nNM, erfordert, wenn jedes Element n Komponenten hat. Bei größeren Datenmengen und eventuell einer größeren Dimension n kann die geforderte Rechenleistung sehr groß werden.

Verbesserte Suchmethoden, die auf einer Verallgemeinerung der Suchmethoden im Fall $n = 1$ beruhen, zum Beispiel der binäre Suche 2.4, bestehen aus zwei Teilen: der (1) Vorbereitung mit der Aufstellung von Indexstrukturen und (2) der Suchoperation selbst. Bei eindimensionalem *range search* erfordert die Vorbereitung $O(N \log N)$ Schritte und die Suchoperation $O(R + \log N)$ Schritte, wenn R die Anzahl der gefundenen Elemente ist.

Betrachtet wird zunächst der Fall $n = 2$, bei dem die beiden Komponenten die x- und y-Koordinaten von Punkten in der Ebene sind. Es wird ein binärer Suchbaum (Kapitel 2.2) aufgebaut mit den Punkten in den Knoten, wobei die x und y Koordinaten abwechselnd als Schlüssel benutzt werden, mit y in der Wurzel. Alle Punkte mit y-Werten unterhalb des Punktes in der Wurzel sind im linken Unterbaum, alle darüber im rechten; die Punkte mit y oberhalb des Wertes der Wurzel und mit x links des Wertes des Unterbaums gehen in den linken Unterbaum des rechten Unterbaums der Wurzel, usw. für die weiteren Ebenen. Auf diese Weise wird die Ebene in einfacher Weise aufgeteilt. Die Konstruktion eines Baumes in zwei Dimensionen aus zufällig verteilten Punkten erfordert im Mittel $2N \log N$ Vergleiche, und *range search* erfordert etwa $R + \log N$ Schritte um R von insgesamt N Punkten zu finden.

Die Methode lässt sich auf mehr ($n > 2$) Dimensionen erweitern. Bei n Dimensionen sind diese Dimensionen in den Ebenen des Baums zyklisch zu benutzen. Die Effizienz mit logarithmischer Proportionalität der Zahl der Schritte kann insbesondere bei einer großen Zahl von Dimensionen durch Ungleichgewichte im Suchbaum abnehmen.

Der Anhang enthält Programme für range search *in einer Tabelle mit beliebiger Dimension. Mit dem Unterprogram* `DFTREE` *wird eine Indexstruktur für die Tabelle in einem eindimensionalen Feld aufgebaut. Für die Suchoperationen gibt es zwei Unterprogramme. Mit* `NCTREE` *werden bei einem Aufruf die Indizes aller gefundenen Elemente in einem Common-Bereich gespeichert. Die Funktion* `ITREE` *gibt jeweils den Index* eines *gefundenen Elementes zurück.*

2.6 Weitere Algorithmen

Gegeben ist eine Menge von n Elementen E_1, E_2, ... E_n. Wenn die Elemente reelle Zahlen sind, dann gibt es zwischen je zwei Elementen eine Relation $\leq$, und man kann durch Sortieren eine wohldefinierte Ordnung der Elemente herstellen. Sortieren ist nicht möglich, wenn zwischen den Elementen eine Relation $\leq$ nicht definiert werden kann, zum Beispiel wenn die Elemente Punkte in einem mehrdimensionalen Raum darstellen. Um trotzdem für eine Menge solcher Elemente eine gewisse *Ordnung* herzustellen, kann man versuchen, andere Relationen zwischen Paaren von Elementen auszunutzen.

Äquivalenzklassen. Die beiden Elemente eines Paares können in einem bestimmten Sinne *äquivalent* (Symbol $\equiv$) sein, mit den folgenden Eigenschaften:

1. Aus $E_i \equiv E_j$ und $E_j \equiv E_k$ folgt: $E_i \equiv E_k$ (Transitivität);
2. Aus $E_i \equiv E_j$ folgt $E_j \equiv E_i$ (Symmetrie);
3. $E_i \equiv E_i$ (Reflexivität).

Durch Äquivalenzrelationen dieser Art zwischen den Paaren einer Menge lassen sich *disjunkte* Äquivalenzklassen von Elementen definieren: Elemente eines Paares der *gleichen* Klasse sind äquivalent. Zum Beispiel teilen die Äquivalenzrelationen

$$E_1 \equiv E_5 \quad E_6 \equiv E_8 \quad E_7 \equiv E_2 \quad E_9 \equiv E_8 \quad E_3 \equiv E_7 \quad E_4 \equiv E_2 \quad E_9 \equiv E_3 \tag{2.2}$$

die Menge der Elemente E_1, E_2, ... E_9 in die beiden Klassen

$$\{E_1, E_5\} \qquad \{E_2, E_3, E_4, E_6, E_7, E_8, E_9\} \ .$$

Ausgehend von einer Liste von äquivalenten Paaren E_j und E_k von Elementen kann man jedem der n Elemente eine Äquivalenzklassen-Nummer zuordnen, so daß zwei Elemente dann und nur dann in derselben Äquivalenzklasse sind, wenn sie dieselbe Äquivalenzklassen-Nummer haben. Dies wird durch den folgenden Algorithmus geleistet.

Es sei $F(j)$ die Klasse von Element E_j mit Index j. Zunächst weist man jedem Element eine eigene Klasse zu, also $F(j) = j$. Diese Klasse kann als Beginn einer Baumstruktur aufgefaßt werden, mit $F(j)$ als Vater von j. In diesem Stadium kann jede Klasse durch ihre Wurzel charakterisiert werden. Jede Markierung einer paarweisen Klassenzugehörigkeit, wie zum Beispiel 'j und k sind äquivalent' wird nun so behandelt, daß j zu seinem höchsten Vorfahr, k zu seinem Vater durchverfolgt wird, und j wird dann k als neuer Vater (oder umgekehrt) zugeordnet. Nachdem alle Relationen in dieser Weise abgearbeitet sind, geht man nochmals durch alle Elemente j durch; ihre Klassenzugehörigkeit $F(j)$ wird gleich demjenigen ihres höchsten Vorfahrs gesetzt, welches dann auch ihre Klassen-Nummer ist. Dieser Algorithmus ist in dem Fortran-Unterprogramm `EQUICL` nach einer Routine bei [58] implementiert und im Anhang aufgeführt.

Bei der Eingabe der $m = 7$ Äquivalenzrelationen Gleichung (2.2) für die 9 Elemente erhält man als Ergebnis die nebenstehende Tabelle `NC(9)` mit den Äquivalenzklassen-Nummern. Die obere Zeile zeigt die Initialisierung (jedes Element bildet eine eigene Klasse), die weiteren Zeilen Zwischenergebnisse und die letzte Zeile das Endergebnis.

1	2	3	4	5	6	7	8	9
5	2	3	4	5	6	7	8	9
5	2	3	4	5	8	7	8	9
5	2	3	4	5	8	2	8	9
5	2	3	4	5	8	2	8	8
5	2	2	4	5	8	2	8	8
5	2	2	2	5	8	2	8	8
5	2	2	2	5	8	2	2	8
5	2	2	2	5	2	2	2	2

Minimum Spanning Tree. Hier geht man aus von einer passend gewählten Definition des *Abstands* zweier Elemente,

$$d_{ij} = d(E_i, E_j) = \quad \text{Abstand der Elemente } E_i \text{ und } E_j \ ,$$

der beim Ordnungsprinzip benutzt wird. Die Paare (E_i, E_j) werden mit dem Index k bezeichnet; der Abstand d_{ij} kann als Gewicht w_k für das k-te Paar bezeichnet werden. Die Zahl der Relationen sei m; für n Elemente ist $m \leq n(n-1)/2$.

Diese Struktur, gebildet aus allen Relationen, wird als gewichteter Graph G bezeichnet. Für einen solchen gewichteten Graphen kann man einen *spanning tree* definieren. Dies ist ein Untergraph, der alle Elemente (Vertizes) so verknüpft, daß das Gebilde die Bedingung einer Baumstruktur erfüllt. Aus den Gewichten, die den Verbindungslinien zugeordnet sind, kann für jeden *spanning tree* ein totales Gewicht $\sum_k w_k$ berechnet werden, wobei die Summe über alle in dem Baum *benutzten* Verbindungen geht. Ein *minimum spanning tree* ist der Baum, der von allen möglichen *spanning trees* das kleinste totale Gewicht hat. Eine mögliche Anwendung ist das Problem, eine Anzahl Stationen durch Leitungen zu verbinden, wobei die Gesamtlänge der Leitung möglichst kurz sein soll.

Es gibt zwei Algorithmen, um einen minimum spanning tree zu konstruieren, einer von Dijkstra [19] und einer von Kruskal [45]. Beide Algorithmen sind *gierig* (greedy) in dem Sinne, daß jede Entscheidung des Algorithmus so getroffen wird, daß sie *lokal* optimal ist. Man kann für beide Algorithmen zeigen, daß dies auch zu einer *global* optimalen Lösung führt.

Der Kruskal-Algorithmus geht von n Elementen (Vertizes) und m Gewichten aus; er ist effizient, falls die Zahl m klein gegenüber der maximal möglichen Zahl $n(n-1)/2$ ist, und arbeitet folgendermaßen: Zunächst werden die Gewichte der Größe nach sortiert, danach arbeitet der Algorithmus auf dieser sortierten Liste. Die Verbindung mit dem kleinsten Gewicht wird jeweils ausgewählt, vorausgesetzt, sie macht *keine* Schleife, was mit dem Äquivalenzklassen-Algorithmus geprüft werden kann. Dies wird wiederholt, bis alle Vertizes aufgebraucht sind. Der Algorithmus ist in dem Fortran-Unterprogramm `MSTREE` im Anhang aufgeführt.

Beispiel 2.2 Minimum spanning tree.

Die Abbildung 2.9 (a) zeigt eine Menge von Elementen als Kreise in der Ebene. Als Abstand zwischen je zwei Elementen wird der euklidische Abstand zwischen den Kreismittelpunkten definiert, wobei Abstände oberhalb eines bestimmten Wertes ignoriert werden.

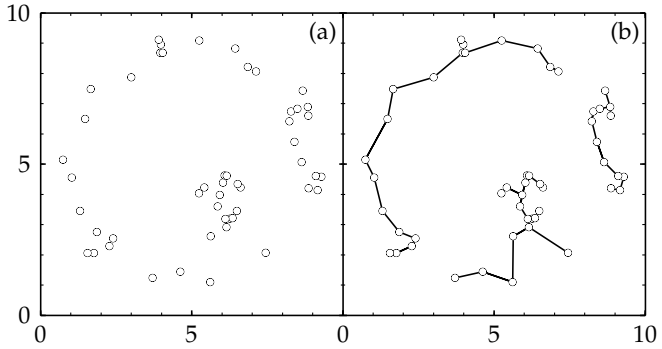


Abbildung 2.9: Konstruktion eines *minimum spanning tree* für eine Anzahl von Objekten (Kreise) (a). Als Gewicht ist der euklidische Abstand unterhalb einer Grenze angegeben worden. Das Ergebnis (b) zeigt mehrere nicht-zusammenhängende Bäume.

Die Abbildung 2.9 (b) zeigt, angegeben durch Linien zwischen Kreisen, das Ergebnis des Kruskal-Algorithmus. Die Linien lassen eine bestimmte Struktur erkennen. Da Abstände oberhalb eines bestimmten Wertes ignoriert wurden, ist als Ergebnis ein *Wald* von mehreren nicht-zusammenhängenden Bäumen entstanden.

3 Methoden der linearen Algebra

3.1 Vektoren und Matrizen

Vektoren. Ein Vektor $\boldsymbol{x}$ ist eine Menge von n Elementen $x_1, x_2, \ldots x_n$ und wird dargestellt in einer vertikalen Spalte von n Zahlen (Spaltenvektor) mit dem ersten Element an der obersten Stelle; der transponierte Vektor $\boldsymbol{x}^T$ ist ein Zeilenvektor

$$\boldsymbol{x} = \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{pmatrix} \qquad \boldsymbol{x}^T = (x_1, x_2, \ldots x_n) \ . \tag{3.1}$$

Das Skalarprodukt (oder innere Produkt) ist das Produkt zweier Vektoren $\boldsymbol{x}$ und $\boldsymbol{y}$ der gleichen Dimension. Der Skalar $\boldsymbol{x}^T\boldsymbol{y}$ wird definiert durch

$$\boldsymbol{x}^T\boldsymbol{y} = \sum_{i=1}^{n} x_i\, y_i = x_1\, y_1 + x_2\, y_2 + \ldots + x_n\, y_n \ . \tag{3.2}$$

Es gilt $\boldsymbol{x}^T\boldsymbol{y} = \boldsymbol{y}^T\boldsymbol{x}$. Wenn $\boldsymbol{x}^T\boldsymbol{y} = 0$ ist, dann sind die beiden Vektoren *orthogonal* zueinander (es sei denn, einer oder beide Faktoren sind Nullvektoren). Die Norm eines Vektors wird definiert durch

$$|\boldsymbol{x}| = \left(\boldsymbol{x}^T\boldsymbol{x}\right)^{1/2} \ .$$

Matrizen. Eine $m \times n$ Matrix $\boldsymbol{A}$ ist eine Anordnung von m Zeilen und n Spalten; das Element A_{ij} ist das Element in der i-ten Zeile und der j-ten Spalte

$$\boldsymbol{A} = (A_{ij}) = \begin{pmatrix} A_{11} & A_{12} & \ldots & A_{1n} \\ A_{21} & A_{22} & \ldots & A_{2n} \\ . & . & \ldots & . \\ A_{m1} & A_{m2} & \ldots & A_{mn} \end{pmatrix} \ . \tag{3.3}$$

Die Transponierte der Matrix $\boldsymbol{A}$ wird mit $\boldsymbol{A}^T$ bezeichnet und hat die Elemente $(\boldsymbol{A}^T)_{ij} = (\boldsymbol{A})_{ji}$.

Die Elemente A_{ii} einer quadratischen Matrix werden *Diagonalelemente* der Matrix $\boldsymbol{A}$ genannt. Die Summe der Diagonalelemente heißt die Spur der Matrix: $\mathrm{Spur}(\boldsymbol{A}) = \sum_i A_{ii}$. Die Einheitsmatrix, geschrieben $\boldsymbol{I}$, hat alle Diagonalelemente gleich 1 und alle anderen Elemente gleich 0.

Die Multiplikation eines Vektors oder einer Matrix mit einem Skalar ist einfach die Multiplikation eines jeden Elementes mit dem Skalar. Addition und Subtraktion von Vektoren und Matrizen werden elementweise ausgeführt; die beiden Operanden müssen in den Dimensionen übereinstimmen.

Das Produkt der $m \times n$ Matrix $\boldsymbol{A}$ und der $n \times k$ Matrix $\boldsymbol{B}$ ist eine $m \times k$ Matrix $\boldsymbol{C}$,

$$\boldsymbol{C} = \boldsymbol{A}\,\boldsymbol{B} \qquad C_{ij} = \sum_{k=1}^{n} A_{ik}\, B_{kj} \ . \tag{3.4}$$

Die Anzahl der Spalten der Matrix $\boldsymbol{A}$ muß mit der Anzahl der Zeilen der Matrix $\boldsymbol{B}$ übereinstimmen. Das Element C_{ij} der Produktmatrix ist gleich dem Skalarprodukt der i-ten Zeile der Matrix $\boldsymbol{A}$ und der j-ten Spalte der Matrix $\boldsymbol{B}$. Die Matrixmultiplikation ist keine vertauschbare Operation; im allgemeinen ist $\boldsymbol{AB} \neq \boldsymbol{BA}$. Für das Setzen von Klammern in Vielfachprodukten gilt

$\boldsymbol{ABC} = (\boldsymbol{AB})\boldsymbol{C} = \boldsymbol{A}(\boldsymbol{BC})$, und beim Transponieren eines Matrixprodukts $(\boldsymbol{AB})^T = \boldsymbol{B}^T\boldsymbol{A}^T$.

Für die Berechnung des Matrixprodukts $\boldsymbol{C} = \boldsymbol{AB}$ sind $m \times k \times n$ Produkte zu berechnen. Für die numerische Berechnung von Mehrfachprodukten wie $\boldsymbol{E} = \boldsymbol{ABC}$ hängt die Zahl der Multiplikationen (aber natürlich nicht das Ergebnis) vom Setzen der Klammern ab. Es sei $\boldsymbol{A}$ eine $m \times n$, $\boldsymbol{B}$ eine $n \times k$ und $\boldsymbol{C}$ eine $k \times l$ Matrix. Berechnet man $\boldsymbol{E}$ als $\boldsymbol{E} = (\boldsymbol{AB})\boldsymbol{C}$, so sind hierzu $m \cdot k \cdot (l + n)$ Multiplikationen notwendig. Berechnet man aber $\boldsymbol{E} = \boldsymbol{A}(\boldsymbol{BC})$, so sind hierzu $n \cdot l \cdot (m + k)$ Multiplikationen notwendig.

Determinanten. Die Determinante einer $n \times n$ Matrix ist definiert durch

$$\det \boldsymbol{A} = \sum_{\text{perm}} (-1)^k A_{1\alpha} A_{2\beta} \dots A_{n\omega}, \tag{3.5}$$

wobei die Indizes $\alpha, \beta, \dots \omega$ alle $n!$ Permutationen der Zahlen 1, 2, $\dots n$ sind; das Vorzeichen $(-1)^k$ eines jeden Terms ist durch die Anzahl der Inversionen in jeder Permutation bestimmt. Die Determinante einer nicht-singulären Matrix ist $\neq 0$. Die Determinante kann über die rekursive Gleichung

$$\det \boldsymbol{A} = \sum_{i=1}^{n} (-1)^{i+j} A_{ij} \det \boldsymbol{A}^{(ij)} \tag{3.6}$$

für jede beliebige Spalte j berechnet werden, oder alternativ durch

$$\det \boldsymbol{A} = \sum_{j=1}^{n} (-1)^{i+j} A_{ij} \det \boldsymbol{A}^{(ij)} \tag{3.7}$$

für jede Zeile i, wobei $\boldsymbol{A}^{(ij)}$ die Matrix ist, die man erhält, wenn man die i-te Zeile und j-te Spalte durchstreicht (Cramersche Regel). Diese Methode erfordert mehr als $n!$ Multiplikationen für eine $n \times n$ Matrix und sollte nur angewandt werden, wenn n klein ist. Bessere Methoden zur Berechnung (siehe Kapitel 3.4 und 3.5) machen Gebrauch von den Gleichungen

$$\det(\boldsymbol{AB}) = \det(\boldsymbol{A}) \cdot \det(\boldsymbol{B}) \qquad \det\left(\boldsymbol{A}^T\right) = \det(\boldsymbol{A}) \ . \tag{3.8}$$

Lineare Transformationen. Eine *lineare* Transformation kann in einer Matrixgleichung geschrieben werden

$$\boldsymbol{y} = \boldsymbol{Ax} \qquad y_i = \sum_{k=1}^{n} A_{ik}\, x_k \ . \tag{3.9}$$

Bei einer linearen Transformation eines n-Vektors $\boldsymbol{x}$ in einen n-Vektor $\boldsymbol{y}$ ist die Matrix $\boldsymbol{A}$ quadratisch. Eine *eindeutige* inverse Transformation wird durch $\boldsymbol{x} = \boldsymbol{A}^{-1}\boldsymbol{y}$ beschrieben, wobei die Matrix $\boldsymbol{A}^{-1}$ die inverse Matrix der Matrix $\boldsymbol{A}$ ist mit

$$\boldsymbol{A}\boldsymbol{A}^{-1} = \boldsymbol{A}^{-1}\boldsymbol{A} = \boldsymbol{I} \ .$$

Die Transformation heißt *regulär* oder *nicht-singulär*, wenn die inverse Matrix $\boldsymbol{A}^{-1}$ existiert; in diesem Fall ist $\det(\boldsymbol{A}) \neq 0$.

System linearer Gleichungen. Die Bestimmung von Unbekannten aus einem System von linearen Gleichungen spielt in der Datenanalyse eine große Rolle. Es sei ein System von m Gleichungen gegeben

$$\begin{aligned}
A_{11}x_1 + A_{12}x_2 + \cdots + A_{1n}x_n &= b_1 \\
A_{21}x_1 + A_{22}x_2 + \cdots + A_{2n}x_n &= b_2 \\
\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots & \\
A_{m1}x_1 + A_{m2}x_2 + \cdots + A_{mn}x_n &= b_m
\end{aligned}$$

mit numerischen Werten für die Koeffizienten A_{ik} und die Konstanten b_i auf der rechten Seite; die n Unbekannten x_k sollen bestimmt werden. In Matrixnotation können die Gleichungen so geschrieben werden

$$\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b} \qquad \sum_{k=1}^{n} A_{ik}\, x_k = b_i \qquad i = 1,\, 2, \ldots m\,. \tag{3.10}$$

Für $n = m$ gibt es ebensoviele Gleichungen wie Unbekannte. Wenn das Gleichungssystem nicht singulär ist, gibt es eine eindeutige Lösung $\boldsymbol{x}$. Ein Gleichungssystem heißt singulär im Fall von Entartung; das ist der Fall, wenn eine oder mehrere der m Gleichungen eine Linearkombination von anderen sind, oder wenn bestimmte Variablen in der gleichen Linearkombination erscheinen. Methoden, Gleichungssysteme zu lösen, können numerisch ungenau sein, wenn die Matrix *fast* singulär ist. Die Anzahl der Operationen zur Lösung eines Gleichungssystems nimmt zu wie n^3. Infolgedessen wächst für große n die Rechenzeit rasch mit n, und zusätzlich können Rundungsfehler ein Problem werden, insebesondere wenn die Konditionszahl $\kappa \gg 1$ ist (siehe Seite 38). Im allgemeinen werden Matrizen mit einem Dimensionsparameter $n > 100$ und Matrizen mit großer Konditionszahl ein Rechnen mit doppelter Genauigkeit erfordern. Große Matrizen haben oftmals eine spezielle Struktur mit vielen Nullelementen (sogenannte *dünne* (engl. sparse) Matrizen), und durch spezielle Methoden kann die Rechenzeit z.B. zur Lösung von Gleichungssystemen reduziert werden.

Bandmatrix. In einer Bandmatrix $\boldsymbol{A}$ sind alle Elemente A_{ij} gleich null für alle i, j mit $|i - j| > m$, d.h. nur die Elemente in der Diagonalen und in einem Band um die Diagonale sind ungleich null. Der Wert m wird Bandbreite genannt. Für $m = 0$ heißt die Matrix *diagonal* und für $m = 1$ *tridiagonal*. Die Bandmatrix ist ein Beispiel für eine Matrix mit spezieller Struktur. Die inverse Matrix einer Bandmatrix mit $m > 0$ ist eine normale Matrix, d.h. die Bandeigenschaft geht bei Inversion verloren. Es gibt jedoch Methoden zur Lösung von Matrixgleichungen, die von der speziellen Bandstruktur Gebrauch machen und die Rechenzeit bedeutend reduzieren.

Beispiel 3.1 Zwei lineare Gleichungen mit zwei Unbekannten.

Für eine 2×2 Matrix $\boldsymbol{A}$ ist die inverse Matrix ein einfacher Ausdruck,

$$\boldsymbol{A} = \begin{pmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{pmatrix} \qquad \boldsymbol{A}^{-1} = \frac{1}{\det \boldsymbol{A}} \begin{pmatrix} A_{22} & -A_{12} \\ -A_{21} & A_{11} \end{pmatrix}\,,$$

mit $\det(\boldsymbol{A}) = A_{11} \cdot A_{22} - A_{12} \cdot A_{21}$.
Für $\det \boldsymbol{A} \neq 0$ existiert die inverse Matrix und sie kann verwendet werden, um eine Matrixgleichung $\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}$ zu lösen durch

$$\begin{aligned} x_1 &= (A_{22}b_1 - A_{12}b_2) \,/\, \det(\boldsymbol{A}) \\ x_2 &= (A_{11}b_2 - A_{21}b_1) \,/\, \det(\boldsymbol{A})\,. \end{aligned}$$

Eigenvektoren und Eigenwerte. Ein nicht-verschwindender Vektor $\boldsymbol{x}$, der die Gleichung

$$\boldsymbol{A}\boldsymbol{x} = \lambda \boldsymbol{x} \qquad \text{oder} \qquad (\boldsymbol{A} - \lambda \boldsymbol{I})\, \boldsymbol{x} = 0 \tag{3.11}$$

erfüllt, wird *Eigenvektor* der (quadratischen) Matrix $\boldsymbol{A}$ genannt; der Skalar λ heißt *Eigenwert*. Die Eigenwerte λ sind durch die Bedingung $\det(\boldsymbol{A} - \lambda \boldsymbol{I}) = 0$ bestimmt. Das charakteristische Polynom n-ten Grades in λ ist $\det(\boldsymbol{A} - \lambda \boldsymbol{I}) = p(\lambda) = (\lambda)^n + (\lambda)^{n-1}$ Spur $(\boldsymbol{A}) + \ldots + \det(\boldsymbol{A})$. Ein Polynom vom n-ten Grad hat genau n Nullstellen. Die n Nullstellen sind die n Eigenwerte $\lambda_1, \lambda_2 \ldots \lambda_n$. Deshalb kann eine allgemeine Matrix $\boldsymbol{A}$ mit reellen Elementen bis zu n unterschiedliche Eigenwerte haben, die im allgemeinen komplex sein können.

Zu quadratischen ($n = m$) Matrizen $\boldsymbol{A}$ mit nicht verschwindenden Eigenwerten kann die inverse Matrix $\boldsymbol{A}^{-1}$ bestimmt werden. Als Kondition der Matrix wird das Verhältnis des größten zum kleinsten Eigenwert bezeichnet: $\kappa = \lambda_{\max}/\lambda_{\min}$. Ein großer Zahlenwert der Kondition bedeutet größere Rundungsfehler bei der Berechnung der inversen Matrix und bei der Lösung von Gleichungssystemen.

Die Transformation einer Matrix $\boldsymbol{A}$ in eine Matrix $\boldsymbol{B}$ durch

$$\boldsymbol{B} = \boldsymbol{C}^{-1}\boldsymbol{A}\boldsymbol{C} \tag{3.12}$$

mit einer regulären Matrix $\boldsymbol{C}$ wird *Ähnlichkeits-Transformation* genannt. Die Eigenwerte einer Matrix ändern sich durch eine Ähnlichkeits-Transformation *nicht*, wie man aus der folgenden Rechnung sieht:

$$\begin{aligned}\det(\boldsymbol{B} - \lambda\boldsymbol{I}) &= \det\left(\boldsymbol{C}^{-1}\boldsymbol{A}\boldsymbol{C} - \lambda\boldsymbol{I}\right) = \det\left(\boldsymbol{C}^{-1}(\boldsymbol{A} - \lambda\boldsymbol{I})\,\boldsymbol{C}\right) \\ &= \det\boldsymbol{C}^{-1}\cdot\det(\boldsymbol{A} - \lambda\boldsymbol{I})\cdot\det\boldsymbol{C} = \det(\boldsymbol{A} - \lambda\boldsymbol{I})\ .\end{aligned}$$

Das charakteristische Polynom ist also invariant unter einer Ähnlichkeits-Transformation. Die durch diese Transformation miteinander verbundenen Matrizen $\boldsymbol{A}$ und $\boldsymbol{B}$ heißen *ähnlich*; sie repräsentieren die gleiche Transformation, aber mit unterschiedlicher Basis. Eine Matrix $\boldsymbol{A}$ kann in eine Diagonalmatrix $\boldsymbol{D} = \mathrm{diag}(\lambda_1, \ldots \lambda_n)$ transformiert werden mit den Eigenwerten als Diagonalelementen.

Speichern von Matrizen. Eine $m \times n$ Matrix $\boldsymbol{A}$ mit Elementen A_{ij} wird in einem Rechner als eine zweidimensionale Tabelle `A` mit Elementen `A(I,J)` gespeichert. Zum Beispiel wird eine 3×2 Matrix

$$\boldsymbol{A} = \begin{pmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \\ A_{31} & A_{32} \end{pmatrix} \quad \text{gespeichert als} \quad \begin{pmatrix} \texttt{A(1,1)} & \texttt{A(1,2)} \\ \texttt{A(2,1)} & \texttt{A(2,2)} \\ \texttt{A(3,1)} & \texttt{A(3,2)} \end{pmatrix}\ .$$

Der Speicher im Rechner ist vom Aufbau her eindimensional, d.h. er hat eine lineare Struktur, in der aufeinanderfolgende Elemente fortlaufend numeriert werden. Wie in Kapitel 2.2 ausgeführt wird, können die Elemente `A(I,J)` einer zweidimensionalen Tabelle `REAL A(M,N)` den linearen Adressen q des Speichers auf zweierlei Weise zugeordnet werden:

$$\begin{aligned} q &= (i-1)*n + j \qquad \text{zeilenweise (lexikalische Ordnung)} \\ q &= (j-1)*m + i \qquad \text{spaltenweise (inverse lexikalische Ordnung);} \end{aligned}$$

in beiden Fällen hängt die Adresse, zusätzlich zu dem Indexpaar i, j, von einem der beiden Dimensionsparameter in der Tabellen-Deklaration ab. Die meisten Programmier-Sprachen verwenden die *zeilenweise* Ordnung (in Fortran wird dagegen die *spaltenweise* Ordnung verwendet).

Die Speicherung von Matrizen in Fortran wird im Anhang (Seite 264) beschrieben.

3.2 Symmetrische Matrizen

Eine quadratische Matrix $\boldsymbol{V}$ heißt *symmetrisch*, wenn $\boldsymbol{V}^T = \boldsymbol{V}$ ist, d.h. $V_{ij} = V_{ji}$. Viele Matrizen in Problemen der Datenanalyse sind symmetrisch mit reellen Elementen; Kovarianzmatrizen sind ein wichtiges Beispiel. Deshalb wird in diesem Kapitel besonderes Gewicht auf numerische Methoden für reelle symmetrische Matrizen gelegt. Die inverse Matrix einer (nicht-singulären) symmetrischen Matrix ist ebenfalls symmetrisch.

Die Eigenwerte einer reellen symmetrischen Matrix sind alle *reell*. Alle Eigenwerte sind positiv für eine positiv-definite symmetrische Matrix. Eigenvektoren zu verschiedenen Eigenwerten sind orthogonal.

Wenn $\boldsymbol{A}$ eine allgemeine $n \times m$ Matrix ist und $\boldsymbol{A}^T$ die transponierte, dann ist

$$\boldsymbol{W} = \boldsymbol{A}^T\boldsymbol{A} \tag{3.13}$$

eine symmetrische, quadratische $n \times n$ Matrix (mit $W_{ij} = W_{ji}$). Der Ausdruck

$$Q(\boldsymbol{x}) = \boldsymbol{x}^T\boldsymbol{W}\,\boldsymbol{x} = \boldsymbol{x}^T\boldsymbol{A}^T\boldsymbol{A}\boldsymbol{x} = (\boldsymbol{A}\boldsymbol{x})^T\,(\boldsymbol{A}\boldsymbol{x}) \ , \tag{3.14}$$

quadratische Form genannt, ist ein nicht-negativer Skalar. Wenn die quadratische Form nur positive Werte annimmt, $Q(\boldsymbol{x}) > 0$ für jeden (nicht-verschwindenden) Vektor $\boldsymbol{x}$, dann wird die Matrix *positiv-definit* genannt.

Eine Produktmatrix der Art

$$\boldsymbol{V}_2 = \boldsymbol{A}\boldsymbol{V}_1\boldsymbol{A}^T \ , \tag{3.15}$$

wobei $\boldsymbol{V}_1$ eine symmetrischen Matrix der Ordnung n und $\boldsymbol{A}$ eine allgemeine $m \times n$ Matrix ist, wird bei der Fehlerfortpflanzung benutzt. Das Ergebnis ist eine symmetrische Matrix $\boldsymbol{V}_2$ der Ordnung m.

Eine symmetrische $n \times n$ Matrix hat $(n^2 + n)/2$ $(\approx n^2/2)$ unterschiedliche Elemente. Weil symmetrische Matrizen gerade in der Datenanalyse besonders häufig auftreten, kann sowohl Speicherplatz als auch Rechenzeit durch eine entsprechende Speicherung und spezielle Programme für Matrixoperationen reduziert werden. Dies lohnt sich insbesondere bei sehr großen Matrizen. Kann ein Problem so formuliert werden, daß die symmetrische Matrix Bandgespalt hat, kann die Reduktion von Speicherplatz und Rechenzeit mehrere Größordnungen betragen. Leider wird davon selten Gebrauch gemacht.

Einige der Fortran-Programme aus dem Anhang sind speziell für symmetrische Matrizen und Bandmatrizen geschrieben, und erfordern eine spezielle Form der Speicherung. Diese wird im Anhang (Seite 264) beschrieben.

3.3 Vertauschungs-Algorithmus

Vertauschen von Variablen. Die Idee wird mit einem einfachen Beispiel eingeführt. Es sei eine lineare Funktion y_2 von vier unabhängigen Variablen $x_1 \ldots x_4$ gegeben:

$$y_2 = A_{21}x_1 + A_{22}x_2 + A_{23}x_3 + A_{24}x_4 \ . \tag{3.16}$$

Die Rollen von y_2 und einer der unabhängigen Variablen, zum Beispiel x_3, werden vertauscht, wenn man die Gleichung (3.16) nach x_3 auflöst.

$$x_3 = -\frac{A_{21}}{A_{23}}x_1 - \frac{A_{22}}{A_{23}}x_2 + \frac{1}{A_{23}}y_2 - \frac{A_{24}}{A_{23}}x_4 \tag{3.17}$$

(natürlich wird $A_{23} \neq 0$ gefordert). Man nimmt an, daß die ursprüngliche Gleichung eine von mehreren Gleichungen ist, also zum Beispiel

$$\begin{aligned} y_1 &= A_{11}x_1 + A_{12}x_2 + A_{13}x_3 + A_{14}x_4 \\ y_2 &= A_{21}x_1 + A_{22}x_2 + A_{23}x_3 + A_{24}x_4 \\ y_3 &= A_{31}x_1 + A_{32}x_2 + A_{33}x_3 + A_{34}x_4 \ . \end{aligned} \tag{3.18}$$

Die Variable x_3 auf der rechten Seite von Gleichung (3.18) kann mit Hilfe der neuen Gleichung (3.17) in x_1, x_2, x_4 und y ausgedrückt werden. Das Ergebnis ist im Schema unten dargestellt.

$$
\begin{array}{c|cccc}
 & x_1 & x_2 & x_3 & x_4 \\
\hline
y_1 = & A_{11} & A_{12} & A_{13} & A_{14} \\
y_2 = & A_{21} & A_{22} & \underline{A_{23}} & A_{24} \\
y_3 = & A_{31} & A_{32} & A_{33} & A_{34}
\end{array}
\quad \Longrightarrow \quad
\begin{array}{c|cccc}
 & x_1 & x_2 & y_2 & x_4 \\
\hline
y_1 = & \alpha_{11} & \alpha_{12} & \dfrac{A_{13}}{A_{23}} & \alpha_{14} \\
x_3 = & -\dfrac{A_{21}}{A_{23}} & -\dfrac{A_{22}}{A_{23}} & \dfrac{1}{A_{23}} & -\dfrac{A_{24}}{A_{23}} \\
y_3 = & \alpha_{31} & \alpha_{32} & \dfrac{A_{33}}{A_{23}} & \alpha_{34}
\end{array}
$$

$$
\begin{aligned}
\alpha_{11} &= A_{11} - \frac{A_{21}A_{13}}{A_{23}} & \alpha_{12} &= A_{12} - \frac{A_{22}A_{13}}{A_{23}} & \alpha_{14} &= A_{14} - \frac{A_{24}A_{13}}{A_{23}} \\
\alpha_{31} &= A_{31} - \frac{A_{21}A_{33}}{A_{23}} & \alpha_{32} &= A_{32} - \frac{A_{22}A_{33}}{A_{23}} & \alpha_{34} &= A_{34} - \frac{A_{24}A_{33}}{A_{23}} \,.
\end{aligned}
$$

Das Element in der Spalte des ausgetauschten x-Elements und in der Zeile des ausgetauschten y-Elements (im Beispiel A_{23}) wird *Pivotelement* genannt; es ist im obigen Schema unterstrichen. Beim Austausch wird das Pivotelement durch sein Inverses ersetzt. Die verbleibenden Elemente in Spalte und Zeile des Pivotelements werden durch das Pivotelement geteilt, verbunden mit einem Vorzeichenwechsel in der Pivotzeile. Die restlichen Elemente werden durch das Produkt der Elemente in der Zeile und Spalte des Pivotelements modifiziert, geteilt durch das Pivotelement; diese Modifikationsregel wird *Rechteck-Regel* genannt. Das Pivotelement muß ungleich null sein. Ein kleiner Absolutwert des Pivotelements kann Rundungsfehler mit sich bringen. Daher sollte stets durch eine Pivot-Suche ein geeignetes Element mit großem Absolutwert als Pivot gewählt werden.

Regeln für den Austauschschritt. Die allgemeinen Regeln für einen Austauschschritt sind die folgenden:

1. Die Elemente außerhalb von Spalte und Zeile des Pivotelements werden nach der Rechteck-Regel transformiert: Die Rechteck-Regel wird auf die vier Elemente angewandt, die in einem Rechteck liegen, mit dem Pivotelement in der Ecke gegenüber dem Element, das ausgetauscht werden soll.
2. Die Elemente der Pivotspalte werden durch das Pivotelement geteilt.
3. Die Elemente der Pivotzeile werden durch das Pivotelement geteilt, das Vorzeichen wird umgekehrt.
4. Das Pivotelement wird durch sein Inverses ersetzt.

Im Fall von m linearen Funktionen von n Variablen erfordert der Austausch $n \times m$ Operationen. Die Austauschoperation ist *reversibel*.

Ein Programmstück (in Fortran), welches einen Austauschschritt an einer $n \times m$ Matrix ausführt, ist im Anhang (Seite 265) beschrieben. Das Element A_{pq} wird dabei als Pivotelement benutzt.

Berechnung der inversen Matrix. Im Fall einer Matrixgleichung $\boldsymbol{Ax} = \boldsymbol{y}$ für n lineare Gleichungen und n Variable wendet man n Austauschschritte auf die Matrix an, um *alle* x-Variablen durch *alle* y-Variablen auszutauschen. Die Matrix wird durch diesen Prozeß, zu dem n^3 Operationen nötig sind, *invertiert*.

Inversion durch n Austauschschritte erfordert die Auswahl von n Pivotelementen. Um den Einfluß von Rundungsfehlern zu reduzieren, sollte bei jedem Schritt das größte verbleibende Element als Pivotelement ausgewählt werden. Die Matrix ist singulär, wenn alle Kandidaten

für ein Pivotelement null sind. Sollten während der Inversion alle Elemente, die als Kandidaten in Frage kommen, klein sein, könnte die Matrix nahezu singulär sein, und weitere Austauschschritte können zu ungenauen Resultaten führen.

Beispiel 3.2 Der Austauschalgorithmus im Fall einer singulären Matrix.

Das Schema zeigt links eine quadratische Matrix mit $n = 3$ und rechts die Matrix nach zwei Austauschschritten, wobei als Pivotelemente die Elemente mit den Indizes $(1,1)$ und $(2,3)$ (unterstrichen) benutzt wurden.

$$\begin{array}{c|ccc} & x_1 & x_2 & x_3 \\ \hline y_1 = & \underline{2} & 3 & -2 \\ y_2 = & 2 & -2 & -1 \\ y_3 = & 4 & 1 & -3 \end{array} \qquad \begin{array}{c|ccc} & y_1 & x_2 & x_3 \\ \hline x_1 = & 1/2 & -3/2 & 1 \\ y_2 = & 1 & -5 & \underline{1} \\ y_3 = & 2 & -5 & 1 \end{array} \qquad \begin{array}{c|ccc} & y_1 & x_2 & y_2 \\ \hline x_1 = & -1/2 & 7/2 & 1 \\ x_3 = & -1 & 5 & 1 \\ y_3 = & 1 & \underline{0} & 1 \end{array}$$

Nach den beiden Austauschoperationen sollte das nächste Pivotelement das mit dem Indexpaar $(3,2)$ sein, der Wert dieses Elements jedoch ist null. Man sieht, daß die Matrix nicht vollständig invertiert werden kann, weil der Rang der Matrix nur 2 ist. Die letzte Zeile des Schemas ist $y_3 = y_1 + y_2$, d.h. daß y_3 eine Linearkombination von y_1 und y_2 ist.

Symmetrische Matrizen. Für symmetrische Matrizen kann der Austauschalgorithmus geringfügig modifiziert werden, so daß alle Zwischenmatrizen ebenfalls symmetrisch sind. Nahezu die Hälfte der Operationen braucht in diesem Fall nicht ausgeführt zu werden. Alle Pivotelemente müssen aus der Diagonalen ausgewählt werden. Zudem muß Regel 3 (siehe Seite 40) modifiziert werden. Das Vorzeichen der Elemente in der Pivotzeile wird *nicht* umgekehrt, und Regel 4 lautet nun: Das Pivotelement wird ersetzt durch sein Inverses mit umgekehrtem Vorzeichen. Dann bleiben alle Zwischenmatrizen und die endgültige Matrix symmetrisch; nach den n Austauschschritten müssen die Vorzeichen aller Elemente der Matrix umgekehrt werden.

In der Implementation kann nach der Pivotsuche die weitere Inversion abgebrochen werden, wenn nur noch ein Pivot mit zu kleinem Betrag gewaählt werden könnte; die entsprechenden Elemente der Matrix werden auf null gesetzt. Dadurch wird eine Untermatrix korrekt invertiert. Diese Art der Lösung ist sinnvoll, wenn zum Beispiel Korrekturen nach der Methode der kleinsten Quadrate berechnet werden; einige der Korrekturen sind dann null.

Diese Implementation (in Fortran) ist im Anhang (Seite 265) aufgeführt als Subroutine `SMSINV`.

3.4 Dreiecksmatrizen

Definition und Eigenschaften. Eine *rechte* (obere) *Dreiecksmatrix*, allgemein durch den Buchstaben $\boldsymbol{R}$ gekennzeichnet, ist eine quadratische Matrix, bei der alle Elemente unterhalb der Diagonalen null sind, d.h.

$$R_{ij} = 0 \quad \text{für} \quad i > j \qquad\qquad \boldsymbol{R} = \begin{pmatrix} R_{11} & R_{12} & R_{13} & \dots & R_{1n} \\ & R_{22} & R_{23} & \dots & R_{2n} \\ & & R_{33} & \dots & R_{3n} \\ & & & & \vdots \\ & & & & R_{nn} \end{pmatrix}.$$

Eine analoge Definition trifft für eine *linke* (untere) *Dreiecksmatrix* zu, allgemein durch den Buchstaben $\boldsymbol{L}$ beschrieben, die folgendes erfüllt:

$$L_{ij} = 0 \quad \text{für} \quad i < j \qquad\qquad \boldsymbol{L} = \begin{pmatrix} L_{11} & & & & \\ L_{21} & L_{22} & & & \\ L_{31} & L_{32} & L_{33} & & \\ \vdots & & & & \\ L_{n1} & L_{n2} & L_{n3} & \dots & L_{nn} \end{pmatrix} .$$

Ein System von Gleichungen ist leicht lösbar, wenn die Matrix des Systems eine rechte oder linke Dreiecksmatrix ist. Dies macht es vorteilhaft, eine nicht-singuläre Matrix $\boldsymbol{A}$ eines Systems von linearen Gleichungen in ein Produkt $\boldsymbol{LR}$ von linken und rechten Dreiecksmatrizen zu transformieren,

$$\boldsymbol{A} = \boldsymbol{LR} . \tag{3.19}$$

Die Transformation wird auch *Zerlegung* (Dekomposition) genannt. Die allgemeine Matrixgleichung $\boldsymbol{Ax} = \boldsymbol{b}$ mit einer Zerlegung $\boldsymbol{A} = \boldsymbol{LR}$,

$$\boldsymbol{Ax} = \boldsymbol{L}\left(\boldsymbol{Rx}\right) = \boldsymbol{b} , \tag{3.20}$$

wird in zwei Schritten gelöst unter Verwendung des Hilfsvektors $\boldsymbol{y}$, der definiert ist durch $\boldsymbol{Rx} = \boldsymbol{y}$:

$$\begin{array}{lrcll} & \boldsymbol{L}\,\boldsymbol{y} & = & \boldsymbol{b} & \text{Vorwärts-Lösung,} \\ \text{und danach} & \boldsymbol{R}\,\boldsymbol{x} & = & \boldsymbol{y} & \text{Rückwärts-Lösung.} \end{array}$$

Für diese Zerlegung gibt es unterschiedliche Methoden. Im Spezialfall von positiv-definiten symmetrischen Matrizen kann man die Cholesky-Zerlegung anwenden (siehe Kapitel 3.6) und bei allgemeinen Matrizen die Crout-Methode (siehe Kapitel 3.5).

Die inverse Matrix einer Dreiecksmatrix ist wiederum eine Dreiecksmatrix. Zudem ist für Dreiecksmatrizen die Determinante leicht als das Produkt aller Diagonalelemente zu berechnen:

$$\det \boldsymbol{R} = \prod_{i=1}^{n} R_{ii} \qquad\qquad \det \boldsymbol{L} = \prod_{i=1}^{n} L_{ii} .$$

Infolgedessen sind alle Diagonalelemente von nicht-singulären Dreiecksmatrizen *von null verschieden*, und Dreiecksmatrizen sind nicht-singulär, wenn alle Diagonalelemente von null verschieden sind.

Lösung von Gleichungssystemen mit Dreiecksmatrizen. Die Lösung nicht-singulärer Systeme mit einer linken oder rechten Dreiecksmatrix geht wie folgt. Angenommen, das lineare System $\boldsymbol{Lx} = \boldsymbol{b}$ für den Vektor $\boldsymbol{x}$ von Unbekannten soll gelöst werden:

$$\begin{pmatrix} L_{11} & & & & \\ L_{21} & L_{22} & & & \\ L_{31} & L_{32} & L_{33} & & \\ \vdots & & & & \\ L_{n1} & L_{n2} & L_{n3} & \dots & L_{nn} \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ \vdots \\ x_n \end{pmatrix} = \begin{pmatrix} b_1 \\ b_2 \\ b_3 \\ \vdots \\ b_n \end{pmatrix} .$$

Die erste Gleichung $L_{11}x_1 = b_1$ enthält nur eine Unbekannte x_1 und kann durch Division gelöst werden,

$$x_1 = \frac{b_1}{L_{11}} .$$

Die zweite Gleichung $L_{21}x_1 + L_{22}x_2 = b_2$ enthält die Komponenten x_1 und x_2. Da x_1 schon bekannt ist, kann die einzige Unbekannte x_2 bestimmt werden,

$$x_2 = \frac{b_2 - L_{21}x_1}{L_{22}} \; .$$

Jede darauf folgende Gleichung enthält nur eine zusätzliche Unbekannte. Die allgemeine Formel für ein linksseitiges Dreieckssystem ist

$$x_i = \frac{b_i - \sum_{j=1}^{i-1} L_{ij}x_j}{L_{ii}} \qquad i = 1, 2, \ldots n \; , \tag{3.21}$$

falls alle $L_{ii} \neq 0$ sind. Dieser Lösungsprozeß mit einer *linken* Dreiecksmatrix wird *Vorwärts-Lösung* genannt (sie beginnt mit x_1). Die Lösung erfordert etwa $n^2/2$ Multiplikationen und Additionen und n Divisionen.

Die Lösung eines linearen Systems mit einer *rechten* Dreiecksmatrix wird *Rückwärts-Lösung* genannt. In diesem Fall ist x_n die erste zu bestimmende Unbekannte, die anderen Unbekannten werden in der Reihenfolge $x_{n-1}, \ldots x_2$, x_1 bestimmt, vorausgesetzt alle $R_{ii} \neq 0$. Die allgemeine Formel für ein quadratisches rechtsseitiges Dreieckssystem ist:

$$x_i = \frac{b_i - \sum_{j=i+1}^{n} R_{ij}x_j}{R_{ii}} \qquad i = n, \ldots 2, 1 \; . \tag{3.22}$$

Inversion von Dreiecksmatrizen. Zu einer nicht-singulären linksseitigen Dreiecksmatrix $\boldsymbol{L}$ (mit $L_{ij} = 0 \quad \text{für} \quad i < j$) gibt es eine inverse Matrix $\boldsymbol{M}$, die ebenfalls linksseitig dreieckig ist (mit $M_{jk} = 0 \quad \text{für} \quad j < k$). Die Elemente der inversen Matrix müssen folgendes erfüllen

$$\sum_{j=1}^{n} L_{ij}M_{jk} = \begin{cases} 1 & \text{for} \quad i = k \\ 0 & \text{for} \quad i \neq k \end{cases} \; .$$

Die Inversion ist ein expliziter Prozeß. Für $i = 1$ wird die Gleichung $L_{11}M_{11} = 1$ durch $M_{11} = 1/L_{11}$ gelöst. Für $i = 2$ gibt es zwei Gleichungen

$$\begin{aligned} L_{21}M_{11} + L_{22}M_{21} &= 0 \\ L_{22}M_{22} &= 1 \; , \end{aligned}$$

die durch

$$M_{21} = -\frac{L_{21}M_{11}}{L_{22}} \qquad M_{22} = \frac{1}{L_{22}}$$

gelöst werden, usw. Die allgemeinen Formeln für die Inversion sind für $i = 1, 2, \ldots \; n$

$$M_{ik} = -\frac{\sum_{j=k}^{i-1} L_{ij}M_{jk}}{L_{ii}} \quad k < i \qquad\qquad M_{ii} = \frac{1}{L_{ii}} \; . \tag{3.23}$$

Formeln für die Inversion einer rechtsseitigen Dreiecksmatrix $\boldsymbol{R}$ lauten analog.

3.5 Allgemeine LR-Zerlegung

Die Crout-Methode ist eine Dreieckszerlegung der $n \times n$ Matrix $\boldsymbol{A}$ nach $\boldsymbol{A} = \boldsymbol{LR}$, wobei $\boldsymbol{L}$ eine linke Dreiecksmatrix mit Diagonalelementen gleich 1 ist und $\boldsymbol{R}$ eine rechte Dreiecksmatrix:

$$\boldsymbol{L} = \begin{pmatrix} 1 & & & & \\ L_{21} & 1 & & & \\ L_{31} & L_{32} & 1 & & \\ \cdots & & & & \\ \cdots & & & & \\ L_{n1} & L_{n2} & L_{n3} & \cdots & 1 \end{pmatrix} \qquad \boldsymbol{R} = \begin{pmatrix} R_{11} & R_{12} & R_{13} & \cdots & R_{1n} \\ & R_{22} & R_{23} & \cdots & R_{2n} \\ & & R_{33} & \cdots & R_{3n} \\ & & & & \vdots \\ & & & & R_{nn} \end{pmatrix} . \tag{3.24}$$

Ein Vergleich der ersten Zeile der Matrix $\boldsymbol{A}$ mit der entsprechenden Zeile der Produktmatrix liefert unmittelbar $A_{11} = R_{11}$, $A_{12} = R_{12}$, $\ldots A_{1n} = R_{1n}$, womit also schon die Elemente der ersten Zeile der Matrix $\boldsymbol{R}$ bestimmt sind. Ein Vergleich der ersten Spalte zeigt: $A_{21} = L_{21}R_{11}$, $A_{31} = L_{31}R_{11}$, $\ldots A_{n1} = L_{n1}R_{11}$, und das erlaubt, die Elemente der ersten Spalte der Matrix $\boldsymbol{L}$ zu bestimmen. Allgemeine Formeln folgen für $k = 1, 2, \ldots n$ aus den alternativen Vergleichen der Zeilen

$$R_{ik} = A_{ik} - \sum_{j=1}^{i-1} L_{ij}R_{jk} \qquad i = 1, 2, \ldots k$$

und der Spalten

$$L_{ik} = \frac{1}{R_{kk}} \left[A_{ik} - \sum_{j=1}^{k-1} L_{ij}R_{jk} \right] \qquad i = k,\, k+1, \ldots n\ .$$

Alle n^2 Elemente der Matrizen $\boldsymbol{L}$ und $\boldsymbol{R}$ werden aus den n^2 Elementen der Matrix $\boldsymbol{A}$ ermittelt. Beide Matrizen $\boldsymbol{L}$ und $\boldsymbol{R}$ können am Ort der Matrix $\boldsymbol{A}$ gespeichert werden, zum Beispiel sieht für $n = 4$ das Speicherschema so aus:

$$[\boldsymbol{A}] = \begin{pmatrix} \texttt{R(1,1)} & \texttt{R(1,2)} & \texttt{R(1,3)} & \texttt{R(1,4)} \\ \texttt{L(2,1)} & \texttt{R(2,2)} & \texttt{R(2,3)} & \texttt{R(2,4)} \\ \texttt{L(3,1)} & \texttt{L(3,2)} & \texttt{R(3,3)} & \texttt{R(3,4)} \\ \texttt{L(4,1)} & \texttt{L(4,2)} & \texttt{L(4,3)} & \texttt{R(4,4)} \end{pmatrix} .$$

Die teilweise Pivotsuche mit Vertauschen von Zeilen untereinander läßt sich leicht durchführen; dies ist nötig, um die Methode stabil zu halten. Das Ergebnis ist somit die Zerlegung einer zeilenweisen Permutation von $\boldsymbol{A}$. In jeder Spalte wird das größte Element unter den Kandidaten in der Spalte, durch Austauschen der Zeilen untereinander, als das Pivotelement R_{kk} verwendet.

Die LR-Zerlegung einer allgemeinen Matrix wird durch die im Anhang (Seite 267) aufgeführte subroutine `GNLRD` *geleistet. Die Lösung eines Gleichungssystem wird durch die subroutine* `GNSOL` *(Seite 267) geleistet. Subroutine* `GNINV` *(Seite 268) berechnet die inverse matrix aus der Zerlegung. Im Anhang (Seite 269) wird das Beispiel einer Matrix inversion gezeigt.*

Bandmatrix. Eine symmetrische Bandmatrix kann, wie auf Seite 265 beschrieben, platzsparend gespeichert werden. Die LR-Zerlegung führt wieder auf Bandmatrizen, die die ursprünglichen Elemente überschreiben können. Damit wird die kompakte Lösung eines Gleichungssystems möglich.

Das Unterprogram `BMSOL` *löst die Gleichung $\boldsymbol{Wx} = \boldsymbol{b}$, wobei $\boldsymbol{W}$ eine symmetrische positiv-semidefinite Bandmatrix der Breite* `NBAND` *ist. Diese Routine ist im Anhang (Seite 269) aufgeführt.*

3.6 Cholesky-Zerlegung

Symmetrische Zerlegung einer symmetrischen Matrix. In vielen Anwendungen, besonders in statistischen Problemen, müssen Systeme von Gleichungen $\boldsymbol{W}\boldsymbol{x} = \boldsymbol{b}$ mit einer *positiv-definiten symmetrischen* Matrix $\boldsymbol{W}$ für einen Vektor $\boldsymbol{x}$ von Unbekannten und einen gegebenen Vektor $\boldsymbol{b}$ gelöst werden. Eine positiv-definite symmetrische Matrix $\boldsymbol{W}$ kann folgendermaßen ausgedrückt werden:

$$\boldsymbol{W} = \boldsymbol{R}^T\boldsymbol{R} \tag{3.25}$$

mit einer rechten Dreiecksmatrix $\boldsymbol{R}$, die Choleskyfaktor oder *Quadratwurzel* der Matrix $\boldsymbol{W}$ genannt wird. Die Matrix $\boldsymbol{R}^T$ ist eine linke Dreiecksmatrix; also ist die Cholesky-Zerlegung eine *symmetrische* $\boldsymbol{LR}$-Zerlegung der Matrix $\boldsymbol{W}$.

Die Choleskyfaktoren können direkt aus dem Produkt $\boldsymbol{R}^T\boldsymbol{R}$ durch elementeweises Vergleichen bestimmt werden. In der ersten Zeile des Produkts erhält man die Gleichungen

$$W_{11} = R_{11}^2 \qquad W_{12} = R_{11}R_{12} \qquad \ldots \qquad W_{1n} = R_{11}R_{1n}\,.$$

Für das Element R_{11} benötigt man eine Quadratwurzel: $R_{11} = \sqrt{W_{11}}$; die anderen Elemente folgen nach dem Bestimmen von R_{11} durch Division:

$$R_{12} = \frac{W_{12}}{R_{11}} \qquad R_{13} = \frac{W_{13}}{R_{11}} \qquad \ldots \qquad R_{1n} = \frac{W_{1n}}{R_{11}}\,.$$

Die zweite Zeile der Produktmatrix enthält Gleichungen wie $W_{22} = R_{12}^2 + R_{22}^2$, die über $R_{22} = \sqrt{W_{22} - R_{12}^2}$ mit Hilfe des schon bekannten Wertes R_{12} gelöst werden. Die weiteren Rechnungen erfolgen nach der Reduktionsformel unten.

Die Matrix $\boldsymbol{W}$ wird schrittweise in die Matrix $\boldsymbol{R}$ transformiert. Man beginnt mit $\boldsymbol{W}^{(0)} = \boldsymbol{W}$ und setzt für $k = 1, 2, \ldots (n-1)$

$$W_{ij}^{(k)} = W_{ij}^{(k-1)} - R_{ki}R_{kj} \qquad i, j = k+1, k+2, \ldots n\,. \tag{3.26}$$

Zuerst ist $k = 1$ und die Elemente mit $i, j = 2, 3, \ldots n$ werden transformiert, dann ist $k = 2$ und die Elemente mit $i, j = 3, 4, \ldots n$ werden transformiert und so weiter mit Hilfe der Formeln

$$R_{ii} = \sqrt{W_{ii}^{(i-1)}} \qquad R_{ij} = \frac{W_{ij}^{(i-1)}}{R_{ii}} \quad \text{für} \quad j > i\,. \tag{3.27}$$

Die Beziehung

$$W_{jj} = \sum_{i=1}^{j} R_{ij}^2 \qquad j = 1, 2, \ldots n$$

zeigt, daß die Elemente des Choleskyfaktors $\boldsymbol{R}$ automatisch durch die positiven Diagonalelemente der Matrix $\boldsymbol{W}$ begrenzt werden.

Ein symmetrisches Speicherschema kann für die Matrizen $\boldsymbol{W}$ und $\boldsymbol{R}$ (mit $n(n+1)/2$ Elementen) benutzt werden, und zwar in demselben Speicherbereich für die Matrizen $\boldsymbol{R}$ und $\boldsymbol{W}$; die Matrix $\boldsymbol{W}$ wird während der Transformation zerstört. Insgesamt verlangt die Cholesky-Zerlegung n Quadratwurzeln und $1/6(n^3 + 3n^2 - 4n)$ Operationen.

Lösung eines Gleichungssystems. Die Lösung des Gleichungssystems $W\boldsymbol{x} = \boldsymbol{b}$ erfolgt gemäß der Cholesky-Zerlegung $\boldsymbol{R}^T\boldsymbol{R}\boldsymbol{x} = \boldsymbol{b}$ in zwei Schritten, indem man den durch $\boldsymbol{R}\boldsymbol{x} = \boldsymbol{y}$ definierten Hilfsvektor $\boldsymbol{y}$ einsetzt:

$$\boldsymbol{R}^T\boldsymbol{y} = \boldsymbol{b} \qquad \text{Vorwärts-Lösung,} \tag{3.28}$$

$$\text{und dann} \quad \boldsymbol{R}\,\boldsymbol{x} = \boldsymbol{y} \qquad \text{Rückwärts-Lösung.} \tag{3.29}$$

Die Cholesky-Zerlegung für eine positiv-definite symmetrische Matrix ist in den Unterprogrammen `CHLRD` *und* `CHSOL` *(in Fortran) im Anhang implementiert.*

Berechnung der inversen Matrix. Die inverse Matrix $\boldsymbol{V} = \boldsymbol{W}^{-1}$ kann anschließend über den inversen Choleskyfaktor $\boldsymbol{R}$ berechnet werden,

$$\boldsymbol{W}^{-1} = \left(\boldsymbol{R}^T\boldsymbol{R}\right)^{-1} = \boldsymbol{R}^{-1}\left(\boldsymbol{R}^T\right)^{-1} = \boldsymbol{R}^{-1}\left(\boldsymbol{R}^{-1}\right)^T ,$$

und das erfordert die inverse Matrix $\boldsymbol{S} = \boldsymbol{R}^{-1}$, die ebenfalls eine rechte Dreiecksmatrix ist. Die Elemente der inversen Matrix $\boldsymbol{V} = \boldsymbol{W}^{-1} = \boldsymbol{S}\boldsymbol{S}^T$ sind für $k = 1, 2, \ldots n$:

$$V_{ik} = \sum_{j=k}^{n} S_{ij}S_{kj} \quad \text{für} \quad i \le k \qquad V_{kk} = \sum_{j=k}^{n} S_{kj}^2 \,. \tag{3.30}$$

Die letztere Formel zeigt, daß es möglich ist, einzelne Diagonalelemente der inversen Matrix $\boldsymbol{V} = \boldsymbol{W}^{-1}$ aus der Matrix $\boldsymbol{S}$ zu berechnen. Das kann für statistische Anwendungen wichtig sein, in denen die invertierte Matrix eine Kovarianz-Matrix ist und nur bestimmte Varianzen (Diagonalelemente) gesucht werden.

Quadratische Form. Die symmetrische Matrix $\boldsymbol{W}$ ist *positiv-definit*, wenn die quadratische Form

$$Q = \boldsymbol{x}^T\boldsymbol{W}\boldsymbol{x} = \sum_{i=1}^{n}\sum_{i=k}^{n} W_{ik}x_i x_k$$

einen positiven Wert für jeden Vektor $\boldsymbol{x} \neq \boldsymbol{0}$ annimmt:

$$Q = \boldsymbol{x}^T\boldsymbol{W}\boldsymbol{x} > 0.$$

Benutzt man die Choleskyfaktoren ($\boldsymbol{W} = \boldsymbol{R}^T\boldsymbol{R}$), so kann die quadratische Form als eine Summe von Quadraten unter Verwendung von $\boldsymbol{y} = \boldsymbol{R}\boldsymbol{x}$ geschrieben werden:

$$Q = \boldsymbol{x}^T\boldsymbol{R}^T\boldsymbol{R}\boldsymbol{x} = \boldsymbol{y}^T\boldsymbol{y}$$

Cholesky-Zerlegung ohne Quadratwurzel. Es gibt eine Variante der Cholesky-Zerlegung mit einer Diagonalmatrix $\boldsymbol{D}$, die Quadratwurzeln vermeidet. Diese Zerlegung ist von der Form

$$\boldsymbol{W} = \boldsymbol{L}\boldsymbol{D}\boldsymbol{L}^{\mathrm{T}} , \tag{3.31}$$

wobei die Matrix $\boldsymbol{L}$ eine linke Dreiecksmatrix mit den Werten 1 auf der Diagonale ist. Statt diese Werte 1 zu speichern, können die Diagonalelemente $\boldsymbol{D}_{ii}$ in der Diagonalen von $\boldsymbol{L}$ gespeichert werden. Damit können die Matrizen $\boldsymbol{L}$ und $\boldsymbol{D}$ auf dem Platz der Matrix $\boldsymbol{W}$ gespeichert werden.

Die Zerlegung erlaubt es die Matrixgleichung $W\boldsymbol{x} = \boldsymbol{b}$ in der Form

$$\boldsymbol{L}\left(\boldsymbol{D}\boldsymbol{L}^{\mathrm{T}}\boldsymbol{x}\right) = \boldsymbol{b}$$

zu schreiben. Die Lösung des Gleichungssystems erfolgt in zwei Schritten, indem man den durch $\boldsymbol{D}\boldsymbol{L}^{\mathrm{T}}\boldsymbol{x} = \boldsymbol{y}$ definierten Hilfsvektor $\boldsymbol{y}$ einsetzt:

$$\begin{aligned} \boldsymbol{L}\boldsymbol{y} &= \boldsymbol{b} && \text{Vorwärts-Lösung,} && (3.32)\\ \text{und dann} \quad \boldsymbol{L}^{\mathrm{T}}\boldsymbol{x} &= \boldsymbol{D}^{-1}\boldsymbol{y} && \text{Rückwärts-Lösung.} && (3.33) \end{aligned}$$

Die Lösung ist einfach, weil $\boldsymbol{L}$ eine Dreiecksmatrix ist. Die Zahl der Operationen ist ähnlich zu der beim Gauß Algorithmus.

Bandmatrizen. Die Verwendung der Cholesky-Zerlegung ist besonders geeignet, wenn die symmetrische und positiv-definite Matrix $\boldsymbol{W}$ eine Bandstruktur hat. Bei der normalen Matrixinversion geht die Bandstruktur verloren. Bei der Cholesky-Zerlegung in beiden Varianten bleibt die Bandstruktur erhalten. Die Zerlegung kann auch bei der platzsparenden Speicherung einer symmetrischen Bandmatrix $\boldsymbol{W}$ (siehe Seite 265) die ursprünglichen Elemente überschreiben. Während die Zahl der Rechenoperationen zur Lösung eines Gleichungssystems bei einer vollen Matrix proportional zu n^3 ist, ist die Zahl bei der Cholesky-Zerlegung einer Bandmatrix mit Bandbreite m nur proportional zu $m^2 n$, und damit linear von der Matrixdimension n abhängig. Die Rechenarbeit wird damit besonders im Fall einer kleinen Bandbreite m wesentlich reduziert.

Elemente der Inversen einer Bandmatrix. Oft wird von einer symmetrischen Bandmatrix nicht die volle Inverse benötigt, sondern nur die Diagonalelement oder die Elemente im Band. Die schnelle Berechnung dieser Elemente ist möglich mit einem speziellen Algorithmus[35, 46].

Die Bandmatrix in der symmetrischen Zerlegung (3.31) der Matrix $\boldsymbol{W}$ wird betrachtet; die Matrix $\boldsymbol{L}$ ist eine Linksdreiecksmatrix mit Diagonalelmenten 1, und $\boldsymbol{D}$ ist diagonal. Für die inverse Matrix $\boldsymbol{Z} = \boldsymbol{W}^{-1}$ gilt:

$$\boldsymbol{Z} = \left(\boldsymbol{L}^{\mathrm{T}}\right)^{-1}\boldsymbol{D}^{-1} + \boldsymbol{Z}\left(\boldsymbol{I} - \boldsymbol{L}\right) \ . \tag{3.34}$$

Weil die Differenz $(\boldsymbol{I} - \boldsymbol{L})$ eine Linksdreiecksmatrix is (Diagonalelemente sind Null), gelten die folgenden Relationen:

$$Z_{ij} = -\sum_{k=j+1}^{n} Z_{ik}\, L_{kj} \qquad\qquad i > j \tag{3.35}$$

$$Z_{ii} = d_{ii}^{-1} - \sum_{k=i+1}^{n} Z_{ik}\, L_{ki} \tag{3.36}$$

Alle Elemente von $\boldsymbol{Z}$, die für die Berechnung von Z_{ij} notwendig sind, sind die Eelemente Z_{ik} in Zeile i mit $k > j+1$ und $L_{kj} \neq 0$. Damit können die Elemente von $\boldsymbol{Z}$, beginnend mit Z_{nn}, in umgekehrter Reihenfolge berechnet werden; alle notwendigen Elemente von $\boldsymbol{Z}$ zur Berechnung von Z_{ij} sind dann bereits berechnet. Sämtliche Elemente von $\boldsymbol{Z}$ können nach den Formeln (3.35), (3.36) berechnet werden, falls erforderlich. Insbesondere können alle Elemente aus dem Band berechnet werden, ohne ein Element außerhalb des Bandes zu benötigen. Die Zahl der Operationen zur Berechnung der Elemente des Bandes is daher proportional zu n, und nicht zu n^3.

3.7 Inversion durch Partitionierung

Eine Matrix $\boldsymbol{A}$, die invertiert werden soll, hat manchmal eine besondere Struktur, die eine signifikante Reduktion des Rechenaufwands bei der Inversion erlaubt. Wichtige Beispiele für solche speziellen Strukturen sind Fälle, in denen eine Untermatrix diagonal oder die Inverse

einer Untermatrix schon bekannt ist. In diesen Fällen kann das Problem der Inversion durch Partitionierung effektiv gelöst werden.

Eine quadratische $n \times n$ Matrix $\boldsymbol{A}$ kann in Untermatrizen partitioniert werden

$$\boldsymbol{A} = \left(\begin{array}{c|c} \boldsymbol{A}_{11} & \boldsymbol{A}_{12} \\ \hline \boldsymbol{A}_{21} & \boldsymbol{A}_{22} \end{array}\right) , \tag{3.37}$$

wobei die Untermatrix $\boldsymbol{A}_{11}$ eine $p \times p$ Quadratmatrix ist und die Untermatrix $\boldsymbol{A}_{22}$ eine $q \times q$ Quadratmatrix mit $p + q = n$. Die Werte $p = 1$ oder $q = 1$ sind möglich, die Untermatrizen $\boldsymbol{A}_{12}$ und $\boldsymbol{A}_{21}$ sind in diesem Fall Vektoren.

Die inverse Matrix $\boldsymbol{B} = \boldsymbol{A}^{-1}$ kann auf dieselbe Weise partitioniert werden:

$$\boldsymbol{B} = \left(\begin{array}{c|c} \boldsymbol{B}_{11} & \boldsymbol{B}_{12} \\ \hline \boldsymbol{B}_{21} & \boldsymbol{B}_{22} \end{array}\right) . \tag{3.38}$$

Das Matrixprodukt $\boldsymbol{AB}$ (mit $\boldsymbol{AB} = \boldsymbol{BA} = \boldsymbol{I}$) ist, ausgedrückt durch die Untermatrizen, durch die folgenden vier Matrixgleichungen gegeben:

$$\begin{aligned} \boldsymbol{A}_{11}\,\boldsymbol{B}_{11} + \boldsymbol{A}_{12}\,\boldsymbol{B}_{21} &= \boldsymbol{I} \\ \boldsymbol{A}_{11}\,\boldsymbol{B}_{12} + \boldsymbol{A}_{12}\,\boldsymbol{B}_{22} &= \boldsymbol{0} \\ \boldsymbol{A}_{21}\,\boldsymbol{B}_{11} + \boldsymbol{A}_{22}\,\boldsymbol{B}_{21} &= \boldsymbol{0} \\ \boldsymbol{A}_{21}\,\boldsymbol{B}_{12} + \boldsymbol{A}_{22}\,\boldsymbol{B}_{22} &= \boldsymbol{I} \end{aligned} \tag{3.39}$$

Durch Multiplikation der dritten Gleichung mit $\boldsymbol{A}_{12} \cdot \boldsymbol{A}_{22}^{-1}$ von links erhält man

$$\boldsymbol{A}_{12}\,\boldsymbol{A}_{22}^{-1}\,\boldsymbol{A}_{21}\,\boldsymbol{B}_{11} + \boldsymbol{A}_{12}\,\boldsymbol{A}_{22}^{-1}\,\boldsymbol{A}_{22}\,\boldsymbol{B}_{21} = \boldsymbol{0}$$

oder

$$\boldsymbol{A}_{12}\,\boldsymbol{A}_{22}^{-1}\,\boldsymbol{A}_{21}\,\boldsymbol{B}_{11} + \boldsymbol{A}_{12}\,\boldsymbol{B}_{21} = \boldsymbol{0}$$

und das ergibt, wenn man es von der ersten Gleichung (3.39) subtrahiert

$$\boldsymbol{A}_{11}\boldsymbol{B}_{11} - \boldsymbol{A}_{12}\boldsymbol{A}_{22}^{-1}\boldsymbol{A}_{21}\boldsymbol{B}_{11} = \left(\boldsymbol{A}_{11} - \boldsymbol{A}_{12}\boldsymbol{A}_{22}^{-1}\boldsymbol{A}_{21}\right)\boldsymbol{B}_{11} = \boldsymbol{I} .$$

Die letzte Gleichung kann für $\boldsymbol{B}_{11}$ durch Matrixinversion einer $p \times p$ Matrix gelöst werden:

$$\boldsymbol{B}_{11} = \left(\boldsymbol{A}_{11} - \boldsymbol{A}_{12}\boldsymbol{A}_{22}^{-1}\boldsymbol{A}_{21}\right)^{-1} .$$

Ist nun die Untermatrix $\boldsymbol{B}_{11}$ durch Matrixinversion bestimmt, kann die Untermatrix $\boldsymbol{B}_{21}$ aus der dritten Gleichung (3.39) durch Matrixmultiplikation ermittelt werden:

$$\boldsymbol{B}_{21} = -\boldsymbol{A}_{22}^{-1}\boldsymbol{A}_{21}\boldsymbol{B}_{11} . \tag{3.40}$$

Die verbleibenden Untermatrizen $\boldsymbol{B}_{12}$ und $\boldsymbol{B}_{22}$ lassen sich aus der zweiten und vierten Gleichung (3.39) bestimmen:

$$\begin{aligned} \boldsymbol{B}_{22} &= \left(\boldsymbol{A}_{22} - \boldsymbol{A}_{21}\boldsymbol{A}_{11}^{-1}\boldsymbol{A}_{12}\right)^{-1} \\ \boldsymbol{B}_{12} &= -\boldsymbol{A}_{11}^{-1}\boldsymbol{A}_{12}\boldsymbol{B}_{22} . \end{aligned} \tag{3.41}$$

Diese Formeln erfordern nur die Inversion kleinerer Matrizen der Größe $p \times p$ und $q \times q$.
Die obigen Formeln zur Berechnung von Untermatrizen können so modifiziert werden, daß nur zwei Matrizen der Ordnung p und q invertiert werden müssen. Wenn $\boldsymbol{A}_{11}^{-1}$ bekannt ist, dann kann $\boldsymbol{B}_{11}$ entsprechend der ersten Gleichung (3.39) so ausgedrückt werden

$$\boldsymbol{B}_{11} = \boldsymbol{A}_{11}^{-1} - \boldsymbol{A}_{11}^{-1}\,\boldsymbol{A}_{12}\,\boldsymbol{B}_{21}\,, \tag{3.42}$$

und das ergibt, wenn man es in die dritte Gleichung einsetzt

$$\boldsymbol{A}_{21}\,\boldsymbol{A}_{11}^{-1} - \boldsymbol{A}_{21}\,\boldsymbol{A}_{11}^{-1}\,\boldsymbol{A}_{12}\,\boldsymbol{B}_{21} + \boldsymbol{A}_{22}\,\boldsymbol{B}_{21} = \boldsymbol{0}\,,$$

und mit Gleichung (3.41)

$$\boldsymbol{A}_{21}\,\boldsymbol{A}_{11}^{-1} + \underbrace{\left(\boldsymbol{A}_{22} - \boldsymbol{A}_{21}\,\boldsymbol{A}_{11}^{-1}\,\boldsymbol{A}_{12}\right)}_{\boldsymbol{B}_{22}^{-1}}\boldsymbol{B}_{21} = \boldsymbol{0}\,,$$

und das löst man durch

$$\boldsymbol{B}_{21} = -\boldsymbol{B}_{22}\boldsymbol{A}_{21}\boldsymbol{A}_{11}^{-1}\,. \tag{3.43}$$

Die Gleichungen für die Untermatrizen der inversen Matrix $\boldsymbol{B}$ lauten insgesamt in der Reihenfolge der Berechnung

$$\begin{aligned}
\boldsymbol{B}_{22} &= \left(\boldsymbol{A}_{22} - \boldsymbol{A}_{21}\boldsymbol{A}_{11}^{-1}\boldsymbol{A}_{12}\right)^{-1} \\
\boldsymbol{B}_{21} &= -\boldsymbol{B}_{22}\boldsymbol{A}_{21}\boldsymbol{A}_{11}^{-1} \\
\boldsymbol{B}_{12} &= -\boldsymbol{A}_{11}^{-1}\boldsymbol{A}_{12}\boldsymbol{B}_{22} \\
\boldsymbol{B}_{11} &= \boldsymbol{A}_{11}^{-1} - \boldsymbol{A}_{11}^{-1}\boldsymbol{A}_{12}\boldsymbol{B}_{21}\,.
\end{aligned} \tag{3.44}$$

In diesem Satz von Formeln wird die inverse $p \times p$ Matrix $\boldsymbol{A}_{11}^{-1}$ benutzt, während die ursprüngliche Matrix $\boldsymbol{A}_{11}$ nicht direkt vorkommt. Nur eine weitere Matrixinversion wird verlangt, und zwar bei der Berechnung der $q \times q$ Matrix $\boldsymbol{B}_{22}$.
Einen weiteren äquivalenten Satz von Gleichungen erhält man, wenn man die Indizes 1 und 2 vertauscht:

$$\begin{aligned}
\boldsymbol{B}_{11} &= \left(\boldsymbol{A}_{11} - \boldsymbol{A}_{12}\boldsymbol{A}_{22}^{-1}\boldsymbol{A}_{21}\right)^{-1} \\
\boldsymbol{B}_{12} &= -\boldsymbol{B}_{11}\boldsymbol{A}_{12}\boldsymbol{A}_{22}^{-1} \\
\boldsymbol{B}_{21} &= -\boldsymbol{A}_{22}^{-1}\boldsymbol{A}_{21}\boldsymbol{B}_{11} \\
\boldsymbol{B}_{22} &= \boldsymbol{A}_{22}^{-1} - \boldsymbol{A}_{22}^{-1}\boldsymbol{A}_{21}\boldsymbol{B}_{12}\,.
\end{aligned} \tag{3.45}$$

Im Fall *symmetrischer Matrizen* $\boldsymbol{A}$ und $\boldsymbol{B}$ vereinfacht sich die Berechnung weiter, weil

$$\boldsymbol{A}_{12} = (\boldsymbol{A}_{21})^T \qquad \text{und} \qquad \boldsymbol{B}_{12} = (\boldsymbol{B}_{21})^T$$

gilt.
Inversion durch Rändern. Partitionieren im Spezialfall $q = 1$ heißt rändern. Die $n \times n$ Matrix $\boldsymbol{A}_n$ hat dann die Gestalt

$$\boldsymbol{A}_n = \left(\begin{array}{cccc|c} a_{11} & a_{12} & \dots & a_{1,n-1} & a_{1n} \\ a_{21} & a_{22} & \dots & a_{2,n-1} & a_{2n} \\ \dots & & & & \dots \\ a_{n-1,1} & a_{n-1,2} & \dots & a_{n-1,n-1} & a_{n-1,n} \\ \hline a_{n1} & a_{n2} & \dots & a_{n,n-1} & a_{nn} \end{array}\right) = \left(\begin{array}{c|c} \boldsymbol{A}_{n-1} & \boldsymbol{u} \\ \hline \boldsymbol{v}^T & a_{nn} \end{array}\right)$$

mit zwei $(n-1)$-Vektoren $\boldsymbol{u}$ und $\boldsymbol{v}$ und einem Skalar a_{nn}. Wenn die Matrix $\boldsymbol{A}_{n-1}^{-1}$ schon bekannt oder leicht zu berechnen ist (wenn sie beispielsweise diagonal ist), dann kann man die Formeln von Gleichung (3.44) benutzen und braucht nur die Inversion einer 1×1 Matrix, d.h. eines Skalars:

$$\alpha_n = a_{nn} - \boldsymbol{v}^T \boldsymbol{A}_{n-1}^{-1} \boldsymbol{u} \ ,$$

wobei α_n der Matrix $\boldsymbol{B}_{22}^{-1}$ entspricht (siehe Gleichung (3.41)). Die anderen Untermatrizen der inversen Matrix werden anschließend als Matrixprodukte berechnet:

$$\boldsymbol{A}_n^{-1} = \left(\begin{array}{c|c} \boldsymbol{A}_{n-1}^{-1} + \dfrac{\boldsymbol{A}_{n-1}^{-1} \boldsymbol{u} \boldsymbol{v}^T \boldsymbol{A}_{n-1}^{-1}}{\alpha_n} & -\dfrac{\boldsymbol{A}_{n-1}^{-1} \boldsymbol{u}}{\alpha_n} \\ \hline -\dfrac{\boldsymbol{v}^T \boldsymbol{A}_{n-1}^{-1}}{\alpha_n} & \dfrac{1}{\alpha_n} \end{array} \right) . \tag{3.46}$$

Die Berechnung vereinfacht sich weiter im Fall einer symmetrischen Matrix, da die Vektoren $\boldsymbol{u}$ und $\boldsymbol{v}$ dann identisch sind.

Im Prinzip könnte man eine Inversionsmethode insgesamt auf dem Rändern aufbauen, indem man mit einer 1×1 Matrix beginnt und die Matrixgröße in Schritten von 1 wachsen läßt. Verglichen mit anderen Methoden haben bei dieser Methode jedoch Rundungsfehler einen größeren Einfluß auf das Ergebnis. Der Gebrauch dieser Methode sollte also auf die oben erwähnten Fälle beschränkt bleiben.

3.8 Diagonalisierung symmetrischer Matrizen

In diesem Kapitel werden *reelle symmetrische* Matrizen $\boldsymbol{V}$ betrachtet; ihre Eigenwerte sind *reell*. Die numerische Bestimmung der Eigenwerte aus den Nullstellen des charakteristischen Polynoms (siehe Kapitel 3.1) ist in der Regel ungenau. Eigenwerte *und* Eigenvektoren können durch eine Transformation mit orthogonalen Matrizen bestimmt werden.

Eine Matrix $\boldsymbol{U}$ heißt *orthogonal*, wenn gilt:

$$\boldsymbol{U}^T \boldsymbol{U} = \boldsymbol{D} \ , \qquad \boldsymbol{D} = \text{Diagonalmatrix},$$

und das bedeutet, daß die Spaltenvektoren der Matrix $\boldsymbol{U}$ orthogonal zueinander sind. Die Matrix $\boldsymbol{U}$ heißt *orthonormal*, wenn die Diagonalmatrix $\boldsymbol{D}$ eine Einheitsmatrix ist:

$$\boldsymbol{U}^T \boldsymbol{U} = \boldsymbol{U} \boldsymbol{U}^T = \boldsymbol{I} \ .$$

Wenn die Zeilenvektoren einer quadratischen Matrix orthogonal und normiert sind, dann sind die Spaltenvektoren ebenfalls orthogonal und normiert.

Zu einer reellen symmetrischen Matrix $\boldsymbol{V}$ kann eine orthonormale Matrix $\boldsymbol{U}$ konstruiert werden, so daß die Matrix $\boldsymbol{V}$ in eine diagonale Matrix übergeführt wird (Diagonalisierung).

$$\boldsymbol{U}^T \boldsymbol{V} \boldsymbol{U} = \boldsymbol{D} \quad \text{und} \quad \boldsymbol{V} = \boldsymbol{U} \boldsymbol{D} \boldsymbol{U}^T \tag{3.47}$$

mit den quadratischen Matrizen $\boldsymbol{V}$, $\boldsymbol{U}$ und $\boldsymbol{D}$, wie oben beschrieben.

$$\boxed{\boldsymbol{V}} = \boxed{\boldsymbol{U}} \times \boxed{\boldsymbol{D}} \times \boxed{\boldsymbol{U}^T}$$

Die Diagonalelemente D_{11}, $D_{22} \ldots D_{nn}$ der Diagonalmatrix $\boldsymbol{D}$ sind die Eigenwerte der Matrix $\boldsymbol{V}$, und die Spaltenvektoren der orthogonalen Matrix $\boldsymbol{U}$ sind die entsprechenden Eigenvektoren.

Die quadratische Form $Q = \boldsymbol{x}^T \boldsymbol{V} \boldsymbol{x}$ des transformierten Vektors $\boldsymbol{y} = \boldsymbol{U}^T \boldsymbol{x}$ kann folgendermaßen geschrieben werden

$$Q = \boldsymbol{x}^T \boldsymbol{V}\, \boldsymbol{x} = \boldsymbol{x}^T \boldsymbol{U} \boldsymbol{D} \boldsymbol{U}^T \boldsymbol{x} = \boldsymbol{y}^T \boldsymbol{D} \boldsymbol{y} = \sum_{i=1}^{n} D_{ii} y_i^2 \,, \tag{3.48}$$

und für alle nicht-verschwindenden $\boldsymbol{y}$ ist $Q > 0$, falls alle Diagonalelemente $D_{ii} > 0$ sind. Die Matrix $\boldsymbol{V}$ wird in diesem Fall *positiv-definit* genannt.

Drehung. Die einfachsten nicht-trivialen orthogonalen Matrizen haben die Form einer Quadratmatrix $\boldsymbol{U}$ mit den Elementen $U_{ii} = 1$ für $i \neq p,\, q$ und den speziellen Elementen

$$\begin{aligned} U_{pp} &= \cos\varphi & U_{pq} &= \sin\varphi \\ U_{qp} &= -\sin\varphi & U_{qq} &= \cos\varphi \,, \end{aligned}$$

die restlichen Elemente sind 0

$$\boldsymbol{U} = \begin{pmatrix} 1 & & & & & & & & \\ & \ddots & & & & & & & \\ & & 1 & & & & & & \\ & & & \cos\varphi & \ldots & \sin\varphi & & & \\ & & & \vdots & & \vdots & & & \\ & & & -\sin\varphi & \ldots & \cos\varphi & & & \\ & & & & & & 1 & & \\ & & & & & & & \ddots & \\ & & & & & & & & 1 \end{pmatrix} \tag{3.49}$$

mit $\boldsymbol{U}^{-1} = \boldsymbol{U}^T$. Man betrachtet die Transformation der reellen, symmetrischen Matrix $\boldsymbol{V}$ in die Matrix $\boldsymbol{V}'$ durch das Produkt

$$\boldsymbol{V}' = \boldsymbol{U}^T \boldsymbol{V} \boldsymbol{U} \tag{3.50}$$

mit einer Matrix $\boldsymbol{U}$ der Gleichung (3.49). Dies wird *Ähnlichkeits-Transformation* genannt. Geometrisch kann man die Transformation als eine Drehung um den Winkel $-\varphi$ in der p,q-Ebene interpretieren; die Matrix $\boldsymbol{U}$ heißt entsprechend Rotationsmatrix. Sie ist charakterisiert durch das Indexpaar (p,q) und den Rotationswinkel φ. Nur die Elemente in den Spalten und Zeilen p und q werden durch die Transformation modifiziert. Die folgenden Formeln gelten für die speziellen Elemente in den Zeilen und Spalten mit den Indizes p und q:

$$\begin{aligned} V'_{pp} &= V_{pp}\cos^2\varphi - 2V_{pq}\cos\varphi\sin\varphi + V_{qq}\sin^2\varphi \\ V'_{qq} &= V_{pp}\sin^2\varphi + 2V_{pq}\cos\varphi\sin\varphi + V_{qq}\cos^2\varphi \\ V'_{pq} &= (V_{pp} - V_{qq})\cos\varphi\sin\varphi + V_{pq}(\cos^2\varphi - \sin^2\varphi) = V'_{qp}. \end{aligned} \tag{3.51}$$

Die übrigen Elemente in den Zeilen p und q für $j = 1,\ 2, \ldots\ n$ werden gemäß den folgenden Formeln transformiert

$$\begin{aligned} V'_{pj} &= V_{pj}\cos\varphi - V_{qj}\sin\varphi \\ V'_{qj} &= V_{pj}\sin\varphi + V_{qj}\cos\varphi \,, \end{aligned} \tag{3.52}$$

und die übrigen Elemente in den Spalten p und q für $i = 1, 2, \ldots n$ gemäß den Formeln

$$\begin{aligned} V'_{ip} &= V_{ip}\cos\varphi - V_{iq}\sin\varphi \\ V'_{iq} &= V_{ip}\sin\varphi + V_{iq}\cos\varphi\,. \end{aligned} \tag{3.53}$$

Alle verbleibenden Elemente (außerhalb der Spalten und Zeilen p und q) bleiben unverändert ($V'_{ij} = V_{ij}$ für $i, j \neq p, q$).

Die oben beschriebene Rotation wird *Jacobi-Rotation* oder (p, q)-Rotation genannt. Das Nicht-Diagonalelement V_{pq} heißt das *Pivotelement*.

Bestimmung des Drehwinkels φ. Der Drehwinkel φ kann so gewählt werden, daß das Pivotelement V_{pq} zu null transformiert wird

$$V'_{pq} = V'_{qp} = (V_{pp} - V_{qq})\cos\varphi\sin\varphi + V_{pq}(\cos^2\varphi - \sin^2\varphi) = 0$$

Dies läßt sich durch die Bedingung

$$\operatorname{cotan}(2\varphi) = \frac{V_{qq} - V_{pp}}{2V_{pq}} = \theta$$

ausdrücken. Der Winkel φ ist durch diese Bedingung nicht eindeutig definiert. Die folgende Festlegung beschränkt den Wert des Drehwinkels auf den Bereich $-\pi/4 < \varphi \leq +\pi/4$. Verwendet man die trigonometrische Relation $\operatorname{cotan}(2\varphi) = (1 - \tan^2\varphi)/(2\tan\varphi)$, wird $\tan\varphi$ zunächst durch die quadratische Gleichung $\tan^2\varphi + 2\theta\tan\varphi - 1 = 0$ definiert und dann gelöst durch

$$\tan\varphi = \begin{cases} \dfrac{1}{\theta + \operatorname{sign}(\theta)\sqrt{1+\theta^2}} & \text{für} \quad \theta \neq 0 \\ 1 & \text{für} \quad \theta = 0 \end{cases}\,. \tag{3.54}$$

Die Werte $\cos\varphi$ und $\sin\varphi$ werden schließlich aus $\tan\varphi$ berechnet zu

$$\cos\varphi = \frac{1}{\sqrt{1+\tan^2\varphi}} \qquad \sin\varphi = \cos\varphi\cdot\tan\varphi$$

Die Transformationsgleichungen zeigen, daß für die Diagonalelemente V_{pp} und V_{qq} die Beziehung $V'_{pp} + V'_{qq} = V_{pp} + V_{qq}$ gilt.

Jacobi-Methode. Die Transformation einer reellen symmetrischen Matrix $\boldsymbol{V}$ in eine Diagonalmatrix $\boldsymbol{D}$ ist in einem Schritt nicht möglich, sondern verlangt Iterationen mit vielen Schritten, bis eine ausreichende Genauigkeit erreicht ist. Eine Standardmethode ist die klassische Jacobi-Methode. In jedem Schritt wird das größte Nicht-Diagonalelement als Pivotelement ausgewählt und zur Bestimmung des Drehwinkels einer (p, q)-Rotation benutzt. Man beginnt mit einer Anfangsmatrix $\boldsymbol{V}^{(0)} = \boldsymbol{V}$, bildet eine Folge von Matrizen $\boldsymbol{V}^{(k)}$ mit $k = 1, 2, \ldots$ und erzeugt

$$\boldsymbol{V}^{(k)} = \boldsymbol{U}_k^T\,\boldsymbol{V}^{(k-1)}\,\boldsymbol{U}_k\,,$$

bis man eine Matrix $\boldsymbol{V}^{(k)}$ nahe einer Diagonalmatrix erhalten hat

$$\boldsymbol{V}^{(k)} = \boldsymbol{U}_k^T\cdots\boldsymbol{U}_2^T\boldsymbol{U}_1^T\boldsymbol{V}\boldsymbol{U}_1\boldsymbol{U}_2\cdots\boldsymbol{U}_k = \boldsymbol{D}\,. \tag{3.55}$$

Die Diagonalelemente D_{jj} der Matrix $\boldsymbol{D}$ sind Näherungswerte der Eigenwerte λ_j der Matrix $\boldsymbol{V}$. Durch die vielen Einzelrotationen wird eine Rotationsmatrix

$$\boldsymbol{U} = \boldsymbol{U}_1\boldsymbol{U}_2\cdots\boldsymbol{U}_k \tag{3.56}$$

gebildet, die nach Konvergenz in der Spalte j einen Näherungswert für den Eigenvektor enthält, entsprechend dem Eigenwert $\lambda_j = D_{jj}$ nach (3.47). Eine einzelne Jacobi-Rotation erfordert etwa $4n$ Operationen, und zur Bestimmung des Pivotelements benötigt man etwa n^2 Vergleiche. Für große Werte von n ist es effizienter, die Drehungen in zyklischer Folge auf alle von null verschiedenen Nicht-Diagonalelemente der Matrizen $\boldsymbol{V}^{(k)}$ anzuwenden, ohne nach dem maximalen Nicht-Diagonalelement zu suchen (zyklische Jacobi-Methode).

Konvergenz der Jacobi-Methode. Die Euklidische Norm der Matrix $\boldsymbol{V}$ ist definiert durch

$$\|\boldsymbol{V}\| = \sum_{i=1}^{n}\sum_{j=1}^{n} V_{ij}^2 = \text{Spur}\left(\boldsymbol{V}^T\boldsymbol{V}\right) . \tag{3.57}$$

Da die Matrix $\boldsymbol{V}$ symmetrisch ist, gilt $\|V\| = \text{Spur}\,\boldsymbol{V}^2 = \sum_k \lambda_k^2$. Die Euklidische Norm ist also invariant unter Jacobi-Drehungen. Die Summe der Quadrate der Nicht-Diagonalelemente

$$S(\boldsymbol{V}) = \sum_{i=1}^{n}\sum_{j=1(j\neq i)}^{n} V_{ij}^2 \tag{3.58}$$

wird bei einer einzelnen Jacobi-Drehung reduziert entsprechend den Gleichungen für die transformierten Matrixelemente

$$S(\boldsymbol{V}') = S(\boldsymbol{U}^T\boldsymbol{V}\boldsymbol{U}) = S(\boldsymbol{V}) - 2V_{pq}^2 + 2V'^2_{pq} = S(\boldsymbol{V}) - 2V_{pq}^2 ,$$

da der Rotationswinkel φ durch die Forderung $V'_{pq} = 0$ definiert ist. Also ist die Wirkung einer einzelnen Jacobi-Rotation eine Verminderung der Summe der Quadrate der Nicht-Diagonalelemente um $2V_{pq}^2$ und ein Anwachsen der Summe der Quadrate der Diagonalelemente um denselben Betrag. Es läßt sich zeigen, daß die klassische Jacobi-Methode und die zyklische Jacobi-Methode (in zeilen- und spaltenweiser Ordnung) konvergent sind, wobei die Summe der Quadrate der Nicht-Diagonalelemente gegen null geht. Die Konvergenz wird *quadratisch* nach einigen anfänglichen Iterationen, und falls die Eigenwerte nicht entartet sind. Eine Jacobi-Rotation erfordert $4n$ Operationen. Im allgemeinen werden etwa $3n^2$ bis $5n^2$ Rotationen notwendig sein; die Diagonalisierung ist also $\mathcal{O}(n^3)$ mit etwa $20n^3$ Operationen.

Ein Fortranprogramm `SMJROT` *zur Berechnung der Eigenwerte und Eigenvektoren einer symmetrischen $N \times N$-Matrix befindet sich im Anhang (Seite 272).*

Weitere Methoden. Für größere Matrizen und insbesondere für Bandmatrizen können andere Methoden als die Jacobi-Methode wirksamer sein. Die *Householder-Methode* [69, 68] kann zunächst einmal angewandt werden, um die Matrix $\boldsymbol{V}$ zu einer tridiagonalen Matrix zu reduzieren. In der Householder-Methode verwendet man spezielle Transformations-Matrizen, in denen eine Zeile und eine Spalte in einem Schritt transformiert werden. Die Eigenwerte der tridiagonalen Matrix können dann durch eine Methode bestimmt werden, die *kubisch* konvergiert. In der Jacobi-Methode wird die Bandstruktur zerstört.

Gleichzeitige Diagonalisierung zweier symmetrischer Matrizen $\boldsymbol{V}_a$ und $\boldsymbol{V}_b$. Gesucht ist eine Transformationsmatrix $\boldsymbol{A}$, die die beiden symmetrischen Matrizen gleichzeitig in Diagonalmatrizen transformiert. Dadurch soll zum Beispiel ein quadratischer Ausdruck

$$Q = \boldsymbol{x}^T\boldsymbol{V}_a\boldsymbol{x} + \boldsymbol{x}^T\boldsymbol{V}_b\boldsymbol{x} \tag{3.59}$$

mit einem Vektor $\boldsymbol{x}$ in einen quadratischen Ausdruck

$$Q = \boldsymbol{y}^T\boldsymbol{y} + \boldsymbol{y}^T\boldsymbol{S}\boldsymbol{y} \tag{3.60}$$

mit einem transformierten Vektor $\boldsymbol{y}$ umgeformt werden, wobei $\boldsymbol{S}$ eine Diagonalmatrix ist. Die Matrix $\boldsymbol{V}_a$, positiv-definit angenommen, hat eine Eigenwert-Zerlegung

$$\boldsymbol{V}_a = \boldsymbol{U}_a \boldsymbol{D} \boldsymbol{U}_a^T \,,$$

wobei $\boldsymbol{U}_a$ eine orthogonale Matrix ist. Die Matrix $\boldsymbol{D}$ ist eine Diagonalmatrix, die in der Form $\boldsymbol{D} = \mathrm{diag}(D_{11}, D_{22}, \ldots D_{nn})$ geschrieben werden kann. Man definiert jetzt eine neue Diagonalmatrix $\boldsymbol{D}^{1/2} = \mathrm{diag}(D_{11}^{1/2}, D_{22}^{1/2}, \ldots D_{nn}^{1/2})$ und ferner $\boldsymbol{D}^{-1/2}$, die als Diagonalelemente die inversen Quadratwurzeln der Diagonalelemente von $\boldsymbol{D}$ hat. Die *symmetrische* Transformationsmatrix $\boldsymbol{M}$ ist definiert durch

$$\boldsymbol{M} = \boldsymbol{U}_a \boldsymbol{D}^{-1/2} \boldsymbol{U}_a^T \,.$$

Mit einer Transformation $\boldsymbol{x} = \boldsymbol{M}\boldsymbol{y}$ erhält man für die quadratische Form Q (3.59)

$$\begin{aligned} Q = \boldsymbol{x}^T \boldsymbol{V}_a \boldsymbol{x} + \boldsymbol{x}^T \boldsymbol{V}_b \boldsymbol{x} &= \boldsymbol{y}^T \boldsymbol{M} \boldsymbol{V}_a \boldsymbol{M} \boldsymbol{y} + \boldsymbol{y}^T \boldsymbol{M} \boldsymbol{V}_b \boldsymbol{M} \boldsymbol{y} \\ &= \boldsymbol{y}^T \boldsymbol{y} + \boldsymbol{y}^T \boldsymbol{M} \boldsymbol{V}_b \boldsymbol{M} \boldsymbol{y} \,. \end{aligned}$$

Die zweite (symmetrische) Matrix $\boldsymbol{M}\boldsymbol{V}_b\boldsymbol{M}$ hat eine Eigenwert-Zerlegung

$$\boldsymbol{M} \boldsymbol{V}_b \boldsymbol{M} = \boldsymbol{U}_b \boldsymbol{S} \boldsymbol{U}_b^T \,.$$

Die endgültige Transformationsmatrix $\boldsymbol{A}$ ergibt sich nun zu

$$\boldsymbol{A} = \boldsymbol{M}\boldsymbol{U}_b = \boldsymbol{U}_a \boldsymbol{D}^{-1/2} \boldsymbol{U}_a^T \boldsymbol{U}_b \,, \tag{3.61}$$

da die Transformation der beiden Matrizen $\boldsymbol{V}_a$ und $\boldsymbol{V}_b$ durch diese Matrix $\boldsymbol{A}$ Diagonalmatrizen ergibt:

$$\boldsymbol{A}^T \boldsymbol{V}_a \boldsymbol{A} = \boldsymbol{I} \qquad \boldsymbol{A}^T \boldsymbol{V}_b \boldsymbol{A} = \boldsymbol{S} \,.$$

Mit der Transformation $\boldsymbol{x} = \boldsymbol{A}\boldsymbol{y}$ erhält man damit den quadratischen Ausdruck der Gleichung (3.60).

3.9 Singulärwert-Zerlegung

In diesem Abschnitt wird erneut das Problem behandelt, ein Gleichungssystem $\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}$ für einen n-Vektor $\boldsymbol{x}$ von Unbekannten mit einer $m \times n$ Matrix $\boldsymbol{A}$ zu lösen. Besonders die Fälle überbestimmter und schlecht konditionierter Systeme werden nun betrachtet.

Überbestimmte Gleichungssysteme. Wenn es mehr Gleichungen als Unbekannte gibt, d.h. $m > n$, dann gibt es im allgemeinen keinen Lösungsvektor $\boldsymbol{x}$; mit anderen Worten, der Restvektor (Residuum) $\boldsymbol{r}$, definiert durch

$$\boldsymbol{r} = \boldsymbol{A}\boldsymbol{x} - \boldsymbol{b} \,, \tag{3.62}$$

wird für keinen Vektor $\boldsymbol{x}$ gleich null. Das Gleichungssystem heißt *überbestimmt*. Die Standardmethode, eine Lösung zu finden, die zumindest *in der Nähe* einer exakten Lösung liegt, ist die Methode kleinster Quadrate, in der die Lösung $\boldsymbol{x}$ durch die Forderung eines Minimalwertes der Quadratsummen der Residuen bestimmt wird. Sie läßt sich folgendermaßen formulieren:

$$\begin{aligned} \boldsymbol{r}^T \boldsymbol{r} &= (\boldsymbol{A}\boldsymbol{x} - \boldsymbol{b})^T (\boldsymbol{A}\boldsymbol{x} - \boldsymbol{b}) \\ &= \boldsymbol{x}^T \boldsymbol{A}^T \boldsymbol{A} \boldsymbol{x} - 2\boldsymbol{x}^T \boldsymbol{A}^T \boldsymbol{b} + \boldsymbol{b}^T \boldsymbol{b} \;=\; \text{Minimum} \,. \end{aligned}$$

Das Minimum wird ermittelt, indem man die Ableitung des Ausdrucks nach $\boldsymbol{x}$ gleich null setzt:

$$\boldsymbol{A}^T\boldsymbol{A}\boldsymbol{x} - \boldsymbol{A}^T\boldsymbol{b} = 0 \quad \text{oder} \quad \boldsymbol{A}^T(\boldsymbol{A}\boldsymbol{x} - \boldsymbol{b}) = \boldsymbol{A}^T\boldsymbol{r} = 0\,.$$

Diese Gleichungen werden *Normalgleichungen* genannt; der Vektor $\boldsymbol{r}$ der Residuen ist *normal* in bezug auf die Spaltenvektoren der Matrix $\boldsymbol{A}$. Das Gleichungssystem hat in der Regel eine eindeutige Lösung

$$\boldsymbol{x} = \left(\boldsymbol{A}^T\boldsymbol{A}\right)^{-1}\boldsymbol{A}^T\boldsymbol{b}\,.$$

Die zugehörigen Berechnungen sind unten veranschaulicht. Als erstes sind die symmetrische Matrix $\boldsymbol{W} = \boldsymbol{A}^T\boldsymbol{A}$ und der Vektor $\boldsymbol{c} = \boldsymbol{A}^T\boldsymbol{b}$ zu berechnen

$$\boxed{\boldsymbol{W}} = \boxed{\boldsymbol{A}^T} \times \boxed{\boldsymbol{A}} \qquad \boxed{\boldsymbol{c}} = \boxed{\boldsymbol{A}^T} \times \boxed{\boldsymbol{b}}\,.$$

Dann läßt sich der Lösungsvektor $\boldsymbol{x}$ als Lösung der Gleichung $\boldsymbol{W}\boldsymbol{x} = \boldsymbol{c}$ ermitteln:

$$\boxed{\boldsymbol{x}} = \boxed{\boldsymbol{W}^{-1}} \times \boxed{\boldsymbol{c}}\,.$$

Will man mögliche Probleme bei der Lösung solcher überbestimmter Gleichungssysteme diskutieren, so stellt man die Lösung in einer anderen Form dar. $\boldsymbol{W}$ kann dann als Produkt geschrieben werden

$$\boldsymbol{W} = \boldsymbol{U}\boldsymbol{D}\boldsymbol{U}^T\,. \tag{3.63}$$

Die Diagonalisierung mit den Matrizen $\boldsymbol{W}$, $\boldsymbol{D}$ und $\boldsymbol{U}$ stellt sich wie folgt dar:

$$\boxed{\boldsymbol{W}} = \boxed{\boldsymbol{U}} \times \boxed{\boldsymbol{D}} \times \boxed{\boldsymbol{U}^T}$$

Die Matrix $\boldsymbol{U}$ ist eine orthonormale und die Matrix D eine diagonale Matrix mit den Eigenwerten λ der Matrix W als Diagonalelemente:

$$\boldsymbol{D} = \begin{pmatrix} \lambda_1 & & & \\ & \lambda_2 & & \\ & & \ddots & \\ & & & \lambda_n \end{pmatrix}\,. \tag{3.64}$$

Die inverse Matrix $\boldsymbol{W}^{-1}$ ergibt sich als Produkt

$$\boldsymbol{W}^{-1} = \boldsymbol{U}\boldsymbol{D}^{-1}\boldsymbol{U}^T\,,$$

wobei die Matrix $\boldsymbol{D}^{-1}$ die inversen Eigenwerte $1/\lambda_i$, $i = 1, 2, \ldots n$ als Diagonalelemente hat. Die Lösung kann also folgendermaßen geschrieben werden:

$$\boldsymbol{x} = \boldsymbol{U}\boldsymbol{D}^{-1}\boldsymbol{U}^T\boldsymbol{c}\,.$$

Ist die Matrix $\boldsymbol{W}$ singulär, dann ist zumindest einer der Eigenwerte null und die Lösung oben unmöglich (aufgrund der Konstruktion der Matrix $\boldsymbol{W}$ können die Eigenwerte nicht negativ sein). Die Spalten der Matrix $\boldsymbol{U}$ (diese sind die Eigenvektoren von $\boldsymbol{W}$), die den *null*-Eigenwerten entsprechen, sind eine orthogonale Basis des Nullraumes von $\boldsymbol{W}$. Jeder beliebige Vektor des Nullraumes kann zur Lösung $\boldsymbol{x}$ addiert werden. Um diese Willkür zu vermeiden, ist es besser, eine Lösung zu definieren, die keinen Beitrag aus dem Nullraum enthält. Dies ist zu erreichen, indem man statt der Matrix $\boldsymbol{D}^{-1}$ eine modifizierte Matrix $\boldsymbol{D}^{*-1}$ definiert, deren Diagonalelemente null für Eigenwerte λ von $\boldsymbol{D}$ mit dem Wert null sind (es wird also ∞ durch null ersetzt!). Die modifizierte Lösung

$$\boldsymbol{x} = \boldsymbol{U}\boldsymbol{D}^{*-1}\boldsymbol{U}^T\boldsymbol{c}$$

enthält dann keinen Vektor aus dem Nullraum. Man kann zeigen, daß dies die Lösung mit dem kleinsten Wert der Länge $|\boldsymbol{x}|$ ist.

Manchmal ist das Gleichungssystem nicht singulär, aber schlecht konditioniert: Einige der Eigenwerte sind sehr klein, und die Inversion ist zumindest ungenau. Dies wird durch eine Zahl κ, dem Verhältnis des größten zum kleinsten Eigenwert ausgedrückt: $\kappa = \lambda_{\max}/\lambda_{\min}$. Häufig nimmt man dann als bevorzugte Lösung eine mit modifizierter Matrix $\boldsymbol{D}^{*-1}$, in der für alle sehr kleinen Eigenwerte von $\boldsymbol{D}$ null gesetzt ist. Man sollte in der Praxis aber die genauere, im nächsten Abschnitt beschriebene Methode verwenden.

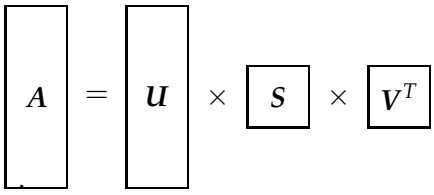
Singulärwert-Zerlegung. Diese Zerlegung ist eine allgemeine Methode, die man anwendet, wenn die Matrix entweder singulär oder nahezu singulär ist. Eine $m \times n$ Matrix $\boldsymbol{A}$ kann immer durch das Matrix-Produkt

$$\boldsymbol{A} = \boldsymbol{U}\boldsymbol{S}\boldsymbol{V}^T \tag{3.65}$$

ausgedrückt werden, worin die Matrix $\boldsymbol{U}$ dieselbe Dimension wie A hat, während die Matrizen $\boldsymbol{S}$ und $\boldsymbol{V}$ quadratische $n \times n$ Matrizen sind. Die Matrizen $\boldsymbol{U}$ und $\boldsymbol{V}$ sollen folgende Eigenschaften haben:

$$\boldsymbol{U}^T\boldsymbol{U} = \mathbf{1} \qquad \boldsymbol{V}^T\boldsymbol{V} = \mathbf{1}$$
$$\boldsymbol{V}\boldsymbol{V}^T = \mathbf{1}$$

d.h. die Matrix $\boldsymbol{U}$ ist eine spalten-orthogonale $m \times n$ Matrix und die Matrix $\boldsymbol{V}$ eine orthogonale $n \times n$ Matrix. Die Matrix S ist diagonal und hat positive oder Null-Elemente, genannt *Singulärwerte*. Die Matrix $\boldsymbol{A}$ stellt sich folgendermaßen dar:

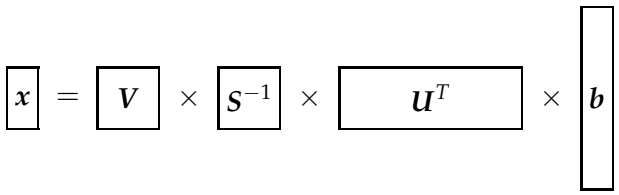


Die Darstellung ist auch für $m < n$ möglich; in diesem Fall sind wenigstens $n - m$ der Singulärwerte null. Im folgenden wird nur $m \geq n$ in Betracht gezogen.

Zuerst wird der Fall des vollen Rangs betrachtet, d.h. alle Singulärwerte S_{jj} sind ungleich null. Im Fall $m = n$ kann die Lösung

$$\boldsymbol{x} = \boldsymbol{V}\boldsymbol{S}^{-1}\left(\boldsymbol{U}^T\boldsymbol{b}\right) \tag{3.66}$$

unmittelbar als die Lösung der Matrixgleichung identifiziert werden, und im Fall $m > n$ entspricht sie der Lösung mit kleinsten Quadraten. Die Matrixoperationen der Gleichung (3.66) im allgemeinen Fall sind:



Ein Vergleich des Matrixprodukts $\boldsymbol{W} = \boldsymbol{A}^T\boldsymbol{A}$ mit $\boldsymbol{A}$ in der Darstellung der Gleichung (3.65)

$$\boldsymbol{W} = \boldsymbol{A}^T\boldsymbol{A} = \boldsymbol{V}\boldsymbol{S}\boldsymbol{S}\boldsymbol{V}^T \tag{3.67}$$

mit der Gleichung (3.63) zeigt, daß $\boldsymbol{D} = \boldsymbol{S}\boldsymbol{S}$ ist, d.h. die Eigenwerte λ der Matrix $\boldsymbol{W}$ sind identisch mit den Quadraten der Singulärwerte.

Auf ähnliche Weise wie oben wird eine modifizierte Matrix $\boldsymbol{S}^{*-1}$ definiert, in der die Diagonalelemente für die kleinen singulären Werte S_{jj} gleich null gesetzt werden. Der Lösungsvektor ist die Lösung kleinster Quadrate mit der kleinsten Länge

$$\boldsymbol{x} = \boldsymbol{V}\boldsymbol{S}^{*-1}\left(\boldsymbol{U}^T\boldsymbol{b}\right) \ . \tag{3.68}$$

4 Statistik

4.1 Einleitung

Historisch begann das Interesse an statistischen Problemen durch die Beschäftigung mit dem Glückspiel. Die vermutlich erste Arbeit ist von Christiaan Huygens, der im Jahre 1657 die Abhandlung 'De Ratiociniis in Ludo Aleae' schrieb. Wichtige Meilensteine waren Arbeiten von Jakob Bernoulli (Ars Conjectandi, 1705), von Laplace im Jahre 1795, der die Gleichung (4.1) formulierte, und von Pascal, der unter anderem in 'Oevres III, 307' schrieb:

> „...res hactenus erravit incerta; nunc autem quae experimento rebellis fuit rationis dominium effugere non potuit. Eam quippe tanta securitate in artem per geometricam reduximus, ut certitudinis eius particeps facta, iam audacter prodeat; et si metheseos demonstrationes cum aleae incertitudine iungendo, et quae contraria videntur conciliando, ab ultraque nominationem suam accipiens, stupendum hunc titulum iure sibi arrogat: aleae Geometrica".

(„...die Sache (d.h. das Glückspiel) war bisher ungewiß; nun aber, nachdem sie sich gegenüber dem Experiment unzugänglich gezeigt hatte, hat sie sich dem Verstand unterwerfen müssen. Wir haben es nämlich durch geometrische Argumentation auf eine berechenbare Kunst zurückgeführt, so daß es Teil an der Strenge der Geometrie hat und zuversichtlich vorangeht; es vereinigt die mathematische Gewißheit mit der Ungewißheit des Würfels, was vorher unvereinbar schien. Es nimmt nun den Namen beider an, und beansprucht den staunenerregenden Titel: Geometrie des Würfels.") [1]

Ein neues wichtiges Werkzeug sind Monte Carlo-Methoden, zusammen mit der großen Rechenleistung, die heutzutage mit vergleichsweise bescheidenen Kosten verfügbar ist. Praktisch brauchbare Lösungen für viele Probleme, die früher wegen ihrer Komplexität gar nicht oder nur mit sehr großem mathematischen Aufwand lösbar waren, können nun mit relativ simpler Monte Carlo-Programmierung angepackt werden (siehe Kapitel 5).

Zu Statistik und statistische Methoden gibt es eine große Auswahl an Literatur, siehe zum Beispiel [20, 38, 3, 10, 9, 26, 14, 53, 58, 51]. Eine sehr lesenswerte Einführung in die statistischen Probleme des Alltags ist [76].

4.2 Wahrscheinlichkeit

Ein formaler Zugang zum Wahrscheinlichkeitsbegriff geht von einem Satz von Axiomen aus, welche Wahrscheinlichkeiten und ihre Verknüpfungen formal definieren. Eine lesenswerte Einführung findet man in [66]. In der Praxis besteht natürlich das Problem, Wahrscheinlichkeiten tatsächlich festzulegen, und hierfür ist ein pragmatisches Vorgehen angesagt. Zwei Ansätze werden verwendet:
1) Falls ein Ereignis auf n verschiedene und gleich wahrscheinliche Arten eintreten kann, und k davon haben die Eigenschaft $\mathcal{A}$, so ist die Wahrscheinlichkeit für das Auftreten von $\mathcal{A}$

$$P(\mathcal{A}) = \frac{k}{n} \tag{4.1}$$

(= *günstige Fälle/mögliche Fälle*).
Man könnte einwenden, daß *gleich wahrscheinlich* einen Zirkelschluß impliziert. Jedoch folgt die Eigenschaft gleicher Wahrscheinlichkeit oft aus Symmetrieeigenschaften des Problems. Beispiele sind würfeln mit einem perfekten Würfel oder ein präzise gefertigtes Rouletterad. Ein

[1] Wir danken Dr. D. Haidt für den Hinweis auf dieses Zitat.

anderes Beispiel ist die statistische Thermodynamik, wo man eine große Zahl identischer Atome hat; dort gelten statistische Argumente mit großer Genauigkeit.

2) Falls es keine Symmetrieargumente gibt, muß man auf eine empirische Definition von Wahrscheinlichkeit zurückgreifen:
Eine Beobachtung wird unter gleichen Bedingungen n mal durchgeführt, und wenn die Beobachtungen voneinander unabhängig sind, und wenn in k Fällen die Eigenschaft $\mathcal{A}$ auftritt, so ist die Wahrscheinlichkeit für das Auftreten von $\mathcal{A}$

$$P(\mathcal{A}) = \lim_{n\to\infty} \frac{k}{n}. \tag{4.2}$$

Man könnte einwenden, daß es schwierig ist, unendlich viele Messungen zu machen. Wie in Kapitel 4.6.2 gezeigt wird, kann man den Fehler von $P(\mathcal{A})$ beliebig klein machen, indem man n dem gewünschten Fehler entsprechend groß wählt.

Beispiel 4.1 Söhne und Töchter.

Jemand hat zwei Kinder. Sie antwortet „ja“ auf die Frage: „Haben Sie eine Tochter“? – und daher hat sie mindestens eine Tochter. Welches ist die Wahrscheinlichkeit, daß sie zwei Töchter hat, falls die Wahrscheinlichkeiten für Sohn und Tochter gleich sind (stimmt nicht ganz genau)? Ehe man eine instinktive Antwort gibt, sollte man Gleichung (4.1) konsultieren. Es gibt a priori vier gleich wahrscheinliche Fälle: [Tochter,Tochter], [erst Tochter, dann Sohn], [erst Sohn, dann Tochter], [Sohn, Sohn]. Es gibt drei mögliche Fälle, weil die Bedingung *mindestens eine Tochter* den vierten Fall eliminiert, und nur der erste Fall ist günstig, und so ist die gesuchte Wahrscheinlichkeit $P = 1/3$.

Kombination von Wahrscheinlichkeiten. Gegeben sind zwei Arten von Ereignissen, $\mathcal{A}$ und $\mathcal{B}$. Die Wahrscheinlichkeit für das Auftreten von $\mathcal{A}$ ist $P(\mathcal{A})$, und die entsprechende Wahrscheinlichkeit von $\mathcal{B}$ ist $P(\mathcal{B})$. Dann ist die Wahrscheinlichkeit, daß $\mathcal{A}$ **oder** $\mathcal{B}$ auftritt:

$$P(\mathcal{A} \text{ oder } \mathcal{B}) = P(\mathcal{A}) + P(\mathcal{B}) - P(\mathcal{A} \text{ und } \mathcal{B}). \tag{4.3}$$

$P(\mathcal{A}$ und $\mathcal{B})$ ist die Wahrscheinlichkeit, daß $\mathcal{A}$ und $\mathcal{B}$ zusammen auftreten.
Falls sich die Ereignisse $\mathcal{A}$ und $\mathcal{B}$ gegenseitig ausschließen, gilt $P(\mathcal{A}$ und $\mathcal{B}) = 0$, und

$$P(\mathcal{A} \text{ oder } \mathcal{B}) = P(\mathcal{A}) + P(\mathcal{B})\,.$$

Ein Spezialfall ist $\mathcal{B} = \overline{\mathcal{A}}$ = Nichtauftreten von $\mathcal{A}$. Offensichtlich schließen sich $\mathcal{A}$ und $\overline{\mathcal{A}}$ gegenseitig aus, und es gilt

$$P(\mathcal{A} \text{ oder } \overline{\mathcal{A}}) = P(\mathcal{A}) + P(\overline{\mathcal{A}}) = 1.$$

Die Wahrscheinlichkeit, daß $\mathcal{A}$ **und** $\mathcal{B}$ zusammen auftreten, ist

$$P(\mathcal{A} \text{ und } \mathcal{B}) = P(\mathcal{A}) \cdot\ P(\mathcal{B}|\mathcal{A}). \tag{4.4}$$

$P(\mathcal{B}|\mathcal{A})$ ist die *bedingte Wahrscheinlichkeit* , daß das Ereignis $\mathcal{B}$ auftritt, vorausgesetzt, das Ereignis $\mathcal{A}$ ist aufgetreten. Falls die Ereignisse $\mathcal{A}$ und $\mathcal{B}$ *unabhängig* sind, gilt $P(\mathcal{B}|\mathcal{A}) = P(\mathcal{B})$, das Auftreten von $\mathcal{B}$ hängt nicht von $\mathcal{A}$ ab, und *nur* dann gilt

$$P(\mathcal{A} \text{ und } \mathcal{B}) = P(\mathcal{A}) \cdot\ P(\mathcal{B}). \tag{4.5}$$

Bayes' Theorem. Aus der Gleichung

$$P(\mathcal{A} \text{ und } \mathcal{B}) = P(\mathcal{A}) \cdot\ P(\mathcal{B}|\mathcal{A}) = P(\mathcal{B}) \cdot\ P(\mathcal{A}|\mathcal{B})$$

erhält man Bayes' Theorem[2]

$$P(\mathcal{A}|\mathcal{B}) = P(\mathcal{B}|\mathcal{A}) \cdot \frac{P(\mathcal{A})}{P(\mathcal{B})}. \tag{4.6}$$

Im allgemeinen Fall von n Ereignis-Klassen mit den Eigenschaften $\mathcal{A}_i$, $(i = 1, 2, \ldots n)$ lautet das Theorem

$$P(\mathcal{A}_i|\mathcal{B}) = \frac{P(\mathcal{B}|\mathcal{A}_i)P(\mathcal{A}_i)}{\sum_{i=1}^{n} P(\mathcal{B}|\mathcal{A}_i) \cdot P(\mathcal{A}_i)} \, . \tag{4.7}$$

Beispiel 4.2 Wahrscheinlichkeit beim Kartenspiel.

Die Wahrscheinlichkeit, beim zufälligen Ziehen aus einem Deck von Skatkarten ein As zu ziehen, ist $4/32 = 1/8$. Die Wahrscheinlichkeit, daß eine zufällig gezogene Karte ein As *oder* ein Pik ist, beträgt $4/32 + 8/32 - 1/32 = 11/32$ nach Gleichung (4.3).

Beispiel 4.3 Geburtstage.

Welches ist die Wahrscheinlichkeit, daß in einer Gruppe von 23 zufällig ausgewählten Leuten mindestens zwei an demselben Tag Geburtstag haben? Diese Wahrscheinlichkeit ist überraschend hoch. Angenommen, ein Geburtstag kann mit gleicher Wahrscheinlichkeit an jedem Tag des Jahres liegen (stimmt nicht ganz genau), dann ist die Wahrscheinlichkeit Q, daß alle 23 Leute ihren Geburtstag an *verschiedenen* Tagen haben,

$$Q = 1 \cdot \left(1 - \frac{1}{365}\right) \cdot \left(1 - \frac{2}{365}\right) \cdots \left(1 - \frac{22}{365}\right) = 0.493 \, ,$$

und deshalb ist die gesuchte Wahrscheinlichkeit $P = 1 - Q = 0.507$.

Beispiel 4.4 Tod in den Bergen.

In einem Buch über die bergsteigerischen Leistungen von Reinhold Messner ist folgendes zu lesen: „Wenn man bedenkt, daß die Wahrscheinlichkeit, bei einer Expedition auf einen Achttausender umzukommen, 3.4% beträgt, dann hatte Messner eine Wahrscheinlichkeit von $(3.4 \cdot 29) = 99\%$, bei seinen 29 Expeditionen getötet zu werden". – Das kann doch nicht wahr sein; was ist, wenn Messner zu einer 30. Expedition aufbricht? In der Tat sollte man nicht so pessimistisch sein. Die Wahrscheinlichkeit, eine Expedition zu überleben ist offensichtlich $1 - 0.034 = 0.966$. Wenn man annimmt, daß die einzelnen Expeditionen unabhängige Ereignisse darstellen, ist die Wahrscheinlichkeit, alle 29 Expeditionen zu überleben, $P = 0.966^{29} = 0.367$.

Beispiel 4.5 Das Ziegenproblem.

In einer Quiz-Sendung werden einer Kandidatin drei geschlossene Türen gezeigt. Hinter zwei dieser Türen ist eine Ziege, und hinter einer ist ein Auto. Sie kann auf eine der Türen zeigen und erhält das, was dahinter ist. Offensichtlich sind die Chancen, das Auto zu erhalten, 33%. Nun wird das Spiel aber abgeändert: Ehe die Tür der Kandidatin geöffnet wird, öffnet der Quizmaster eine andere Tür, und dahinter ist eine Ziege. Sollte die Kandidatin nun ihre

[2]Thomas Bayes, Britischer Geistlicher, 1702-1761, Minister in Tunbridge Wells, 1742, "An Essay toward solving a Problem in the Doctrine of Chances".

Meinung ändern und eine andere Tür wählen, um ihre Chancen zu verbessern? Die Antwort, die damals zu erheblichen Kontroversen führte, heißt *ja*. Um das einzusehen, kann man ohne Einschränkung der Allgemeinheit annehmen, daß das Auto hinter Türe Nr. 3 ist. Falls die Kandidatin ursprünglich Tür 1 gewählt hat, muß der Quizmaster Tür Nr. 2 öffnen, um eine Ziege vorzuzeigen. Falls die Kandidatin nun ihre Meinung ändert und Tür Nr. 3 wählt, ist die Chance, das Auto zu erhalten, $P_1 = 100\%$. Falls die Kandidatin zunächst Tür Nr. 2 wählt, geht das Argument genauso, also $P_2 = 100\%$. Falls die Kandidatin zunächst Tür Nr. 3 wählt und dann ihre Meinung ändert, erhält sie mit Sicherheit eine Ziege, also $P_3 = 0$. Die Wahrscheinlichkeit, eine gegebene Tür, etwa Nr. 2, zu wählen *und* das Auto zu erhalten, ist $0.33 \cdot P_2$, da sie jede der drei Türen 1, 2 oder 3 zunächst mit der gleichen Wahrscheinlichkeit wählt. Da sie immer nur genau eine der drei Türen wählen kann, kann man die drei Wahrscheinlichkeiten addieren und erhält für die Wahrscheinlichkeit, das Auto zu erhalten

$$P = \frac{1}{3} \cdot P_1 \; + \frac{1}{3} \cdot P_2 \; + \frac{1}{3} \cdot P_3 = \frac{1}{3} \; + \frac{1}{3} \; + 0 = \frac{2}{3}.$$

Änderung der ursprünglichen Wahl der Türe hat also die Chancen verdoppelt. Leute, die der Logik mißtrauen, können das Problem auch mit Monte Carlo-Methoden lösen.

Fallstricke bei statistischen Problemen. Angenommen, beim Roulette ist zehnmal hintereinander schwarz erschienen. Sollte man jetzt nicht auf rot setzen? Natürlich haben sich die Chancen für rot im nächsten Spiel nicht geändert. Unabhängig davon, was vorher war, sind sie nach wie vor 18/37. Mit demselben Argument sieht man ein, daß die Sequenz schwarz, schwarz, schwarz, schwarz, schwarz, schwarz genauso wahrscheinlich ist wie schwarz, rot, schwarz, rot, schwarz, rot.

Oft wird Gleichung (4.1) so angewandt, daß man günstige/mögliche Fälle berechnet. Dabei ist jedoch zu beachten, daß alle Fälle *gleich wahrscheinlich* sein müssen (siehe Beispiel 4.1 für richtiges Vorgehen).

Viele Fehlschlüsse geschehen, wenn man nicht beachtet, daß Ereignisse korreliert sein können. Dann ist Gleichung (4.4) zu verwenden statt Gleichung (4.5). Hier kann das Gefühl ebenso täuschen, zum Beispiel bei folgender Geschichte: „Hans und Katrin leben schon lange zusammen. Nun, endlich, werden sie im schönen Monat Mai heiraten". Frage: Welche der beiden folgenden Aussagen ist wahrscheinlicher: a) Hans wird im Mai heiraten oder b) Hans wird Katrin im Mai heiraten? Viele neigen zu Antwort b), aber dies ist falsch, und zwar wegen Gleichung (4.4) und weil Wahrscheinlichkeiten immer kleiner als eins sind.

Ereignisse, die unabhängig scheinen, sind es manchmal nicht: Angenommen man benötigt einen Brandmelder, der mit sehr hoher Wahrscheinlichkeit funktioniert. Ein Rauchmelder hat eine Zuverlässigkeit von 99.9%, d.h. er versagt in 0.1% der Fälle. Das ist vielleicht nicht gut genug. Daher installiert man einen *unabhängigen* Wärme- plus Infrarotdetektor. Angenommen, dieser versagt in 1% der Fälle, welches ist die Wahrscheinlichkeit, daß beide Detektoren versagen? Sie ist nicht notwendigerweise 1% mal $0.1\% = 10^{-5}$. Es könnte sein, daß beide Detektoren von demselben Rechner ausgelesen werden, und deshalb nicht unabhängig sind, oder daß die Person schläft, welche den Alarm überwachen sollte.

Die harmlos aussehende Gleichung (4.1) kann zu nichttrivialen Einsichten führen. Sie kann zum Beispiel einen wissenschaftlichen Anstrich für Murphy's Gesetz liefern, welches lautet: *Alles was schiefgehen kann, geht auch schief.*[3] Da es im allgemeinen viel mehr Möglichkeiten gibt, etwas falsch als es richtig zu machen, ist die Zahl der günstigen Möglichkeiten für Katastrophen größer als für die andere Alternative, und deshalb wird man öfter Mißerfolg als Erfolg antreffen. Ein Beispiel ist das Anstehen in einer Schlange. Jeder weiß, daß die anderen Schlangen schneller vorankommen als die eigene. Falls es zum Beispiel fünf Schlangen gibt, so ist die

[3] ursprüngliche Formulierung: Falls man etwas auf zwei oder mehr Wegen machen kann, und einer davon führt zu einer Katastrophe, so wird jemand diesen Weg beschreiten.

Wahrscheinlichkeit, daß wenigstens eine davon schneller ist als die eigene, 80% (eine der fünf Schlangen muß die schnellste sein; die Chance, nicht in dieser zu sein, ist $1 - 1/5$).
Nichtbeachtung von Gleichung (4.1) kann zu *Beweisen* für Astrologie, Hexerei oder was weiß ich führen: Ein Mann, dessen Horoskop einen Lotteriegewinn für nächste Woche voraussagte, gewinnt tatsächlich. Zu unwahrscheinlich, um durch Zufall erklärt zu werden? Nicht notwendigerweise. Die Wahrscheinlichkeit eines Lotteriegewinns mag ja sehr klein sein, um jedoch die Zahl der günstigen Fälle zu erhalten, muß man diese kleine Zahl mit der Zahl der möglichen Fälle multiplizieren. Dies ist die Zahl der Personen, die Lotterie spielen und an Astrologie glauben, und das können viele Millionen sein, so daß das Produkt einer sehr kleinen und einer sehr großen Zahl sehr wohl eins oder größer sein kann. Dieser eine Fall geht durch die Presse, aber die 1 000 000 Fehlschläge sind nicht so medienwirksam und werden nicht beachtet.
Eine Sammlung interessanter, überraschender Einsichten in statistische Phänomene aus dem täglichen Leben findet man bei [44].

4.3 Verteilungen

4.3.1 Grundbegriffe

Eine *Zufallsvariable* kann aufgrund statistisch unkontrollierter Einflüsse eine Reihe verschiedener Werte annehmen. Man unterscheidet *diskrete Zufallsvariable* und *kontinuierliche Zufallsvariable*.

Diskrete Zufallsvariable. Sie wird mit r_i bezeichnet und kann nur endlich viele verschiedene Werte annehmen, die durch eine Folge ganzer Zahlen indiziert werden können, $i = a, \ldots b$ (a und b sind die Kleinst- und Größtwerte von i). Die Wahrscheinlichkeit, daß bei einer Beobachtung oder Messung der Wert r_i für die Zufallsvariable auftritt, wird mit $P(r_i)$ bezeichnet. Als Wahrscheinlichkeit ist $P(r_i)$ eine Zahl zwischen 0 und 1. Summiert man die Wahrscheinlichkeiten *aller* möglichen Fälle, so muß sich 1 ergeben,

$$\sum_{i=a}^{b} P(r_i) = 1.$$

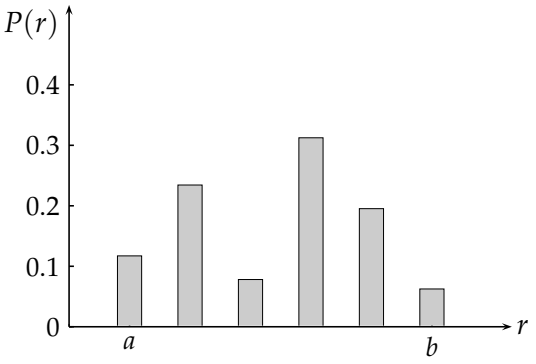


Abbildung 4.1: Verteilung einer diskreten Zufallsvariablen r zwischen den Grenzen a und b.

Ein wichtiger Begriff ist der **Mittelwert** einer diskreten Zufallsvariablen. Der Mittelwert $\langle r \rangle$ der diskreten Zufallsvariablen r ist definiert als

$$\langle r \rangle = \sum_{i=a}^{b} r_i \cdot P(r_i) \ .$$

Im Beispiel eines Rouletterads sind die Zufallsvariablen die Zahlen von 0 bis 36, auf welche die Kugel fallen kann. Ihre Wahrscheinlichkeiten sollten alle gleich sein, $P(r_i) = 1/37 \quad$ für alle $i = 0, 1, \ldots 36$.

Kontinuierliche Zufallsvariable. Sie können ein Kontinuum von Werten annehmen. Typischerweise sind sie das Resultat einer Messung oder einer Beobachtung. Sie können durch eine reelle Zahl x bezeichnet werden. Als Beispiel kann eine Temperaturmessung oder eine Messung der Körpergröße dienen. Die Wahrscheinlichkeit, daß eine solche Messung der Größe x einen Wert $a \leq x < b$ ergibt, wird bezeichnet als

$$P(a \leq x \leq b) = \int_a^b f(x)\ \mathrm{d}x,$$

wobei $f(x)$ die *Wahrscheinlichkeitsdichte* der Variablen x ist; dies ist in Abbildung 4.2 gezeigt. Eine Wahrscheinlichkeitsdichte ist nichtnegativ und muß auf 1 normiert sein,

$$f(x) \geq 0 \quad \text{und} \quad \int_{-\infty}^{\infty} f(x)\,\mathrm{d}x = 1\ .$$

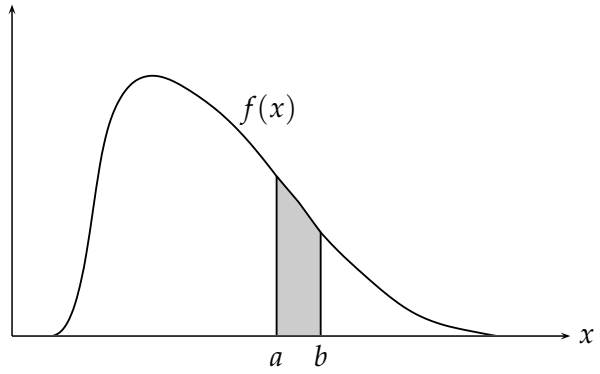


Abbildung 4.2: Verteilung einer kontinuierlichen Zufallsvariablen x. Die graue Fläche entspricht der Wahrscheinlichkeit $P(a \leq x \leq b)$.

Der **Mittelwert** der Zufallsvariablen x ist definiert als

$$\langle x \rangle = \int_{-\infty}^{\infty} x f(x)\,\mathrm{d}x. \tag{4.8}$$

Physiker haben die (schlechte) Angewohnheit, Wahrscheinlichkeitsdichten als Wahrscheinlichkeitsverteilungen oder Verteilungsfunktionen zu bezeichnen. Diese Namen sind jedoch für die integrierte Wahrscheinlichkeitsdichte reserviert. Die (integrierte) *Verteilungsfunktion* (Wahrscheinlichkeitsverteilung) $F(x)$ ist definiert als

$$F(x) = \int_{-\infty}^{x} f(x')\,\mathrm{d}x' \qquad \text{mit} \quad F(-\infty) = 0 \quad \text{und} \quad F(+\infty) = 1\ .$$

$F(x_0)$ ist die Wahrscheinlichkeit, daß $x \leq x_0$ ist. Offensichtlich sind $f(x)$ und $F(x)$ durch die Beziehung

$$f(x) = \frac{\mathrm{d}F(x)}{\mathrm{d}x}$$

verknüpft.

Die Wahrscheinlichkeitsdichte $f(x)$ ist *keine* Wahrscheinlichkeit! Dagegen geht die Größe $f(x)\,\Delta x$ in der Tat gegen die Wahrscheinlichkeit, daß x zwischen x und $x + \Delta x$ ist für $\Delta x \to 0$. Hieraus ist auch ersichtlich, daß die Wahrscheinlichkeit dafür gleich null ist, daß x einen präzisen ($\Delta x = 0$) vorher festgelegten Wert hat.

4.3.2 Erwartungswerte und Momente

Erwartungswerte. Ein wichtiger Parameter, der eine Wahrscheinlichkeitsdichte charakterisiert, ist der Erwartungswert der Funktion $h(x)$. Er ist definiert als

$$E[h] = \int_{-\infty}^{\infty} h(x) f(x)\, dx.$$

Ein wichtiger Spezialfall ist der Erwartungswert von x

$$E[x] = \int_{-\infty}^{\infty} x f(x)\, \mathrm{d}x\,, \tag{4.9}$$

der offensichtlich identisch mit dem Mittelwert $\langle x \rangle$ ist und die Eigenschaft $E[ax] = \langle ax \rangle = a\langle x \rangle$ hat. Der Erwartungswert ist per definitionem ein *linearer* Operator:

$$E[a \cdot g(x) + b \cdot h(x)] = a \cdot E[g(x)] + b \cdot E[h(x)]\ .$$

Momente. Die Erwartungswerte von x^n und von $(x - \langle x \rangle)^n$ werden n-te *algebraische Momente* μ_n und n-te *zentrale Momente* μ'_n genannt. Der Erwartungswert von x, oder Mittelwert μ von x, ist gleich dem ersten algebraischen Moment μ_1,

$$\mu = \mu_1 = E[x] = \langle x \rangle \quad \text{und} \quad \mu'_1 = E[x - \langle x \rangle] = 0.$$

Varianz und Standardabweichung. Das zweite zentrale Moment μ'_2 ist ein Maß für die Breite einer Wahrscheinlichkeitsdichte, man nennt es **Varianz** $V[x]$, und seine Definition ist also

$$V[x] = E\left[(x - \langle x \rangle)^2\right] = \int_{-\infty}^{+\infty} (x - \langle x \rangle)^2 f(x)\, \mathrm{d}x\,. \tag{4.10}$$

Für die Berechnung der Varianz sind die folgenden Formeln nützlich:

$$V[x] = E\left[x^2\right] - \langle x \rangle^2 \qquad V[ax] = E\left[a^2 (x - \langle x \rangle)^2\right] = a^2 \cdot V[x]\ . \tag{4.11}$$

Beweis:

$$\begin{aligned} V[x] = E\left[(x - \langle x \rangle)^2\right] &= E\left[\left(x^2 - 2x\langle x \rangle + \langle x \rangle^2\right)\right] \\ &= E\left[x^2\right] - 2E[x]\langle x \rangle + \langle x \rangle^2 = E\left[x^2\right] - (E[x])^2 = E\left[x^2\right] - \langle x \rangle^2\ . \end{aligned}$$

Die Varianz ist per definitionem positiv und wird auch mit σ^2 bezeichnet; die Größe σ heißt **Standardabweichung**. Physiker nennen σ oft den *Fehler*,

$$\sigma = \sqrt{V[x]} \quad \text{oder} \quad V[x] = \sigma^2.$$

Der Wert von σ ist ein Maß für die Größe der statistischen Schwankungen der Zufallsvariablen um den Mittelwert.

Höhere Momente. Einige der anderen Momente oder ihre Funktionen haben ebenfalls spezielle Namen. Die *Schiefe* ist definiert als die dimensionslose Größe $\beta_1 = \mu_3^2/\mu_2^3$; sie ist null für symmetrische und positiv oder negativ für unsymmetrische Wahrscheinlichkeitsdichten.

Eine Wahrscheinlichkeitsdichte ist eindeutig definiert durch alle ihre Momente.

Beweis: Man betrachtet zwei Wahrscheinlichkeitsdichten $f_1(x)$ und $f_2(x)$. Man entwickelt ihre Differenz in eine Potenzreihe:

$$f_1(x) - f_2(x) = c_0 + c_1 x + c_2 x^2 + \dots\ .$$

und bildet die positive Größe

$$\int (f_1(x) - f_2(x))^2 \, \mathrm{d}x = \int (f_1(x) - f_2(x)) \, (c_0 + c_1 x + c_2 x^2 + \ldots) \, \mathrm{d}x \geq 0.$$

Mit der Definition für die Momente wird das zweite Integral

$$c_0(1-1) + c_1(\mu_1(1) - \mu_1(2)) + c_2(\mu_2(1) - \mu_2(2)) + \ldots \geq 0\,,$$

wobei die Argumente 1,2 der Momente μ_n sich jeweils auf die Wahrscheinlichkeitsdichten $f_1(x)$ und $f_2(x)$ beziehen. Falls nun die beiden Wahrscheinlichkeitsdichten identische Momente haben, also $\mu_n(1) = \mu_n(2)$, dann ist das letzte Integral null. Da der Integrand von $\int (f_1(x) - f_2(x))^2 \, \mathrm{d}x$ überall ≥ 0 ist, folgt daß $f_1(x) \equiv f_2(x)$ sein muß.

Momente für diskrete Zufallsvariable. Sie sind analog wie für die kontinuierlichen Variablen definiert, nur daß die Integrale durch die entsprechenden Summen ersetzt sind.

4.3.3 Charakterisierung von Wahrscheinlichkeitsdichten

Viele Wahrscheinlichkeitsdichten haben ein ausgeprägtes Maximum und können gekennzeichnet werden durch einen *Lokalisierungsparameter*, der angibt, *wo* die Verteilung hauptsächlich ist (zum Beispiel Mittelwert, Gleichung (4.9)), und durch einen *Dispersionsparameter*, der die *Breite* angibt (zum Beispiel Standardabweichung und Varianz, Gleichung (4.10)). Weitere nützliche Größen sind folgende:

Median. Der Median $x_{0.5}$ ist neben dem Mittelwert ein nützlicher *Lokalisierungsparameter* einer Verteilung. Die Definition

$$\int_{-\infty}^{x_{0.5}} f(x) \, \mathrm{d}x = 0.5$$

bedeutet, daß ein zufällig gewählter Wert von x mit gleicher Wahrscheinlichkeit oberhalb bzw. unterhalb des Medians $x_{0.5}$ liegt. So läßt sich zum Beispiel die Länge des Physikstudiums an der Universität Hamburg am besten durch den Median charakterisieren, welcher in diesem Fall 12 Semester beträgt, d.h. 50% der Studenten schließen ihr Studium bis zum 12. Semester ab. Der Mittelwert würde in diesem Fall ein verzerrtes Bild liefern, da einige Studenten mehr als 50 Semester haben, und diese würden die Mittelwertbildung stark beeinflussen.

Wahrscheinlichster Wert. Der wahrscheinlichste Wert x_m ist derjenige x-Wert, bei dem die Wahrscheinlichkeitsdichte $f(x)$ ihr Maximum hat.

Quadratischer Mittelwert (rms). Der quadratische Mittelwert, (*root-mean-square*-Wert) ist die Wurzel des Erwartungswertes des Quadrats der Zufallsvariablen (vgl. Gleichung (4.11))

$$x_{rms} = \sqrt{E\left[x^2\right]} = \sqrt{V\left[x\right] + \langle x \rangle^2}\,.$$

Im Falle eines verschwindenden Mittelwertes ist der *rms*-Wert gleich der Quadratwurzel aus der Varianz.

FWHM. Die volle Breite einer Verteilung auf halber Höhe des Maximums (*full-width-half-maximum*) läßt sich leicht aus Verteilungen abschätzen. Für eine Gauß-Verteilung ist FWHM $= 2\sigma\sqrt{2\ln 2} = 2.3545\,\sigma$.

Beispiel 4.6 Maxwellsche Geschwindigkeitsverteilung eines idealen Gases.

Die Wahrscheinlichkeitsdichte des Betrags der Geschwindigkeit v der Moleküle in einem idealen Gas bei der absoluten Temperatur T ist

$$f(v) = N \cdot (m/2\pi kT)^{\frac{3}{2}} \exp(-mv^2/2kT) \cdot 4\pi v^2 \;;$$

dabei ist m die Molekülmasse und k die Boltzmann-Konstante. Der wahrscheinlichste Wert der Geschwindigkeit, der von den Parametern m und T abhängt, beträgt

$$v_m = (2kT/m)^{1/2} \qquad \text{(wahrscheinlichster Wert)} .$$

Die Wahrscheinlichkeitsdichte ist in Abbildung 4.3 als Funktion der normierten Geschwindigkeit v/v_m dargestellt. Die charakteristischen Parameter der Verteilung

$$\begin{aligned} \langle v \rangle &= (8kT/\pi m)^{1/2} = 1.128 \cdot v_m \qquad && \text{(Mittelwert)} \\ v_{0.5} &= 1.098 \cdot v_m && \text{(Median)} \\ v_{\rm rms} &= (3kT/m)^{1/2} = 1.225 \cdot v_m && \text{(rms)} \end{aligned}$$

sind in der Abbildung eingezeichnet.

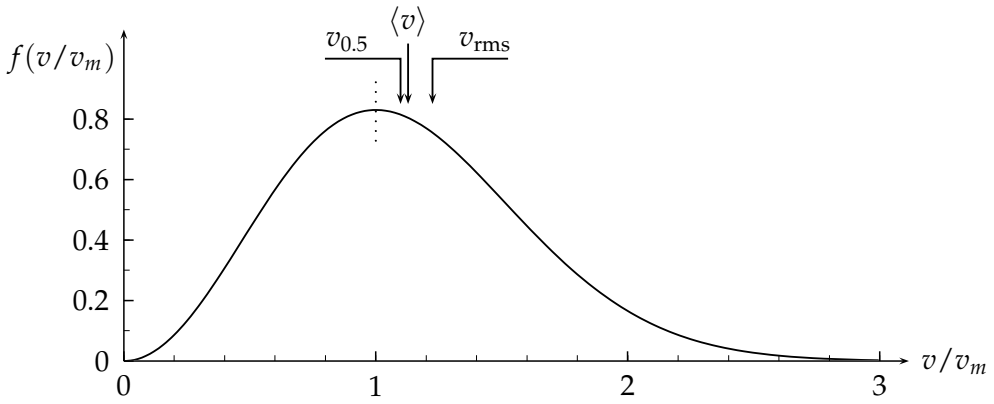


Abbildung 4.3: Maxwellsche Geschwindigkeitsverteilung eines idealen Gases als Funktion von v/v_m (v_m = wahrscheinlichster Wert der Geschwindigkeit v). Der Mittelwert $\langle v \rangle$, der rms-Wert $v_{\rm rms}$, und der Median $v_{0.5}$ sind markiert.

4.4 Spezielle diskrete Verteilungen

4.4.1 Kombinatorik

Für r verschiedene Objekte gibt es $1 \cdot 2 \cdots (r-1) \cdot r = r!$ verschiedene Möglichkeiten, die Objekte in einer Reihe anzuordnen ($0! = 1! = 1$). Die Zahl von Möglichkeiten, r Objekte aus n verschiedenen Objekten auszuwählen, wobei es auf die Reihenfolge der Auswahl ankommt, ist

$$P_n^r = n(n-1)(n-2)\cdots(n-r+1) = \frac{n!}{(n-r)!} .$$

Falls es auf die Reihenfolge der Auswahl nicht ankommt, muß die obenstehende Zahl durch $r!$ dividiert werden, und man erhält

$$C_n^r = \frac{P_n^r}{r!} = \binom{n}{r} = \frac{n!}{r!(n-r)!} = \left(\frac{n}{r}\right) \cdot \left(\frac{n-1}{r-1}\right) \cdots \left(\frac{n-r+1}{1}\right) .$$

Diese ganzen Zahlen heißen Binomialkoeffizienten. Für numerische Rechnungen sollte man den letztgenannten Ausdruck verwenden. Die Binomialkoeffizienten erscheinen im *Binomialtheorem*

$$(p+q)^n = \sum_{r=0}^{n} \binom{n}{r} p^r \cdot q^{n-r} . \tag{4.12}$$

Fakultät und Gammafunktion. Die *Fakultät* $n!$ einer positiven ganzen Zahl n ist das Produkt $1 \cdot 2 \cdots (n-1) \cdot n$; sie hat die Eigenschaft $n! = n \cdot (n-1)!$ (mit $0! = 1$). Für große n gilt als Näherung die *Stirlingsche Formel*

$$\ln n! \approx \left(n + \frac{1}{2}\right) \ln n - n + \ln \sqrt{2\pi}\,.$$

Die Fakultät kann für nicht ganzzahlige Werte des Arguments verallgemeinert werden durch

$$x! = \int_0^\infty u^x e^{-u} \mathrm{d}u\,.$$

Die Stirlingsche Formel gilt auch für nicht ganzzahlige Werte.

Die *Gammafunktion* $\Gamma(x)$ ist definiert durch $\Gamma(x+1) = x!$. Sie hat die Eigenschaft $\Gamma(x+1) = x\Gamma(x)$. Einige Werte für die Gammafunktion sind

x	$-1/2$	0	$1/2$	1	$3/2$	2	$5/2$	3	$7/2$	4
$\Gamma(x)$	$-2\sqrt{\pi}$	∞	$\sqrt{\pi}$	1	$\sqrt{\pi}/2$	1	$3\sqrt{\pi}/4$	2	$15\sqrt{\pi}/8$	6

Nützliche Formeln sind

$$\Gamma(x) = \frac{\Gamma(x+1)}{x} \qquad \Gamma(x) = (x-1)\Gamma(x-1)\,.$$

In numerischen Rechnungen ist es oft vorteilhaft, $\ln \Gamma(x)$ zu wählen statt $\Gamma(x)$, um eine Überschreitung des Zahlenbereichs zu vermeiden.

Beispiel 4.7 Permutationen und Kombinationen.

Man betrachtet den Fall von $r = 3$ Objekten, die aus $n = 5$ verschiedenen Objekten ausgewählt werden: Dazu gibt es $\binom{5}{3} = 10$ Kombinationen. Es gibt $r! = 6$ Möglichkeiten, die drei Objekte in einer verschiedenen Reihenfolge anzuordnen. Falls zum Beispiel in einem Quiz der Gewinner drei Dinge aus einer Menge von insgesamt fünf auswählen darf, so hat er dabei $\binom{5}{3} = 10$ verschiedene Möglichkeiten.

4.4.2 Binomialverteilung

Falls die Wahrscheinlichkeit für das Auftreten eines Ereignisses p ist, so ist die Wahrscheinlichkeit, daß es bei n Versuchen *genau* r mal auftritt, gegeben durch die Binomialverteilung

$$\boxed{P(r) = \binom{n}{r} p^r (1-p)^{n-r} \qquad r = 0,\, 1,\, 2 \ldots\, n}\,. \tag{4.13}$$

Dies ist leicht zu sehen: Die Wahrscheinlichkeit, daß das Ereignis in jedem der r ersten Versuche auftritt und nicht in den $n-r$ letzten, ist $p^r(1-p)^{n-r}$; diese Reihenfolge ist nur eine von insgesamt $\binom{n}{r} = n!/\,(r!(n-r)!)$ möglichen. Beispiele von Binomialverteilungen sind in Abbildung **??** zu sehen.

Es ist leicht zu sehen, daß $P(r)$ korrekt auf 1 normiert ist. Dazu betrachtet man den Spezialfall $q = 1 - p$ von Gleichung (4.12) :

$$(p + (1-p))^n = 1^n = 1 = \sum_{r=0}^{n} \binom{n}{r} p^r (1-p)^{n-r} = \sum_{r=0}^{n} P(r)\,.$$

Der *Mittelwert* von r ist

$$\langle r \rangle = E[r] = \sum_{r=0}^{n} rP(r) = np\,. \tag{4.14}$$

Die Varianz $V[r] = \sigma^2$ ist

$$V[r] = \sum_{r=0}^{n} (r - \langle r \rangle)^2 \cdot P(r) = np(1-p)\,. \tag{4.15}$$

Beweis: Man geht aus von der Binomialentwicklung

$$f(t) = (pt+q)^n = \sum_{r=0}^{n} p^r t^r q^{n-r} \binom{n}{r}$$

und differenziert nach dem Parameter t

$$\frac{\mathrm{d}f}{\mathrm{d}t} = np(pt+q)^{n-1} = \sum_{r=0}^{n} rp^r t^{r-1} q^{n-r} \binom{n}{r}\,.$$

Für $t = 1$ und mit $p + q = 1$ erhält man

$$np = \sum_{r=0}^{n} rp^r (1-p)^{n-r} \binom{n}{r} = \sum rP(r) = \langle r \rangle\,.$$

Die *Varianz* erhält man in ähnlicher Weise:

$$\frac{\mathrm{d}^2 f}{\mathrm{d}t^2} = np^2(n-1)(pt+q)^{n-2} = \sum r(r-1)t^{r-2} p^r q^{n-r} \binom{n}{r}$$

und, wieder für $t = 1$ und mit $p + q = 1$

$$n(n-1)p^2 = \sum r^2 P(r) - \langle r \rangle\,,$$

woraus man den Ausdruck für die Varianz Gleichung (4.15) erhält, wenn man die Gleichungen (4.14) und (4.11) benutzt.

Beispiel 4.8 Würfel.

Wie groß ist die Wahrscheinlichkeit, mit $n = 6$ Würfen eines Würfels genau null mal die 6, genau zweimal die 6, und mindestens einmal die 6 zu erhalten? Für einen korrekten Würfel ist $p = 1/6$ und

$$\begin{aligned} P(0) &= \left(\frac{1}{6}\right)^0 \cdot \left(\frac{5}{6}\right)^6 \binom{6}{0} = 33.5\% \\ P(2) &= \left(\frac{1}{6}\right)^2 \cdot \left(\frac{5}{6}\right)^4 \binom{6}{2} = 20.1\% \\ P(\geq 1) &= (1 - P(0)) = 66.5\%\,. \end{aligned}$$

Beispiel 4.9 Werfen einer Münze.

Beim Werfen einer Münze gibt es die beiden Möglichkeiten *Kopf* und *Wappen*. Die untenstehende Tabelle zeigt die Wahrscheinlichkeit für r mal *Kopf* in $n = 5$ Würfen, für zwei verschiedene Werte der Wahrscheinlichkeit p für *Kopf*, $p = 0.5$ und $p = 0.6$.

	$r =$	0	1	2	3	4	5	
$p = 0.5$	$P(r) =$	1/32	5/32	10/32	10/32	5/32	1/32	
		3.13	15.63	31.23	31.23	15.63	3.13	in %
$p = 0.6$	$P(r) =$	1.02	7.68	23.04	34.56	25.92	7.78	in %

Für $p = 0.5$ ist die Verteilung symmetrisch. Die *kleine* Änderung von $p = 0.5$ zu $p = 0.6$ hat einen großen Einfluß auf die Wahrscheinlichkeit, besonders große oder kleine Werte von r zu erhalten.

Beispiel 4.10 Wie viele Versuche sind notwendig?

Wie viele Versuche n sind notwendig, um mit einer Wahrscheinlichkeit von $\alpha = 90\%$ mindestens ein Ereignis zu erhalten, wenn $p = 0.5$ ist? Der Wert von n muß die Ungleichung $P(r \geq 1) \geq \alpha$ erfüllen. Dies kann geschrieben werden als $P(r \geq 1) = 1 - P(0) \geq \alpha$ und

$$P(0) = (1-p)^n \leq 1 - \alpha \qquad \text{oder} \qquad n \geq \frac{\ln(1-\alpha)}{\ln(1-p)} \, .$$

Für die gegebenen Zahlen $p = 0.5$ und $\alpha = 0.9$ erhält man das numerische Resultat $n \geq \ln 0.1 / \ln 0.5 = 3.32$, d.h. $n = 4$ Versuche sind notwendig.

4.4.3 Poisson-Verteilung

Die Poisson-Verteilung gibt die Wahrscheinlichkeit an, genau r Ereignisse zu erhalten, wenn die Zahl n der Versuche sehr groß und die Wahrscheinlichkeit für das Auftreten eines Ereignisses p in einem einzigen Versuch sehr klein ist, mit einem endlichen Mittelwert $\langle r \rangle = \mu = np$. Die Poisson-Verteilung kann als Grenzwert der Binomialverteilung abgeleitet werden (siehe Beispiel 4.11) und hat nur einen Parameter, nämlich den Mittelwert μ. Die Poisson-Verteilung ist gegeben durch

$$\boxed{P(r) = \frac{\mu^r e^{-\mu}}{r!} \qquad r = 0,\, 1,\, 2,\, \ldots} \, .$$

Ausgehend von $P(0) = e^{-\mu}$ können weitere Werte mit der Rekursionsformel

$$P(r+1) = P(r) \cdot \mu/(r+1)$$

berechnet werden. Es ist leicht zu sehen, daß die Poisson-Verteilung korrekt auf 1 normiert ist,

$$\sum_{r=0}^{\infty} P(r) = e^{-\mu} \sum_{r=0}^{\infty} \frac{\mu^r}{r!} = e^{-\mu}\, e^{+\mu} = 1 \, .$$

Der Mittelwert der Poisson-Verteilung ist $\langle r \rangle = \mu$.
Beweis:

$$E\,[r] = \sum_{r=0}^{\infty} r \frac{e^{-\mu}\mu^r}{r!} = \sum_{r=1}^{\infty} r \frac{e^{-\mu}\mu^r}{r!}$$
$$= \mu \sum_{r=1}^{\infty} r \frac{e^{-\mu}\mu^{r-1}}{(r-1)!r} = \mu \sum_{r=1}^{\infty} \frac{e^{-\mu}\mu^{r-1}}{(r-1)!} = \mu \sum_{s=0}^{\infty} \frac{e^{-\mu}\mu^s}{s!} = \mu \, .$$

Der Ausdruck für die *Varianz* kann folgendermaßen abgeleitet werden: Aus $V\,[r] = np(1-p)$ für die Binomialverteilung Gleichung (4.15) erhält man, mit $p \to 0$, $V\,[r] = np = \mu$:

$$V\,[r] = \sigma^2 = E\left[(r-\mu)^2\right] = \mu \, . \tag{4.16}$$

Die Poisson-Verteilung tritt in vielen Fällen auf, in denen man Dinge oder Ereignisse zählt, wie zum Beispiel die Zahl von Kernreaktionen oder von Teilchen-Zerfällen oder die Zahl der gefangenen Fische in einem Angelwettbewerb. Beispiele von Poisson-Verteilungen sind in Abbildung 4.4 zu sehen. Ein nützliche Tabelle mit Konfidenzgrenzen der Poisson-Verteilung ist Tabelle 6.1.

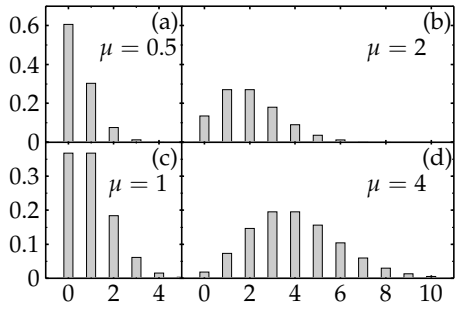


Abbildung 4.4: Poisson-Verteilungen mit den Mittelwerten $\mu = 0.5$, $\mu = 1$, $\mu = 2$ und $\mu = 4$.

Beispiel 4.11 Ableitung der Poisson-Verteilung aus der Binomialverteilung.

Die Binomialverteilung ist

$$P(r) = \binom{n}{r} p^r (1-p)^{n-r}$$

mit einem Mittelwert $\mu = np$. Man ersetzt p durch μ/n und erhält

$$P(r) = \frac{n(n-1)\cdots(n-r+1)}{r!} \left(\frac{\mu}{n}\right)^r \left(1-\frac{\mu}{n}\right)^{n-r} = \frac{n(n-1)\cdots(n-r+1)}{n^r} \frac{\mu^r}{r!} \left(1-\frac{\mu}{n}\right)^{n-r} ,$$

und durch Umordnen erhält man

$$P(r) = \frac{\mu^r}{r!} \left[1 \cdot \left(1-\frac{1}{n}\right)\left(1-\frac{2}{n}\right)\cdots\left(1-\frac{r-1}{n}\right) \frac{(1-\mu/n)^n}{(1-\mu/n)^r}\right] .$$

Man erhöht die Zahl n und erniedrigt p, wobei man das Produkt $\mu = pn$ konstant hält. In der Grenze $n \to \infty$ wird der eine Faktor in der Klammer

$$\lim_{n\to\infty} (1-\mu/n)^n = e^{-\mu} ,$$

und der andere Faktor wird 1, und so erhält man die Poisson-Verteilung

$$\lim_{n\to\infty} P(r) = \frac{\mu^r}{r!} e^{-\mu} .$$

Beispiel 4.12 Alternative Ableitung der Poisson-Verteilung.

Falls λ die mittlere Rate für das Auftreten eines Ereignisses ist, so ist die Wahrscheinlichkeit für das Auftreten eines einzelnen Ereignisses in dem kleinen Zeitintervall δt gegeben durch $\lambda\ \delta t$, also proportional zur Länge des Zeitintervalls,

$$\text{Wahrscheinlichkeit (ein Ereignis in } (t, t+\delta)) = \lambda\, \delta t + o(\delta t) .$$

Die Wahrscheinlichkeit für kein Ereignis ($r = 0$) im Intervall $(t, t + \delta t)$, falls kein Ereignis im Intervall $(0, t)$ ist, ist

$$\begin{aligned} P_0(t + \delta t) &= P_0(t)\,(1 - \lambda\,\delta t + o(\delta t)) \\ \frac{P_0(t + \delta t) - P_0(t)}{\delta t} &= -\lambda P_0(t) + \frac{o(\delta t)}{\delta t}\ . \end{aligned}$$

In der Grenze $\delta t \to 0$ ist die linke Seite die Ableitung von $P_0(t)$ nach der Zeit t, und die Gleichung wird eine Differentialgleichung für $P_0(t)$ mit der Lösung

$$P_0(t) = \exp(-\lambda t)\ .$$

Dies ist die Poisson-Wahrscheinlichkeit für $r = 0$ Ereignisse, bei einem Mittelwert $\mu = \lambda t$. Für $r = 1$ kann das Ereignis entweder im Intervall $(0, t)$ oder im Intervall $(t, t + \delta t)$ liegen:

$$P_1(t + \delta t) = P_1(t)\,(1 - \lambda\,\delta t + o(\delta t)) + P_0(t)\,(\lambda\,\delta t + o(\delta t))$$

Nach einer Umformung erhält man in der Grenze $\delta t \to 0$ die Differentialgleichung

$$\frac{dP_1(t)}{dt} = -\lambda P_1(t) + \lambda\, e^{-\lambda t} \quad \text{mit der Lösung} \quad P_1(t) = \lambda t\, e^{-\lambda t}\ .$$

Im allgemeinen Fall ($r \geq 1$) gilt die Differentialgleichung

$$\frac{dP_r(t)}{dt} = -\lambda P_r(t) + \lambda\, P_{r-1}(t) \quad \text{mit der Lösung} \quad P_r(t) = \frac{(\lambda t)^r}{r!} e^{-\lambda t}\ .$$

Dies ist die Poisson-Verteilung mit Mittelwert $\mu = \lambda t$.

Beispiel 4.13 Tod durch Pferdetritte in der preußischen Armee.

Diese Geschichte kann man in vielen alten Statistikbüchern lesen. In der preußischen Armee wurde für jedes Jahr die Zahl der tödlichen Unfälle durch Huftritte registriert, und zwar für 10 Armeecorps über 20 Jahre (also insgesamt 200 Zahlen). Diese sind in der Tabelle aufgeführt.

Zahl der Todesfälle r	0	1	2	3	4	5	6	Summe
Zahl der Corps-Jahre mit r Todesfällen	109	65	22	3	1	0	0	200
Erwartete Zahl	108.7	66.3	20.2	4.1	0.6	0.07	0.01	

Die Gesamtzahl von Todesfällen ist $1 \cdot 65 + 2 \cdot 22 + 3 \cdot 3 + 4 = 122$, und die mittlere Zahl von Toten pro Corps und pro Jahr ist 122/200=0.61. Die nach einer Poisson-Verteilung mit Mittelwert $\mu = 0.61$ erwartete Zahl von Fällen ist ebenfalls in der Tabelle zu sehen. Die Übereinstimmung zwischen den erwarteten und den beobachteten Zahlen ist sehr gut – eigentlich zu gut.

Beispiel 4.14 Druckfehler.

Ein Buch von 500 Seiten enthält 50 Druckfehler. Angenommen, die Druckfehler sind zufällig über das Buch verteilt, welches ist die Wahrscheinlichkeit, daß auf einer bestimmten Seite genau null, einer oder zwei Druckfehler sind? Antwort: $\mu = 50/500 = 0.1$= mittlere Zahl von Fehlern/Seite, und $P(0) = e^{-0.1} = 90.5\%$, $P(1) = 0.1 \cdot e^{-0.1} = 9.05\%$, $P(2) = 0.01e^{-0.1}/2 = 0.45\%$.

Transformation Poisson-verteilter Daten auf einheitliche Varianz. In Histogrammen folgt die Zahl der Einträge in jedem Intervall oft einer Poisson-Verteilung. Falls die Zahlen der Einträge

in den einzelnen Intervallen stark unterschiedlich sind, kann man die Genauigkeit der einzelnen Datenwerte nicht leicht auf einen Blick einschätzen, weil sie alle verschiedene Varianzen haben. Die folgende Formel transformiert die Zahl der Einträge in jedem Intervall r_i zu neuen Variablen y_i, welche alle ungefähr dieselbe Varianz von 1 haben:

$$y_i = 2 \cdot \sqrt{r_i} \quad \text{oder auch} \quad y_i = \sqrt{r_i} + \sqrt{r_i + 1}\,.$$

Dies sieht man leicht durch Anwendung von Gleichung (4.51). Die letztere Transformation hat eine Varianz von 1.0 ($\pm 6\%$) für alle Argumente $r_i > 1$.

4.5 Spezielle Wahrscheinlichkeitsdichten

4.5.1 Gleichverteilung

Diese Wahrscheinlichkeitsdichte ist konstant zwischen den Grenzen $x = a$ und $x = b$

$$\boxed{f(x) = \begin{cases} \dfrac{1}{b-a} & a \le x < b \\ 0 & \text{außerhalb} \end{cases}}\,. \tag{4.17}$$

Sie ist in Abbildung 4.5 gezeigt. Mittelwert und Varianz sind

$$\langle x \rangle = E[x] = \frac{a+b}{2} \qquad V[x] = \sigma^2 = \frac{(b-a)^2}{12}\,.$$

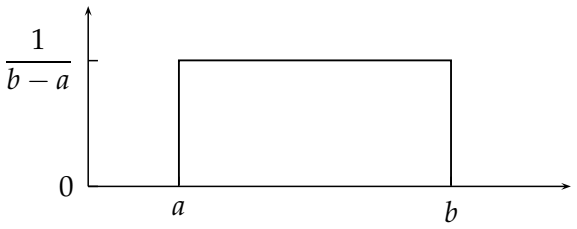


Abbildung 4.5: Die Gleichverteilung mit konstanter Dichte zwischen den Grenzen a und b.

Die Gleichverteilung wird oft $U(a,b)$ geschrieben. Besonders wichtig ist die Verteilung $U(0,1)$ mit den Grenzen 0 und 1, die eine Varianz $1/12$ (Standardabweichung $\sigma = 1/\sqrt{12}$) hat.

4.5.2 Normalverteilung

Die *Normal-* oder *Gauß-Verteilung*[4] ist die wichtigste Wahrscheinlichkeitsdichte wegen ihrer großen Bedeutung in der Praxis (siehe Kapitel 4.6.3). Die Wahrscheinlichkeitsdichte ist

$$\boxed{f(x) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\dfrac{(x-\mu)^2}{2\sigma^2}} \qquad x \in (-\infty, \infty)}\,. \tag{4.18}$$

Die Abbildung 4.6 zeigt eine Seite aus der Originalarbeit von Gauß "Motus Corporum Coelestium", wo er die Gauß-Funktion einführt.

Die Normalverteilung wird von zwei Parametern bestimmt, μ und σ. Durch direkte Rechnung zeigt man, daß $\mu = E[x]$ der Mittelwert ist und $\sigma = \sqrt{V[x]}$ die Standardabweichung. Die

[4] Korrekt: *Gauß'sche Wahrscheinlichkeitsdichte*; dem allgemeinen Brauch folgend wird sie hier auch als *Gauß-Verteilung* bezeichnet.

$\frac{1}{2}k\Delta\Delta$ + Const., sive designando basin logarithmorum hyperbolicorum per e, supponendoque Const. $= \log \varkappa$,

$$\varphi\Delta = \varkappa e^{\frac{1}{2}k\Delta\Delta}$$

Porro facile perspicitur, k necessario negativam esse debere, quo Ω revera fieri possit maximum, quamobrem statuemus $\frac{1}{2}k = -hh$; et quum per theorema elegans primo ab ill. LAPLACE inventum, integrale $\int e^{-hh\Delta\Delta}\mathrm{d}\Delta$, a $\Delta = -\infty$ usque ad $\Delta = +\infty$, fiat $= \frac{\sqrt{\pi}}{h}$ (denotando per π semicircumferentiam circuli cuius radius 1), functio nostra fiet

$$\varphi\Delta = \frac{h}{\sqrt{\pi}} e^{-hh\Delta\Delta}$$

178.

Functio modo eruta omni quidem rigore errorum probabilitates exprimere certo non potest: quum enim errores possibiles semper limitibus certis coërceantur, errorum maiorum probabilitas semper evadere deberet $= 0$, dum formula nostra semper valorem finitum exhibet. Attamen hic defectus, quo omnis functio analytica natura sua laborare debet, ad omnes usus practicos nullius momenti est, quum valor functionis nostrae tam rapide decrescat, quamprimum $h\Delta$ valorem considerabilem acquisivit, ut tuto ipsi 0 aequivalens censeri possit. Praeterea ipsos errorum limites absoluto rigore assignare, rei natura numquam permittet.

Ceterum constans h tamquam mensura praecisionis observationum considerari poterit. Si enim probabilitas erroris Δ in aliquo observationum systemate per $\frac{h}{\sqrt{\pi}} e^{-hh\Delta\Delta}$, in alio vero systemate observationum magis minusve exactarum per $\frac{h'}{\sqrt{\pi}} e^{-h'h'\Delta\Delta}$ exprimi concipitur, exspectatio, in observatione aliqua e systemate priori errorem inter limites $-\delta$ et $+\delta$ contineri, exprimetur per integrale $\int \frac{h}{\sqrt{\pi}} e^{-hh\Delta\Delta}\mathrm{d}\Delta$ a $\Delta = -\delta$ usque ad $\Delta = +\delta$ sumtum, et perinde exspectatio, errorem alicuius observationis e systemate posteriori limites $-\delta'$ et $+\delta'$ non egredi, exprimetur per integrale $\int \frac{h'}{\sqrt{\pi}} e^{-h'h'\Delta\Delta}\mathrm{d}\Delta$ a $\Delta = -\delta'$ usque ad $\Delta = +\delta'$ extensum: ambo autem integralia manifesto aequalia fiunt, quoties habetur $h\delta = h'\delta'$. Quodsi igitur e. g. $h' = 2h$, aeque facile in systemate priori error duplex committi poterit, ac simplex in posteriori, in quo casu observationibus posterioribus secundum vulgarem loquendi morem praecisio duplex tribuitur.

Abbildung 4.6: Einführung der Gauß-Funktion in der Originalarbeit von Gauß: THEORIA MOTUS CORPORUM COELESTIUM in SECTIONIBUS CONICIS SOLEM AMBIENTIUM, Auctore CAROLO FRIDERICO GAUSS, Hamburgi sumtibus Frid. Perthes et J.H.Besser (1809).

Wahrscheinlichkeitsdichte für die Gauß-Verteilung wird mit $N(\mu, \sigma^2)$ bezeichnet. Die Wahrscheinlichkeitsdichte mit Mittelwert $\mu = 0$ und $\sigma^2 = 1$ heißt *standardisierte Gauß-Verteilung*, abgekürzt $N(0, 1)$. Diese Wahrscheinlichkeitsdichte mit $\sigma = 1$ und $\mu = 0$ ist in Abbildung 4.7 gezeigt. Die Gauß-Verteilung kann hergeleitet werden als Grenzfall der Binomialverteilung (Beispiel 4.16) für große Werte von n und r, und auf ähnliche Weise auch als Grenzfall der Poisson-Verteilung für große Werte von μ. Wenn man über die Gauß-Verteilung integriert, erhält man

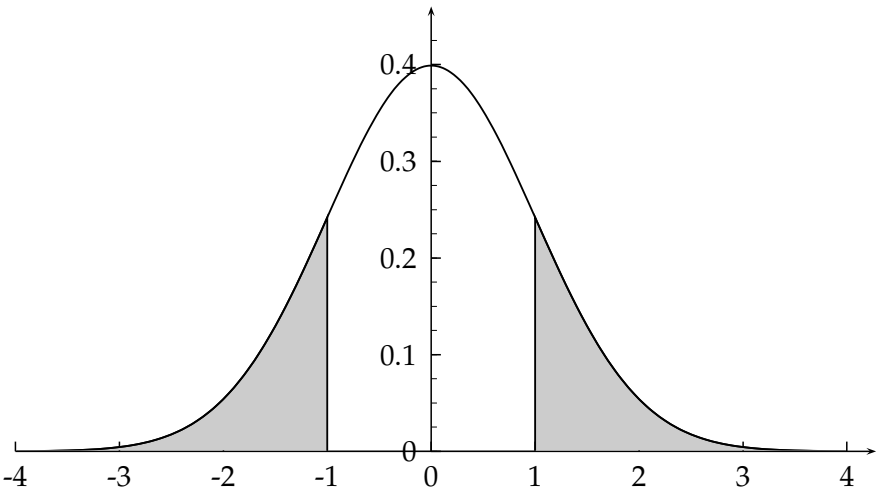


Abbildung 4.7: Die standardisierte Gauß-Verteilung (Mittelwert $\mu = 0$ und Standardabweichung $\sigma = 1$). Die graue Fläche zeigt die Wahrscheinlichkeit für $|x - \mu| \geq 1\sigma$; sie entspricht einer Wahrscheinlichkeit von 31.74 %.

die Wahrscheinlichkeit, daß

$$\begin{array}{lll} |x-\mu| \geq \sigma & (x \text{ außerhalb } \pm 1\sigma) & \text{ist: } 31.74\% \\ |x-\mu| \geq 2\sigma & (x \text{ außerhalb } \pm 2\sigma) & \text{ist: } 4.55\% \\ |x-\mu| \geq 3\sigma & (x \text{ außerhalb } \pm 3\sigma) & \text{ist: } 0.27\%\,. \end{array}$$

Die Abbildung 4.8 zeigt diesen Zusammenhang. Es wird oft vergessen, daß im Mittel rund 32% der Fälle *außerhalb* einer Standardabweichung liegen müssen. Falls also *fast keine*, d.h. viel weniger als 32% der Fälle außerhalb einer Standardabweichung liegen, so ist das verdächtig.

Beispiel 4.15 Vollmond und Verkehrsunfälle.

Passieren mehr Verkehrsunfälle an Tagen mit Vollmond? Um das zu ergründen, wird die Zahl der Unfälle in vielen deutschen Städten verglichen, und man findet, daß in Hamburg die mittlere Zahl von Unfällen an Tagen mit Vollmond 10.0 mit einer Standardabweichung von 1.0 ist, und an den anderen Tagen ist sie 7.0 mit vernachlässigbar kleinem Fehler. Dies scheint ein sehr signifikanter Effekt (3σ) und erscheint in den Zeitungen, da die Wahrscheinlichkeit kleiner als 0.14% ist, daß man diese Zahl von Unfällen oder eine noch größere Zahl durch Zufall erhält. Aber dies hat in Wirklichkeit nichts zu bedeuten. Falls man zum Beispiel 200 Städte getestet hat, so ist die Wahrscheinlichkeit, daß eine Stadt oder mehrere rein statistisch um mehr als 3 Standardabweichungen vom Mittelwert nach oben abweichen, nach der Tabelle 4.1 gleich $1 - 0.9987^{200} = 0.23$, und diese Wahrscheinlichkeit ist nicht klein.

Volle Breite auf halber Höhe (FWHM). Dieser Begriff ist oft nützlich, um auf einfache Weise die Standardabweichung einer Gaußkurve zu schätzen. Die Verbindung zwischen FWHM und

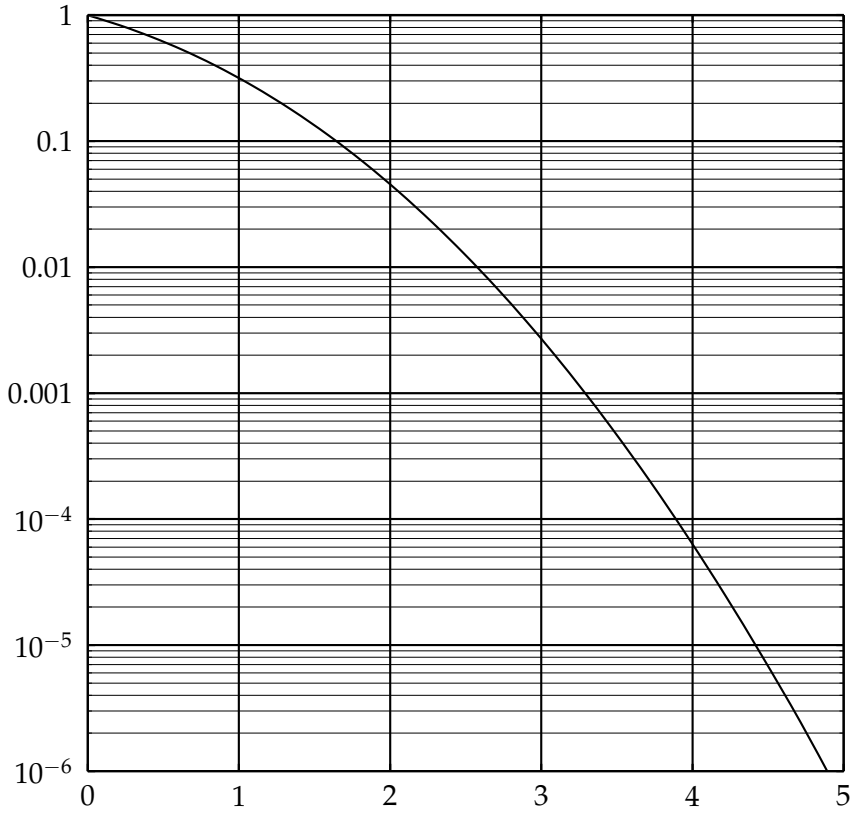


Abbildung 4.8: Wahrscheinlichkeit für eine Abweichung von mehr als r Standardabweichungen nach oben oder unten bei einer Gauß-Verteilung, aufgetragen gegen die Anzahl r der Standardabweichungen.

σ ist

$$\text{FWHM} = 2\sigma\sqrt{2\ln 2} = 2.355\,\sigma\ .$$

Integrierte Gaußfunktion. Sie wird mit $\Phi(x)$ bezeichnet,

$$\Phi(x) = \frac{1}{\sqrt{2\pi}\sigma}\int_{-\infty}^{x} e^{-(t-\mu)^2/2\sigma^2}\,\mathrm{d}t\,.$$

Tabelle 4.1 zeigt die integrierte Verteilungsfunktion $\Phi(x)$ für die standardisierte Gauß-Wahrscheinlichkeitsdichte, d.h. für $\mu = 0$ und $\sigma = 1$. Falls die Wahrscheinlichkeitsdichte nicht standardisiert ist und x_{ns} der Argumentwert ist, muß man die Tabelle beim Wert $(x_{ns} - \mu)/\sigma$ benutzen. Die integrierte Verteilungsfunktion kann durch die Gauß'sche Fehlerfunktion $\mathrm{erf}(x)$ ausgedrückt werden,

$$\mathrm{erf}(x) = \frac{2}{\sqrt{\pi}}\int_0^x e^{-t^2}\,\mathrm{d}t\ .$$

mit $\mathrm{erf}(0) = 0$ und $\mathrm{erf}(\infty) = 1$. Die integrierte Verteilungsfunktion ist dann

$$\Phi(x) = \frac{1}{2}\left(1 + \mathrm{erf}\left(\frac{x-\mu)}{\sqrt{2}\sigma}\right)\right)\ . \tag{4.19}$$

z	0.00	0.01	0.02	0.03	0.04	0.05	0.06	0.07	0.08	0.09
0.0	0.5000	0.5040	0.5080	0.5120	0.5160	0.5199	0.5239	0.5279	0.5319	0.5359
0.1	0.5398	0.5438	0.5478	0.5517	0.5557	0.5596	0.5636	0.5675	0.5714	0.5753
0.2	0.5793	0.5832	0.5871	0.5910	0.5948	0.5987	0.6026	0.6064	0.6103	0.6141
0.3	0.6179	0.6217	0.6255	0.6293	0.6331	0.6368	0.6406	0.6443	0.6480	0.6517
0.4	0.6554	0.6591	0.6628	0.6664	0.6700	0.6736	0.6772	0.6808	0.6844	0.6879
0.5	0.6915	0.6950	0.6985	0.7019	0.7054	0.7088	0.7123	0.7157	0.7190	0.7224
0.6	0.7257	0.7291	0.7324	0.7357	0.7389	0.7422	0.7454	0.7486	0.7517	0.7549
0.7	0.7580	0.7611	0.7642	0.7673	0.7703	0.7734	0.7764	0.7794	0.7823	0.7852
0.8	0.7881	0.7910	0.7939	0.7967	0.7995	0.8023	0.8051	0.8078	0.8106	0.8133
0.9	0.8159	0.8186	0.8212	0.8238	0.8264	0.8289	0.8315	0.8340	0.8365	0.8389
1.0	0.8413	0.8438	0.8461	0.8485	0.8508	0.8531	0.8554	0.8577	0.8599	0.8621
1.1	0.8643	0.8665	0.8686	0.8708	0.8729	0.8749	0.8770	0.8790	0.8810	0.8830
1.2	0.8849	0.8869	0.8888	0.8907	0.8925	0.8944	0.8962	0.8980	0.8997	0.9015
1.3	0.9032	0.9049	0.9066	0.9082	0.9099	0.9115	0.9131	0.9147	0.9162	0.9177
1.4	0.9192	0.9207	0.9222	0.9236	0.9251	0.9265	0.9279	0.9292	0.9306	0.9319
1.5	0.9332	0.9345	0.9357	0.9370	0.9382	0.9394	0.9406	0.9418	0.9429	0.9441
1.6	0.9452	0.9463	0.9474	0.9484	0.9495	0.9505	0.9515	0.9525	0.9535	0.9545
1.7	0.9554	0.9564	0.9573	0.9582	0.9591	0.9599	0.9608	0.9616	0.9625	0.9633
1.8	0.9641	0.9649	0.9656	0.9664	0.9671	0.9678	0.9686	0.9693	0.9699	0.9706
1.9	0.9713	0.9719	0.9726	0.9732	0.9738	0.9744	0.9750	0.9756	0.9761	0.9767
2.0	0.9772	0.9778	0.9783	0.9788	0.9793	0.9798	0.9803	0.9808	0.9812	0.9817
2.1	0.9821	0.9826	0.9830	0.9834	0.9838	0.9842	0.9846	0.9850	0.9854	0.9857
2.2	0.9861	0.9864	0.9868	0.9871	0.9875	0.9878	0.9881	0.9884	0.9887	0.9890
2.3	0.9893	0.9896	0.9898	0.9901	0.9904	0.9906	0.9909	0.9911	0.9913	0.9916
2.4	0.9918	0.9920	0.9922	0.9925	0.9927	0.9929	0.9931	0.9932	0.9934	0.9936
2.5	0.9938	0.9940	0.9941	0.9943	0.9945	0.9946	0.9948	0.9949	0.9951	0.9952
2.6	0.9953	0.9955	0.9956	0.9957	0.9959	0.9960	0.9961	0.9962	0.9963	0.9964
2.7	0.9965	0.9966	0.9967	0.9968	0.9969	0.9970	0.9971	0.9972	0.9973	0.9974
2.8	0.9974	0.9975	0.9976	0.9977	0.9977	0.9978	0.9979	0.9979	0.9980	0.9981
2.9	0.9981	0.9982	0.9982	0.9983	0.9984	0.9984	0.9985	0.9985	0.9986	0.9986
3.0	0.9987	0.9987	0.9987	0.9988	0.9988	0.9989	0.9989	0.9989	0.9990	0.9990
3.1	0.9990	0.9991	0.9991	0.9991	0.9992	0.9992	0.9992	0.9992	0.9993	0.9993
3.2	0.9993	0.9993	0.9994	0.9994	0.9994	0.9994	0.9994	0.9995	0.9995	0.9995
3.3	0.9995	0.9995	0.9995	0.9996	0.9996	0.9996	0.9996	0.9996	0.9996	0.9997
3.4	0.9997	0.9997	0.9997	0.9997	0.9997	0.9997	0.9997	0.9997	0.9997	0.9998

Tabelle 4.1: Tabelle der integrierten standardisierten Gauß-Verteilung $F(z) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{z} e^{-x^2/2} \mathrm{d}x$.

Die Fehlerfunktion ist auf den meisten Rechnersystemen vorhanden. Eine Näherung für grosse Werte von x ist

$$1 - \mathrm{erf}(x) \approx \frac{2}{\pi} \cdot \frac{e^{-x^2}}{2x}.$$

Beispiel 4.16 Die Gauß-Verteilung als Grenzfall der Binomialverteilung für sehr große n.

Die Binomialverteilung wird um ihren Maximalwert entwickelt, der nahe ihrem Mittelwert $\langle r \rangle = n \cdot p$ ist

$$\begin{aligned} \ln P(r) &= \ln \left(p^r \cdot q^{n-r} \cdot \frac{n!}{r! \cdot (n-r)!} \right) \\ &= r \ln p + (n-r) \ln q + \ln n! - \ln r! - \ln(n-r)! \, . \end{aligned}$$

Für große Werte von n kann das Maximum mit Stirlings Formel berechnet werden

$$\frac{\mathrm{d} \ln P(r)}{\mathrm{d}r} = 0 \approx \ln p - \ln q - \ln r + \ln(n-r)$$

(mit der Näherung $\mathrm{d} \ln x! / \mathrm{d}x \approx \ln x$, und deshalb $r_{max} = np = \langle r \rangle$).

Entwicklung um $\langle r\rangle = np$:

$$\ln P(r) = \ln P(\langle r\rangle) + (r - \langle r\rangle)\frac{d\ln P(r)}{\mathrm{d}r}|_{r=\langle r\rangle} + \frac{1}{2!}(r - \langle r\rangle)^2 \cdot \frac{d^2 \ln P(r)}{\mathrm{d}r^2}|_{r=\langle r\rangle} + \dots$$

Der lineare Term verschwindet, weil im Maximum die erste Ableitung null ist. Die zweite Ableitung ist $-1/(npq) = -1/(np(1-p))$. Daraus folgt

$$P(r) = \text{konst} \cdot e^{-(r-\langle r\rangle)^2/(2np(1-p))} .$$

Dies hat die Form einer Gauß-Verteilung mit Mittelwert $\mu = \langle r\rangle$ und $\sigma^2 = np(1-p)$ (siehe Gleichung (4.15)). Da die Poisson-Verteilung ein Grenzfall der Binomialverteilung für große Werte von n ist, kann sie ebenfalls mit einer Gauß-Verteilung für große Werte von μ angenähert werden. Beispiele hierfür sind in Abbildung 4.9 und Abbildung 4.10 zu sehen. Der Vergleich zeigt, daß für nicht zu kleine Werte ($\mu \geq 10$) die Gauß-Verteilung eine gute Näherung an die Poisson-Verteilung ist, außer weit außerhalb des Maximums.

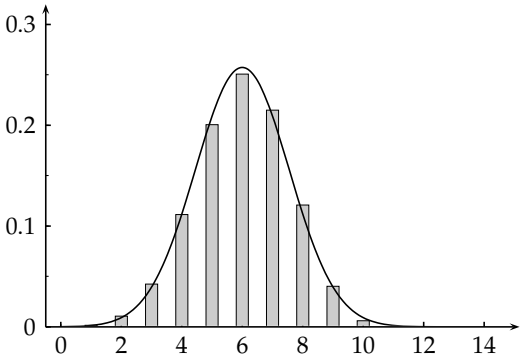


Abbildung 4.9: Binomialverteilung für $n = 10$, $p = 0.6$, $np = 6$, $\sigma = \sqrt{np(1-p)} = 1.55$ zusammen mit der Näherung durch die Gauß-Verteilung $1/(\sqrt{2\pi}\sigma)\exp(-(r-np)^2/2\sigma^2)$.

Beispiel 4.17 Münzenwerfen.

Wenn man eine Münze $n = 1000$ mal wirft, welches ist die Wahrscheinlichkeit für *genau* $r = 510$ mal *Kopf*, angenommen $p = 0.5$ für die Wahrscheinlichkeit von *Kopf*? Die Binomialverteilung ist dafür zuständig, aber numerisch schwierig anzuwenden. Die Näherung durch die Gauß-Verteilung liefert

$$f(r)\Delta r = \frac{1}{\sqrt{2\pi}\sigma} e^{-(r-\mu)^2/2\sigma^2} \Delta r .$$

Mit $\Delta r = 1$, $r = 510$, $\langle r\rangle = \mu = 500$, $\sigma^2 = np(1-p) = 250$ erhält man für die Wahrscheinlichkeit $f(r)\Delta r = 2.1\%$.

4.5.3 Gammaverteilung

Ihre Wahrscheinlichkeitsdichte ist gegeben durch (siehe Beispiel 4.18)

$$\boxed{f(x|k) = \frac{x^{k-1}e^{-x}}{\Gamma(k)}} .$$

Für ganzzahlige Werte von k mit $k \geq 1$ gibt die Gamma-Wahrscheinlichkeitsdichte die Verteilung der Wartezeit $t = x$ bis zum k-ten Ereignis in einem Poisson-verteilten Prozess mit

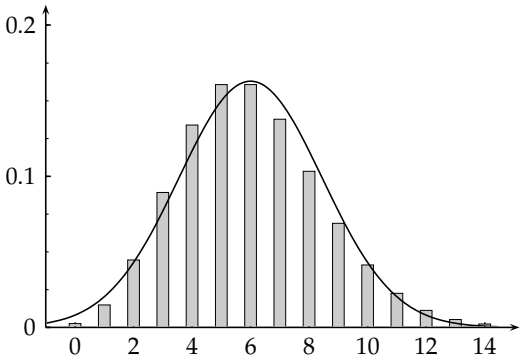


Abbildung 4.10: Die Poisson-Verteilung für $\mu = 6$, $\sigma = \sqrt{6} = 2.45$, zusammen mit der Näherung durch die Gauß-Verteilung $1/(\sqrt{2\pi}\sigma)\exp(-(r-\mu)^2/2\sigma^2)$.

Mittelwert $\mu = 1$ an. Die Verallgemeinerung für andere Werte von μ ist

$$\boxed{f(x|k,\mu) = \frac{x^{k-1}\mu^k e^{-\mu x}}{\Gamma(k)}} \; .$$

Der Parameter k beeinflußt die Form der Verteilung, während μ nur ein Skalenparameter ist. Für $\mu = 1/2$ und $k = n/2$ hat die Gammaverteilung die Form einer χ^2-Verteilung mit n Freiheitsgraden.

Beispiel 4.18 Zeitdifferenz zwischen zwei Ereignissen.

Ziel ist die Berechnung der Wahrscheinlichkeitsdichte $f(t)$ für die Zeitdifferenz t zwischen zwei Ereignissen, wobei die Ereignisse zufällig mit einer mittleren Rate λ auftreten. Als Beispiel kann der radioaktive Zerfall mit einer mittleren Zerfallsrate λ dienen. Die Wahrscheinlichkeit für *ein* Ereignis im Zeitintervall δt und *kein* Ereignis im Intervall t ist

$$f(t) \cdot \delta t = P_0(t)\,(\lambda\,\delta t + o(\delta t)) \; .$$

Indem man das Resultat für $P_0(t)$ aus Beispiel 4.12 einsetzt, erhält man

$$f(t) = \lambda\, e^{-\lambda t} \; .$$

Die Wartezeit zwischen zwei aufeinanderfolgenden Ereignissen folgt also einer Exponentialverteilung. Dies ist die Wahrscheinlichkeitsdichte, die man für die Länge von Lücken (auch räumliche) zwischen zufällig verteilten Ereignissen erwartet. Eine paradoxe Konsequenz ist, daß die wahrscheinlichste Zeitdifferenz zwischen zwei radioaktiven Zerfällen null ist, unabhängig vom Wert der mittleren Lebensdauer. Diese Wahrscheinlichkeitsdichte zeigt Abbildung 4.11.

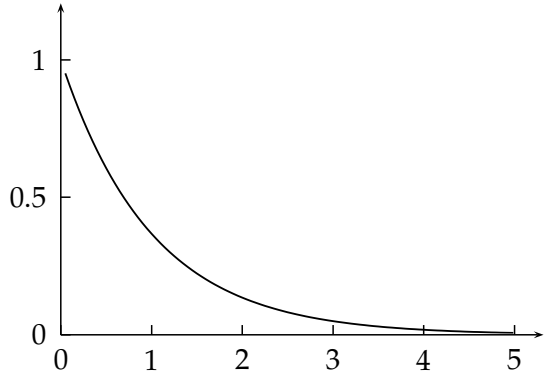


Abbildung 4.11: Die exponentielle Dichte $f(x;\lambda) = \lambda\exp(-\lambda t)$ für $\lambda = 1$.

Allgemein erhält man die Zeit zwischen dem ersten und dem k-ten Ereignis, indem man ein Ereignis im Intervall δt und $k-1$ Ereignisse im Intervall von 0 bis t fordert. Mit der Poisson-Wahrscheinlichkeit für $k-1$ Ereignisse erhält man

$$f_k(t) = \frac{(\lambda t)^{k-1}}{(k-1)!} e^{-\lambda t} \cdot \lambda = \frac{t^{k-1}\lambda^k e^{-\lambda t}}{(k-1)!} \,.$$

Dies führt also auf eine Gammaverteilung.

4.5.4 Charakteristische Funktionen

Für eine Zufallsvariable mit Wahrscheinlichkeitsdichte $f(x)$ definiert man die *charakteristische Funktion* $\phi(t)$ als den Erwartungswert von e^{itx},

$$\phi(t) = E\left[e^{itx}\right] = \int_{-\infty}^{+\infty} e^{itx} \cdot f(x)\,\mathrm{d}x \,. \tag{4.20}$$

Die inverse Operation ist

$$f(x) \;= \frac{1}{2\pi} \int_{-\infty}^{+\infty} e^{-itx} \cdot \phi(t)\,\mathrm{d}t \,.$$

Die charakteristische Funktion sowie ihre erste und zweite Ableitung für den speziellen Wert $t = 0$ lassen sich leicht berechnen:

$$\phi(0) = 1 \quad \text{(Normierung)}$$

$$\frac{\mathrm{d}\phi(0)}{\mathrm{d}t} = \int_{-\infty}^{+\infty} ixe^{itx} f(x)\,\mathrm{d}x \bigg|_{t=0} = i\langle x\rangle \tag{4.21}$$

$$\frac{\mathrm{d}^2\phi(0)}{\mathrm{d}t^2} = \int_{-\infty}^{+\infty} -x^2 e^{itx} f(x)\,\mathrm{d}x \bigg|_{t=0} = -E\left[x^2\right] = -\left(\sigma^2 + \langle x\rangle^2\right) \,. \tag{4.22}$$

Beispiel 4.19 Charakteristische Funktion einer Gauß-Wahrscheinlichkeitsdichte.

Die charakteristische Funktion einer Gauß-Wahrscheinlichkeitsdichte der Form $1/(\sigma\sqrt{2\pi})e^{-x^2/2\sigma^2}$ mit Mittelwert $\mu = 0$ ist

$$\begin{aligned}
\phi_{\mathrm{G}}(t) &= \frac{1}{\sigma\sqrt{2\pi}} \int_{-\infty}^{+\infty} e^{itx} e^{-x^2/2\sigma^2}\,\mathrm{d}x \\
&= \frac{1}{\sigma\sqrt{2\pi}} \int_{-\infty}^{+\infty} e^{-((x-it\sigma^2)^2 - (it\sigma^2)^2)/2\sigma^2}\,\mathrm{d}x \\
&= \frac{1}{\sigma\sqrt{2\pi}} e^{-t^2\sigma^2/2} \int_{-\infty}^{+\infty} e^{-(x-it\sigma^2)^2/2\sigma^2}\,\mathrm{d}x \,.
\end{aligned}$$

Daraus folgt

$$\phi_{\mathrm{G}}(t) = e^{-t^2\sigma^2/2} \,. \tag{4.23}$$

Die charakteristische Funktion ist also wieder eine Gaußfunktion in der Variablen t mit Varianz $1/\sigma^2$.

Die charakteristische Funktion hat eine wichtige Bedeutung bei der Faltung zweier Variablen: Die charakteristische Funktion der Faltung zweier Variablen erhält man als das Produkt ihrer charakteristischen Funktionen.

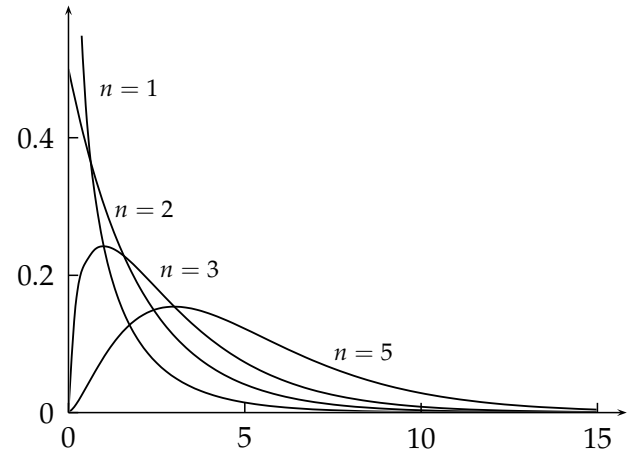


Abbildung 4.12: Die χ^2-Verteilungen für $n = 1$, 2, 3 und 5 Freiheitsgrade.

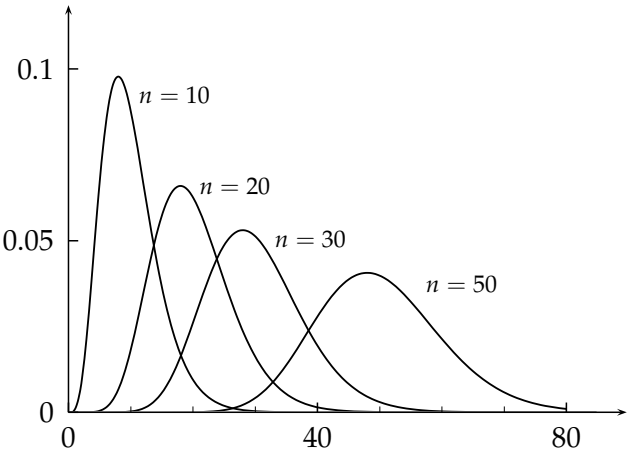


Abbildung 4.13: Die χ^2-Verteilungen für $n = 10$, 20, 30 und 50 Freiheitsgrade.

4.5.5 χ^2-Verteilung

Falls $x_1,\ x_2, \ldots x_n$ unabhängige Zufallsvariable sind, die alle einer Gauß-Wahrscheinlichkeitsdichte folgen mit Mittelwert 0 und Varianz 1, so folgt die Summe $u = \chi^2$ der n Quadrate

$$u = \chi^2 = \sum_{i=1}^{n} x_i^2$$

einer χ^2-Verteilung $f_n(u) = f_n(\chi^2)$ mit n Freiheitsgraden. Die Wahrscheinlichkeitsdichte ist

$$\boxed{f_n(u) \ = \frac{\frac{1}{2}\left(\frac{u}{2}\right)^{n/2-1} e^{-u/2}}{\Gamma(n/2)}} \,. \tag{4.24}$$

Die Wahrscheinlichkeitsdichte $f_n(u) = f_n(\chi^2)$ hat ein Maximum bei $(n-2)$. Der Mittelwert der χ^2-Verteilung ist n und die Varianz ist $2n$

$$\langle u \rangle = \langle \chi^2 \rangle = n \qquad V\left[u\right] = V\left[\chi^2\right] = 2n \,. \tag{4.25}$$

Die χ^2-Verteilung spielt bei statistischen Tests eine große Rolle. Die Abbildungen 4.12 und 4.13 zeigen die χ^2-Verteilung für verschiedene Werte von n, der Zahl der Freiheitgrade (degrees of freedom, *d.o.f*). Die integrierte Verteilung $F(u) = \int_0^u f_n(v)\mathrm{d}v$ ist in den Abbildungen 4.14 und 4.15 zu sehen.

Um die Wahrscheinlichkeitsdichte von χ^2 herzuleiten, betrachtet man zunächst den Spezialfall $n = 1$, $u = x^2$. Die Wahrscheinlichkeitsdichte in der Variablen x ist

$$f_x(x) = \frac{1}{\sqrt{2\pi}} e^{-x^2/2} \,.$$

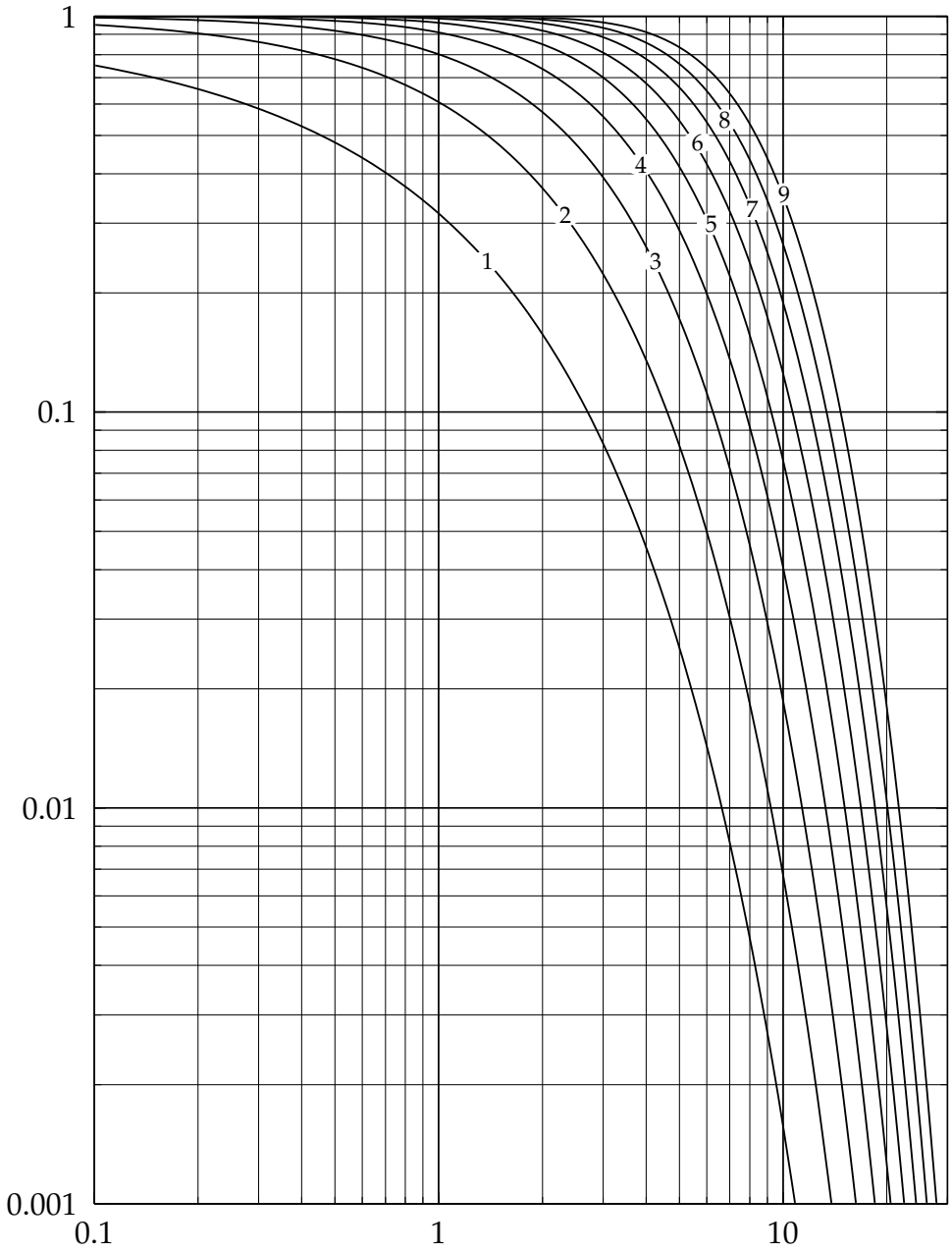


Abbildung 4.14: Die Größe $1 - P = 1 - F(\chi^2) = 1 - \int_0^{\chi^2} f_n(v)\,\mathrm{d}v$ gegen χ^2 für n Freiheitsgrade ($n = 1, 2, \ldots 9$). Für ein gegebenes n gibt die Kurve die Wahrscheinlichkeit an, daß bei einer Stichprobe ein Wert mindesten so groß wie χ^2 auftritt, zum Beispiel wird für $n = 9$ ein Wert $\chi^2 > 10$ in 35% der Stichproben auftreten.

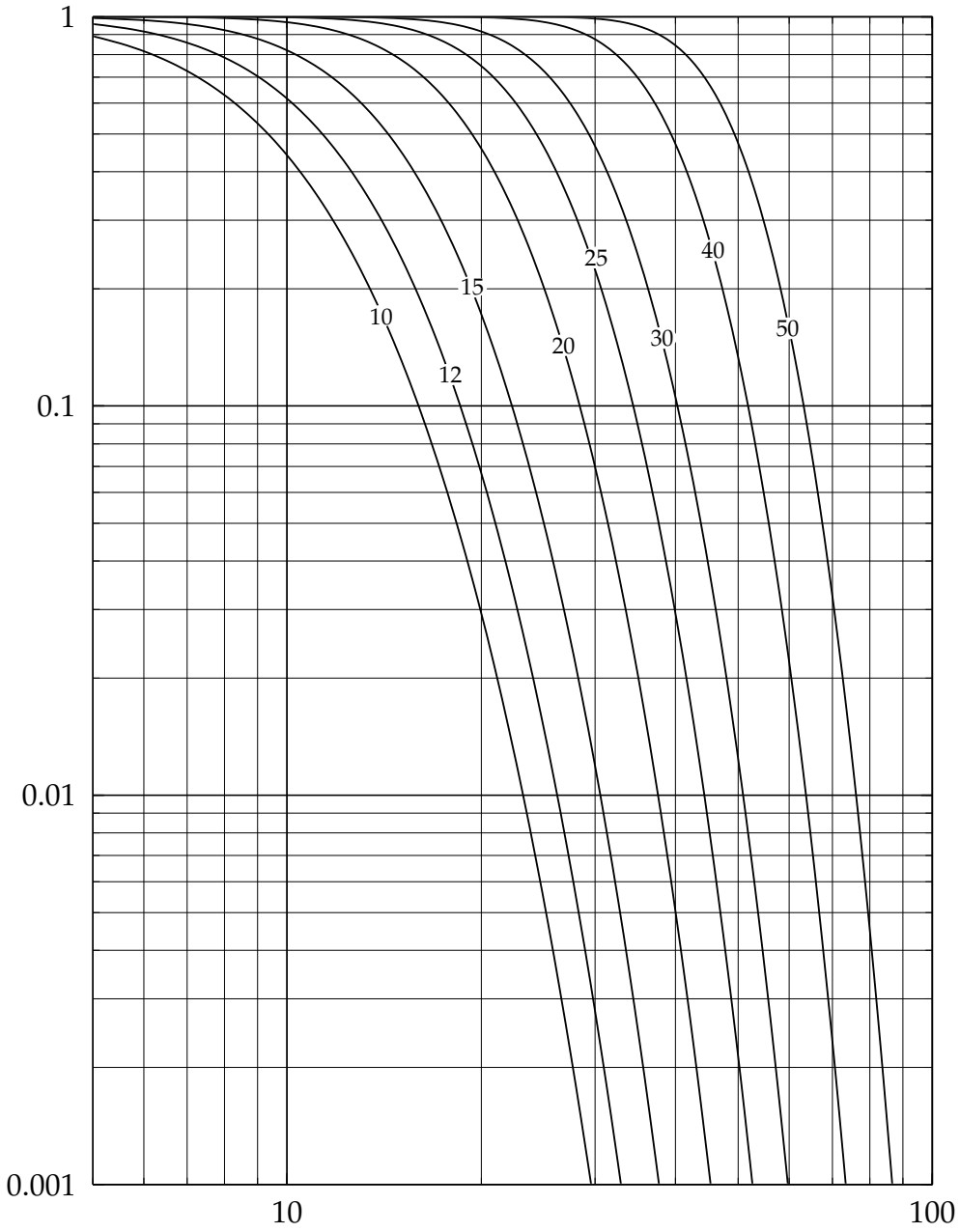


Abbildung 4.15: Die Größe $1 - P$ gegen χ^2 für n Freiheitsgrade ($n = 10$, 12, 15, 20, 25, 30, 40 und 50).

Mit der Ersetzung $x^2 = u$, $\mathrm{d}x = (1/2\sqrt{u})\mathrm{d}u$ erhält man die Wahrscheinlichkeitsdichte in u

$$f_1(u)\mathrm{d}u = \frac{2}{2\sqrt{2u\pi}} e^{-u/2}\mathrm{d}u\,.$$

Der zusätzliche Faktor 2 berücksichtigt, daß es zwei Werte von x gibt für jeden Wert von u: $x = \pm\sqrt{u}$. Die Wahrscheinlichkeitsdichte $f_1(u)$ hat die Form

$$f_1(u) = \frac{u^\alpha}{\alpha!\beta^{\alpha+1}} e^{-u/\beta}$$

mit $\alpha = -\frac{1}{2}$ und $\beta = 2$. Die charakteristische Funktion von $f_1(u)$ ist (siehe Kapitel 4.5.4):

$$\phi_1(t) = \frac{1}{\alpha!\beta^{\alpha+1}} \int_0^\infty e^{itu}\; u^\alpha e^{-u/\beta}\,\mathrm{d}u\,.$$

Mit der Ersetzung $y = u(1/\beta - it)$ wird

$$\phi_1(t) = \left(\left(\frac{1}{\beta} - it\right)^{-1-\alpha} / (\alpha!\beta^{\alpha+1})\right) \cdot \int_0^\infty e^{-y} y^\alpha\,\mathrm{d}y\,.$$

Das Integral ist eine Gammafunktion und gleich $\alpha!$. Dann wird

$$\phi_1(t) = \left(\frac{1}{\beta} - it\right)^{-1-\alpha} / \beta^{\alpha+1}\,.$$

Für $n > 1$ ist die χ^2-Verteilung eine Faltung von n Verteilungen $f_1(u)$, und deren charakteristische Funktion $\phi_n(t)$ ist deshalb die n-te Potenz von $\phi_1(t)$ (siehe Kapitel 4.10 und Kapitel 4.6.3),

$$\phi_n(t) = (\phi_1(t))^n = \beta^{-n(\alpha+1)} \cdot \left(\frac{1}{\beta} - it\right)^{-n(\alpha+1)}\,.$$

Diese Funktion hat, bis auf einen konstanten Faktor, dieselbe Form wie $\phi_1(t)$, falls man α durch α_n ersetzt gemäß $-n(\alpha+1) = -1-\alpha_n$ oder $\alpha_n = n(\alpha+1) - 1$, und mit $\alpha = -1/2$ erhält man $\alpha_n = n/2 - 1$. Deshalb hat die Wahrscheinlichkeitsdichte, welche zu $\phi_n(t)$ gehört, dieselbe Form wie $f_1(u)$, falls man α durch α_n ersetzt. Das Resultat ist

$$f_n(u) = \frac{u^{n/2-1}}{(n/2-1)!2^{n/2}} e^{-u/2}\,,$$

welches identisch mit Gleichung (4.24) ist. Mittelwert und Varianz erhält man mit Gleichung (4.21) und Gleichung (4.22):

$$\frac{\mathrm{d}\phi_n(0)}{\mathrm{d}t} = i\langle u\rangle = 2in/2 \qquad \text{oder} \quad \langle\chi^2\rangle = n\,,$$

und

$$\frac{d^2\phi_n(0)}{\mathrm{d}t^2} = -(\sigma^2 + n^2) = -n^2 - 2n\,,$$

woraus $\sigma^2 = 2n$ folgt.

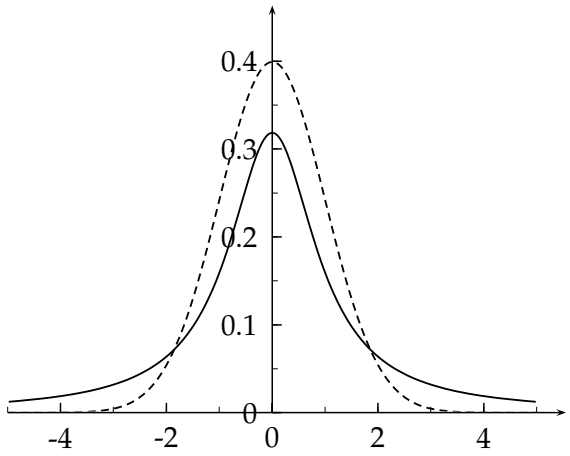


Abbildung 4.16: Die Cauchy-Wahrscheinlichkeitsdichte $f(x)$ mit FWHM $= 2$, die identisch mit der t-Verteilung mit einem Freiheitsgrad ist. Die zum Vergleich eingezeichnete standardisierte Gauß-Verteilung $N(0,1)$ fällt für große Werte von $|x|$ viel schneller ab.

4.5.6 Cauchy-Verteilung

Die Cauchy-Verteilung hat die Dichte

$$\boxed{f(x) = \frac{1}{\pi}\frac{1}{1+x^2}}.$$

Für große Werte von x nimmt sie nur langsam ab, und deshalb erfordert sie Vorsicht in der Anwendung (siehe Abbildung 4.16).

Der Erwartungswert von x ist undefiniert, und alle Momente (Varianz, Schiefe, usw.) sind divergent, falls man die Grenzen bis $\pm\infty$ gehen läßt. In physikalischen Anwendungen wird diese Wahrscheinlichkeitsdichte oft *Breit-Wigner*-Verteilung genannt

$$f(x) = \frac{1}{\pi}\frac{\Gamma/2}{(x-x_0)^2+\Gamma^2/4}.$$

Sie beschreibt die Energieverteilung nahe einer Resonanz. In der Tat ist diese Verteilung die Fouriertransformierte einer Exponentialverteilung; für Energiezustände, die exponentiell mit der Zeit zerfallen, wird die Energieverteilung durch die Breit-Wigner-Kurve gegeben. Die Verteilung ist symmetrisch um das Maximum bei x_0; der Parameter Γ ist die volle Breite auf halber Höhe, kurz FWHM (full width at half maximum) genannt. In der Wirklichkeit reicht die Wahrscheinlichkeitsdichte nur bis zu einem endlichen Wert der Variablen, so daß die Integrale nicht divergieren.

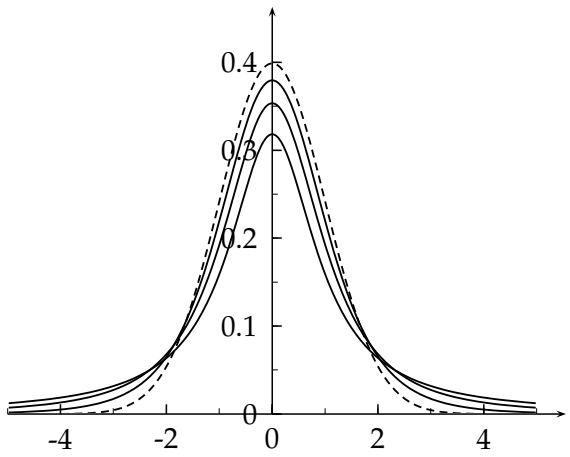


Abbildung 4.17: Die Studentschen t-Verteilungen für $n = 1$, 2 und 5 Freiheitsgrade. Zum Vergleich ist gestrichelt die standardisierte Gauß-Verteilung $N(0,1)$ eingezeichnet, der sich die t-Verteilung mit wachsender Zahl der Freiheitsgrade n annähert.

4.5.7 *t*-Verteilung

Die *t*-Verteilung tritt auf bei Tests der statistischen Verträglichkeit eines Stichproben-Mittelwertes $\bar{x}$ mit einem vorgegebenen Mittelwert μ, oder der statistischen Verträglichkeit zweier Stichproben-Mittelwerte. Die Wahrscheinlichkeitsdichte der *t*-Verteilung ist gegeben durch

$$\boxed{f_n(t) = \frac{1}{\sqrt{n\pi}} \frac{\Gamma((n+1)/2)}{\Gamma(n/2)} \left(1 + \frac{t^2}{n}\right)^{-(n+1)/2}} \qquad -\infty < t < +\infty\,. \tag{4.26}$$

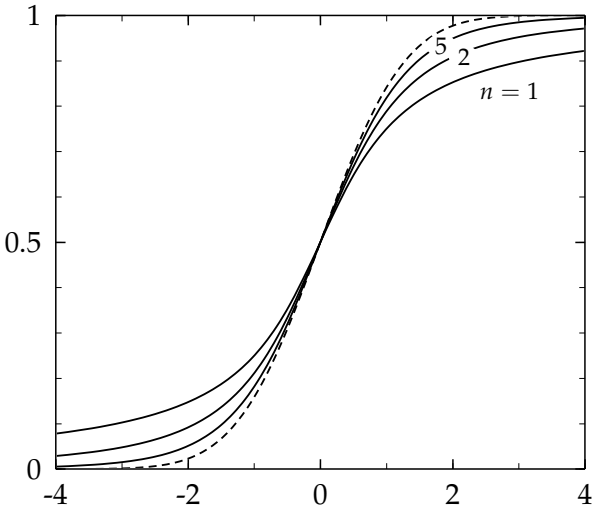


Abbildung 4.18: Die integrierten Student-schen *t*-Verteilungen $\int_{-\infty}^{t} f(t)\,\mathrm{d}t$ für $n = 1, 2$ und 5 Freiheitsgrade. Zum Vergleich ist gestrichelt die integrierte standardisierte Gauß-Verteilung $N(0,1)$ eingezeichnet, der sich die *t*-Verteilung mit wachsender Zahl der Freiheitsgrade n annähert.

Eine Anwendung der t-Verteilung bei der Prüfung von Hypothesen findet man in Kapitel 9.1.3. Die *t*-Verteilung ist in Abbildung 4.17 gezeigt. Sie ist symmetrisch um null; für $n = 1$ ist sie gleich der Cauchy-Verteilung und für große n geht sie gegen eine Gauß-Verteilung. Eine Ableitung der Gleichung (4.26) findet man zum Beispiel bei [3]. Abbildungen 4.18 und 4.19 zeigen die integrierte t-Verteilung für verschiedene Zahlen von Freiheitsgraden. Man beachte, dass in Abbildung 4.19 die Wahrscheinlichkeit, dass $\mid t \mid > t_0$ ist, aufgetragen ist, und wegen $\mid t \mid$ ist dies die **zweiseitige** Wahrscheinlichkeit. Die Tabelle 4.2 zeigt die Quantile der t-Verteilung.

Für die vorgegebene Wahrscheinlichkeit P, einen Wert $\leq t$ zu erhalten, sind die entsprechenden *t*-Werte in der Tabelle gegeben für verschiedene Werte von n. Zum Beispiel ist die Wahrscheinlichkeit 90%, bei $n = 4$ Freiheitsgraden einen Wert t kleiner als 1.533 zu erhalten.

4.5.8 *F*-Verteilung

Gegeben sind n_1 Stichprobenwerte einer Zufallsvariablen x und n_2 Stichprobenwerte derselben Zufallsvariablen. Die beste Schätzung der Varianzen aus den beiden Datenkollektionen seien s_1^2 und s_2^2 (siehe Gleichung (4.28)). Die Zufallszahl F, definiert durch $F = s_1^2/s_2^2$, folgt dann einer *F*-Verteilung mit (n_1, n_2) Freiheitsgraden. Es ist eine Konvention, den größeren der beiden Werte s_1^2, s_2^2 im Zähler der Gleichung für F zu haben, so daß F immer größer als eins ist. Die Wahrscheinlichkeitsdichte von F ist gegeben durch

$$\boxed{f(F) = \left(\frac{n_1}{n_2}\right)^{n_1/2} \cdot \frac{\Gamma((n_1+n_2)/2)}{\Gamma(n_1/2)\Gamma(n_2/2)} \cdot F^{(n_1-2)/2} \left(1 + \frac{n_1}{n_2}F\right)^{-(n_1+n_2)/2}}\,.$$

Für große Werte von n_1 und n_2 geht die *F*-Verteilung asymptotisch gegen die Gauß-Verteilung, aber ziemlich langsam. In der Praxis benötigt man die Quantile Q der F-Verteilung für eine Konfidenz α, gegeben durch $\int_1^Q f(F)\mathrm{d}F = \alpha$, d.h. die Wahrscheinlichkeit, einen Wert $F \leq Q$

$P =$	0.750	0.840	0.900	0.950	0.980	0.990	0.995
$n =$							
1	1.000	1.819	3.078	6.31	15.89	31.82	63.66
2	0.816	1.312	1.886	2.92	4.85	6.96	9.92
3	0.765	1.189	1.638	2.35	3.48	4.54	5.84
4	0.741	1.134	1.533	2.13	3.00	3.75	4.60
5	0.727	1.104	1.476	2.01	2.76	3.36	4.03
6	0.718	1.084	1.440	1.94	2.61	3.14	3.71
7	0.711	1.070	1.415	1.89	2.52	3.00	3.50
8	0.706	1.060	1.397	1.86	2.45	2.90	3.35
9	0.703	1.052	1.383	1.83	2.40	2.82	3.25
10	0.700	1.046	1.372	1.81	2.36	2.76	3.17
12	0.695	1.037	1.356	1.78	2.30	2.68	3.05
15	0.691	1.029	1.341	1.75	2.25	2.60	2.95
20	0.687	1.020	1.325	1.72	2.20	2.53	2.84

Tabelle 4.2: Quantile der t-Verteilung, $P = \int_{-\infty}^{t} f_n(x)\mathrm{d}x$

$n_1 =$	1	2	3	4	5	10	15
$n_2 =$							
1	3.31	4.38	4.83	5.07	5.22	5.54	5.65
2	1.72	2.13	2.27	2.35	2.40	2.49	2.53
3	1.41	1.71	1.80	1.85	1.88	1.93	1.95
4	1.29	1.54	1.61	1.65	1.67	1.71	1.72
5	1.22	1.44	1.51	1.54	1.55	1.58	1.59
6	1.18	1.39	1.45	1.47	1.48	1.50	1.51
7	1.15	1.35	1.40	1.42	1.43	1.45	1.45
8	1.12	1.32	1.37	1.39	1.40	1.41	1.41
9	1.11	1.30	1.35	1.36	1.37	1.38	1.38
10	1.10	1.28	1.33	1.34	1.35	1.36	1.35
12	1.08	1.25	1.30	1.31	1.32	1.32	1.32
15	1.06	1.23	1.27	1.28	1.29	1.29	1.28

Tabelle 4.3: Quantile der F-Verteilung, Konfidenz = 0.68

$n_1 =$	1	2	3	4	5	10	15
$n_2 =$							
1	39.86	49.50	53.59	55.83	57.24	60.19	61.22
2	8.53	9.00	9.16	9.24	9.29	9.39	9.42
3	5.54	5.46	5.39	5.34	5.31	5.23	5.20
4	4.55	4.32	4.19	4.11	4.05	3.92	3.87
5	4.06	3.78	3.62	3.52	3.45	3.30	3.24
6	3.78	3.46	3.29	3.18	3.11	2.94	2.87
7	3.59	3.26	3.07	2.96	2.88	2.70	2.63
8	3.46	3.11	2.92	2.81	2.73	2.54	2.46
9	3.36	3.01	2.81	2.69	2.61	2.42	2.34
10	3.29	2.92	2.73	2.61	2.52	2.32	2.24
12	3.18	2.81	2.61	2.48	2.39	2.19	2.10
15	3.07	2.70	2.49	2.36	2.27	2.06	1.97

Tabelle 4.4: Quantile der F-Verteilung, Konfidenz = 0.90

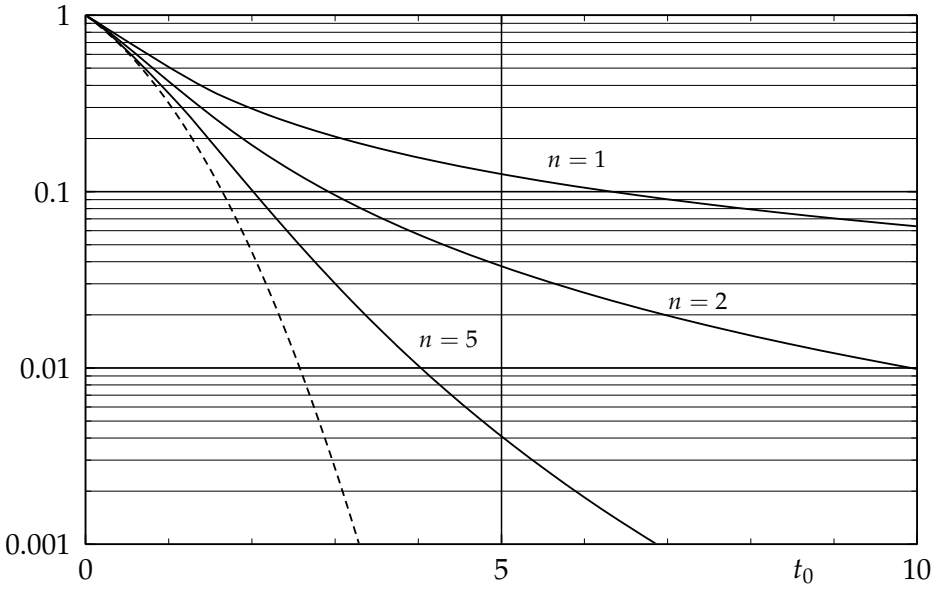


Abbildung 4.19: Wahrscheinlichkeit P, daß $|t| > t_0$ ist für die t-Verteilung mit $n = 1, 2$ und 5, aufgetragen gegen t_0. Gezeigt wird auch die Kurve für die Gauß-Verteilung (gestrichelt), der sich die t-Verteilung für großes n nähert. Die t-Verteilung mit einem Freiheitsgrad ist identisch mit der Cauchy-Verteilung mit FWHM = 2.

$n_1 =$	1	2	3	4	5	10	15
$n_2 =$							
1	161.4	199.5	215.7	224.6	230.2	241.9	245.9
2	18.51	19.00	19.16	19.25	19.30	19.40	19.43
3	10.13	9.55	9.28	9.12	9.01	8.79	8.70
4	7.71	6.94	6.59	6.39	6.26	5.96	5.86
5	6.61	5.79	5.41	5.19	5.05	4.73	4.62
6	5.99	5.14	4.76	4.53	4.39	4.06	3.94
7	5.59	4.74	4.35	4.12	3.97	3.64	3.51
8	5.32	4.46	4.07	3.84	3.69	3.35	3.22
9	5.12	4.26	3.86	3.63	3.48	3.14	3.01
10	4.97	4.10	3.71	3.48	3.33	2.98	2.85
12	4.75	3.89	3.49	3.26	3.11	2.75	2.62
15	4.55	3.68	3.29	3.06	2.90	2.54	2.40

Tabelle 4.5: Quantile der F-Verteilung, Konfidenz = 0.95

zu erhalten, ist α. Die Tabellen 4.3, 4.4, 4.5 und 4.6 zeigen die Quantile der F-Verteilung für verschiedene Werte der beiden Freiheitsgrade n_1 und n_2. Ein Beispiel für die Anwendung der F-Verteilung findet man in Kapitel 9.1.4

4.6 Theoreme

4.6.1 Die Tschebyscheff-Ungleichung

Diese Ungleichung liefert eine obere Schranke für die Wahrscheinlichkeit, daß eine Zufallsvariable mehr als eine vorgegebene Zahl von Standardabweichungen vom Mittelwert abweicht. Die Form der Wahrscheinlichkeitsdichte spielt dabei keine Rolle, die Varianz muß aber bekannt sein. Die Ungleichung lautet:

$n_1 =$	1	2	3	4	5	10	15
$n_2 =$							
1	4052.	5000.	5403.	5625.	5764.	6056.	6157.
2	98.50	99.00	99.17	99.25	99.30	99.40	99.43
3	34.12	30.82	29.46	28.71	28.24	27.23	26.87
4	21.20	18.00	16.69	15.98	15.52	14.55	14.20
5	16.26	13.27	12.06	11.39	10.97	10.05	9.72
6	13.74	10.92	9.78	9.15	8.75	7.87	7.56
7	12.25	9.55	8.45	7.85	7.46	6.62	6.31
8	11.26	8.65	7.59	7.01	6.63	5.81	5.51
9	10.56	8.02	7.00	6.42	6.06	5.26	4.96
10	10.04	7.56	6.56	5.99	5.64	4.85	4.56
12	9.33	6.93	5.96	5.41	5.06	4.30	4.01
15	8.68	6.36	5.42	4.89	4.56	3.81	3.52

Tabelle 4.6: Quantile der F-Verteilung, Konfidenz = 0.99

Die Wahrscheinlichkeit für $|x - \langle x\rangle| \geq k\sigma$ ist kleiner oder gleich $1/k^2$ für eine Zufallsvariable x mit beliebiger Wahrscheinlichkeitsdichte $f(x)$ und der Standardabweichung σ.

Beweis: Der Beweis geht von der Definition der Varianz σ^2 aus:

$$\sigma^2 = \int_{-\infty}^{+\infty} (x - \langle x\rangle)^2 \cdot f(x)\,\mathrm{d}x$$
$$= \left[\int_{-\infty}^{\langle x\rangle - k\sigma} + \int_{\langle x\rangle - k\sigma}^{\langle x\rangle + k\sigma} + \int_{\langle x\rangle + k\sigma}^{+\infty}\right] (x - \langle x\rangle)^2 \cdot f(x)\,\mathrm{d}x$$

Die Integranden in allen drei Integralen sind positiv. Wenn man das mittlere Integral wegläßt, erhält man eine Ungleichung.

$$\sigma^2 \geq \int_{-\infty}^{\langle x\rangle - k\sigma} (x - \langle x\rangle)^2 \cdot f(x)\,\mathrm{d}x + \int_{\langle x\rangle + k\sigma}^{+\infty} (x - \langle x\rangle)^2 \cdot f(x)\,\mathrm{d}x\,.$$

Im ersten bzw. letzten Integral gelten die Ungleichungen $x < \langle x\rangle - k\sigma$ bzw. $x > \langle x\rangle + k\sigma$, und so

$$\sigma^2 \geq k^2\sigma^2 \left[\int_{-\infty}^{\langle x\rangle - k\sigma} f(x)\,\mathrm{d}x + \int_{\langle x\rangle + k\sigma}^{+\infty} f(x)\,\mathrm{d}x\right]\,.$$

Der Ausdruck in Klammern ist die Wahrscheinlichkeit für $|x - \langle x\rangle| \geq k\sigma$, womit die Tschebyscheff-Ungleichung bewiesen ist.

Die Ungleichung gilt unter sehr allgemeinen Bedingungen, dafür ist sie aber auch ziemlich schwach. Sie ist nützlich für theoretische Überlegungen oder in solchen Fällen, wo man über die Verteilung sehr wenig weiß.

4.6.2 Das Gesetz der großen Zahl

Angenommen, daß in n statistisch unabhängigen Experimenten das Ereignis j insgesamt n_j mal aufgetreten ist. Die Zahlen n_j folgen einer Binomialverteilung, und das Verhältnis $h_j = n_j/n$ ist die Zufallsvariable. Der Erwartungswert von h_j, $E\left[h_j\right]$, ist die Wahrscheinlichkeit p_j für das Ereignis j

$$p_j = E\left[h_j\right] = E\left[n_j/n\right]\,.$$

Wie genau ist nun die Schätzung h_j für die unbekannte Wahrscheinlichkeit p_j? Weil die Zufallszahl $n_j = n \cdot h_j$ einer Binomialverteilung gehorcht, ist die Varianz von h_j nach Gleichung (4.11)

und Gleichung (4.15)

$$V[h_j] = \sigma^2(h_j) = \sigma^2(n_j/n) = \frac{1}{n^2} \cdot \sigma^2(n_j) = \frac{1}{n^2} \cdot np_j(1-p_j)\,.$$

Da das Produkt $p_j(1-p_j)$ immer $\leq \frac{1}{4}$ ist, gilt die Ungleichung

$$\sigma^2(h_j) < 1/n\,,$$

bekannt als *Gesetz der großen Zahl:* Der Fehler der Schätzung h_j für die Wahrscheinlichkeit p_j kann so klein wie gewünscht gemacht werden, indem man n groß genug wählt; seine obere Schranke ist $1/\sqrt{n}$.

4.6.3 Der Zentrale Grenzwertsatz

Der zentrale Grenzwertsatz (ZGS) ist der wichtigste Satz in der Statistik wegen seines allgemeinen Charakters und seiner breiten Anwendbarkeit. Unter anderem erklärt er die zentrale Bedeutung der Gauß-Verteilung. Er lautet folgendermaßen:

Die Wahrscheinlichkeitsdichte der Summe $w = \sum_{i=1}^{n} x_i$ einer Stichprobe aus n unabhängigen Zufallsvariablen x_i mit einer beliebigen Wahrscheinlichkeitsdichte mit Mittelwert $\langle x \rangle$ und Varianz σ^2 geht in der Grenze $n \to \infty$ gegen eine Gauß-Wahrscheinlichkeitsdichte mit Mittelwert $\langle w \rangle = n \cdot \langle x \rangle$ und Varianz $V[w] = n \cdot \sigma^2$.

Die Variablen x_i brauchen nicht alle aus derselben Wahrscheinlichkeitsdichte zu kommen; das Theorem gilt auch dann mit sinngemäßen Änderungen.

Zum Beweis des ZGS betrachtet man als erstes den Mittelwert

$$E\,[w] = \langle w \rangle = E\left[\sum_{i=1}^{n} x_i\right] = \sum_i E\,[x_i] = nE\,[x] = n \cdot \langle x \rangle\,,$$

was den ersten Teil des Satzes beweist. Um den Ausdruck für die Varianz herzuleiten, betrachtet man

$$V\,[w] = E\left[(w - \langle w \rangle)^2\right] = E\left[\left(\sum_i x_i - \sum_i \langle x_i \rangle\right)^2\right] = E\left[\left(\sum_i (x_i - \langle x_i \rangle)\right)^2\right]$$
$$= E\left[\sum_i (x_i - \langle x_i \rangle)^2 + \sum_{i \neq k} (x_i - \langle x_i \rangle)(x_k - \langle x_k \rangle)\right]\,.$$

Die letzte Doppelsumme gibt null, weil die Variablen nach Voraussetzung unkorreliert sind, und deshalb folgt

$$V\,[w] = E\left[\sum_{i=1}^{n} (x_i - \langle x_i \rangle)^2\right] = nV\,[x]\,.$$

Für die Tatsache, daß w gegen eine Gauß-Verteilung geht, können hier nur Plausibilitätsargumente angegeben werden: Angenommen, daß die Wahrscheinlichkeitsdichte der Zufallsvariablen x in der Summe w gegeben ist durch $f(x)$, dann ist nach Kapitel 4.10 die Wahrscheinlichkeitsdichte der Summe w

$$f_w(w) = \int f(x_1) f(x_2) \ldots f(x_n) \delta(w - \sum x_i)\, \mathrm{d}x_1\, \mathrm{d}x_2 \ldots\, \mathrm{d}x_n\,.$$

Die charakteristische Funktion von $f_w(w)$ ist

$$\phi_w(t) = \int e^{itw} f(x_1)f(x_2)\cdots f(x_n)\delta(w - \sum x_i)\,\mathrm{d}x_1\,\mathrm{d}x_2\ldots\,\mathrm{d}x_n\,\mathrm{d}w\,.$$

Nach Integration über w und mit $w = \sum x_i$ folgt

$$\phi_w(t) = \int e^{itx_1} f(x_1)\,\mathrm{d}x_1 \int e^{itx_2} f(x_2)\,\mathrm{d}x_2 \cdots \int e^{itx_n} f(x_n)\,\mathrm{d}x_n = \left[\int e^{itx} f(x)\,\mathrm{d}x\right]^n\,,$$

welches auf das folgende wichtige Theorem führt:

Falls die Wahrscheinlichkeitdichte $f(x)$ die charakteristische Funktion $\phi(t)$ hat, so hat die Zufallsvariable $w = \sum_{i=1}^n x_i$ die charakteristische Funktion $\phi_n(t) = [\phi(t)]^n$.

Entwickelt man $\phi(t)$ um null, so erhält man mit der Gleichung (4.21) und der Gleichung (4.22)

$$\phi(t) = \phi(0) + \frac{\mathrm{d}\phi(0)}{\mathrm{d}t}t + \frac{d^2\phi(0)}{\mathrm{d}t^2}\frac{t^2}{2} + \ldots = 1 + it\langle x\rangle - (\sigma^2 + \langle x\rangle^2)t^2/2 + \ldots$$

Ohne Einschränkung der Allgemeinheit kann man $\langle x\rangle = 0$ setzen und erhält $\phi(t) \approx 1 - \sigma^2 t^2/2$, und

$$\phi_w(t) \approx \phi_n(t) \approx \left[1 - \frac{\sigma^2 t^2 n}{2n}\right]^n\,,$$

was für $n \to \infty$ übergeht in $\mathrm{e}^{-n\sigma^2 t^2/2}$. Dies ist die charakteristische Funktion von $\mathrm{e}^{-w^2/2n\sigma^2}$ (siehe Gleichung (4.23)). Deshalb geht w gegen eine Gauß-Wahrscheinlichkeitsdichte mit Varianz $V[w] = n\sigma^2$. Bereits für mäßig große Werte von n kann die Gauß-Verteilung eine ganz gute Näherung sein, allerdings nur für nicht zu große Abweichungen vom Mittelwert.

Der ZGS hat auch im täglichen Leben qualitative Anwendungen. Immer wenn eine Größe auf die Summe vieler zufallsverteilter Gegebenheiten zurückgeht, dann sollte sie näherungsweise Gauß-verteilt sein. So hängen zum Beispiel die pädagogischen Fähigkeiten von Professoren von vielen Qualitäten ab, wie pädagogisches Geschick, Sorgfalt in der Vorbereitung, Beherrschung des Stoffes, Klarheit, Persönlichkeit usw. Falls eine Anzahl von Professoren nach ihren pädagogischen Fähigkeiten evaluiert wird, erwartet man bei einem objektiven Verfahren Gauß-ähnliche Kurven. Dies konnte bei einem Feldversuch an der Universität Hamburg im großen und ganzen bestätigt werden.

4.7 Stichproben

4.7.1 Auswertung von Stichproben.

Sehr häufig hat man es mit der Auswertung von Daten zu tun, welche als Zufallsvariable, entnommen einer im allgemeinen unbekannten Wahrscheinlichkeitsdichte, aufgefaßt werden können. Man nennt eine solche Gesamtheit eine *Stichprobe.* Konkret möge sie aus n Zufallsvariablen x_1, x_2, … x_{n-1}, x_n, genommen aus der Wahrscheinlichkeitsdichte $f(x)$ mit Mittelwert μ und Varianz σ^2, bestehen. Man möchte dann den im allgemeinen unbekannten Mittelwert μ und die Varianz der Wahrscheinlichkeitsdichte $f(x)$ aus der vorliegenden Stichprobe bestimmen. Als Beispiel kann man sich die zehnfache Messung der Temperatur eines Körpers vorstellen. Das Problem besteht dann darin, aus diesen zehn Messungen die beste Schätzung für den Mittelwert der Temperatur und seinen Fehler zu berechnen.

In Kapitel 6.3.1 wird gezeigt, daß die beste Schätzung $\bar{x}$ für den *Mittelwert* und die beste Schätzung s^2 für die *Varianz* gegeben sind durch

$$\bar{x} = \frac{1}{n}\sum_{i=1}^{n} x_i \qquad \text{(Stichproben-Mittelwert)} \tag{4.27}$$

$$s^2 = \frac{1}{n-1}\sum_{i=1}^{n} (x_i - \bar{x})^2 \qquad \text{(Stichproben-Varianz)} \; . \tag{4.28}$$

Dieselben Gleichungen erhält man auch, wenn man fordert, daß die Summe der Abweichungsquadrate $\sum(x_i - \bar{x})^2$ ein Minimum wird. Es ist zu beachten, daß $\bar{x}$ und s^2 Zufallsvariablen sind. In der Schätzung s^2 für die Varianz wird die Summe durch $(n-1)$ und nicht durch n dividiert, weil die beste Schätzung $\bar{x}$ für den Mittelwert und nicht der (unbekannte) wahre Wert $\langle x \rangle$ benutzt wurde. Dies wird im folgenden näher begründet.

Das Stichprobenmittel $\bar{x}$ hat als Zufallsvariable nach dem zentralen Grenzwertsatz (ZGS) Kapitel 4.6.3 eine Gauß-Verteilung mit Varianz

$$V[\bar{x}] = \frac{\sigma^2}{n} \, . \tag{4.29}$$

Deshalb ist die beste Schätzung des Mittelwertes aus den n Werten der Stichprobe gegeben durch $\bar{x} \pm s/\sqrt{n}$, wobei $s/\sqrt{n}$ die Schätzung für die Standardabweichung von $\bar{x}$ ist.

Wie gut sind diese Schätzungen? Dazu betrachtet man als erstes die Schätzung $\bar{x}$ des Mittelwertes. Ihr Erwartungswert ist definitionsgemäß

$$E[\bar{x}] = E\left[\frac{1}{n}\sum_{i=1}^{n} x_i\right] = \frac{1}{n}\sum_{i=1}^{n} E[x_i] = \frac{1}{n} \cdot n\langle x \rangle \, .$$

Dieses Ergebnis zeigt, daß der Erwartungswert des Stichprobenmittels $E[\bar{x}]$ gleich dem wahren Mittelwert $\langle x \rangle$ ist; man sagt, die Schätzung sei *erwartungstreu* oder *unverzerrt*.

Für die Schätzung s^2 der wahren Varianz σ^2 kann man eine ähnliche Überlegung durchführen. Dazu formt man Gleichung (4.28) um

$$\begin{aligned} s^2 &= \frac{1}{n-1}\sum_{i=1}^{n} [(x_i - \mu) - (\bar{x} - \mu)]^2 \\ &= \frac{1}{n-1}\sum_{i=1}^{n} \left[(x_i - \mu)^2 + (\bar{x} - \mu)^2 - 2(x_i - \mu)(\bar{x} - \mu)\right] \\ &= \frac{1}{n-1}\left[\sum_{i=1}^{n} (x_i - \mu)^2 - \sum_{i=1}^{n} (\bar{x} - \mu)^2\right] \end{aligned}$$

mit der Zwischenrechnung

$$\begin{aligned} -2\sum_{i=1}^{n} (x_i - \mu)(\bar{x} - \mu) &= -2 \cdot (\bar{x} - \mu) \cdot \sum_{i=1}^{n} (x_i - \mu) \\ &= -2n \cdot (\bar{x} - \mu) \cdot (\bar{x} - \mu) = -2 \cdot \sum_{i=1}^{n} (\bar{x} - \mu)^2 \, . \end{aligned}$$

Der Erwartungswert von s^2 ist dann

$$\begin{aligned} E[s^2] &= \frac{1}{n-1}\left\{E\left[\sum_{i=1}^{n} (x_i - \mu)^2\right] - E\left[\sum_{i=1}^{n} (\bar{x} - \mu)^2\right]\right\} \\ &= \frac{1}{n-1}\left\{n \cdot \sigma^2 - n \cdot \frac{\sigma^2}{n}\right\} = \frac{n-1}{n-1}\sigma^2 = \sigma^2 \, , \end{aligned} \tag{4.30}$$

weil $E\left[(\bar{x}-\mu)^2\right] = \sigma^2/n$ nach dem zentralen Grenzwertsatz. Deshalb ist die Stichprobenvarianz s^2 ein *unverzerrter Schätzwert* für die wahre Varianz $\langle s^2 \rangle = \sigma^2$.
Wie genau ist diese Schätzung s^2 der Varianz? Die Frage führt auf die Varianz der Wahrscheinlichkeitsdichte der Zufallsvariablen s^2. Aus Gleichung (4.28) und Gleichung (4.11) folgt die Varianz von s^2:

$$V\left[s^2\right] = V\left[\frac{1}{n-1}\sum_{i=1}^{n}(x_i-\bar{x})^2\right] = \frac{\sigma^4}{(n-1)^2} V\left[\sum_{i=1}^{n}\frac{(x_i-\bar{x})^2}{\sigma^2}\right] .$$

Aus der Herleitung von Gleichung (4.30) kann man entnehmen, daß die Summe $\sum_{i=1}^{n}(x_i - \bar{x})^2/\sigma^2$ ersetzt werden kann durch die Summe $\sum_{i=1}^{n-1}(x_i - \langle x \rangle)^2/\sigma^2$. Diese Summe folgt einer χ^2-Verteilung mit $(n-1)$ Freiheitsgraden, und ihre Varianz ist nach Gleichung (4.25) gegeben durch $2(n-1)$, und deshalb ist die Varianz von s^2

$$V\left[s^2\right] = \frac{2\sigma^4}{n-1} .$$

Die Standardabweichung $\sigma(s^2)$ der Wahrscheinlichkeitsdichte von s^2 ist dann

$$\sigma(s^2) = \sqrt{V\left[s^2\right]} = \sigma^2 \cdot \sqrt{\frac{2}{n-1}} .$$

Das Endergebnis für die Genauigkeit der Schätzung s^2 kann dann folgendermaßen durch die Standardabweichung von s^2 beschrieben werden:

$$s^2 \pm \sigma(s^2) = s^2 \pm \sigma^2 \cdot \sqrt{2/(n-1)} .$$

Die beste Schätzung s für die wahre Standardabweichung σ erhält man als die Wurzel aus der Gleichung für s^2, also mit $s \approx \sigma$

$$\left(s^2 \pm \sigma^2 \cdot \sqrt{\frac{2}{n-1}}\right)^{1/2} \approx s\left(1 \pm \sqrt{\frac{2}{n-1}}\right)^{1/2} \approx s \cdot \left(1 \pm \sqrt{\frac{1}{2(n-1)}}\right) ,$$

d.h. die Standardabweichung von s hat den Schätzwert

$$\sigma(s) = \frac{s}{\sqrt{2(n-1)}} .$$

4.7.2 Gewichte

Manchmal haben die Daten x_i der Stichprobe bekannte Standardabweichungen σ_i, die im Prinzip alle verschieden sein können. Als Beispiel kann die Bildung des Mittelwertes verschiedener unabhängiger Bestimmungen einer physikalischen Größe (etwa der Elektronenmasse) dienen, wobei jede Messung einen anderen Fehler hat. Das Stichprobenmittel ist in diesem Fall gegeben durch Gleichung (6.5)

$$\bar{x} = \frac{\sum_{i=1}^{n}(x_i/\sigma_i^2)}{\sum_{i=1}^{n}(1/\sigma_i^2)} ,$$

wobei es üblich ist, die Gewichte $w_i = 1/\sigma_i^2$ einzuführen. Für den Fall, daß die Standardabweichungen σ_i alle gleich und gleich σ sind, erhält man Gleichung (4.27) zurück. Die Varianz des Stichprobenmittels $\bar{x}$ ist nach Gleichung (6.6)

$$V\left[\bar{x}\right] = \frac{1}{\sum_{i=1}^{n}(1/\sigma_i^2)} = \frac{1}{\sum_{i=1}^{n} w_i} .$$

Dies führt, falls alle σ_i gleich sind ($\sigma_i = \sigma$), zurück zu $V[\overline{x}] = \sigma^2/n$.

Einen anderen Fall, nicht zu verwechseln mit dem vorhergehenden, hat man, wenn man die einzelnen Daten x_i selbst auf eine solche Art mit Gewichten w_i versieht, indem man jeden einzelnen Wert w_i-fach zählt. Ist zum Beispiel x_i die Zahl der Paare in einem Tanzkurs, und will man zu Einzelpersonen übergehen, so erhält jedes Paar das Gewicht $w_i = 2$, und die Zahl der Personen im Tanzkurs ist $w_i \cdot x_i$.
Allgemein ist die Summe der gewichteten Ereignisse

$$w = \sum_i x_i w_i \,.$$

Die Varianz von w ist nach Gleichung (4.51)

$$V[w] = \sum_i w_i^2 V[x_i] \quad .$$

Beispiel 4.20 Einnahmen eines Viehhändlers.

Ein Händler verkauft Schafe, Pferde und Kühe. Ein Schaf kostet $w_1 = 100$ Euro, ein Pferd $w_2 = 500$ Euro und eine Kuh $w_3 = 200$ Euro. Die Erfahrung sagt ihm, daß er im Mittel über viele Markttage an einem Tag x_1 Schafe, x_2 Pferde und x_3 Kühe verkauft. Die mittlere Zahl der an einem Tag verkauften Tiere ist dann $\overline{x} = \sum_{i=1}^{3} x_i$ und die mittleren Einnahmen sind $\overline{w} = \sum_{i=1}^{3} w_i x_i$. Die Standardabweichung der Zahl verkaufter Tiere, falls die einzelnen Verkaufsvorgänge unabhängig sind, ist dann $\sigma_x = \sqrt{\overline{x}}$, und die Standardabweichung der Einnahmen ist, mit $V[x_i] = x_i$ (Poisson-Statistik),

$$\sigma_w = \sqrt{V[w]} = \sqrt{\sum_{i=1}^{3} w_i^2 x_i} \,.$$

4.7.3 Numerische Berechnung von Stichprobenmittel und -Varianz

Solche Berechnungen erfordern oft einige Sorgfalt. Die Stichprobe enthalte n Werte x_i, $i = 1, 2, \ldots n$. Die Formeln sind:

$$\overline{x} = \frac{1}{n} \sum_{i=1}^{n} x_i \qquad\qquad s^2 = \frac{1}{n-1} \sum_{i=1}^{n} (x_i - \overline{x})^2 \,.$$

Die Berechnung kann in zwei Schleifen über die Daten erfolgen. In der ersten Schleife wird die Summe $R_x = \sum_i x_i$ berechnet, woraus das Stichprobenmittel $\overline{x} = R_x/n$ folgt. Dieses Stichprobenmittel wird dann in der nächsten Schleife über alle Daten benutzt, um die Summe $R_{xx} = \sum_i (x_i - \overline{x})^2$ zu berechnen, woraus die Stichprobenvarianz $s^2 = R_{xx}/(n-1)$ folgt. Dagegen ist nichts einzuwenden, wenn alle Daten im Speicher verfügbar sind. Oft werden die Daten aber sequentiell eingelesen, oder es mag andere Gründe geben, warum man beide Werte in einer einzigen Schleife berechnen möchte. In diesem Fall benutzt man zur Berechnung der Varianz die Gleichung (4.11)

$$s^2 = \frac{1}{n-1} \sum_{i=1}^{n} (x_i - \overline{x})^2 = \frac{1}{n-1} \left(\sum_{i=1}^{n} x_i^2 - \frac{1}{n} \left(\sum_{i=1}^{n} x_i \right)^2 \right) \,.$$

Es ist also möglich, die beiden Resultate in einer einzigen Schleife zu erhalten. Man bildet die beiden Summen

$$S_x = \sum_{i=1}^{n} x_i \qquad S_{xx} = \sum_{i=1}^{n} x_i^2$$

in einer Schleife über alle Daten und berechnet dann Stichprobenmittel und -Varianz gemäß

$$\overline{x} = \frac{1}{n} S_x \qquad s^2 = \frac{1}{n-1}\left(S_{xx} - \frac{1}{n} S_x^2\right) .$$

Dies sieht im Prinzip gut aus, jedoch gibt es unter Umständen Schwierigkeiten bei der Berechnung von s^2, wo die Differenz zweier großer positiver Zahlen gebildet wird. In diesem Fall ist es besser, für den Stichprobenmittelwert eine erste grobe Näherung x_e (zum Beispiel den ersten Wert x_1) zu benutzen. Man berechnet dann

$$T_x = \sum_{i=1}^{n} (x_i - x_e) \qquad T_{xx} = \sum_{i=1}^{n} (x_i - x_e)^2$$

wieder in einer Schleife, und die Resultate

$$\overline{x} = x_e + \frac{1}{n} T_x \qquad s^2 = \frac{1}{n-1}\left(T_{xx} - \frac{1}{n} T_x^2\right)$$

sind meist numerisch erheblich genauer als die nach dem erstgenannten Verfahren.

4.8 Mehrdimensionale Verteilungen

4.8.1 Zufallsvariable in zwei Dimensionen

Die zweidimensionale Wahrscheinlichkeitsdichte $f(x, y)$ der zwei Zufallszahlen x und y ist definiert durch die Wahrscheinlichkeit, das Variablenpaar (x, y) in den Intervallen $a \leq x < b$ und $c \leq y < d$ zu finden

$$P(a \leq x < b\,, c \leq y < d) = \int_c^d \int_a^b f(x, y)\, \mathrm{d}x\, \mathrm{d}y\,.$$

Die zweidimensionale Wahrscheinlichkeitsdichte $f(x, y)$ gehorcht der Normierung

$$\int_{-\infty}^{+\infty} \int_{-\infty}^{+\infty} f(x, y)\, \mathrm{d}x\, \mathrm{d}y = 1\,.$$

Abbildung 4.20 zeigt als Beispiel eine zweidimensionale Wahrscheinlichkeitsdichte.

Wenn die Wahrscheinlichkeitsdichte $f(x, y)$ als Produkt $f(x, y) = g(x) \cdot h(y)$ der zwei Funktionen $g(x)$ und $h(y)$ geschrieben werden kann, dann sind die zwei Zufallsvariablen *unabhängig*. Projektionen der zweidimensionalen Verteilung $f(x, y)$ auf die Achsen heißen *Randverteilungen*.

$$f_x(y) = \int_{-\infty}^{\infty} f(x, y)\, \mathrm{d}x \qquad f_y(x) = \int_{-\infty}^{\infty} f(x, y)\, \mathrm{d}y\,.$$

Sie repräsentieren die Wahrscheinlichkeitsdichte der Zufallsvariablen x bzw. y, wenn jeweils die andere Variable ignoriert wird.

Schnitte durch $f(x,y)$ heißen *bedingte Wahrscheinlichkeitsdichten*. Für einen gegebenen festen Wert $x = x_0$ ist die bedingte Wahrscheinlichkeitsdichte gegeben durch

$$f(y|x_0) = \frac{f(x_0,y)}{\int f(x_0,y)\,\mathrm{d}y} = \frac{f(x_0,y)}{f_y(x_0)}.$$

Aus der Randverteilung $f_y(x)$ und der bedingten Wahrscheinlichkeitsdichte $f(y|x)$ erhält man die gemeinsame Wahrscheinlichkeitsdichte zurück:

$$f(x,y) = f(y|x)\ f_y(x)\,.$$

Die Randverteilung $f_x(y)$ ist dann gegeben durch

$$f_x(y) = \int_{-\infty}^{\infty} f(y|x) f_y(x)\,\mathrm{d}x.$$

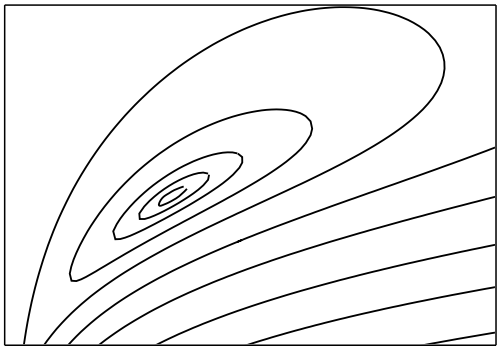


Abbildung 4.20: Beispiel einer zweidimensionalen Wahrscheinlichkeitsdichte $f(x,y)$. Gezeigt sind Linien $f(x,y) =$ konst.

Die Definitionen der Mittelwerte und Varianzen sind naheliegende Verallgemeinerungen des eindimensionalen Falls:

$$\langle x\rangle = E[x] = \iint x f(x,y)\,\mathrm{d}x\,\mathrm{d}y = \int x f_y(x)\,\mathrm{d}x \tag{4.31}$$
$$\langle y\rangle = E[y] = \iint y f(x,y)\,\mathrm{d}x\,\mathrm{d}y = \int y f_x(y)\,\mathrm{d}y \tag{4.32}$$
$$V[x] = \iint (x-\langle x\rangle)^2 f(x,y)\,\mathrm{d}x\,\mathrm{d}y = \sigma_x^2 \tag{4.33}$$
$$V[y] = \iint (y-\langle y\rangle)^2 f(x,y)\,\mathrm{d}x\,\mathrm{d}y = \sigma_y^2\,. \tag{4.34}$$

Zusätzlich führt man den Begriff der **Kovarianz** zwischen x und y ein

$$\sigma_{xy} = \mathrm{cov}(x,y) = \iint (x-\langle x\rangle)\,(y-\langle y\rangle)\,f(x,y)\,\mathrm{d}x\,\mathrm{d}y\,.$$

Die Kovarianz kann durch den **Korrelationskoeffizienten** ρ_{xy} ausgedrückt werden

$$\sigma_{xy} = \rho_{xy}\,\sqrt{\sigma_x^2\sigma_y^2}\,.$$

Er liegt zwischen $+1$ und -1. Wenn die zwei Variablen x und y voneinander unabhängig sind, folgt $\rho_{xy} = 0$. Umgekehrt folgt aber nicht notwendigerweise, daß die beiden Variablen unkorreliert sind, wenn $\rho_{xy} = 0$ ist. Falls $\rho_{xy} = +1$ oder $= -1$ ist, besteht eine vollständige Korrelation zwischen den zwei Variablen, und die eine kann durch $\pm$ die andere ersetzt werden. Um zu beweisen, daß die Ungleichung

$$|\rho_{xy}| = \frac{|\sigma_{xy}|}{\sqrt{\sigma_x^2\sigma_y^2}} \le 1 \tag{4.35}$$

gilt, betrachtet man den Erwartungswert

$$E\left[(a(x-\langle x\rangle)+b(y-\langle y\rangle))^2\right] = a^2\sigma_x^2 + b^2\sigma_y^2 + 2ab\sigma_{xy}\,.$$

Dieser Ausdruck ist immer ≥ 0 für beliebige Werte von a und b. Deshalb ist (siehe Kapitel 3.2, 3.6 und 3.8)

$$\begin{vmatrix} \sigma_x^2 & \sigma_{xy} \\ \sigma_{xy} & \sigma_y^2 \end{vmatrix} = \sigma_x^2\sigma_y^2 - \sigma_{xy}^2 = \sigma_x^2\sigma_y^2(1-\rho_{xy}^2) \geq 0\,,$$

woraus Ungleichung (4.35) folgt.

4.8.2 Zweidimensionale Gauß-Verteilung

Dies ist ein wichtiger Spezialfall. Die Wahrscheinlichkeitsdichte lautet

$$f(x,y)\,\mathrm{d}x\,\mathrm{d}y = \frac{1}{2\pi\sigma_x\sigma_y\sqrt{1-\rho^2}} \cdot \exp\left\{-\frac{1}{2(1-\rho^2)} \left[\left(\frac{x-\xi}{\sigma_x}\right)^2 - 2\rho\left(\frac{x-\xi}{\sigma_x}\right)\left(\frac{y-\eta}{\sigma_y}\right) + \left(\frac{y-\eta}{\sigma_y}\right)^2\right]\right\} \mathrm{d}x\,\mathrm{d}y\,. \quad (4.36)$$

Aus diesen Ausdrücken erhält man die Randverteilungen für x und y

$$f_y(x) = \int_{-\infty}^{+\infty} f(x,y)\,\mathrm{d}y = \frac{1}{\sqrt{2\pi}\sigma_x}\exp\left\{-\frac{1}{2}\left(\frac{x-\xi}{\sigma_x}\right)^2\right\} \quad (4.37)$$

$$f_x(y) = \int_{-\infty}^{+\infty} f(x,y)\,\mathrm{d}x = \frac{1}{\sqrt{2\pi}\sigma_y}\exp\left\{-\frac{1}{2}\left(\frac{y-\eta}{\sigma_y}\right)^2\right\}\,. \quad (4.38)$$

Mittelwerte und Varianzen der Randverteilungen sind ξ, η und ${\sigma_x}^2$, ${\sigma_y}^2$. Zusammen mit Gleichungen (4.31) bis (4.34) folgt auch, daß

$$\langle x\rangle = \xi \qquad \langle y\rangle = \eta$$
$$V[x] = \sigma_x^2 \qquad V[y] = \sigma_y^2\,.$$

Der Korrelationskoeffizient ist $\rho_{xy} = \rho$.
Beweis: Man transformiert den Exponenten zu

$$-\frac{1}{2(1-\rho^2)}\cdot\left(\frac{x-\xi}{\sigma_x} - \frac{\rho(y-\eta)}{\sigma_y}\right)^2 - \frac{(y-\eta)^2}{2\sigma_y^2}\,.$$

Durch Integration über x erhält man Gleichung (4.38). Dann folgt

$$\langle y\rangle = \int y f(x,y)\,\mathrm{d}x\,\mathrm{d}y = \int y\left(\int f(x,y)\,\mathrm{d}x\right)\mathrm{d}y = \int y f_x(y)\,\mathrm{d}y = \eta\,.$$

Ohne Verlust der Allgemeinheit kann man $\xi = \eta = 0$ setzen, und dann folgt

$$\sigma_{xy} = \int\int xy f(x,y)\,\mathrm{d}x\,\mathrm{d}y$$
$$= N\int\int x\cdot\exp\left(-\frac{(x-\rho\sigma_x y/\sigma_y)^2}{2(1-\rho^2)\sigma_x^2}\right)\mathrm{d}x\cdot y\cdot\exp\left(-\frac{y^2}{2\sigma_y^2}\right)\mathrm{d}y.$$

Das erste Integral über x gibt $N'\rho\sigma_x y/\sigma_y$, und das Integral über y wird

$$\frac{N'\rho\sigma_x}{\sigma_y}\int y^2\cdot\exp(-\frac{y^2}{2\sigma_y^2})\,\mathrm{d}y = \rho\sigma_x\sigma_y\,.$$

Die bedingten Wahrscheinlichkeitsdichten sind

$$\begin{aligned} f(x|y_0) &= \frac{f(x,y_0)}{f_x(y_0)} && (4.39)\\ &= \frac{1}{\sqrt{2\pi}\sigma_x\sqrt{1-\rho^2}}\exp\left\{-\frac{1}{2(1-\rho^2)}\left[\frac{x-\xi}{\sigma_x}-\rho\frac{y_0-\eta}{\sigma_y}\right]^2\right\} \\ f(y|x_0) &= \frac{f(x_0,y)}{f_y(x_0)} && (4.40)\\ &= \frac{1}{\sqrt{2\pi}\sigma_y\sqrt{1-\rho^2}}\exp\left\{-\frac{1}{2(1-\rho^2)}\left[\frac{y-\eta}{\sigma_y}-\rho\frac{x_0-\xi}{\sigma_x}\right]^2\right\}\,. \end{aligned}$$

Die **bedingten Mittelwerte** sind

$$E\left[x|y_0\right] = \int_{-\infty}^{+\infty} x f(x|y_0)\,\mathrm{d}x = \xi+\rho\sigma_x\left(\frac{y_0-\eta}{\sigma_y}\right) \quad (4.41)$$

$$E\left[y|x_0\right] = \int_{-\infty}^{+\infty} y f(y|x_0)\,\mathrm{d}y = \eta+\rho\sigma_y\left(\frac{x_0-\xi}{\sigma_x}\right)\,. \quad (4.42)$$

Die Varianzen der bedingten Wahrscheinlichkeitsdichten sind $\sigma_x^2(1-\rho^2)$ bzw. $\sigma_y^2(1-\rho^2)$. Beide werden null für vollständig korrelierte Variable.

Kurven konstanter Wahrscheinlichkeitsdichte haben die Form von Ellipsen in der x-y-Ebene, da die Funktion im Exponenten eine beschränkte quadratische Form ist (siehe Abbildung 4.21).

Für die (1σ)-Kontur, wo die Wahrscheinlichkeitsdichte $f(x,y)$ auf $1/\sqrt{\mathrm{e}}$ des Maximalwertes gefallen ist, ist die Ellipse im Rechteck $|x-\xi|\le\sigma_x$, $|y-\eta|\le\sigma_y$ eingeschrieben. Zum Beweis benutzt man Gleichung (4.36) mit $\xi=\eta=0$ ohne Verlust der Allgemeinheit; die (1σ)-Kontur ist gegeben durch

$$x^2/\sigma_x^2-2\rho xy/\sigma_x\sigma_y+y^2/\sigma_y^2 = 1-\rho^2\,.$$

Differenzieren nach x führt mit $y'=0$ auf $x_m=\rho\sigma_x y/\sigma_y$. Einsetzen in die Ellipsengleichung gibt $y_m=\pm\sigma_y$. Die Wahrscheinlichkeit, daß das Variablenpaar x,y innerhalb dieser Ellipse liegt, ist $1-1/\sqrt{e}=39\%$. Nach Integration über y gehorcht die Variable x einer Gauß-Verteilung mit Mittelwert ξ und Varianz σ^2 (siehe Gleichung (4.37)).

Beispiel 4.21 Coulombsche Vielfachstreuung.

Die Bahn eines geladenen Teilchens, welches bei $y=0$, $x=0$ entlang der x-Achse startet, weicht bei Durchgang durch Materie von einer Geraden infolge der Coulombschen Vielfachstreuung ab (siehe Abbildung 4.22).

Man betrachtet nun die Projektion der Bahn in die xy-Ebene. Nach Durchlaufen der Schichtdicke x ist der projizierte Abstand y von der x-Achse und der projizierte Winkel θ_y mit der x-Achse gegeben durch die zweidimensionale Wahrscheinlichkeitsdichte in y und θ_y

$$f(y,\theta_y)\mathrm{d}y\mathrm{d}\theta_y = \frac{2\sqrt{3}}{\pi}\frac{1}{{\theta_s}^2x^2}\,exp\left[-\frac{4}{{\theta_s}^2}\left(\frac{{\theta_y}^2}{x}-\frac{3y\theta_y}{x^2}+\frac{3y^2}{x^3}\right)\right]\,\mathrm{d}y\,\mathrm{d}\theta_y$$

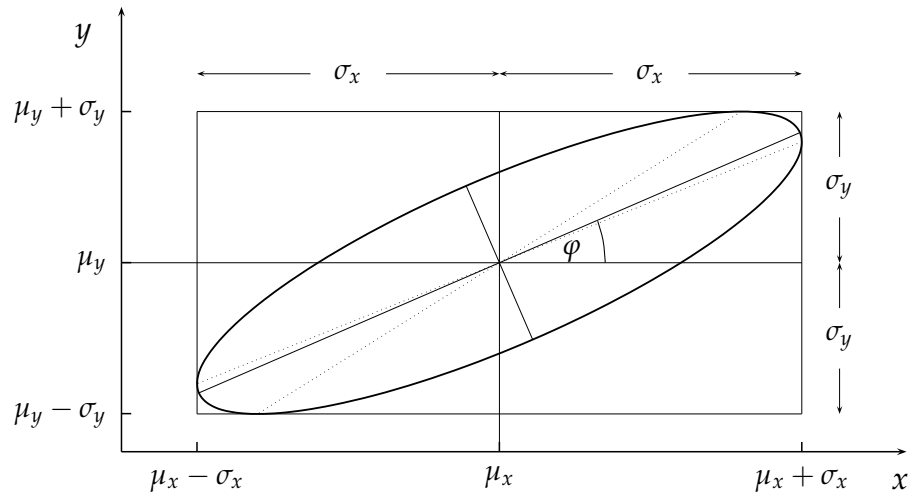


Abbildung 4.21: Die elliptische 1σ-Kontur der zweidimensionalen Normalverteilung mit Mittelwerten μ_x, μ_y und Standardabweichungen σ_x, σ_y. Der Korrelationskoeffizient in dieser Abbildung ist $\rho = +0.8$. Eingezeichnet sind die Hauptachsen der Ellipse unter den durch $\tan 2\varphi = \pm(2\rho\sigma_x\sigma_y)/(\sigma_x^2 - \sigma_y^2)$ gegebenen Winkeln φ und $\varphi + \pi/2$. Die punktierten Linien geben die bedingten Mittelwerte $x - \mu_x = (y - \mu_y)\rho\sigma_x/\sigma_y$ und $y - \mu_y = (x - \mu_x)\rho\sigma_y/\sigma_x$ an.

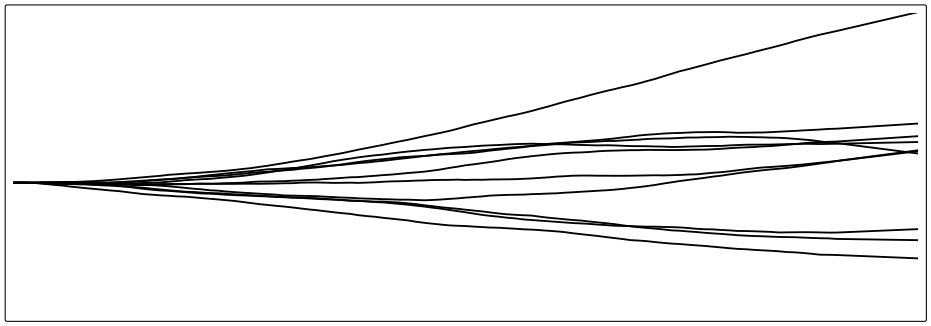


Abbildung 4.22: Beispiel für Vielfachstreuung. Gezeigt sind die Bahnen von zehn geladenen Teilchen, die vom gleichen Punkt horizontal starten und durch Vielfachstreuung ihre Richtung ändern.

mit der charakteristischen Größe

$$\theta_s{}^2 = \left(\frac{E_s}{p\beta}\right)^2 / X_0 \; ;$$

dabei ist $E_s = 21.2$ MeV, $X_0 =$ Strahlungslänge, $p =$ Impuls, und $\beta = p/E$ ist Geschwindigkeit$/c$. Ein Koeffizientenvergleich von θ_y^2, $y\,\theta_y$ und y^2 im Exponenten mit dem entsprechenden Ausdruck in Gleichung (4.36) liefert $\rho^2 = 3/4$, $\sigma_\theta^2 = x\,\theta_s^2/2$ und $\sigma_y^2 = x^3\,\theta_s^2/6$. Gemittelt über alle Streuwinkel θ_y ist die Varianz des projizierten y-Abstandes von der x-Achse folglich

$$V\,[y] = \langle y^2 \rangle = \sigma_y{}^2 = x^3 \cdot \theta_s{}^2/6 \; .$$

Ebenso ist die Varianz des projizierten Streuwinkels θ_y, gemittelt über alle Trajektorien

$$V\left[\theta_y\right] = \langle {\theta_y}^2 \rangle = x\,{\theta_s}^2/2\,.$$

Streuwinkel θ_y und Ablage y sind korreliert. Wenn man zum Beispiel Teilchen mit einer ganz bestimmten Auslenkung $y = y_0$, etwa durch eine Zählerkoinzidenz auswählt, so ist der mittlere projizierte Streuwinkel dieser Teilchen nicht mehr null, sondern $\langle \theta_y \rangle = \rho y \sigma_\theta / \sigma_y = 3y_0/2x$ (siehe Gleichung (4.41)). Die Varianz des projizierten Streuwinkels um diesen neuen Mittelwert folgt aus Gleichung (4.41)

$$V\left[\theta_y(y_0)\right] = (1-\rho^2)\langle \theta_y^2 \rangle = \frac{1}{4}\frac{x{\theta_s}^2}{2}\,,$$

weil $1 - \rho^2 = 1/4$.

4.8.3 Mehrdimensionale Wahrscheinlichkeitsdichten

Die Wahrscheinlichkeitsdichte eines n-Vektors $\boldsymbol{x}$ von Zufallsvariablen mit Komponenten x_i, $(i = 1, 2, \ldots n)$ kann geschrieben werden als $f(x_1,\, x_2, \ldots x_n) = f(\boldsymbol{x})$. Hieraus lassen sich Wahrscheinlichkeiten definieren durch Integration über eine oder mehrere der Variablen x_i analog wie in Kapitel 4.8.1. Der Mittelwert der Variablen x_i ist

$$E\left[x_i\right] = \langle x_i \rangle = \int x_i f(x_1, x_2, \ldots, x_n)\,\mathrm{d}x_1\,\mathrm{d}x_2 \ldots\,\mathrm{d}x_n$$
$$= \int x_i \cdot f(\boldsymbol{x})\,\mathrm{d}x_1\,\mathrm{d}x_2 \ldots\,\mathrm{d}x_n\,.$$

In n Dimensionen können alle n Mittelwerte der n Variablen x_i durch den folgenden n-Vektor zusammengefaßt werden: $\langle \boldsymbol{x} \rangle = E\left[\boldsymbol{x}\right]$.
Als Verallgemeinerung der Varianz definiert man die **Kovarianzmatrix** durch

$$\boldsymbol{V} = \boldsymbol{V}\left[\boldsymbol{x}\right] = E\left[\left(\boldsymbol{x} - E\left[\boldsymbol{x}\right]\right)\left(\boldsymbol{x} - E\left[\boldsymbol{x}\right]\right)^T\right]\,. \tag{4.43}$$

Damit ergeben sich als Diagonalelemente der Matrix $\boldsymbol{V}$ die Varianzen und als Nicht-Diagonalelemente die Kovarianzen:

$$V_{ii} = \mathrm{var}\left(x_i\right) = \int \left(x_i - \langle x_i \rangle\right)^2 \cdot f\left(\boldsymbol{x}\right)\,\mathrm{d}x_1\,\mathrm{d}x_2 \ldots\,\mathrm{d}x_n$$
$$V_{ij} = \mathrm{cov}\left(x_i, x_j\right) = \int \left(x_i - \langle x_i \rangle\right)\left(x_j - \langle x_j \rangle\right) \cdot f(\boldsymbol{x})\,\mathrm{d}x_1\,\mathrm{d}x_2 \ldots\,\mathrm{d}x_n\,.$$

Die Kovarianzmatrix

$$\boldsymbol{V} = \begin{pmatrix} \mathrm{var}(x_1) & \mathrm{cov}(x_1, x_2) & \ldots & \mathrm{cov}(x_1, x_n) \\ \mathrm{cov}(x_2, x_1) & \mathrm{var}(x_2) & \ldots & \mathrm{cov}(x_2, x_n) \\ \ldots & & & \\ \ldots & & & \\ \mathrm{cov}(x_n, x_1) & \mathrm{cov}(x_n, x_2) & \ldots & \mathrm{var}(x_n) \end{pmatrix}$$

ist eine symmetrische $n \times n$-Matrix. Für die Diagonalelemente $V_{ii} = \mathrm{var}\left(x_i\right)$ (Varianzen), die stets positiv sind, schreibt man auch σ_i^2, und für die Nicht-Diagonalelemente $V_{ij} = V_{ji} = \mathrm{cov}\left(x_i, x_j\right)$ (Kovarianzen), die positiv und negativ sein können, schreibt man auch σ_{ij},

$$\boldsymbol{V} = \begin{pmatrix} \sigma_1^2 & \sigma_{12} & \sigma_{13} & \ldots & \sigma_{1n} \\ \sigma_{21} & \sigma_2^2 & \sigma_{23} & \ldots & \sigma_{2n} \\ & & & & \\ .. & & & & \\ \sigma_{n1} & \sigma_{n2} & \sigma_{n3} & \ldots & \sigma_n^2 \end{pmatrix}\,.$$

Wenn als Resultat einer Messung Parameter angegeben werden, so werden gewöhnlich als Parameterfehler die Wurzeln der Diagonalelemente angegeben und als Fehlerbalken eingetragen. Jedoch hängen die Konfidenzintervalle im mehrdimensionalen Raum auch von den Kovarianzen ab; falls Korrelationen zwischen verschiedenen Datenpunkten existieren, geben die Fehlergrenzen allein keine zureichende Auskunft über die Übereinstimmung zwischen der Messung und der Erwartung.

4.8.4 Mehrdimensionale Gauß-Verteilung

Die Wahrscheinlichkeitsdichte der Gaußfunktion in n Dimensionen enthält als Parameter den n-Vektor der Mittelwerte $\boldsymbol{\mu_x} = \langle \boldsymbol{x} \rangle$ der n Variablen und die n^2 Elemente der Kovarianzmatrix V_{ij}:

$$\boxed{f(\boldsymbol{x}) = \frac{|\boldsymbol{G}|^{1/2}}{(2\pi)^{n/2}} \exp\left(-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu_x})^T\, \boldsymbol{G}\, (\boldsymbol{x}-\boldsymbol{\mu_x})\right)}, \tag{4.44}$$

wobei die Matrix $\boldsymbol{G} = \boldsymbol{V}^{-1}$ und $|\boldsymbol{G}| = \det\, \boldsymbol{G} = |\boldsymbol{V}^{-1}|$ ist. Die n-dimensionale Gauß-Verteilung ist das einfachste Modell einer Wahrscheinlichkeitsdichte von n korrelierten Zufallsvariablen, da die einzigen Parameter, welche die Wahrscheinlichkeitsdichte charakterisieren, die n Mittelwerte und die n^2 Elemente der Kovarianzmatrix $\boldsymbol{V}$ sind, von denen nur $(n^2+n)/2$ unabhängig sind.

Daß die Matrix $\boldsymbol{G}$ in Gleichung (4.44) gleich dem Inversen der Kovarianzmatrix ist, also $\boldsymbol{G} = \boldsymbol{V}^{-1}$, sieht man folgendermaßen: Um zu beweisen, daß

$$\langle x_i \rangle = \mu_i = \int x_i f(\boldsymbol{x})\, \mathrm{d}x_1\, \mathrm{d}x_2 \ldots\, \mathrm{d}x_n$$

ist, substituiert man $u_i = x_i - \mu_i$, $\mathrm{d}u_i = \mathrm{d}x_i$; dies führt zu

$$\langle x_i \rangle = \int (u_i + \mu_i) f(\boldsymbol{u})\, \mathrm{d}u_1\, \mathrm{d}u_2 \ldots = \int u_i f(\mathbf{u})\, \mathrm{d}u_1 \cdots + \mu_i \int f(\boldsymbol{u})\, \mathrm{d}u_i \ldots .$$

Das erste Integral ist gleich null, wie man sieht, falls man $\boldsymbol{u}^T \boldsymbol{G} \boldsymbol{u}$ in der Funktion $f(\boldsymbol{u})$ auf Diagonalform bringt durch eine lineare Transformation, und das zweite Integral ist gleich μ_i (Normierung). Nun folgt mit den Abkürzungen

$$g(\boldsymbol{x}) = -\frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu_x})^T G (\boldsymbol{x}-\boldsymbol{\mu_x}) \qquad , \qquad f(\boldsymbol{x}) = N e^{g(\boldsymbol{x})}$$

die Gleichung

$$E\left[(x_i - \mu_i)\right] = E\left[x_i\right] - \mu_i = 0 = \int (x_i - \mu_i) N e^{g(\boldsymbol{x})}\, \mathrm{d}x_1\, \mathrm{d}x_2 \ldots \mathrm{d}x_n\,,$$

und deshalb

$$\frac{\partial}{\partial \mu_l} E\left[x_i - \mu_i\right] = \frac{\partial}{\partial \mu_l} \int (x_i - \mu_i) N e^{g(\boldsymbol{x})}\, \mathrm{d}x_1\, \mathrm{d}x_2 \ldots\, \mathrm{d}x_n = 0\,,$$

und deshalb

$$0 = -\delta_{il} \cdot \int N e^{g(\boldsymbol{x})}\, \mathrm{d}x_1\, \mathrm{d}x_2 \ldots\, \mathrm{d}x_n + \int N (x_i - \mu_i) e^{g(\boldsymbol{x})} \frac{\partial g(\boldsymbol{x})}{\partial \mu_l}\, \mathrm{d}x_1\, \mathrm{d}x_2 \ldots\, \mathrm{d}x_n \tag{4.45}$$

wobei

$$\frac{\partial g(\boldsymbol{x})}{\partial \mu_l} = \sum_k G_{kl}(x_k - \mu_k)$$

wegen $G_{kl} = G_{lk}$. Das erste Integral in der Gleichung (4.45) ist $-\delta_{il}$, und deshalb kann die Gleichung geschrieben werden

$$\begin{aligned}
\sum_k \int N G_{kl}(x_k - \mu_k)(x_i - \mu_i) e^{\mathcal{G}(\boldsymbol{x})}\, \mathrm{d}x_1\, \mathrm{d}x_2 \ldots \mathrm{d}x_n &= \delta_{il} \quad \text{oder} \\
\sum_k G_{kl} \int N(x_k - \mu_k)(x_i - \mu_i) e^{\mathcal{G}(\boldsymbol{x})}\, \mathrm{d}x_1\, \mathrm{d}x_2 \ldots \mathrm{d}x_n &= \delta_{il} \quad \text{oder} \\
\sum_k G_{kl} \int (x_k - \mu_k)(x_i - \mu_i) f(\boldsymbol{x})\, \mathrm{d}x_1\, \mathrm{d}x_2 \ldots \mathrm{d}x_n &= \delta_{il}\,. \qquad (4.46)
\end{aligned}$$

Das Integral ist definitionsgemäß gleich dem Matrixelement V_{ki} der Kovarianzmatrix, und deshalb lautet Gleichung (4.46) wegen der Symmetrie der Matrix $\boldsymbol{G}$

$$\sum_k G_{lk} \cdot V_{ki} = \delta_{li} \quad \text{oder} \quad \boldsymbol{G} \cdot \boldsymbol{V} = \mathbf{1}\,,$$

woraus man sieht, daß $\boldsymbol{G} = \boldsymbol{V}^{-1}$ ist.

4.9 Transformation von Wahrscheinlichkeitsdichten

4.9.1 Transformation einer einzelnen Variablen

Die Wahrscheinlichkeitsdichte $f_x(x)$ der Variablen x sei vorgegeben. Falls x in eine andere Variable y transformiert wird vermöge $y = y(x)$, so besteht die Aufgabe in der Berechnung der Wahrscheinlichkeitsdichte $f_y(y)$ in der neuen Variablen y.
Zur Herleitung des Tranformationsgesetzes bedenkt man, daß das Intervall $(x,\, x + \mathrm{d}x)$ transformiert wird in das Intervall $(y,\, y + \mathrm{d}y)$, und daß die Flächen unter den Kurven der beiden Wahrscheinlichkeitsdichten in den jeweiligen Intervallen gleich sein müssen, also $f_x(x)\mathrm{d}x = f_y(y)\mathrm{d}y$, woraus folgt

$$f_y(y) = f_x(x(y)) \left| \frac{\mathrm{d}x}{\mathrm{d}y} \right| \,,$$

wobei $x = x(y)$ die inverse Funktion zu $y(x)$ ist, d.h. $y \equiv y(x(y))$. Falls die Transformation nicht eineindeutig ist, muß über alle Zweige summiert werden (innerhalb eines Zweigs darf das Vorzeichen der Ableitung $\mathrm{d}y/\mathrm{d}x$ sich nicht ändern). Die Vorschrift ist dann

$$f_y(y) = \sum_{\text{Zweige}} \frac{f_x(x(y))}{|\mathrm{d}y/\mathrm{d}x|}\,.$$

Beispiel 4.22 Die Transformation $x \to x^2$.

Die Transformation $x \to y = x^2$ ist nicht eineindeutig (außer man läßt nur ein Vorzeichen für x zu); deshalb gibt es zwei Zweige, einen für positive und einen für negative x. Mit $\mathrm{d}y/\mathrm{d}x = 2x$ kann das Resultat geschrieben werden als

$$f_y(y) = \frac{1}{2|x|}\left(f_x(x) + f_x(-x)\right) = \frac{1}{2\sqrt{y}}\left(f_x(+\sqrt{y}) + f_x(-\sqrt{y})\right)\,.$$

Transformation von Mittelwert und Varianz, Fehlerfortpflanzung. Nichtlineare Transformationen können zu komplizierten Ausdrücken führen. Jedoch gibt es eine einfache Näherung für den Mittelwert. Zunächst entwickelt man

$$y(x) = y\left(\langle x\rangle\right) + \left(x - \langle x\rangle\right) \left.\frac{\mathrm{d}y}{\mathrm{d}x}\right|_{x=\langle x\rangle} + \frac{1}{2}\left(x - \langle x\rangle\right)^2 \left.\frac{\mathrm{d}^2 y}{\mathrm{d}x^2}\right|_{x=\langle x\rangle} + \ldots$$

und erhält einen Näherungswert für den Mittelwert, indem man nur Terme bis zur zweiten Ordnung mitnimmt:

$$E[y] \approx y(\langle x\rangle) + E[x-\langle x\rangle]\left.\frac{\mathrm{d}y}{\mathrm{d}x}\right|_{x=\langle x\rangle} + \frac{1}{2}E\left[(x-\langle x\rangle)^2\right]\left.\frac{\mathrm{d}^2y}{\mathrm{d}x^2}\right|_{x=\langle x\rangle}$$

Nach Definition ist der zweite Term auf der rechten Seite null und der dritte ist proportional σ_x^2. Deshalb ist der Mittelwert der transformierten Variablen y

$$\langle y\rangle = y(\langle x\rangle) + \frac{1}{2}\sigma_x^2\left.\frac{\mathrm{d}^2y}{\mathrm{d}x^2}\right|_{x=\langle x\rangle} + \dots \ . \tag{4.47}$$

Die Gleichung enthält einen Korrekturterm der zweiten Ordnung; jedoch hat die Transformation $x \to y$ des gemessenen Wertes x bereits einen Fehler der ersten Ordnung wegen des Fehlers σ_x von x, und deshalb wird der Term zweiter Ordnung oft weggelassen. Die Varianz wird folgendermaßen näherungsweise transformiert: Man entwickelt y um den Mittelwert $\langle x\rangle$

$$y(x) = y(\langle x\rangle) + \left.\frac{\mathrm{d}y}{\mathrm{d}x}\right|_{\langle x\rangle}((x-\langle x\rangle) + \dots \ .$$

Die Varianz von y ist, mit $\langle y\rangle \approx y(\langle x\rangle)$,

$$V[y] = E\left[(y-\langle y\rangle)^2\right] = E\left[\left(\left.\frac{\mathrm{d}y}{\mathrm{d}x}\right|_{\langle x\rangle}\cdot(x-\langle x\rangle)\right)^2\right]$$
$$= \left(\frac{\mathrm{d}y}{\mathrm{d}x}\right)^2 \cdot E\left[(x-\langle x\rangle)^2\right] = \left(\frac{\mathrm{d}y}{\mathrm{d}x}\right)^2 \cdot V[x] \ .$$

Dies ist das Gesetz der *Fehlerfortpflanzung* für eine einzige Zufallsvariable.

Lineare Transformationen. Dieser Fall ist relativ einfach. Die allgemeinste lineare Transformation ist $y = ax + b$ und deshalb

$$x = (y-b)/a \quad \text{und} \quad f_y(y) = \frac{1}{|a|}f_x\left(\frac{y-b}{a}\right) \ .$$

Die transformierte Wahrscheinlichkeitsdichte in y ist verschoben, und sie hat eine Maßstabsänderung erfahren. Mittelwert und Varianz in der neuen y Variablen sind

$$\begin{aligned} E[y] &= a\cdot E[x] + b \qquad \langle y\rangle = a\langle x\rangle + b \\ V[y] &= E\left[(ax+b-(a\langle x\rangle+b))^2\right] = a^2E\left[(x-\langle x\rangle)^2\right] = a^2V[x] \ . \end{aligned}$$

4.9.2 Transformation mehrerer Variabler und Fehlerfortpflanzung

Lineare Transformation. Eine lineare Transformation sei gegeben durch $\boldsymbol{y} = \boldsymbol{B}\boldsymbol{x}$, welche den n-Vektor $\boldsymbol{x}$ mit Mittelwert $\boldsymbol{\mu_x}$ und Kovarianzmatrix $\boldsymbol{V}(\boldsymbol{x})$ in den m-Vektor $\boldsymbol{y}$ überführt, vermöge der $m \times n$ Matrix $\boldsymbol{B}$. Man beachte, daß $B_{ik} = \partial y_i/\partial x_k$, und daß im allgemeinen n und m verschieden sein können. Die Vektoren der Mittelwerte (Erwartungswerte) $\boldsymbol{\mu_y}$ von $\boldsymbol{y}$ und die Kovarianzmatrix $\boldsymbol{V}[\boldsymbol{y}]$ von $\boldsymbol{y}$ sind gegeben durch

$$\langle \boldsymbol{y}\rangle = \boldsymbol{\mu_y} = \boldsymbol{B}\boldsymbol{\mu_x} \quad \text{und} \quad \boldsymbol{V}[\boldsymbol{y}] = \boldsymbol{B}\boldsymbol{V}[\boldsymbol{x}]\ \boldsymbol{B}^T \ , \tag{4.48}$$

wobei $\boldsymbol{B}^T$ die zu $\boldsymbol{B}$ transponierte Matrix ist. Der Ausdruck für $\boldsymbol{V}[\boldsymbol{y}]$ ist das *Fehlerfortpflanzungsgesetz*. Die Gleichungen (4.48) sind leicht abzuleiten

$$\langle \boldsymbol{y} \rangle = E[\boldsymbol{y}] = E[\boldsymbol{B}\boldsymbol{x}] = \boldsymbol{B}E[\boldsymbol{x}] = \boldsymbol{B}\boldsymbol{\mu}_{\boldsymbol{x}} ,$$

weil der Erwartungswert ein linearer Operator ist. Die Kovarianzmatrix $\boldsymbol{V}[\boldsymbol{y}]$ von y ist gegeben durch

$$\begin{aligned} \boldsymbol{V}[\boldsymbol{y}] &= E\left[(\boldsymbol{y}-\boldsymbol{\mu}_{\boldsymbol{y}})(\boldsymbol{y}-\boldsymbol{\mu}_{\boldsymbol{y}})^T)\right] = E\left[(\boldsymbol{B}\boldsymbol{x}-\boldsymbol{B}\boldsymbol{\mu}_{\boldsymbol{x}})(\boldsymbol{B}\boldsymbol{x}-\boldsymbol{B}\boldsymbol{\mu}_{\boldsymbol{x}})^T\right] \\ &= E\left[\boldsymbol{B}(\boldsymbol{x}-\boldsymbol{\mu}_{\boldsymbol{x}})(\boldsymbol{x}-\boldsymbol{\mu}_{\boldsymbol{x}})^T\boldsymbol{B}^T\right] \\ &= \boldsymbol{B}\cdot E\left[(\boldsymbol{x}-\boldsymbol{\mu}_{\boldsymbol{x}})(\boldsymbol{x}-\boldsymbol{\mu}_{\boldsymbol{x}})^T\right]\boldsymbol{B}^T = \boldsymbol{B}\boldsymbol{V}[\boldsymbol{x}]\,\boldsymbol{B}^T \end{aligned}$$

Die Berechnung der multidimensionalen Wahrscheinlichkeitsdichte geht nach bewährten Regeln für den Spezialfall $n = m$, d.h. falls die Zahl der Variablen vor und nach der Transformation gleich ist. Dann ist $\boldsymbol{B}$ eine quadratische Matrix und die Wahrscheinlichkeitsdichte ist

$$f_y(\boldsymbol{y}) = f_x(\boldsymbol{x}(\boldsymbol{y}))\ \det|\boldsymbol{B}^{-1}|$$

mit $\boldsymbol{x}(\boldsymbol{y}) = \boldsymbol{B}^{-1}\boldsymbol{y}$. Diese Gleichung ist ein Spezialfall von Gleichung (4.49).

Allgemeiner Fall. Die Transformation ist durch Gleichungen der Form

$$y_i = y_i(x_1, x_2, \ldots x_n) = y_i(\boldsymbol{x})$$

gegeben. Falls die Zahl der x- und y-Variablen gleich ist, dann kann die Wahrscheinlichkeitsdichte $f_y(\boldsymbol{y})$ wenigstens im Prinzip aus $f_x(\boldsymbol{x})$ berechnet werden. Wie in Kapitel 4.9.1 dargestellt, müssen die Wahrscheinlichkeiten aufsummiert über gleichgroße infinitesimale Gebiete in den Variablenräumen x_i und y_i gleich sein:

$$f_x(\boldsymbol{x})\,\mathrm{d}x_1\,\mathrm{d}x_2\ldots\,\mathrm{d}x_n = f_y(\boldsymbol{y})\,\mathrm{d}y_1\,\mathrm{d}y_2\ldots\,\mathrm{d}y_n ,$$

wobei $f_x(\boldsymbol{x})$ und $f_y(\boldsymbol{y})$ die multidimensionalen Wahrscheinlichkeitsdichten der Variablen $\boldsymbol{x}$ und $\boldsymbol{y}$ sind. Das Transformationsgesetz ist dann

$$f_y(\boldsymbol{y}) = f_x(\boldsymbol{x})\cdot J\left(\frac{x}{y}\right) , \tag{4.49}$$

wobei die x_i in den y_i ausgedrückt werden müssen, und $J(x/y)$ die Jakobideterminante ist, gegeben durch

$$J\left(\frac{x}{y}\right) = \left|\frac{\partial x_1 \partial x_2 \cdots \partial x_n}{\partial y_1 \partial y_2 \cdots \partial y_n}\right| .$$

In dem allgemeinen Fall einer nichtlinearen Transformation von dem n-Vektor $\boldsymbol{x}$ in den m-Vektor $\boldsymbol{y}$ ist meist nur eine Näherungsrechnung für die Kovarianzmatrix möglich. Das Standardverfahren benutzt die Matrix $\boldsymbol{B}$ der Ableitungen

$$\boldsymbol{B} = \begin{pmatrix} \partial y_1/\partial x_1 & \partial y_1/\partial x_2 & \ldots & \partial y_1/\partial x_n \\ \partial y_2/\partial x_1 & \partial y_2/\partial x_2 & \ldots & \partial y_2/\partial x_n \\ \ldots & & & \\ \ldots & & & \\ \partial y_m/\partial x_1 & \partial y_m/\partial x_2 & \ldots & \partial y_m/\partial x_n \end{pmatrix} ,$$

wobei jede Ableitung an der Stelle des Mittelwertes $\boldsymbol{\mu}_y$ berechnet wird. Die folgende wichtige Relation zwischen den Kovarianzmatrizen $V[y]$ und $V[x]$ heißt **allgemeines Gesetz der Fehlerfortpflanzung:**

$$\boldsymbol{V}[\boldsymbol{y}] = \boldsymbol{B}\boldsymbol{V}[\boldsymbol{x}]\boldsymbol{B}^T, \tag{4.50}$$

wovon Gleichung (4.48) ein Spezialfall ist. Sie kann abgeleitet werden, indem man $y_i(\boldsymbol{x})$ um seinen Mittelwert $\mu_i = y_i(\boldsymbol{\mu_x})$ entwickelt:

$$y_i(\boldsymbol{x}) - \mu_i = \sum_{k=1}^{n} (x_k - \langle x_k \rangle) \frac{\partial y_i}{\partial x_k} + \cdots ,$$

und das V_{il}-Element der Kovarianzmatrix ist dann

$$\begin{aligned} V_{il}[y] &= E\left[(y_i - \mu_i)(y_l - \mu_l)\right] \\ &\approx E\left[\left(\sum_k (x_k - \langle x_k \rangle)\frac{\partial y_i}{\partial x_k}\right)\left(\sum_j (x_j - \langle x_j \rangle)\frac{\partial y_l}{\partial x_j}\right)\right] \\ &= E\left[\sum_{kj} (x_k - \langle x_k \rangle)(x_j - \langle x_j \rangle)\frac{\partial y_i}{\partial x_k}\frac{\partial y_l}{\partial x_j}\right] . \end{aligned}$$

Weil $\partial y_i / \partial x_k$ das ik-Element der Matrix $\boldsymbol{B}$ ist, sieht man leicht, daß V_{il} das il-Element in Gleichung (4.50) ist.

4.9.3 Beispiele zur Fehlerfortpflanzung

Spezialfall (keine Korrelationen). Wenn die Variablen x_i nicht korreliert sind, dann ist die Kovarianzmatrix $\boldsymbol{V}[x]$ diagonal, und die Diagonalelemente der Kovarianzmatrix von y sind dann

$$V_{ii} = \sigma_{yi}^2 = V[y_i] = \sum_{k=1}^{n} \left(\frac{\partial y_i}{\partial x_k}\right)^2 \cdot V[x_k] = \sum_{k=1}^{n} \left(\frac{\partial y_i}{\partial x_k}\right)^2 \cdot \sigma_{x_k}^2 . \tag{4.51}$$

Jedoch, selbst wenn die x_i-Werte nicht korreliert sind, so sind es die y_i-Werte im allgemeinen sehr wohl, d.h. $V_{ik}[y] \neq 0$ für $i \neq k$ (siehe Beispiel 4.26).

Beispiel 4.23 Transformation zweier Variabler $x_1, x_2 \to y_1, y_2$.

Die Varianz von y_1 wird

$$\begin{aligned} V[y_1] &= \sum_{k=1}^{2}\sum_{j=1}^{2} B_{1k} V_{kj}(\mathbf{x}) B_{1j} \\ &= \left(\frac{\partial y_1}{\partial x_1}\right)^2 \sigma^2(x_1) + \left(\frac{\partial y_1}{\partial x_2}\right)^2 \sigma^2(x_2) + 2\frac{\partial y_1}{\partial x_1}\frac{\partial y_1}{\partial x_2}\mathrm{cov}(x_1, x_2) . \end{aligned} \tag{4.52}$$

Bei unkorrelierten Variablen x_1, x_2 fehlt der dritte Term, und man erhält die vereinfachte Form des Fehlerfortpflanzungsgesetzes Gleichung (4.51).

Beispiel 4.24 Fehlerfortpflanzung – einfache Fälle.

Das Volumen eines Zylinders ist $v = \pi r^2 h$ mit $r =$ Radius und $h =$ Höhe. Wenn r und h mit den Standardabweichungen σ_r und σ_h gemessen sind, dann ist die Varianz für die Bestimmung des Volumens v gegeben durch

$$V[v] = \sigma_v^2 = \left(\frac{\partial v}{\partial r}\right)^2 \sigma_r^2 + \left(\frac{\partial v}{\partial h}\right)^2 \sigma_h^2 = (2\pi rh)^2 \sigma_r^2 + (\pi r^2)^2 \sigma_h^2 .$$

Der relative Fehler der Volumenbestimmung ist

$$\frac{\sigma_v}{v} = \sqrt{\left(\frac{2\sigma_r}{r}\right)^2 + \left(\frac{\sigma_h}{h}\right)^2} .$$

Ein noch einfacherer Fall ist die Summe oder Differenz zweier unkorrelierter Zufallsvariabler, $y = x_1 \pm x_2$,

$$V[y] = \left(\frac{\partial y}{\partial x_1}\right)^2 \sigma_{x_1}^2 + \left(\frac{\partial y}{\partial x_2}\right)^2 \sigma_{x_2}^2 = \sigma_{x_1}^2 + \sigma_{x_2}^2 .$$

Ein anderer einfacher Fall ist die Transformation einer einzigen Variablen, $y = f(x)$,

$$V[y] = \left(\frac{\mathrm{d}y}{\mathrm{d}x}\right)^2 V[x] .$$

Es folgt daraus, daß für den Fall $y = ax^n$ der relative Fehler von y gegeben ist durch $\sigma_y/y = n\sigma_x/x$. Ist zum Beispiel der Radius einer Kugel mit 1% Genauigkeit bekannt, so ist ihr Volumen nur mit 3% Genauigkeit bekannt.

Beispiel 4.25 Rotation.

Rotation um den Winkel θ ist ein wichtiger Spezialfall. Die Variablen x_1, x_2 und y_1, y_2 seien durch eine Rotation um die z-Achse verknüpft: $\boldsymbol{y} = \boldsymbol{B}\boldsymbol{x}$, wobei $\boldsymbol{B}$ die Rotationsmatrix und $\boldsymbol{B}^T$ die transponierte Rotationsmatrix ist

$$\boldsymbol{B} = \begin{pmatrix} c & s \\ -s & c \end{pmatrix} \qquad \boldsymbol{B}^T = \begin{pmatrix} c & -s \\ s & c \end{pmatrix}$$

mit den Abkürzungen $c = \cos\theta$ und $s = \sin\theta$. Die Kovarianzmatrix $\boldsymbol{V}[\boldsymbol{x}]$ von x_1, x_2 sei

$$\boldsymbol{V}[\boldsymbol{x}] = \begin{pmatrix} \sigma_1^2 & \mathrm{cov}(x_1, x_2) \\ \mathrm{cov}(x_1, x_2) & \sigma_2^2 \end{pmatrix} ,$$

wobei $\sigma_1^2 = V[x_1]$ und $\sigma_2^2 = V[x_2]$. Die Kovarianzmatrix von y_1, y_2 ist dann

$$\boldsymbol{V}[\boldsymbol{y}] = \boldsymbol{B}\boldsymbol{V}[\boldsymbol{x}]\boldsymbol{B}^T .$$

Die Elemente von $\boldsymbol{V}[\boldsymbol{y}]$ sind

$$\begin{aligned} V_{11} &= \sigma_{y1}^2 = V[y_1] = c^2{\sigma_1}^2 + s^2{\sigma_2}^2 + 2s \cdot c\ \mathrm{cov}(x_1, x_2) \\ V_{22} &= \sigma_{y2}^2 = V[y_2] = c^2{\sigma_2}^2 + s^2{\sigma_1}^2 - 2s \cdot c\ \mathrm{cov}(x_1, x_2) \\ V_{12} &= V_{y,21} = \mathrm{cov}(y_1, y_2) = s \cdot c({\sigma_2}^2 - {\sigma_1}^2) + (c^2 - s^2)\mathrm{cov}(x_1, x_2) . \end{aligned}$$

Wenn $\mathrm{cov}(y_1, y_2) \neq 0$, dann sind y_1 und y_2 korreliert. Diese Korrelation kann jedoch durch eine Rotation um den Winkel θ_0 beseitigt werden, d.h. $\mathrm{cov}(y_1, y_2)$ kann zu null gemacht werden. Die Bedingung lautet

$$\tan 2\theta_0 = \frac{2\mathrm{cov}(x_1, x_2)}{{\sigma_1}^2 - {\sigma_2}^2} .$$

Weiterhin ist beachtenswert, daß selbst wenn x_1, x_2 unkorreliert sind, die transformierten Variablen y_1, y_2 im allgemeinen korreliert sein können. Dasselbe gilt auch für allgemeinere Transformationen, wie das nächste Beispiel zeigt.

Beispiel 4.26 Transformation in Polarkoordinaten.

Zwei unkorrelierte Variable x, y haben die Kovarianzmatrix $\boldsymbol{V}$ von der Form

$$\boldsymbol{V} = \begin{pmatrix} {\sigma_x}^2 & 0 \\ 0 & {\sigma_y}^2 \end{pmatrix} .$$

Man transformiert in Polarkoordinaten:

$$\begin{aligned} r^2 &= x^2 + y^2 \\ \theta &= \arctan(y/x) . \end{aligned}$$

Die $\boldsymbol{B}$-Matrix ist

$$\boldsymbol{B} = \begin{pmatrix} \frac{\partial r}{\partial x} & \frac{\partial r}{\partial y} \\ \frac{\partial \theta}{\partial x} & \frac{\partial \theta}{\partial y} \end{pmatrix} = \begin{pmatrix} \frac{x}{r} & \frac{y}{r} \\ \frac{-y}{r^2} & \frac{x}{r^2} \end{pmatrix} .$$

Die Elemente der Kovarianzmatrix in r und θ sind

$$\begin{aligned} V_{rr} &= V[r] = \frac{1}{r^2}(x^2{\sigma_x}^2 + y^2{\sigma_y}^2) \\ V_{\theta\theta} &= V[\theta] = \frac{1}{r^4}(y^2{\sigma_x}^2 + x^2{\sigma_y}^2) \\ V_{r\theta} &= \mathrm{cov}(r, \theta) = \frac{xy}{r^3}({\sigma_y}^2 - {\sigma_x}^2) . \end{aligned}$$

Die Variablen r, θ sind offensichtlich korreliert (außer wenn $\sigma_x^2 = \sigma_y^2$).

4.10 Faltung

Zwei Zufallsvariable x und y seien durch ihre Wahrscheinlichkeitsdichten $f_x(x)$ und $f_y(y)$ gegeben. Offensichtlich ist ihre Summe $w = x + y$ ebenfalls eine Zufallsvariable. Die Wahrscheinlichkeitsdichte der Summe w sei $f_w(w)$. Sie wird erhalten durch eine **Faltung** von x mit y:

$$\begin{aligned} f_w(w) = \int\int f_x(x) f_y(y) \delta(w - x - y)\, \mathrm{d}x\, \mathrm{d}y \\ = \int f_x(x) f_y(w - x)\, \mathrm{d}x = \int f_y(y) f_x(w - y)\, \mathrm{d}y . \end{aligned}$$

Beispiel 4.27 Faltung zweier Gleichverteilungen.

Angenommen $f_x(x)$ und $f_y(y)$ sind beide von der Form einer Gleichverteilung

$$f_x(x) = \begin{cases} 1 & \text{falls } 0 \le x \le 1 \\ 0 & \text{sonst} \end{cases}$$

und dieselben Gleichungen für $f_y(y)$. Die Wahrscheinlichkeitsdichte von $w = x + y$ hat eine dreieckige Form mit $f_w(0) = f_w(2) = 0$ und $f_w(1) = 1$ (siehe Abbildung 4.23). Bei der Herleitung müssen mühsam die Integrationsgrenzen beachtet werden.

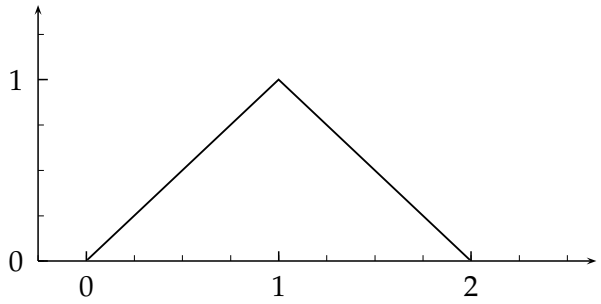


Abbildung 4.23: Faltung von zwei Gleichverteilungen.

Faltung zweier Gauß-Verteilungen. Gegeben sind zwei Gauß-Verteilungen von x und y mit Mittelwerten μ_x und μ_y und Varianzen ${\sigma_x}^2$ und ${\sigma_y}^2$. Durch direkte Berechnung findet man, daß die Wahrscheinlichkeitsdichte von $w = x + y$ wieder eine Gauß-Verteilung ist, mit Mittelwert $\mu_w = \mu_x + \mu_y$ und Varianz $V[w] = {\sigma_w}^2 = {\sigma_x}^2 + {\sigma_y}^2$.
Manchmal läßt sich eine Faltung elegant über die charakteristische Funktion berechnen (siehe Kapitel 4.5.4): Die charakteristische Funktion von $f_w(w)$ ist nach Definition

$$\phi_w(t) = \int e^{iwt} f_w(w)\,\mathrm{d}w = \int e^{iwt} f_x(x) f_y(y) \delta(w - x - y)\,\mathrm{d}w\,\mathrm{d}x\,\mathrm{d}y = \int e^{ixt} f_x(x)\,\mathrm{d}x \int e^{iyt} f_y(y)\,\mathrm{d}y = \phi_x(t) \cdot \phi_y(t)$$

Die charakteristische Funktion der Faltung zweier Variablen erhält man als das Produkt ihrer charakteristischen Funktionen.

Beispiel 4.28 Faltung zweier Gauß-Verteilungen.

Gegeben sind $f_x(x) = Ne^{-x^2/2{\sigma_x}^2}$ und $f_y(y) = Ne^{-y^2/2{\sigma_y}^2}$. Ihre charakteristischen Funktionen sind (Gleichung (4.23)):

$$\begin{aligned} \phi_x &= N_x \cdot e^{-t^2{\sigma_x}^2/2} \qquad\qquad \phi_y = N_y \cdot e^{-t^2{\sigma_y}^2/2} \\ \phi_w &= \phi_x \cdot \phi_y = N_w \cdot e^{-t^2({\sigma_x}^2+{\sigma_y}^2)/2}\,. \end{aligned}$$

Der letzte Ausdruck ist die charakteristische Funktion von

$$f_w(w) = N_w \cdot e^{-w^2/2({\sigma_x}^2+{\sigma_y}^2)}\,,$$

und deshalb folgt $w = x + y$ wieder einer Gauß-Verteilung mit Varianz $\sigma_x^2 + \sigma_y^2$.

Beispiel 4.29 Faltung zweier Poisson-Verteilungen.

Diese Faltung führt wieder zu einer Poisson-Verteilung. Man geht aus von zwei Poisson-Verteilungen mit Mittelwerten μ_x und μ_y

$$\begin{aligned} P_x &= \frac{{\mu_x}^x \cdot e^{-\mu_x}}{x!} \qquad P_y = \frac{{\mu_y}^y \cdot e^{-\mu_y}}{y!} \\ w &= x + y\,. \end{aligned}$$

Die charakteristische Funktion von P_x ist

$$\begin{aligned}
\phi_x &= \sum_0^\infty \frac{e^{itx} e^{-\mu_x} {\mu_x}^x}{x!} = e^{-\mu_x} \sum_0^\infty \frac{\left(\mu_x \cdot e^{it}\right)^x}{x!} \\
&= e^{-\mu_x} \exp\left(\mu_x \cdot e^{it}\right) = \exp\left(\mu_x (e^{it} - 1)\right) \\
\phi_y &= \exp\left(\mu_y (e^{it} - 1)\right) \\
\phi_w &= \phi_x \cdot \phi_y = \exp\left((\mu_x + \mu_y)(e^{it} - 1)\right) .
\end{aligned}$$

Die letzte Gleichung ist die charakteristische Funktion einer Poisson-Verteilung mit Mittelwert $\mu_w = \mu_x + \mu_y$.

5 Monte Carlo-Methoden

5.1 Einführung

Ein historisches Beispiel für die Anwendung von Monte Carlo-Methoden ist Buffon's Nadel[1]. Eine Nadel der Länge ℓ wird willkürlich auf eine Fläche mit gleich weit voneinander entfernten parallelen Geraden geworfen, mit einem Abstand ℓ zwischen den Geraden. Man kann zeigen, daß die Häufigkeit $p = k/n$ für k = Zahl der Ereignisse, mit der die Nadel eine Gerade trifft, bei insgesamt n Versuchen im Mittel $2/\pi$ ist. In Abbildung 5.1 ist die Größe $2/p$ aufgetragen; sie zeigt, wie man sich dem Wert $2/p = \pi$ mit zunehmender Zahl der Versuche annähert, und wie die Fluktuationen allmählich abnehmen.

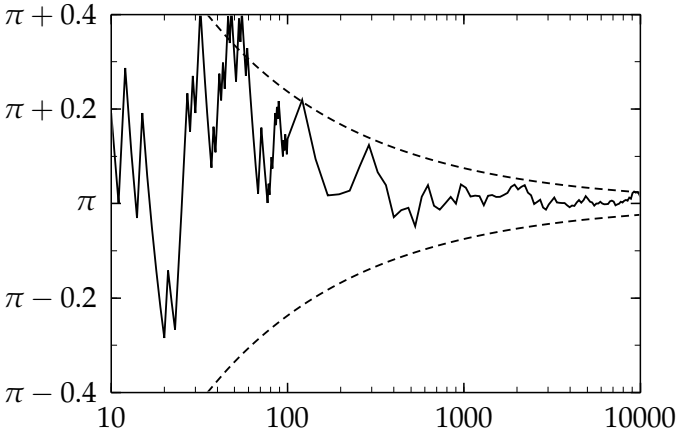


Abbildung 5.1: Bestimmung der Zahl π mit der Methode von Buffon. Die Linie zeigt den sich der Zahl π annähernden Wert von $2/p$ als Funktion der Zahl n der Versuche. Die gestrichelten Kurven geben die Größe des erwarteten statistischen Fehlers von $\pm 2.37/\sqrt{n}$ an.

Ein anderes Beispiel ist das Abpflücken von Blütenblättern: „Sie liebt mich, liebt mich nicht, liebt mich, $\cdots$".

Die Verfügbarkeit schneller Rechner hat die Monte Carlo-(MC-)Methode zu einem unentbehrlichen Hilfsmittel bei vielen statistischen Problemen gemacht, die zu kompliziert sind, um auf analytischem Wege gelöst zu werden. Selbst wenn man eine Lösung in geschlossener Form finden kann, ist es oft einfacher und schneller, die MC-Methode anzuwenden, und selbst wenn man eine analytische Lösung hat, bietet die MC-Methode eine unabhängige Überprüfung.

Grundlegend für die MC-Methode ist die Erzeugung von Zufallszahlen. Es gibt physikalische Prozesse, die wirklich zufällig sind, und im Prinzip könnte man sie verwenden, um Zufallszahlen zu erhalten. In der Praxis funktioniert dies jedoch nicht, und man benutzt Zahlen, die man durch einen Algorithmus mit Hilfe eines Rechners erhält. Da nun ein Rechner eine deterministische Maschine ist, können die Zahlen, die er produziert, nicht wirklich zufällig sein. Zum Beispiel wird jeder Algorithmus schließlich die Folge *zufälliger* Zahlen wiederholen; die Länge dieser Folge wird Periode genannt. Die Erzeugung von Zufallszahlen ist also keine einfache Sache. Eine Periode von $2^{36} \approx 10^{11}$ mag zum Beispiel lang erscheinen, sie muß jedoch mit den $\gg 10^9$ Operationen/s verglichen werden, die Rechner mit Parallelarchitektur auszuführen imstande sind. Gute Algorithmen für Zufallszahlen haben weitaus längere Perioden als die genannte. Überdies können *zufällige* Zahlen Korrelationen innerhalb der Zahlenfolgen aufweisen, die eine MC-Rechnung total zunichte machen können. Deshalb müssen Algorithmen für Zufallszahlen sorgfältig geprüft werden. Aber selbst der beste Algorithmus wird, genau genommen, nur Pseudo-Zufallszahlen liefern, selbst wenn die Güte dieser Zahlen allen normalen praktischen Anforderungen genügt. In Anlehnung an die allgemeine Praxis werden diese Pseudo-Zufallszahlen im weiteren Verlauf einfach Zufallszahlen genannt.

[1] Graf G.L.L. von Buffon, 1707-1788

Eine ausführliche Behandlung findet man in der klassischen Arbeit von Knuth [42], eine weitere Auswahl von Arbeiten ist [58, 78, 79, 13, 27, 28, 10].

5.2 Zufallszahlengeneratoren

MC-Rechnungen benötigen Zufallszahlen für eine Vielfalt von verschiedenen statistischen Anwendungen. Üblicherweise werden zunächst Zufallszahlen u entsprechend einer gleichförmigen Wahrscheinlichkeitsdichte durch einen Algorithmus erzeugt und in Zufallszahlen anderer Verteilungen transformiert, falls erforderlich. Die gleichförmige Wahrscheinlichkeitsdichte zwischen den Grenzen 0 und 1 bezeichnet man mit $U(0,1)$, die Zufallszahlen selbst mit u.

$$U(0,1) = \begin{cases} 1 & \text{falls } 0 < u < 1 \\ 0 & \text{sonst} \end{cases} .$$

So wie hier formuliert, sind die genauen Werte (0,1) der Endpunkte ausgenommen, da sie Probleme verursachen können. Weiterhin werden oft Zufallszahlen der standardisierten Normalverteilung, bezeichnet mit $N(0,1)$ (Mittelwert 0 und Varianz 1), benötigt; diese Zufallszahlen werden mit z_i bezeichnet. In diesem Kapitel wird durchgehend diese Notation

$$\begin{aligned} u_i &\in U(0,1) \quad \text{gleichförmig} \\ z_i &\in N(0,1) \quad \text{standardisierte Normalverteilung} \end{aligned}$$

verwendet.

Erzeugung gleichförmiger Zufallszahlen. Einer der einfachsten Generatoren ist der allgemeine *linear kongruente Generator*

$$I_j = (a \cdot I_{j-1} + c) \mod m \qquad u_j = I_j/m$$

Er benötigt drei ganzzahlige Konstanten, den Multiplikator a, den Summanden c und den Modul m. Generatoren mit Summand $c = 0$ werden *multiplikative linear kongruente Generatoren* genannt. Beginnt man mit einer Anfangszahl I_0, genannt *Saatzahl (seed)*, so wird eine Folge I_1, I_2, ... von ganzen Zahlen zwischen 0 und $m-1$ gebildet. Eine reelle Zufallszahl u_j im Bereich 0 bis 1 wird mittels Division durch m berechnet. Im obigen Algorithmus kann der genaue Wert $u_j = 1$ niemals auftreten, und der genaue Wert $u_j = 0$ wird gewöhnlich durch einen Extratest ausgeschlossen. Die Zahlenfolge ist periodisch mit einer maximalen Periode von m. Die wirkliche Periode hängt von den Konstanten a und c ab, und das Maximum m wird unter folgenden Bedingungen erreicht:

1. $c \neq 0$;
2. der Summand c und der Modul m haben keinen gemeinsamen Teiler;
3. $(a-1)$ ist ein Vielfaches der Primzahl p für jede Primzahl p, die ein Teiler des Moduls m ist;
4. $(a-1)$ ist ein Vielfaches von 4, falls der Modul m ein Vielfaches von 4 ist.

Zufallszahlen sollten unkorreliert sein, d.h. die Wahrscheinlichkeit des Auftretens einer Zufallszahl u_j in einer Folge sollte nicht von der vorangehenden Zufallszahl u_{j-1} oder anderen vorangehenden Zahlen abhängen. Wegen des deterministischen Charakters des Generators lassen sich solche Korrelationen zwischen den Zufallszahlen in einer Folge nicht völlig vermeiden. Man kann zum Beispiel zeigen, daß k-Tupel aufeinander folgender Zufallszahlen, im k-dimensionalen Raum aufgezeichnet, auf $(k-1)$-dimensionalen Hyperebenen liegen, wenn sie nach obiger Methode gebildet werden. Es gibt höchstens $m^{1/k}$ solcher Ebenen, und eine gute Wahl der Konstanten a, c und m wird eine gleichmäßige Verteilung der Ebenen erreichen.

Beispiel: für $m = 2^{32} = 4\,294\,967\,296$ und $k = 10$ ist der Wert von $m^{1/k} = 9.19$. Ferner sollte beachtet werden, daß im allgemeinen die weniger signifikanten Bits einer Zufallszahl *weniger zufällig* sind als die Bits höherer Ordnung.

Eine Funktion zur Erzeugung von Zufallszahlen ist gewöhnlich in Rechnern verfügbar. Gewöhnlich gibt es dabei die Möglichkeiten, die *Saat-Zahl* (seed) zu definieren und die aktuelle Zufallszahl zur späteren Fortsetzung der Rechnung zu speichern. Auf diese Weise kann man sicherstellen, daß Programme bei der wiederholten Ausführung statistisch unabhängig laufen und nicht wieder mit derselben ursprünglichen Zufallszahl beginnen.

Ein einfacher Zufallszahlengenerator (in Fortran) ist im Anhang abgedruckt.

Geeignete Konstanten für einen einfachen linear kongruenten Generator mit einer Periode oberhalb von 10^5 sind:

a	205	281	1021	421
c	29573	28411	25673	54773
m	139968	134456	121500	259200

Die Zahlen in der Tabelle sind verhältnismäßig klein und vermeiden die Möglichkeit einer Überschreitung des Zahlenbereichs (*integer overflow*). Die Behandlung eines Overflow müßte zusätzlich codiert werden.

Eine andere Klasse von Zufallszahlen-Generatoren sind *Fibonacci Generatoren*. Eine Zufallszahl u_n wird rekursiv aus vorangehenden Zufallszahlen nach einem Schema berechnet, das der Berechnung von Fibonacci-Zahlen ähnlich ist. Ein Beispiel ist der Generator

$$u_n \;=\; (u_{n-24} + u_{n-55}) \bmod 1 \qquad n \geq 55\,,$$

wobei $x \mod 1 \;=\; x - \mathrm{int}(x)$. Die Periode ist etwa 2^{55}. Diese Generatoren müssen vor der ersten Verwendung initialisiert werden.

Der folgende Generator wurde bei einer sehr verbreiteten MC-Rechnung der *QCD Gitter-Eichtheorie* benutzt:

$$\begin{aligned} x_n &= (x_{n-10} - x_{n-24} - c_{n-1}) \bmod 1 \\ c_n &= 0 \text{ falls } x_n \geq 0\,,\ \text{sonst } c_n = 1\,. \end{aligned}$$

Er erwies sich im *gap test* als unzulänglich (siehe Kapitel 5.3). Um ihn zu verbessern, wird die Chaos-Theorie bemüht, um mit folgendem Rezept einen besseren Generator zu erhalten: Man erzeugt 24 Zufallszahlen, schließt die nächsten 300 aus, usw. Natürlich impliziert dies einen Verlust an Effizienz [50, 36].

Keiner dieser einfachen Generatoren ist in der heutigen Praxis voll befriedigend. Moderne Monte Carlo-Generatoren kombinieren meist zwei oder mehr einfache Zufallszahlen-Generatoren. Zwei Verfahren sind üblich, um Zufallszahlen mit minimalen Korrelationen und außerordentlich großen Periodenlängen zu erhalten.

1. Kombination: Zwei Zufallszahlen werden mit je einem Generator erzeugt, und eine neue Zufallszahl durch Kombination der zwei Zahlen gebildet, wobei die Operationen $+$, $-$ oder bitweises exklusives ODER üblich sind.
2. Durchmischung: Ein Speicher wird mit einer Reihe von Zufallszahlen gefüllt, und das Resultat eines zweiten Zufallszahlen-Generators wird benutzt, um die Adresse der nächsten Zufallszahl im Speicher zu bestimmen. Vor der ersten Benutzung muß der Speicher natürlich mit Zufallszahlen gefüllt werden.

Naiv könnte man denken, daß kompliziertere Formeln wie zum Beispiel $x_{n+1} = \ln(\cos x_n) \bmod 1$ sehr zufällige Zufallszahlen produzieren würden. Dies ist jedoch in der Regel nicht der Fall [42].

Gutbrod's Zufallszahlengenerator [27, 28]. Dieser Generator mit dem Namen `RANSHI`, gedacht für professionelle Anwendungen, wurde ursprünglich für einen i860-Microprocessor konzipiert, ist aber universell einsetzbar. Er hat bisher in Tests keinerlei Unzulänglichkeiten gezeigt, und seine mittlere Periode ist etwa $\sqrt{\pi T/8}$ mit $T = (1 - 1/e) \cdot 2^{n(N+2)}$, wobei n die Zahl der Bits im Rechner-Wort (zum Beispiel $n = 32$) und N die Zahl der Worte im Pufferspeicher ist. Die Periode ist also sehr lang. Der Generator arbeitet nach dem allgemeinen Prinzip, Zufallszahlen in einem Puffer willkürlich zu mischen, in diesem besonderen Fall imitiert der Algorithmus Zusammenstöße von Teilchen mit Spin. Er benutzt einen Puffer von N Worten, die im Prozessor gespeichert sind. Die Zahl N muß groß genug sein, um eine gute Durchmischung sicherzustellen. Um schnell zu sein, führt der Generator Schiebeoperationen statt Multiplikationen aus.

Ein Fortranprogramm `RANSHI`, das gegenüber der Originalversion von Gutbrod leicht verändert ist, befindet sich im Anhang.

5.3 Prüfung von Zufallszahlengeneratoren

Zufallszahlengeneratoren mit einem Defekt wie zum Beispiel Korrelationen können numerische Ergebnisse von MC-Rechnungen verzerren und völlig unbrauchbar machen. Deshalb wurden viele Tests für Zufallszahlengeneratoren entwickelt [42].

Test auf gleichförmige Verteilung. Das Intervall $(0, 1)$ wird in k gleiche Unterintervalle der Länge $1/k$ unterteilt. N Zufallszahlen u_i werden erzeugt, und es wird gezählt, wieviele dieser Zufallszahlen in jedes dieser k Unterintervalle hineinfallen. Nennt man die Zahl der Fälle in jedem Unterintervall N_i, $i = 1, 2, \ldots k$, dann sollte (für $N/k \geq 10$) die Summe

$$\chi^2 = \sum_{i=1}^{k} \frac{(N_i - N/k)^2}{N/k}$$

näherungsweise einer χ^2-Verteilung mit $(k - 1)$ Freiheitsgraden folgen, d.h. im Mittel sollte das Verhältnis $\chi^2/(k - 1)$ eins sein. Man kann analoge Ausdrücke für nicht-gleichmäßige Verteilungen konstruieren. Zum Nachprüfen sollte ein Histogramm der erzeugten Zufallszahlen angefertigt und mit der gewünschten Verteilung verglichen werden.

Einen ähnlichen Test kann man für Paare von Zufallszahlen (u_i, u_j) machen, die ebenfalls gleichmäßig in einem zweidimensionalen Gitter mit Intervallen von gleichem Abstand verteilt sein sollten. Zusätzlich sollte ein *Kolmogorov-Smirnov Test* durchgeführt werden (siehe Kapitel 9.1.5).

Korrelationstest. Wenn n sukzessive Zufallszahlen als die Koordinaten eines Punktes im n-dimensionalen Raum aufgezeichnet sind, so ist die Menge dieser Punkte nicht willkürlich verteilt, sondern liegt auf Hyperebenen. Bei einem guten Zufallszahlengenerator sollten diese Ebenen etwa gleichmäßig verteilt sein. Der Effekt ist ausgeprägter, wenn man nur die Bits niedriger Ordnung nimmt, die also getrennt getestet werden sollten. Ein Programm für diesen Test ist bei [42] zu finden.

Sequenz-(up-down-)Test. Zwei aufeinander folgende Zufallszahlen u_j und u_{j+1} werden verglichen, und das Bit 0 bzw. 1 wird den Fällen $u_{j+1} > u_j$ bzw. $u_{j+1} < u_j$ zugeordnet. Lange Folgen solcher Bits werden erzeugt, und die Anzahl der Folgen von Nullen und Einsen der Länge k bestimmt; diese Zahl wird $N(k)$ genannt. Dann ist offensichtlich $\sum_{k=1}^{N} k \cdot N(k) = N$,

wenn insgesamt $N+1$ Zufallszahlen erzeugt worden sind. Die Summe aller Zahlen $N(k)$ ist $\sum_{k=1}^{N} N(k) = (2N-1)/3$. Für unkorrelierte Zufallszahlen erwartet man im Mittel die folgenden Werte:

$$N(1) = \frac{5N+1}{12} \qquad N(2) = \frac{11N-14}{60} \tag{5.1}$$

$$N(k) = \frac{2\left[\left(k^2+3k+1\right)N - \left(k^3+3k^2-k-4\right)\right]}{(k+3)!}\,. \tag{5.2}$$

Systematische, signifikante Abweichungen von den erwarteten Zahlen weisen auf Korrelationen zwischen Paaren von Zufallszahlen hin.

Lücken-(gap-)Test. Man wählt zwei Zahlen α, β: $0 \le \alpha < \beta \le 1$. Man erzeugt $(r+1)$ Zufallszahlen, die gleichmäßig im Intervall $(0,1)$ verteilt sind. Die Wahrscheinlichkeit, daß die ersten r Zahlen außerhalb des Intervalls (α, β) liegen und die letzte $(r+1)$ste Zahl innerhalb, sollte sein

$$P_r = p(1-p)^r \quad \text{mit } p = \beta - \alpha\,.$$

Random Walk-Test. Man wählt eine Zahl $0 < \alpha \ll 1$. Man bildet eine große Zahl von Zufallszahlen und registriert die Zahl der Fälle r, in denen eine Zufallszahl kleiner als α erscheint. Man erwartet eine Binomialverteilung für r mit $p = \alpha$. Der Test ist sehr empfindlich für große Werte von r. Der Test sollte auch gemacht werden, indem man die Zahl der Zufallszahlen zählt, die größer als $(1-\alpha)$ sind. Man erwartet in diesem Fall natürlich für r dieselbe Verteilung.

5.4 Zufallszahlen für beliebige Verteilungen

Kontinuierliche Zufallsvariable. Die Aufgabe ist, Zufallszahlen x_i gemäß einer Wahrscheinlichkeitsdichte $f(x)$ zu erzeugen. Eine Standardtechnik beginnt mit Zufallszahlen u_i, die entsprechend einer gleichförmigen Verteilung erzeugt werden, und transformiert sie. Wenn $U(0,1)$ die gleichförmige Verteilung ist, muß gelten:

$$f(x)\mathrm{d}x = U(0,1)\,\mathrm{d}u \qquad \int_{-\infty}^{x} f(t)\,\mathrm{d}t = F(x) = u.$$

Die Gleichung $u = F(x)$ kann für die Variable x gelöst werden durch

$$x = F^{(-1)}(u)$$

mit der Umkehrfunktion $F^{(-1)}(u)$. Für eine Folge gleichförmiger Zufallszahlen u_j folgen die Zufallszahlen $x_j = F^{(-1)}(u_j)$ nach Konstruktion der Wahrscheinlichkeitsdichte $f(x)$. Diese direkte Methode kann angewandt werden, wenn das Integral als analytische Funktion $F(x)$ ausgedrückt und wenn die Funktion $u = F(x)$ leicht invertiert werden kann. Die Methode ist in Abbildung 5.2 skizziert.

Diskrete Zufallsvariable. Man betrachtet eine diskrete Zufallsvariable x_i mit einer endlichen Zahl n verschiedener möglicher Werte $x_1, x_2, \ldots x_n$. Eine Methode ist, alle Werte der kumulativen Wahrscheinlichkeit für alle Werte $k = 1, 2, \ldots n$ im voraus zu berechnen:

$$P_{k+1} = \sum_{i=1}^{k} P(x_i) \quad \text{für alle } k \text{ mit } k = 1,\, 2, \ldots n \quad \text{mit } P_1 = 0,\; P_{n+1} = 1\,,$$

wobei $P(x_i)$ die Wahrscheinlichkeit des diskreten Wertes x_i ist. Die kumulative Wahrscheinlichkeit P_k hat Sprünge bei jedem Wert x_i der diskreten Zufallsvariablen. Alle Werte P_k werden in

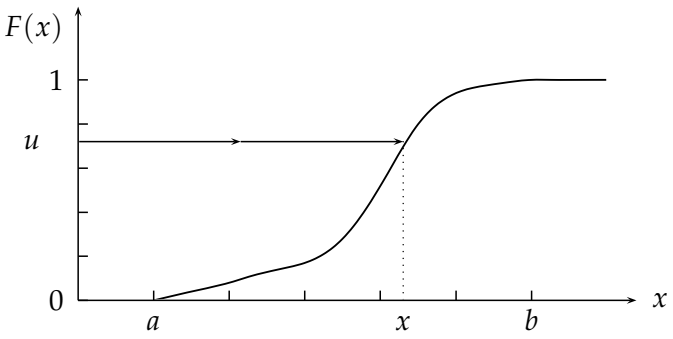


Abbildung 5.2: Erzeugung von Zufallszahlen einer kontinuierlichen Verteilung $f(x)$. Die Abbildung zeigt die kumulative Verteilung $F(x)$. Eine Zufallszahl $u \in U(0,1)$ wird in die Zufallszahl $x = F^{-1}(u)$ transformiert.

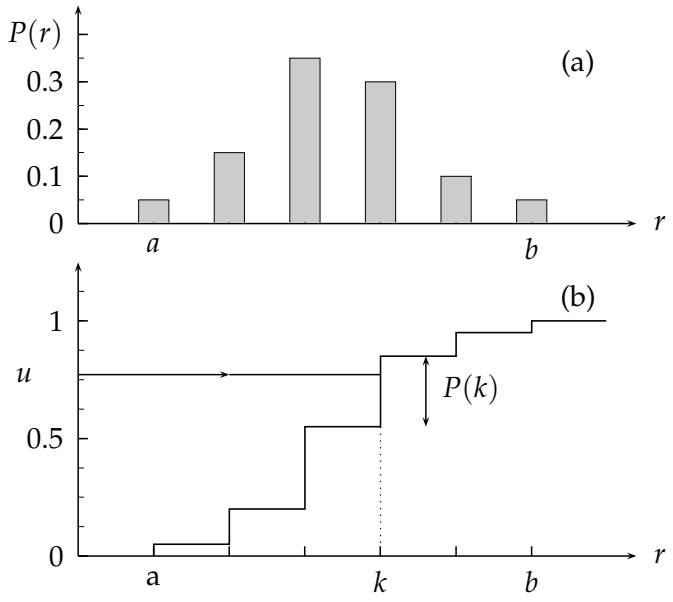


Abbildung 5.3: Erzeugung von Zufallszahlen einer diskreten Verteilung gemäß einem Histogram (a). Abbildung (b) zeigt die kumulative Verteilung. Ein Zufallszahl $u \in U(0,1)$ wird in die diskrete Zufallszahl k transformiert.

einer Tabelle gespeichert. Zu einem gegebenen Zufallswert u hat man dann die Einträge in der Tabelle mit $P_{k-1} \leq u < P_k$ aufzusuchen; k ist dann der Index der gewählten Zufallsvariablen. Dazu kann ein binärer Suchalgorithmus verwendet werden, wie in der folgenden Funktion. Die Methode ist in Abbildung 5.3 skizziert.

Beispiel 5.1 Zufallszahlen aus einer linearen Wahrscheinlichkeitsdichte.

Eine linear anwachsende Wahrscheinlichkeitsdichte ist im Intervall 0 bis 1 gegeben als $f(x) = 2x$ mit $0 \leq x \leq 1$. Die direkte Methode führt auf die Wurzelfunktion:

$$u = \int_0^x 2t \, \mathrm{d}t = x^2 \qquad x_j = \sqrt{u_j} \, .$$

Eine alternative Methode, Zufallszahlen mit dieser Wahrscheinlichkeitsdichte zu erzeugen, die die Wurzelfunktion vermeidet, ist, zwei Zufallszahlen u_j, $u_{j+1} \in U(0,1)$ zu bilden, und die größere der beiden auszuwählen: $x_j = \max(u_j, u_{j+1})$.

Beispiel 5.2 Erzeugung von Zufallszahlen für eine Exponentialverteilung.

Die normierte Exponentialverteilung für die Variable x ist durch die Wahrscheinlichkeitsdichte $f(x, \lambda) = \lambda \exp(-\lambda x)$ für $x \geq 0$ gegeben, und sie ist null für negative Werte von x

$$u = \int_0^x \lambda \exp\left(-\lambda t\right) \, \mathrm{d}t = 1 - \exp\left(-\lambda x\right) \, .$$

Das Ergebnis ist die Formel $x_j = -\ln\left(1-u_j\right)/\lambda$ oder, weil u_i und $1-u_i$ beide im Intervall (0,1) gleichmäßig verteilt sind, $x_j = -\ln\left(u_j\right)/\lambda$ für eine Folge von exponentiell verteilten Zufallszahlen x_j.
Wenn es sich in einer speziellen Anwendung um sehr große Zufallswerte handelt (zum Beispiel sehr lange Lebenszeiten $t \gg \tau = 1/\lambda$), dann könnte die obige Methode nicht genau genug sein. Sehr große Werte von x werden durch sehr kleine Werte von u erzeugt. Wegen der diskreten Natur von Gleitkommazahlen in einem Rechner werden deshalb sehr große Werte von x diskret sein. Wird ein *kongruenter Generator* mit einem Modul m benutzt, dann ist die kleinste Zahl, die ungleich 0 ist, gegeben durch $1/m$, und der größte transformierte Zufallswert wird $\ln m$ sein, und die nächst größten Werte sind $\ln m - \ln 2$, $\ln m - \ln 3$, usw.

Beispiel 5.3 Zusammengesetzte Ausdrücke.

Man betrachtet als Beispiel den einfachen Potenzausdruck

$$f(x) = \frac{5}{6}\cdot 2x + \frac{1}{6}\cdot 5x^4 \qquad 0 \le x \le 1\,.$$

Man schreibt ihn um zu

$$f(x) = \frac{5}{6}f_1(x) + \frac{1}{6}f_2(x)\,,$$

wobei $f_1(x) = 2x$ und $f_2(x) = 5x^4$ normierte Wahrscheinlichkeitsdichten sind. Die kumulativen Verteilungen für $f_1(x)$ und $f_2(x)$ können invertiert werden, und man hat dann den folgenden Algorithmus für Zufallszahlen für $f(x)$:
Man wählt zwei Zufallszahlen u_1, u_2 und erhält

$$x_i = \begin{cases} \sqrt{u_1} & \text{wenn } u_2 \le 5/6 \\ u_1^{1/5} & \text{wenn } u_2 > 5/6 \end{cases}\,.$$

Verfahren *mit roher Gewalt*. Wenn es keinen einfachen Weg gibt, um nach der analytischen Methode vorzugehen, kann man nach dem folgenden Algorithmus Zufallszahlen entsprechend einer gegebenen Wahrscheinlichkeitsdichte $f(x)$ erzeugen, was jedoch oft nicht sehr effektiv ist. Angenommen, daß die Variable x auf den Bereich $a < x < b$ begrenzt ist. Man bestimmt eine obere Schranke c für $f(x)$ mit $c \ge \max(f(x))$; $\max(f(x))$ ist das Maximum von $f(x)$ im Bereich (a,b). Dann folgt:

1. Man wählt x_i gleichmäßig verteilt zwischen a und b: $x_i = a + u_i\cdot(b-a)$ mit $u_i \in U(0,1)$.
2. Man wählt eine weitere Zufallszahl $u_j \in U(0,1)$.
3. Wenn $f(x_i) < u_j \cdot c$ ist, geht man nach 1., sonst akzeptiert man x_i als Zufallszahl.

Die Effizienz dieses Algorithmus kann erhöht werden, wenn man eine Funktion $s(x)$ finden kann, die die ungefähre Gestalt von $f(x)$ hat und deren Stammfunktion umgekehrt werden kann. Dann wählt man eine Konstante c, so daß für alle x in (a,b) gilt: $c\cdot s(x) > f(x)$. Mit

$$\int_{-\infty}^{x} s(t)\,\mathrm{d}t = S(x) \qquad x_i = S^{(-1)}(u_i)$$

kann man den folgenden Algorithmus anwenden:

1. Man wählt eine Zufallszahl $u_i \in U(0,1)$ und berechnet $x_i = S^{(-1)}(u_i)$.
2. Man wählt eine Zufallszahl $u_j \in U(0,1)$.
3. Wenn $f(x_i) \le u_j \cdot c \cdot s(x_i)$ ist, geht man nach 1., sonst akzeptiert man x_i als Zufallszahl.

Beweis: Der Zufallszahl x_i entspricht eine $s(x)$-Verteilung. Die Wahrscheinlichkeit, daß sie in Schritt 3 akzeptiert wird, ist $f(x_i)/(c\cdot s(x_i))$; Multiplikation der Wahrscheinlichkeiten ergibt $f(x)/c$.

5.5 Zufallszahlen für spezielle Verteilungen

5.5.1 Zufallswinkel und -vektoren

Zwei Dimensionen. Ein Zufallswinkel φ, gleichmäßig verteilt innerhalb von $(0, 2\pi)$, wird durch $\varphi_i = 2\pi u_i$ erzeugt. Oft werden $\sin\varphi$ und $\cos\varphi$ oder ein Zufallseinheitsvektor $(\cos\varphi, \sin\varphi)$ gefordert. Der folgende Algorithmus vermeidet den Gebrauch von trigonometrischen Funktionen:

1. Man erzeugt zwei Zufallszahlen u_1, $u_2 \in U(0,1)$.
2. Man berechnet $v_1 = 2\,u_1 - 1$ und $v_2 = u_2$ (v_1 ist gleichförmig in $(-1,+1)$ und v_2 in $(0,1)$).
3. Man berechnet $r^2 = v_1^2 + v_2^2$. Wenn $r^2 > 1$ ist, geht man nach 1., sonst
4. berechnet man den Sinus und Kosinus des Winkels φ durch

$$\sin\varphi_i = 2\frac{v_1 v_2}{r^2} \qquad \cos\varphi_i = \frac{v_1^2 - v_2^2}{r^2}\,.$$

Die beiden Werte stellen einen Zufallseinheitsvektor in zwei Dimensionen dar. Weder eine trigonometrische Funktion noch die Wurzelfunktion werden benötigt.

Drei Dimensionen. Ein Polarwinkel θ wird zusätzlich zu den Werten von $\sin\varphi$ und $\cos\varphi$ gesucht. Entsprechend dem Raumwinkelelement

$$\mathrm{d}\Omega = \sin\theta\;\mathrm{d}\theta\;\mathrm{d}\varphi \;=\; |\mathrm{d}\cos\theta|\;\mathrm{d}\varphi$$

wird der Polarwinkel durch $\cos\theta_i = v_i = 2u_i - 1$ gebildet. Aus dieser Zufallsvariablen wird der Winkel θ berechnet durch $\theta_i = \arccos(\cos\theta_i)$ oder $\sin\theta_i = \sqrt{1-\cos^2\theta_i} = \sqrt{1-v_i^2}$. Der Zufallseinheitsvektor in drei Dimensionen hat dann die Komponenten $\sin\theta_i\cos\varphi_i$, $\sin\theta_i\sin\varphi_i$ und $\cos\theta_i$.

5.5.2 Normalverteilung

Standardisierte Normalverteilung. Ein einfacher, aber nur angenähert richtiger Generator für Zufallszahlen $z_i \in N(0,1)$, die der standardisierten Gauß-Verteilung $(1/\sqrt{2\pi})\cdot\exp(-x^2/2)$ folgen, basiert auf dem Zentralen Grenzwertsatz:

$$z_i = \sum_{j=1}^{12} u_j - 6\,.$$

Der nächste Algorithmus ist präzise und bedient sich der Methode von Kapitel 5.4: Eine χ^2-verteilte Zufallszahl mit zwei Freiheitsgraden ist leicht zu erzeugen, weil die Dichteverteilung für zwei Freiheitsgrade die Exponentialverteilung ist: $f(\chi^2)\,\mathrm{d}\chi^2 = 1/2\exp(-\chi^2/2)\,\mathrm{d}\chi^2$. Die Zufallszahl $v = -2\ln u$ folgt dieser Verteilung. In der $z_1\,z_2$-Ebene ist v das Quadrat des Abstands vom Ursprung: $v = z_1^2 + z_2^2 = r^2 = (r\cos\varphi)^2 + (r\sin\varphi)^2$. Nimmt man einen gleichverteilten Winkel φ an, so werden zwei unabhängige *standardisierte normalverteilte Zufallszahlen* von v erzeugt durch $z_1 = \sqrt{v}\cos\varphi$ und $z_2 = \sqrt{v}\sin\varphi$. Diesen Algorithmus von Box und Muller kann man durch die vorher beschriebene Methode vereinfachen, die trigonometrische Funktionen vermeidet; dieser vereinfachte Algorithmus lautet wie folgt:

1. Man erzeugt zwei Zufallszahlen u_1, $u_2 \in U(0,1)$.
2. Man berechnet $v_1 = 2u_1 - 1$ und $v_2 = 2u_2 - 1$ (v_1 und v_2 sind gleichförmig verteilt in $(-1,+1)$).
3. Man berechnet $r^2 = v_1^2 + v_2^2$. Wenn $r^2 > 1$, geht man nach 1., sonst

4. berechnet man z_1 und z_2 gemäß

$$z_1 = v_1 \sqrt{\frac{-2 \ln r^2}{r^2}} \qquad z_2 = v_2 \sqrt{\frac{-2 \ln r^2}{r^2}} \,.$$

Die beiden Zahlen z_1 und z_2 sind unabhängig voneinander normal verteilt.

Normalverteilte Zufallszahlen. Eine Zufallszahl x_i aus der Normalverteilung mit Mittelwert μ und Standardabweichung σ

$$N(\mu, \sigma) = \frac{1}{\sqrt{2\pi}\sigma} \cdot \exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right)$$

erhält man aus einer standardisierten, normalverteilten Zufallszahl z_i durch $x_i = \mu + \sigma \cdot z_i$.

Gauß-verteilte Zufallszahlen in n Dimensionen. Die n-dimensionale Normalverteilung wird durch den Vektor $\boldsymbol{\mu}$ von Mittelwerten und die Kovarianz-Matrix $\boldsymbol{V}$ definiert. Die gegebene $n \times n$ Kovarianz-Matrix $\boldsymbol{V}$ kann durch das Produkt $\boldsymbol{R}^T\boldsymbol{R}$ mit einer *rechten Dreiecksmatrix* $\boldsymbol{R}$ dargestellt werden, indem man die Cholesky-Zerlegung (siehe Kapitel 3.6) anwendet. Aus n Zufallszahlen z_i der standardisierten Gauß-Verteilung konstruiert man einen Zufallsvektor $\boldsymbol{z}$ und transformiert in den gewünschten Zufallsvektor $\boldsymbol{x}$ vermöge

$$\boldsymbol{x} = \boldsymbol{\mu} + \boldsymbol{R}^T\boldsymbol{z} \,.$$

Abbildung 5.4 zeigt als Beispiel 3000 Gauß-verteilte positiv korrelierte Zufallszahlen.

Beispiel 5.4 Erzeugung eines zweidimensionalen Vektors für eine Normalverteilung mit vorgegebener Kovarianz-Matrix.

Man nimmt an, daß die Mittelwerte $\boldsymbol{\mu}$ und die Kovarianz-Matrix $\boldsymbol{V}$ gegeben sind durch

$$\boldsymbol{\mu} = \begin{pmatrix} \mu_1 \\ \mu_2 \end{pmatrix} = \begin{pmatrix} 10 \\ 20 \end{pmatrix} \qquad \boldsymbol{V} = \begin{pmatrix} V_{11} & V_{12} \\ V_{21} & V_{22} \end{pmatrix} = \begin{pmatrix} 9 & 2 \\ 2 & 1 \end{pmatrix} .$$

Für eine zweidimensionale Kovarianz-Matrix $\boldsymbol{V}$ mit den Elementen $V_{11} \ldots V_{22}$ lauten die Gleichungen für die Cholesky-Zerlegung $\boldsymbol{R}^T\boldsymbol{R}$ wie folgt (Kapitel 3.6):

$$R_{11} = \frac{V_{11}}{\sqrt{V_{11}}} = \sqrt{V_{11}} \qquad R_{12} = \frac{V_{12}}{\sqrt{V_{11}}} \qquad R_{22} = \sqrt{V_{22} - R_{12}^2}$$

mit dem Ergebnis

$$\boldsymbol{R} = \begin{pmatrix} R_{11} & R_{12} \\ 0 & R_{22} \end{pmatrix} = \begin{pmatrix} 3 & 2/3 \\ 0 & \sqrt{5/9} \end{pmatrix} .$$

Mit zwei unabhängigen Zufallszahlen z_1 und z_2 aus der Standard-Normalverteilung liefert die folgende Gleichung einen Zufallsvektor gemäß der vorgegebenen Kovarianz-Matrix:

$$\begin{aligned} \boldsymbol{x} = \boldsymbol{\mu} + \boldsymbol{R}^T\boldsymbol{z} &= \begin{pmatrix} \mu_1 \\ \mu_2 \end{pmatrix} + \begin{pmatrix} R_{11} & 0 \\ R_{12} & R_{22} \end{pmatrix} \cdot \begin{pmatrix} z_1 \\ z_2 \end{pmatrix} \\ &= \begin{pmatrix} \mu_1 + R_{11}\, z_1 \\ \mu_2 + R_{12}\, z_1 + R_{22}\, z_2 \end{pmatrix} = \begin{pmatrix} 10 + 3\, z_1 \\ 20 + 2/3\, z_1 + \sqrt{5/9}\, z_2 \end{pmatrix} . \end{aligned}$$

Es ist zu beachten, daß dieselbe Zufallszahl z_1 in beiden Komponenten erscheint; dies liefert die Korrelation zwischen den beiden Komponenten, die durch die gegebene Kovarianz-Matrix gefordert wird.

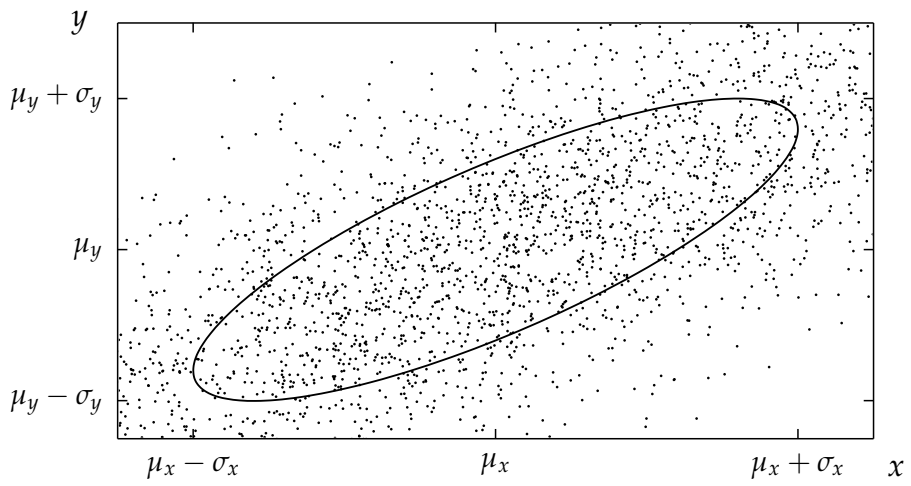


Abbildung 5.4: Stichprobe vom Umfang 3000 aus einer zweidimensionalen Normalverteilung.

5.5.3 Poisson-Verteilung

Die Poisson-Verteilung hat einen einzigen Parameter, den Mittelwert μ. Um Poisson-verteilte Zufallszahlen zu erhalten, erzeugt man unabhängige, exponentiell verteilte Strecken (Wartezeiten zwischen Ereignissen): wenn die Summe der Strecken größer ist als der Mittelwert μ, dann ist die gesuchte Zufallszahl um eins kleiner als die Zahl der Glieder in der Summe. Gleichbedeutend mit der Addition exponentieller Abweichungen ist, die Länge gleichverteilter Strecken zu multiplizieren und sie mit $e^{-\mu}$ zu vergleichen:

1. $x = e^{-\mu}$, $k = 0$, $A = 1$.
2. Man wählt eine Zufallszahl $u_i \in U(0,1)$ und ersetzt A durch $u_i \cdot A$.
3. Wenn $A \geq x$ ist, ersetzt man k durch $k+1$ und geht nach 2. Sonst ist
4. $k =$ Zufallszahl.

Dieser Algorithmus liegt der Funktion `MCPOI` *zugrunde, die im Anhang enthalten ist.*

Für einen großen Wert von μ (etwa $\mu > 10$) kann man eine Näherung durch eine Gauß-Verteilung benutzen:

$$x = \max\left(0, \lfloor \mu + z\sqrt{\mu} + 0.5 \rfloor\right) ,$$

wobei z eine Zufallszahl aus der standardisierten Normalverteilung $N(0,1)$ ist, und $\lfloor x \rfloor$ die größte ganze Zahl $\leq x$ bedeutet.

5.5.4 χ^2-Verteilung

Für eine gerade Zahl n von Freiheitsgraden wird eine Zufalls-χ^2-Variable aus dem Produkt von $n/2$ gleichförmig verteilten Zahlen $u_i \in U(0,1)$ gebildet; für eine ungerade Zahl n von Freiheitsgraden hat man das Quadrat einer Zufallszahl z aus einer standardisierten Normalverteilung zu addieren:

$$x = -2\ln\left(\prod_{j=1}^{n/2} u_j\right) \quad \text{für} \quad \text{gerade } n ,$$

$$x = -2\ln\left(\prod_{j=1}^{(n-1)/2} u_j\right) + z^2 \quad \text{für} \quad \text{ungerade } n .$$

Für große n (etwa $n > 30$) kann man eine Näherung durch eine Gauß-Funktion machen: die Zufallsvariable $y = \sqrt{2\chi^2} - 2\sqrt{2n-1}$ ist angenähert $N(0,1)$-verteilt. Man erzeugt eine Zufallszahl $z \in N(0,1)$ der standardisierten Normalverteilung und berechnet

$$x = \frac{1}{2}\left(z + \sqrt{2n-1}\right)^2$$

(für $z < -\sqrt{2n-1}$ wird die Zahl verworfen und eine neue generiert).

5.5.5 Cauchy-Verteilung

Der folgende Algorithmus kann benutzt werden, um Zufallszahlen gemäß einer Cauchy-Wahrscheinlichkeitsdichte $1/(\pi(1+x^2))$ zu bilden:

1. Man erzeugt zwei Zufallszahlen u_1, $u_2 \in U(0,1)$.
2. Man berechnet $v_1 = 2u_1 - 1$ und $v_2 = 2u_2 - 1$ (v_1 und v_2 sind gleichförmig in $(-1,+1)$).
3. Man berechnet $r^2 = v_1^2 + v_2^2$ und beginnt erneut mit 1., wenn $r^2 > 1$.
4. Man berechnet $x = 0.5\, v_1/v_2$.

Die Zufallsvariable x folgt einer Cauchy-Verteilung mit Mittelwert 0 und einer vollen Breite bei halber Höhe (FWHM) von 1. Durch die Transformation $E = E_0 + x\Gamma$ wird eine Zufallszahl aus der Cauchy-Verteilung mit dem Zentrum bei E_0 und der vorgegebenen Breite Γ gebildet.

Die Routine `MCBIN` *(in Fortran) erzeugt Zufallszahlen k gemäß einer Binomialverteilung. Sie ist im Anhang aufgeführt.*

5.6 Monte Carlo-Integration

5.6.1 Integration in einer Dimension

Die Aufgabe ist, das eindimensionale Integral einer Funktion $g(x)$

$$\mathrm{I} = \int_a^b g(x)\,\mathrm{d}x$$

mit MC-Methoden zu berechnen. Das Integral kann folgendermaßen geschrieben werden:

$$I = \int_a^b 1 \cdot g(x)\,\mathrm{d}x = (b-a)\,E[g(x)]\ ,$$

worin $E[g(x)]$ die Bedeutung des Erwartungswertes von $g(x)$ für eine zwischen den Grenzen a und b gleichförmige Wahrscheinlichkeitsdichte hat. Wenn u_i aus einer gleichförmigen Wahrscheinlichkeitsdichte im Intervall $(0,1)$ gewählt wird, dann ist die Verteilung von $x_i = a + u_i(b-a)$ gleichförmig in (a,b), und das Integral kann angenähert werden mit einer Menge von n Werten x_i durch

$$I \approx I_{MC} = \frac{b-a}{n}\sum_{i=1}^{n} g(x_i). \tag{5.3}$$

Der Fehler σ_I der Schätzung I_{MC} des Integrals hängt von der Varianz von g_i ab:

$$V[I_{MC}] = \sigma_I^2 = V\left[\frac{b-a}{n}\sum_{i=1}^{n} g_i\right] = \left(\frac{b-a}{n}\right)^2 V\left[\sum_{i=1}^{n} g_i\right] = \frac{(b-a)^2}{n}\,V\left[g_i\right]\ , \tag{5.4}$$

entsprechend dem *Zentralen Grenzwertsatz*. Diese Gleichung zeigt, daß die Varianz, d.h. das Quadrat des Fehlers von I_{MC}, mit $1/n$ abnimmt (n = Zahl der Glieder der Summe) und proportional zur Varianz des Integranden $g(x)$ über dem Integrationsintervall ist. Die Varianz von g_i kann durch die Summen von g_i und g_i^2 ausgedrückt werden:

$$V\left[g_i\right] = E\left[\left(g_i - \bar{g}\right)^2\right] \approx \left[\frac{\sum g_i^2}{n} - \left(\frac{\sum g_i}{n}\right)^2\right] .$$

Dies ergibt das endgültige Resultat für die Varianz des Schätzwertes I_{MC} des Integrals:

$$V[I_{MC}] = \sigma_I^2 = \frac{(b-a)^2}{n}\left[\frac{\sum g_i^2}{n} - \left(\frac{\sum g_i}{n}\right)^2\right] = \frac{1}{n}\left[(b-a)^2\,\frac{\sum g_i^2}{n} - I^2\right] . \tag{5.5}$$

5.6.2 Varianzreduzierende Methoden

Im vorhergehenden Abschnitt wurde gezeigt, daß der Fehler bei der Berechnung des Integrals durch die Varianz des Integranden $g(x)$ bestimmt ist. Es ist deshalb nützlich, Methoden zur Reduktion dieser Varianz zu betrachten.

Importance Sampling. Wenn man eine bestimmte Wahrscheinlichkeitsdichte $f(x)$ hat, die im Intervall (a, b) normiert ist, dann kann unter der Bedingung, daß sie integrierbar und in geschlossener Form umkehrbar ist, das Integral der Funktion $g(x)$ folgendermaßen geschrieben werden:

$$I = \int_a^b g(x)\,\mathrm{d}x = \int_a^b \left[\frac{g(x)}{f(x)}\right] f(x)\,\mathrm{d}x = E\left[\frac{g(x)}{f(x)}\right] ,$$

wobei der Erwartungswert von $r(x) = g(x)/f(x)$ für die Wahrscheinlichkeitsdichte $f(x)$ berechnet werden muß. Dieser Erwartungswert wird näherungsweise ermittelt, indem man eine Menge von n x-Werten benutzt, die man der Wahrscheinlichkeitsdichte $f(x)$ entnimmt.

$$I \approx \frac{b-a}{n}\sum_{i=1}^{n}\frac{g(x_i)}{f(x_i)} \quad \text{mit} \quad x_i \in (a,\, b) \quad \text{entsprechend der Dichte } f(x) .$$

Das Verfahren ist gut, wenn man eine einfache Wahrscheinlichkeitsdichte $f(x)$ finden kann, die dem Integranden $g(x)$ ähnlich ist, und die eine einfache Erzeugung von Zufallszahlen erlaubt. Die Varianz von $r(x) = g(x)/f(x)$

$$V\left[r_i\right] = E\left[\left(r_i - (\bar{r})\right)^2\right]$$

sollte klein sein; das ist der Fall, wenn das Verhältnis $r(x)$ nahezu konstant ist. Die Verwendung einer Dichtefunktion $f(x)$ erlaubt auch, Singularitäten zu behandeln (zum Beispiel, wenn $g(x) \to \infty$). Wenn die Integrationsgrenzen a und b die Werte $\pm\infty$ einschließen, muß eine Transformation der Variablen x in eine andere mit endlichen Integrationsgrenzen durchgeführt werden.

Die Anwendung des *importance sampling* kann schwierig werden, wenn $f(x)$ in einem bestimmten x-Bereich sehr klein wird. Ferner wird es schwierig bei mehr als einer Dimension.

Kontroll-Funktion. Eine andere Methode der Varianzreduktion ist das Subtrahieren einer Funktion $f(x)$, die analytisch integriert werden kann und $g(x)$ ähnlich ist:

$$\int_a^b g(x)\,\mathrm{d}x = \int_a^b f(x)\,\mathrm{d}x + \int_a^b \left[g(x) - f(x)\right]\,\mathrm{d}x ,$$

wobei das erste Integral auf der rechten Seite analytisch und das zweite nach MC-Methoden integriert wird. Die Varianz dieses Integrals kann durch geeignete Wahl von $f(x)$ stark reduziert werden.

Partitionierung. Die Varianz kann reduziert werden, indem man das Integrationsintervall in zwei oder mehr Intervalle aufteilt. Die Intervalle sollten so gewählt werden, daß der Integrand $g(x)$ im gegebenen Intervall möglichst wenig variiert.

Als Beispiel betrachtet man das Integral der Funktion $g(x)$

$$I = \int_a^b g\,(x)\,\mathrm{d}x.$$

Wenn man es nach Gleichung (5.3) mit dem Ergebnis $I_1 = I_{MC}$ berechnet, ist seine Varianz

$$V\,[I_1] = \frac{(b-a)^2}{n}\,V\,[g] \approx \frac{(b-a)^2}{n}\;\cdot\;\left[\frac{\sum {g_i}^2}{n} - \left(\frac{\sum g_i}{n}\right)^2\right].$$

Nun wird das Intervall (a,b) in zwei Intervalle gleicher Länge (a,c) und (c,b) geteilt, mit jeweils $(n/2)$ Werten x_i. Das Integral wird dann

$$I_2 = \frac{(c-a)}{n/2}\sum_1 g_i + \frac{(b-c)}{n/2}\sum_2 g_i\,,$$

und seine Varianz ist

$$V\,[I_2] = \frac{(c-a)^2}{n/2}\left[\frac{\sum_1 {g_i}^2}{n/2} - \left(\frac{\sum_1 g_i}{n/2}\right)^2\right] + \frac{(b-c)^2}{n/2}\left[\frac{\sum_2 {g_i}^2}{n/2} - \left(\frac{\sum_2 g_i}{n/2}\right)^2\right].$$

Mit $(c-a) = (b-c) = (b-a)/2$ erhält man

$$V\,[I_2] = \frac{(b-a)^2}{n}\left(\frac{\sum_1 {g_i}^2}{n} + \frac{\sum_2 {g_i}^2}{n} - 2\left[\left(\frac{\sum_1 g_i}{n}\right)^2 + \left(\frac{\sum_2 g_i}{n}\right)^2\right]\right)$$

und

$$\begin{aligned} &V\,[I_1] - V\,[I_2] \\ &= \frac{(b-a)^2}{n}\left(-\frac{(\sum_1 g_i + \sum_2 g_i)^2}{n^2} + 2\left[\frac{(\sum_1 g_i)^2}{n^2} + \frac{(\sum_2 g_i)^2}{n^2}\right]\right) \\ &\qquad = \frac{(b-a)^2}{n}\left(\frac{(\sum_1 g_i - \sum_2 g_i)^2}{n^2}\right) > 0\,, \end{aligned} \tag{5.6}$$

d.h. wenn man das Integral in zwei Abschnitten (I_2) berechnet, erhält man eine kleinere Varianz.

Gewichtung. Angenommen, ein Integral über eine Funktion $g(x)$ ist berechnet worden, und man möchte die Integration für eine modifizierte Funktion $g_m(x)$ wiederholen. Statt die zeitaufwendige MC-Berechnung zu wiederholen, kann man die Werte aus der ersten Integration benutzen und sie mit den Gewichten $w_i = g_m(x_i)/g(x_i)$ versehen.

$$I' = \int g_m(x)\,\mathrm{d}x \approx \frac{b-a}{n}\sum g_m(x_i) = \frac{b-a}{n}\sum g(x_i)\,w_i\,.$$

Dies ist ein gutes Verfahren, vorausgesetzt die Werte $g(x_i)$ stehen noch zur Verfügung.

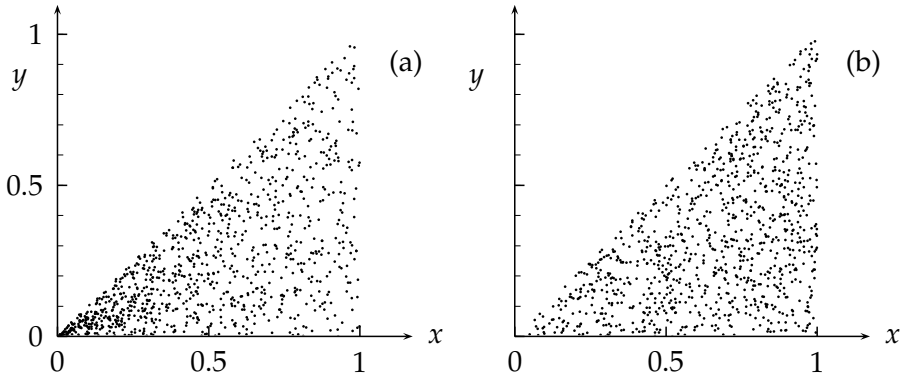


Abbildung 5.5: 1000 Zufallszahlen im Dreieck mit den Eckpunkten $(x, y) = (0, 0)$, $(1, 0)$ und $(1, 1)$. In der Abbildung (a) wird x gleichförmig in $(0, 1)$ und y gleichförmig in $(0, x)$ gewählt; diese Punkte sind nicht gleichmäßig in der Ebene verteilt. In der Abbildung (b) werden zwei Zufallszahlen u_1 und u_2 gleichförmig in $(0, x)$ gewählt und $x = \max(u_1, u_2)$ und $y = \min(u_1, u_2)$ gesetzt; diese Punkte sind gleichmäßig in der Ebene verteilt.

Beispiel 5.5 Ein zweidimensionales Integral.

Es ist das zweidimensionale Integral von $g(x, y)$ in dem zwischen der positiven x-Achse und der Geraden $y = x$ von $x = 0$ bis $x = 1$ gelegenen Integrationsgebiet zu berechnen,

$$I = \int_0^1 \int_0^x g(x, y)\, \mathrm{d}y\, \mathrm{d}x\,,$$

Die Abbildung 5.5 zeigt den dreieckigen Integrationsbereich. Der folgende Algorithmus scheint plausibel:

man wählt x_i, gleichförmig in $(0, 1)$
man wählt y_i, gleichförmig in $(0, x_i)$

und berechnet das Integral durch die Summe

$$I = \frac{1}{n} \sum_{i=1}^{n} g_i\,(x_i, y_i) - \text{das ist aber falsch}.$$

Bei dieser Wahl sind die Punkte im Integrationsbereich *nicht* gleichmäßig verteilt. Dieses Beispiel zeigt, daß es in anscheinend einfachen Problemen durchaus Fallen geben kann. Ein korrekter Algorithmus ist:

$n = 0, I = 0$
(1) $n := n + 1$
man wählt eine erste Zufallszahl $u_1 \in U(0, 1)$
man wählt eine zweite Zufallszahl $u_2 \in U(0, 1)$
$v_1 = \max(u_1, u_2)$
$v_2 = \min(u_1, u_2)$
$I := I + g(v_1, v_2)$
wenn $n < N$ ist, geht man nach (1)
$I := I/N$

5.6.3 Vergleich mit numerischer Integration

Trapezregel. Die numerische Integration in ihrer einfachsten Form wird nach der Trapez-Regel ausgeführt: Das Integral $I = \int_a^b g(x)\,\mathrm{d}x$ wird näherungsweise berechnet durch

$$I_T = h\left(\frac{1}{2}g_0 + g_1 + \cdots + g_{n-1} + \frac{1}{2}g_n\right),$$

wobei das Intervall (a,b) in n gleiche Intervalle der Länge $h = (b-a)/n$ geteilt worden ist, und

$$g_i = g(x_i) \quad x_i = a + i\,h, \quad i = 0,\,1,\ldots n.$$

Um den Fehler von I_T herzuleiten, betrachtet man die Reihenentwicklung in einem Intervall der Breite h

$$\begin{aligned} I &= \int_0^h g(x)\,\mathrm{d}x = \int_0^h (g(0) + x\,g'(0) + \frac{x^2}{2}g''(0) + \cdots)\,\mathrm{d}x \\ &= g(0)\cdot h + \frac{1}{2}\cdot h^2 g'(0) + \frac{1}{6}\cdot h^3 g''(0) + \cdots \\ I_T &= h\,(\frac{1}{2}g(0) + \frac{1}{2}g(h)) \\ &= \frac{1}{2}h\,(g(0) + (g(0) + h\,g'(0) + \frac{h^2}{2}g''(0) + \cdots)). \end{aligned}$$

Die Differenz ergibt eine Abschätzung des Fehlers der Trapezformel zu $I_T - I = (1/12)\,g''(0)\,h^3 + \cdots$. Summiert man die Beiträge der n Intervalle der Länge h des Integrations-Intervalls (a,b), wendet den Zentralen Grenzwertsatz an und setzt $h = (b-a)/n$, dann ist die Standardabweichung von I_T

$$\langle (I_T - I)^2\rangle^{1/2} \approx n\cdot\langle g''\rangle\frac{(b-a)^3}{12n^3} = \langle g''\rangle\cdot\frac{(b-a)^3}{12n^2}.$$

Der Fehler der Trapezformel wird also mit wachsender Zahl n der berechneten Funktionswerte kleiner mit der Proportionalität $1/n^2$.

Simpsonsche Regel. Bei dieser Regel wählt man eine gerade Zahl n, also eine gerade Zahl von Intervallen

$$I_S = \frac{h}{3}\,(g_0 + 4\,g_1 + 2\,g_2 + 4\,g_3 + \cdots + 4\,g_{n-1} + g_n).$$

Sie führt zu einem Fehler, der noch schneller mit der Zahl n der Intervalle abnimmt, nämlich mit $1/n^4$. Im Vergleich dazu fällt der Fehler eines Integrals, das nach MC-Methoden berechnet worden ist, nur mit $1/\sqrt{n}$ ab (siehe Gleichung (5.4)). Darum sind konventionelle Integrations-Methoden in einer Dimension immer besser als MC-Methoden.

Für Integrale in vielen Dimensionen ist die Sache jedoch anders. In einem d-dimensionalen Raum hat man in jeder Dimension $\sqrt[d]{n}$ Intervalle mit n = Anzahl der Punkte, für die der Integrand berechnet werden muß, und der Fehler aus der Trapez-Regel ist dann proportional zu $1/(\sqrt[d]{n})^2 = n^{-2/d}$. Im Gegensatz dazu ist der Fehler bei der MC-Methode immer noch proportional zu $n^{-1/2}$. Die MC-Methode ist also dann besser, wenn gilt

$$-\frac{2}{d} \le -\frac{1}{2} \quad \text{oder} \quad d \ge 4.$$

Zusätzlich hat die MC-Methode die folgenden Vorteile:

- Die Integrationsgrenzen können einfach behandelt werden; das ist wichtig in vieldimensionalen Räumen, wo das Spezifizieren von Integrationsgrenzen analytisch ein Problem werden kann.
- Die Genauigkeit kann kontinuierlich verbessert werden, indem man einfach mehr MC-Terme zur Summe addiert, während viele numerische Integrationsroutinen erfordern, daß die Zahl der Terme nur in Potenzen von 2 anwachsen kann.
- Unabhängige Kontrolle: Selbst wenn es eine analytische oder numerische Lösung gibt, kann die MC-Methode eine Kontrolle bieten, was in komplizierten Fällen nützlich sein kann.

5.6.4 Quasi-Zufallszahlen

In Gleichung (5.6) ist gezeigt worden, daß die Unterteilung des Integrationsintervalls in Teilintervalle die Varianz vermindert und die Genauigkeit der Integration für eine gegebene Gesamtzahl von Stützstellen des Integranden erhöht. Folgt man diesem Vorgehen bis zu seinem logischen Ende, dann führt das zu einer äquidistanten Wahl der Koordinatenwerte im Integrationsbereich. Dies jedoch kann zu starken Quantisierungseffekten an den Grenzen des Gitters führen. Quasi-Zufallszahlen bieten einen Weg aus dieser Schwierigkeit. Sie überdecken den Integrationsbereich quasi gleichmäßig. Ihre Verwendung erfordert allerdings Vorsicht, zum Beispiel ist es wichtig, daß die n Zufallszahlengeneratoren für die n Dimensionen voneinander unabhängig sind. Im untenstehenden Algorithmus wird dies durch die Wahl verschiedener Primzahlen für die verschiedenen Dimensionen erreicht.

Richtmyer-Generator. Dieser Generator erzeugt Quasi-Zufallszahlen. Hierbei sind die Komponenten $x_{i,j}$ des i-ten n-Vektors (Komponenten $j = 1, 2, \ldots n$), mit $0 < x_{i,j} < 1$, gegeben durch

$$x_{i,j} = (i \cdot S_j) \bmod 1\,,$$

wobei $S_j = \sqrt{P_j}$ und P_j die j-te Primzahl ist, und $x \bmod 1 = x - \mathrm{int}(x)$.
Um Rundungsfehler zu vermeiden, sollte $x_{i,j}$ iterativ berechnet werden:
$x_{i+1,j} = (x_{i,j} + (S_j \bmod 1)) \bmod 1$.
Um die Güte eines Quasi-Zufallszahlengenerators zu beurteilen, definiert man eine *Diskrepanz* genannte Größe

$$g_i = \frac{1}{m}\,(v_i - m\,V_i)\,,$$

wobei m die Anzahl der n-dimensionalen Zufallsvektoren $x_{i,j}$, $1 \le i \le m$, ist und $1 \le j \le n$ der Komponentenindex mit $0 \le x_{i,j} \le 1$. Ein bestimmter Zufallsvektor $x_{i,j}$ definiert ein Volumen

$$V_i = \prod_{j=1}^{n} x_{i,j}.$$

Sind die Zufallsvektoren gleichmäßig verteilt, dann erwartet man $m\,V_i$ Vektoren innerhalb des Volumens. Die tatsächliche Zahl der Punkte in diesem Volumen sei v_i. Ein Maß für die Gleichmäßigkeit ist dann durch einen geeigneten Mittelwert über die g_i gegeben, zum Beispiel $\langle {g_n}^2 \rangle^{1/2}$. Dies nennt man Diskrepanz. Sie sollte klein sein und etwa wie $1/m$ abnehmen.

6 Schätzung von Parametern

6.1 Problemstellung und Kriterien

Der wissenschaftlichen Methode liegt die Vorstellung zugrunde, daß Messungen wenigstens im Prinzip wiederholt werden können und dann unter denselben Bedingungen dasselbe Resultat geben. Dies ist natürlich eine Idealisierung. In Wirklichkeit sind Messungen nie ganz exakt, und die Bedingungen bleiben nie exakt gleich. Oft ist es auch unmöglich, die Umstände genau zu kontrollieren, man denke etwa an Effekte, hervorgerufen durch thermische Fluktuationen oder verursacht von Quantenprozessen. Deshalb werden wiederholte Messungen nie genau dieselben Ergebnisse liefern. Diese Differenz zwischen aufeinanderfolgenden Messungen einer Größe und damit auch die Differenz zwischen einer Messung und ihrem wahren Wert kann meistens nicht auf eine einzige Störursache zurückgeführt werden, sondern wird die Folge einer Reihe mehr oder weniger unkontrollierbarer Einflüsse sein, welche man *zufällig* nennt.
Fehler dieser Art nennt man *statistische Fehler.* Die Aufgabe ist nun, aus fehlerbehafteten Messungen die genauest möglichen Ergebnisse zu erarbeiten, zusammen mit Aussagen über ihre Zuverlässigkeit und ihre Grenzen.

Neben statistischen Fehlern gibt es auch noch *systematische Fehler* – dies ist eine wichtige Unterscheidung: Systematische Fehler können zum Beispiel verursacht sein durch Fehler in der Methode, zum Beispiel falsche Meßinstrumente, oder durch falsche Formeln bei der Auswertung. Systematische Fehler müssen anders als statistische Fehler behandelt werden: Sie können nicht durch Mittelung über mehrere Messungen reduziert werden und begrenzen selbst bei sehr kleinen statistischen Fehlern die Genauigkeit des Ergebnisses.

Wegen der unvermeidlichen Meßfehler wird man nach optimalen Verfahren suchen, um ein möglichst genaues Ergebnis aus fehlerbehafteten Messungen zu erzielen und um den Fehler des Endergebnisses zu bestimmen. Dies nennt man Schätzung von Parametern, und es ist ein ganz allgemeines Problem in den quantitativen Wissenschaften. Einfache Beispiele für die Schätzung von Parametern sind die Schätzung des Mittelwertes und der Varianz der zugrundeliegenden Wahrscheinlichkeitsdichte anhand einer Stichproben-Meßreihe (siehe Kapitel 4.7).

Formal möge eine Messung von n unabhängigen Werten $x_1, x_2, \ldots x_n$ (d.h. eine Stichprobe vom Umfang n) der (ein- oder mehrdimensionalen) Zufallsvariablen x vorliegen. Die Aufgabe besteht darin, aus diesen Daten eine beste Schätzung eines oder mehrerer Parameter zu erhalten. Diese besten Schätzungen sind ihrerseits Zufallsvariable, und ebenso wichtig wie die Werte selbst sind ihre Fehler und Korrelationskoeffizienten. Wenn die Meßwerte einer bekannten Wahrscheinlichkeitsdichte $f(x|a)$ folgen, welche von einem oder mehreren Parametern a abhängt, kann das Problem zum Beispiel durch die Maximum-Likelihood-Methode gelöst werden.

Allgemeine Kriterien. Es ist wünschenswert, daß eine Methode zur Bestimmung von Parametern die folgenden vier Kriterien erfüllt, wobei mit $\widehat{a}$ der Schätzwert des Parameters und mit a_0 sein wahrer Wert bezeichnet wird:

1. Konsistenz: $\lim_{n\to\infty} \widehat{a} = a_0$.
2. Erwartungstreue: $E[\widehat{a}] = a_0$, wobei $E[\widehat{a}]$ der Erwartungswert von $\widehat{a}$ ist.
3. Effektivität: die Varianz von $\widehat{a}$ sollte möglichst klein sein.
4. Robustheit (gegenüber falschen Daten oder falschen Voraussetzungen).

Diese Kriterien können oft nicht alle erfüllt werden. Insbesondere die beiden letzten Kriterien sind sehr häufig miteinander im Widerspruch. Zum Beispiel liefert die Methode der Maximum-

Likelihood eine sehr effiziente Schätzung von Parametern, kann aber andererseits sehr abweichende Resultate geben, wenn die angenommene Form der Wahrscheinlichkeitsdichte nicht richtig ist. Es muß deshalb oft ein dem Problem angemessener Kompromiß gewählt werden.

Die beiden am häufigsten benutzten Methoden sind die Maximum-Likelihood- Methode und die Methode der kleinsten Quadrate. Maximum-Likelihood wird in diesem Kapitel besprochen, während den kleinsten Quadraten ein eigenes Kapitel gewidmet ist (Kapitel 7).

6.2 Robuste Schätzung von Mittelwerten

Die Schätzung von Mittelwerten ist ein Standardproblem der Datenauswertung. Ein einfacher Fall ist die Schätzung eines Stichproben-Mittelwertes (siehe Kapitel 4.7).

Beispiel 6.1 Mittelwert einer Verteilung aus einer Stichprobe.

Es soll eine Schätzung des Mittelwertes einer Wahrscheinlichkeitsdichte anhand einer Stichprobe von n unabhängigen Werten x_i vorgenommen werden. Die Wahrscheinlichkeitsdichte selbst muß dabei nicht spezifiziert sein. Die Schätzung des Mittelwertes ist

$$\bar{x} = \frac{1}{n}\sum_i^n x_i \,. \tag{6.1}$$

Diese Schätzung, welche mit der Methode der kleinsten Quadrate abgeleitet werden kann, ist konsistent wegen des Zentralen Grenzwertsatzes, vorausgesetzt, die Varianz der Wahrscheinlichkeitsdichte ist endlich. Sie ist auch erwartungstreu, weil

$$E\left[\bar{x}\right] = \frac{1}{n}\sum E\left[x_i\right] = \langle x \rangle \,.$$

Die Frage nach der Robustheit und Effizienz der Schätzung ist schwieriger zu beantworten.

Mittelwert einer symmetrischen Verteilung. Es wird zunächst der einfachere Fall einer symmetrischen Wahrscheinlichkeitsdichte behandelt. Ihr Zentralwert ist gleich dem Mittelwert. Liegt eine Stichprobe einer symmetrischen unbekannten Verteilung vor, so ist eine offensichtliche Methode zur Schätzung des Zentralwertes der Stichprobenmittelwert $\bar{x}$. Wie sich leicht zeigen läßt, ist dieser Schätzwert konsistent nach dem Gesetz der großen Zahl unter sehr allgemeinen Voraussetzungen (die Varianz der zugrundeliegenden Verteilung muß endlich sein). Wenn die Stichprobe aus einer Gauß-Verteilung stammt, dann folgt Gleichung (6.1) aus der Maximum-Likelihood-Methode (siehe Beispiel 6.6). In diesem Fall ist der Schätzwert auch optimal in dem Sinne, daß die Varianz minimal ist.

Wenn die Verteilung allerdings nicht die Normalverteilung ist, ist der Stichprobenmittelwert *nicht* der effizienteste, zum Beispiel wenn zu einer Normalverteilung noch ein Anteil einer Cauchy- oder Breit-Wigner-Verteilung

$$f(E|\overline{E}) = \frac{A}{(E-\overline{E})^2 + \Gamma^2/4}$$

hinzukommt. Der Stichprobenmittelwert ist dann nicht *robust*.

Ist die Form der Verteilung bekannt, kann der Zentralwert der Verteilung (und damit der Mittelwert und andere Parameter der Verteilung) mit der Maximum-Likelihood-Methode bestimmt werden; das Ergebnis hat dann die asymptotisch kleinstmögliche Varianz. Die Effizienz anderer Schätzwerte ist definiert als das Verhältnis dieser minimalen Varianz zur Varianz des betreffenden Schätzwertes. Zum Beispiel hat der *Median* bei einer Normalverteilung eine Effizienz von 64%.

Ein robuster Schätzwert für den Zentralwert einer symmetrischen Verteilung ist der getrimmte Mittelwert (Mittelwert mit Abschneiden). Hat man insgesamt n Meßpunkte, so bleiben bei dieser Methode die $(1-2r)n/2$ kleinsten und die $(1-2r)n/2$ größten Werte unberücksichtigt, und es wird der Mittelwert der verbleibenden $2rn$ zentralen Einzelwerte gebildet. Für $r = 0.5$ ist dies der Stichproben-Mittelwert, und im Grenzfall $r \to 0$ ist dies der Median. Die Abbildung 6.1 zeigt die asymptotischen Effizienzen als Funktion des Parameters r für drei zum Zentralwert symmetrische Verteilungen; für die Gauß-Verteilung, für die Cauchy-Verteilung und für die doppelt-exponentielle Verteilung[1]. Bei der Wahl von $r = 0.23$ ist für diese drei Verteilungen die Effizienz mindestens 82%. Dieser Wert sollte also für eine robuste Schätzung benutzt werden, um bei Unkenntnis der tatsächlichen Verteilung die *minimal* mögliche Effizienz zu *maximieren*.

Die Methode des getrimmten Mittelwertes ist vielseitig und allgemeiner einsetzbar; zum Beispiel wird sie verwendet, um die Punktzahlen mehrerer Schiedsrichterwertungen zu mitteln (Weglassen der besten und der schlechtesten Bewertung). Die Strategie sollte in allen nicht völlig klaren Fällen mit Monte Carlo-Methoden überprüft werden.

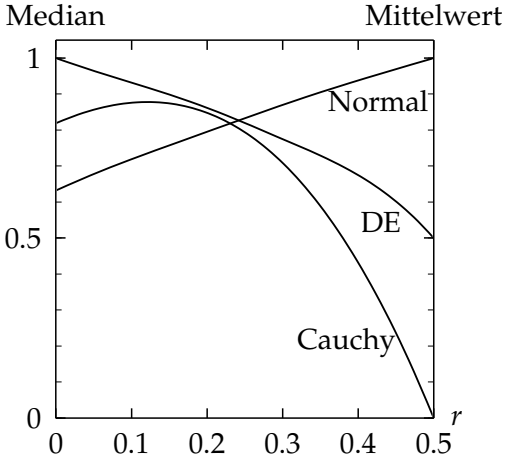


Abbildung 6.1: Aymptotische Effizienz des getrimmten Mittelwertes als Funktion des Parameters r für verschiedene symmetrische Verteilungen: die Normalverteilung, die Cauchy-Verteilung und die doppelt-exponentielle Verteilung (DE).

Mittelwert einer unsymmetrischen Verteilung. Angenommen, es ist der Mittelwert einer unsymmetrischen Verteilung zu bestimmen. Durch die *Unsymmetrie* ist die Benutzung des getrimmten Mittelwertes problematisch. Oft wird eine derartige Unsymmetrie durch die Überlagerung einer Gauß-Funktion mit einem unsymmetrischen Untergrund hervorgerufen.

Eine Möglichkeit ist es, den Verlauf des Signals und des Untergrunds zu parametrisieren und an die Daten nach der Maximum-Likelihood-Methode oder der Methode der kleinsten Quadrate anzupassen. Das Ergebnis wird durch die gewählte Parametrisierung beeinflußt und kann nicht-robust sein, wenn es empfindlich von der genauen Form der Parametrisierung abhängt.

Im Fall von unsymmetrischen Verteilungen ist es oft das beste, eine Variablentransformation vorzunehmen, so daß die neue Variable angenähert Gauß-verteilt ist. Dies zeigt das Beispiel 6.2.

Beispiel 6.2 Mittlerer Impuls eines Teilchenstrahls.

Ein Strahl von Teilchen wird in einer Spurkammer nachgewiesen. Im Magnetfeld der Kammer beschreiben die Teilchen eine gekrümmte Bahn, wobei der Zusammenhang zwischen Krümmungsradius R, Impuls p und Magnetfeld B gegeben ist durch $p = eBR$ (e = Teilchenladung).

[1]Sie besteht aus zwei symmetrisch vom Zentralwert x_0 abfallenden Exponentialverteilungen, also $f(x) = 0.5\exp(x - x_0)$ für $x \leq x_0$ und $f(x) = 0.5\exp(x_0 - x)$ für $x > x_0$.

Aus der Messung der Sagitta $s = l^2/8R$ (l = Spurlänge) wird der Radius R und dann der Impuls p berechnet. Bei diesem Verfahren können sehr große Werte von p auftreten, wenn die Meßgenauigkeit σ etwa gleich der Sagitta s ist: Aufgrund von Meßfehlern kann man sehr kleine Werte von s erhalten, was zu riesigen Werten für p führt. Die gemessene Verteilung von p ist also sehr asymmetrisch. Viel besser ist es deshalb, statt p die Variable $1/p$ zu betrachten, die Gauß-verteilt ist.

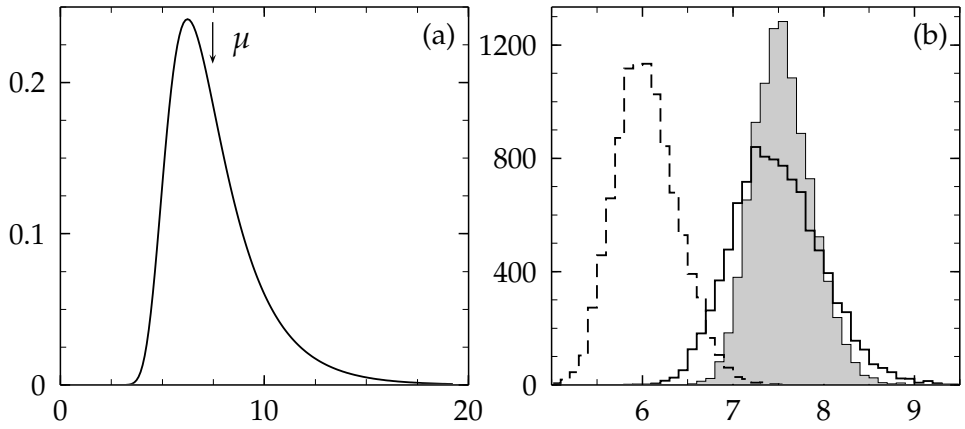


Abbildung 6.2: Die Wahrscheinlichkeitsdichte der verschobenen Moyal-Verteilung mit einem Mittelwert $\mu = 7.5$ ist links (a) gezeigt. Die rechte Abbildung (b) zeigt das Ergebnis von Bestimmungen des Mittelwertes aus 10 000 Stichproben vom Umfang 20. Das graue Histogramm zeigt den Mittelwert aus Maximum-Likelihood-Anpassungen, diese Methode hat die minimale Varianz. Das durchgezogene Histogramm zeigt die Stichprobenmittelwerte; im Mittel wird der Parameter richtig bestimmt, die Effizienz ist jedoch nur 44%. Die Effizienz der Mittelwertbestimmung kann durch Weglassen von jeweils 9 der 20 Einzelwerte der Stichprobe auf 55% verbessert werden, wie das gestrichelte Histogramm zeigt. Dabei wird jedoch der Mittelwert zu kleineren Werten hin verschoben.

Beispiel 6.3 Landau-Verteilung.

Die Landau-Verteilung beschreibt die Verteilung des Energieverlusts durch Ionisation beim Durchgang eines geladenen Teilchens durch eine dünne Materieschicht. Sie hat einen langen Schwanz bei großen Werten des Energieverlusts. Die Form der Landau-Verteilung kann durch die Moyal-Verteilung angenähert werden, die durch

$$f(x)\,\mathrm{d}x = \frac{1}{\sqrt{2\pi}} \exp\left(-\frac{1}{2}\left(x + \exp\left(-x\right)\right)\right) \mathrm{d}x$$

definiert ist; die Moyal-Verteilung hat ihr Maximum bei $x = 0$ und den Mittelwert $\mu = 1.27$. Durch die Transformation $x = 1.27 + \left(E - \bar{E}\right)/b$ wird die Verteilung des Energieverlusts E um den Mittelwert $\bar{E}$ beschrieben. Diese Dichteverteilung ist in Abbildung 6.2(a) für einen Mittelwert von $\bar{E} = 7.5$ gezeigt.

Es werden Stichproben vom Umfang 20 aus dieser Verteilung angenommen, aus denen der Mittelwert des Energieverlusts, also $\bar{E}$ zu bestimmen ist. Das Ergebnis einer Simulation mit 10

000 solcher Stichproben ist in Abbildung 6.2(b) gezeigt. Das graue Histogramm zeigt den Parameter $\bar{E}$ aus Maximum-Likelihood-Anpassungen. Das Histogramm der Stichproben-Mittelwerte (durchgezogenes Histogramm) ist deutlich breiter; diese Breite wird durch die nicht geringe Wahrscheinlichkeit für sehr großen Energieverlust verursacht.

Eine gute Strategie, um solche Einflüsse bei der Berechnung des mittleren Energieverlusts zu vermeiden, besteht wieder darin, einen gewissen Prozentsatz der größten Werte der Stichprobe wegzulassen und dann erst den Mittelwert zu berechnen. Die Simulationsrechnung zeigt, daß man die geringste relative Breite beim Weglassen von neun der 20 Werte erhält; das Ergebnis wird im gestrichelten Histogramm gezeigt. Die systematische Verschiebung von $\bar{E}$ müßte korrigiert werden. Die Simulation zeigt, daß die Effizienz des Mittelwertes von 44% durch Weglassen fast der Hälfte der Daten auf 55% gesteigert werden kann. Das Ergebnis der Maximum-Likelihood-Anpassung ist jedoch deutlich besser.

Beispiel 6.4 Mittelwert einer Gleichverteilung.

Der Mittelwert einer Gleichverteilung (siehe Gleichung (4.17)) kann mit Hilfe von Gleichung (6.1) bestimmt werden. Aber dies ist nicht die effektivste, d.h. die genaueste Schätzung. Ihre asymptotische Effizienz im Sinne der Definition von Seite 126 ist sogar null, da die Varianz der bestmöglichen Schätzung in diesem speziellen Beipiel wie $1/n^2$ gegen null geht und nicht wie $1/n$. Diese beste Schätzung ist gegeben durch $\overline{x} = (\hat{x} + \check{x})/2$, wobei $\hat{x}$ und $\check{x}$ die größten bzw. kleinsten Werte der Stichprobe sind. Die Varianz dieser Schätzung ist minimal.

6.3 Die Maximum-Likelihood-Methode

6.3.1 Prinzip der Maximum-Likelihood

Es mögen n Messungen der Zufallsvariablen x vorliegen, wobei x eine einzelne Variable oder ein Vektor von Variablen sein kann. Die den Meßwerten $x_1, x_2, \ldots x_n$ zugrundeliegende Wahrscheinlichkeitsdichte $f(x|a)$ soll bekannt sein, und a steht für einen oder mehrere unbekannte Parameter, von denen die Wahrscheinlichkeitsdichte abhängt. Die Aufgabe besteht darin, die beste Schätzung $\widehat{a}$ des Parameters aus den vorliegenden Meßdaten zu gewinnen.

Die *Maximum-Likelihood-Methode* geht von der ein- oder mehrdimensionalen Wahrscheinlichkeitsdichte $f(x|a)$ der gemessenen Werte aus und bildet die **Likelihood-Funktion**

$$L(a) = f(x_1|a) \cdot f(x_2|a) \cdots f(x_n|a) = \prod_{i=1}^{n} f(x_i|a)\,.$$

Die Funktion $L(a)$ ist für eine gegebene Stichprobe x_i eine Funktion der Parameter a und gibt sozusagen die Wahrscheinlichkeit an, bei einer vorgegebenen Wahl der Parameter a gerade diese Meßwerte zu erhalten. Sie ist aber *nicht* eine Wahrscheinlichkeitsdichte in den Parametern a. Gemäß dem Maximum-Likelihood-Prinzip ist die beste Schätzung von a derjenige Wert $\widehat{a}$, welcher $L(a)$ zu einem Maximum macht, d.h. welcher die Wahrscheinlichkeit maximiert, gerade den beobachteten Satz von x_i-Werten zu erhalten. Die formale Bedingung ist deshalb

$$L(a) = \text{Maximum}\,.$$

Es ist wichtig, darauf zu achten, daß $f(x|a)$ *für alle Werte von a auf eins normiert ist,*

$$\int_\Omega f(\boldsymbol{x}|a)\,\mathrm{d}\boldsymbol{x} = 1 \text{ für alle } a.$$

Diese Normierung muß während aller Schritte aufrecht erhalten werden, bei denen a variiert wird, um das Minimum von $L(a)$ zu finden. Dies bedingt einen großen Teil des numerischen Aufwands der Methode.

Das Maximum wird wie üblich durch Differenzieren gewonnen.

$$\frac{\mathrm{d}L(a)}{\mathrm{d}a} = 0 \qquad \text{oder} \quad \frac{\partial L}{\partial a_k} = 0 \quad \text{für alle } k\,.$$

In der Praxis arbeitet man mit dem Logarithmus der Likelihood-Funktion $l(a) = \ln L(a)$ – sie heißt Log-Likelihood-Funktion. Da der Logarithmus eine monotone Funktion ist, hat er sein Maximum an derselben Stelle, und die Bedingung für das Maximum wird

$$l(a) = \ln\, L(a) = \sum_{i=1}^{n} \ln\, f(x_i|a) = \text{Maximum} \quad \text{oder} \quad \frac{\mathrm{d}l(a)}{\mathrm{d}a} = 0 \tag{6.2}$$

für den Fall eines einzigen Parameters. Für einen Vektor $\boldsymbol{a}$ von m Parametern a_i, $i = 1, 2, \ldots m$ hat man

$$\frac{\partial l(\boldsymbol{a})}{\partial a_k} = 0 \quad \text{für} \quad k = 1, 2, \ldots m\,.$$

Wenn das Maximum am Rande des Parameterraumes von a liegt, müssen spezielle Verfahren zur Anwendung kommen (siehe Kapitel 8.6).

In diesem Buch wird aus Konsistenzgründen fortan die *negative* Log-Likelihood-Funktion

$$F(a) = -l(a) = -\sum_{i=1}^{n} \ln\, f(x_i|a) \tag{6.3}$$

benutzt. In diesem Fall muß man natürlich $F(a)$ *minimieren*.[2] Die Maximum-Likelihood-Methode ist in aller Regel *konsistent*, aber nicht notwendigerweise erwartungstreu (siehe Kapitel 6.6). In der Grenze $n \to \infty$ kann die Konsistenz bewiesen werden, und in diesem Fall ist die Methode auch erwartungstreu. Für diesen Grenzfall kann auch bewiesen werden, daß die Methode *effizient* ist. Das sieht eigentlich ganz gut aus. Die Methode hat aber auch ihre Probleme. In aller Regel erfordert ihre Anwendung einen großen Rechenaufwand. Mit modernen Rechnern sollte dies aber kein ernsthaftes Problem mehr sein. Jedoch, was wichtiger ist, erfordert sie weiterhin die a priori Kenntnis der Form der Wahrscheinlichkeitsdichte $f(x|a)$. Wenn diese falsch angenommen wird, so wird auch das Ergebnis falsch. Eine wichtige Kontrolle ist in diesem Zusammenhang, ob die Wahrscheinlichkeitsdichte mit den besten Werten der Parameter auch tatsächlich die Daten richtig wiedergibt. Eine solche Kontrolle erfordert meist einen zusätzlichen Aufwand, der jedoch geleistet werden sollte. Man sollte sich vergewissern, daß die Daten mit den Werten der Parameter verträglich sind, und zwar *überall*. Im Fall mehrdimensionaler Verteilungen muß man den Raum der Daten in n-dimensionale Intervalle teilen und die Prüfung in allen Intervallen durchführen.

Beispiel 6.5 Zerfallswinkelverteilung eines Elementarteilchens.

Die Zerfallswinkelverteilung eines bestimmten Teilchens möge durch die folgende Wahrscheinlichkeitsdichte gegeben sein

$$f(x|a) = \frac{1}{2}\,(1 + ax) \qquad \text{mit} \quad x = \cos\vartheta\,.$$

Angenommen, n Werte $x_i = \cos\vartheta_i$ (ϑ_i = Zerfallswinkel) sind gemessen worden. Die Aufgabe besteht darin, die beste Schätzung $\widehat{a}$ des Zerfallsparameters a zu finden, welcher das Teilchen

[2] Manchmal wird auch noch ein Faktor 2 in die Definition eingeschlossen, um den Ausdruck dem einer Summe kleinster Quadrate ähnlich zu machen (siehe Gleichung (6.8)).

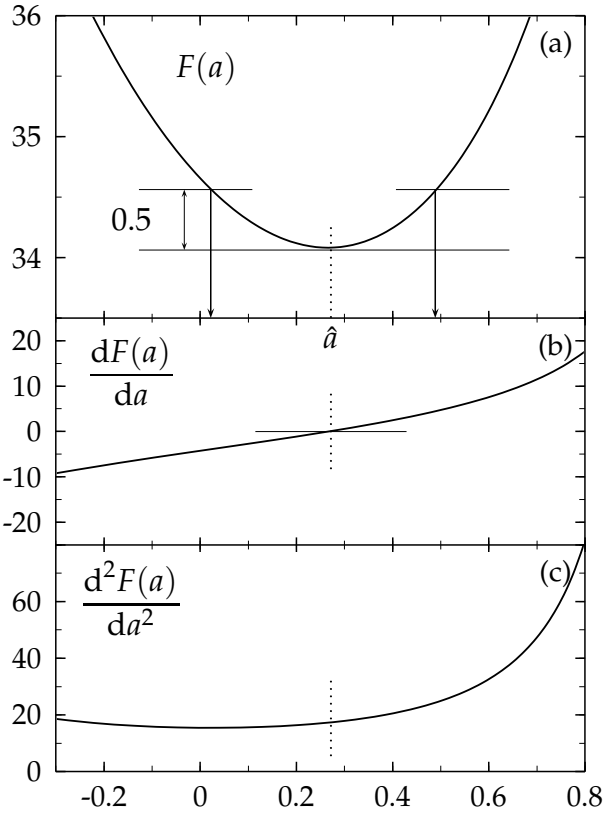


Abbildung 6.3: Die negative Log-Likelihood-Funktion $F(a)$ für die Winkelverteilung (a) als Funktion des Parameters a. Der beste Schätzwert $\hat{a}$ ist durch das Minimum der Funktion $F_{\min}$ definiert. Die Werte des Parameters a mit $F(a) = F_{\min} + 0.5$ definieren die 1σ-Grenzen (Pfeile). Die erste Ableitung $\mathrm{d}F/\mathrm{d}a$ in Abbildung (b) ist angenähert linear. Die zweite Ableitung $\mathrm{d}^2F/\mathrm{d}a^2$ in Abbildung (c) ist im Bereich innerhalb $\pm 1\sigma$ angenähert konstant, insbesondere bei größeren Werten gibt es jedoch deutliche Abweichungen.

charakterisiert. Die Wahrscheinlichkeitsdichte ist für Werte $-1 \leq x \leq +1$ bereits richtig normiert, und kann so direkt in der negativen Log-Likelihood-Funktion verwendet werden

$$F(a) = -\sum_{i=1}^{n} \ln \frac{1}{2}\left(1 + a x_i\right) \ .$$

In diesem Fall kann das Minimum von $F(a)$ direkt dem Verlauf von $F(a)$ gegen a entnommen werden. Abbildung 6.3 zeigt als Beispiel die Funktion $F(a)$ für eine Stichprobe von *Meßwerten*, die durch eine Monte Carlo-Simulation mit dem wahren Wert des Parameters $a_0 = 0.25$ gewonnen wurden. Außerdem sind die erste und zweite Ableitung nach a gezeigt. Die zweite Ableitung ist nahezu konstant, d.h. die Kurve ist näherungsweise eine Parabel und die zugehörige Likelihood-Funktion ist näherungsweise gaußisch.

Beispiel 6.6 Mittelwert einer Gauß-Verteilung.

In diesem Beispiel wird angenommem, daß die Meßwerte x_i bekannte Werte ihrer Standardabweichung σ_i haben, die im übrigen alle verschieden sein können. Die Gauß-Wahrscheinlichkeitsdichte ist

$$f(x_i|a) = \frac{1}{\sqrt{2\pi}\sigma_i} \cdot \exp\left[-\frac{(x_i - a)^2}{2\sigma_i^2}\right] .$$

In diesem Fall bedeutet der Parameter a den Mittelwert. Die negative Log-Likelihood-Funktion ist

$$F(a) = \text{konst} + \frac{1}{2}\sum_{i=1}^{n} \frac{(x_i - a)^2}{\sigma_i^2} . \tag{6.4}$$

Seine beste Schätzung gemäß dem Maximum-Likelihood-Prinzip ergibt sich aus der Bedingung

$$\frac{\mathrm{d}F(a)}{\mathrm{d}a} = -\sum_{i=1}^{n} \frac{x_i - a}{\sigma_i^2} = 0\,.$$

Man führt die Gewichte $w_i = 1/\sigma_i^2$ ein, und erhält das Resultat

$$\widehat{a} = \frac{\sum x_i\, w_i}{\sum w_i}\,. \tag{6.5}$$

Die beste Schätzung des Mittelwertes $\widehat{a}$ ist also das gewichtete Mittel aller Werte x_i.
Die beste Schätzung von a ist ihrerseits eine Zufallsvariable. Wie genau ist die Schätzung? Nach Gleichung (6.11) ist ihre Standardabweichung

$$\sigma(\widehat{a}) = \left(\left. \frac{\mathrm{d}^2 F}{\mathrm{d}a^2} \right|_{\widehat{a}} \right)^{-1/2}\,.$$

Für das Beispiel erhält man

$$\sigma(\widehat{a}) = \frac{1}{\sqrt{\sum w_i}} = \frac{1}{\sqrt{\sum (1/\sigma_i^2)}}\,. \tag{6.6}$$

Wenn alle Gewichte gleich sind, also $\sigma_i = \sigma$, dann gilt für den Mittelwert und für den Fehler des Mittelwertes

$$\bar{x} = \widehat{a} = \frac{\sum_{i=1}^{n} x_i}{n}\,, \qquad \sigma(\widehat{a}) = \frac{\sigma}{\sqrt{n}}\,. \tag{6.7}$$

Dies sind die Formeln in Kapitel 4.7.

6.3.2 Methode der kleinsten Quadrate

Wenn den Daten eine Gauß-Verteilung zugrundeliegt, erhält man die folgende Verbindung zwischen der Maximum-Likelihood-Methode und der Methode der kleinsten Quadrate: Die negative Log-Likelihood-Funktion ist in diesem Fall gegeben durch Gleichung (6.4). Man muß nunmehr ein *Minimum* dieser Funktion suchen. Das Minimum von $F(a)$ ist gleichbedeutend mit dem Minimum der Summe

$$S(a) = \sum_{i=1}^{n} \frac{(x_i - a)^2}{\sigma_i^2}\,, \tag{6.8}$$

wobei die folgende wichtige Beziehung besteht zwischen der negativen Log-Likelihood-Funktion und der Summe S der Abweichungsquadrate der Methode der kleinsten Quadrate:

$$F(a) = \text{konst} + \frac{1}{2} S(a)\,. \tag{6.9}$$

Oft besteht das Problem darin, daß an einer Reihe von Stützstellen z_i Werte für $y_i(z_i)$ vorgegeben sind, und der Parameter a bedeutet in diesem Fall den theoretisch vorausgesagten Mittelwert $y_t(z_i|a')$ für y an der Stelle z_i, der von dem Parameter a' abhängt. Mit den Ersetzungen $x_i \to y(z_i)$ und $a \to y_t(z_i|a')$ erhält man die Standardformulierung des Prinzips der kleinsten Quadrate,

$$S(a) = \sum_{i=1}^{n} \frac{(y_i(z_i) - y_t(z_i|a))^2}{\sigma_i^2} = \text{Minimum}, \qquad \frac{\mathrm{d}S}{\mathrm{d}a} = 0\,, \tag{6.10}$$

wobei σ_i den statistischen Fehler der Messungen bedeutet.

Die Methode kann auf mehrere Parameter $a_1, a_2, \ldots a_N$ erweitert werden. Die Bedingung für das Minimum ist dann gegeben durch die Gleichungen

$$\frac{\partial S}{\partial a_k} = 0 \quad \text{für} \quad k = 1, 2, \ldots N\,.$$

Die Methode der kleinsten Quadrate erscheint hier als ein Sonderfall der Maximum-Likelihood-Methode im Falle von Gauß-verteilten Meßgrößen. Man kann jedoch die Methode der kleinsten Quadrate aus einem allgemeinen Prinzip herleiten ohne diese spezielle Annahme über die Verteilung. Die Methode wird im Kapitel 7 ausführlich dargestellt.

6.4 Fehlerberechnung bei der Maximum-Likelihood-Methode

Ein Parameter. Der Fall eines einzigen Parameters ist relativ einfach. Am besten zeichnet man die negative Log-Likelihood-Funktion $F(a)$ als Funktion von a auf. Man definiert dann analog zur Gauß-Verteilung Fehlergrenzen entsprechend 1σ, 2σ, 3σ durch die Forderung, daß die Funktion um $1/2$, 2, $9/2$ zugenommen hat, verglichen mit dem Minimalwert bei $\widehat{a}$. Dieses intuitive Verfahren hat seine Rechtfertigung darin, daß die Likelihood-Funktion häufig näherungsweise Gauß-verteilt ist. Die Fehlergrenzen können allerdings unsymmetrisch um $\widehat{a}$ liegen, und dann ist die Schätzung nicht erwartungstreu.

Im asymptotischen Fall $n \to \infty$ nähert sich die Likelihood-Funktion einer Gauß-Funktion, und die Varianz $V[\widehat{a}] \to 0$. Die negative Log-Likelihood-Funktion $F(a)$ kann dann um ihren Minimalwert entwickelt werden, wobei man beachtet, daß $\mathrm{d}F(a)/\mathrm{d}a = 0$ ist für $a = \widehat{a}$ nach Definition.

Dann gilt

$$F(a) = F(\widehat{a}) + \frac{1}{2} \cdot \frac{\mathrm{d}^2 F}{\mathrm{d}a^2} \cdot (a - \widehat{a})^2 + \ldots$$

oder

$$L(a) \cong \text{konst} \cdot \exp\left(-\frac{1}{2} \cdot \frac{\mathrm{d}^2 F}{\mathrm{d}a^2}(a - \widehat{a})^2\right) = \text{konst} \cdot \exp\left(-\frac{(a - \widehat{a})^2}{2\sigma^2}\right).$$

Die Likelihood-Funktion hat in dieser Grenze in der Nähe des Maximums die Form einer Gauß-Funktion, und durch Vergleich der Exponenten folgt

$$\sigma(\widehat{a}) = \left(\left.\frac{\mathrm{d}^2 F}{\mathrm{d}a^2}\right|_{\widehat{a}}\right)^{-1/2} . \tag{6.11}$$

Die negative Log-Likelihood-Funktion hat die Form einer Parabel und ihre zweite Ableitung ist eine Konstante. Der Wert von $F(a)$ um das Minimum bei $a = \widehat{a} \pm r \cdot \sigma$ ist

$$F(\widehat{a} \pm r \cdot \sigma) = F(\widehat{a}) + \frac{1}{2} r^2 \ . \tag{6.12}$$

Die Gesamtwahrscheinlichkeit, die in den Konfidenzintervallen $\widehat{a} \pm \sigma$ oder $\widehat{a} \pm r \cdot \sigma$ enthalten ist, kann aus der Gauß-Verteilung berechnet werden. Wenn die tatsächliche Log-Likelihood-Funktion ein parabolisches Verhalten zeigt, kann die Standardabweichung aus Gleichung (6.11) bestimmt werden. Wenn die Log-Likelihood-Funktion aber von einem parabolischen Verhalten

abweicht, muß man eine nichtlineare Transformation der Variablen a in eine Variable $z = z(a)$ suchen, so daß $F(z)$ ein parabolisches Verhalten hat, und dann kann wie oben verfahren werden, um die Standardabweichung σ_z von z zu bestimmen. Abbildung 6.5 zeigt dies an einem Beispiel. Wegen der Invarianzeigenschaft der Maximum-Likelihood-Schätzungen ist die beste Schätzung $\widehat{z} = z(\widehat{a})$, und die Identitäten

$$\begin{aligned} F(\widehat{z}+\sigma_z) &= F(\widehat{z}) + \frac{1}{2} = F(\widehat{a}) + \frac{1}{2} = F(\widehat{a}+\sigma_r) \\ F(\widehat{z}-\sigma_z) &= F(\widehat{z}) + \frac{1}{2} = F(\widehat{a}) + \frac{1}{2} = F(\widehat{a}-\sigma_l) \end{aligned} \tag{6.13}$$

gestatten die Bestimmung der *rechtsseitigen* und *linksseitigen* Standardabweichung σ_r und σ_l für den Parameter a. Man kann Aussagen über a machen, ohne tatsächlich die Transformation durchzuführen, und die rechtsseitige und linksseitige Standardabweichung für a bestimmen, indem man an die Stelle geht, an welcher der Wert $F(\hat{a})$ des Minimums um 1/2 zugenommen hat (siehe Abbildung 6.4). Oft hat man das Problem, Messungen mit asymmetrischen Fehlergrenzen zu kombinieren. Dazu gibt es Rezepte [61].
Sind die Fehlergrenzen nicht symmetrisch, so ist im übrigen nicht mehr garantiert, dass jeweils 16% ausserhalb der σ_r- bzw. der σ_l- Fehlergrenzen liegen [62].

Das Resultat einer Likelihood-Anpassung wird üblicherweise in der Form $x \pm \sigma$ oder $x^{+\sigma_r}_{-\sigma_l}$ angegeben (eine Standardabweichung). Intervalle, die eine größere Wahrscheinlichkeit einschließen, können ebenfalls mit dem entsprechenden Verfahren bestimmt werden mit einer Funktionsdifferenz von $r^2/2$ bei r Standardabweichungen.

Mehrere Parameter $a_1, a_2, \ldots a_N$**:** Hier ist die Likelihood-Funktion

$$L(a_1, a_2, \ldots a_N) = \prod_{i=1}^{n} f(x_i, a_1, a_2, \ldots a_N)\,.$$

Entwickelt man die negative Log-Likelihood-Funktion um ihr Minimum bei $\widehat{a}$, so erhält man, da bei $a = \widehat{a}$ die ersten Ableitungen verschwinden,

$$\begin{aligned} &F(a_1, a_2, \ldots a_N) \\ &= F(\widehat{a}_1, \widehat{a}_2, \ldots \widehat{a}_N) + \frac{1}{2}\sum_{i,k} \frac{\partial^2 F}{\partial a_i \cdot \partial a_k}(a_i - \widehat{a}_i)(a_k - \widehat{a}_k) + \ldots \\ &= F(\widehat{a}_1, \widehat{a}_2, \ldots, \widehat{a}_N) + \frac{1}{2}\sum_{i,k} G_{ik}(a_i - \widehat{a}_i)(a_k - \widehat{a}_k) + \ldots\ . \end{aligned}$$

Asymptotisch nähert sich die Likelihood-Funktion $L(\boldsymbol{a}) = \exp(-F(\boldsymbol{a}))$ einer Gauß-Wahrscheinlichkeitsdichte für die Variablen $\widehat{a}_i$ an, und nach Gleichung (4.44) ist die Kovarianzmatrix $\boldsymbol{V}$ des Vektors $\boldsymbol{a}$ gegeben durch

$$\boldsymbol{V} = \boldsymbol{G}^{-1} \qquad \text{mit} \qquad G_{ik} = \frac{\partial^2 F}{\partial a_i \partial a_k}\,, \tag{6.14}$$

genommen am Minimum $\widehat{a}$. Diese anschauliche Ableitung kann im asymptotischen Fall richtig begründet werden (siehe Kapitel 6.6.) Es ist wichtig, zu beachten, daß die Matrix $\boldsymbol{G}$ die Form der Hesse-Matrix $\boldsymbol{H}$ in Kapitel 8.1.2 hat. Bei Methoden, welche zur Optimierung die Hesse-Matrix $\boldsymbol{H} = \boldsymbol{G}$ und ihre inverse Matrix benutzen, ist die Kovarianzmatrix bereits bekannt. Im nicht- asymptotischen Fall ist die inverse Matrix der Hesse-Matrix im Minimum eine Näherung für die Kovarianzmatrix.

Im Fall von zwei Parametern zeichnet man Konturlinien als Linien gleicher Likelihood $F(\boldsymbol{a}) = F(\widehat{\boldsymbol{a}}) + 1/2\ r^2$. Abweichungen vom asymptotischen Verhalten können näherungsweise durch *asymmetrische* Fehler beschrieben werden, analog zum Fall mit einer Variablen. Formal ist eine Funktion $F_{\min}(a_i)$ des i-ten Parameters von $\boldsymbol{a}$ definiert durch

$$F_{\min}(a_i) = \min F(\boldsymbol{a}) \ ,$$

welche das *Minimum* von $F(\boldsymbol{a})$ bezüglich *aller anderen* Parameter a_i ist. Dann werden durch

$$\begin{aligned} F_{\min}(\widehat{a}_i + \sigma_r) &= F(\widehat{\boldsymbol{a}}) + \frac{1}{2} \\ F_{\min}(\widehat{a}_i - \sigma_l) &= F(\widehat{\boldsymbol{a}}) + \frac{1}{2} \end{aligned}$$

die Standardabweichungen wie im Fall einer Variablen definiert. Diese Methode ist im Programm MINUIT [37] unter dem Namen MINOS implementiert.

6.5 Anwendungen der Maximum-Likelihood-Methode

6.5.1 Beispiele

Beispiel 6.7 Mittelwert der Poisson-Verteilung.

Eine Schätzung $\widehat{\mu}$ des Parameters μ = Mittelwert einer Poisson-Verteilung soll anhand von n Einzelbeobachtungen erfolgen, die in der Form von ganzen Zahlen $x_1, x_2, \ldots x_n$ vorliegen. Die Poisson-Wahrscheinlichkeit mit Mittelwert μ für das Auftreten der Anzahl x_j ist

$$P(x_j) = \frac{\mu^{x_j}}{x_j!} e^{-\mu} \ .$$

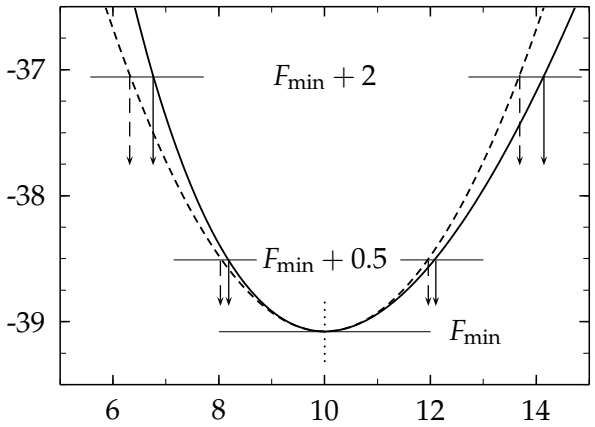


Abbildung 6.4: Die negative Log-Likelihood-Funktion $F(\mu)$ für die Bestimmung des Parameters μ der Poisson-Verteilung aus drei Einzelwerten x_j mit einer Summe von $\sum_j x_j = 30$. Zum Vergleich ist auch die parabolische Näherung (gestrichelt) gezeigt.

Die negative Log-Likelihood-Funktion ist

$$F(\mu) = -\sum_{j=1}^{n} \ln P(x_j) = -\sum_{j=1}^{n} \left[x_j \ln \mu - \mu - \ln x_j! \right] = -\ln \mu \cdot \sum_{j=1}^{n} x_j + n\mu + \text{konst}\,. \tag{6.15}$$

Die Terme $\ln x_j!$ hängen nicht von μ ab und sind deshalb Konstante bezüglich μ. Die ersten und zweiten Ableitungen von $F(\mu)$ nach μ sind also

$$\frac{\mathrm{d}F}{\mathrm{d}\mu} = n - \frac{1}{\mu} \sum_{j=1}^{n} x_j \qquad \frac{\mathrm{d}^2F}{\mathrm{d}\mu^2} = \frac{1}{\mu^2} \sum_{j=1}^{n} x_j \,.$$

Die Maximum-Likelihood-Schätzung $\widehat{\mu}$ folgt aus der Forderung, daß die erste Ableitung null ist. Die Schätzung der Varianz ist gegeben durch Gleichung (6.11), und so folgt

$$\widehat{\mu} = \frac{1}{n}\sum_{j=1}^{n} x_j \qquad\qquad V[\widehat{\mu}] = \frac{\widehat{\mu}}{n}\,.$$

Die beste Schätzung des Mittelwertes der Poisson-Verteilung ist also der arithmetische Mittelwert der Einzelbeobachtungen – ein einleuchtendes Ergebnis. Abbildung 6.4 zeigt die negative Log-Likelihood-Funktion an einem Beispiel. Eine weitere nützliche Information zur Poisson-Verteilung ist Tabelle 6.1.

Beispiel 6.8 Mittelwert und Varianz der Binomialverteilung.

Ein Einzelereignis habe die Wahrscheinlichkeit p. Bei n Versuchen möge man es insgesamt r mal erhalten. Die Wahrscheinlichkeit dafür ist die Binomialwahrscheinlichkeit

$$P(r) = p^r(1-p)^{(n-r)}\binom{n}{r}\,.$$

Die negative Log-Likelihood-Funktion ist

$$F(p) = -r\ln p - (n-r)\ln(1-p) - \ln\binom{n}{r}\,.$$

Die Schätzung $\widehat{p}$ für die Wahrscheinlichkeit p erhält man aus

$$\frac{\mathrm{d}F}{\mathrm{d}p} = -\frac{r}{p} + \frac{n-r}{1-p} = 0\,.$$

Das Resultat für die Schätzung von p ist einleuchtend,

$$\widehat{p} = \frac{r}{n}\,.$$

Die Varianz der Schätzung ist mit $r = n\widehat{p}$

$$V\,[\widehat{p}] = \sigma(\widehat{p})^2 = \left(\frac{d^2F}{dp^2}\right)^{-1} = \frac{\widehat{p}(1-\widehat{p})}{n}\,.$$

Beispiel 6.9 Gleichverteilung.

Die Wahrscheinlichkeitsdichte der Gleichverteilung ist

$$f(x) = \frac{1}{b-a} \qquad a < x < b\,.$$

Sie enthält zwei Parameter a und b, den Anfangs- und den Endpunkt. Um diese Parameter aus einer Stichprobe von beobachteten Werten $x_j, j = 1, 2, \ldots n$ zu schätzen, bildet man die negative Log-Likelihood-Funktion

$$F(a,b) = -\ln\left[\prod_{j=1}^{n}\frac{1}{b-a}\right] = n\cdot\ln(b-a)\,.$$

Hierbei hängt $F(a,b)$ nicht von den Beobachtungswerten ab (!). Die Ableitung nach a oder b führt zu nichts Gescheitem. Jedoch sieht man der Funktion $F(a)$ unmittelbar an, daß sie ein Minimum für das kleinste Intervall $a\ldots b$ hat. Deshalb sind die Maximum-Likelihood-Schätzungen von a und b die kleinsten und größten Werte der Stichprobe

$$\widehat{a} = \min_{j=1}^{n}(x_j) \qquad\qquad \widehat{b} = \max_{j=1}^{n}(x_j)\,.$$

Beispiel 6.10 Akzeptanz eines Experiments.

Dieses Beispiel aus der Praxis der Hochenergiephysik behandelt die Bestimmung der Lebensdauer eines Elementarteilchens aus seiner Zerfallslänge, d.h. aus dem bis zu seinem Zerfall zurückgelegten Weg. Um konkret zu sein, betrachtet man den Zerfall des K^0_S Mesons in $\pi^+ + \pi^-$ in einem Detektor, welcher die Zerfallslänge ℓ im Intervall

$$\ell_1 = 1 \text{ cm} \quad \text{bis} \qquad \ell_2 = 10 \text{ cm}$$

zu messen gestattet; dies ist der Akzeptanzbereich. Die Wahrscheinlichkeit einer Beobachtung außerhalb dieses Intervalls ist null, und dies muß bei der Aufstellung der Likelihood-Funktion berücksichtigt werden. Nach dem exponentiellen Zerfallsgesetz ist die Wahrscheinlichkeitsdichte der Zerfallszeiten $f(t)$ proportional zu $e^{-t/\tau}$, wobei τ die mittlere Lebensdauer ist. Jede Messung, welche nur im Zeitintervall $t_1 \ldots t_2$ sensitiv ist, muß jeweils richtig normiert sein gemäß

$$\int_{t_1}^{t_2} f(t)\,\mathrm{d}t = 1\,,$$

und damit wird die Zerfallszeitverteilung

$$f(t|\tau)\,\mathrm{d}t = \frac{e^{-t/\tau}}{\tau\left(e^{-t_1/\tau} - e^{-t_2/\tau}\right)}\,\mathrm{d}t\,,$$

gültig im Ruhesystem des Teilchens (die mittlere Lebensdauer ist im Ruhesystem definiert). Die Wahrscheinlichkeitsdichte muß als Funktion der Zerfallslänge ℓ ausgedrückt werden, welche im Bereich $\ell_1 \ldots \ell_2$ gemessen werden kann. Der Zusammenhang zwischen der Zerfallszeit t (im Ruhesystem) und der Zerfallslänge ist $\ell = \beta c\gamma t$ mit $\beta = v/c$ (v = Teilchengeschwindigkeit, c = Lichtgeschwindigkeit), wobei ein Faktor $\gamma = 1/\sqrt{1-\beta^2}$ für die relativistische Zeitdilatation enthalten ist. Mit der Masse m (in GeV/c^2) und dem Impuls p (in GeV/c) kann die Zerfallslänge geschrieben werden als $\ell = ctp/m$, und die Wahrscheinlichkeitsdichte wird

$$f(\ell|\tau)\,\mathrm{d}\ell = \left(\frac{m}{c\tau\,p}\right)\frac{e^{-\ell\,m/(c\tau\,p)}}{\left(e^{-\ell_1\,m/(c\tau\,p)} - e^{-\ell_2\,m/(c\tau\,p)}\right)}\,\mathrm{d}\ell\,.$$

Die Abbildung 6.5 zeigt die negative Log-Likelihood-Funktion, aufgetragen gegen $c\tau$, die mittlere Lebensdauer mal der Lichtgeschwindigkeit in cm, für eine Stichprobe von 50 Einzelmessungen. Das Minimum ist in der Nähe des erwarteten Wertes $c\tau = 2.68$ cm.

Die Form der negativen Log-Likelihood-Funktion ist unsymmetrisch um das Minimum. Wenn statt des Parameters $c\tau$ in der Abbildung 6.5(a) der inverse Wert, also $1/c\tau$, benutzt wird (Abbildung 6.5(b)), ist die Form der Funktion in guter Näherung symmetrisch. Aus beiden Abbildungen können die Parameter-Fehler in konsistenter Weise bei den Funktionswerten $F_{\min} + 1/2$ abgelesen werden. Für den Fall asymmetrischer Fehlergrenzen ist jedoch nicht garantiert, dass jeweils 16% ausserhalb der rechts- bzw. linksseitigen Standardabweichung liegen [62].

6.5.2 Histogramme

Wenn die Gesamtzahl n der Datenpunkte groß ist, kann man das Ergebnis oft mit Vorteil in Form eines Histogramms präsentieren, wobei die x-Achse in J Intervalle (*bins*) eingeteilt wird. Oft will man dann eine von einem Parameter a abhängende Wahrscheinlichkeitsdichte $f(x|a)$ an die Zahl der Datenpunkte in jedem der J Intervalle anpassen. Die Zahl der Eintragungen

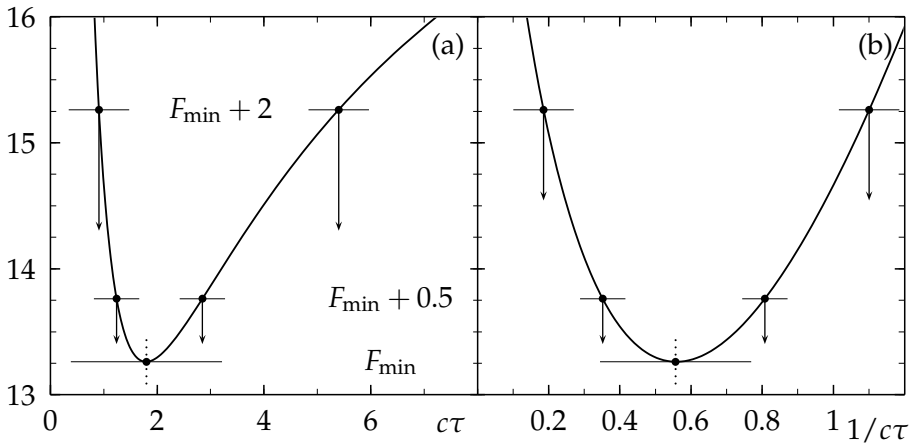


Abbildung 6.5: Die negative Log-Likelihood-Funktion als Funktion der Lebensdauer $c\tau$ (a) und der inversen Lebensdauer $1/c\tau$ (b).

im j-ten Intervall sei n_j, mit $j = 1, 2, \ldots J$ und $\sum_{j=1}^{J} n_j = n$. Die Zahlen der Eintragungen n_j im Intervall J sind Zufallsvariable, verteilt gemäß einer Poisson-Verteilung mit Erwartungswert μ_j, der von dem Parameter a abhängt:

$$P(n_j|\mu_j) = \frac{\mu^{n_j} e^{-\mu_j}}{n_j!} \; .$$

Die Poisson-Verteilung beschreibt die statistische Verteilung der Zahl der Eintragungen in jedem Intervall. Der Erwartungswert $\mu_j(a)$ dieser Zahl wird gewonnen durch Integration der Wahrscheinlichkeitsdichte $f(x|a)$ über die Länge des Intervalls und anschließende Multiplikation mit der Größe der Stichprobe n,

$$\mu_j(a) = n \int_{\text{bin j}} f(x|a)\, \mathrm{d}x \approx n \cdot f(x_c|a) \cdot \Delta x \, ,$$

wobei x_c der x-Wert in der Intervallmitte und Δx die Intervallbreite ist. Nun kann die Maximum-Likelihood-Methode auf die Daten in dem Histogramm angewandt werden. Die negative Log-Likelihood-Funktion ist

$$F(a) = -\sum_{j=1}^{J} \ln \left(\frac{\mu_j^{n_j} e^{-\mu_j}}{n_j!} \right) = -\sum_{j=1}^{J} n_j \ln \mu_j + \sum_{j=1}^{J} \mu_j + \sum_{j=1}^{J} \ln(n_j!) \; . \tag{6.16}$$

Die Gleichung gilt auch, wenn die Zahlen n_j in einzelnen Intervallen klein oder sogar null sind; sie ist also universell anwendbar. Der letzte Term kann für die Minimierung ignoriert werden, da er für die vorgegebenen Daten konstant ist. Dies ist die Formel, die in allen Anwendungen mit Poisson-Statistik verwendet werden sollte, da sie erwartungstreu ist.

Der Funktionswert im Minimum scheint schwierig zu interpretieren. Man kann ihn aber angenähert auf eine χ^2-Form bringen, wenn man eine passende Konstante addiert, nämlich

$$\sum_{j=1}^{J} \left(n_j \ln n_j - n_j - \ln(n_j!) \right) \; .$$

Sie hängt nur von den gemessenen Zahlen n_j ab und ändert sich nicht, wenn der Parameter μ_j variiert wird; die neue negative Log-Likelihood-Funktion hat ihr Minimum an derselben Stelle.

Sie ist

$$F_m(a) = -\sum_{j=1}^{J} \left(n_j \ln \frac{\mu_j}{n_j} - (\mu_j - n_j) \right) + \text{konst}$$

(für $n_j = 0$ muß der Klammerausdruck durch $-\mu_j$ ersetzt werden). Nun hat für $\mu_j \approx n_j$ die Größe $S_m(a) = 2F_m(a)$ eine einfache Interpretation. Mit der Näherung

$$\ln \frac{\mu_j}{n_j} = \ln \left(1 + \frac{\mu_j - n_j}{n_j} \right) \approx \frac{\mu_j - n_j}{n_j} - \frac{1}{2} \left(\frac{\mu_j - n_j}{n_j} \right)^2$$

erhält man

$$S_m(a) = \sum_j \frac{(\mu_j - n_j)^2}{n_j} \,. \tag{6.17}$$

Diese Likelihood-Funktion, genannt Neyman's Chi-Quadrat, hat also wieder eine anschauliche näherungsweise Interpretation. Für kleine Werte n_j ist die Näherung offensichtlich schlecht, denn sie gibt Intervallen mit kleinen Werte n_j ein gr0ßes Gewicht; bei wachsenden n_j geht der Fehler allerdings gegen Null. Eine etwas bessere Näherung erhält man, wenn man in diesem Fall im Nenner n_j durch $\mu_j(a)$ ersetzt:

$$S_m(a) = \sum_j \frac{(\mu_j - n_j)^2}{\mu_j(a)} \,. \tag{6.18}$$

Bei einem Fit mit der Bestimmung des Minimums von $S_m(a)$ bezüglich des Parameters a wird das Ergebnis einen Bias zu größeren Werten des Nenners $\mu_j(a)$ haben. Man kann diesen Bias verhindern durch folgende Strategie: Man wiederholt den Fit iterativ, indem man im Nenner den Werte von $\mu_j(a)$ auf dem Wert der vorigen Iteration festhält. Damit wird die bessere Näherung (6.18) benutzt, und die Ableitung des Nenners bezüglich des Parameters a verschwindet.

Für den Fall einer Gauß-Verteilung wird die negative Log-Likelihood-Funktion

$$\begin{aligned} F(a) &= -\sum_{j=1}^{J} \ln \left(\frac{1}{\sqrt{2\pi}\sigma_j} \exp(-(n_j - \mu_j)^2 / 2\sigma_j^2) \right) \\ &= \frac{1}{2} \sum_{j=1}^{J} \frac{\left(n_j - \mu_j\right)^2}{\sigma_j^2} + \text{konst}\,. \end{aligned} \tag{6.19}$$

Bis auf einen Faktor $1/2$ vor der Summe ist die Gleichung dann identisch mit der Gleichung (6.10) der Methode der kleinsten Quadrate. In diesem Fall Gauß-verteilter Fehler folgt die Größe $2F(\widehat{a})$ also einer χ^2-Verteilung mit k Freiheitsgraden, wobei k = Zahl der Intervalle J *minus* Zahl der angepaßten Parameter ist. Der Wert von $2F(\widehat{a})/k$ sollte für eine akzeptable Anpassung etwa eins sein. In diesem Fall enthält der Wert von $F(\widehat{a})$ also nützliche Information über die Güte der Anpassung.

6.5.3 Erweiterte Maximum-Likelihood-Methode

Bei der üblichen Anwendung der Maximum-Likelihood-Methode werden unter Benutzung der korrekt normierten Wahrscheinlichkeitsdichte Parameter $\boldsymbol{a}$ dieser Dichte aus der Messung von n Ereignissen bestimmt. Bei bestimmten Problemen ist auch die mittlere Anzahl der Ereignisse selbst ein Parameter, der zu bestimmen ist. Man benutzt dann nicht eine normierte

Wahrscheinlichkeitsdichte $f(x|\boldsymbol{a})$, sondern einen Ausdruck $g(x|\boldsymbol{a})$, dessen Integral im Meßbereich Ω der *Anzahl erwarteter Ereignisse* N entspricht,

$$N = \int_\Omega g(x|\boldsymbol{a})\,\mathrm{d}x\,,$$

wobei N von den Parametern $\boldsymbol{a}$ abhängt. In der erweiterten Maximum-Likelihood-Methode wird für dieses Problem die negative Log-Likelihood-Funktion

$$F(\boldsymbol{a}) = -\sum_{i=1}^{n} \ln(g(x_i|\boldsymbol{a}) + \int_\Omega g(x|\boldsymbol{a})\mathrm{dx} \tag{6.20}$$

definiert.

Zur Herleitung dieser Funktion setzt man den Ausdruck $g(x|\boldsymbol{a}) = N \cdot f(x|\boldsymbol{a})$ mit einer korrekt normierten Dichte $f(x|\boldsymbol{a})$. Die übliche Likelihood-Funktion $L(\boldsymbol{a}) = f(x_1|\boldsymbol{a}) \cdot f(x_2|\boldsymbol{a}) \cdots f(x_n|\boldsymbol{a})$ wird multipliziert mit der Poisson-Wahrscheinlichkeit $P = N^n \mathrm{e}^{-N}/n!$, bei einem Erwartungswert von N Ereignissen tatsächlich n Ereignisse zu beobachten. Dieser im Rahmen des Likelihood-Prinzips liegende Ansatz ergibt die negative Log-Likelihood-Funktion

$$F(\boldsymbol{a}) = -\sum_{i=1}^{n} \ln(f(x_i|\boldsymbol{a}) - n\ln N + N + \ln n!\,,$$

die, abgesehen von dem konstanten Term $\ln n!$, identisch mit der Funktion (6.20) ist.

Zur Herleitung kann man auch von dem Fall eines Histogramms ausgehen, bei dem für jedes Intervall die Poisson-Verteilung angesetzt wird. Im Grenzfall sehr kleiner Intervallbreiten ist die Zahl der Ereignisse pro Intervall entweder 0 oder 1, und man erhält in diesem Grenzfall aus der Gleichung (6.15) wiederum die Gleichung (6.20).

Vorsicht ist bei der erweiterten Maximum-Likelihood-Methode allerdings bei der Berechnung der Kovarianzmatrix der Parameter geboten, welche nicht immer mit der Standardmethode übereinstimmt. Hierfür sei auf den ausführlichen Artikel von Barlow [4] verwiesen.

6.6 Eigenschaften der Maximum-Likelihood-Methode

6.6.1 Konsistenz

Wenn $\widehat{a}$ die beste Schätzung des Parameters a ist als Lösung der Maximum Likelihood-Gleichung (6.2), dann ist diese Schätzung konsistent, wenn $\widehat{a}$ sich dem wahren Wert a_0 beliebig annähert, wenn die Zahl der Beobachtungen n beliebig groß wird

$$\lim_{n\to\infty} \widehat{a} = a_0\,.$$

Dies ist in der Tat der Fall. Beweis: Die Bedingung für das Minimum der negativen Log-Likelihood-Funktion $F(a)$ lautet

$$\frac{\mathrm{d}F}{\mathrm{d}a} = \frac{\mathrm{d}}{\mathrm{d}a}\sum_{i=1}^{n} \ln f(x_i|a) = 0 \quad \text{für} \quad a = \widehat{a}\,.$$

Man führt die Funktion $Q(a)$ ein durch

$$Q(a) = -\frac{1}{n}\cdot\frac{\mathrm{d}F}{\mathrm{d}a} = \frac{1}{n}\sum_{i=1}^{n}\frac{\mathrm{d}}{\mathrm{d}a}(\ln f(x_i|a))\,, \tag{6.21}$$

wobei nach Definition der Wert $Q(\widehat{a}) = 0$ ist. Mit der Abkürzung

$$y(x) = \frac{\mathrm{d} \ln f(x|a)}{\mathrm{d}a} \quad \text{für} \quad a = a_0$$

ist der Erwartungswert von y gegeben durch (siehe Kapitel 4.3.2)

$$E[y] = \int \frac{\mathrm{d}}{\mathrm{d}a} \ln f(x|a)|_{a=a_0} f(x|a_0)\, \mathrm{d}x\,.$$

Das Integral ist null, wie man nach einer Umformung sieht.

$$\int \frac{\mathrm{d}}{\mathrm{d}a} \ln f(x|a)|_{a=a_0} f(x|a_0)\, \mathrm{d}x = \frac{\mathrm{d}}{\mathrm{d}a} \int f(x|a)|_{a=a_0}\, \mathrm{d}x = 0\,,$$

weil $\int f(x|a)\, \mathrm{d}x = 1$ die Normierung für *alle* Werte von a ist. Auf der anderen Seite hat man

$$Q(a_0) = \frac{1}{n} \sum_{i=1}^{n} y_i\,.$$

Nach dem Zentralen Grenzwertsatz geht die Summe gegen $E[y]$. Deshalb ist

$$\lim_{n\to\infty} Q(a_0) = E[y] = 0 = Q(\widehat{a})\,,$$

was den Beweis abschließt.

6.6.2 Erwartungstreue

Eine Schätzung heißt erwartungstreu, wenn $E[\widehat{a}] = a_0$. Man sollte beachten, daß die Maximum-Likelihood-Schätzung $\widehat{a}$ im nicht-asymptotischen Fall im allgemeinen nicht erwartungstreu (also nicht unverzerrt) ist. Dies ist leicht zu sehen, wenn man den Parameter $a \to a'$ transformiert entsprechend $a = g(a')$. Die Likelihood-Funktion $L(a)$ geht damit in die Funktion $L(g(a')) = L'(a')$ über. Die Funktion $L'(a')$ hat ihr Maximum an der entsprechenden Stelle $\widehat{a'}$ mit $\widehat{a} = g(\widehat{a'})$. Dies ist eine wichtige Invarianzeigenschaft der Maximum-Likelihood-Methode. Die Transformation in den neuen Parameter kann jedoch zu einer Verzerrung führen (oder eine solche beseitigen). Dies zeigt Abbildung 6.5 an einem Beispiel. Wenn die Maximum-Likelihood-Kurve und damit die Fehler symmetrisch um $\widehat{a}$ sind, dann sind sie es im allgemeinen nicht mehr in dem neuen Parameter a', da die a-Skala in eine a'-Skala geändert ist. Die Fehlergrenzen sind dann nicht mehr symmetrisch, was zu einer Verzerrung führt. Formal folgt das auch aus Gleichung (4.47). Danach wird der Erwartungswert $E[\widehat{a}]$ nicht gleich $E[g(\widehat{a'})]$ sein, während $\widehat{a} = g(\widehat{a'})$, und deshalb können die Schätzungen $\widehat{a}$ und $\widehat{a'}$ nicht beide gleichzeitig erwartungstreu sein. Für $n \to \infty$ werden die zwei Erwartungswerte gleich, da dann σ_x^2 in Gleichung (4.47) und damit auch σ_a^2 gegen null geht nach dem Zentralen Grenzwertsatz. Die Maximum-Likelihood-Schätzung wird dann asymptotisch erwartungstreu.

6.6.3 Asymptotische Annäherung an die Gauß-Funktion

Man entwickelt die Funktion $Q(a)$ (Gleichung (6.21)) um den wahren Wert a_0 des Parameters,

$$Q(a) = Q(a_0) + (a - a_0) \left.\frac{\mathrm{d}Q(a)}{\mathrm{d}a}\right|_{a_0} + \ldots\,.$$

Da $\widehat{a}$ eine konsistente Schätzung ist, folgt $\widehat{a} \to a_0$ in der asymptotischen Grenze. Dann kann man quadratische und höhere Terme vernachlässigen, und $Q(a)$ wird eine lineare Funktion von a, und weiterhin gilt

$$\widehat{a} - a_0 = -\frac{Q(a_0)}{\mathrm{d}Q/\mathrm{d}a} = -\frac{1}{n} \sum_i^n \frac{\mathrm{d} \ln f(x_i|a)}{\mathrm{d}a} \Big/ \left(\mathrm{d}Q/\mathrm{d}a\right)|_{a_0}\,.$$

Nach dem Zentralen Grenzwertsatz nimmt die Zufallsvariable $\widehat{a} - a_0$ asymptotisch die Form einer Gauß-Verteilung an mit Mittelwert null und Varianz

$$V\left[(\widehat{a} - a_0)\right] = V\left[\widehat{a}\right] = \frac{1}{n} E\left[\left(\frac{\mathrm{d}\ln f}{\mathrm{d}a}\right)^2\right] \Big/ \left(\frac{\mathrm{d}Q}{\mathrm{d}a}\Big|_{a_0}\right)^2 . \tag{6.22}$$

Dieser Ausdruck wird umgeformt zu

$$E\left[\frac{\mathrm{d}^2\ln f}{\mathrm{d}a^2}\right] = E\left[\frac{\mathrm{d}}{\mathrm{d}a}\left(\frac{\mathrm{d}\ln f}{\mathrm{d}a}\right)\right]$$
$$= E\left[\frac{\mathrm{d}}{\mathrm{d}a}\left(\frac{1}{f}\frac{\mathrm{d}f}{\mathrm{d}a}\right)\right] = E\left[\frac{1}{f}\frac{\mathrm{d}^2 f}{\mathrm{d}a^2}\right] - E\left[\left(\frac{\mathrm{d}\ln f}{\mathrm{d}a}\right)^2\right] .$$

Nunmehr gilt

$$E\left[\frac{1}{f}\frac{\mathrm{d}^2 f}{\mathrm{d}a^2}\right] = \int \frac{1}{f}\frac{\mathrm{d}^2 f}{\mathrm{d}a^2}\cdot f\,\mathrm{d}x = \frac{\mathrm{d}^2}{\mathrm{d}a^2}\int f\,\mathrm{d}x = 0\,,$$

weil das Integral gerade die Normierung und deshalb konstant ist. Deshalb ist

$$E\left[\left(\frac{\mathrm{d}\ln f}{\mathrm{d}a}\right)^2\right] = -E\left[\frac{\mathrm{d}^2\ln f}{\mathrm{d}a^2}\right] . \tag{6.23}$$

Weiterhin geht $\mathrm{d}Q/\mathrm{d}a$ für $n \to \infty$ gegen $E\left[\mathrm{d}^2\ln f/\mathrm{d}a^2\right]$ nach dem Zentralen Grenzwertsatz. Setzt man den Ausdruck in Gleichung (6.22) ein und benutzt Gleichung (6.23), so erhält man

$$V\left[\widehat{a}\right] = -\frac{1}{n} E\left[\frac{\mathrm{d}^2\ln f}{\mathrm{d}a^2}\right]^{-1} . \tag{6.24}$$

Wenn man den Erwartungswert durch den Mittelwert ersetzt, erhält man Gleichung (6.11) unter Verwendung von Gleichung (6.3).

Gleichung (6.24) ist nützlich, weil sie die zu erwartende Genauigkeit der Parameterschätzung angibt. Man berechnet sie vermöge

$$E\left[\frac{\mathrm{d}^2\ln f}{\mathrm{d}a^2}\right] = \int_\Omega \frac{\mathrm{d}^2\ln f}{\mathrm{d}a^2}\cdot f\,\mathrm{d}x\,.$$

Man kann zeigen, daß die Varianz nach Gleichung (6.24) oder nach Gleichung (6.11) die kleinstmögliche ist: Die Maximum Likelihood-Methode ist asymptotisch effizient. Dies folgt aus dem *Theorem für die minimale Varianz*

$$V\left[\widehat{a}\right] \geq -\frac{1}{n} E\left[\frac{\mathrm{d}^2\ln f}{\mathrm{d}a^2}\right]^{-1} .$$

Wegen eines Beweises siehe zum Beispiel [3, 10].

6.7 Konfidenzgrenzen

Angenommen, eine Größe mit dem wahren Wert x_w wird gemessen, wobei die Meßapparatur den Gauß-verteilten Fehler σ hat. Das Ergebnis der Messung ist eine Zufallsvariable x_m, die einer Gauß-Verteilung mit Mittelwert x_w und Standardabweichung σ gehorcht. Häufig hat man jedoch das umgekehrte Problem: Eine Messung mit dem bekannten Fehler σ liefert den Wert x_m. Was kann man über den wahren Wert x_w der Größe x aussagen? Der wahre Wert ist unbekannt, jedoch sind Wahrscheinlichkeitsaussagen über obere und untere Grenzen für x_w möglich. Diese Grenzen nennt man *Konfidenzgrenzen.*

Im folgenden sind zwei einfache Fälle, normalverteilte und Poisson-verteilte Messgrössen, behandelt. Aber nicht immer ist es so einfach. Dann ist oft der Einsatz von Monte Carlo-Methoden angebracht.

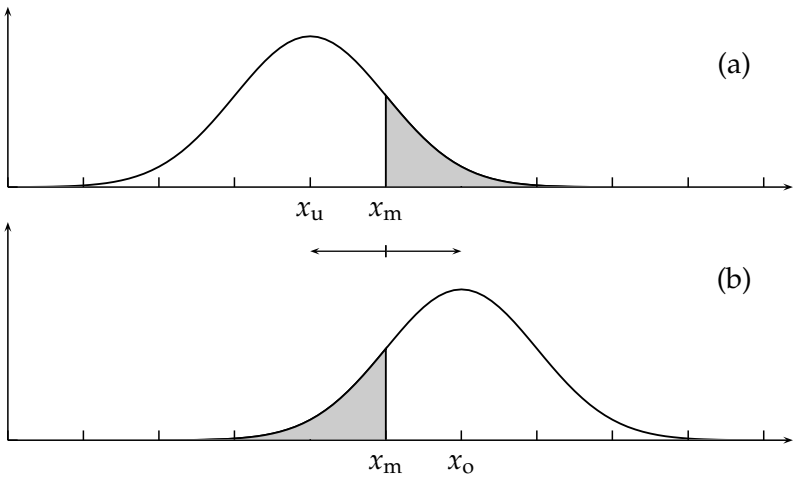


Abbildung 6.6: Normalverteilungen mit Mittelwerten x_u (a) und x_o (b). Diese beiden Werte sind die unteren und oberen Grenzen mit einer (einseitigen) Konfidenz von je $(100 - 15.87)\% = 84.13\%$ bei einem beobachteten Wert von x_m. Die grauen Flächen entsprechen einer Wahrscheinlichkeit von 15.87%.

Normalverteilte Meßgrößen. Der einfachste Fall ist eine Messung mit Gauß-verteilten Fehlern. Um eine obere Grenze x_o zu erhalten, nimmt man an, der wahre Wert von x wäre x_o. Man verlangt dann, daß das Meßergebnis x_m oder ein noch kleineres nur mit der (kleinen) Wahrscheinlichkeit α_o auftritt,

$$\alpha_o = \int_{-\infty}^{x_m} N(x_o, \sigma)\,\mathrm{d}x = \int_{-\infty}^{x_m} \frac{1}{\sqrt{2\pi}\sigma} \exp(-(x - x_o)^2/2\sigma^2)\,\mathrm{d}x\,.$$

Dies ist eine Gleichung für x_o, wenn α_o vorgegeben ist, und ist eine sichere obere Grenze für den wahren Wert x_w (siehe Abbildung 6.6.) Auf Grund der Messung kann man also eine obere Grenze x_o mit einer *Konfidenz* von $1 - \alpha_o$ angeben, d.h. die Aussage, daß $x_w > x_o$ ist, hat eine Wahrscheinlichkeit von *weniger* als α_o. Kleine Werte von α_o führen zu großen (vorsichtigen) Werten von x_o.

Entsprechend definiert man eine untere Konfidenzgrenze x_u mit der Konfidenz $1 - \alpha_u$ durch

die Forderung, daß man den gegebenen Meßwert x_m oder einen noch größeren mit der (kleinen) Wahrscheinlichkeit α_u erhält, wenn der wahre Wert x_u ist,

$$\alpha_u = \int_{x_m}^{\infty} N(x_u, \sigma)\,\mathrm{d}x = \int_{x_m}^{\infty} \frac{1}{\sqrt{2\pi}\sigma} \exp(-(x - x_u)^2/2\sigma^2)\,\mathrm{d}x$$

(siehe Abbildung 6.6). Hieraus folgt wieder x_u als Funktion von α_u. Man sagt, die untere Grenze von x_w sei x_u mit einer Konfidenz von $1 - \alpha_u$. Man kann somit den wahren Wert zwischen eine obere und untere Grenze einschließen und sagen, der wahre Wert liegt zwischen x_u und x_o mit einer Konfidenz von $1 - \alpha_o - \alpha_u$. Die Wahrscheinlichkeit für diese Aussage ist kleiner oder gleich $\alpha_o + \alpha_u$.

Will man zum Beispiel die gesuchte Größe mit einer Konfidenz von 68% mit symmetrischen Grenzen einschließen, so entspricht dies im Falle einer Gauß-Verteilung $\alpha_o = \alpha_u = 15.87\%$, und $x_o = x_m + \sigma$, $x_u = x_m - \sigma$. Ist man nur an einer oberen Grenze interessiert und wählt $x_o = x_m + \sigma$, so entspricht dies einer Konfidenzgrenze von $(100 - 15.87)\% = 84.13\%$. Es ist deshalb bei der Angabe von Konfidenzgrenzen wichtig, zu spezifizieren, ob man eine einseitige obere bzw. eine einseitige untere oder eine beidseitige Grenze meint. Bei beidseitigen Grenzen wählt man in der Regel symmetrische obere und untere Konfidenzen, also $\alpha_o = \alpha_u$. Gebräuchlich sind Zahlenwerte von 95% für die Konfidenz, entsprechend etwa $2\,\sigma$ im Fall von gaußischen Fehlern. Von der Unsitte, Konfidenzgrenzen in σ anzugeben, sollte man ganz absehen.

Poisson-verteilte Meßgrößen. Für den Fall von Poisson-verteilten Größen sind die Überlegungen ganz analog, nur müssen die Integrale durch Summen ersetzt werden. Das folgende steht für einen häufig vorkommenden Fall. Angenommen, man beobachtet von einem schwach radioaktiven Präparat 9 Zerfälle. Was kann man hieraus über die *mittlere Zahl* von Zerfällen in dem betreffenden Meßzeitintervall sagen, also über den Mittelwert μ, den einzigen Parameter der zugrundeliegenden Poisson-Verteilung? Die Antwort erfolgt wieder durch Angabe von Konfidenzgrenzen.

Man erhält eine obere Grenze μ_o für den Mittelwert μ, indem man fordert, daß eine Poisson-Verteilung mit dem Mittelwert μ_o zu der beobachteten Zahl n oder einer noch kleineren Zahl mit der (kleinen) Wahrscheinlichkeit α_o führt. Man erhält so μ_o, die obere Grenze für die Konfidenz $1 - \alpha_o$. Die Bedingung für μ_o lautet in Gleichungsform

$$\alpha_o = \sum_{r=0}^{n} \frac{\exp(-\mu_o)\,\mu_o^r}{r!}.$$

Die Lösung erhält man aus der Gleichung

$$\alpha_o = \sum_{r=0}^{n} \frac{\exp(-\mu_o)\,\mu_o^r}{r!} = 1 - P(\chi^2_{2n+2} \le 2\,\mu_o)\,, \tag{6.25}$$

wobei $P(\chi_k^2 \le 2\mu_o)$ die χ^2-Wahrscheinlichkeit für k Freiheitsgrade ist, einen Wert $\le 2\mu_o$ zu erhalten.

Zum Beweis betrachtet man

$$\int_{2m}^{\infty} \varphi(u)\mathrm{d}u = \int_{2m}^{\infty} \frac{\exp(-\mu/2)\,\mu^\beta\,\mathrm{d}u}{\beta^m\,2^{k/2}}$$

und benutzt die Beziehung

$$\int u^b\,e^{au}\mathrm{d}u = e^{au}\left[\frac{u^b}{a} + \sum_{k=1}^{b} \frac{(-1)^k\,b(b-1)\ldots(b-k+1)}{a^{k+1}} u^{b-k}\right].$$

In ähnlicher Weise erhält man eine untere Grenze μ_u für den Mittelwert, indem man fordert, daß eine Poisson-Verteilung mit dem Mittelwert μ_u nur mit einer (kleinen) Wahrscheinlichkeit α_u zu der beobachteten Zahl n oder einer noch größeren führt; in Gleichungsform ist die Lösung

$$\alpha_u = \sum_{r=n}^{\infty} \frac{e^{-\mu_u}\,\mu_u^r}{r!} = P(\chi^2_{2n} \le 2\mu_u). \tag{6.26}$$

In der Praxis hat man bei der Beobachtung kleiner Signale oft das Problem, daß Untergrundbeiträge bei der Berechnung der Grenzen beachtet werden müssen. Von Feldman und Cousins[21] wurde eine auf klassischer Statistik beruhende Methode entwickelt, die unphysikalische Grenzen vermeidet und einheitlich obere Grenzen und zweiseitige Intervalle behandelt.

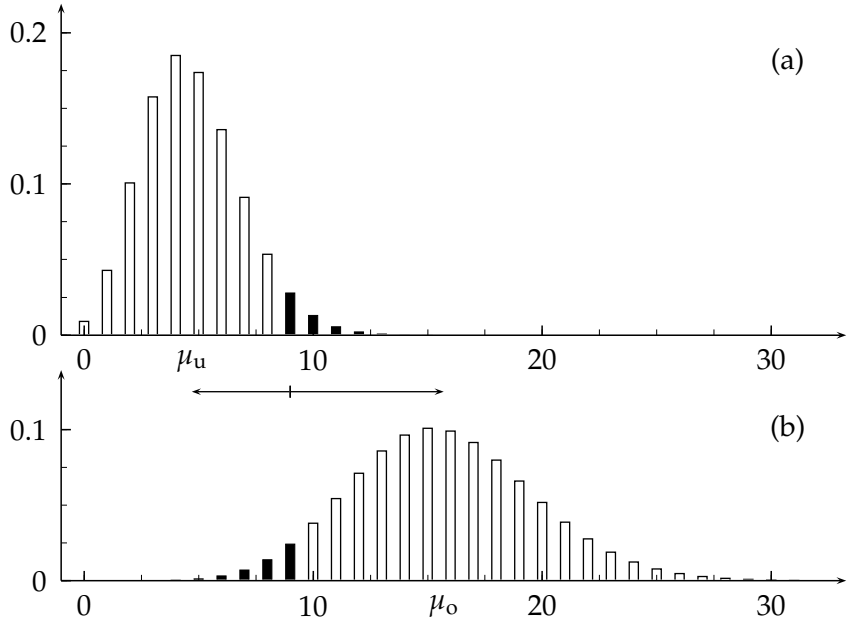


Abbildung 6.7: Poissonverteilungen mit Parameter $\mu_u = 4.7$ (a) und $\mu_o = 15.7$ (b). Diese beiden Werte sind die unteren und oberen Grenzen für eine (einseitige) Konfidenz von 95% bei einem beobachteten Wert von $r = 9$. Der Anteil der schwarzen Balken entspricht jeweils 5%.

Beispiel 6.11 Konfidenzintervall bei der Poisson-Verteilung.

Es werden $n = 9$ radioaktive Zerfälle beobachtet. Welches ist die zugrundeliegende mittlere Zahl von Zerfällen? Man wählt ein 90% Konfidenzintervall mit symmetrischen Konfidenzgrenzen $\alpha_o = \alpha_u = 5\%$, entsprechend etwa $1.645\,\sigma$ bei einer Gauß-Verteilung. Die oberen und unteren Grenzen sind definiert durch die beiden Gleichungen

$$\alpha_u = 0.05 = \int_0^{\chi^2=2\mu_u} \varphi_{18}(t)\,\mathrm{d}t \qquad \alpha_o = 0.05 = 1 - \int_0^{\chi^2=2\mu_o} \varphi_{20}(t)\,\mathrm{d}t$$

und es folgen daraus die Grenzen $\mu_u = 4.7$ und $\mu_o = 15.7$ (siehe Tabelle 6.1). Das Resultat (mit 90% Konfidenz) ist $\mu = 9^{+6.7}_{-4.3}$, zu vergleichen mit den *naiven* (und falschen) Grenzen $\mu = 9 \pm 1.645\sqrt{9} = 9 \pm 4.9$ aus der Gaußschen Näherung. Die Vorgehensweise zur Bestimmung der Grenzen ist in der Abbildung 6.7 dargestellt, wobei die Tabelle 6.1 benutzt wird.

n	84.2%	90%	95%	97.5%	97.5%	95%	90%	84.2%
0					3.69	3.00	2.30	1.84
1	0.17	0.11	0.05	0.03	5.57	4.74	3.89	3.30
2	0.71	0.53	0.36	0.24	7.22	6.30	5.32	4.64
3	1.37	1.10	0.82	0.62	8.77	7.75	6.68	5.92
4	2.08	1.74	1.37	1.09	10.24	9.15	7.99	7.17
5	2.84	2.43	1.97	1.62	11.67	10.51	9.27	8.39
6	3.62	3.15	2.61	2.20	13.06	11.84	10.53	9.59
7	4.42	3.89	3.29	2.81	14.42	13.15	11.77	10.77
8	5.23	4.66	3.98	3.45	15.76	14.43	12.99	11.95
9	6.05	5.43	4.70	4.12	17.08	15.71	14.21	13.11
10	6.89	6.22	5.43	4.80	18.39	16.96	15.41	14.27
11	7.73	7.02	6.17	5.49	19.68	18.21	16.60	15.42
12	8.58	7.83	6.92	6.20	20.96	19.44	17.78	16.56
13	9.44	8.65	7.69	6.92	22.23	20.67	18.96	17.70
14	10.30	9.47	8.46	7.65	23.49	21.89	20.13	18.84
15	11.17	10.30	9.25	8.40	24.74	23.10	21.29	19.96
16	12.04	11.14	10.04	9.15	25.98	24.30	22.45	21.09
17	12.91	11.98	10.83	9.90	27.22	25.50	23.61	22.21
18	13.79	12.82	11.63	10.67	28.45	26.69	24.76	23.33
19	14.68	13.67	12.44	11.44	29.67	27.88	25.90	24.44
20	15.56	14.53	13.25	12.22	30.89	29.06	27.05	25.55

Tabelle 6.1: Untere (links) und obere Grenzen (rechts) des Parameters μ der Poisson-Verteilung bei beobachteten Werten von 0 bis 20. Angegeben ist die einseitige Konfidenz.

6.8 Bayes'sche Statistik

Was kann man über den wahren Wert einer Größe aufgrund einer fehlerbehafteten Messung aussagen? Nach den Ausführungen des vorhergehenden Kapitels können lediglich Aussagen über obere und untere Grenzen mit einer gewissen Wahrscheinlichkeit gemacht werden. Dies ist der Standpunkt der *klassischen* Statistik. In der Bayes'schen Statistik macht man Aussagen über den wahren Wert selbst, zum Beispiel *der wahre Wert von x ist* $x_m \pm \sigma$ *mit einer Wahrscheinlichkeit von 68%*. Diese Aussage, die im Falle gaußischer Fehler numerisch mit derjenigen der klassischen Statistik übereinstimmt, ist ja das, was man eigentlich wissen will, aber sie hat ihren Preis. Man benötigt für eine derartige Aussage eine zusätzliche Annahme, welche oft stillschweigend gemacht wird. Dies ist die Annahme, daß der wahre Wert mit einer gewissen Wahrscheinlichkeit irgendwo in der Nähe des gemessenen ist. Um dies zu formulieren, benutzt man das Bayes'sche Theorem in der Form (siehe Gleichung (4.6))

$$f(a|x) = f(x|a) \cdot f(a)/g(x)\,, \tag{6.27}$$

wobei x für die Daten steht und a für den Parameter, der durch Anpassung an die Daten bestimmt werden soll. In dieser Gleichung ist $f(a|x)$ die bedingte Wahrscheinlichkeit von a, vorausgesetzt, die Daten x sind gegeben, und $f(x|a)$ ist die bedingte Wahrscheinlichkeit der Daten x für eine bestimmte Wahl des Parameters a. Diese letztere Funktion kann nach den Regeln der Statistik, etwa als Likelihood-Funktion, bestimmt werden (siehe Kapitel 6.3.1). Was man eigentlich möchte, ist die Funktion $f(a|x)$. Um sie zu berechnen, benötigt man weiterhin $f(a)$ und $g(x)$. Die Funktion $g(x)$, die Wahrscheinlichkeitsdichte der Daten, liegt fest; sie bleibt während der Anpassung konstant und kann weggelassen oder in der Normierung absorbiert werden. Die zweite Funktion $f(a)$, die Wahrscheinlichkeitsdichte von a, ist streng genommen eine unsinnige Größe, da der Parameter a einen genau bestimmten, wenn auch unbekannten Wert hat. Sie muß also interpretiert werden als der *Glaube*, daß a einen bestimmten Wert hat. Dieser Einbruch des Glaubens in die Statistik wird von vielen abgelehnt, jedoch ist dies notwendig, wenn man Aussagen über die wahren Werte von Parametern machen will. Dies ist

vermutlich harmlos, solange man sich darüber im klaren ist, was man macht.

In der Sprache der Bayes'schen Statistik heißt $f(a)$ der *Prior*, und $f(a|x)$ der *Posterior*.

Als Beispiel möge eine Messung der Elektronenmasse dienen. Wenn man einen Wert von m_e= 0.511 MeV/c^2 erhält, mit einer Standardabweichung $\sigma = 0.001$, so ist es nicht unsinnig, zu glauben, daß die wahre Elektronenmasse mit großer Wahrscheinlichkeit in der Nähe von 0.511 liegt.

Setzt man $f(a)$ = konstant, so erhält man $f(a|x) = N \cdot f(x|a)$ mit N gleich einer Normierungskonstanten. Es folgt, daß mit dieser Wahl von $f(a)$ die Aussagen in Kapitel 6.7 zu numerisch identischen Aussagen über die Grenzen des Parameters a führen. Das Beispiel mit der Gauß-Verteilung führt zu der Aussage: *Der Wert der Elektronenmasse ist* $m_e \pm \sigma$.

Es mag vernünftig erscheinen, wie oben geschehen, eine Gleichverteilung in a für den Prior $f(a)$ anzunehmen, aber es mag ebenso gute (oder schlechte) Gründe geben, stattdessen eine Gleichverteilung in a^2 oder $\log a$ zu benutzen, und dies wird zu verschiedenen Resultaten für die Schätzung von a und seinen Fehlern führen. Um den Einfluß dieser Willkür abzuschätzen, betrachtet man das folgende Beispiel, das leicht verallgemeinert werden kann.
Angenommen, μ ist der Mittelwert einer Wahrscheinlichkeitsdichte und $\bar{x}$ der Mittelwert einer Stichprobe aus dieser Wahrscheinlichkeitsdichte. Nach dem Zentralen Grenzwertsatz ist die Wahrscheinlichkeitsdichte der Zufallsvariablen $\bar{x}$ näherungsweise gegeben durch die Gauß-Verteilung $N(\mu, \sigma)$, wobei σ^2 die Varianz von $\bar{x}$ ist. Wenn der Prior als $f(\mu)$ angenommen wird, ist die gesuchte Wahrscheinlichkeitsdichte von μ nach Gleichung (6.27) gegeben durch

$$f(\mu|\overline{x}) = N(\mu, \sigma) \cdot f(\mu),$$

wobei eine Normierungskonstante weggelassen wurde. Nach der Maximum-Likelihood-Methode ist die beste Schätzung für μ diejenige, welche $f(\mu|\overline{x})$ zu einem Maximum macht, was zu der folgenden Gleichung für μ führt

$$\frac{\mathrm{d}f(\mu|\overline{x})}{\mathrm{d}\mu} = 0 = \frac{\mathrm{d}N(\mu,\sigma)}{\mathrm{d}\mu} \cdot f(\mu) + N(\mu,\sigma) \cdot \frac{\mathrm{d}f(\mu)}{\mathrm{d}\mu},$$

oder

$$\frac{\mathrm{d}\ln N(\mu,\sigma)}{\mathrm{d}\mu} = -\frac{1}{f(\mu)}\frac{\mathrm{d}f(\mu)}{\mathrm{d}\mu} = \frac{(\overline{x}-\mu)}{\sigma^2}.$$

Für $f(\mu)$ = konstant erhält man das Standardresultat $\mu = \overline{x}$, im allgemeinen Fall erhält man aber

$$\mu = \overline{x} + \sigma^2 \cdot \frac{1}{f(\mu)}\frac{\mathrm{d}f(\mu)}{\mathrm{d}\mu}. \tag{6.28}$$

Der Einfluß der Wahl des Priors ist also klein, wenn der Zusatzterm zu $\overline{x}$ klein gegenüber σ ist, also

$$\sigma \cdot \frac{\mathrm{d}f(\mu)}{\mathrm{d}\mu} \ll f(\mu).$$

Entwickelt man $f(\mu)$ um μ_0 mit $f(\mu) = f(\mu_0)\,(1 + \alpha(\mu - \mu_0))$, dann lautet die Bedingung $\alpha\sigma \ll 1$. In Worten: Die relative Änderung des Priors $f(\mu)$ über eine Standardabweichung muß klein sein.

Die sogenannten Bayesianer halten diesen Zugang für vernünftig und natürlich. Mit Argumenten ähnlich denen der Gleichung (6.28) machen sie geltend, daß die Willkür für genaue Messungen klein ist, und daß die statistische Methode sowieso noch andere willkürliche Annahmen

enthält. Sei dem wie dem wolle. Auf jeden Fall bietet die Bayes'sche Statistik ein wohldefiniertes Verfahren, um a priori Wissen bei der Schätzung von Parametern einzubringen, zum Beispiel um absolute Grenzen für den Wert von Parametern zu berücksichtigen, wie sie aus Erhaltungssätzen oder dem gesunden Menschenverstand folgen.

Was ist, wenn man über die a priori Wahrscheinlichkeit gar nichts weiß? Es gibt Leute, die in diesem Fall die Wahrscheinlichkeit zu 0.5 festsetzen. Das kann nicht immer ein guter Rat sein, wie das folgende eher scherzhafte Beispiel zeigt: Was ist die Wahrscheinlichkeit, daß es auf dem Begleitplaneten von Centaurus α Kaninchen gibt? Darüber weiß man nichts, also ist die Wahrscheinlichkeit $P_K = 0.5$. Und Löwen? Auch hier ist die Wahrscheinlichkeit $P_L = 0.5$. Und so weiter. Die Wahrscheinlichkeit, daß mindestens eine dieser Lebensformen auf diesem Planeten existiert, ist das *oder* aller dieser Wahrscheinlichkeiten, nämlich $1 - P_R \cdot P_L \cdots P_N = 1 - 0.5^N$, welches beliebig nahe zu eins = Gewißheit gebracht werden kann, wenn man N groß genug wählt. Was ist hier los? Offensichtlich sind die Wahrscheinlichkeiten korreliert, was in der Rechnung ignoriert wurde.

Eine Anwendung der Bayes'schen Statistik ist die Interpretation der Likelihood-Funktion als proportional zur Wahrscheinlichkeitsdichte der Parameter $\boldsymbol{a}$. Man kann eine Wahrscheinlichkeitsdichte für $\boldsymbol{a}$ definieren durch die Bayes'sche Beziehung $f_a(\boldsymbol{a}|x) = f_0(\boldsymbol{a}) \cdot L(\boldsymbol{a}|x)$, wobei $f_0(\boldsymbol{a})$ der Prior von $\boldsymbol{a}$ ist. Setzt man $f_0(\boldsymbol{a}) = \text{konst}$, so steht die Ableitung der Kovarianzmatrix der Parameter $\boldsymbol{a}$ in Kapitel 6.4 auf formal richtigem Boden. Die Willkür bei der Wahl des Priors $f_0(\boldsymbol{a})$ sollte in der asymptotischen Grenze keine Rolle mehr spielen, weil dann $\sigma \to 0$ ist, und der Einfluß des Priors verschwindet vermöge Gleichung (6.28).

Beispiel 6.12 Suche nach einem Signal.

Man möchte das Vorhandensein eines Signals über einem Störuntergrund feststellen. Dieses Signal könnte zum Beispiel die Zählrate sein, herrührend von einer radioaktiven Verseuchung, der Untergrund die normale Umgebungsradioaktivität plus Höhenstrahlung. Die Untergrundzählrate R_u möge genau bekannt sein. Die Rate des gesuchten Signals sei R_x. Bei der Messung über ein Zeitintervall T hat man n Zählimpulse gefunden. Der erwartete Mittelwert von n ist $\langle n \rangle = R_u\, T + R_x\, T$. Die beste Schätzung für den Mittelwert der gesuchten Zählrate ist

$$\overline{R}_x = \frac{n}{T} - R_u.$$

Die Standardabweichung von $\overline{R}_x$ ist

$$\sigma(\overline{R}_x) = \sqrt{\sigma^2\left(\frac{n}{T}\right) + \sigma^2(R_u)} = \sqrt{\frac{n}{T^2} + 0} = \frac{\sqrt{n}}{T}.$$

Numerisches Beispiel: $R_u = 10, T = 10, n = 104$. Dann ist

$$\begin{array}{lcl} \overline{R}_x & = & 0.4 \\ \sigma(\overline{R}_x) & = & 1.02 \end{array} .$$

Wie lautet nun das Resultat für R_x und seine Fehler? Ist das Signal überhaupt signifikant? Innerhalb der Bayes'schen Statistik ist es naheliegend, für den Prior den folgenden Ansatz zu machen

$$f(R_x) = 0 \quad \text{für} \quad R_x < 0 \quad \text{und} \quad f(R_x) = \text{konst} \quad \text{für} \quad R_x \geq 0\,.$$

Darin wird zum Ausdruck gebracht, daß die Zählrate nicht negativ sein kann. Dann hat man für $R_x \geq 0$ mit einem Normierungsfaktor N_0 und Gleichung (6.27)

$$f(R_x|\overline{R}_x) = f(\overline{R}_x|R_x)f(R_x)N_0 = N(\overline{R}_x, \sigma^2) \cdot N_0$$
$$= \frac{\frac{1}{\sqrt{2\pi}\sigma} \cdot \exp\left(\frac{-(R_x-\overline{R}_x)^2}{2\sigma^2}\right)}{\frac{1}{\sqrt{2\pi}\sigma} \cdot \int_0^\infty \exp\left(\frac{-(R_x-\overline{R}_x)^2}{2\sigma^2}\right) \mathrm{d}R_x} = \frac{1}{0.65} \cdot \frac{1}{\sqrt{2\pi}\sigma} \cdot \exp\left(\frac{-(R_x-\overline{R}_x)^2}{2\sigma^2}\right) \quad (6.29)$$

für $R_x \geq 0$ und null für $R_x < 0$. Die Funktion $f(R_x|\overline{R}_x)$ ist in Abbildung 6.8 zu sehen.

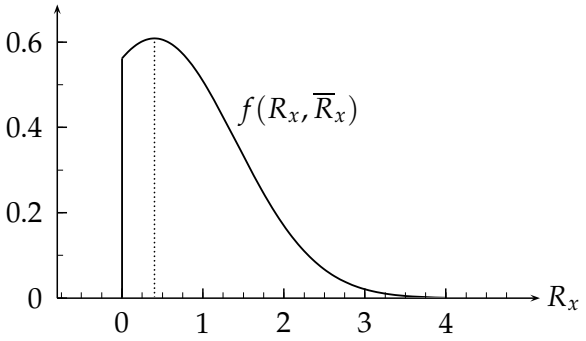


Abbildung 6.8: Die Wahrscheinlichkeit $f(R_x|\overline{R}_x)$ für die wahre mittlere Zählrate R_x, wenn die gemessene mittlere Zählrate $\overline{R}_x = 0.4$ ist.

Um eine obere Grenze R_o für R_x zu erhalten, die einer Standardabweichung (einseitig) entspricht, d.h. einer Wahrscheinlichkeit von $\alpha = 0.16$, hat man als Bedingung

$$\frac{1}{0.65\sqrt{2\pi}\sigma} \int_{R_o}^{\infty} \exp(\frac{-(R_x-\overline{R}_x)^2}{2\,\sigma^2})\mathrm{d}R_x = \alpha = 0.16\,.$$

Dies ist eine Gleichung für R_o, sie gibt, nach R_o aufgelöst, eine obere Grenze von $R_o = 1.69$. Das Resultat ist dann

$$R_x = 0.4^{+1.29}_{-0.4}$$

mit einer Wahrscheinlichkeit von $1 - \alpha = 84\%$, daß $\overline{R}_x$ zwischen diesen Grenzen liegt. Umgekehrt kann man mit 84% Konfidenz ausschließen, daß das Signal größer ist als 1.69, vorausgesetzt, man akzeptiert den Bayes'schen Ansatz. Es ist ja eine Gleichverteilung für den Prior angenommen worden; hätte man angenommen, daß $f(R_x) \sim R_x^2$ für $R_x > 0$ ist, so hätte man eine etwas andere obere Grenze erhalten.

Für das Problems eines kleines Signals mit Untergrund wurde von Feldman und Cousins[21] eine Methode entwickelt, die unphysikalische Grenzen vermeidet und einheitlich obere Grenzen und zweiseitige Intervalle behandelt (siehe auch Seite 145).

Beispiel 6.13 Katastrophenwahrscheinlichkeit.

Dieses Beispiel zeigt die Grenzen der klassischen Statistik im Vergleich zur Bayes'schen. Angenommen, 100 Raketenstarts in einer Reihe waren erfolgreich. Welches ist die Wahrscheinlichkeit eines Fehlstarts beim 101. Versuch bei sonst unveränderten äußeren Bedingungen? Die Antwort erfordert die Kenntnis der Wahrscheinlichkeit p eines Fehlstarts bei einem Versuch. Um eine obere Grenze p_o für die Wahrscheinlichkeit eines Fehlstarts mit 95% Konfidenz zu erhalten, fordert man, daß für diesen Wert von p die Wahrscheinlichkeit, bei 100 Versuchen null Fehlstarts zu erhalten, *mindestens* gleich $(100 - 95)\%$ =5% ist. Unter Verwendung der Binomialverteilung erhält man

$$P(100, 0) = p_o^0(1 - p_o)^{100} = \alpha = 0.05\,,$$

wobei $P(100, 0)$ die Binomialwahrscheinlichkeit für 0 Fehlstarts bei 100 Versuchen ist. Die obenstehende Gleichung liefert $100 \ln(1 - p_o) = \ln \alpha$ oder $p_o = 0.03$. Man hat eine obere Grenze von 3% (mit 95% Konfidenz) für die Wahrscheinlichkeit eines Fehlstarts beim nächsten Versuch erhalten.

Um eine Wahrscheinlichkeit für einen Fehlstart und nicht nur eine Grenze zu erhalten, benötigt man die Bayes'sche Statistik. Angenommen, der Prior von p wäre eine Gleichverteilung zwischen null und der 95% Grenze. Dann ist die Wahrscheinlichkeit p gegeben durch

$$p = \frac{1}{p_o} \int_0^{p_o} P(1, p)\, \mathrm{d}p = \frac{1}{p_o} \int_0^{p_o} e^{-p} p\, \mathrm{d}p = 1.5\% .$$

Hätte man stattdessen eine exponentiell fallende Wahrscheinlichkeitsdichte für den Prior angenommen, proportional zu $\exp(-p \cdot \ln(1/\alpha)/p_o)$, dann wäre das Resultat $p = 1\%$.

Als Literatur zur Bayes'schen Statistik sei genannt [57, 15, 83, 5].

6.9 Systematische Fehler

Systematische Fehler beeinflussen jeden Einzelwert einer Meßreihe in ähnlicher Weise. Sie können verursacht sein durch falsche Meßmethoden, wie zum Beispiel falsche Eichung oder fehlerhaftes Gerät, falsche Auswertemethoden, Nichtberücksichtigung von Untergrund, zeitliche Änderung der Meßbedingungen usw. Systematische Fehler können nicht durch eine bessere statistische Genauigkeit reduziert werden, und bei sehr kleinen statistischen Fehlern bestimmen sie die Genauigkeit der Messung.

Es gibt kein allgemeines Rezept zur Bestimmung der systematischen Fehler. Bewährt hat sich, ein und dieselbe Größe mit verschiedenen Methoden, zu verschiedenen Zeiten und mit verschiedener Reihenfolge der Einzelmessungen zu bestimmen. Man muß sich einen Eindruck von der Genauigkeit der Eichung der verschiedenen Instrumente und anderer Konstanten verschaffen, die in die Auswertung der Messungen eingehen. Manchmal kann man Erhaltungssätze oder allgemeine Gesetzmäßigkeiten benutzen, um die Genauigkeit zu überprüfen.

Das nächste Problem ist die Einbeziehung systematischer Fehler in die Fehlerrechnung. Wenn es viele systematische Fehlerquellen gibt, kann man unter Berufung auf den Zentralen Grenzwertsatz die Beiträge der einzelnen systematischen Fehler quadratisch addieren. Statistische und systematische Fehler sind statistisch unabhängig. Sie sollten getrennt aufgeführt werden, also zum Beispiel

$$x = 10.0 \pm 1.2(\text{stat.}) \pm 1.0(\text{syst.}) .$$

Wenn mehrere systematische Fehlerquellen da sind, gibt es weitere Komplikationen. Generell sind systematische Fehler stets korreliert, und dies muß sorgfältig beachtet werden.

Für eine formale Behandlung betrachtet man einen Vektor von Variablen x_i mit Mittelwerten μ_i und Standardabweichungen σ_i. Das Element i, j der Kovarianzmatrix ohne systematische Fehler ist (siehe Gleichung (4.43))

$$V_{ij} = \int (x_i - \mu_i)(x_j - \mu_j) f(\boldsymbol{x}) \mathrm{d}\boldsymbol{x} .$$

Sind systematische Fehler vorhanden, so können die Werte x_i systematisch um s_i verschoben sein. Das Matrixelement $V_{\text{ges},ij}$ für den gesamten Fehler *mit* Einschluß der systematischen Fehler ist dann

$$V_{\text{ges},ij} = \int (x_i - s_i - \mu_i)(x_j - s_j - \mu_j) f(\boldsymbol{x})\, \mathrm{d}\boldsymbol{x} .$$

Dies kann ausgedrückt werden durch

$$V_{\text{ges},ij} = V_{ij} + s_i s_j \int f(\boldsymbol{x})\,\mathrm{d}\boldsymbol{x} - s_i \int (x_j - \mu_j) f(\boldsymbol{x})\,\mathrm{d}\boldsymbol{x} - s_j \int (x_i - \mu_i) f(\boldsymbol{x})\,\mathrm{d}\boldsymbol{x}\,.$$

Die beiden letzten Integrale sind null, und deshalb folgt

$$V_{\text{ges},ij} = V_{ij} + s_i s_j\,. \tag{6.30}$$

Statistische und systematische Fehler sind unabhängig nach Voraussetzung, und deshalb kann man sie quadratisch addieren. Der Kovarianzterm $s_i s_j$ mit $i \neq j$ gibt ein Maß für die Größe von Korrelationen, die durch systematische Fehler hervorgerufen werden können.

Die Abschätzung von systematischen Fehlern erfolgt oft mit roher Gewalt. Dabei wird die Analyse (zum Beispiel Entfaltung) für jeden systematischen Fehlereinfluß wiederholt, wobei jeder systematische Fehlereinfluß (Index k) zu $\pm$ seiner Standardabweichung δ_k angenommen wird. Wenn sich dabei der Wert von x_i für den k-ten systematischen Fehlereinfluß um $\delta_k \cdot (r_k)_i$ ändert, ist das Element der Kovarianzmatrix der systematischen Fehler

$$V_{\text{syst},ij} = \sum_k \delta_k^2 \cdot (r_k)_i \cdot (r_k)_j\,.$$

Ein wichtiger Spezialfall ist ein *Normierungsfehler.* Er beeinflußt alle Meßwerte in derselben Weise, indem er sie mit dem Faktor $(1 \pm \epsilon)$ multipliziert, wobei ϵ die Standardabweichung des relativen Normierungsfehlers ist. Damit wird $s_i = \epsilon x_i$ in Gleichung (6.30), und damit das Element i, j der Kovarianzmatrix des Normierungsfehlers nach Gleichung (6.30)

$$V_{\text{syst},ij} = \epsilon^2 x_i x_j\,.$$

Es ist leicht zu sehen, daß für jeden einzelnen Beitrag der systematischen Fehler die Elemente der Kovarianzmatrix einer vollständigen Korrelation ($|\rho| = 1$) der Komponenten entsprechen, und die Normierungsfehler positiv korreliert sind ($\rho = +1$).

Eine ausführliche Behandlung systematischer Fehler findet sich auch bei [3].

Schätzung von Parametern mit systematischen Fehlern. Ein gängiges Verfahren wird hier für den einfachen Fall unkorrelierter normalverteilter Daten y_i mit Standardabweichung σ_i beschrieben. Ohne systematische Fehler würde man die Parameter bestimmen, indem man die Funktion S in Gleichung (6.10) minimiert. Wenn die Daten einen systematischen Fehler r_i haben, können sie um den Betrag ηr_i systematisch verschoben sein, wobei η einer standardisierten Gauß-Verteilung gehorcht. Eine solche mögliche Verschiebung wird durch einen zusätzlichen Summanden in der χ^2-Summe berücksichtigt. Man minimiert dann den Ausdruck

$$S = \sum_i^n \frac{(y_i - f(x_i, a) + \eta r_i)^2}{\sigma_i^2} + \eta^2\,,$$

der Gleichung (6.10) ersetzt.

Sind die Fehlergrenzen unsymmetrisch, so gibt es kein vollständig akzeptables Verfahren, sondern nur Rezepte. Ein Problem ist, dass die Wahrscheinlichkeitsdichte systematischer Fehlerquellen i.a. nicht bekannt ist. Hier ist die Lektüre einer Arbeit von R.Barlow [60] zu empfehlen. Er beschreibt die Vor- und Nachteile verschiedener Rezepte, mit asymmetrischen Fehlern umzugehen, und wie man mehrere systematische Fehler in diesem Fall kombiniert. Er kritisiert dabei auch die Sitte, im Fall mehrerer unsymmetrischer systematischer Fehler die Quadratsumme der positiven und negativen Fehler getrennt zu berechnen. Dies ist schlecht, weil es dem Zentralen Grenzwertsatz widerspricht. Ob im Fall unsymmetrischer Fehler der gemessene Wert y_i verschoben werden sollte, ist Gegenstand kontroverser Diskussionen.

7 Methode der kleinsten Quadrate

7.1 Einleitung

Die Geschichte der kleinsten Quadrate geht zurück zum Beginn des 19. Jahrhunderts, als Legendre, Gauß und Laplace diese Methode einführten und ihre Eigenschaften untersuchten. Die Methode der kleinsten Quadrate ist einfach zu handhaben und sehr effektiv, um Daten zu analysieren [47].

In diesem Kapitel werden direkte Meßwerte mit der Eigenschaft von Zufallsvariablen, *Daten* genannt, durchweg mit y_i bezeichnet. Eine Meßreihe besteht aus n Meßwerten y_i, $i = 1, 2, \ldots n$. Wegen der Meßfehler weichen die gemessenen Werte von den *wahren* Werten um einen Betrag ab, welcher durch die Standardabweichung σ bzw. die Varianz σ^2 beschrieben wird. Im Sinne der Statistik sind die y_i eine Stichprobe, welcher eine Wahrscheinlichkeitsdichte zugrundeliegt. Es sei weiterhin angenommen, daß eine gewisse funktionelle Beziehung, *Modell* genannt, für die *wahren* Werte existiert. Dieses Modell kann von zusätzlichen Variablen a_j, $j = 1, 2, \ldots p$ abhängen. Sie werden *Parameter* genannt; für sie gibt es keine direkten Messungen. Das Modell wird im allgemeinsten Fall als eine oder mehrere Gleichungen der Form

$$f(a_1, a_2, \ldots a_p, y_1, y_2, \ldots y_n) = 0 \tag{7.1}$$

dargestellt. Diese Gleichungen heißen *Bedingungen*. Das Modell mit den Gleichungen (7.1) kann benutzt werden, um Korrekturen Δy_i für die Meßwerte y_i zu finden, so daß die korrigierten Werte die Bedingungen (7.1) exakt erfüllen.

Das **Prinzip der kleinsten Quadrate** verlangt, daß die *Summe der Quadrate* der Korrekturen Δy_i (Differenzen zwischen Daten und Modell = Residuen) den kleinstmöglichen Wert annimmt. Im einfachsten Fall unkorrelierter Daten, die alle die gleiche Standardabweichung haben, kann das Prinzip der kleinsten Quadrate durch die Forderung

$$S = \sum_{i=1}^{n} \Delta y_i^2 = \text{Minimum} \tag{7.2}$$

ausgedrückt werden. Wegen der zusätzlichen Information, die aus den Bedingungsgleichungen (7.1) kommt, kann damit die Genauigkeit in der Kenntnis der Meßwerte verbessert werden. Man kann hieraus auch Werte für die nicht gemessenen Parameter unter allgemeinen Bedingungen ermitteln, und man kann dies als eine *indirekte* Messung der Parameter betrachten.

Eine einfache Anwendung ist der Fall von n wiederholten Messungen y_i der Variablen y. Die Bedingung der kleinsten Quadrate lautet

$$S = \sum_{i=1}^{n} (y - y_i)^2 = \text{Minimum} \ ,$$

sie wird erfüllt durch den Mittelwert der Meßdaten und ergibt $\hat{y} = \sum_{i=1}^{n} y_i / n$. Das Symbol $\hat{}$ wird benutzt, um den *Schätzwert* des wahren Mittelwertes y zu bezeichnen. Der Schätzwert $\hat{y}$ ist seinerseits eine Zufallsvariable mit einer Standardabweichung, die kleiner ist als diejenige der einzelnen Meßwerte.

Die Methode der kleinsten Quadrate hat einige optimale statistische Eigenschaften und führt darüber hinaus oft zu einfachen Lösungen. Andere ähnliche Vorschriften wie

$$\sum_{i=1}^{n} |\Delta y_i| = \text{Minimum} \qquad \text{oder} \qquad \max|\Delta y_i| = \text{Minimum}$$

führen im allgemeinen zu komplizierteren Lösungen als die Bedingung (7.2). Eine Lösung der Gleichung der kleinsten Quadrate wird üblicherweise in Matrixschreibweise formuliert, die sich auch gut zur Umsetzung auf einem Rechner eignet. Im allgemeinen Fall von Daten, beschrieben durch einen n-Vektor $\boldsymbol{y}$ mit voneinander verschiedenen Standardabweichungen und mit Korrelationen, welche durch die Kovarianzmatrix $\boldsymbol{V}$ beschrieben werden, lautet die Bedingung der kleinsten Quadrate in Matrixform

$$S = \boldsymbol{\Delta y}^T \boldsymbol{V}^{-1} \boldsymbol{\Delta y} \quad = \quad \text{Minimum}\,, \tag{7.3}$$

wobei $\boldsymbol{\Delta y}$ der Residuenvektor ist. Durch Spezialisierung erhält man Gleichung (7.2) zurück. Normalerweise hängt die quadratische Summe S von Parametern a_j ab, sodaß man eine Funktion $S(a_1, a_2, \ldots)$ zu betrachten hat. Einige optimale Eigenschaften der Methode folgen bereits ohne spezielle Annahmen über die *Form* der Wahrscheinlichkeitsdichte oder der Meßfehler. Im Spezialfall einer Gauß-Verteilung der Messungen läßt sich das Prinzip der kleinsten Quadrate der Gleichungen (7.2) und (7.3) aus der Maximum-Likelihood-Methode herleiten (Kapitel 6.3.2). Die Beziehung zwischen der Summe der Quadrate $S(a_1, a_2, \ldots)$ und der negativen Log-Likelihood-Funktion $F(a_1, a_2, \ldots)$ ist (bis auf eine additive Konstante, die ohne Einschränkung der Allgemeinheit zu null gewählt werden kann) gegeben durch

$$S(a_1, a_2, \ldots) = 2 \cdot F(a_1, a_2, \ldots)\,, \tag{7.4}$$

sie unterscheiden sich also um einen Faktor 2.

Die kleinsten Quadrate können auf eine große Klasse von Problemen angewandt werden. Ein häufiger Fall ist ein Modell der Form

$$y = f(x, a_1, a_2, \ldots a_p)\,, \tag{7.5}$$

d.h. die einzelnen Messungen y_i werden von einer Funktion *vorausgesagt*, die von den Parametern a_j sowie von einer Variablen x abhängt, die einen festen Wert x_i für die jeweilige Messung y_i besitzt; in diesem Fall gestattet die Methode die Bestimmung der Parameter. Eine andere Anwendung besteht in der Bestimmung einer näherungsweisen funktionellen Beziehung zwischen den Variablen x und y; in diesem Fall kann zum Beispiel eine Polynomdarstellung in x angesetzt werden. Eine weitere Anwendung besteht in einer Verbesserung der Genauigkeit der Daten. Die Bedingung der kleinsten Quadrate liefert weiterhin automatisch ein Kriterium dafür, ob die Meßdaten statistisch verträglich mit der modellmäßigen Parametrisierung sind.

7.2 Lineare kleinste Quadrate

7.2.1 Bestimmung von Parameterwerten

Dieses Kapitel handelt von der Bestimmung von Parameterwerten $\boldsymbol{a}$ aus Messungen anhand eines *linearen* Modells. Der Vektor $\boldsymbol{a}$ der Parameter hat p Elemente $a_1, a_2, \ldots a_p$. Diese Parameter stehen in Beziehung zu einem anderen Vektor $\boldsymbol{y}$ von n Zufallsvariablen mit Elementen $y_1, y_2, \ldots y_n$, den Meßwerten. Der Erwartungswert von y_i ist gegeben als Funktion $y = f(x_i, \boldsymbol{a})$ der Variablen x. Die Funktion $f(x, \boldsymbol{a})$ hängt *linear* von den p Parametern a_j ab und kann in der Form

$$y(x) = f(x, \boldsymbol{a}) = a_1 f_1(x) + a_2 f_2(x) + \ldots + a_p f_p(x) \tag{7.6}$$

geschrieben werden. Eine Reihe von n Messungen y_i, $i = 1, 2, \ldots n$ wird bei den x-Werten x_i, $i = 1, 2, \ldots n$ vorgenommen; nach Gleichung (7.6) ist der Erwartungswert jeder Einzelmessung y_i gegeben durch

$$E[y_i] = f(x_i, \bar{\boldsymbol{a}}) = \bar{y}_i\,, \tag{7.7}$$

wobei die Elemente von $\bar{\boldsymbol{a}}$ die wahren Werte des Parameters $\boldsymbol{a}$ sind. Die Variable x ist eine unabhängige Variable (ohne Fehler); sie kann ebenfalls ein Vektor sein.

Residuen r_i sind definiert als die Differenzen zwischen den Meßwerten y_i und den nach dem Modell erwarteten Werten $f(x_i, \boldsymbol{a})$ für beliebige Werte von $\boldsymbol{a}$,

$$r_i = y_i - f(x_i, \boldsymbol{a}) \qquad \text{Residuum .} \tag{7.8}$$

Die Residuen für $\boldsymbol{a} = \bar{\boldsymbol{a}}$ haben die Eigenschaften

$$E[r_i] = 0 \qquad E[r_i^2] = V[r_i] = \sigma_i^2 \ , \tag{7.9}$$

d.h. ihr Erwartungswert ist null und die Varianz σ_i^2. Es wird keine Annahme über die Wahrscheinlichkeitsdichte der Residuen gemacht, insbesondere ist es nicht nötig, daß sie gaußverteilt sind. Die einzigen Annahmen sind *Unverzerrtheit* und eine *endliche Varianz*. In diesem Kapitel wird der Einfachheit halber angenommen, daß alle Meßwerte dieselbe Varianz haben und unkorreliert sind; dann verschwinden die Kovarianzen.

7.2.2 Normalgleichungen im Fall gleicher Fehler

Alle Daten sollen die gleiche Varianz (und Standardabweichung) haben und unkorreliert sein. Dann erhält man in diesem und den folgenden Abschnitten verhältnismäßig einfache Ausdrücke.

Nach dem Prinzip der kleinsten Quadrate muß die Summe der Quadrate der Residuen in Bezug auf die Parameter $a_1,\ a_2, \ldots a_p$ minimiert werden. Die Summe S ist eine quadratische Funktion der Parameter a_j

$$S = \sum_{i=1}^{n} r_i^2 = \sum_{i=1}^{n} \left(y_i - a_1 f_1(x_i) - a_2 f_2(x_i) - \ldots - a_p f_p(x_i)\right)^2 \ . \tag{7.10}$$

Als Bedingung für das Minimum müssen alle partiellen Ableitungen nach den Parametern a_j verschwinden:

$$\begin{aligned}
\frac{\partial S}{\partial a_1} &= 2\sum_{i=1}^{n} f_1(x_i)\ \left(a_1 f_1(x_i) + a_2 f_2(x_i) + \ldots + a_p f_p(x_i) - y_i\right) = 0 \\
\frac{\partial S}{\partial a_2} &= 2\sum_{i=1}^{n} f_2(x_i)\ \left(a_1 f_1(x_i) + a_2 f_2(x_i) + \ldots + a_p f_p(x_i) - y_i\right) = 0 \\
\cdots \quad &\quad \cdots \\
\frac{\partial S}{\partial a_p} &= 2\sum_{i=1}^{n} f_p(x_i)\ \left(a_1 f_1(x_i) + a_2 f_2(x_i) + \ldots + a_p f_p(x_i) - y_i\right) = 0 \ .
\end{aligned}$$

Die Bedingung kann in Form der sogenannten *Normalgleichungen* geschrieben werden (alle Summen laufen über $i = 1,\ 2,\ \ldots n$)

$$\begin{array}{lcccl}
a_1 \sum f_1^2(x_i) & + & \ldots & + & a_p \sum f_1(x_i) f_p(x_i) \ = \ \sum y_i f_1(x_i) \\
a_1 \sum f_2(x_i) f_1(x_i) & + & \ldots & + & a_p \sum f_2(x_i) f_p(x_i) \ = \ \sum y_i f_2(x_i) \\
\ldots & & & & \\
\ldots & & & & \\
a_1 \sum f_p(x_i) f_1(x_i) & + & \ldots & + & a_p \sum f_p^2(x_i) \ = \ \sum y_i f_p(x_i) \ .
\end{array}$$

Die Schätzwerte von $a_1,\ a_2, \ldots a_p$ nach kleinsten Quadraten folgen als die Lösung dieser Normalgleichungen.

7.2.3 Matrixschreibweise

Die Matrixschreibweise und Matrixalgebra vereinfachen die Formulierung wesentlich. Die $n \times p$ Werte $f_j(x_i)$ können als Elemente einer $n \times p$ Matrix $\boldsymbol{A}$ aufgefaßt werden, mit n Zeilen und p Spalten. Die p Parameter a_j können als Elemente eines p-Vektors $\boldsymbol{a}$ (oder einer $p \times 1$ Matrix) aufgefaßt werden.

$$\boldsymbol{A} = \begin{pmatrix} f_1(x_1) & f_2(x_1) & \dots & f_p(x_1) \\ f_1(x_2) & f_2(x_2) & \dots & f_p(x_2) \\ \dots & & & \\ \dots & & & \\ \dots & & & \\ \dots & & & \\ f_1(x_n) & f_2(x_n) & \dots & f_p(x_n) \end{pmatrix} \tag{7.11}$$

$$\boldsymbol{a} = \begin{pmatrix} a_1 \\ a_2 \\ \cdot \\ \cdot \\ \cdot \\ a_p \end{pmatrix} \qquad \boldsymbol{y} = \begin{pmatrix} y_1 \\ y_2 \\ \cdot \\ \cdot \\ \cdot \\ \cdot \\ y_n \end{pmatrix} \tag{7.12}$$

Das Produkt $\boldsymbol{Aa}$ ist der n-Vektor der Erwartungswerte der n Messungen $y_1, y_2, \dots y_n$, die als Elemente eines n-Vektors $\boldsymbol{y}$ angesehen werden können. Mit der Definition des n-Vektors der Residuen $\boldsymbol{r}$ (Gleichung (7.8)) wird

$$\boldsymbol{r} = \boldsymbol{y} - \boldsymbol{Aa}\,. \tag{7.13}$$

Der Ausdruck (7.10) kann dann in Matrixschreibweise geschrieben werden

$$S = \boldsymbol{r}^T\boldsymbol{r} = (\boldsymbol{y} - \boldsymbol{Aa})^T(\boldsymbol{y} - \boldsymbol{Aa}) = \boldsymbol{y}^T\boldsymbol{y} - 2\,\boldsymbol{a}^T\boldsymbol{A}^T\boldsymbol{y} + \boldsymbol{a}^T\boldsymbol{A}^T\boldsymbol{Aa} \tag{7.14}$$

(der Index T bezeichnet die transponierte Matrix bzw. den transponierten Vektor). Die Bedingung für das Minimum von S lautet

$$-2\,\boldsymbol{A}^T\boldsymbol{y} + 2\,\boldsymbol{A}^T\boldsymbol{A}\hat{\boldsymbol{a}} = 0\,, \tag{7.15}$$

oder in der Matrixform der Normalgleichungen

$$(\boldsymbol{A}^T\boldsymbol{A})\hat{\boldsymbol{a}} = \boldsymbol{A}^T\boldsymbol{y}\,. \tag{7.16}$$

Hierbei ist $\hat{\boldsymbol{a}}$ der Vektor der Schätzwerte der Parameter. Das Produkt $\boldsymbol{A}^T\boldsymbol{A}$ ist die symmetrische $p \times p$ Matrix

$$\boldsymbol{A}^T\boldsymbol{A} = \begin{pmatrix} \sum f_1^2(x_i) & \sum f_1(x_i)\,f_2(x_i) & \dots & \sum f_1(x_i)\,f_p(x_i) \\ \sum f_2(x_i)\,f_1(x_i) & \sum f_2^2(x_i) & \dots & \sum f_2(x_i)\,f_p(x_i) \\ \dots & & & \\ \dots & & & \\ \dots & & & \\ \sum f_p(x_i)\,f_1(x_i) & \sum f_p(x_i)\,f_2(x_i) & \dots & \sum f_p^2(x_i) \end{pmatrix} \tag{7.17}$$

und das Produkt $\boldsymbol{A}^T\boldsymbol{y}$ ist ein p-Vektor (alle Summen laufen über $i = 1, 2, \ldots n$)

$$\boldsymbol{A}^T\boldsymbol{y} = \begin{pmatrix} \sum y_i f_1(x_i) \\ \sum y_i f_2(x_i) \\ \cdots \\ \cdots \\ \cdots \\ \sum y_i f_p(x_i) \end{pmatrix} . \tag{7.18}$$

Die Lösung der Gleichung (7.16)

$$\boxed{\hat{\boldsymbol{a}} = (\boldsymbol{A}^T\boldsymbol{A})^{-1}\boldsymbol{A}^T\boldsymbol{y}} \tag{7.19}$$

liefert die beste Schätzung von $\hat{\boldsymbol{a}}$ und kann mit den Standardverfahren der Matrixalgebra berechnet werden.

7.2.4 Kovarianzmatrix der Parameter

Gleichung (7.19) zeigt, daß der Schätzwert $\hat{\boldsymbol{a}}$ des Parametervektors aus einer linearen Transformation des Vektors $\boldsymbol{y}$ der Meßwerte hervorgeht. Deshalb kann die Kovarianzmatrix des Parametervektors mit den Standardmethoden der Fehlerfortpflanzung aus der Kovarianzmatrix des Vektors $\boldsymbol{y}$ berechnet werden. Die Kovarianzmatrix $\boldsymbol{V}[\boldsymbol{y}]$ ist die quadratische $n \times n$ Matrix

$$\boldsymbol{V}[\boldsymbol{y}] = \begin{pmatrix} \mathrm{var}(y_1) & \mathrm{cov}(y_1, y_2) & \ldots & \mathrm{cov}(y_1, y_n) \\ \mathrm{cov}(y_2, y_1) & \mathrm{var}(y_2) & \ldots & \mathrm{cov}(y_2, y_n) \\ \ldots & & & \\ \ldots & & & \\ \mathrm{cov}(y_n, y_1) & \mathrm{cov}(y_n, y_2) & \ldots & \mathrm{var}(y_n) \end{pmatrix} , \tag{7.20}$$

wobei $\mathrm{var}(y_i)$ die Varianz von y_i ist mit der üblichen Schreibweise σ_i^2; weiterhin ist $\mathrm{cov}(y_i, y_k)$ die Kovarianz von y_i und y_k, geschrieben σ_{ik}. Im vorliegenden Fall ist die Kovarianzmatrix eine Diagonalmatrix

$$\boldsymbol{V}[\boldsymbol{y}] = \sigma^2\boldsymbol{I}_n = \begin{pmatrix} \sigma^2 & 0 & 0 & \ldots & 0 \\ 0 & \sigma^2 & 0 & \ldots & 0 \\ & & & & \\ .. & & & & \\ 0 & 0 & 0 & \ldots & \sigma^2 \end{pmatrix} , \tag{7.21}$$

wobei $\boldsymbol{I}_n$ eine $n \times n$ Einheitsmatrix ist, da in $\boldsymbol{V}[\boldsymbol{y}]$ alle Varianzen gleich σ^2 und alle Kovarianzen null sind. Für eine lineare Beziehung $\hat{\boldsymbol{a}} = \boldsymbol{B}\boldsymbol{y}$ ist die Kovarianzmatrix für $\hat{\boldsymbol{a}}$ gegeben durch die Standardformel der Fehlerfortpflanzung (Gleichung (4.50))

$$\boldsymbol{V}[\hat{\boldsymbol{a}}] = \boldsymbol{B}\boldsymbol{V}[\boldsymbol{y}]\boldsymbol{B}^T . \tag{7.22}$$

Setzt man für $\boldsymbol{B}$ den Ausdruck aus Gleichung (7.19) ein, (d.h. $\boldsymbol{B} = (\boldsymbol{A}^T\boldsymbol{A})^{-1}\boldsymbol{A}^T$), so erhält man die Matrix

$$\boldsymbol{V}[\hat{\boldsymbol{a}}] = (\boldsymbol{A}^T\boldsymbol{A})^{-1}\boldsymbol{A}^T\boldsymbol{V}[\boldsymbol{y}]\boldsymbol{A}(\boldsymbol{A}^T\boldsymbol{A})^{-1} .$$

Für $\boldsymbol{V}[\boldsymbol{y}] = \sigma^2\boldsymbol{I}_n$ vereinfacht sich dies zur endgültigen Formel

$$\boxed{\boldsymbol{V}[\hat{\boldsymbol{a}}] = \sigma^2\,(\boldsymbol{A}^T\boldsymbol{A})^{-1}} . \tag{7.23}$$

Die Kovarianzmatrix der besten Schätzung der Parameterwerte $\hat{\boldsymbol{a}}$ ist also einfach die inverse Matrix der Produktmatrix $(\boldsymbol{A}^T\boldsymbol{A})$ in den Normalgleichungen (7.16) mal dem Skalar σ^2. Man beachte, daß $\boldsymbol{V}[\hat{\boldsymbol{a}}]$ nur von der festen Matrix $\boldsymbol{A}$ abhängt und von σ^2, aber nicht von den Werten der Residuen. Wichtig ist auch, daß die Varianz σ^2 keinen Einfluß auf die Parameterwerte hat, wenn σ^2 für alle Meßwerte gleich ist.

7.2.5 Quadratsumme der Residuen

Die Summe $\hat{S}$ der Quadrate der Residuen im Minimum von S folgt, indem man die Lösung $\hat{\boldsymbol{a}}$ aus Gleichung (7.19) in Gleichung (7.14) einsetzt

$$\hat{S} = \boldsymbol{y}^T\boldsymbol{y} - 2\hat{\boldsymbol{a}}^T\boldsymbol{A}^T\boldsymbol{y} + \hat{\boldsymbol{a}}^T\boldsymbol{A}^T\boldsymbol{A}\left(\boldsymbol{A}^T\boldsymbol{A}\right)^{-1}\boldsymbol{A}^T\boldsymbol{y} = \boldsymbol{y}^T\boldsymbol{y} - \hat{\boldsymbol{a}}^T\boldsymbol{A}^T\boldsymbol{y}\,. \tag{7.24}$$

Der Ausdruck zeigt, daß die Summe der Quadrate $\boldsymbol{y}^T\boldsymbol{y}$ vermindert wird durch das Skalarprodukt von $\hat{\boldsymbol{a}}$ und $\boldsymbol{A}^T\boldsymbol{y}$. Die Summe der Residuenquadrate kann also berechnet werden, ohne die einzelnen Residuen r_i zu berechnen. In der Praxis kann dies jedoch zu großen Rundungsfehlern führen (Differenz zweier großer Zahlen), und es ist deshalb genauer, $\hat{S}$ aus den einzelnen Werten r_i zu berechnen. Der Erwartungswert $E[\hat{S}]$ ist

$$\boxed{E[\hat{S}] = \sigma^2(n-p)}\,. \tag{7.25}$$

Für einen Beweis siehe Abschnitt 7.3.3.
Wenn die Varianz σ^2 der Meßdaten $\boldsymbol{y}$ nicht bekannt ist, so erhält man aus $\hat{S}$ nach Gleichung (7.25) den Schätzwert

$$\hat{\sigma}^2 = \hat{S}/(n-p). \tag{7.26}$$

Die Größe $\hat{\sigma}^2$ ist eine Zufallsvariable. Für große Werte von $(n-p)$ ist dies eine gute Schätzung.

7.2.6 Korrektur der Datenwerte

Nach der Berechnung der Parameter mit linearen kleinsten Quadraten können Werte der Funktion $f(x)$ für beliebige x bestimmt werden durch

$$\hat{y}(x) = f(x,\hat{\boldsymbol{a}}) = \sum_{j=1}^{p}\hat{a}_j f_j(x)\,. \tag{7.27}$$

Speziell für die Werte x_i, die zu den Meßwerten y_i gehören, sind die korrigierten Datenpunkte gegeben durch $\hat{\boldsymbol{y}} = \boldsymbol{A}\hat{\boldsymbol{a}}$. Da $\hat{\boldsymbol{y}}$ eine lineare Funktion der Parameter $\hat{\boldsymbol{a}}$ ist, welche die Kovarianzmatrix $\boldsymbol{V}[\hat{\boldsymbol{a}}]$ besitzen, liefert die Fehlerfortpflanzung die Kovarianzmatrix

$$\boldsymbol{V}[\hat{\boldsymbol{y}}] = \boldsymbol{A}\boldsymbol{V}[\hat{\boldsymbol{a}}]\boldsymbol{A}^T = \sigma^2\boldsymbol{A}(\boldsymbol{A}^T\boldsymbol{A})^{-1}\boldsymbol{A}^T \tag{7.28}$$

des Vektors $\hat{\boldsymbol{y}}$. Die Größe der Differenzen zwischen Messung y_i und dem berechneten Wert $\hat{y}_i$ sollte kontrolliert werden. Geeignet zur Kontrolle ist insbesondere die normierte Korrektur, die am Ende von Abschnitt 7.7.2 behandelt wird.

7.3 Lösungseigenschaften

Schätzwerte, welche mit der Methode der kleinsten Quadrate gewonnen wurden, haben wichtige statistische Eigenschaften, die zusammen mit dem relativ einfachen numerischen Formalismus den weiten Anwendungsbereich der Methode erklären. Die folgenden Aussagen sind beschränkt auf lineare Probleme. Die Annahmen in Gleichung (7.9) in Kapitel 7.2.1 über die Eigenschaften der Daten waren

1. die Daten sind unverzerrt (erwartungstreu), und
2. alle Varianzen sind gleich,

also in Gleichungsform

$$E[\boldsymbol{y} - \boldsymbol{A}\bar{\boldsymbol{a}}] = 0 \quad \text{oder} \quad E[\boldsymbol{y}] = \boldsymbol{A}\bar{\boldsymbol{a}} \qquad \boldsymbol{V}[\boldsymbol{y} - \boldsymbol{A}\bar{\boldsymbol{a}}] = \sigma^2 \boldsymbol{I}_n \,. \tag{7.29}$$

Im folgenden wird gezeigt, daß

1. Schätzwerte nach kleinsten Quadraten unverzerrt (erwartungstreu) sind, und
2. den kleinsten Fehler von allen linearen Schätzwerten haben.

Für den Beweis muß *keine* Annahme über die Wahrscheinlichkeitsdichte der Residuen gemacht werden, insbesondere ist die Annahme einer Gauß-Verteilung nicht notwendig.

7.3.1 Erwartungstreue

Schätzungen mit kleinsten Quadraten nach Gleichung (7.19) sind erwartungstreue Schätzungen der wahren Werte $\bar{\boldsymbol{a}}$, d.h.

$$E[\hat{\boldsymbol{a}}] = \bar{\boldsymbol{a}}. \tag{7.30}$$

Zum Beweis benutzt man die Gleichungen (7.29) und (7.19)

$$E[\hat{\boldsymbol{a}}] = (\boldsymbol{A}^T\boldsymbol{A})^{-1}\boldsymbol{A}^T E[\boldsymbol{y}] = (\boldsymbol{A}^T\boldsymbol{A})^{-1}\boldsymbol{A}^T\boldsymbol{A}\bar{\boldsymbol{a}} = \bar{\boldsymbol{a}}\,. \tag{7.31}$$

Die Schätzwerte sind also unverzerrt, aber nur dann, wenn die Daten unverzerrt sind und das Modell korrekt ist. Wenn die Daten verzerrt sind, werden es im allgemeinen auch die Parameterschätzungen sein. Ebenso muß das Modell korrekt sein; wenn zum Beispiel eine Geradenbeziehung an Daten angepaßt wird, die tatsächlich auf einer Parabel liegen, so werden die Werte für die Steigung verzerrt sein und außerdem von dem Anpassungsbereich abhängen. Eine andere Ursache für Verzerrung kann ein einzelner falsch gemessener Datenpunkt sein.

7.3.2 Das Gauß-Markoff-Theorem

Das *Gauß-Markoff-Theorem* macht die Aussage, daß von allen Klassen von Schätzwerten $\boldsymbol{a}^*$, die

1. erwartungstreu sind, und die
2. lineare Funktionen der Daten sind,

die Schätzwerte nach kleinsten Quadraten $\hat{\boldsymbol{a}}$ die Eigenschaft

$$V[\hat{\boldsymbol{a}}]_{jj} \leq V[\boldsymbol{a}^*]_{jj} \quad \text{für alle } j \tag{7.32}$$

haben, d.h. die Schätzung nach kleinsten Quadraten hat den *kleinstmöglichen Fehler*.

Beweis: Der Schätzwert $\hat{\boldsymbol{a}}$ nach Gleichung (7.19) hat nach Gleichung (7.23) die Varianz $\boldsymbol{V}[\hat{\boldsymbol{a}}] = \sigma^2\,(\boldsymbol{A}^T\boldsymbol{A})^{-1}$. Nun betrachtet man einen anderen linearen und *erwartungstreuen* Schätzwert $\boldsymbol{a}^* = \boldsymbol{U}\boldsymbol{y}$ mit einer Matrix $\boldsymbol{U}$. Der Erwartungswert und die Varianz des Schätzwertes $\boldsymbol{a}^*$ sind, mit Gleichung (7.29)

$$E[\boldsymbol{a}^*] = \boldsymbol{U}E[\boldsymbol{y}] = \boldsymbol{U}\boldsymbol{A}\bar{\boldsymbol{a}} = \bar{\boldsymbol{a}} \qquad \boldsymbol{V}[\boldsymbol{a}^*] = \sigma^2\boldsymbol{U}\boldsymbol{U}^T \tag{7.33}$$

und folglich $\boldsymbol{U}\boldsymbol{A} = \boldsymbol{I}_p$, wobei $\boldsymbol{I}_p$ eine $p \times p$ Einheitsmatrix ist. Wegen $\boldsymbol{U}\boldsymbol{A} = \boldsymbol{I}_p$ kann das Produkt $\boldsymbol{U}\boldsymbol{U}^T$ mit der Abkürzung $\boldsymbol{C} = \boldsymbol{A}^T\boldsymbol{A}$ in der Form

$$\boldsymbol{U}\boldsymbol{U}^T = \boldsymbol{C}^{-1} + (\boldsymbol{U} - \boldsymbol{C}^{-1}\boldsymbol{A}^T)(\boldsymbol{U} - \boldsymbol{C}^{-1}\boldsymbol{A}^T)^T \tag{7.34}$$

geschrieben werden. Damit wird die Kovarianzmatrix mit (7.23)

$$\boldsymbol{V}[\boldsymbol{a}^*] = \boldsymbol{V}[\hat{\boldsymbol{a}}] + \sigma^2(\boldsymbol{U} - \boldsymbol{C}^{-1}\boldsymbol{A}^T)(\boldsymbol{U} - \boldsymbol{C}^{-1}\boldsymbol{A}^T)^T\,. \tag{7.35}$$

Das Produkt auf der rechten Seite hat Diagonalelemente ≥ 0 (sie sind null nur für die kleinste-Quadrate Lösung $\boldsymbol{U} = \boldsymbol{C}^{-1}\boldsymbol{A}^T = (\boldsymbol{A}^T\boldsymbol{A})^{-1}\boldsymbol{A}^T$), und es folgt die Ungleichung (7.32).

7.3.3 Erwartungswert der Summe der Residuenquadrate

Der Erwartungswert $\hat{S}$ der Summe der Residuenquadrate ist nach Gleichung (7.25) gegeben durch $E[\hat{S}] = \sigma^2(n-p)$. Diese Eigenschaft folgt aus der Definition von $\hat{S}$ und Gleichung (7.16)

$$\hat{S} = (\boldsymbol{y} - \boldsymbol{A}\hat{\boldsymbol{a}})^T(\boldsymbol{y} - \boldsymbol{A}\hat{\boldsymbol{a}}) = \boldsymbol{y}^T\boldsymbol{y} - \boldsymbol{y}^T\boldsymbol{A}\hat{\boldsymbol{a}}\,. \tag{7.36}$$

Zur Berechnung des Erwartungswertes wird die Gleichung durch $\bar{\boldsymbol{a}}$ anstatt $\hat{\boldsymbol{a}}$ ausgedrückt (Matrix $\boldsymbol{C} = \boldsymbol{A}^T\boldsymbol{A}$)

$$\hat{S} = (\boldsymbol{y} - \boldsymbol{A}\bar{\boldsymbol{a}})^T(\boldsymbol{I}_n - \boldsymbol{A}\boldsymbol{C}^{-1}\boldsymbol{A}^T)(\boldsymbol{y} - \boldsymbol{A}\bar{\boldsymbol{a}})\,. \tag{7.37}$$

Man prüft diese Identität, indem man die rechte Seite ausmultipliziert und beachtet, daß die Beziehung $\boldsymbol{A}^T\boldsymbol{A}\boldsymbol{C}^{-1} = \boldsymbol{C}^{-1}\boldsymbol{A}^T\boldsymbol{A} = \boldsymbol{I}_p$ gilt; mit Gleichung (7.16) erhält man den Ausdruck aus Gleichung (7.36) zurück. Führt man einen Vektor $\boldsymbol{z}$ und eine Matrix $\boldsymbol{U}$ ein gemäß

$$\boldsymbol{z} = \boldsymbol{y} - \boldsymbol{A}\bar{\boldsymbol{a}} \qquad\qquad \boldsymbol{U} = \boldsymbol{I}_n - \boldsymbol{A}\boldsymbol{C}^{-1}\boldsymbol{A}^T\,, \tag{7.38}$$

dann kann die Summe $\hat{S}$ ausgedrückt werden als $\hat{S} = \boldsymbol{z}^T\boldsymbol{U}\boldsymbol{z}$. Nach Definition (Gleichung (7.29)) sind Erwartungswert und Varianz von $\boldsymbol{z}$

$$E[\boldsymbol{z}] = E[\boldsymbol{y} - \boldsymbol{A}\bar{\boldsymbol{a}}] = 0 \qquad\qquad \boldsymbol{V}[\boldsymbol{z}] = \boldsymbol{V}[\boldsymbol{y} - \boldsymbol{A}\bar{\boldsymbol{a}}] = \sigma^2\boldsymbol{I}_n. \tag{7.39}$$

Für die Komponenten z_i, z_k bedeutet die letzte Gleichung $V[z_i] = E[z_i^2] = \sigma^2$ und $E[z_i z_k] = 0$. Für $\hat{S}$ erhält man

$$\hat{S} = \sum_i \sum_k U_{ik}\, z_i\, z_k. \tag{7.40}$$

Wenn man nun den Erwartungswert von $\hat{S}$ bildet, so erhält man, da alle Terme mit $i \neq k$ null sind,

$$E[\hat{S}] = \sum_i U_{ii}\, E[z_i^2] = \sigma^2 \sum_i U_{ii} = \sigma^2\, \mathrm{Spur}(\boldsymbol{U}) \tag{7.41}$$

(die Spur einer quadratischen Matrix ist die Summe ihrer Diagonalelemente). Die Spur von $\boldsymbol{U}$ ist

$$\begin{aligned} \mathrm{Spur}(\boldsymbol{U}) &= \mathrm{Spur}(\boldsymbol{I}_n - \boldsymbol{A}\boldsymbol{C}^{-1}\boldsymbol{A}^T) = \mathrm{Spur}(\boldsymbol{I}_n) - \mathrm{Spur}(\boldsymbol{A}\boldsymbol{C}^{-1}\boldsymbol{A}^T) \\ &= \mathrm{Spur}(\boldsymbol{I}_n) - \mathrm{Spur}(\boldsymbol{C}^{-1}\boldsymbol{A}^T\boldsymbol{A}) \\ &= \mathrm{Spur}(\boldsymbol{I}_n) - \mathrm{Spur}(\boldsymbol{I}_p) = n - p. \end{aligned} \tag{7.42}$$

Also ist der Erwartungswert von $\hat{S}$

$$E[\hat{S}] = \sigma^2(n-p). \tag{7.43}$$

Gauß-verteilte Meßfehler. Im Falle gauß-verteilter Fehler ist die Wahrscheinlichkeitsdichte von $\hat{S}$ bekannt. Das Verhältnis $\hat{S}/\sigma^2$ hat eine χ^2-Verteilung mit $(n-p)$ Freiheitsgraden. Damit kann man einen χ^2-Test machen, um zu prüfen, ob das Modell und die Daten statistisch verträglich sind. In der Praxis kann man diesen Test näherungsweise auch dann verwenden, wenn die Wahrscheinlichkeitsdichte etwas von einer Gauß-Verteilung abweicht.

7.4 Der allgemeine Fall unterschiedlicher Fehler

7.4.1 Die Gewichtsmatrix

Wenn die einzelnen Datenpunkte *verschiedene* Genauigkeit haben, kann der Formalismus von Kapitel 7.2 nicht unverändert benutzt werden. In diesem allgemeinen Fall gilt $\boldsymbol{V}[y_i] = \sigma_i^2$, d.h. jeder Meßpunkt y_i hat eine Varianz σ_i^2, die für jeden Meßpunkt verschieden sein kann. Wenn die einzelnen Datenpunkte *statistisch unabhängig* sind, dann sind alle Kovarianzen $\mathrm{cov}(y_i, y_k)$ null, und die Kovarianzmatrix ist

$$\boldsymbol{V}[\boldsymbol{y}] = \begin{pmatrix} \sigma_1^2 & 0 & 0 & \dots & 0 \\ 0 & \sigma_2^2 & 0 & \dots & 0 \\ & \dots & & & \\ 0 & 0 & 0 & \dots & \sigma_n^2 \end{pmatrix} . \tag{7.44}$$

Nur die Diagonalelemente sind also von null verschieden. Der Ausdruck für die Summe der Residuenquadrate S (siehe Gleichung (7.10)) lautet nunmehr

$$S = \sum_i \frac{r_i^2}{\sigma_i^2} = \text{Minimum} . \tag{7.45}$$

Um diese Forderung in Matrixschreibweise zu formulieren, führt man die Gewichtsmatrix $\boldsymbol{W}(\boldsymbol{y})$ ein als inverse Matrix der Kovarianzmatrix $\boldsymbol{V}[\boldsymbol{y}]$,

$$\boldsymbol{W}(\boldsymbol{y}) = \boldsymbol{V}[\boldsymbol{y}]^{-1} = \begin{pmatrix} 1/\sigma_1^2 & 0 & 0 & \dots & 0 \\ 0 & 1/\sigma_2^2 & 0 & \dots & 0 \\ & \dots & & & \\ & \dots & & & \\ 0 & 0 & 0 & \dots & 1/\sigma_n^2 \end{pmatrix} , \tag{7.46}$$

d.h. die Diagonalelemente sind gleich den inversen Varianzen. Der Ausdruck für S lautet dann:

$$S = \boldsymbol{r}^T \boldsymbol{W}(\boldsymbol{y}) \boldsymbol{r} = (\boldsymbol{y} - \boldsymbol{A}\boldsymbol{a})^T \boldsymbol{W}(\boldsymbol{y}) \ (\boldsymbol{y} - \boldsymbol{A}\boldsymbol{a}). \tag{7.47}$$

Dieser Ausdruck hat allgemeinere Gültigkeit und gilt auch im allgemeinen Fall korrelierter Daten; dann sind die Matrizen $\boldsymbol{V}[\boldsymbol{y}]$ und $\boldsymbol{W}(\boldsymbol{y}) = \boldsymbol{V}[\boldsymbol{y}]^{-1}$ nicht diagonal. In diesem Fall kann man mit einer orthogonalen Transformation wieder auf Diagonalform kommen und damit die Äquivalenz zu Gleichung (7.47) zeigen. Zum Beweis macht man eine lineare Transformation des Datenvektors $\boldsymbol{y}$ durch eine $n \times n$ Matrix $\boldsymbol{U}^T$ in einen neuen Vektor $\boldsymbol{z}$,

$$\boldsymbol{z} = \boldsymbol{U}^T \boldsymbol{y} . \tag{7.48}$$

Nach dem Gesetz der Fehlerfortpflanzung ist die Kovarianzmatrix des neuen Vektors $\boldsymbol{z}$

$$\boldsymbol{V}[\boldsymbol{z}] = \boldsymbol{U}^T \boldsymbol{V}[\boldsymbol{y}] \boldsymbol{U} . \tag{7.49}$$

Im Kapitel 3.8 über lineare Algebra wird gezeigt, daß für jede symmetrische Matrix $\boldsymbol{V}[\boldsymbol{y}]$ eine orthogonale Matrix $\boldsymbol{U}$ existiert, welche $\boldsymbol{V}[\boldsymbol{y}]$ in eine Diagonalmatrix $\boldsymbol{V}[\boldsymbol{z}]$ transformiert. Dann ist auch die inverse Matrix $\boldsymbol{W}(\boldsymbol{z})$ diagonal, und die Gleichung für S ist

$$S = (\boldsymbol{z} - \boldsymbol{U}^T \boldsymbol{A}\boldsymbol{a})^T \boldsymbol{W}(\boldsymbol{z}) (\boldsymbol{z} - \boldsymbol{U}^T \boldsymbol{A}\boldsymbol{a}) . \tag{7.50}$$

Diese Gleichung ist identisch mit Gleichung (7.47)

$$\begin{aligned} S &= (\boldsymbol{U}^T \boldsymbol{y} - \boldsymbol{U}^T \boldsymbol{A}\boldsymbol{a})^T \boldsymbol{W}(\boldsymbol{z}) (\boldsymbol{U}^T \boldsymbol{y} - \boldsymbol{U}^T \boldsymbol{A}\boldsymbol{a}) \\ &= (\boldsymbol{y} - \boldsymbol{A}\boldsymbol{a})^T \boldsymbol{U} \boldsymbol{W}(\boldsymbol{z}) \boldsymbol{U}^T (\boldsymbol{y} - \boldsymbol{A}\boldsymbol{a}). \end{aligned} \tag{7.51}$$

Dies sieht man, indem man Gleichung (7.49) von links mit $\boldsymbol{U}\boldsymbol{W}(\boldsymbol{z})$ und von rechts mit $\boldsymbol{U}^T$ multipliziert und beachtet, daß $\boldsymbol{U}^T\boldsymbol{U} = \boldsymbol{U}\boldsymbol{U}^T = \boldsymbol{I}_n$ und $\boldsymbol{W} = \boldsymbol{W}^T$ ist. Man erhält dann

$$\boldsymbol{U}\boldsymbol{W}(\boldsymbol{z})\boldsymbol{U}^T = \boldsymbol{V}[\boldsymbol{y}]^{-1} = \boldsymbol{W}(\boldsymbol{y})\,, \tag{7.52}$$

d.h. Gleichungen (7.47) und (7.50) sind identisch.

7.4.2 Lösung im Fall der allgemeinen Kovarianzmatrix

Die Summe der Quadrate der gewichteten Residuen

$$S = \boldsymbol{r}^T\boldsymbol{W}(\boldsymbol{y})\boldsymbol{r} = (\boldsymbol{y} - \boldsymbol{A}\boldsymbol{a})^T\boldsymbol{W}(\boldsymbol{y})(\boldsymbol{y} - \boldsymbol{A}\boldsymbol{a}) \tag{7.53}$$

muß nun bezüglich der Parameter minimiert werden. Nach den Regeln lassen sich nun Gleichungen für $\hat{\boldsymbol{a}}$ und die Kovarianzmatrix $\boldsymbol{V}[\hat{\boldsymbol{a}}]$ herleiten ($\boldsymbol{W}(\boldsymbol{y})$ wird jetzt mit $\boldsymbol{W}$ bezeichnet).

$$\boxed{\begin{array}{rcl} \hat{\boldsymbol{a}} & = & (\boldsymbol{A}^T\boldsymbol{W}\boldsymbol{A})^{-1}\boldsymbol{A}^T\boldsymbol{W}\boldsymbol{y} \\ \boldsymbol{V}[\hat{\boldsymbol{a}}] & = & (\boldsymbol{A}^T\boldsymbol{W}\boldsymbol{A})^{-1} \end{array}} \tag{7.54}$$

(der Faktor σ^2 in Gleichung (7.23) ist bereits im Matrix-Produkt enthalten). Die Summe der Residuenquadrate für $\boldsymbol{a} = \hat{\boldsymbol{a}}$ hat mit Gleichung (7.54) die Form

$$\hat{S} = \boldsymbol{y}^T\boldsymbol{W}\boldsymbol{y} - \hat{\boldsymbol{a}}^T\boldsymbol{A}^T\boldsymbol{W}\boldsymbol{y} \tag{7.55}$$

und den Erwartungswert

$$\boxed{E[\hat{S}] = n - p}\,. \tag{7.56}$$

Die Kovarianzmatrix der korrigierten Datenpunkte ist (vgl. Gleichung (7.28))

$$\boldsymbol{V}[\hat{\boldsymbol{y}}] = \boldsymbol{A}(\boldsymbol{A}^T\boldsymbol{W}\boldsymbol{A})^{-1}\boldsymbol{A}^T\,. \tag{7.57}$$

7.5 Kleinste Quadrate in der Praxis

7.5.1 Normalgleichungen für unkorrelierte Daten

Normalerweise sind direkte Meßwerte als Eingabe für kleinste Quadrate *unkorreliert*, d.h. die Kovarianzmatrix $\boldsymbol{V}[\boldsymbol{y}]$ und die Gewichtsmatrix $\boldsymbol{W}$ sind *diagonal*. Dies wird in diesem ganzen Abschnitt angenommen, und es vereinfacht die Berechnung der Matrixprodukte $\boldsymbol{C} = \boldsymbol{A}^T\boldsymbol{W}\boldsymbol{A}$ und $\boldsymbol{b} = \boldsymbol{A}^T\boldsymbol{W}\boldsymbol{y}$, die man für die Lösung

$$\hat{\boldsymbol{a}} = \boldsymbol{C}^{-1}\boldsymbol{b} \tag{7.58}$$

benötigt.

Die Diagonalelemente der Gewichtsmatrix $\boldsymbol{W}$ aus Gleichung (7.46) werden mit $w_i = 1/\sigma_i^2$ bezeichnet. Jeder Datenpunkt y_i mit seinem Gewicht w_i macht einen unabhängigen Beitrag zu den Matrixprodukten der Endformeln, abhängig von der i-ten Zeile der Matrix $\boldsymbol{A}$. Nennt man diese i-te Zeile $\boldsymbol{A}_i$,

$$\boldsymbol{A}_i = \left(d_1,\, d_2,\, \ldots d_p\right)$$

dann können die Beiträge dieser Zeile zu $\boldsymbol{C}$ und $\boldsymbol{b}$ geschrieben werden als die $p \times p$ Matrix $w_i\boldsymbol{A}_i^T\boldsymbol{A}_i$ und der p-Vektor $w_i\boldsymbol{A}_i^T y_i$. Die Beiträge einer einzelnen Zeile sind dann

	d_1	d_2	$\ldots$	d_p
d_1	$w_i d_1^2$	$w_i d_1 d_2$	$\ldots$	$w_i d_1 d_p$
d_2		$w_i d_2^2$	$\ldots$	$w_i d_2 d_p$
$\ldots$			$\ldots$	$\ldots$
d_p				$w_i d_p^2$

	y_i
d_1	$w_i d_1 y_i$
d_2	$w_i d_2 y_i$
$\ldots$	$\ldots$
d_p	$w_i d_p y_i$

, (7.59)

wobei die symmetrischen Elemente in der unteren Hälfte weggelassen sind.

7.5.2 Geradenanpassung

In diesem Abschnitt wird eine Geradenanpassung mit der Funktion

$$y = f(x, a_1, a_2) = a_1 + a_2 x \tag{7.60}$$

behandelt, wobei a_1 und a_2 die zwei Parameter sind, welche die Gerade bestimmen. Die Meßwerte y_i liegen an den genau bekannten Punkten x_i vor. Dann hat die Matrix $\boldsymbol{A}$ die folgende Form

$$\boldsymbol{A} = \begin{pmatrix} 1 & x_1 \\ 1 & x_2 \\ 1 & x_3 \\ \dots & \\ 1 & x_n \end{pmatrix} . \tag{7.61}$$

Zur Bestimmung der Parameter müssen die Produkte $\boldsymbol{A}^T\boldsymbol{W}\boldsymbol{A}$ und $\boldsymbol{A}^T\boldsymbol{W}\boldsymbol{y}$ gebildet werden (alle Summen gehen über $i = 1, 2, \dots n$, $w_i = 1/\sigma_i^2$),

$$\begin{aligned} \boldsymbol{A}^T\boldsymbol{W}\boldsymbol{A} &= \begin{pmatrix} \sum w_i & \sum w_i x_i \\ \sum w_i x_i & \sum w_i x_i^2 \end{pmatrix} = \begin{pmatrix} S_1 & S_x \\ S_x & S_{xx} \end{pmatrix} \\ \boldsymbol{A}^T\boldsymbol{W}\boldsymbol{y} &= \begin{pmatrix} \sum w_i y_i \\ \sum w_i x_i y_i \end{pmatrix} = \begin{pmatrix} S_y \\ S_{xy} \end{pmatrix} , \end{aligned} \tag{7.62}$$

d.h. die folgenden Summen müssen gebildet werden (die Summe S_{yy} wird später benötigt):

$$\begin{array}{lllllll} S_1 &=& \sum w_i & & S_y &=& \sum w_i y_i \\ S_x &=& \sum w_i x_i & & S_{xy} &=& \sum w_i x_i y_i \\ S_{xx} &=& \sum w_i x_i^2 & & S_{yy} &=& \sum w_i y_i^2 \, . \end{array}$$

Das System linearer Gleichungen

$$\begin{pmatrix} S_1 & S_x \\ S_x & S_{xx} \end{pmatrix} \begin{pmatrix} a_1 \\ a_2 \end{pmatrix} = \begin{pmatrix} S_y \\ S_{xy} \end{pmatrix} \tag{7.63}$$

wird für die Parameter der Geraden a_1 und a_2 gelöst mit Hilfe der inversen Matrix

$$\begin{pmatrix} S_1 & S_x \\ S_x & S_{xx} \end{pmatrix}^{-1} = \frac{1}{D} \begin{pmatrix} S_{xx} & -S_x \\ -S_x & S_1 \end{pmatrix} \tag{7.64}$$

mit Determinante $D = S_1 S_{xx} - S_x^2$. Die Lösung ist

$$\begin{aligned} \hat{a}_1 &= (S_{xx} S_y - S_x \, S_{xy})/D \\ \hat{a}_2 &= (-S_x \, S_y + S_1 \, S_{xy})/D \, , \end{aligned} \tag{7.65}$$

und die Kovarianzmatrix ist

$$\boldsymbol{V}[\hat{\boldsymbol{a}}] = \frac{1}{D} \begin{pmatrix} S_{xx} & -S_x \\ -S_x & S_1 \end{pmatrix} . \tag{7.66}$$

Weiterhin ist die Summe der Residuenquadrate

$$\hat{S} = S_{yy} - \hat{a}_1 S_y - \hat{a}_2 S_{xy} \, , \tag{7.67}$$

d.h. die Summe der Quadrate der *direkten* Meßwerte S_{yy} wird vermindert um das Produkt des Lösungsvektors mit dem Vektor der rechten Seite von Gleichung (7.63).
Für einen Wert $\hat{y} = \hat{a}_1 + \hat{a}_2 x$, berechnet an der Stelle x, ist die Standardabweichung die Wurzel aus der Varianz $V[\hat{y}]$, die nach dem Gesetz der Fehlerfortpflanzung (Gleichung (4.52)) folgendermaßen berechnet wird:

$$V[\hat{y}] = V[\hat{a}_1] + x^2 V[\hat{a}_2] + 2xV[\hat{a}_1, \hat{a}_2] = (S_{xx} - 2xS_x + x^2 S_1)/D\,. \tag{7.68}$$

Die Abbildung 7.1 zeigt ein Beispiel für eine Geradenanpassung.

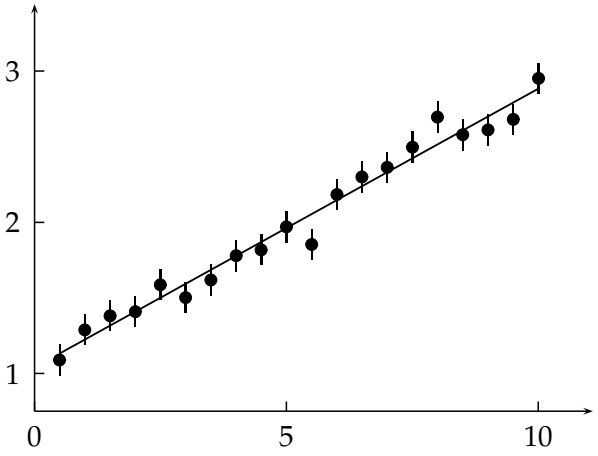


Abbildung 7.1: Anpassung einer Geraden $f(x) = a_1 + a_2 x$ an 20 Datenpunkte.

Die Parameter $\hat{a}_1$ und $\hat{a}_2$ sind im allgemeinen korreliert ($S_x \neq 0$). Die Korrelation zwischen den beiden Parametern einer Geradenanpassung kann jedoch durch eine Koordinatentransformation $x \to x'$, mit $x_i^{'} = x_i - \bar{x}$, beseitigt werden mit der Wahl

$$\bar{x} = \frac{\sum_{i=1}^{n} w_i x_i}{\sum_{i=1}^{n} w_i} = \frac{S_x}{S_1}\,,$$

d.h. man wählt den gewichteten Mittelwert der x-Werte als Ursprung der neuen x-Achse. Nennt man die Summen in dem neuen $x^{'}$-Koordinatensystem $S_x^{'}$, $S_{xx}^{'}$ usw., dann hat die Lösung für die Parameter $a_1^{'}$ und $a_2^{'}$ die einfache Form

$$a_1^{'} = \bar{y} = \frac{S_y}{S_1} \qquad\qquad a_2^{'} = \frac{S_{xy}^{'}}{S_{xx}^{'}}\,,$$

d.h. die Formel für die beste Geradenanpassung $f(x) = \bar{y} + a_2^{'}(x - \bar{x})$ enthält die Mittelwerte $\bar{x}$ und $\bar{y}$.

Geradenanpassung bei Fehlern in beiden Variablen. Beide Variablen x_i und y_i mögen fehlerbehaftet sein mit Standardabweichungen σ_{xi} bzw. σ_{yi}. Dann minimiert man die Summe der Quadrate des Abstands der Fehlerellipsen von der Geraden, also

$$S(a_1, a_2) = \sum_{i=1}^{n} \frac{(y_i - a_1 - a_2 x_i)^2}{N_i}\,,$$

mit der Abkürzung $N_i = \sigma_{y_i}^2 + {a_2}^2 \sigma_{x_i}^2$. Die Gleichungen

$$\frac{\partial S}{\partial a_1} = 0 \qquad \frac{\partial S}{\partial a_2} = 0$$

müssen unter Umständen mit numerischen Methoden gelöst werden; die Forderung $\partial S/\partial a_1 = 0$ führt auf die Gleichung

$$\hat{a}_1 = \frac{\sum y_i/N_i - \hat{a}_2 \sum x_i/N_i}{\sum 1/N_i}\,. \tag{7.69}$$

Ein mögliche Methode besteht darin, ein Minimum der Summe S zu suchen, indem man $\hat{a}_2$ variiert und $\hat{a}_1$ der Gleichung (7.69) entnimmt. Das Problem kann auch mit den allgemeinen Methoden des Kapitels 8 gelöst werden, die auch die Kovarianzmatrix der Parameter aus der Matrix der zweiten Ableitungen von $S(a_1, a_2)$ ergeben.

Lineare Regression. In diesem Zusammenhang bedeutet Regression die Anpassung einer Geraden durch eine Punktmenge in der x-y-Ebene, wobei man die Summe der Quadrate der senkrechten Abstände der Punkte von der Geraden minimiert. Dies ist ein Spezialfall des oben behandelten Problems, in dem alle Standardabweichungen denselben Wert σ haben. Man minimiert also

$$S = \sum_{i=1}^{n} \frac{(y_i - a_1 - a_2 x_i)^2}{(1 + {a_2}^2)\sigma^2}\,,$$

und dies führt auf die Lösung

$$\hat{a}_1 = \bar{y} - \hat{a}_2 \bar{x} \qquad \hat{a}_2 = q \pm \sqrt{q^2 + 1}\,,$$

mit den Abkürzungen $\bar{y} = \sum y_i/n$, $\bar{x} = \sum x_i/n$, und

$$q = \frac{\sum (y_i - \bar{y})^2 - \sum (x_i - \bar{x})^2}{2\sum (y_i - \bar{x})(x_i - \bar{x})}\,.$$

Eines der Vorzeichen der Wurzel gibt die beste Geradenanpassung.

7.5.3 Reduktion der Matrixgröße

In vielen Problemen mit einer sehr großen Anzahl von Parametern enthalten viele der Gleichungen zu einzelnen Meßwerten nur einige wenige der Parameter. Dann enthält die Produktmatrix $\boldsymbol{A}^T\boldsymbol{W}\boldsymbol{A}$ der Normalgleichungen (7.58) viele Elemente mit dem Wert null. Im folgenden Beispiel erlaubt diese Struktur der Produktmatrix, ihre Größe erheblich zu reduzieren. Da der Rechenaufwand bei der Lösung eines linearen Gleichungssystems bei n Unbekannten proportional zu n^3 ist, kann ein Problem mit sehr vielen Parametern erst dadurch in der Praxis lösbar werden.

Die Meßreihe bestehe aus einer größeren Zahl von Teilmessungen; jede Teilmessung möge zum einen von *globalen* Parametern $\boldsymbol{a}$ und zum anderen von *lokalen* Parametern $\boldsymbol{\alpha}_i$ abhängen, wobei die letzteren nur in den Teilmessungen i auftauchen. Ferner sei angenommen, daß nur die globalen Parameter und nicht die vielen lokalen von Interesse sind. In diesem Fall können alle lokalen Parameter eliminiert und damit die Größe der Matrix stark reduziert werden, entsprechend der kleineren Zahl der globalen Parameter.

Die Beiträge zur Produktmatrix zerfallen dabei in drei Teile. Der erste Teil kommt von einer symmetrischen Matrix $\boldsymbol{C}$ mit einer Dimension, die durch die Anzahl der Komponenten des globalen Parametervektors $\boldsymbol{a}$ gegeben ist. Der zweite Beitrag (auf der Diagonalen) kommt von einer weiteren symmetrischen Matrix $\boldsymbol{\Gamma}_i$, welche nur von den lokalen Parametern $\boldsymbol{\alpha}_i$ einer Teilmeßreihe abhängt. Der dritte Beitrag ist eine Rechteck-Matrix $\boldsymbol{G}_i$, deren Zeilenzahl durch die globalen Parameter $\boldsymbol{a}$ und deren Spaltenzahl durch die lokalen Parameter $\boldsymbol{\alpha}_i$ gegeben ist. Außerdem gibt es zwei Beiträge zu Vektoren der Normalgleichungen, $\boldsymbol{b}_i$ für die globalen und $\boldsymbol{\beta}_i$

für die lokalen Parameter. Die folgende Gleichung zeigt diese Beiträge in dem großen System der Normalgleichungen.

$$\left(\begin{array}{c|c|c|c} \sum \boldsymbol{C}_i & \cdots & \boldsymbol{G}_i & \cdots \\ \hline \vdots & \ddots & 0 & 0 \\ \hline \boldsymbol{G}_i^T & 0 & \boldsymbol{\Gamma}_i & 0 \\ \hline \vdots & 0 & 0 & \ddots \end{array}\right) \cdot \left(\begin{array}{c} \boldsymbol{a} \\ \hline \vdots \\ \hline \boldsymbol{\alpha}_i \\ \hline \vdots \end{array}\right) = \cdot \left(\begin{array}{c} \sum \boldsymbol{b}_i \\ \hline \vdots \\ \hline \boldsymbol{\beta}_i \\ \hline \vdots \end{array}\right) \tag{7.70}$$

In dieser Matrixgleichung enthalten die Matrizen $\boldsymbol{C}_i$, $\boldsymbol{\Gamma}_i$, $\boldsymbol{G}_i$ und die Vektoren $\boldsymbol{b}_i$ und $\boldsymbol{\beta}_i$ je einen einzelnen Beitrag der i-ten Teilmessung. Wenn man die globalen Parameter ignoriert, könnte man die Normalgleichungen $\boldsymbol{\Gamma}_i \boldsymbol{\alpha}_i = \boldsymbol{\beta}_i$ für jede Teilmessung getrennt wie üblich lösen durch

$$\boldsymbol{\alpha}_i = \boldsymbol{\Gamma}_i^{-1} \boldsymbol{\beta}_i \,. \tag{7.71}$$

Das gesamte System der Normalgleichungen hat eine spezielle Struktur, da viele Untermatrizen null sind, wie man in Gleichung (7.70) erkennt. In der Matrix in Gleichung (7.70) gibt es außer über die Untermatrizen $\boldsymbol{G}_i$ und $\boldsymbol{C}_i$ keine Verbindung zwischen den lokalen Parametern.

Da das Ziel ist, die globalen Parameter $\boldsymbol{a}$ zu bestimmen, kann man nun die Partionierungsformeln von Kapitel 3.7 anwenden. Diesen Formalismus kann man nacheinander auf jede der Teilmessungen anwenden und erhält Normalgleichungen

$$\left(\begin{array}{c} \boldsymbol{C}' \end{array}\right) \left(\begin{array}{c} \boldsymbol{a} \end{array}\right) = \left(\begin{array}{c} \boldsymbol{b}' \end{array}\right) \,, \tag{7.72}$$

die sich nur noch auf die globalen Parameter beziehen, mit einer modifizierten Matrix $\boldsymbol{C}'$ und einem modifizierten Vektor $\boldsymbol{b}'$

$$\boldsymbol{C}' = \sum_i \boldsymbol{C}_i - \sum_i \boldsymbol{G}_i \boldsymbol{\Gamma}_i^{-1} \boldsymbol{G}_i^T \qquad \boldsymbol{b}' = \sum_i \boldsymbol{b}_i - \sum_i \boldsymbol{G}_i \left(\boldsymbol{\Gamma}_i^{-1} \boldsymbol{\beta}_i \right) \,. \tag{7.73}$$

Dieser Satz von Normalgleichungen enthält explizit nur die globalen Parameter, aber auch die Informationen aus allen Teilmessungen, die die globalen Parameter beeinflussen. Der Klammerausdruck in Gleichung (7.73) stellt dabei die Lösung aus Gleichung (7.71) dar, bei der die globalen Parameter ignoriert wurden. Die Lösung

$$\boldsymbol{a} = \boldsymbol{C}'^{-1} \boldsymbol{b}' \tag{7.74}$$

repräsentiert den Lösungsvektor $\boldsymbol{a}$ mit Kovarianzmatrix $\boldsymbol{C}'^{-1}$. Diese Methode ist direkt und erfordert keine Iteration. Ein *iteratives Vorgehen* kann in den folgenden Fällen nötig werden:

- die Gleichungen hängen von den globalen Parametern nicht linear ab; in dem iterativen Vorgehen werden die Gleichungen bei jedem Iterationsschritt linearisiert;
- die Daten enthalten Ausreißer, die iterativ mit einer Reihe immer engerer Schnitte entfernt werden;
- die Genauigkeit der Daten ist zunächst nicht bekannt und muß daher nachträglich aus der Anpassungsrechnung bestimmt werden.

7.6 Nichtlineare kleinste Quadrate

Die Funktion $f(x, \boldsymbol{a})$ (= Erwartungswert der Meßgröße y am Ort x) hängt oft nichtlinear von den Parametern a_j ab. Ein Beispiel ist die Funktion $f(x, \boldsymbol{a}) = a_1 \cdot \exp(a_2 x)$, wobei $f(x, \boldsymbol{a})$ von a_1 und a_2 nichtlinear abhängt (d.h. die Ableitungen $f(x, \boldsymbol{a})$ nach den a_j sind nicht konstant, sondern hängen von den Parameterwerten ab). In diesem Fall ist die Funktion S keine quadratische Form in den Parametern wie im linearen Fall.

Nichtlineare Probleme können im allgemeinen nur durch Iteration gelöst werden, indem in jedem Iterationsschritt eine Linearisierung des Problems vorgenommen wird. Dabei sind gute Startwerte für die Parameter $\boldsymbol{a}$ erforderlich, die man vielleicht aus einer Teilmenge der Daten erhalten kann.

Man sollte beachten, daß die Eigenschaften der Methode der kleinsten Quadrate, wie sie in Kapitel 7.3 allgemein für beliebige zugrundeliegende Wahrscheinlichkeitsdichten hergeleitet wurden, nicht mehr streng gültig sind. Wenn aber die Abweichung von der Linearität klein ist, werden diese Ergebnisse ungefähr richtig bleiben. Wie in Kapitel 7.1 erwähnt, kann im Fall gauß-verteilter Fehler die Methode der kleinsten Quadrate aus dem Prinzip der Maximum-Likelihood auch im nichtlinearen Fall abgeleitet werden und hat damit auch die allgemeinen Eigenschaften der Maximum-Likelihood-Methode.

7.6.1 Linearisierung

Linearisierung der Funktion $f(x, \boldsymbol{a})$ erfordert gute Startwerte für den Parametervektor $\boldsymbol{a}$; diese seien bezeichnet mit $\boldsymbol{a}^*$. Die Funktion $f(x, \boldsymbol{a})$ wird ersetzt durch eine lineare Taylor-Entwicklung,

$$f(x, \boldsymbol{a}) \approx f(x, \boldsymbol{a}^*) + \sum_{j=1}^{p} \frac{\partial f}{\partial a_j}(a_j - a_j^*), \tag{7.75}$$

wobei die Ableitungen an der Stelle $\boldsymbol{a}^*$ genommen werden. Mit dieser Linearisierung werden mit linearen kleinsten Quadraten Korrekturen $\boldsymbol{\Delta a} = \boldsymbol{a} - \boldsymbol{a}^*$ für die Näherungswerte $\boldsymbol{a}^*$ berechnet.

Die Residuen $r_i = y_i - f(x_i, \boldsymbol{a})$ sind in Matrixnotation und in linearer Näherung

$$\boldsymbol{r} = \boldsymbol{y} - \boldsymbol{A}\boldsymbol{\Delta a} - \boldsymbol{f}\,, \tag{7.76}$$

wobei $\boldsymbol{A}$ die Jacobi-Matrix

$$\boldsymbol{A} = \begin{pmatrix} \partial f(x_1)/\partial a_1 & \partial f(x_1)/\partial a_2 & \dots & \partial f(x_1)/\partial a_p \\ \partial f(x_2)/\partial a_1 & \partial f(x_2)/\partial a_2 & \dots & \partial f(x_2)/\partial a_p \\ & & & \\ \dots & & & \\ \partial f(x_n)/\partial a_1 & \partial f(x_n)/\partial a_2 & & \partial f(x_n)/\partial a_p) \end{pmatrix} \tag{7.77}$$

ist, und $\boldsymbol{f}$ der Näherungsvektor (Elemente $f(x_i)$), alle an der Stelle $\boldsymbol{a}^*$. Aus der Linearisierung des Problems

$$S = \boldsymbol{r}^T \boldsymbol{W} \boldsymbol{r} = (\boldsymbol{y} - \boldsymbol{A}\boldsymbol{\Delta a} - \boldsymbol{f})^T \boldsymbol{W} (\boldsymbol{y} - \boldsymbol{A}\boldsymbol{\Delta a} - \boldsymbol{f}) = \text{ Minimum} \tag{7.78}$$

folgen die Normalgleichungen (vgl. Gleichung (7.58))

$$(\boldsymbol{A}^T \boldsymbol{W} \boldsymbol{A})\boldsymbol{\Delta a} = \boldsymbol{A}^T \boldsymbol{W} (\boldsymbol{y} - \boldsymbol{f}) \tag{7.79}$$

mit der Lösung

$$\boldsymbol{\Delta a} = (\boldsymbol{A}^T \boldsymbol{W} \boldsymbol{A})^{-1} \boldsymbol{A}^T \boldsymbol{W} (\boldsymbol{y} - \boldsymbol{f}) \; . \tag{7.80}$$

Die so erhaltenen Korrekturen $\boldsymbol{\Delta a}$ müssen zu der Näherung $\boldsymbol{a}^*$ addiert werden und geben eine neue (und meistens bessere) Näherung. Die Berechnung der Korrekturen wird dann, ausgehend von der neuen Näherung, wiederholt. Bei guten Startwerten werden die Korrekturen rasch klein, und die Näherung $\boldsymbol{a}^*$ wird nahe der richtigen Lösung des nichtlinearen Problems sein. Die Gleichungen für den Erwartungswert von $\hat{S}$ und für die Kovarianzmatrix $\boldsymbol{V}(\hat{\boldsymbol{a}})$, Gleichungen (7.56) und (7.54), bleiben näherungsweise gültig.

7.6.2 Konvergenz

In der Regel wird ein Iterationsschritt des linearisierten Problems der Lösung des nichtlinearen Problems näher kommen,

$$S(\boldsymbol{a}^* + \boldsymbol{\Delta a}) < S(\boldsymbol{a}^*) \; . \tag{7.81}$$

Diese Ungleichung wird nicht immer erfüllt sein. Man kann aber zeigen, daß es ein λ gibt, so daß

$$S(\boldsymbol{a}^* + \lambda \boldsymbol{\Delta a}) < S(\boldsymbol{a}^*) \; . \tag{7.82}$$

Damit hat man eine einfache Methode in solchen Fällen, in denen nach der Lösung des linearisierten Problems der neue Wert von S, $S(\boldsymbol{a}^* + \boldsymbol{\Delta a})$ größer ist als $S(\boldsymbol{a}^*)$: $\boldsymbol{\Delta a}$ wird um einen Faktor, zum Beispiel um den Faktor zwei, reduziert, und der Test wird dann wiederholt, bis ein Wert kleiner als $S(\boldsymbol{a}^*)$ erreicht ist. Es bleibt noch, eine Methode zu finden, um die Konvergenz zu erkennen. Dabei benutzt man, daß das Produkt der Vektoren $\boldsymbol{\Delta a}$ und $\boldsymbol{A}^T \boldsymbol{W} (\boldsymbol{y} - \boldsymbol{f})$

$$\Delta S = \boldsymbol{\Delta}\, \boldsymbol{a}^T \boldsymbol{A}^T \boldsymbol{W} (\boldsymbol{y} - \boldsymbol{f}) \tag{7.83}$$

im linearen Fall die Differenz in S vom Minimum mißt (Gleichung (7.55)). Für $\Delta S < 1.0$ ist der Iterationsschritt innerhalb einer Standardabweichung. Normalerweise ist die lineare Näherung meistens gut, und die Differenz ΔS wird mit jedem Iterationsschritt um ungefähr eine Größenordnung kleiner. Die Iteration kann beendet werden, wenn zum Beispiel $\Delta S < 0.1$ wird, weil dann weitere Korrekturen klein sind im Vergleich zu den statistischen Fehlern der Parameter.

Bei schlechten Startwerten und stark nichtlinearen Problemen kann das Konvergenzverhalten der hier dargestellten einfachen Methoden allerdings auch sehr schlecht sein; bei solchen Problemen wird auf die allgemeineren Optimierungsmethoden verwiesen, die in Kapitel 8 dargestellt werden.

7.7 Kleinste Quadrate mit Nebenbedingungen

7.7.1 Einleitung

In diesem Kapitel wird der allgemeinste Fall, wie in Kapitel 7.1 beschrieben, behandelt. Das Problem ist das folgende: Ein Satz von n Messungen y_i, $i = 1, 2, \ldots n$ liegt vor. Wie zuvor sollen dies unverzerrte Messungen der wahren Werte $\bar{y}_i$ sein, also

$$E[y_i] = \bar{y}_i \quad . \tag{7.84}$$

Die Kovarianzmatrix der Meßwerte sei $\boldsymbol{V}[\boldsymbol{y}]$. Das zugrundeliegende Modell möge von p Parametern a_j, $j = 1, 2, \ldots p$ abhängen und bestehe aus m Gleichungen der Art

$$f_k(\bar{\boldsymbol{a}}, \bar{\boldsymbol{y}}) = 0 \quad , \quad k = 1, 2, \ldots m \, , \tag{7.85}$$

wobei $\bar{\boldsymbol{a}}$ und $\bar{\boldsymbol{y}}$ die Vektoren der wahren Werte der Parameter und der Meßwerte sind. Diese Gleichungen werden **Nebenbedingungen** genannt. Ein einfacher Fall ist $p = 0$, d.h. es gibt Gleichungen zwischen den gemessenen Variablen, aber keine ungemessenen Variablen. Ein Beispiel ist die Messung der drei Winkel und der drei Seiten eines Dreiecks. Die sechs Variablen erfüllen drei Nebenbedingungen, da ein Dreieck durch drei Bestimmungsstücke vollständig bestimmt ist.

Die gemessenen Werte $\boldsymbol{y}$ werden die Nebenbedingungen nicht exakt erfüllen, und *Korrekturen* $\boldsymbol{\Delta y}$ zu den *gemessenen* Werten müssen bestimmt werden, so daß $\boldsymbol{y} + \boldsymbol{\Delta y}$ die Nebenbedingungen erfüllt. Im allgemeinen Fall weiterer *ungemessener* Variablen müssen ihre Werte ebenfalls ermittelt werden. Nach kleinsten Quadraten muß die gewichtete Summe der Quadrate

$$S(\boldsymbol{y}) = \boldsymbol{\Delta y}^T \boldsymbol{W} \boldsymbol{\Delta y} \qquad \boldsymbol{W} = \boldsymbol{V}[\boldsymbol{y}]^{-1} \tag{7.86}$$

ein Minimum sein.
Manchmal kann das allgemeine Problem in eine Form passend zu den Kapiteln 7.2 bis 7.6 gebracht werden. Oft sind die Nebenbedingungen aber nichtlinear und dann geht es so nicht. Eine allgemeine Methode zur Bestimmung lokaler Extrema einer Funktion mehrerer Veränderlicher mit Nebenbedingungen ist die Methode der Lagrangeschen Multiplikatoren, welche den Vorteil einer symmetrischen Behandlung aller Variablen hat. In dieser Methode werden m zusätzliche Parameter λ_k (Lagrangesche Multiplikatoren) eingeführt, einer für jede Nebenbedingung. Formal wird eine neue Funktion definiert,

$$L(\boldsymbol{y}) = S(\boldsymbol{y}) + 2 \sum_{k=1}^{m} \lambda_k f_k(\boldsymbol{a}, \boldsymbol{y}) \ , \tag{7.87}$$

wobei ein Faktor 2 eingeführt wird, der sich bei den Ableitungen gerade heraushebt. Die notwendige Bedingung für ein lokales Extremum dieser Funktion bezüglich aller Parameter $\boldsymbol{y}$, $\boldsymbol{a}$ und $\boldsymbol{\lambda}$ ist dann äquivalent zu der Bedingung für ein Minimum von $S(\boldsymbol{y})$ unter den Bedingungen $f_k(\boldsymbol{a}, \boldsymbol{y}) = 0$.
Wenn die Bedingungen linear sind, kann man die Lösung in einem Schritt finden; bei nichtlinearen Bedingungen muß eine Lösung iterativ mit linearen Näherungen gefunden werden. Dazu benötigt man Startwerte für die Variablen. Im Fall von Meßwerten können diese selbst als Startwerte benutzt werden. Für nicht gemessene Parameter muß man sich anders behelfen.

Startwerte der Parameter werden im folgenden mit $\boldsymbol{y}$ und $\boldsymbol{a}$ bezeichnet. Es seien $\boldsymbol{y}^*$ und $\boldsymbol{a}^*$ die Werte der Parameter nach dem jeweils letzten Iterationsschritt, mit $\boldsymbol{\Delta y}^* = \boldsymbol{y}^* - \boldsymbol{y}$ und $\boldsymbol{\Delta a}^* = \boldsymbol{a}^* - \boldsymbol{a}$. Der nächste Iterationsschritt, gemessen von $\boldsymbol{y}^*$ und $\boldsymbol{a}^*$ aus, ist dann $(\boldsymbol{\Delta y} - \boldsymbol{\Delta y}^*)$ bzw. $(\boldsymbol{\Delta a} - \boldsymbol{\Delta a}^*)$, wobei die Schritte $\boldsymbol{\Delta y}$ und $\boldsymbol{\Delta a}$ von den Anfangswerten $\boldsymbol{y}$ und $\boldsymbol{a}$ aus gemessen werden. Die linearisierten Gleichungen sind

$$f_k(\boldsymbol{a}^*, \boldsymbol{y}^*) + \sum_{j=1}^{p} \frac{\partial f_k}{\partial a_j} (\Delta a_j - \Delta a_j^*) + \sum_{i=1}^{n} \frac{\partial f_k}{\partial y_i} (\Delta y_i - \Delta y_i^*) \approx 0 \ , \tag{7.88}$$

wobei die Funktionswerte und alle Ableitungen an der Stelle $\boldsymbol{a}^* = \boldsymbol{a} + \boldsymbol{\Delta a}^*$ und $\boldsymbol{y}^* = \boldsymbol{y} + \boldsymbol{\Delta y}^*$ berechnet werden (für die erste Iteration ist $\boldsymbol{\Delta y}^* = 0$ und $\boldsymbol{\Delta a}^* = 0$). In Vektorschreibweise sind die linearisierten Bedingungen

$$\boldsymbol{f}^* + \boldsymbol{A}(\boldsymbol{\Delta a} - \boldsymbol{\Delta a}^*) + \boldsymbol{B}(\boldsymbol{\Delta y} - \boldsymbol{\Delta y}^*) = 0 \, , \tag{7.89}$$

oder

$$\boldsymbol{A}\boldsymbol{\Delta a} + \boldsymbol{B}\boldsymbol{\Delta y} - \boldsymbol{c} = 0 \qquad \text{mit} \qquad \boldsymbol{c} = \boldsymbol{A}\boldsymbol{\Delta a}^* + \boldsymbol{B}\boldsymbol{\Delta y}^* - \boldsymbol{f}^* \tag{7.90}$$

$$\boldsymbol{A} = \begin{pmatrix} \partial f_1/\partial a_1 & \partial f_1/\partial a_2 & \dots & \partial f_1/\partial a_p \\ \partial f_2/\partial a_1 & \partial f_2/\partial a_2 & \dots & \partial f_2/\partial a_p \\ & & & \\ \dots & & & \\ \partial f_m/\partial a_1 & \partial f_m/\partial a_2 & \dots & \partial f_m/\partial a_p \end{pmatrix} \tag{7.91}$$

$$\boldsymbol{f}^* = \begin{pmatrix} f_1(\boldsymbol{a}^*, \boldsymbol{y}^*) \\ f_2(\boldsymbol{a}^*, \boldsymbol{y}^*) \\ \\ \dots \\ f_m(\boldsymbol{a}^*, \boldsymbol{y}^*) \end{pmatrix}$$

$$\boldsymbol{B} = \begin{pmatrix} \partial f_1/\partial y_1 & \partial f_1/\partial y_2 & \dots & \partial f_1/\partial y_n \\ \partial f_2/\partial y_1 & \partial f_2/\partial y_2 & \dots & \partial f_2/\partial y_n \\ \dots & & & \\ \partial f_m/\partial y_1 & \partial f_m/\partial y_2 & \dots & \partial f_m/\partial y_n \end{pmatrix} .$$

Die Funktion L, die bezüglich $\boldsymbol{\Delta y}$, $\boldsymbol{\Delta a}$ und $\boldsymbol{\lambda}$ minimiert werden soll, ist

$$L = \boldsymbol{\Delta y}^T \boldsymbol{W} \boldsymbol{\Delta y} + 2\boldsymbol{\lambda}^T (\boldsymbol{A}\boldsymbol{\Delta a} + \boldsymbol{B}\boldsymbol{\Delta y} - \boldsymbol{c}) . \tag{7.92}$$

Die Bedingungen für ein Extremum erhält man durch Differentiation,

$$\boxed{\begin{array}{lllll} \boldsymbol{W}\boldsymbol{\Delta y} & & + & \boldsymbol{B}^T\boldsymbol{\lambda} & = & 0 \\ & & & \boldsymbol{A}^T\boldsymbol{\lambda} & = & 0 \\ \boldsymbol{B}\boldsymbol{\Delta y} & + \; \boldsymbol{A}\boldsymbol{\Delta a} & & & = & \boldsymbol{c} \end{array}} . \tag{7.93}$$

Dieses System gekoppelter Matrixgleichungen (insgesamt $(n + p + m)$ Gleichungen) wird gelöst für die Unbekannten $\boldsymbol{\Delta y}$, $\boldsymbol{\Delta a}$ und $\boldsymbol{\lambda}$.

7.7.2 Nebenbedingungen ohne ungemessene Parameter

In diesem Fall bestehen die Nebenbedingungen nur zwischen Meßgrößen (d.h. $p = 0$). Die Gleichung (7.93) reduziert sich dann zu

$$\boxed{\begin{array}{lllll} \boldsymbol{W}\boldsymbol{\Delta y} & + & \boldsymbol{B}^T\boldsymbol{\lambda} & = & 0 \\ \boldsymbol{B}\boldsymbol{\Delta y} & & & = & \boldsymbol{c} \end{array}} . \tag{7.94}$$

Die Lösung erhält man folgendermaßen: Multiplikation der ersten Gleichung mit $\boldsymbol{W}^{-1}$ von links liefert

$$\boldsymbol{\Delta y} = -\boldsymbol{W}^{-1}\boldsymbol{B}^T\boldsymbol{\lambda} . \tag{7.95}$$

Eingesetzt in die zweite Gleichung ergibt sich $\boldsymbol{B}\boldsymbol{W}^{-1}\boldsymbol{B}^T\boldsymbol{\lambda} = -\boldsymbol{c}$. Diese Gleichung wird nach $\boldsymbol{\lambda}$ aufgelöst: $\boldsymbol{\lambda} = -\boldsymbol{W}_B\,\boldsymbol{c}$, wobei die Matrix

$$\boldsymbol{W}_B = \left(\boldsymbol{B}\boldsymbol{W}^{-1}\boldsymbol{B}^T\right)^{-1} \tag{7.96}$$

eingeführt wird, die auch später benötigt wird. Die Lösung für $\boldsymbol{\lambda}$ kann in Gleichung (7.95) eingesetzt werden und ergibt

$$\boldsymbol{\Delta y} = \boldsymbol{W}^{-1}\boldsymbol{B}^T\boldsymbol{W}_B\,\boldsymbol{c} . \tag{7.97}$$

Für *nichtlineare* Probleme ist dies der erste Schritt einer Iteration. Der nächste Schritt setzt $\boldsymbol{\Delta y}^* = \boldsymbol{\Delta y}$, und das wird wiederholt, bis die Bedingungen für die korrigierten Werte $\boldsymbol{y} + \boldsymbol{\Delta y}$ mit einer vorgegebenen Genauigkeit erfüllt sind (das Problem der Konvergenz wird in Abschnitt 7.7.3 behandelt). Der Wert $\hat{\boldsymbol{y}} = \boldsymbol{y} + \boldsymbol{\Delta y}$, wobei $\boldsymbol{\Delta y}$ die zuletzt berechnete Korrektur ist, wird als Lösung genommen.

Die Kovarianzmatrix von $\hat{\boldsymbol{y}}$ erhält man durch Fehlerfortpflanzung. Es gilt $\hat{\boldsymbol{y}} = \boldsymbol{y} + \boldsymbol{\Delta y}$, wobei $\boldsymbol{\Delta y}$ aus Gleichung (7.97) in $\boldsymbol{y}$ ausgedrückt werden muß. Dazu substituiert man $\boldsymbol{c}$ aus Gleichung (7.90), $\boldsymbol{c} = \boldsymbol{B}(\boldsymbol{y}^* - \boldsymbol{y}) - \boldsymbol{f}^*$, und erhält $\boldsymbol{\Delta y} = -\boldsymbol{W}^{-1}\boldsymbol{B}^T\boldsymbol{W}_B\boldsymbol{B}\boldsymbol{y} + \text{konst}$ und damit

$$\hat{\boldsymbol{y}} = (I - \boldsymbol{W}^{-1}\boldsymbol{B}^T\boldsymbol{W}_B\boldsymbol{B})\boldsymbol{y} + \text{konst}\,. \tag{7.98}$$

Damit wird die Varianz nach Gleichung (4.50)

$$\boldsymbol{V}[\hat{\boldsymbol{y}}] = (\boldsymbol{I} - \boldsymbol{W}^{-1}\boldsymbol{B}^T\boldsymbol{W}_B\boldsymbol{B})\boldsymbol{V}(\boldsymbol{y})(\boldsymbol{I} - \boldsymbol{W}^{-1}\boldsymbol{B}^T\boldsymbol{W}_B\boldsymbol{B})^T$$

und mit $\boldsymbol{V}[\boldsymbol{y}] = \boldsymbol{W}^{-1}$

$$\begin{aligned}\boldsymbol{V}[\hat{\boldsymbol{y}}] &= \boldsymbol{W}^{-1} - 2\boldsymbol{W}^{-1}\boldsymbol{B}^T\boldsymbol{W}_B\boldsymbol{B}\boldsymbol{W}^{-1} \\ &\quad + \boldsymbol{W}^{-1}\boldsymbol{B}^T\boldsymbol{W}_B\boldsymbol{B}\boldsymbol{W}^{-1}\boldsymbol{B}^T\boldsymbol{W}_B\boldsymbol{B}\boldsymbol{W}^{-1} \\ &= \boldsymbol{W}^{-1} - \boldsymbol{W}^{-1}\boldsymbol{B}^T\boldsymbol{W}_B\boldsymbol{B}\boldsymbol{W}^{-1}\,.\end{aligned} \tag{7.99}$$

Diese Lösungsmethode erfordert die Berechnung der inversen Matrix der symmetrischen $m \times m$ Matrix $\boldsymbol{B}\boldsymbol{W}^{-1}\boldsymbol{B}^T$. Die Matrix $\boldsymbol{W}$ erscheint nur als ihre inverse Matrix $\boldsymbol{W}^{-1} = \boldsymbol{V}[\boldsymbol{y}]$.

Eine andere Lösungsmethode besteht darin, die zwei Matrixgleichungen (7.94) als eine einzige Gleichung mit partitionierten Matrizen zu schreiben,

$$\begin{pmatrix} \boldsymbol{W} & \boldsymbol{B}^T \\ \boldsymbol{B} & 0 \end{pmatrix} \begin{pmatrix} \boldsymbol{\Delta y} \\ \boldsymbol{\lambda} \end{pmatrix} = \begin{pmatrix} 0 \\ \boldsymbol{c} \end{pmatrix}. \tag{7.100}$$

Die inverse Matrix der partitionierten Matrix

$$\begin{pmatrix} \boldsymbol{W} & \boldsymbol{B}^T \\ \boldsymbol{B} & 0 \end{pmatrix}^{-1} = \begin{pmatrix} \boldsymbol{C}_{11} & \boldsymbol{C}_{21}^T \\ \boldsymbol{C}_{21} & \boldsymbol{C}_{22} \end{pmatrix} \tag{7.101}$$

hat die Untermatrizen

$$\begin{aligned}\boldsymbol{C}_{11} &= \boldsymbol{W}^{-1} - \boldsymbol{W}^{-1}\boldsymbol{B}^T\boldsymbol{W}_B\boldsymbol{B}\boldsymbol{W}^{-1} \\ \boldsymbol{C}_{21} &= \boldsymbol{W}_B\boldsymbol{B}\boldsymbol{W}^{-1} \\ \boldsymbol{C}_{22} &= -\boldsymbol{W}_B\end{aligned} \tag{7.102}$$

(siehe Kapitel 3.7). Die endgültige Lösung kann in der folgenden Form geschrieben werden, mit $\hat{\boldsymbol{y}} = \boldsymbol{y} + \boldsymbol{\Delta y}$:

$$\begin{pmatrix} \hat{\boldsymbol{y}} \\ \hat{\boldsymbol{\lambda}} \end{pmatrix} = \begin{pmatrix} \boldsymbol{y} \\ 0 \end{pmatrix} + \begin{pmatrix} \boldsymbol{C}_{11} & \boldsymbol{C}_{21}^T \\ \boldsymbol{C}_{21} & \boldsymbol{C}_{22} \end{pmatrix} \begin{pmatrix} 0 \\ \boldsymbol{c} \end{pmatrix} = \begin{pmatrix} \boldsymbol{y} + \boldsymbol{W}^{-1}\boldsymbol{B}^T\boldsymbol{W}_B\,\boldsymbol{c} \\ -\boldsymbol{W}_B\,\boldsymbol{c} \end{pmatrix}. \tag{7.103}$$

Die Kovarianzmatrix der kombinierten Vektoren $\hat{\boldsymbol{y}}$ und $\hat{\boldsymbol{\lambda}}$ erhält man durch Fehlerfortpflanzung aus der Kovarianzmatrix $\boldsymbol{V} = \boldsymbol{W}^{-1}$ und benutzt, daß die Ableitung von $\boldsymbol{c}$ nach $\boldsymbol{y}$ gleich $-\boldsymbol{B}$ ist. Das Ergebnis ist

$$\boldsymbol{V}\left[\begin{pmatrix} \hat{\boldsymbol{y}} \\ \hat{\boldsymbol{\lambda}} \end{pmatrix}\right] = \begin{pmatrix} \boldsymbol{C}_{11} & 0 \\ 0 & -\boldsymbol{C}_{22} \end{pmatrix}. \tag{7.104}$$

Die Kovarianzmatrix der verbesserten Werte $\hat{\boldsymbol{y}}$ ist also die erste $n \times n$ Untermatrix $\boldsymbol{C}_{11}$. Diese Matrix ist singulär mit Rangdefekt m, da die korrigierten Werte m Bedingungen erfüllen.

Kontrolle der Eingabedaten und der Korrekturen. Eine nützliche Größe zur Kontrolle der Konsistenz der Eingabedaten ist die Kovarianzmatrix der Korrekturen $\boldsymbol{\Delta y}$ (siehe Zeile über Gleichung (7.98))

$$\begin{aligned} \boldsymbol{V}[\boldsymbol{\Delta y}] &= (\boldsymbol{W}^{-1}\boldsymbol{B}^T\boldsymbol{W}_B\boldsymbol{B})\boldsymbol{V}[\boldsymbol{y}](\boldsymbol{W}^{-1}\boldsymbol{B}^T\boldsymbol{W}_B\boldsymbol{B})^T \\ &= \boldsymbol{W}^{-1}\boldsymbol{B}^T\boldsymbol{W}_B\boldsymbol{B}\boldsymbol{W}^{-1}\boldsymbol{B}^T\boldsymbol{W}_B\boldsymbol{B}\boldsymbol{W}^{-1} \\ &= \boldsymbol{W}^{-1}\boldsymbol{B}^T\boldsymbol{W}_B\boldsymbol{B}\boldsymbol{W}^{-1}\,, \end{aligned} \tag{7.105}$$

da $\boldsymbol{W}_B\,(\boldsymbol{B}\boldsymbol{W}^{-1}\boldsymbol{B}^T) = \boldsymbol{I}$. Diese Kovarianzmatrix ist also die Differenz zwischen den Kovarianzmatrizen $\boldsymbol{V}[\boldsymbol{y}]$ und $\boldsymbol{V}[\hat{\boldsymbol{y}}]$,

$$\boldsymbol{V}[\boldsymbol{\Delta y}] = \boldsymbol{V}[\boldsymbol{y}] - \boldsymbol{V}[\hat{\boldsymbol{y}}]\,, \tag{7.106}$$

und die Standardabweichung von Δy_i ist $\sigma_i = \sqrt{V[\boldsymbol{y}]_{ii} - V[\hat{\boldsymbol{y}}]_{ii}}$. Diese einfache Formel für die Varianz der Korrekturen, die ganz allgemein für Lösungen der Methode der kleinsten Quadrate gilt, erlaubt die Definition von normierten Korrekturen (*pull*)

$$p_i = \Delta y_i/\sigma_i\,, \tag{7.107}$$

die empfindliche Tests der Konsistenz von Daten und Modell erlauben. Der Erwartungswert der p_i ist null und der Erwartungswert der Varianz (und Standardabweichung) ist 1. Wenn viele gleichartige Daten vorliegen, kann man zur Kontrolle der Konsistenz Histogramme der normierten Korrekturen p_i erzeugen. Die Verteilung sollte der standardisierten Gauß-Verteilung folgen. Abweichung im Mittelwert und in der Breite deuten auf systematische Probleme der Daten und des Modells hin. Zu beachten ist, daß die systematische Verschiebung einer *einzigen* Meßgröße Auswirkungen auch auf die *anderen* normierten Korrekturen haben wird.

7.7.3 Der allgemeine Fall

Das Gleichungssystem (7.93) wird für den allgemeinen Fall als eine einzige Gleichung mit partitionierten Matrizen geschrieben

$$\begin{pmatrix} \boldsymbol{W} & 0 & \boldsymbol{B}^T \\ 0 & 0 & \boldsymbol{A}^T \\ \boldsymbol{B} & \boldsymbol{A} & 0 \end{pmatrix} \begin{pmatrix} \boldsymbol{\Delta y} \\ \boldsymbol{\Delta a} \\ \boldsymbol{\lambda} \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ \boldsymbol{c} \end{pmatrix}\,. \tag{7.108}$$

Aus dieser Formulierung erkennt man, daß der einzige Unterschied zwischen dem Fall mit gemessenen Variablen und dem mit gemessenen und ungemessenen Variablen (Parametern) darin besteht, daß gewisse Elemente der Matrix auf der linken Seite null sind, d.h. die Gewichte, die den *ungemessenen* Variablen entsprechen, sind null. Deshalb ist keine prinzipielle Unterscheidung zwischen gemessenen und ungemessenen Variablen nötig. Das Gleichungssystem (7.108) kann mit Standardmethoden gelöst werden.

Die inverse Matrix der partitionierten Matrix wird als ebensolche Matrix geschrieben

$$\begin{pmatrix} \boldsymbol{W} & 0 & \boldsymbol{B}^T \\ 0 & 0 & \boldsymbol{A}^T \\ \boldsymbol{B} & \boldsymbol{A} & 0 \end{pmatrix}^{-1} = \begin{pmatrix} \boldsymbol{C}_{11} & \boldsymbol{C}_{21}^T & \boldsymbol{C}_{31}^T \\ \boldsymbol{C}_{21} & \boldsymbol{C}_{22} & \boldsymbol{C}_{32}^T \\ \boldsymbol{C}_{31} & \boldsymbol{C}_{32} & \boldsymbol{C}_{33} \end{pmatrix}\,. \tag{7.109}$$

Die Abkürzungen

$$\boldsymbol{W}_B = (\boldsymbol{B}\boldsymbol{W}^{-1}\boldsymbol{B}^T)^{-1} \tag{7.110}$$

$$\boldsymbol{W}_A = (\boldsymbol{A}^T\boldsymbol{W}_B\boldsymbol{A}) \tag{7.111}$$

werden im folgenden benutzt. Damit sind die Elemente $\boldsymbol{C}_{11} \ldots \boldsymbol{C}_{33}$

$$\begin{aligned}
\boldsymbol{C}_{11} &= \boldsymbol{W}^{-1} - \boldsymbol{W}^{-1}\boldsymbol{B}^T\boldsymbol{W}_B\boldsymbol{B}\boldsymbol{W}^{-1} \\
&\quad +\boldsymbol{W}^{-1}\boldsymbol{B}^T\boldsymbol{W}_B\boldsymbol{A}\boldsymbol{W}_A^{-1}\boldsymbol{A}^T\boldsymbol{W}_B\boldsymbol{B}\boldsymbol{W}^{-1} \\
\boldsymbol{C}_{21} &= -\boldsymbol{W}_A^{-1}\boldsymbol{A}^T\boldsymbol{W}_B\boldsymbol{B}\boldsymbol{W}^{-1} \\
\boldsymbol{C}_{22} &= \boldsymbol{W}_A^{-1} \\
\boldsymbol{C}_{31} &= \boldsymbol{W}_B\boldsymbol{B}\boldsymbol{W}^{-1} - \boldsymbol{W}_B\boldsymbol{A}\boldsymbol{W}_A^{-1}\boldsymbol{A}^T\boldsymbol{W}_B\boldsymbol{B}\boldsymbol{W}^{-1} \\
\boldsymbol{C}_{32} &= \boldsymbol{W}_B\boldsymbol{A}\boldsymbol{W}_A^{-1} \\
\boldsymbol{C}_{33} &= -\boldsymbol{W}_B + \boldsymbol{W}_B\boldsymbol{A}\boldsymbol{W}_A^{-1}\boldsymbol{A}^T\boldsymbol{W}_B\,. \qquad (7.112)
\end{aligned}$$

Die Korrekturen $\boldsymbol{\Delta y}$, $\boldsymbol{\Delta a}$ und die Lagrangeschen Multiplikatoren $\boldsymbol{\lambda}$ erhält man durch Multiplikation,

$$\begin{aligned}
\boldsymbol{\Delta y} &= \boldsymbol{C}_{31}^T\,\boldsymbol{c} &&= (\boldsymbol{W}^{-1}\boldsymbol{B}^T\boldsymbol{W}_B - \boldsymbol{W}^{-1}\boldsymbol{B}^T\boldsymbol{W}_B\boldsymbol{A}\boldsymbol{W}_A^{-1}\boldsymbol{A}^T\boldsymbol{W}_B)\,\boldsymbol{c} \\
\boldsymbol{\Delta a} &= \boldsymbol{C}_{32}^T\,\boldsymbol{c} &&= \boldsymbol{W}_A^{-1}\boldsymbol{A}^T\boldsymbol{W}_B\,\boldsymbol{c} \qquad (7.113) \\
\boldsymbol{\lambda} &= \boldsymbol{C}_{33}\,\boldsymbol{c} &&= (-\boldsymbol{W}_B + \boldsymbol{W}_B\boldsymbol{A}\boldsymbol{W}_A^{-1}\boldsymbol{A}^T\boldsymbol{W}_B)\,\boldsymbol{c}\,.
\end{aligned}$$

Die Kovarianzmatrix des kombinierten Vektors $\hat{\boldsymbol{y}}$, $\hat{\boldsymbol{a}}$, $\hat{\boldsymbol{\lambda}}$ ist

$$\boldsymbol{V}\left[\begin{pmatrix}\hat{\boldsymbol{y}}\\ \hat{\boldsymbol{a}}\\ \hat{\boldsymbol{\lambda}}\end{pmatrix}\right] = \begin{pmatrix}\boldsymbol{C}_{11} & \boldsymbol{C}_{21}^T & 0\\ \boldsymbol{C}_{21} & \boldsymbol{C}_{22} & 0\\ 0 & 0 & -\boldsymbol{C}_{33}\end{pmatrix}. \qquad (7.114)$$

Die Untermatrix, die $\hat{\boldsymbol{y}}$ und $\hat{\boldsymbol{a}}$ entspricht, ist also wieder identisch mit der entsprechenden Untermatrix der inversen Matrix.

Die gewichtete Quadratsumme S in Gleichung (7.86) kann berechnet werden als Skalarprodukt

$$\hat{S} = -\boldsymbol{\lambda}^T(\boldsymbol{c} - \boldsymbol{A}\boldsymbol{\Delta a})\,. \qquad (7.115)$$

Der Erwartungswert von S ist (für ein korrektes Modell) gegeben durch

$$E[\hat{S}] = m - p\,, \qquad (7.116)$$

und S folgt einer χ^2-Verteilung mit $(m-p)$ Freiheitsgraden, wenn die Meßwerte gauß-verteilt und die Nebenbedingungen linear sind. Die Ausführungen über Konvergenz in Abschnitt 7.6.2 gelten auch hier. Eine notwendige Bedingung für Konvergenz ist, daß die Änderungen ΔS von S klein werden (zum Beispiel $\Delta S < 0.1$). Die Bedingungen müssen auch für die korrigierten Werte $\boldsymbol{a} + \boldsymbol{\Delta a}$, $\boldsymbol{y} + \boldsymbol{\Delta y}$ mit einer vorgegebenen numerischen Genauigkeit erfüllt sein, zum Beispiel

$$F = \sum_k |f_k(\boldsymbol{a} + \boldsymbol{\Delta a}, \boldsymbol{y} + \boldsymbol{\Delta y})| < \epsilon\ , \qquad (7.117)$$

wobei der Wert von ϵ von dem Problem abhängt. Dieses Kriterium verlangt, daß die numerischen Werte aller Bedingungen von derselben Größenordnung sind, was durch Skalenfaktoren erreicht werden kann. Wenn der Wert von F bei einer Iteration zunimmt, sollte man auf das Verfahren von Abschnitt 7.6.2 zurückgreifen und die Korrektur um einen Faktor verkleinern, bis eine Abnahme von F eintritt.

8 Optimierung

8.1 Einleitung

8.1.1 Optimierungsprobleme und Minimierung

Optimierungsprobleme gibt es in vielen Bereichen der Forschung. Komplexe Phänomene werden durch mathematische Modelle beschrieben, die von einem oder mehreren Parametern x abhängen. Im Fall von n Parametern x_i, $i = 1,\ 2, \ldots n$ werden diese zu einem n-Vektor $\boldsymbol{x}$ zusammengefaßt. Um die Eigenschaften des Modells zu bestimmen, die der Wirklichkeit so gut wie möglich entsprechen, wird ein von dem Variablenvektor $\boldsymbol{x}$ abhängiges Gütemaß durch eine Funktion $F(\boldsymbol{x})$ definiert. Das Problem besteht dann darin, solche Werte für die Variablen zu finden, für welche die Funktion $F(\boldsymbol{x})$ einen *optimalen Wert* annimmt, und das erfordert die *Minimierung* oder Maximierung der Funktion $F(\boldsymbol{x})$, um den kleinsten oder größten Wert zu finden, der von der Funktion angenommen wird. Minimierung und Maximierung einer Funktion sind natürlich äquivalent, da das Problem der Maximierung durch Änderung des Vorzeichens der Funktion in das der Minimierung überführt werden kann. In diesem Kapitel wird daher nur das Problem der Minimierung behandelt.

Die typischen Probleme bei der Analyse physikalischer Meßdaten sind *statistischer Natur*. Hier bietet sich die (negative) Log-Likelihood-Funktion (siehe Kapitel 6.3.1) als Gütemaß $F(\boldsymbol{x})$ an. Optimierungsprobleme erfordern nicht nur, das Minimum (oder Maximum) einer Funktion zu bestimmen, sondern es ist auch Information über die Form der Funktion im Bereich um die Lösung notwendig, um Aussagen über die *Unsicherheit* der geschätzten Werte der Parameter machen zu können; dies bedeutet im allgemeinen die Bestimmung der Kovarianzmatrix $\boldsymbol{V}$ der Parameter. Die allgemeinen Methoden dieses Kapitels können auch auf die Minimierung der Summe S quadratischer Abweichungen im Sinne der Methode der kleinsten Quadrate (Kapitel 7) angewandt werden. Dabei ist zu beachten, daß sich die Quadratsumme S und der Wert der negativen Log-Likelihood-Funktion um den Faktor 2 unterscheiden (Gleichung (7.4)).

Oft enthalten Optimierungsprobleme als wesentlichen Bestandteil Bedingungen und Einschränkungen, die erlaubte Bereiche der Werte der Parameter festlegen. Solche Einschränkungen, bezeichnet als *Nebenbedingungen* oder *Zwangsbedingungen*, können durch *Gleichungen* oder durch *Ungleichungen* gegeben sein. Werte des Parametervektors $\boldsymbol{x}$, die alle Bedingungen erfüllen, heißen *erlaubt*. Im allgemeinen sind Probleme der Minimierung mit Nebenbedingungen schwieriger zu lösen.

Probleme der Optimierung in Technik, Wissenschaft und Wirtschaft enthalten oftmals eine große Anzahl von Nebenbedingungen. Dabei wird die optimale Lösung häufig stärker durch die Nebenbedingungen bestimmt als durch die Funktion $F(\boldsymbol{x})$ selbst, die häufig linear ist (lineare Programmierung). Viele verschiedene Methoden wurden entwickelt, um mit der großen Vielfalt der Probleme fertigzuwerden. In diesem Kapitel wird die Betonung jedoch auf Probleme gelegt, die sich aus der Datenanalyse von physikalischen Experimenten ergeben, insbesondere auf die Minimierung von (negativen) Log-Likelihood-Funktionen. Dabei gibt es, wenn überhaupt, nur eine kleine Zahl von Nebenbedingungen, und die Funktion $F(\boldsymbol{x})$ hat einen glatten Verlauf, der im Bereich des Minimums angenähert eine quadratische Funktion der Variablen ist.

Die *Optimalität* eines bestimmten Punktes der Funktion $F(\boldsymbol{x})$ wird durch den Vergleich mit Punkten der Umgebung bestimmt. Man bezeichnet einen Punkt $\boldsymbol{x}^*$ als lokales Minimum, wenn für alle Punkte $\boldsymbol{x}$ mit $\boldsymbol{x} \neq \boldsymbol{x}^*$ in der Umgebung von $\boldsymbol{x}^*$ die Ungleichung

$$F(\boldsymbol{x}^*) < F(\boldsymbol{x}) \tag{8.1}$$

gilt. Der Punkt, an dem $F(\boldsymbol{x})$ seinen kleinsten Wert annimmt, heißt das *globale Minimum*. Es

gibt nur in Sonderfällen effiziente Methoden, um das globale Minimum zu finden; für eine befriedigende Lösung von Problemen ist dies jedoch meist kein Hindernis.

In der Praxis ist die obige Definition für ein lokales Minimum nicht ausreichend; man kann Bedingungen aufstellen, die auf Ableitungen basieren, wobei vorausgesetzt wird, daß die Funktion $F(\boldsymbol{x})$ einen *glatten* Verlauf hat.

8.1.2 Minimierung ohne Nebenbedingungen

Das allgemeine Problem der Minimierung ohne Nebenbedingungen ist durch

$$\boxed{\text{minimiere } \; F(\boldsymbol{x})\,, \qquad \boldsymbol{x} \in R^n} \tag{8.2}$$

gegeben. Im allgemeinen ist die Funktion $F(\boldsymbol{x})$ eine *glatte* Funktion der Parameter, zumindest in Lösungsnähe, und das bedeutet, daß kontinuierliche Ableitungen der Funktion bis zur zweiten oder höheren Ordnung existieren.

$F(\boldsymbol{x})$ sei gegeben als Funktion eines n-Vektors $\boldsymbol{x}$ von n reellen Variablen (Parametern) $x_1,\ x_2, \ldots x_n$. Der n-Vektor partieller Ableitungen von F ist eine Vektorfunktion von $\boldsymbol{x}$, er wird bezeichnet als *Gradient* $\boldsymbol{g}(\boldsymbol{x}) = \nabla F(\boldsymbol{x})$ der Funktion $F(\boldsymbol{x})$ (∇ ist der Nabla-Operator)

$$\boldsymbol{g}(\boldsymbol{x}) = \begin{pmatrix} \partial F/\partial x_1 \\ \partial F/\partial x_2 \\ \vdots \\ \partial F/\partial x_n \end{pmatrix} . \tag{8.3}$$

Die $n \times n$ zweiten partiellen Ableitungen bilden eine quadratische und symmetrische Matrix, bezeichnet als *Hesse-Matrix* $\boldsymbol{H}(\boldsymbol{x}) = \nabla^2 F(\boldsymbol{x})$ der Funktion F

$$\boldsymbol{H}(\boldsymbol{x}) = \begin{pmatrix} \partial^2 F/\partial x_1^2 & \partial^2 F/\partial x_1 \partial x_2 & \cdots & \partial^2 F/\partial x_1 \partial x_n \\ \partial^2 F/\partial x_2 \partial x_1 & \partial^2 F/\partial x_2^2 & \cdots & \partial^2 F/\partial x_2 \partial x_n \\ \cdots & & & \\ \partial^2 F/\partial x_n \partial x_1 & \partial^2 F/\partial x_n \partial x_2 & \cdots & \partial^2 F/\partial x_n^2 \end{pmatrix} . \tag{8.4}$$

Methoden zur Minimierung glatter Funktionen liegt im allgemeinen die Taylor-Entwicklung zugrunde

$$F(\boldsymbol{x}_k + \boldsymbol{\Delta x}) = F(\boldsymbol{x}_k) + \boldsymbol{g}_k^T \boldsymbol{\Delta x} + \frac{1}{2} \boldsymbol{\Delta x}^T \boldsymbol{H}_k \boldsymbol{\Delta x} + \ldots \ . \tag{8.5}$$

In dieser Gleichung werden der Gradient $\boldsymbol{g}_k$ und die Hesse-Matrix an der Stelle $\boldsymbol{x} = \boldsymbol{x}_k$ berechnet.

Optimalitäts-Bedingungen für den Fall n=1. Zunächst wird der Fall einer Variablen $(n = 1)$ betrachtet. Eine notwendige Bedingung für ein lokales Minimum der Funktion $F(x)$ für den Wert x^* ist

$$\left. \frac{\mathrm{d}}{\mathrm{d}x} F(x) \right|_{x=x^*} = 0 \ . \tag{8.6}$$

Dies ist die Bedingung für einen *stationären* Punkt (Tangente parallel zur x-Achse). Diese Bedingung ist nicht ausreichend, weil sie auch für ein Maximum oder einen Wendepunkt gilt. Eine zusätzliche, ausreichende Bedingung ist gegeben durch die Ungleichung

$$\left. \frac{\mathrm{d}^2}{\mathrm{d}x^2} F(x) \right|_{x=x^*} > 0 \,. \tag{8.7}$$

Diese Bedingung bedeutet, daß $F(x^*)$ kleiner ist als jeder andere Funktionswert $F(x)$ in der Umgebung von x^*. Sie ist nicht anwendbar auf den Fall, daß das Minimum an einer Unstetigkeitstelle der Funktion liegt.

Optimalitäts-Bedingungen im allgemeinen Fall. Im *mehrdimensionalen Fall* sind hinreichende Bedingungen für ein lokales Minimum gegeben durch

$\partial F/\partial x_i = 0$ für alle i (stationärer Punkt), und

$\boldsymbol{H}(\boldsymbol{x}^*)$ positiv-definit.

Die Eigenschaften der Hesse-Matrix $\boldsymbol{H}$ und das entsprechende geometrische Verhalten der Funktion $F(\boldsymbol{x})$ werden am besten unter Verwendung der *spektralen Zerlegung* von $\boldsymbol{H}$ erörtert. Die Eigenwerte λ und die Eigenvektoren $\boldsymbol{u}$ der Matrix $\boldsymbol{H}$ werden durch die Gleichung

$$\boldsymbol{H}\boldsymbol{u} = \lambda \boldsymbol{u} \; . \tag{8.8}$$

definiert. Eine symmetrische Matrix hat n reelle Eigenwerte λ_i und n Eigenvektoren $\boldsymbol{u}_i$, $i = 1, 2, \ldots n$, die ein orthogonales System bilden (siehe Kapitel 3.8). Die Eigenvektoren werden im folgenden als normiert angenommen, $\boldsymbol{u}_i^T \boldsymbol{u}_i = 1$. Die orthogonale Matrix $\boldsymbol{U}$ mit den (normierten) Eigenvektoren $\boldsymbol{u}$ als Spalten transformiert $\boldsymbol{H}$ in eine Diagonalmatrix $\boldsymbol{D}$, deren Diagonalelemente die Eigenwerte sind

$$\boldsymbol{D} = \boldsymbol{U}^T \boldsymbol{H} \boldsymbol{U} = \begin{pmatrix} \lambda_1 & 0 & \cdots & 0 \\ 0 & \lambda_2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \lambda_n \end{pmatrix} \; . \tag{8.9}$$

Da $\boldsymbol{U}^{-1} = \boldsymbol{U}^T$ ist, kann die Hesse-Matrix $\boldsymbol{H}$ in Form einer Produktmatrix oder durch die Summe von dyadischen Produkten von Eigenvektoren ausgedrückt werden

$$\boldsymbol{H} = \boldsymbol{U}\boldsymbol{D}\boldsymbol{U}^T = \sum_{i=1}^{n} \lambda_i \boldsymbol{u}_i \boldsymbol{u}_i^T \; . \tag{8.10}$$

Die inverse Matrix $\boldsymbol{H}^{-1}$ der Hesse-Matrix $\boldsymbol{H}$ hat, falls sie existiert, identische Eigenvektoren, die Eigenwerte von $\boldsymbol{H}^{-1}$ sind die Inversen der Eigenwerte von $\boldsymbol{H}$. Die *spektrale Norm* einer symmetrischen Matrix ist definiert als der größte Eigenwert von $\boldsymbol{H}$,

$$\|\boldsymbol{H}\| = \lambda_{\max} \qquad \|\boldsymbol{H}^{-1}\| = 1/\lambda_{\min} \; . \tag{8.11}$$

Die *Konditionszahl* κ einer Matrix $\boldsymbol{H}$ wird definiert durch

$$\kappa = \|\boldsymbol{H}^{-1}\| \cdot \|\boldsymbol{H}\| = \lambda_{\max}/\lambda_{\min} \; . \tag{8.12}$$

Die Inverse einer Matrix mit einem großen Wert der Konditionszahl ($\kappa \gg 1$) kann numerisch ungenau sein.

Man betrachtet eine quadratische Funktion $F(\boldsymbol{x})$ und ihren Gradienten $\boldsymbol{g}(\boldsymbol{x})$

$$F(\boldsymbol{x}) = \boldsymbol{g}_0^T \boldsymbol{x} + \frac{1}{2} \boldsymbol{x}^T \boldsymbol{H}_0 \boldsymbol{x} \qquad\qquad \boldsymbol{g}(\boldsymbol{x}) = \boldsymbol{g}_0 + \boldsymbol{H}_0 \boldsymbol{x} \; . \tag{8.13}$$

Die Variation dieser Funktion um einen stationären Punkt $\boldsymbol{x}^*$ (mit $\boldsymbol{g}(\boldsymbol{x}^*) = 0$) ist in Richtung eines normierten Eigenvektors $\boldsymbol{u}_i$ gegeben durch

$$F(\boldsymbol{x}^* + h\boldsymbol{u}_i) = F(\boldsymbol{x}^*) + \frac{1}{2} \lambda_i h^2 \; . \tag{8.14}$$

Für einen positiven Eigenwert $\lambda_i > 0$ *wächst* die Funktion bei positiven und negativen Werten der Schrittgröße h an. Eine *positiv-definite Matrix* $\boldsymbol{H}$ hat nur *positive* Eigenwerte λ_i, und die Funktion $F(\boldsymbol{x}^* + \boldsymbol{\Delta x})$ wird bei einem Schritt $\boldsymbol{\Delta x}$ in *jeder* Richtung anwachsen. Ein stationärer Punkt $\boldsymbol{x}^*$ der Funktion muß den Gleichungen

$$\boldsymbol{g}(\boldsymbol{x}^*) = 0 \qquad \text{oder} \qquad \boldsymbol{H}_0 \boldsymbol{x}^* = -\boldsymbol{g}_0 \tag{8.15}$$

genügen (mit Gleichung (8.13)). Für eine positiv-definite Matrix $\boldsymbol{H}$ existiert eine Lösung dieser Gleichung

$$\boldsymbol{x}^* = -\boldsymbol{H}_0^{-1} \boldsymbol{g}_0 \,, \tag{8.16}$$

und das Minimum einer quadratischen Funktion kann deshalb in einem Schritt gefunden werden. Hierauf basiert die Newton-Methode, in der allgemeine Funktionen in quadratischer Näherung minimiert werden. Für quadratische Funktionen sind die Konturen mit $n = 2$ (die Linien konstanter Funktionswerte) Ellipsen, deren Achsen in Richtung der Eigenvektoren liegen. Für $n > 2$ sind die entsprechenden Flächen Hyperellipsoide.

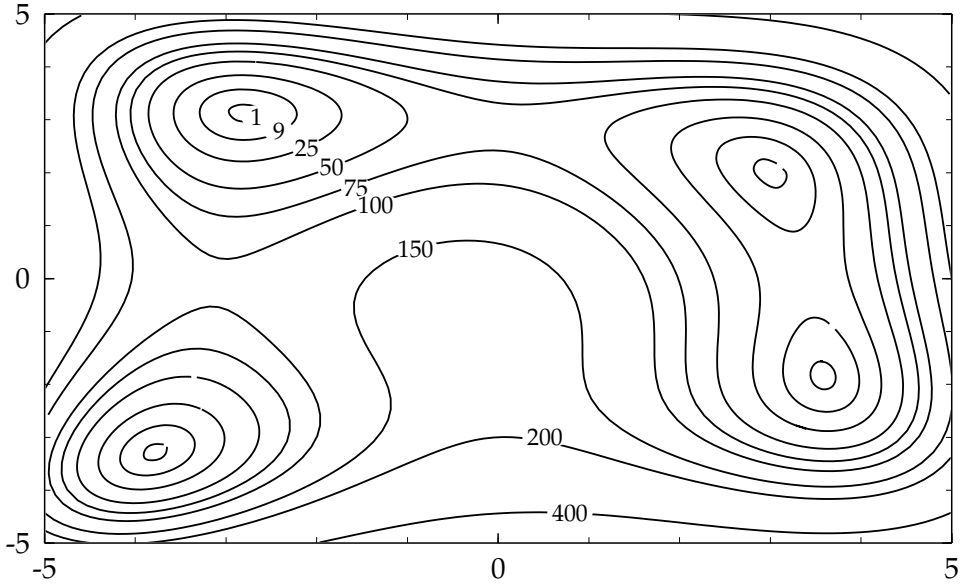


Abbildung 8.1: Konturlinien der Funktion $f(x_1, x_2) = (x_1^2 + x_2 - 11)^2 + (x_1 + x_2^2 - 7)^2$ mit Funktionswerten zwischen 1 und 400. Die Funktion hat vier Minima mit Konturlinien um die Minima, die durch Ellipsen angenähert werden können. Zwischen den Minima befinden sich Sattelpunkte.

Die Abbildung 8.1 zeigt Konturlinien (Linien gleichen Funktionswertes) für die Funktion

$$f(x_1, x_2) = (x_1^2 + x_2 - 11)^2 + (x_1 + x_2^2 - 7)^2 \,.$$

Diese Funktion besitzt mehrere Minima und ist damit eine Testfunktion, die eine zu optimierende Funktion mit mehr als einer Lösung modelliert. Man erkennt in einem Bereich um die Minima herum angenähert elliptische Konturlinien. In diesem Bereich ist die Hesse-Matrix positiv-definit. Zwischen den Minima gibt es jeweils einen Sattelpunkt. Die Funktion ist in

den Sattelpunkten stationär (verschwindender Gradient), sie hat jedoch dort kein lokales Minimum. Die Hesse-Matrix ist an diesen Punkten indefinit, mit positiven und negativen Eigenwerten. Jede Linearkombination von Eigenvektoren mit negativen Eigenwerten zeigt in Richtung *kleinerer* Funktionswerte.

Eine *positiv-semidefinite* Matrix $\boldsymbol{H}$ hat neben positiven Eigenwerten mindestens einen verschwindenden Eigenwert. Wenn ein stationärer Punkt existiert, dann wird die Funktion in Richtung der Eigenvektoren mit verschwindenden Eigenwerten unverändert bleiben. Wenn es sehr kleine Eigenwerte gibt, dann sind die Funktionswerte gegenüber Änderungen bestimmter Linearkombinationen von Variablen unempfindlich. Die Lösung solcher schlecht konditionierter Probleme kann sehr instabil sein.

Für glatte Funktionen gibt es effiziente Methoden zur Minimierung, die vom Gradienten und der Hesse-Matrix Gebrauch machen. Diese Methoden sind besonders für quadratische Funktionen ($\boldsymbol{H}$ = konstant) oder nahezu quadratische Funktionen effizient, aber oft auch für nichtglatte Funktionen. Suchmethoden, die keinen Gebrauch machen von Gradient oder Hesse-Matrix, sind auch für Funktionen mit nicht-glattem Verhalten geeignet; das Erkennen von Konvergenz ist im allgemeinen schwierig bei Suchmethoden. Likelihood-Funktionen (genauer gesagt: Log-Likelihood-Funktionen) verhalten sich zumindest in der Umgebung der Lösung sehr oft angenähert quadratisch.

8.1.3 Allgemeine Bemerkungen zu Methoden der Minimierung

Bei Optimierungsproblemen stellt es sich oft heraus, daß anscheinend einfache Rechnungen sehr sorgfältig durchgeführt werden müssen, um numerische Instabilitäten zu vermeiden. Hinweise auf solche Probleme werden im folgenden gegeben, doch können nicht alle Fälle abgedeckt werden. Es ist deshalb empfehlenswert, wann immer möglich, vorhandene gut getestete Software zu verwenden.

Selbst bei gut getesteter Software gibt es einige Aspekte, die der Anwender beachten sollte. Ein Algorithmus für die Minimierung glatter Funktionen unter Verwendung von Ableitungen kann nur dann gut funktionieren, wenn die vom Anwender vorgegebene Funktion keine, auch keine kleinen, Unstetigkeiten hat. Wenn zum Beispiel die Berechnung des Funktionswertes eine numerische Integration erfordert, sollte ein Algorithmus mit festen Abbruchbedingungen einem adaptiven Verfahren vorgezogen werden, das vielleicht genauer ist, jedoch kleine Diskontinuitäten erzeugen kann. Ähnliche Probleme ergeben sich bei Näherung durch Reihen oder beim Interpolieren in Tabellen. Sehr oft ist es wichtiger, daß $F(\boldsymbol{x})$ glatt ist als besonders genau.

Der häufigste Anwendungsfehler bei Problemen der Minimierung ist die Berechnung von Ableitungen mit fehlerhaften analytischen Formeln. Eine direkte Methode, diese Formeln zu prüfen, ist der Vergleich mit numerischen Ableitungen.

Häufig sind die Größenordnungen der Werte der Variablen stark unterschiedlich. Skalieren von Variablen, so daß sie im betrachteten Bereich in ähnlichen Größenordnungen variieren, kann die Funktionsweise eines Algorithmus verbessern. Auch die Funktionswerte in Nebenbedingungen sollten von ähnlicher Größenordnung sein.

8.2 Eindimensionale Minimierung

Methoden für die Minimierung einer Funktion mit nur *einer* Variablen sind Teil vieler Algorithmen zur Lösung von komplexeren Problemen. Bei diesen Methoden ist die Minimierung von Funktionen der Art

$$f(z) = F(\boldsymbol{x} + z \cdot \boldsymbol{\Delta x})$$

erforderlich, wobei $\Delta \boldsymbol{x}$ ein Schrittvektor in einer Suchrichtung ist; es handelt sich also um Minimierung entlang einer Geraden im R^n (*line search*).

An diesem einfachen eindimensionalen Fall können einige Aspekte von Minimierungsmethoden betrachtet werden; Suchmethoden und Methoden, die mit Ableitungen arbeiten, werden in ihrem Konvergenzverhalten verglichen.

8.2.1 Suchmethoden

In Suchmethoden zur Minimierung einer Funktion werden nur Funktionswerte benutzt, keine Ableitungen. Diese Methoden verlangen, daß die Funktion $f(x)$ *unimodal* ist. Eine Funktion $f(x)$ wird in einem Intervall (a, b), das ein Minimum einschließt, als unimodal bezeichnet, wenn das Minimum eindeutig ist (bei einer differenzierbaren Funktion hat die erste Ableitung nur *eine* Nullstelle). Ist die Funktion unimodal und gelten (unter der Annahme von $x_1 < x_2 < x_3$) die Ungleichungen

$$f(x_2) < f(x_1) \quad \text{und} \quad f(x_2) < f(x_3)\,, \tag{8.17}$$

dann schließt das Intervall x_1, x_3 ein Minimum ein (Abbildung 8.2 mit $i = 3$).

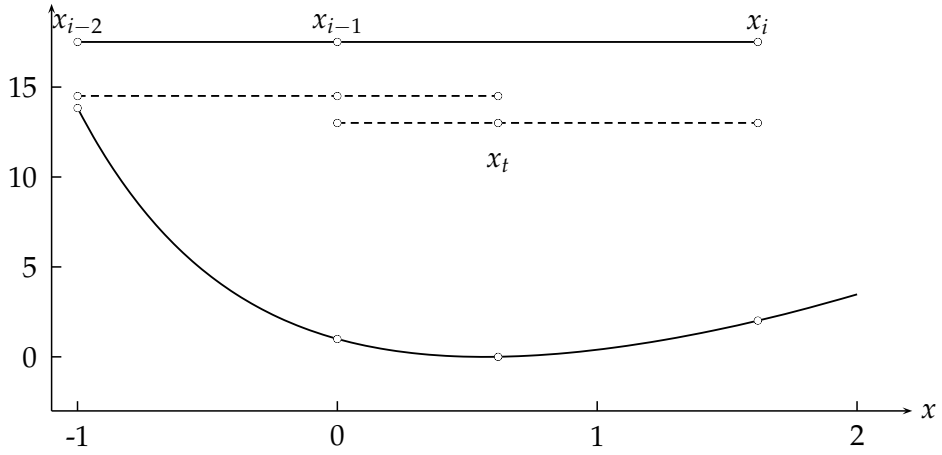


Abbildung 8.2: Verkleinerung des Einschlußintervalls durch einen Testpunkt x_t. Die obere waagerechte Linie zeigt das ursprüngliche Intervall, das das Minimum der unimodularen Funktion umschließt. Es kann durch einen weiteren Punkt x_t verkleinert werden; die beiden gestrichelten waagerechten Linien zeigen die beiden möglichen Ergebnisse.

Suchmethoden der eindimensionalen Minimierung haben zwei Phasen; zuerst müssen x-Werte gefunden werden, die das Minimum *einschließen*, und dann muß die Länge dieses Einschlußintervalls *reduziert* werden.

Einschluß des Minimums. Ein einfacher Algorithmus beginnt mit zwei Anfangswerten x_1 und x_2, die so geordnet sind, daß $f(x_1) > f(x_2)$ ist. Dann wird die Funktion nacheinander für weitere x-Werte

$$x_i = x_{i-1} + \alpha(x_{i-1} - x_{i-2}) \quad i = 3, 4, \ldots \tag{8.18}$$

berechnet (mit einem festen Faktor α, zum Beispiel $\alpha = 3$), bis ein Funktionswert mit

$$f(x_i) > f(x_{i-1}) \tag{8.19}$$

gefunden ist. Dann ist ein Minimum zwischen x_i und x_{i-2} (für eine unimodale Funktion) eingeschlossen, wie in Abbildung 8.2 gezeigt wird.

Reduktion des Einschlußintervalls. Die Größe des Einschlußintervalls muß reduziert werden. Das Intervall sei (x_{i-2}, x_i) mit einem inneren Punkt x_{i-1}, der die Ungleichungen $f(x_{i-1}) < f(x_{i-2})$, $f(x_{i-1}) < f(x_i)$ erfüllt. Mit einem Funktionswert an einem weiteren inneren Testpunkt x_t kann das Einschlußintervall reduziert werden, wie in Abbildung 8.2 gezeigt ist. Die Länge des Intervalls $(x_i - x_{i-2})$ wird entweder auf $(x_i - x_{i-1})$ oder auf $(x_t - x_{i-2})$ reduziert. Das Verfahren kann dann iterativ fortgesetzt werden, bis das Einschlußintervall klein genug ist. Die notwendige Anzahl der Funktionsberechnungen hängt entscheidend von der Strategie ab, mit der die Testpunkte gesetzt werden.

Methode des Goldenen Schnitts. Bei der Methode des Goldenen Schnitts wird eine Reduzierung der Intervallgröße um einen festen Faktor mit jedem neuen Funktionswert erreicht, unabhängig vom Verhalten der als unimodal vorausgesetzten Funktion. Ein Intervall (x_1, x_3) wird durch einen Punkt x_2 nach dem *Goldenen Schnitt* geteilt, wenn für die Verhältnisse der Intervallabschnitte

$$\frac{x_3 - x_2}{x_3 - x_1} = \frac{x_2 - x_1}{x_3 - x_2} \tag{8.20}$$

gilt (Abbildung 8.3), d.h. die kleinere Teilstrecke verhält sich zur größeren wie die größere zur ganzen Strecke. Für dieses Verhältnis folgt aus dieser Forderung der Wert τ mit

$$\tau = (\sqrt{5} - 1)/2 = 0.618034\ldots \tag{8.21}$$

Dies ist der Reduktionsfaktor für die Länge des Intervalls. Nimmt man an, daß der innere Punkt x_2 das Intervall der Länge $x_3 - x_1$ durch den Goldenen Schnitt (Gleichung (8.20)) teilt, dann ist der Testpunkt x_t definiert durch

$$\frac{x_t - x_1}{x_3 - x_1} = \tau\ . \tag{8.22}$$

Wenn $f(x_t) < f(x_2)$ gilt, ist das neue Intervall (x_2, x_3) (Abbildung 8.3), andernfalls wäre das neue Intervall (x_1, x_t). In jedem Fall ist das neue reduzierte Intervall bereits wieder durch einen inneren Punkt nach dem Goldenen Schnitt geteilt, und die Methode kann fortgesetzt werden. Man erreicht damit bei n Funktionswerten eine Reduzierung um den Faktor τ^n. Will man zum Beispiel die Länge des Intervalls um einen Faktor 100 reduzieren, so sind zehn Funktionsberechnungen notwendig, da $\tau^{10} \approx 0.00813$.

Wendet man die Methode des Goldenen Schnitts an, so sollte man in der ersten Phase (Einschluß des Minimums) als Faktor α in Gleichung (8.18) $\alpha = \tau^{-1} = 1 + \tau = 1.618034\ldots$ verwenden. Dann ist das Intervall von Anfang an durch den Goldenen Schnitt geteilt, und der Übergang zur Reduktion des Intervalls geschieht einfach entsprechend Gleichung (8.22).

Die Methode des Goldenen Schnitts ist eine *robuste* Methode, da der Reduktionsfaktor nicht vom Verhalten der Funktion abhängt. Bei vielen Funktionen jedoch kommt man mit den in Kapitel 8.2.3 beschriebenen kombinierten Methoden mit weniger Funktionsberechnungen aus.

8.2.2 Die Newton-Methode

Eine direkte Methode zur Bestimmung des Minimums einer Funktion mit Hilfe von Ableitungen läßt sich aus der Methode ableiten, mit der *Nullstellen* einer Funktion bestimmt werden. Die klassische *Newton-Raphson-Methode* zur Bestimmung der Nullstelle einer Funktion ist gegeben durch die Iterationsvorschrift

$$x_{k+1} = \Phi(x_k) \quad \text{mit} \quad \Phi(x) = x - \frac{f(x)}{f'(x)} \tag{8.23}$$

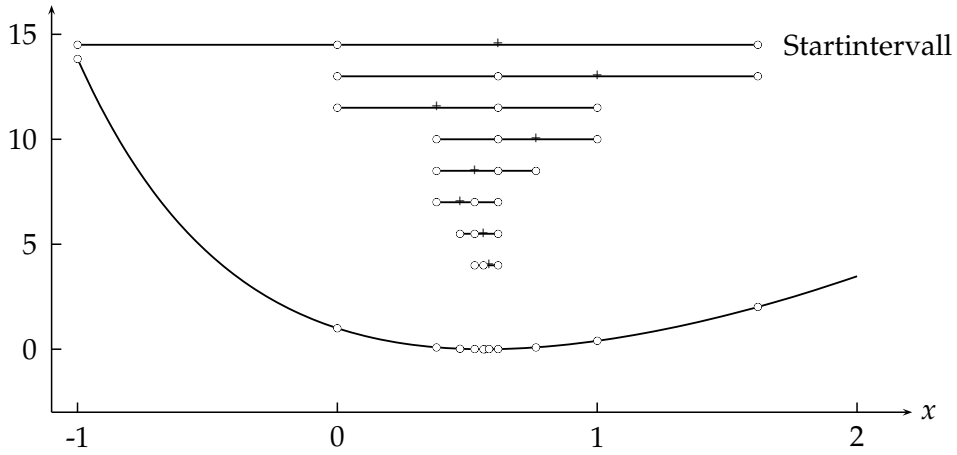


Abbildung 8.3: Bestimmung des Minimums der Funktion $f(x) = (\exp(-x) - x)^2$ nach der Methode des goldenen Schnitts. Die waagerechten Linien zeigen, von oben nach unten, jeweils das durch zwei Punkte begrenzte Intervall, das durch den mittleren Punkt nach dem goldenen Schnitt geteilt wird. Durch einen neuen Punkt kann das Intervall jeweils um den Faktor $\tau = 0.618$ verkürzt werden.

für $k = 0, 1, 2, \ldots$. Nimmt man einen Anfangswert x_0, so werden weitere Werte nach der Vorschrift (8.23) erzeugt, die, wie man hofft, gegen einen *Fixpunkt* x^* mit $x^* = \Phi(x^*)$ konvergieren. Die Abbildung 8.4 zeigt die Konvergenz bei der Bestimmung der Nullstelle der Gleichung $f(x) = \exp(-x) - x$.

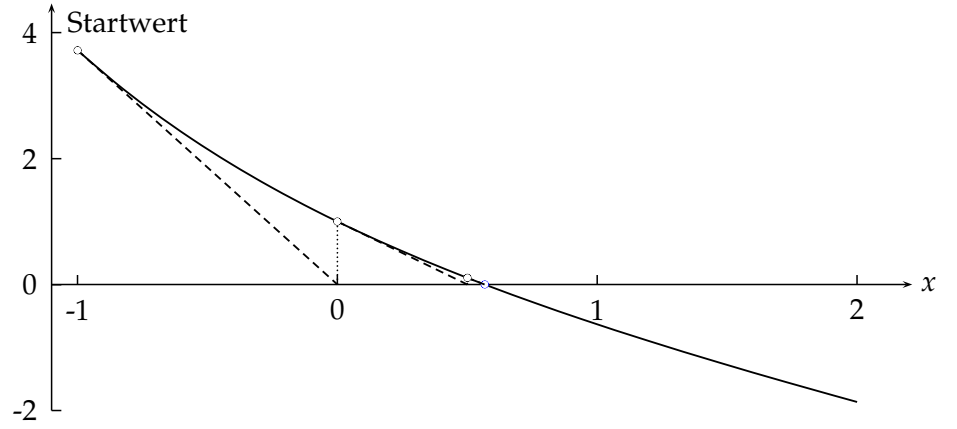


Abbildung 8.4: Bestimmung der Nullstelle der Funktion $f(x) = \exp(-x) - x$ nach der Newton-Methode mit $x_0 = -1$ als Startwert. Gestrichelt gezeichnet für die ersten beiden Iterationen ist die jeweilige Tangente an den Näherungswert, deren Schnittpunkt mit der x-Achse den nächsten Näherungswert ergibt.

Um das *Minimum* einer Funktion $f(x)$ zu finden, wendet man die obige Methode auf die Ableitung $f'(x)$ an. Diese Methode, genannt die *Newton-Methode*, wird definiert durch die Iterati-

onsvorschrift

$$x_{k+1} = \Phi(x_k) \quad \text{mit} \quad \Phi(x) = x - \frac{f'(x)}{f''(x)} \tag{8.24}$$

für $k = 0, 1, 2, \ldots$. Die Funktion muß zweimal kontinuierlich differenzierbar sein. Die Newton-Methode kann aus der Entwicklung der Funktion $f(x)$ nach einer Taylor-Reihe bis zum quadratischen Glied hergeleitet werden

$$f(x+h) \approx f(x) + \frac{h}{1!}f'(x) + \frac{h^2}{2!}f''(x) \tag{8.25}$$

$$f'(x+h) \approx f'(x) + h\, f''(x)\ . \tag{8.26}$$

Aus der Forderung $f'(x+h) = 0$ folgt

$$x_{k+1} = x_k + h \qquad \text{mit} \qquad h = -\left.\frac{f'(x)}{f''(x)}\right|_{x=x_k}\ . \tag{8.27}$$

Man beginnt bei einem Anfangswert x_0 und erzeugt nach der Iterationsvorschrift (8.24) eine Folge

$$x_0,\, x_1,\, x_2, \ldots\ x_k,\, x_{k+1} \ldots\ ,$$

die, so Gott will, gegen einen Fixpunkt x^* mit $x^* = \Phi(x^*)$ konvergiert, der die Lösung des Problems darstellt. Das Konvergenzverhalten hängt vom Verhalten der Ableitungen ab und ist nicht garantiert. Wenn bei einem bestimmten Schritt $f''(x) = 0$ ist, wird der nächste Schritt unendlich sein. Für $f''(x) < 0$ wird die Folge eher gegen ein Maximum als gegen ein Minimum konvergieren.

Konvergenzverhalten von Iterationsmethoden. Eine Iterationsmethode $x_{k+1} = \Phi(x_k)$ heißt *lokal konvergent von wenigstens der Ordnung p*, wenn für alle Anfangswerte x_0 aus einer Umgebung des Fixpunktes x^* und eine positive Konstante c

$$\frac{|x_{k+1} - x^*|}{|x_k - x^*|^p} < c \qquad \text{oder} \qquad |x_{k+1} - x^*| < c \cdot |x_k - x^*|^p \tag{8.28}$$

für alle k gilt; im linear konvergenten Fall (mit $p = 1$) wird die Bedingung $c < 1$ gefordert. Gilt dies für jedes x_0, heißt die Methode *global konvergent*. Wenn für die Ableitungen der Funktion $\Phi(x)$

$$\Phi^{(j)}(x^*) = 0 \quad \text{für alle j } = 1, \ldots (p-1) \tag{8.29}$$

$$\text{aber} \qquad \Phi^{(p)}(x^*) \neq 0 \tag{8.30}$$

gilt, dann folgt

$$\lim_{k\to\infty} \frac{|x_{k+1} - x^*|}{|x_k - x^*|^p} = \frac{\Phi^{(p)}(x^*)}{p!}\ . \tag{8.31}$$

Also wird in diesem Fall die iterative Methode lokal konvergent von der Ordnung p sein. Für *lineare* Konvergenz ($p = 1$) wird

$$|\Phi'(x^*)| < 1 \tag{8.32}$$

verlangt; ist die Konstante $\Phi'(x^*)$ positiv, so wird die Folge monoton gegen x^* konvergieren, während für eine negative Konstante die Folge x_k um x^* alternieren wird. Lineare Konvergenz

kann sehr langsam sein und viele Iterationen zum Erreichen einer bestimmten Genauigkeit erfordern. Daher ist *superlineare* Konvergenz mit Ordnung $p > 1$ erwünscht; die *quadratische* Konvergenz ($p = 2$) verlangt meist nur wenige Iterationen, wobei sich größenordnungsmäßig die Anzahl der korrekten Stellen bei jeder Iteration verdoppelt.

Konvergenzverhalten der Newton-Methode. Bei der Newton-Methode zur Bestimmung eines Minimums gemäß

$$\Phi(x) = x - \frac{f'(x)}{f''(x)} \tag{8.33}$$

mit $f''(x) > 0$ wird wegen $f'(x^*) = 0$ die erste Ableitung $\Phi'(x)$ an der Stelle x^* verschwinden,

$$\Phi'(x^*) = 1 - \frac{(f''(x^*))^2 - f'(x^*)\, f'''(x^*)}{(f''(x^*))^2} = \frac{f'(x^*)\, f'''(x^*)}{(f''(x^*))^2} = 0 \ , \tag{8.34}$$

während die zweite Ableitung im Minimum

$$\Phi''(x^*) = \frac{f'''(x^*)}{f''\ (x^*)} \tag{8.35}$$

im allgemeinen ungleich null sein wird. Daraus folgt, daß die Newton-Methode zumindest lokal *quadratisch* konvergent ist. Die Newton-Methode wird in der Praxis allgemein sehr schnell gegen die Lösung konvergieren, wenn der Anfangswert nahe bei der Lösung liegt. Es ist zu beachten, daß es nicht notwendig ist, den Funktionswert selbst zu berechnen. Es wird jedoch empfohlen, dies zu tun und bei jeder Iteration die Differenzen $\Delta x = x_{k+1} - x_k$ und $\Delta f = f(x_{k+1}) - f(x_k)$ zu prüfen.

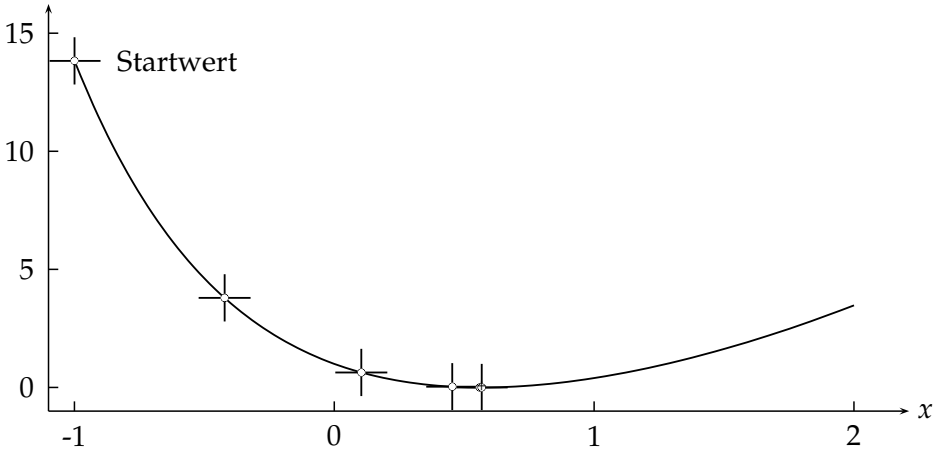


Abbildung 8.5: Bestimmung des Minimums der Funktion $f(x) = (\exp(-x) - x)^2$ nach der Newton-Methode mit $x_0 = -1$ als Startwert. Die Näherungswerte (Kreuze) konvergieren monoton gegen das Minimum bei $x = 0.567143$.

Beispiel 8.1 Minimum einer Funktion mit der Newton-Methode.

Die Konvergenz der Newton-Methode zur Bestimmung des Minimums der Funktion $f(x) = (\exp(-x) - x)^2$ beim Startwert $x_0 = -1$ wird in Abbildung 8.5 und in der untenstehenden Tabelle veranschaulicht. Das Minimum der Funktion ist die Lösung der Gleichung $x = \exp(-x) = 0.567143$.

Iteration	x	$f(x)$	Δx
0	0.000 000	1.000 000 00	
1	0.400 000	0.073 072 93	0.400 000
2	0.551 967	0.000 568 75	0.151 967
3	0.567 018	0.000 000 04	0.015 051
4	0.567 143	0.000 000 00	0.000 125

8.2.3 Kombinierte Methoden

Die beiden bisher besprochenen Methoden sind komplementär. Suchmethoden erfordern keine Ableitungen, und die Reduktion des Suchintervalls kann linear sein, wenn erst ein Einschlußintervall gefunden ist. Die Newton-Methode verlangt Ableitungen und kann sehr schnell konvergieren. Eine robuste *und* schnelle Methode kann durch Kombination von zwei Methoden konstruiert werden, ohne daß Ableitungen benötigt werden.

Die *Polynom-Interpolationsmethoden* benutzen die Kenntnis des Minimums eines Polynoms, das durch bereits berechnete Funktionswerte vorgegeben ist. In der Praxis ist es nicht nötig, über die dritte, selbst nicht über die zweite Ordnung des Polynoms hinauszugehen.

Angenommen, daß für eine unimodale Funktion die Funktionswerte f_1, f_2 und f_3 an drei Punkten x_1, x_2, x_3 mit der Ordnung

$$x_1 < x_2 < x_3 \qquad f(x_2) < f(x_1) \qquad f(x_2) < f(x_3) \tag{8.36}$$

bereits berechnet worden sind. Dann wird das Minimum durch x_1 und x_3 eingeschlossen, und das Minimum einer Parabel durch die drei Punkte kann berechnet werden.

Die parabolische Interpolation ist identisch mit der Newton-Methode für die Berechnung des Testpunktes

$$x_t = x_2 - \frac{f'(x_2)}{f''(x_2)}\,, \tag{8.37}$$

wenn die Ableitungen numerisch aus den Funktionswerten bei x_1, x_2 und x_3 berechnet werden

$$\begin{aligned} f'(x_2) &\approx \frac{1}{(x_3 - x_1)}\left[(f_3 - f_2)\,\frac{x_2 - x_1}{x_3 - x_2} + (f_2 - f_1)\frac{x_3 - x_2}{x_2 - x_1}\right] \\ f''(x_2) &\approx \frac{2}{(x_3 - x_1)}\left[\frac{f_3 - f_2}{x_3 - x_2} - \frac{f_2 - f_1}{x_2 - x_1}\right]\,. \end{aligned} \tag{8.38}$$

Für ein durch x_2 mit $d = x_2 - x_1 = x_3 - x_2$ symmetrisch geteiltes Einschlußintervall vereinfachen sich die Formeln zu

$$f'(x_2) \approx \frac{f_3 - f_1}{2d} \qquad f''(x_2) \approx \frac{f_3 - 2f_2 + f_1}{d^2}\,. \tag{8.39}$$

Probleme der Berechnung von Ableitungen mit numerischen Methoden werden in Kapitel 8.4.4 besprochen. Die symmetrische Unterteilung des Intervalls führt zu den kleinsten numerischen Abweichungen von den wahren Werten der Ableitung.

Der durch das Minimum der Parabel definierte Testpunkt wird auf jeden Fall im Intervall $[(x_1 + x_2)/2, (x_2 + x_3)/2]$ liegen. Für ein zunächst symmetrisch geteiltes Intervall wird dessen Größe nach der Reduktion zwischen der Hälfte und drei Viertel des Originalintervalls sein. Bei mehrfacher Anwendung der Parabelmethode kann das Intervall stark asymmetrisch geteilt sein, und dann wird die Berechnung weiterer Testpunkte durch parabolische Interpolation unstabil. Man erkennt dies in der Abbildung 8.6, in der das Minimum der Funktion $f(x) = (\exp(-x) - x)^2$ mit Hilfe der Parabelinterpolation bestimmt wird.

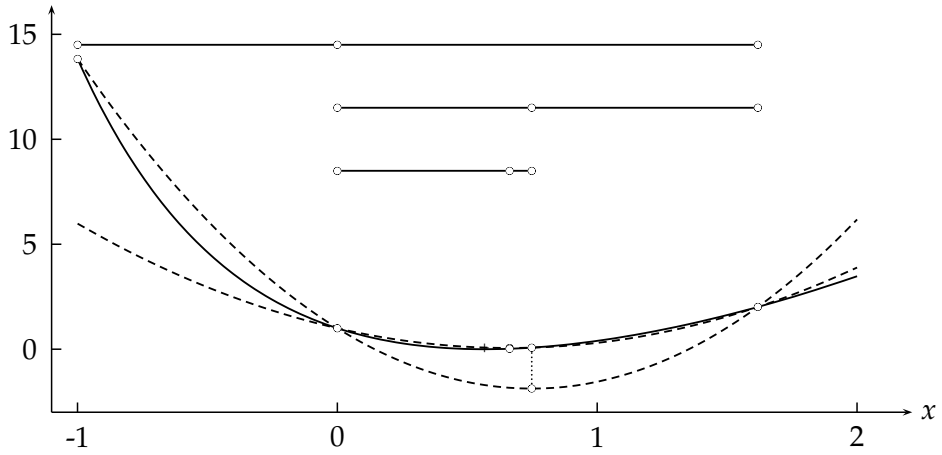


Abbildung 8.6: Bestimmung des Minimums der Funktion $f(x) = (\exp(-x) - x)^2$ mit Hilfe der Parabelinterpolation. Die gestrichelten Kurven sind Parabeln durch drei Punkte der Funktion; das Minimum der Parabel ist jeweils die nächste Testkoordinate. Die waagerechten Linien zeigen, von oben nach unten, jeweils das durch zwei Punkte begrenzte Intervall mit einem mittleren Punkt. Die zweite Parabel kommt der Funktion im Bereich um das Minimum zwar schon sehr nahe, führt jedoch zu einer stark asymmetrischen Teilung des Intervalls.

Deshalb empfiehlt sich in einem iterativen Verfahren der folgende Algorithmus:

Ein Testpunkt x_t wird durch die parabolische Interpolation Gleichung (8.37) berechnet. Liegt x_t nahe bei x_2, wird stattdessen ein anderer Testpunkt x_t berechnet mit der Gleichung

$$x_t = x_2 + (\tau - \tau^2)(x_3 - x_1), \tag{8.40}$$

mit $\tau = 0.618034$ entsprechend der Methode des Goldenen Schnitts, wenn x_2 näher bei x_1 als bei x_3 liegt; sonst werden x_1 und x_3 vertauscht. Wenn diese Formel wiederholt auf drei Punkte eines Einschlußintervalls angewandt wird, dann führt sie zu einer Teilung des Intervalls durch den Goldenen Schnitt. Diese kombinierte Methode ist im allgemeinen erfolgreicher als die (reine) Methode des Goldenen Schnitts.

8.3 Suchmethoden für den Fall mehrerer Variabler

Bei *Suchmethoden* wird die Funktion $F(\boldsymbol{x})$ an bestimmten Testpunkten des Parameterraums berechnet, und die Funktionswerte werden verglichen. Es werden keine Ableitungen verwendet, deshalb ist ihre Anwendung sehr einfach, und sie funktionieren auch bei nicht-glatten Funktionen, d.h. bei unstetigen Funktionen oder Funktionen mit Unstetigkeiten in ihren Ableitungen. Sie sind jedoch weniger effizient verglichen mit Methoden, die Ableitungen verwenden. Ferner kann keine Garantie für die Konvergenz gegeben werden, und nur begrenzte Information über das Verhalten der Funktion in der Nähe des gefundenen Minimums steht zur Verfügung. Selbst wenn die Funktion einige Unstetigkeiten hat, kann die Verwendung von Methoden mit Ableitungen weitaus effektiver sein. Oft ist die analytische Berechnung der Ableitungen einer glatten Funktion schwierig; in solchen Fällen ist es meist günstiger, mit numerischen Ableitungen zu arbeiten, statt eine Suchmethode zu verwenden.

8.3.1 Gitter- und Monte Carlo-Suchmethoden

Die einfachsten Suchmethoden sind die Gitter- und die Monte Carlo-Suchmethode. Die *Gittersuchmethode* in n Dimensionen innerhalb eines rechteckigen Bereichs mit k Unterteilungen in jeder Dimension erfordert k^n Funktionsberechnungen; dies wird für große n schnell unpraktikabel. Wenn zum Beispiel die anfängliche Bereichsgröße um einen Faktor 10 in jeder Dimension reduziert werden soll, müssen bei $n = 10$ Parametern schon 10^{10} Funktionsberechnungen durchgeführt werden.

Monte Carlo-Methoden sind bei großen n geeigneter. Die Monte Carlo-Suchmethode benutzt innerhalb eines Bereichs, der in jeder Dimension durch obere und untere Grenzen $x_i^{(l)}$ und $x_i^{(u)}$ für $i = 1,\ 2, \ldots n$ definiert ist, einen Testpunkt $\boldsymbol{x}^{(t)}$ mit den Komponenten

$$x_i^{(t)} = r_i x_i^{(l)} + (1 - r_i)\ x_i^{(u)} = x_i^{(u)} - r_i(x_i^{(u)} - x_i^{(l)})\,, \tag{8.41}$$

wobei die r_i gleichmäßig zwischen 0 und 1 verteilte Zufallszahlen sind (siehe Kapitel 5 und 5.6.4). Eine andere Monte Carlo-Methode verwendet Werte für den *Richtungskosinus* d_i, definiert durch

$$d_i = \frac{r_i}{(r_1^2 + r_2^2 + \ldots r_n^2)^{\frac{1}{2}}} \qquad i = 1,\, 2,\, \ldots n\,, \tag{8.42}$$

wobei die r_i gauß-verteilte Zufallszahlen mit Mittelwert null und einem festen Wert der Standardabweichung sind. Der Testpunkt $\boldsymbol{x}^{(t)}$ wird dann ermittelt, indem man zum vorangegangenen Punkt mit dem kleinsten Funktionswert in jeder Dimension $\Delta x_i\ d_i$ hinzuaddiert, wobei die Δx_i vorgegebene Schrittgrößen sind.

Man benutzt zuweilen Monte Carlo-Methoden, um Anfangspunkte für schnelle Methoden der Minimierung zu erzeugen. Damit kann man prüfen, ob es in einem bestimmten Bereich des Parameterraums mehr als ein lokales Minimum gibt.

8.3.2 Einfache Parametervariation

In der *Methode der einfachen Parametervariation* sucht man nach einem Minimum, wobei man jeweils nur *eine* Variable auf einmal betrachtet und eine der im Kapitel 8.2 diskutierten Methoden für eindimensionale Minimierung verwendet. Ist ein Zyklus mit der Minimierung nach allen n Richtungen beendet, wird ein weiterer Zyklus durchgeführt. Für eine glatte Funktion wird die Methode gegen die Lösung konvergieren; die Konvergenz kann jedoch sehr langsam sein, selbst für eine streng quadratische Funktion, wenn die Parameter korreliert sind. Diese langsame Konvergenz ist für eine Funktion, die von zwei Parametern abhängt, in Abbildung 8.7 gezeigt.

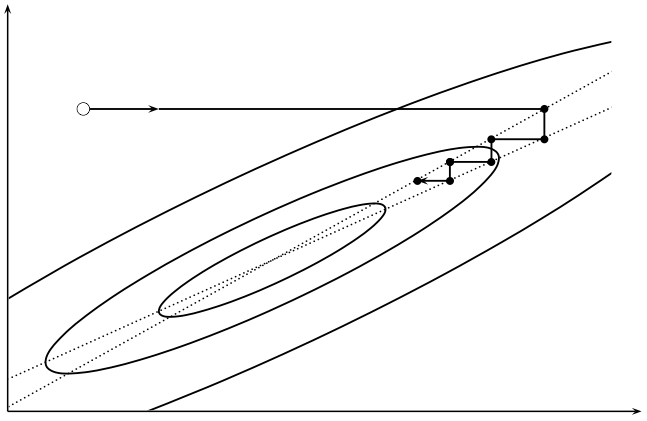


Abbildung 8.7: Die Methode der einfachen Parametervariation bei einer quadratischen Funktion von zwei Variablen. Durch die Korrelation der Parameter ist die Konvergenz langsam.

Sind jedoch die Parameter unkorreliert, so wird man das Minimum für eine quadratische Funktion nach einem Zyklus gefunden haben. Dasselbe gilt für den Fall korrelierter Parameter, wenn die eindimensionale Minimierung in Richtung der Eigenvektoren der Hesse-Matrix $\boldsymbol{H}$ erfolgt. So kann unter besonderen Umständen die Suche in n orthogonalen Richtungen eine effiziente Methode sein. In der Literatur werden viele Modifikationen der Methode einfacher Parametervariation beschrieben. Allgemein versucht man dabei während eines Zyklus Informationen über die Gestalt der Funktion zu erhalten, um dann eine bessere Wahl von n Richtungen zur Verwendung im nächsten Zyklus zu treffen.

8.3.3 Die Simplex-Methode

Die *Simplex-Methode* von Nelder und Mead [56] ist das Beispiel einer *direkten Suchmethode* zur Minimierung, die keine Ableitungen verwendet; es wird jedoch auf effiziente Weise Information aus vorangegangenen Funktionsberechnungen benutzt. Der Rechenaufwand der Methode ist gering, und sie *konvergiert* im allgemeinen in einem gewissen Sinne systematisch gegen das Minimum einer Funktion in einem vieldimensionalen Raum.

Auf jeder Stufe des Algorithmus, angewandt auf ein Problem in n Dimensionen, wird eine Menge von $n+1$ Punkten im n-dimensionalen Raum

$$\boldsymbol{x}_1,\ \boldsymbol{x}_2,\ \ldots \boldsymbol{x}_n,\ \boldsymbol{x}_{n+1}$$

betrachtet. Die Funktionswerten $F_j = F(\boldsymbol{x}_j)$ an diesen Punkten sind so geordnet, daß

$$F_1 \leq F_2 \leq \ldots F_n \leq F_{n+1}$$

gilt. Diese Punkte können als Eckpunkte eines *Polyeders* oder *Simplexes*[1] im n-dimensionalen Raum von Variablen betrachtet werden. In zwei Dimensionen ist es ein Dreieck, in drei Dimensionen ein Tetraeder. Ein nicht-entartetes Simplex umschließt ein endliches n-dimensionales Volumen; die Vektoren von einem Punkt zu allen anderen n Punkten definieren eine Menge von n linear unabhängigen Richtungen.

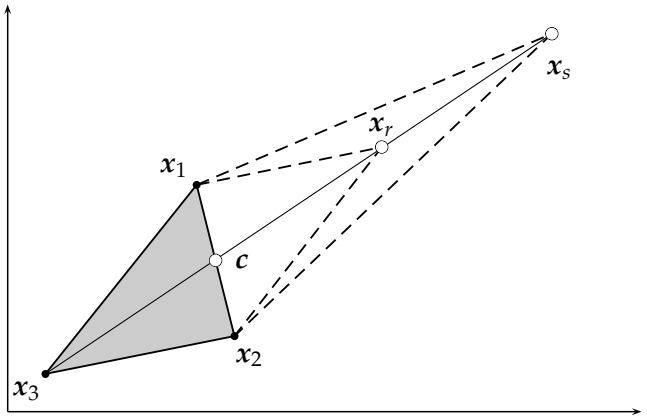


Abbildung 8.8: Einige Schritte der Simplex-Methode, ausgehend vom Simplex $(\boldsymbol{x}_1, \boldsymbol{x}_2, \boldsymbol{x}_3)$ mit dem Schwerpunkt $\boldsymbol{c}$. Die Punkte $\boldsymbol{x}_r$ und $\boldsymbol{x}_s$ sind Testpunkte.

Die Minimierung besteht in einer Folge von Zyklen mit Funktionsberechnungen; in jedem Zyklus wird ein neuer Punkt erzeugt, der den *schlechtesten* Punkt $\boldsymbol{x}_{n+1}$ ersetzt, d.h. den Punkt mit dem höchsten Funktionswert. Auf diese Weise wird in jedem neuen Zyklus eine neues Simplex gebildet.

[1] Die Methode wird Simplex-Methode genannt, hat aber nichts zu tun mit der bekannten Simplex-Methode für lineares Programmieren.

In dieser Methode spielt zusätzlich zu den $(n+1)$ Punkten ein weiterer Punkt, der *Schwerpunkt*, eine wichtige Rolle. Der Schwerpunkt ist definiert als der Mittelwert der besten n Punkte

$$\boldsymbol{c} = \frac{1}{n} \sum_{j=1}^{n} \boldsymbol{x}_j \ .$$

Zu Beginn eines jeden Zyklus wird ein Testpunkt $\boldsymbol{x}_r$ durch *Reflexion* des schlechtesten Punktes am Schwerpunkt berechnet,

$$\boldsymbol{x}_r = \boldsymbol{c} + \alpha \cdot (\boldsymbol{c} - \boldsymbol{x}_{n+1})$$

(der Faktor $\alpha > 0$ ist eine Konstante), und der Funktionswert F_r wird bei $\boldsymbol{x}_r$ berechnet. In Abhängigkeit vom Wert von F_r unterscheidet man die folgenden drei Fälle:

[$F_1 \leq F_r \leq F_n$]: Der Testpunkt $\boldsymbol{x}_r$ ist ein mittlerer Punkt; er ist nicht besser als der vorangegangene beste (niedrigste) Punkt, und er ist nicht schlechter als der zweithöchste Punkt. Der schlechteste Punkt $\boldsymbol{x}_{n+1}$ fällt weg und der Testpunkt wird als neuer Punkt aufgenommen.

[$F_r \leq F_1$]: Der Testpunkt $\boldsymbol{x}_r$ ist der bisher beste (niedrigste) Punkt, und das zeigt an, daß die Suchrichtung eine *gute* Richtung war. Um diese Richtung weiter zu prüfen, wird ein neuer Punkt $\boldsymbol{x}_s = \boldsymbol{c} + \beta \cdot (\boldsymbol{x}_r - \boldsymbol{c})$ (der Faktor $\beta > 1$ ist eine Konstante) bestimmt und der Funktionswert F_s berechnet. Für $F_s < F_r$ ist die Erweiterung erfolgreich, und $\boldsymbol{x}_s$ ersetzt $\boldsymbol{x}_{n+1}$; andernfalls ist die Erweiterung mißlungen, und $\boldsymbol{x}_r$ ersetzt $\boldsymbol{x}_{n+1}$.

[$F_r > F_n$]: Das Simplex is zu groß und muß verkleinert werden. Für $F_r < F_{n+1}$ ersetzt der Testpunkt $\boldsymbol{x}_r$ den schlechtesten Punkt $\boldsymbol{x}_{n+1}$. Dann wird durch Verkleinerung ein neuer Testpunkt $\boldsymbol{x}_s$ definiert: $\boldsymbol{x}_s = \boldsymbol{c} - \gamma(\boldsymbol{c} - \boldsymbol{x}_{n+1})$, wobei γ eine Konstante mit $0 < \gamma < 1$ ist. Wenn der kontrahierte Punkt $\boldsymbol{x}_s$ mit $F_s = F(\boldsymbol{x}_s) < F_{n+1}$ eine Verbesserung ist, dann wird $\boldsymbol{x}_{n+1}$ durch diesen Punkt $\boldsymbol{x}_s$ ersetzt; andernfalls hat es im Zyklus keine Verbesserung gegeben, und eine Verkleinerung erfolgt um den besten Punkt $\boldsymbol{x}_1$. Alle Punkte außer dem besten Punkt werden durch neue Punkte ersetzt, die definiert sind durch $\boldsymbol{x}_j = \boldsymbol{x}_1 + \delta(\boldsymbol{x}_j - \boldsymbol{x}_1)$ für $j = 2, \ldots n+1$; δ ist eine Konstante mit $0 < \delta < 1$. Damit sind n neue Funktionsberechnungen notwendig.

Typische Werte für die Konstanten sind: $\alpha = 1$, $\beta = 2$, $\gamma = 0.5$ und $\delta = 0.5$.

Die Methode wird in Abbildung 8.8 dargestellt. Ein Konvergenzkriterium für die Simplex-Methode kann auf der Differenz zwischen F_1 und F_{n+1} beruhen. In der Literatur gibt es etliche Modifikationen des Verfahrens. Das Simplex kann sich nach einigen Zyklen stark verformt haben und unausgewogen sein, sodaß die weitere Konvergenz sehr langsam wird. Ist der beste Punkt über viele Zyklen unverändert geblieben, dürfte es besser sein, mit einem neuen Simplex neu zu beginnen, wobei man vielleicht die beiden besten Punkte beibehält.

Beispiel 8.2 Minimierung einer Funktion mit der Simplex-Methode.

Die Abbildungen 8.9 und **??** zeigen die Anwendung der Simplex-Methode auf die Minimierung der Rosenbrook-Funktion

$$f(x_1, x_2) = 100(x_2 - x_1^2)^2 + (1 - x_1)^2 \ ,$$

die oft zum Test von Minimierungsmethoden verwendet wird. Üblich ist der Startpunkt $(x_1, x_2) = (-1.2, 1)$, von dem aus das Minimum bei $(x_1, x_2) = (1, 1)$ durch ein sehr schmales und flaches, parabelförmiges Tal erreicht werden muß. Die Abbildung 8.9 zeigt, daß sich die Methode bei

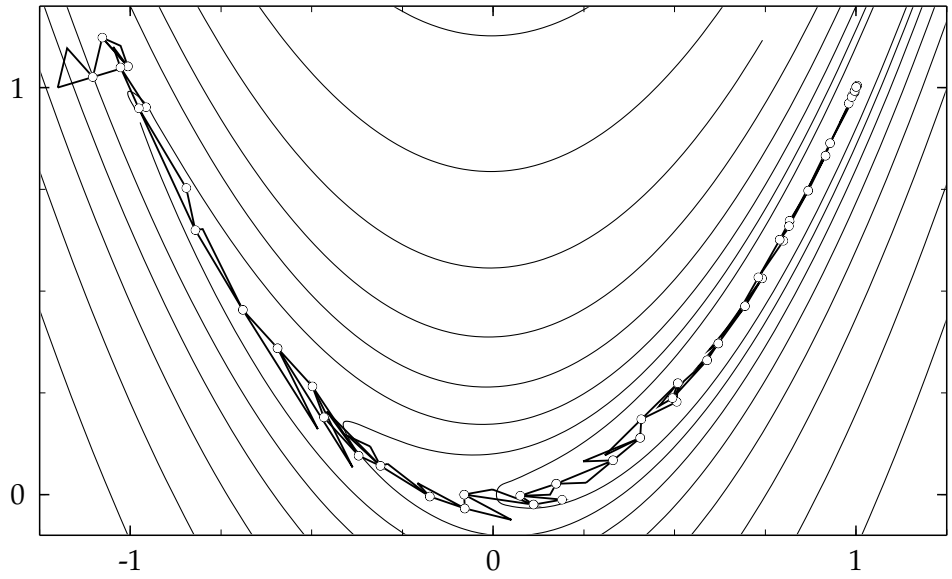


Abbildung 8.9: Minimierung der Rosenbrook-Funktion $f(x_1, x_2) = 100(x_2 - x_1^2)^2 + (1 - x_1)^2$ mit der Simplex-Methode vom Startwert $(-1.2, 1.0)$ bis zum Minimum bei $(1.0, 1.0)$. Die niedrigsten Funktionswerte bei jedem Zyklus sind durch Kreise, und der Simplex ist bei jedem zweiten Zyklus durch Linien angegeben. Die Konturlinien der Funktion sind für die Funktionswerte 1, 2, 4, 8, 16, 32, 64 und 128 gezeichnet. Die Funktion hat ein sehr enges parabelförmiges Tal.

dieser schwierigen Minimumsuche gut bewährt. Am Ort des Minimums ist die Matrix der zweiten Ableitungen singulär.

Die Abbildung **??** zeigt vergößert den Bereich um den Startwert herum. Bei jedem zweiten Zyklus ist das Simplex, hier ein Dreieck, als graue Fläche eingezeichnet. Man erkennt, daß sich die Form des Simplex gut dem Verlauf der Funktion anpaßt, die durch Konturlinien dargestellt ist. Dadurch wird, auch ohne Ableitungen zu benutzen, die Information aus vorhergehenden Funktionsberechnungen sinnvoll genutzt.

8.4 Minimierung ohne Nebenbedingungen

Die in diesem Kapitel diskutierten Methoden zur Lösung des Problems

$$\boxed{\text{minimiere } F(\mathbf{x}) \ , \qquad \mathbf{x} \in R^n} \tag{8.43}$$

sind für glatte Funktionen effizient und, richtig implementiert, auch robust. Für glatte Funktionen wird gefordert, daß die erste und die zweite Ableitung existieren und stetig sind.

Die Methoden sind iterativ. Beginnt man mit einem Anfangspunkt $\boldsymbol{x}_0$, so werden verbesserte Punkte $\boldsymbol{x}_k$ mit einem kleineren Funktionswert für $k = 1, 2, \ldots$ bestimmt, bis Konvergenz erreicht ist. Die Methoden basieren auf der quadratischen Näherung der Funktion $F(\boldsymbol{x})$ an jedem Punkt $\boldsymbol{x}_k$:

$$F(\boldsymbol{x}_k + \boldsymbol{\Delta x}) \approx F_k + \boldsymbol{g}_k^T \boldsymbol{\Delta x} + \frac{1}{2} \boldsymbol{\Delta x}^T \boldsymbol{H}_k \boldsymbol{\Delta x} \tag{8.44}$$

$$\boldsymbol{g}(\boldsymbol{x}_k + \Delta \boldsymbol{x}) \approx \boldsymbol{g}_k + \boldsymbol{H}_k \boldsymbol{\Delta x} \ , \tag{8.45}$$

wobei der Funktionswert und die Ableitungen an der Stelle des augenblicklichen Punkts berechnet werden,

$$F_k = F(\boldsymbol{x}_k)\,, \tag{8.46}$$

$$(g_k)_i = \left.\frac{\partial F}{\partial x_i}\right|_{\boldsymbol{x}=\boldsymbol{x}_k} \qquad i = 1,\ 2,\ \ldots n\,, \tag{8.47}$$

$$(H_k)_{ij} = \left.\frac{\partial^2 F}{\partial x_i \partial x_j}\right|_{\boldsymbol{x}=\boldsymbol{x}_k} \qquad i,j = 1,\ 2,\ \ldots n\,. \tag{8.48}$$

Besonders in statistischen Anwendungen (wo die Funktion $F(\boldsymbol{x})$ eine negative Log-Likelihood-Funktion ist) ist die quadratische Näherung oftmals zumindest in Lösungsnähe gut, und die Hesse-Matrix ist im Parameterraum nahezu konstant. Die Zahl der an einem bestimmten Punkt $\boldsymbol{x}_k$ zu berechnenden Größen wächst quadratisch mit der Zahl der Parameter n: Zusätzlich zum Funktionswert gibt es n Komponenten des Gradienten $\boldsymbol{g}_k$ und $n(n+1)/2$ verschiedene Elemente der Hesse-Matrix $\boldsymbol{H}_k$, insgesamt $1 + n(n+3)/2$ Größen, verglichen mit nur einer Größe (dem Funktionswert) bei einfachen Suchmethoden. Die gesamte Rechenzeit für die Berechnung eines Schrittes wächst also bei Verwendung von Ableitungen quadratisch mit n; die schnelle Konvergenz dieser Methoden jedoch wiegt im allgemeinen diesen Effekt mehr als auf. Natürlich kann die Bestimmung der Formeln für die Ableitungen sehr mühsam sein. Numerische und Näherungsmethoden für die Berechnung von Ableitungen reduzieren aber den Aufwand und erhalten die grundlegenden Vorteile der Methoden. Zudem wird in statistischen Anwendungen die Hesse-Matrix $\boldsymbol{H}$ am Lösungspunkt ohnedies gebraucht, nämlich für die Berechnung der Kovarianzmatrix der Parameter.

8.4.1 Die Newton-Methode als Standardverfahren

Die *Newton-Methode* ist die Standard-Methode für die Funktionsminimierung; die Eigenschaften anderer Methoden werden in der Regel an den Konvergenzeigenschaften und an der Geschwindigkeit dieser Methode gemessen. Ausgehend von der quadratischen Näherung (Gleichung (8.45)) und unter Annahme einer positiv-definiten Hesse-Matrix $\boldsymbol{H}$ ist der Newton-Schritt $\boldsymbol{\Delta x}$, entsprechend der Forderung $\boldsymbol{g}_k + \boldsymbol{H}_k \boldsymbol{\Delta x} = 0$, gegeben durch

$$\boldsymbol{\Delta x}_N = -\boldsymbol{H}_k^{-1} \boldsymbol{g}_k\,. \tag{8.49}$$

Die Newton-Methode wird in Abbildung 8.10 bei der Minimierung einer Funktion von zwei Variablen veranschaulicht. Die Konturlinien dieser quadratischen Funktion sind Ellipsen. Der Gradient $\boldsymbol{g}$ im Startpunkt steht senkrecht auf der Konturlinie durch diesen Punkt. Der Newton-Schritt führt in diesem Fall direkt zum Minimum.

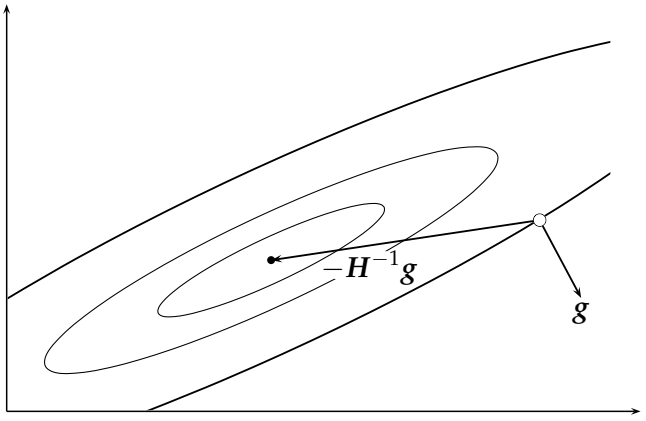


Abbildung 8.10: Ein Schritt der Newton-Methode bei einer quadratischen Funktion von zwei Variablen, deren Konturlinien Ellipsen sind. Gezeigt ist der Gradientenvektor $\boldsymbol{g}$ am Startpunkt, und der Newton-Schritt $-\boldsymbol{H}^{-1}\boldsymbol{g}$, der in diesem Fall genau auf das Minimum zeigt.

Analog zum eindimensionalen Fall kann man einen iterativen Algorithmus durch $\boldsymbol{x}_{k+1} = \boldsymbol{x}_k + \boldsymbol{\Delta x}_N$ definieren. Ein solcher Algorithmus wird in vielen Fällen tatsächlich sehr schnell gegen die Lösung konvergieren, jedoch kann der Algorithmus in bestimmten Fällen auch instabil sein; der Fall $F_{k+1} > F_k$ kann auftreten mit einem divergierenden Verhalten des Algorithmus, besonders in Bereichen, die weit entfernt von der Lösung liegen und wo die quadratische Näherung noch ungenau ist.

Die Eigenschaften der Newton-Schritte $\boldsymbol{\Delta x}$ müssen daher näher untersucht werden. Ein Schrittvektor $\boldsymbol{\Delta x}$ definiert eine *Richtung* (Einheitsvektor $\boldsymbol{\Delta x}/|\boldsymbol{\Delta x}|$) und eine *Größe* $|\boldsymbol{\Delta x}|$ des Schrittes. In dem unten beschriebenen Verfahren wird die Richtung direkt in der iterativen Minimierungsprozedur benutzt, während die Schrittgröße, obwohl sie in der Newton-Methode eigentlich bereits definiert ist, meist modifiziert wird, um den Algorithmus *robuster* zu machen.

Zunächst wird gezeigt, daß die durch den Newton-Schritt definierte Richtung zu *kleineren* Funktionswerten (*abwärts*) führt. Eine Richtung $\boldsymbol{\Delta x}$ führt dann abwärts, wenn $\boldsymbol{g}_k^T \boldsymbol{\Delta x} < 0$ gilt, d.h. der Kosinus des Winkels zwischen dem Gradientenvektor und dem Schrittvektor muß negativ sein. Das trifft in der Tat für den Newton-Schritt $\boldsymbol{\Delta x}_N = -\boldsymbol{H}_k^{-1}\boldsymbol{g}_k$ zu, weil die Matrix $\boldsymbol{H}_k^{-1}$ positiv-definit ist, und daher mit Gleichung (8.49)

$$\boldsymbol{g}_k^T\, \boldsymbol{\Delta x}_N = -\boldsymbol{g}_k^T \boldsymbol{H}_k^{-1} \boldsymbol{g}_k < 0 \tag{8.50}$$

folgt. Durch das Skalarprodukt von Gradient und Newton-Schritt wird die wichtige (positive) Größe d mit

$$d = -\boldsymbol{g}_k^T\, \boldsymbol{\Delta x}_N \tag{8.51}$$

definiert, die das Verhalten der Funktion $F(\boldsymbol{x})$ in der quadratischen Näherung bestimmt. Die Größe d ist auch gleich dem quadratischen Ausdruck

$$d = \boldsymbol{g}_k^T\, \boldsymbol{H}_k^{-1}\, \boldsymbol{g}_k = \boldsymbol{\Delta x}_N^T\, \boldsymbol{H}_k\, \boldsymbol{\Delta x}_N \quad . \tag{8.52}$$

Diese Gleichung kann folgendermaßen interpretiert werden: Die Größe d ist gleich dem Quadrat der Norm des Gradienten, wenn die inverse Matrix $\boldsymbol{H}_k^{-1}$ als Metrik benutzt wird, und sie ist auch das Quadrat der Norm des Newton-Schrittes $\boldsymbol{x}_N$, wenn die Matrix $\boldsymbol{H}_k$ als Metrik benutzt wird. Beide, die Norm des Gradienten und die des Newton-Schrittes, müssen während der Konvergenz sehr klein werden.

Aus der Funktion $F(\boldsymbol{x})$ und dem Newton-Schritt $\boldsymbol{\Delta x}_N$ wird eine eindimensionale Funktion

$$f(z) = F(\boldsymbol{x}_k + z \cdot \boldsymbol{\Delta x}_N) \tag{8.53}$$

einer (skalaren) Variablen z konstruiert. In der quadratischen Näherung der Funktion $F(\boldsymbol{x})$ folgt der quadratische Ausdruck

$$f(z) \approx F_k + \left(\boldsymbol{g}_k^T\, \boldsymbol{\Delta x}_N\right) z + \frac{1}{2}\left(\boldsymbol{\Delta x}_N^T\, \boldsymbol{H}_k\, \boldsymbol{\Delta x}_N\right) z^2 \,. \tag{8.54}$$

In dieser Näherung sind die Funktion $f(z)$ und ihre Ableitung $f'(z)$ durch die oben definierte Größe d vollständig bestimmt:

$$\begin{aligned} f(z) &= F_k + d\left(z^2/2 - z\right) \\ f'(z) &= d\,(z-1) \quad . \end{aligned} \tag{8.55}$$

Das Verhalten der Funktion wird in Abbildung 8.11 gezeigt. Da die Richtung des Newton-Schrittes abwärts geht, müssen nur positive Werte von z betrachtet werden. Das Minimum der

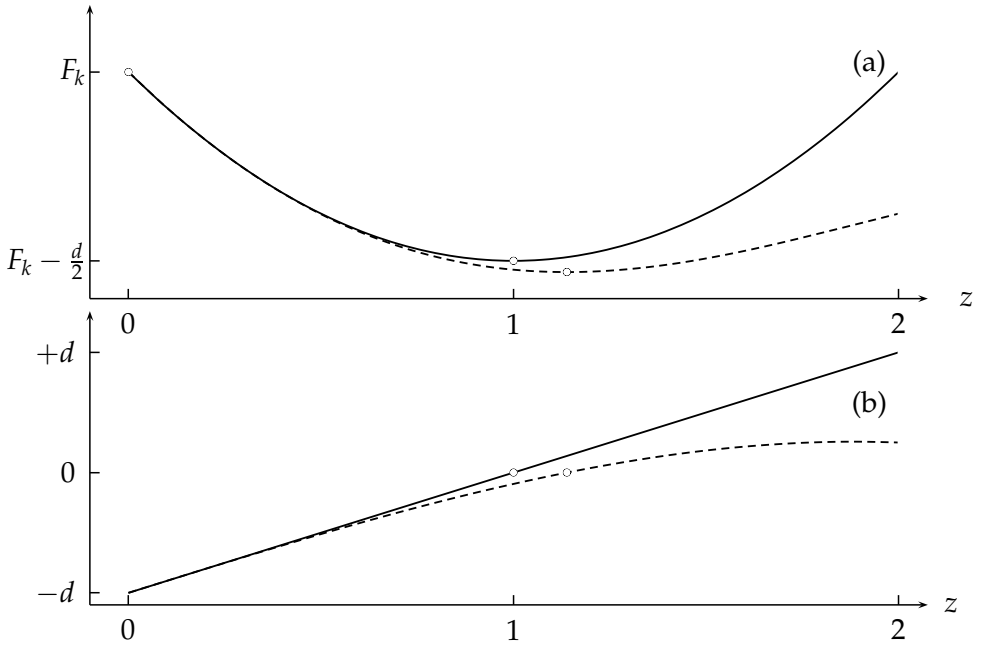


Abbildung 8.11: Die Funktion $f(z) = F_k + d(z^2/2 - z)$ (a) und die Ableitung $f'(z) = d(z-1)$ (b) als durchgezogene Kurven. Gestrichelt gezeigt ist die mögliche Abhängigkeit innerhalb einer Minimierung, die vom idealisierten Verlauf abweicht.

quadratischen Funktion $f(z)$ liegt bei $z = 1$ mit einem Funktionswert $f(1) = F_k - d/2$, also um $d/2$ unter dem Funktionswert bei $z = 0$; der Funktionswert wird um $d/2$ reduziert. Die anfängliche Steigung der Funktion $f(z)$ bei $z = 0$ ist $-d$.

In praktischen Anwendungen wird das Verhalten der Funktion $f(z)$ von diesem *idealisierten* Verhalten abweichen; zum Beispiel kann selbst der Fall $f(1) > f(0)$ eintreten. Ein *stabiler* Newton-Algorithmus schließt eine genaue Minimierung der durch Gleichung (8.53) definierten Funktion $f(z)$ bei jeder Iteration ein. Diese Minimierung erfolgt im n-dimensionalen Raum des Parameters $\boldsymbol{x}$, entlang der durch die Richtung des Newton-Schrittes definierten Geraden (*line search*). Dadurch wird in jedem Iterationsschritt ein Wert von $z_{\min}$ bestimmt, der $f(z)$ minimiert. Dies kann durch eine der in Kapitel (8.2) beschriebenen Methoden geschehen. Der nächste Punkt im n-dimensionalen Raum ist dann

$$\boldsymbol{x}_{k+1} = \boldsymbol{x}_k + z_{\min} \cdot \boldsymbol{\Delta x}_N \quad . \tag{8.56}$$

Die Steigung $f'(z)$ der Funktion $f(z)$, die anfangs gleich $-d$ ist, sollte auf einen kleinen Bruchteil (zum Beispiel ≈ 0.01) dieses Wertes reduziert werden.

Im allgemeinen liegen die Werte $z_{\min}$ während der Iterationen nahe bei dem *idealen* Wert 1, was bedeutet, daß die Funktion $F(\boldsymbol{x})$ tatsächlich in guter Näherung quadratisch ist. Man kann zeigen, daß die Newton-Methode mit genauer eindimensionaler Minimierung *quadratisch konvergent* ist, wenn die Werte $z_{\min}$ während der Iterationen gegen 1 gehen. Im allgemeinen ist das Minimum der Funktion $F(\boldsymbol{x})$ nach einigen Iterationen mit guter Genauigkeit ermittelt; wenn die Iteration dennoch fortgesetzt wird, werden die Schritte $\boldsymbol{\Delta x}_N$ extrem klein und durch Rundungsfehler bestimmt; in diesem Fall können die Werte $z_{\min}$ null werden oder viel kleiner als eins, mit vernachlässigbarem Fortschritt der Minimierung.

Das *Erkennen* der Konvergenz ist ein wichtiger Teil des Algorithmus insgesamt. Um ein sinnvolles Kriterium für die Konvergenz definieren zu können, müssen die Eigenschaften und

die Genauigkeit der Funktion $F(\boldsymbol{x})$ bekannt sein. Wenn $F(\boldsymbol{x})$ eine negative Log-Likelihood-Funktion in einer statistischen Anwendung ist, sind sehr allgemeine Aussagen über das Verhalten der Funktion möglich (siehe Kapitel 6.6). Die Bedingung

$$F(\boldsymbol{x}^* + \boldsymbol{\Delta x}) = F(\boldsymbol{x}^*) + \frac{1}{2} \tag{8.57}$$

definiert die Kontur der Funktion um das Minimum $\boldsymbol{x}^*$ innerhalb *einer* Standardabweichung. Daher repräsentiert das Anwachsen der Funktion $F(\boldsymbol{x})$ um $1/2$ verglichen mit dem Minimum einen Schritt entsprechend *einer* Standardabweichung. Innerhalb eines solchen Schrittes verhält sich die Funktion im allgemeinen mit einer sehr guten Genauigkeit quadratisch, und ein weiterer Schritt reicht aus für eine genaue Bestimmung des Minimums. Diese Funktionsänderung entspricht einem Wert von 1 der oben definierten Größe d. Für ein sicheres Konvergenzkriterium kann daher der Wert von d zugrundegelegt werden; sicherheitshalber kann man auch die tatsächliche Änderung der Funktionswerte bei einer Iteration in den Konvergenztest einbeziehen,

$$d < \varepsilon \qquad \text{und} \qquad F_k - F_{k+1} < \varepsilon \tag{8.58}$$

mit einem Wert $\varepsilon < 1$. Der Wert 0.01 für ε wird empfohlen; der genaue Wert sollte ohne Bedeutung sein, da die Konvergenz nahe der Lösung gut sein sollte.

Ein *Standardalgorithmus* für die Anwendung der Newton-Methode und anderer ähnlicher Methoden, die mit der Minimierung entlang einer Geraden im n-dimensionalen Raum arbeiten, ist in dem folgenden Kasten dargestellt.

Dieser Algorithmus kann als Standardalgorithmus für die Minimierung glatter Funktionen dienen; statt des Newton-Algorithmus kann dabei auch ein anderer Algorithmus zur Bestimmung des Suchvektors gewählt werden.

0. Man definiert den Anfangswert $\boldsymbol{x}_0$, berechnet $F(\boldsymbol{x}_0)$ und setzt $k = 0$.

1. Berechnung des Suchvektors $\boldsymbol{\Delta x}$. Der Newton-Suchvektor wird ermittelt durch die Lösung von $\boldsymbol{H}_k \boldsymbol{\Delta x}_N = -\boldsymbol{g}_k$.

2. Eindimensionale Minimierung. Die Funktion $F(\boldsymbol{x})$ wird in der Richtung des Suchvektors minimiert (*line search*), indem man die eindimensionale Funktion $f(z) = F(\boldsymbol{x}_k + z \cdot \boldsymbol{\Delta x}_N)$ minimiert.

3. Iteration. Man definiert $\boldsymbol{x}_{k+1} = \boldsymbol{x}_k + z_{\min} \cdot \boldsymbol{\Delta x}_N$, wobei $z_{\min}$ der Punkt des Minimums von $f(z)$ ist, und erhöht k um 1.

4. Konvergenztest. Die Konvergenz wird geprüft, wobei die Größe d und die Differenz $F_{k-1} - F_k$ zugrundegelegt werden. Wenn die Konvergenzkriterien erfüllt sind, schließt man mit $\boldsymbol{x}_k$ als Lösung ab; andernfalls geht man zurück zu Schritt 1; bei Erreichen einer Maximalzahl von Iterationen muß der Algorithmus abgebrochen werden, ohne daß das Minimum gefunden wurde.

Beispiel 8.3 Anpassung einer Exponentialfunktion an Daten.

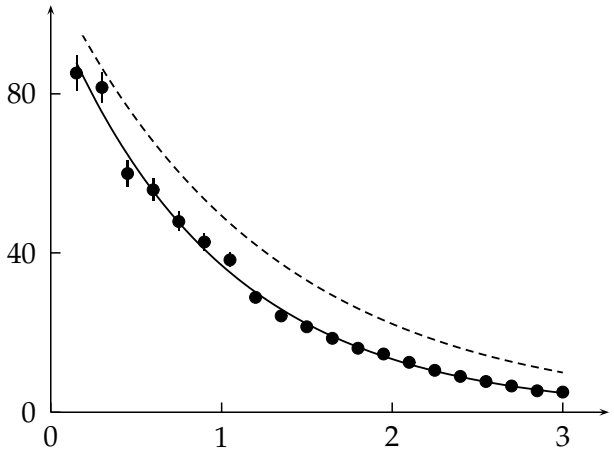


Abbildung 8.12: Anpassung einer Exponentialfunktion an 20 Datenpunkte. Die gestrichelte Kurve zeigt die Exponentialfunktion mit den Startwerten der Parameter.

Die Newton-Methode wird benutzt, um eine Exponentialfunktion $f(x_1, x_2) = x_1 \exp(-x_2 \cdot t)$ an Datenpunkte anzupassen. Die Abbildung 8.12 zeigt die Datenpunkte als Funktion der Variablen t. Als zu minimierende Funktion $F(\boldsymbol{x}) = F(x_1, x_2)$ wird bei diesem Problem eine Summe quadratischer Abweichungen gemäß der Methode der kleinsten Quadrate definiert. Der Parameter x_2, die *logarithmische* Steigung, steht im Exponenten, und bei solchen nichtlinearen Abhängigkeiten ist die quadratische Näherung nur in einem engen Bereich um das Minimum gut. Die Abbildung 8.12 zeigt als gestrichelte Kurve den exponentiellen Verlauf mit den gewählten Startwerten der Anpassung.

Die Konturlinien der Funktion $F(\boldsymbol{x}) = F(x_1, x_2)$ sind in Abbildung 8.13 dargestellt. Man erkennt angenähert Ellipsen um das Minimum bei etwa $(x_1, x_2) = (100, 1)$ herum. Im Bereich um den Startwert der Anpassung sind jedoch die Konturlinien nicht mehr Ellipsen, die Krümmung der Konturlinien zeigt nicht mehr in Richtung des Minimums. Die durchgezogenen Geraden zeigen die Richtung der Newton-Schritte für mehrere Iterationen. Die Punkte auf den Geraden zeigen, für welche Koordinaten die Funktion bei der eindimensionalen Minimierung in Richtung der Newton-Schritte berechnet wurde. Die erste Iteration führt zunächst zu einer sehr viel kleineren logarithmischen Steigung (x_2) und einem kleineren Faktor (x_1). Die nächsten Iterationen führen dann schnell zum Minimum.

Die ersten und zweiten Ableitungen der Funktion $F(x_1, x_2)$ sind in diesem Beispiel numerisch berechnet worden. Die Kreise in Abbildung 8.13 zeigen, an welchen Stellen die Funktion berechnet wurde. Die bei der numerischen Berechnung der Ableitungen benutzten Schrittweiten wurden dabei im Verlauf der Iterationen dem Problem angepaßt.

Das Beispiel zeigt, daß die Newton-Methode gute Startwerte der Parameter erfordert. Stehen keine guten Startwerte zur Verfügung, so ist es sinnvoll, einige Iterationen mit einem Suchverfahren, zum Beispiel mit der Simplex-Methode, auszuführen. Die Abbildung 8.14 zeigt das gleiche Problem wie Abbildung 8.13, jedoch ist zunächst, vom gleichem Startpunkt ausgehend, mit der Simplex-Methode eine angenäherte Minimierung durchgeführt worden; die gestrichelte Kurve zeigt, daß diese Methode bereits einen guten Näherungswert für die Lösung liefert. In dem engen Bereich um das Minimum herum ist dann die Newton-Methode effektiver; wenige Iterationen mit der Newton-Methode ergeben das Minimum mit großer Genauigkeit.

Methode des steilsten Abstiegs. Der Suchvektor $\boldsymbol{\Delta x}$ wird bei dieser Methode als negativer Gradientenvektor gewählt,

$$\boldsymbol{\Delta x}_{\mathrm{SD}} = -\boldsymbol{g}_k \,. \tag{8.59}$$

Obwohl diese Wahl natürlich erscheint und nur der Gradient erforderlich ist (und nicht die Hesse-Matrix), gibt es ernsthafte Nachteile. Im Gegensatz zur Newton-Methode ist die natürliche Schrittgröße nicht spezifiziert. Die Konvergenzrate ist nur linear; näherungsweise gilt der Aus-

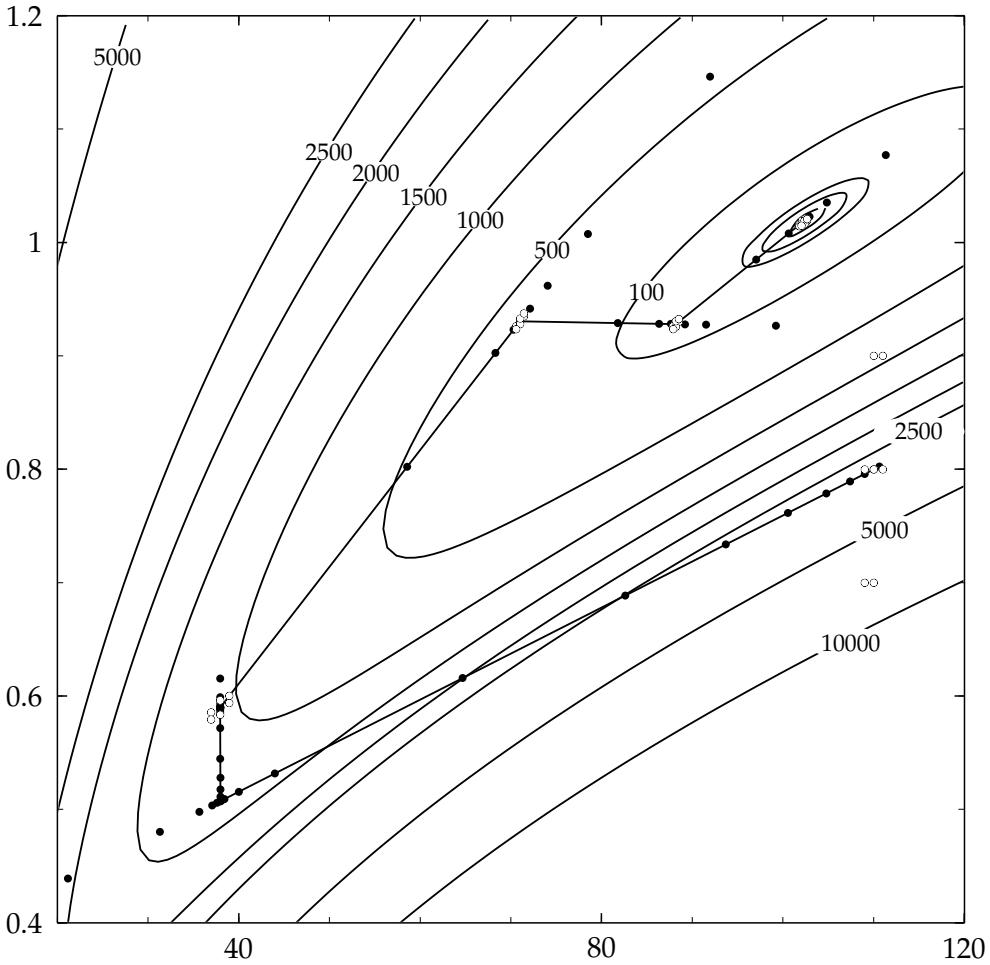


Abbildung 8.13: Minimierung der χ^2-Funktion für die Anpassung einer Exponentialfunktion mit dem Startwert $(x_1, x_2) = (110, 0.8)$. Die ausgefüllten Kreise sind Punkte, bei denen bei der eindimensionalen Minimierung längs der Newton-Richtung die χ^2-Funktion berechnet wurde. Die offenen Kreise sind Punkte, bei denen bei der numerischen Differentiation die χ^2-Funktion berechnet wurde. Um das Minimum mit einem Wert der χ^2-Funktion von 15.56 bei $(x_1, x_2) = (102.4, 1.018)$ sind die Konturlinien für eine, zwei und drei Standardabweichungen gezeichnet. Weiterhin sind die Konturlinien für Funktionswerte 100 ... 10000 gezeichnet.

druck

$$\frac{F(\boldsymbol{x}_{k+1}) - F(\boldsymbol{x}^*)}{F(\boldsymbol{x}_k) - F(\boldsymbol{x}^*)} = c \approx \left(\frac{\lambda_{\max} - \lambda_{\min}}{\lambda_{\max} + \lambda_{\min}}\right)^2 = \left(\frac{\kappa - 1}{\kappa + 1}\right)^2 \quad , \tag{8.60}$$

wobei $\lambda_{\max}$ und $\lambda_{\min}$ der größte und der kleinste Eigenwert und κ die Konditions-Zahl der Matrix $\boldsymbol{H}$ sind (siehe Gleichung (8.12)). Für einen großen Wert von κ wird die Konstante c nahe bei eins liegen, und die Konvergenz wird dementsprechend langsam sein.

Wenn die Hesse-Matrix bekannt ist, kann auch bei der Methode des steilsten Abstiegs eine

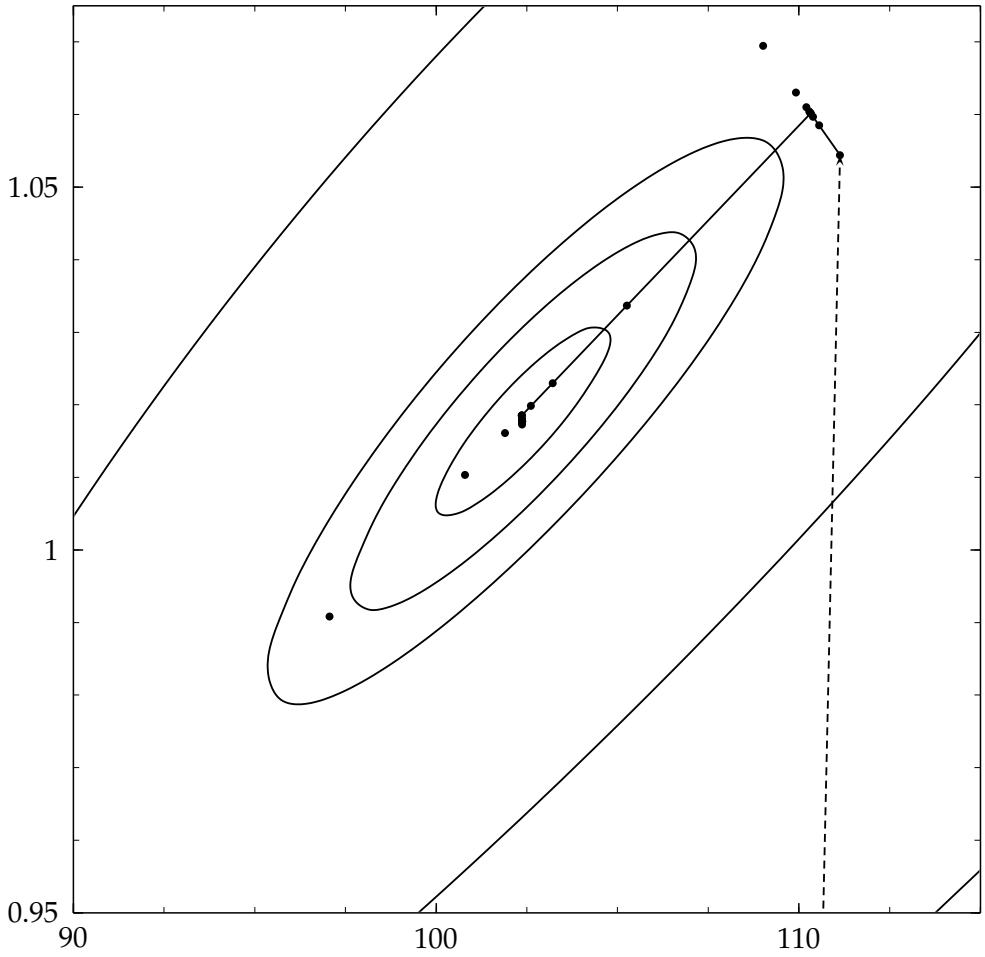


Abbildung 8.14: Minimierung der gleichen χ^2-Funktion wie in Abbildung 8.13 bei gleichem Startwert. Jedoch wird hier zunächst eine einfache Minimierung mit der Simplex-Methode durchgeführt, die insgesamt zu einem Schritt entsprechend der gestrichelten Linie führt. Die anschließende Minimierung mit der Newton-Methode braucht dann nur noch wenige Iterationen.

natürliche Schrittgröße berechnet werden:

$$\boldsymbol{\Delta x}_{\mathrm{SD}} = -\left(\frac{\boldsymbol{g}_k^T \boldsymbol{g}_k}{\boldsymbol{g}_k^T \boldsymbol{H}_k \boldsymbol{g}_k}\right) \boldsymbol{g}_k \quad . \tag{8.61}$$

Für eine quadratische Funktion $F(\boldsymbol{x})$ lautet dann die in der eindimensionalen Minimierung benutzte Funktion $f(z)$

$$f(z) = F(\boldsymbol{x}_k) + \frac{(\boldsymbol{g}_k^T \boldsymbol{g}_k)^2}{\boldsymbol{g}_k^T \boldsymbol{H}_k \boldsymbol{g}_k}\left(\frac{1}{2}z^2 - z\right) \tag{8.62}$$

(man vergleiche mit Gleichung (8.55)), die ihr Minimum bei $z = 1$ annimmt.

8.4.2 Modifizierte Newton-Methoden

Ist die Hesse-Matrix $\boldsymbol{H}$ nicht positiv-definit, braucht die quadratische Näherung der Funktion $F(\boldsymbol{x})$ kein Minimum zu haben, nicht einmal einen stationären Punkt, und die Standard

Newton-Methode muß *modifiziert* werden. Bei den meisten dieser Methoden wird eine positiv-definite Matrix $\overline{\boldsymbol{H}}$ durch eine kleine Veränderung von $\boldsymbol{H}$ konstruiert und eine Suchrichtung $\boldsymbol{\Delta x}_M$ durch

$$\overline{\boldsymbol{H}}\,\boldsymbol{\Delta x}_M = -\boldsymbol{g} \tag{8.63}$$

definiert. Diese Suchrichtung ist abwärts gerichtet wegen

$$\mathbf{g}^T\,\boldsymbol{\Delta x}_M = -\mathbf{g}^T\,\overline{\boldsymbol{H}}^{-1}\,\mathbf{g} < 0 \tag{8.64}$$

für eine positiv-definite Matrix $\overline{\boldsymbol{H}}$.

Das Problem kann anhand der spektralen Zerlegung von $\boldsymbol{H}$ behandelt werden. Nach Gleichung (3.47) ist

$$\boldsymbol{H} = \boldsymbol{U}\boldsymbol{D}\boldsymbol{U}^T = \sum_i \lambda_i \boldsymbol{u}_i \boldsymbol{u}_i^T \ , \tag{8.65}$$

wobei die Vektoren $\boldsymbol{u}_i$ die Eigenvektoren von $\boldsymbol{H}$ sind, und $\boldsymbol{D}$ die Diagonalmatrix ist mit den Eigenwerten λ_i von $\boldsymbol{H}$ in der Diagonalen. Mindestens einer der Eigenwerte ist negativ oder null, wenn $\boldsymbol{H}$ nicht positiv-definit ist. Eine der Methoden besteht nun darin, eine modifizierte Diagonalmatrix $\overline{\boldsymbol{D}}$ mit Diagonalelementen

$$\overline{\lambda}_i = \max\left(|\lambda_i|, \delta\right) \tag{8.66}$$

einzuführen, wobei δ eine positive Konstante ist; dann ist

$$\overline{\boldsymbol{H}} = \boldsymbol{U}\overline{\boldsymbol{D}}\boldsymbol{U}^T \ . \tag{8.67}$$

Die Methode ist zeitaufwendig, da sie die Diagonalisierung von $\boldsymbol{H}$ verlangt. Eine einfachere Modifizierung ist gegeben durch die Formel

$$\overline{\boldsymbol{H}} = \boldsymbol{H} + \alpha \boldsymbol{I}_n \qquad (\boldsymbol{I}_n = n \times n \,\text{Einheitsmatrix}) \ . \tag{8.68}$$

Für einen großen Wert des Faktors α bestimmt diese Gleichung eine Suchrichtung, die nahe der negativen Gradientenrichtung liegt, und für einen kleinen Wert des Faktors α liegt die Suchrichtung nahe bei der Newton-Richtung. In gewissem Sinn erlaubt der Faktor α, zwischen den beiden Methoden zu interpolieren. Die Bestimmung der richtigen Größe von α ist natürlich ein Problem. Die Wahl

$$\alpha > |\lambda^-_{\min}| \quad , \tag{8.69}$$

wobei $\lambda^-_{\min}$ der *negative* Eigenwert von $\boldsymbol{H}$ mit dem *größten* Betrag ist, würde zu einer positiv-definiten Matrix $\overline{\boldsymbol{H}}$ führen. Ein Wert

$$\alpha \leq \frac{|\boldsymbol{g}^T\boldsymbol{H}\boldsymbol{g}|}{\boldsymbol{g}^T\boldsymbol{g}} \tag{8.70}$$

wird durch den Wert der Schrittgröße gemäß der Gleichung (8.61) nahegelegt.

Eine schnelle und elegante modifizierte Newton-Methode basiert auf einer Modifikation der Cholesky-Faktorisierung [23].

8.4.3 Bestimmung der Hesse-Matrix

Bei der Newton- und der modifizierten Newton-Methode ist die Berechnung des Gradienten und der Hesse-Matrix für die jeweilige Näherung $\boldsymbol{x}_k$ von $\boldsymbol{x}^*$ notwendig. In der Praxis kann die Berechnung von $\boldsymbol{g}$ und $\boldsymbol{H}$ schwierig und aufwendig sein, daher sucht man nach Verfahren der näherungsweisen Berechnung. In diesem Kapitel werden Methoden diskutiert, die nur die direkte Berechnung des *Gradientenvektors* erfordern. Hieraus wird dann eine Näherung der Hesse-Matrix konstruiert.

Numerische Differentiation. Eine naheliegende Methode zur Berechnung der Hesse-Matrix ist die numerische Differentiation unter Verwendung des Gradienten. Mit der Näherung

$$\boldsymbol{g}(\boldsymbol{x}+\boldsymbol{\Delta x}) \approx \boldsymbol{g}(\boldsymbol{x}) + \boldsymbol{H}\boldsymbol{\Delta x} \tag{8.71}$$

erhält man mit $\boldsymbol{\Delta x} = h\,\boldsymbol{u}_i$, wobei $\boldsymbol{u}_i$ der Einheitsvektor in der i-ten Richtung und h eine Schrittgröße ist,

$$\boldsymbol{H}\boldsymbol{u}_i \approx \frac{1}{h}\left(\boldsymbol{g}(\boldsymbol{x}+h\boldsymbol{u}_i) - \boldsymbol{g}(\boldsymbol{x})\right) \quad . \tag{8.72}$$

Dies ist der i-te Spaltenvektor der Matrix $\boldsymbol{H}$. Verwendet man n Einheitsvektoren in der Richtung der Parameterachsen, so erhält man eine Näherung $\boldsymbol{H}_a$, die durch

$$\boldsymbol{H}_S = \frac{1}{2}\left(\boldsymbol{H}_a + \boldsymbol{H}_a^T\right) \tag{8.73}$$

symmetrisiert wird. Diese Rechnung erfordert $(n+1)$ Berechnungen des Gradienten und bewahrt die prinzipiellen Vorteile der direkten oder der modifizierten Newton-Methode.

Methoden mit numerischen Ableitungen erfordern eine sorgfältige Wahl der Schrittparameter h. Dies ist nicht einfach, da ein kleines h ungenaue Ergebnisse aufgrund von Rundungsfehlern liefert, aber andererseits ein kleines h notwendig ist, wenn $\boldsymbol{g}(\boldsymbol{x}_k)$ klein wird.

Methoden mit variabler Metrik. Die Methoden mit *variabler Metrik* (*Quasi-Newton-Methoden*) [16] benutzen die Änderung des Gradienten während der Iterationen, um eine Anfangsschätzung der Hesse-Matrix laufend zu verbessern. Man spricht von variabler Metrik, weil die Matrix $\boldsymbol{H}$, die zur Berechnung der Newton-Schritte benutzt wird, variabel ist.

Die Anfangsmatrix $\boldsymbol{H_0}$ kann die Einheitsmatrix oder eine durch numerische Methoden erhaltene Schätzung sein. Die Methoden mit variabler Metrik basieren auf der Näherung

$$\boldsymbol{g}(\boldsymbol{x}_k+\boldsymbol{\Delta x}) \approx \boldsymbol{g}(\boldsymbol{x}_k) + \boldsymbol{H}_k\boldsymbol{\Delta x} \quad . \tag{8.74}$$

Der Schritt $\boldsymbol{\Delta x}$ wird aus der Gleichung

$$\boldsymbol{H}_k\,\boldsymbol{\Delta x}_{\mathrm{VM}} = -\boldsymbol{g}_k \tag{8.75}$$

berechnet, und die nächste Näherung von $\boldsymbol{x}^*$ ergibt sich nach einer eindimensionalen Minimierung in der Richtung des Schrittes $\boldsymbol{\Delta x}_{\mathrm{VM}}$ als

$$\boldsymbol{x}_{k+1} = \boldsymbol{x}_k + z_{\min}\,\boldsymbol{\Delta x}_{\mathrm{VM}} \quad . \tag{8.76}$$

Jetzt läßt sich eine Matrix $\boldsymbol{U}_k$ zur Verbesserung der Näherung von $\boldsymbol{H}$ gemäß

$$\boldsymbol{H}_{k+1} = \boldsymbol{H}_k + \boldsymbol{U}_k \tag{8.77}$$

definieren, so daß die verbesserte Näherung der Hesse-Matrix genau die Beziehung

$$\boldsymbol{H}_{k+1}\left(\boldsymbol{x}_{k+1} - \boldsymbol{x}_k\right) = \boldsymbol{g}_{k+1} - \boldsymbol{g}_k \tag{8.78}$$

erfüllt. Die Beziehung kann man mit den Vektoren $\boldsymbol{\delta} = \boldsymbol{x}_{k+1} - \boldsymbol{x}_k$ und $\boldsymbol{\gamma} = \boldsymbol{g}_{k+1} - \boldsymbol{g}_k$ auch in der Form

$$\boldsymbol{H}_{k+1}\boldsymbol{\delta} = \boldsymbol{\gamma} \tag{8.79}$$

schreiben. Durch diese Forderung ist die Aktualisierungsmatrix $\boldsymbol{U}_k$ noch nicht eindeutig definiert; es gibt verschiedene Formeln, die eingehend untersucht worden sind. Die Formel von Broydon/Fletcher/Goldfarb/ Shanno (BFGS), die als die effektivste Formel gilt, lautet [12, 22, 24, 71]

$$\boldsymbol{H}_{k+1} = \boldsymbol{H}_k - \frac{(\boldsymbol{H}_k\boldsymbol{\delta})\,(\boldsymbol{H}_k\boldsymbol{\delta})^T}{\boldsymbol{\delta}^T\boldsymbol{H}_k\boldsymbol{\delta}} + \frac{\boldsymbol{\gamma}\boldsymbol{\gamma}^T}{\boldsymbol{\gamma}^T\boldsymbol{\delta}} \quad . \tag{8.80}$$

Die Matrizen sind positiv-definit, wenn und nur wenn

$$\boldsymbol{\gamma}^T\boldsymbol{\delta} > 0 \tag{8.81}$$

gilt, und dies muß bei der eindimensionalen Minimierung bei jeder Iteration erfüllt sein. Wenn in jeder Iteration eine genaue Minimierung längs der durch den Newton-Schritt definierten Geraden (line search) ausgeführt wird, so hat man bei einer quadratischen Funktion $F(\boldsymbol{x})$ (die Matrix $\boldsymbol{H}$ ist dann eine Konstante) nach n Iterationen

$$\boldsymbol{H}_{k+1} = \boldsymbol{H} \; ; \tag{8.82}$$

dabei wurden n linear unabhängige Suchvektoren erzeugt. Im allgemeinen werden die Elemente der angenäherten Matrix $\boldsymbol{H}$ nicht genau gegen die Elemente der wahren Hesse-Matrix konvergieren. Die Konvergenz der n-dimensionalen Minimierung jedoch ist superlinear, selbst wenn $\boldsymbol{H}$ nicht gegen die wahre Hesse-Matrix konvergiert.

Ein ernstes Problem bei der Anwendung von Methoden mit variabler Metrik ist, daß die Eigenschaft der angenäherten Hesse-Matrix, positiv-definit zu sein, durch Rundungsfehler verlorengehen kann. Es empfiehlt sich daher, eine *genaue* eindimensionale Minimierung durchzuführen. Wenn der Suchvektor durch Gleichung (8.75) mit eindimensionaler Minimierung (Gleichung (8.76)) definiert wird, dann lautet die BFGS-Formel

$$\boldsymbol{H}_{k+1} = \boldsymbol{H}_k + \frac{\boldsymbol{g}_k\boldsymbol{g}_k^T}{\boldsymbol{g}_k^T\,\boldsymbol{\Delta x}_{\mathrm{VM}}} + \frac{\boldsymbol{\gamma}\boldsymbol{\gamma}^T}{z_{\min}\cdot\boldsymbol{\gamma}^T\,\boldsymbol{\Delta x}_{\mathrm{VM}}} \, . \tag{8.83}$$

Es gibt auch entsprechende Formeln direkt für die inverse Matrix $\boldsymbol{H}^{-1}$ von $\boldsymbol{H}$.

8.4.4 Numerische Differentiation

Bei den bisher in diesem Kapitel besprochenen Methoden ist zumindest die Berechnung des Gradienten erforderlich. Auch dies kann numerisch geschehen, wobei jedesmal $(n+1)$ oder $(2n+1)$ Berechnungen der Funktion $F(\boldsymbol{x})$ erforderlich sind. Man kann sogar einen Schritt weiter gehen und die Hesse-Matrix mit endlichen Differenzen berechnen, wozu eine Anzahl von Funktionsberechnungen proportional zu n^2 erforderlich ist – und das ist sehr zeitaufwendig. Ein Kompromiß ist, daß man die Hesse-Matrix nur zu Beginn der Iteration berechnet (und vielleicht nach der Konvergenz), die ersten Ableitungen während der Iterationen, und dort die Näherung der Hesse-Matrix mit Methoden der variablen Metrik verbessert.

Numerische Fehler bei der Berechnung der Ableitungen beeinflussen die Optimierung wesentlich, und die Schrittgrößen in ihrer Berechnung müssen sorgfältig bedacht werden. Sind die numerischen Ableitungen bestimmt, erfolgt die Minimierung effizient mit superlinearer Konvergenz und ist im allgemeinen besser als bei Methoden ohne Ableitungen.

Das Problem der Schrittgröße bei der numerischen Berechnung von Ableitungen erfordert eine detaillierte Betrachtung. Selbst wenn die Ableitungen analytisch berechnet werden können, kann es vorteilhaft sein, sie zusätzlich numerisch zu berechnen und die Ergebnisse zu vergleichen. Langsame Konvergenz kann durch Fehler in der Berechnung der analytischen Ableitungen hervorgerufen sein; ein Vergleich mit numerischen Ableitungen ermöglicht oftmals, den Fehler zu lokalisieren.

Eine einfache Schätzung der ersten Ableitung einer Funktion $f(x)$ erfolgt über Vorwärts- oder Rückwärts-Differenzen,

$$f'(x) \approx \frac{f(x+h) - f(x)}{h} \qquad f'(x) \approx \frac{f(x) - f(x-h)}{h} \ , \tag{8.84}$$

und erfordert nur *eine* Funktionsberechnung zusätzlich zum Wert $f(x)$. Die wesentlichen Beiträge zum Fehler des geschätzten $f'(x)$ sind der *Abbruchfehler* Δ_t und der *Rundungsfehler* Δ_c. Der Abbruchfehler entsteht durch Vernachlässigung höherer Ableitungen in der Taylor-Reihe

$$f(x+h) = f(x) + hf'(x) + \frac{h^2}{2}f''(x) + \ldots \tag{8.85}$$

Der Beitrag des Abbruchfehlers zum geschätzten Wert der ersten Ableitung $f'(x)$ ist

$$\Delta_t = \frac{h}{2}f''(x+\zeta h) \approx \frac{h}{2}M_2 \qquad 0 < \zeta < 1 \ , \tag{8.86}$$

wobei M_2 eine Schranke für die zweite Ableitung ist. Der Rundungsfehler entsteht dadurch, daß sich die berechneten Funktionswerte $f(x), f(x+h), \ldots$ von den tatsächlichen Werten durch Rundungsfehler unterscheiden. Ist ε_f der relative Fehler $\Delta f/f$ eines errechneten Funktionswertes, so kann ein möglicher Fehler des geschätzten Wertes der ersten Ableitung $f'(x)$ durch

$$\Delta_c \approx \frac{2}{h}\varepsilon_f M_0 \tag{8.87}$$

abgeschätzt werden, wobei M_0 eine Schranke für den absoluten Funktionswert ist. Die beiden Beiträge zum Fehler des geschätzten Wertes von $f'(x)$ sind jeweils proportional und umgekehrt proportional zur Schrittgröße h. Der Wert von h, der die Summe beider Fehler (linear oder quadratisch addiert) minimiert, ist gegeben durch

$$h_1 = 2\sqrt{\frac{\varepsilon_f M_0}{M_2}} \ , \tag{8.88}$$

und beide Beiträge sind für diesen Wert annähernd gleich groß.

Ehe dies weiter verfolgt wird, werden *zentrale Differenzen-Formeln* behandelt. Für diese Formeln braucht man zwar *zwei* Funktionswerte (zusätzlich zu $f(x)$), doch haben sie den Vorteil, daß man zusätzlich eine Schätzung der zweiten Ableitung erhält,

$$f'(x) \approx \frac{f(x+h) - f(x-h)}{2h} \qquad f''(x) \approx \frac{f(x+h) - 2f(x) + f(x-h)}{h^2} \ . \tag{8.89}$$

In der zentralen Differenzen-Formel ist für $f'(x)$ der Abbruchfehler viel kleiner, da der Beitrag von $f''(x)$ verschwindet; der verbleibende Fehler ist proportional zu h^2 und zu $f'''(x)$. Für die Schätzung von $f''(x)$ ist der Abbruchfehler proportional zu h^2 und zu $f''''(x)$. Zentrale Differenzen-Formeln erlauben größere Schrittgrößen und halten den Beitrag von Rundungsfehlern klein. Die Schätzung der zweiten Ableitung erlaubt auch eine Schätzung von M_2 und damit die Berechnung von h_1 gemäß der Gleichung (8.88).

Es ist aufschlußreich, den Fehler des geschätzten Wertes von $f''(x)$ zu betrachten. Der relative Fehler infolge von Rundungsfehlern ist für eine Schrittgröße $h = \alpha h_1$ mit einem konstanten Faktor α

$$\frac{\Delta f''(x)}{f''(x)} \approx \frac{1}{\alpha^2} \,, \tag{8.90}$$

so daß also der Fehler für $h = h_1$ genauso groß ist wie der Wert selbst. Dies zeigt zudem, daß der Fehler des geschätzten $f'(x)$ aus Vorwärts- und Rückwärts-Differenzen mit h_1 als Schrittgröße relativ groß ist, wenn $f'(x)$ nahe bei null liegt, während der Fehler vernachlässigbar ist, wenn $f'(x)$ groß ist. Da während einer Minimierung der Wert der Ableitung natürlich klein werden wird (während der Wert der zweiten Ableitung allgemein eine Konstante $\neq 0$ werden wird), sollten zentrale Differenzen zumindest für die letzten Iterationen verwendet werden. Eine sicherere Methode ist, immer zentrale Differenzen zu benutzen, zum Beispiel mit $\alpha = 5$, denn damit hat man wenige % relative Fehler für das geschätzte $f''(x)$ und eine genaue Schätzung von $f'(x)$, auch wenn der Wert selbst klein ist.

Eine untere Grenze für den relativen Fehler ε_f dürfte die Rechner-Genauigkeit ε_M sein, die typisch von der Größenordnung 10^{-7} (einfache Genauigkeit) ist. Abbildung **??** zeigt Rundungsfehler bei der Berechnung einer einfachen Funktion mit einfacher Genauigkeit. Durch kleine Schritte bei der numerischen Ableitung kann ein völlig falsches Ergebnis entstehen. Die mit doppelter Genauigkeit berechnete Funktion zeigt in diesem Bereich keine Probleme. Sehr oft enthält die Funktion eine Summe vieler Terme, wodurch sich deutlich größere relative Fehler für die Summe ergeben können, besonders wenn die Vorzeichen der Terme verschieden sind. Es sollte zumindest die Summierung mit doppelter Genauigkeit vorgenommen werden, um nicht zusätzliche Fehler zu erhalten.

In einer typischen Anwendung geht es um die Minimierung einer Summe von Quadraten von Abweichungen zwischen experimentellen Daten y_i und Funktionswerten g_i, also eine Summe von Termen der Form $(g_i - y_i)^2$; eine Schätzung des relativen Fehlers eines Terms ist

$$\varepsilon_f = \frac{2\,\varepsilon_\mathrm{M}\,\bar{g}}{\sigma} \,, \tag{8.91}$$

wobei $\bar{g}$ und σ typische Werte für die Funktionswerte g_i und die Differenzen $(g_i - y_i)$ sind. Der relative Fehler kann also viel größer als ε_M sein, vor allem wenn die Daten genau sind. Es empfiehlt sich deshalb das folgende Vorgehen: Wenn die Daten sehr genau sind ($\bar{g}/\sigma$ groß, etwa > 50), sollte mit doppelter Genauigkeit gerechnet werden. In vielen Fällen jedoch wird das Verhältnis $\bar{g}/\sigma$ nicht sehr groß sein, etwa < 50, dann ist $\varepsilon_f \approx 100\,\epsilon_M$, und mit $\alpha = 5$ sollte die Schrittgröße

$$h_c = 100\sqrt{\frac{\varepsilon_M M_0}{M_2}} \tag{8.92}$$

für zentrale Differenzen einen relativen Fehler von wenigen % für das geschätzte $f''(x)$ haben.

Schließlich werden Formeln für die näherungsweise Berechnung der Elemente des Gradienten und der Hesse-Matrix im *mehrdimensionalen Fall* angegeben,

$$g_j \approx \frac{F(\boldsymbol{x} + h\boldsymbol{u}_j) - F(\boldsymbol{x} - h\boldsymbol{u}_j)}{2h} \tag{8.93}$$

$$H_{jj} \approx \frac{F(\boldsymbol{x} + h\boldsymbol{u}_j) - 2F(\boldsymbol{x}) + F(\boldsymbol{x} - h\boldsymbol{u}_j)}{h^2} \tag{8.94}$$

(die Vektoren $\boldsymbol{u}_j$ sind die Einheitsvektoren in j-Richtung). Es müssen $2n$ Funktionsberechnungen für eine mehrdimensionale Funktion von n Parametern durchgeführt werden. Die Berechnung der $(n^2 - n)/2$ verschiedenen Nicht-Diagonalelemente der Hesse-Matrix erfordert *eine* zusätzliche Funktionsberechnung für jedes Matrixelement

$$H_{jk} \approx \frac{F(\boldsymbol{x} + h_j\boldsymbol{u}_j + h_k\boldsymbol{u}_k) + F(\boldsymbol{x}) - F(\boldsymbol{x} + h_j\boldsymbol{u}_j) - F(\boldsymbol{x} + h_k\boldsymbol{u}_k)}{h_j h_k} \,. \tag{8.95}$$

Die Schrittgrößen zur Berechnung der Ableitungen mit zentralen Differenzen können während der Iterationen leicht dem Problem angepaßt werden, indem man die Werte der zweiten Ableitungen benutzt.

8.5 Gleichungen als Nebenbedingungen

Das allgemeine Problem mit Nebenbedingungen in der Form von Gleichungen ist

$$\boxed{\begin{array}{ll} \text{minimiere } F(\boldsymbol{x}), & \boldsymbol{x} \in R^n \\ h_j(\boldsymbol{x}) = 0 & j = 1, 2, \ldots m \end{array}} \,. \tag{8.96}$$

Zusätzlich zur Funktion $F(\boldsymbol{x})$ der n Parameter $\boldsymbol{x}$ existieren m Gleichungen $h_j(\boldsymbol{x}) = 0$ zwischen den Variablen als Nebenbedingungen, die bei einer Lösung erfüllt sein müssen. Jeder Punkt $\boldsymbol{x}$ des n-dimensionalen Parameterraums, der alle m Nebenbedingungen erfüllt, heißt ein *erlaubter* Punkt. Die Menge aller erlaubten Punkte heißt *erlaubter Bereich.* Ein erlaubter Punkt $\boldsymbol{x}^*$ ist optimal, wenn $F(\boldsymbol{x}^*) < F(\boldsymbol{x})$ für alle erlaubten $\boldsymbol{x}$ in der Nachbarschaft von $\boldsymbol{x}^*$ ist.

Wenn die Nebenbedingungen *linear* sind, können sie durch eine Matrixgleichung

$$\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b} \tag{8.97}$$

ausgedrückt werden, wobei $\boldsymbol{A}$ eine $m \times n$ Matrix ist und $\boldsymbol{b}$ ein m-Vektor. Wenn die Matrix den vollen Rang m hat, definieren die Nebenbedingungen einen $(n - m)$-dimensionalen Unterraum. Dasselbe gilt für nichtlineare differenzierbare Nebenbedingungen, wenn die Jacobi-Matrix $\boldsymbol{A}$ mit Elementen

$$A_{ji} = \frac{\partial h_j}{\partial x_i} \tag{8.98}$$

den vollen Rang m hat.

Manchmal können die Nebenbedingungen durch eine *Variablentransformation* vermieden werden. Zum Beispiel kann man statt der Variablen x_1, x_2 und x_3 mit der Nebenbedingung $x_1^2 + x_2^2 + x_3^2 = R^2$ (erlaubte Punkte auf einer Kugel mit Radius R) besser die zwei Winkel θ und φ benutzen (Transformation in Kugelkoordinaten). In vielen Anwendungen gibt es eine Nebenbedingung von der Form $\sum_{i=1}^{n} x_i = 1$. In diesem Fall kann man die Nebenbedingung vermeiden, indem man die Variable x_n durch $\left(1 - \sum_{i=1}^{n-1} x_i\right)$ ersetzt. Dadurch geht allerdings die symmetrische Behandlung aller Variablen verloren.

8.5.1 Lineare Nebenbedingungen

Man hat *lineare Nebenbedingungen* nach Gleichung (8.97) mit einer Matrix $\boldsymbol{A}$ vom Rang m.

Eine notwendige Bedingung dafür, daß der Vektor $\boldsymbol{x}$ auf ein lokales Minimum zeigt, kann mit Hilfe der *Lagrangeschen Multiplikatoren* formuliert werden. Dazu wird ein Vektor $\boldsymbol{\lambda}$ von m reellen Zahlen, den Lagrangeschen Multiplikatoren, eingeführt sowie die Lagrangefunktion

$$\Phi(\boldsymbol{x}, \boldsymbol{\lambda}) = F(\boldsymbol{x}) + (\boldsymbol{A}\boldsymbol{x} - \boldsymbol{b})^T \boldsymbol{\lambda} \,. \tag{8.99}$$

Mit der Abkürzung $\boldsymbol{g}(\boldsymbol{x})$ für den Gradienten $\nabla F(\boldsymbol{x})$ muß der optimale Punkt $(\boldsymbol{x}^*, \boldsymbol{\lambda}^*)$ die Gleichung

$$\nabla\Phi(\boldsymbol{x},\boldsymbol{\lambda})|_{\boldsymbol{x}^*,\boldsymbol{\lambda}^*} = \boldsymbol{g}(\boldsymbol{x}^*) + \boldsymbol{A}^T\boldsymbol{\lambda}^* = 0 \tag{8.100}$$

erfüllen. In einem stationären Punkt $\boldsymbol{x}^*$ muß der Gradient der Funktion $F(\boldsymbol{x})$ also eine Linearkombination der Zeilen der Matrix $\boldsymbol{A}$ sein,

$$\boldsymbol{g}(\boldsymbol{x}^*) = -\boldsymbol{A}^T\,\boldsymbol{\lambda}^* \,. \tag{8.101}$$

Jeder erlaubte Schritt $\boldsymbol{\Delta x}$ von einem erlaubten Punkt $\boldsymbol{x}$ (der $\boldsymbol{Ax} = \boldsymbol{b}$ genügt) zu einem anderen erlaubten Punkt muß der Gleichung $\boldsymbol{A\Delta x} = 0$ gehorchen. Der Unterraum erlaubter Schritte $\boldsymbol{\Delta x}$ kann durch eine $n \times (n-m)$ Matrix $\boldsymbol{Z}$ dargestellt werden; die $m \times (n-m)$ Produktmatrix der Matrizen $\boldsymbol{A}$ und $\boldsymbol{Z}$ muß null sein,

$$\boldsymbol{AZ} = 0 \,. \tag{8.102}$$

Wenn die Matrix $\boldsymbol{Z}$ bestimmt ist, dann ist jeder Schritt $\boldsymbol{\Delta x} = \boldsymbol{Z}z$ ein erlaubter Schritt für *jeden* $(n-m)$-Vektor z. Man setzt diesen Ausdruck in die Taylor-Entwicklung von $F(\boldsymbol{x})$ um den Punkt $\boldsymbol{x}^*$ ein,

$$F(\boldsymbol{x}+\boldsymbol{\Delta x}) = F(\boldsymbol{x}) + \boldsymbol{\Delta x}^T\boldsymbol{g}(\boldsymbol{x}) + \frac{1}{2}\,\boldsymbol{\Delta x}^T\boldsymbol{H}\,\boldsymbol{\Delta x} + \ldots \,. \tag{8.103}$$

Dies führt zu der folgenden Näherung als Funktion von z, mit der Abkürzung $\boldsymbol{g}^*$ für $\boldsymbol{g}(\boldsymbol{x}^*)$ und $\boldsymbol{H}^*$ für $\boldsymbol{H}(\boldsymbol{x}^*)$,

$$F(z) = F(\boldsymbol{x}^*) + z^T\left(\boldsymbol{Z}^T\boldsymbol{g}^*\right) + \frac{1}{2}z^T\left(\boldsymbol{Z}^T\boldsymbol{H}^*\boldsymbol{Z}\right)z + \ldots \,, \tag{8.104}$$

wobei $(\boldsymbol{Z}^T\boldsymbol{g})$ und $(\boldsymbol{Z}^T\boldsymbol{HZ})$ *projizierte Gradienten* und *projizierte Hesse-Matrix* genannt werden. Als notwendige Bedingung dafür, daß bei $\boldsymbol{x}$ ein lokales Minimum ist, muß $z^T\boldsymbol{Z}^T\boldsymbol{g}^*$ verschwinden für jeden $(n-m)$-Vektor z, und damit muß die Bedingung

$$\boldsymbol{Z}^T\boldsymbol{g}^* = 0 \tag{8.105}$$

erfüllt sein. Diese Bedingung ist äquivalent zu Gleichung (8.101), wie man durch Multiplikation dieser Gleichung mit $\boldsymbol{Z}^T$ von links und mit Hilfe von Gleichung (8.102) sieht.

Der vollständige Satz von *hinreichenden Bedingungen* ist also

1. $\boldsymbol{Ax}^* = \boldsymbol{b}$ (Erfüllung der Nebenbedingungen),
2. $\boldsymbol{g}(\boldsymbol{x}^*) = -\boldsymbol{A}^T\boldsymbol{\lambda}^*$ oder $\boldsymbol{Z}^T\boldsymbol{g}^* = 0$,
3. $\boldsymbol{Z}^T\boldsymbol{H}^*\boldsymbol{Z}$ positiv definit.

Die Hesse-Matrix $\boldsymbol{H}^*$ muß im übrigen nicht positiv definit sein, wohl aber die projizierte Hesse-Matrix.

Es werden nun zwei Lösungsmethoden vorgestellt.

Methode der Lagrangeschen Multiplikatoren. Diese Methode ist angebracht, wenn die Zahl m der Nebenbedingungen klein ist.

Aus der quadratischen Näherung von $F(\boldsymbol{x}_k)$ am Ort des Näherungsvektors $\boldsymbol{x}_k$ (Gleichung 8.103), eingesetzt in die Lagrange-Funktion (Gleichung 8.99), erhält man durch Differenzieren

$$\begin{pmatrix} \boldsymbol{H}_k & \boldsymbol{A}^T \\ \boldsymbol{A} & 0 \end{pmatrix}\begin{pmatrix} \boldsymbol{\Delta x} \\ \boldsymbol{\lambda} \end{pmatrix} \quad = \quad \begin{pmatrix} -\boldsymbol{g}_k \\ 0 \end{pmatrix} \,. \tag{8.106}$$

Multipliziert man die erste der Gleichungen (8.106) von links mit $\boldsymbol{A}\boldsymbol{H}_k^{-1}$ und benutzt zudem die zweite Gleichung $\boldsymbol{A}\boldsymbol{\Delta x} = 0$, so erhält man den Vektor der Lagrangeschen Multiplikatoren

$$\boldsymbol{\lambda}_k = -\left(\boldsymbol{A}\boldsymbol{H}_k^{-1}\boldsymbol{A}^T\right)^{-1}\boldsymbol{A}\boldsymbol{H}_k^{-1}\,\boldsymbol{g}_k\,. \tag{8.107}$$

Diese Lösung, eingesetzt in die erste der Gleichungen (8.106) ergibt den Schritt $\boldsymbol{\Delta x}$ gemäß

$$\begin{aligned}\boldsymbol{\Delta x} &= -\boldsymbol{H}_k^{-1}\left(\boldsymbol{g}_k + \boldsymbol{A}^T\boldsymbol{\lambda}_k\right) \\ &= \left(\boldsymbol{H}_k^{-1} - \boldsymbol{H}_k^{-1}\boldsymbol{A}^T(\boldsymbol{A}\boldsymbol{H}_k^{-1}\boldsymbol{A}^T)^{-1}\boldsymbol{A}\boldsymbol{H}_k^{-1}\right)(-\boldsymbol{g}_k)\,.\end{aligned} \tag{8.108}$$

Dieser Schritt $\boldsymbol{\Delta x}$ muß als Richtung bei der Minimierung entlang einer Geraden (line search) benutzt werden (Schritt 2 des Standardalgorithmus in Kapitel 8.4.1 auf Seite 192). Gleichung 8.108 zeigt, wie der Gradient durch die Lagrangeschen Multiplikatoren modifiziert wird. Der vor dem Vektor $(-\boldsymbol{g}_k)$ stehende Matrixausdruck ist die Kovarianzmatrix des Parametervektors $\boldsymbol{x}$.

Wenn der Näherungsvektor $\boldsymbol{x}_k$ kein erlaubter Punkt ist, dann ist der Vektor $\boldsymbol{h}_k = \boldsymbol{A}\boldsymbol{x}_k - \boldsymbol{b}$ nicht null, und dann muß auf der rechten Seite von Gleichung (8.106) die Null durch $-\boldsymbol{h}_k$ ersetzt werden. Außerdem muß Gleichung (8.107) geändert werden zu

$$\boldsymbol{\lambda}_k = -\left(\boldsymbol{A}\boldsymbol{H}_k^{-1}\boldsymbol{A}^T\right)^{-1}\left(\boldsymbol{A}\boldsymbol{H}_k^{-1}\,\boldsymbol{g}_k - \boldsymbol{h}_k\right)\,. \tag{8.109}$$

Der Schritt $\boldsymbol{\Delta x}$, der aus $\boldsymbol{\lambda}_k$ folgt, sollte auf den erlaubten Punkt $\boldsymbol{x}_k + \boldsymbol{\Delta x}$ führen.

Reduktion der Variablen. Wenn die Zahl m der Nebenbedingungen groß ist, empfiehlt sich diese Methode. Sie beruht auf der Matrix $\boldsymbol{Z}$ aus Gleichung (8.102). Mit dieser Matrix kann ein möglicher Schritt $\boldsymbol{\Delta x}$ ausgedrückt werden als

$$\boldsymbol{\Delta x} = \boldsymbol{Z}\boldsymbol{z}\,. \tag{8.110}$$

Die Matrix $\boldsymbol{Z}$ in dieser Gleichung kann man folgendermaßen erhalten: Die $m \times n$ Matrix $\boldsymbol{A}$ wird partitioniert in eine *nichtsinguläre* $m \times m$ Matrix $\boldsymbol{A}_1$ und eine $m \times (n-m)$ Matrix $\boldsymbol{A}_2$,

$$\boldsymbol{A} = \left(\begin{array}{c|c} \boldsymbol{A}_1 & \boldsymbol{A}_2 \end{array}\right)\,. \tag{8.111}$$

Ohne Einschränkung der Allgemeinheit kann man annehmen, daß die ersten m Spalten von $\boldsymbol{A}$ die nichtsinguläre Matrix $\boldsymbol{A}_1$ bilden (andernfalls müssen die Spalten von $\boldsymbol{A}$ umgeordnet werden). Dann ist die Matrix $\boldsymbol{Z}$

$$\boldsymbol{Z} = \begin{pmatrix} -\boldsymbol{A}_1^{-1}\boldsymbol{A}_2 \\ \boldsymbol{I}_{n-m} \end{pmatrix}\,, \tag{8.112}$$

denn das Produkt $\boldsymbol{A}\boldsymbol{Z}$ verschwindet nach Definition entsprechend $-\boldsymbol{A}_1\boldsymbol{A}_1^{-1}\boldsymbol{A}_2 + \boldsymbol{A}_2 = 0$. Setzt man $\boldsymbol{\Delta x}$ aus Gleichung (8.110) in die quadratische Näherung für $F(\boldsymbol{x}_k + \boldsymbol{\Delta x})$ ein, so erhält man den Ausdruck

$$\begin{aligned}F(\boldsymbol{x}_k + \boldsymbol{\Delta x}) &= F_k + \left(\boldsymbol{g}_k^T\boldsymbol{Z}\right)\boldsymbol{z} + \frac{1}{2}\boldsymbol{z}^T\left(\boldsymbol{Z}^T\boldsymbol{H}_k\boldsymbol{Z}\right)\boldsymbol{z} + \ldots \\ &= F_k + \boldsymbol{g}_{\boldsymbol{z}}^T\boldsymbol{z} + \frac{1}{2}\boldsymbol{z}^T\boldsymbol{H}_{\boldsymbol{z}}\boldsymbol{z}\end{aligned} \tag{8.113}$$

mit dem Gradienten $\boldsymbol{g}_{\boldsymbol{z}} = \boldsymbol{Z}^T\boldsymbol{g}_k$ und der Hesse-Matrix $\boldsymbol{H}_{\boldsymbol{z}} = \boldsymbol{Z}^T\boldsymbol{H}_k\boldsymbol{Z}$. Die Bedingung für einen stationären Punkt in dieser Näherung ist

$$\boldsymbol{g}_{\boldsymbol{z}} + \boldsymbol{H}_{\boldsymbol{z}}\boldsymbol{z} = 0\,, \tag{8.114}$$

und weil jeder Vektor $\boldsymbol{z}$ mit Gleichung (8.110) auf einen erlaubten Schritt $\boldsymbol{\Delta x}$ führt, ergibt sich

$$\boldsymbol{\Delta x} = \boldsymbol{Z z} = -\boldsymbol{Z H_z^{-1} g_z} = -\boldsymbol{Z}(\boldsymbol{Z}^T \boldsymbol{H}_k \boldsymbol{Z})^{-1} \boldsymbol{Z}^T \boldsymbol{g}_k \ . \tag{8.115}$$

Die Lagrangeschen Multiplikatoren folgen aus Gleichung (8.106) zu

$$\boldsymbol{\lambda}_k = -(\boldsymbol{A}\boldsymbol{A}^T)^{-1} \boldsymbol{A}(\boldsymbol{g}_k + \boldsymbol{H}_k \boldsymbol{\Delta x}) \ . \tag{8.116}$$

Gleichungen (8.108) und (8.115) für die Berechnung des Schrittes sind äquivalent. Um die Konvergenz zu prüfen (Schritt 4 des Standardalgorithmus auf Seite 192), muß ein Test auf die Bedingungen (8.97) durchgeführt werden. Bei einem erlaubten Startwert $\boldsymbol{x}_0$ sind die Bedingungen immer erfüllt.

8.5.2 Nichtlineare Nebenbedingungen

Das Minimierungsproblem möge m nichtlineare Nebenbedingungen $h_j(\boldsymbol{x})$ für die Gleichung (8.96) haben. Die Lagrangefunktion des Problems ist

$$\Phi(\boldsymbol{x}, \boldsymbol{\lambda}) = F(\boldsymbol{x}) + \boldsymbol{h}^T \boldsymbol{\lambda} \ . \tag{8.117}$$

Das Problem erfordert eine iterative Lösung, die zu kleineren Funktionswerten $F(\boldsymbol{x})$ führt und gleichzeitig den Vektor $\boldsymbol{x}$ näher an die Lösung der Nebenbedingungen bringt.

Kostenmethode. Die Kostenmethode reduziert das Problem auf ein Problem ohne Nebenbedingungen. Man minimiert den Ausdruck

$$P(\boldsymbol{x}, \rho) = F(\boldsymbol{x}) + \frac{\rho}{2} \boldsymbol{h}^T(\boldsymbol{x}) \boldsymbol{h}(\boldsymbol{x}) \ , \tag{8.118}$$

der einen (quadratischen) Kostenfaktor $\boldsymbol{h}^T(\boldsymbol{x})\boldsymbol{h}(\boldsymbol{x})$ mit einem Kostenparameter ρ enthält. Die Lösung hängt von dem Parameter ρ ab. Sie wird bezeichnet mit $\boldsymbol{x}^*(\rho)$ und wird im allgemeinen näher an der wahren Lösung liegen. Man kann zeigen, daß

$$\lim_{\rho \to \infty} \boldsymbol{x}^*(\rho) = \boldsymbol{x}^* \ .$$

Dies sollte einen jedoch nicht veranlassen, immer größere Werte von ρ zu wählen, weil dann die Konditionszahl κ der Hesse-Matrix von $P(\boldsymbol{x}, \rho)$ zunimmt und die Hesse-Matrix schließlich singulär wird. Im Minimum von $P(\boldsymbol{x}, \rho)$ ist die Bedingung

$$\boldsymbol{g} + \rho \boldsymbol{A}^T \boldsymbol{h} = 0 \tag{8.119}$$

erfüllt, wobei die Matrix $\boldsymbol{A}$ die Jacobi-Matrix der Nebenbedingungen ist (siehe Gleichung (8.98)). Durch Vergleich mit der Lagrangebedingung (8.101) erhält man einen Schätzwert der Lagrangeschen Multiplikatoren aus

$$\boldsymbol{\lambda} = \rho \, \boldsymbol{h}(\boldsymbol{x}^*(\rho)) \ . \tag{8.120}$$

Methode der Lagrangeschen Multiplikatoren. Die Schritte $\boldsymbol{\Delta x}$ werden in einem iterativen Verfahren ähnlich wie im vorhergehenden Fall aus den linearisierten Nebenbedingungen berechnet. Aus

$$\boldsymbol{h}(\boldsymbol{x}_k + \boldsymbol{\Delta x}) \approx \boldsymbol{h}(\boldsymbol{x}_k) + \boldsymbol{A}_k \boldsymbol{\Delta x} \ , \tag{8.121}$$

wobei $\boldsymbol{A}_k$ die Jacobi-Matrix beim Iterationsschritt $\boldsymbol{x}_k$ ist, folgt die Bedingung

$$\boldsymbol{A}_k \boldsymbol{\Delta x} = -\boldsymbol{h}_k \, . \tag{8.122}$$

Die quadratische Näherung der Lagrangefunktion (8.117) führt unter Weglassen konstanter Terme auf

$$\Phi(\Delta\boldsymbol{x}, \boldsymbol{\lambda}) = \boldsymbol{g}_k^T \Delta\boldsymbol{x} + \frac{1}{2}\Delta\boldsymbol{x}^T \boldsymbol{W}_k \Delta\boldsymbol{x} + \Delta\boldsymbol{x}^T \boldsymbol{A}_k^T \boldsymbol{\lambda} + \dots \tag{8.123}$$

Diese Funktion muß minimiert werden, wobei die Hesse-Matrix $\boldsymbol{W}_k$ gegeben ist durch

$$\boldsymbol{W}_k = \boldsymbol{H}_k + \sum_{j=1}^{m} \lambda_j \boldsymbol{K}_j(\boldsymbol{x}_k) \ , \tag{8.124}$$

wobei $\boldsymbol{K}_j(\boldsymbol{x})$ die Hesse-Matrix der Nebenbedingungen ist und die λ_j die Näherungen der Lagrangeschen Multiplikatoren sind. Die Bedingung für einen stationären Punkt in dieser Näherung ist $\boldsymbol{W}_k \Delta\boldsymbol{x} + \boldsymbol{A}_k^T \boldsymbol{\lambda} = -\boldsymbol{g}_k$; sie kann mit Gleichung (8.122) zu einer Matrixgleichung kombiniert werden, ähnlich wie Gleichung (8.106),

$$\begin{pmatrix} \boldsymbol{W}_k & \boldsymbol{A}_k^T \\ \boldsymbol{A}_k & 0 \end{pmatrix} \begin{pmatrix} \Delta\boldsymbol{x} \\ \boldsymbol{\lambda} \end{pmatrix} \quad = \quad \begin{pmatrix} -\boldsymbol{g}_k \\ -\boldsymbol{h}_k \end{pmatrix} \ . \tag{8.125}$$

Ihre Lösung ist analog zu den Gleichungen (8.107) und (8.108)

$$\begin{aligned} \boldsymbol{\lambda}_k &= -(\boldsymbol{A}_k \boldsymbol{W}_k^{-1} \boldsymbol{A}_k^T)^{-1} (\boldsymbol{A}_k \boldsymbol{W}_k^{-1} \boldsymbol{g}_k - \boldsymbol{h}_k) \\ \Delta\boldsymbol{x} &= -\boldsymbol{W}_k^{-1} \left(\boldsymbol{g}_k + \boldsymbol{A}_k^T \boldsymbol{\lambda}_k \right) \ . \end{aligned} \tag{8.126}$$

Wenn der Schritt $\Delta\boldsymbol{x}$ im Standardalgorithmus von Kapitel 8.4.1 benutzt wird, dann sollte eine Minimierung längs der durch den Schritt definierten Geraden (line search) durchgeführt werden (Schritt 2 des Standardalgorithmus auf Seite 192). Unter Umständen sollte man bei den ersten Iterationen eine quadratische Kostenfunktion benutzen. Die Konvergenz ist meist nur in einer kleinen Umgebung der Lösung gut. Einen Startwert $\boldsymbol{x}_0$ kann man durch die Kostenfunktionsmethode erhalten, womit der Algorithmus robuster wird.

8.6 Ungleichungen als Nebenbedingungen

Das allgemeine Minimierungsproblem mit Ungleichungen ist gegeben durch

$$\boxed{\begin{array}{ll} \text{minimiere } F(\boldsymbol{x}) \, , & \boldsymbol{x} \in R^n \\ h_j(\boldsymbol{x}) \geq 0 & j = 1, 2, \dots m \end{array}} \ . \tag{8.127}$$

Im optimalen Punkt müssen die m Ungleichungen $h_j(\boldsymbol{x}) \geq 0$ gelten, wobei die Bedingungen entweder *aktiv* (=-Zeichen) oder *inaktiv* (>-Zeichen) sein können.

Lineare Ungleichungen werden formuliert durch eine $m \times n$ Matrix $\boldsymbol{A}$ und einen m-Vektor $\boldsymbol{b}$ in der Ungleichung

$$\boldsymbol{A}\boldsymbol{x} - \boldsymbol{b} \geq 0 \qquad \text{oder} \qquad \boldsymbol{a}_j^T \boldsymbol{x} - b_j \geq 0 \quad j = 1, 2, \dots m \ , \tag{8.128}$$

wobei $\boldsymbol{a}_j^T$ der j-te Zeilenvektor der Matrix $\boldsymbol{A}$ ist.

Probleme mit Ungleichungen sind meist komplizierter als Probleme mit Gleichungen als Nebenbedingungen, außer in ganz einfachen Fällen wie Schranken. Die Einführung von Schranken ist oft nötig, zum Beispiel wenn ein anzupassender Parameter (etwa eine Masse) nur positive Werte annehmen darf.

Ungleichungen, und besonders *Schranken* an Variable, können im Prinzip durch Transformationen eliminiert werden. Trigonometrische Funktionen werden dabei oft verwendet, um ein endliches Parameterintervall in ein unendliches zu verwandeln und damit eine Schranke zu umgehen. Oft erzeugt diese Methode aber mehr Probleme als sie löst; zum Beispiel können Sattelpunkte entstehen, ebenso können Rundungsfehler und eine verzerrte Hesse-Matrix für Schwierigkeiten sorgen.

Manchmal können aber Transformationen ein Problem tatsächlich vereinfachen. Zum Beispiel kann ein Problem mit einem n-dimensionalen Parametervektor und einer Ungleichung der Form

$$h_j(\boldsymbol{x}) \geq 0 \tag{8.129}$$

folgendermaßen transformiert werden: Eine zusätzliche Dummy-Variable x_{n+1} wird eingeführt. Sie soll nicht-negativ sein, und die Ungleichung (8.129) wird ersetzt durch die beiden Bedingungen

$$h_j(\boldsymbol{x}) - x_{n+1} = 0 \qquad\qquad x_{n+1} \geq 0\ . \tag{8.130}$$

Dadurch wird ein Problem mit nichtlinearen Ungleichungen in eines mit nichtlinearen Nebenbedingungen und linearen Ungleichungen transformiert, das eventuell leichter lösbar ist.

8.6.1 Optimierungsbedingungen

Es wird nur der Fall linearer Ungleichungen (8.128) behandelt. Ungleichungen mit dem $>$ Zeichen

$$\boldsymbol{a}_j^T \boldsymbol{x} > b_j \tag{8.131}$$

an einem möglichen Punkt werden *inaktiv* genannt und erlauben Schritte in jede Richtung. Ungleichungen, welche an einem erlaubten Punkt die Bedingung mit dem $=$-Zeichen erfüllen,

$$\boldsymbol{a}_j^T \boldsymbol{x} = b_j\ , \tag{8.132}$$

heißen *aktiv*; in diesem Fall sind erlaubte Schritte eingeschränkt entweder durch

$$\boldsymbol{a}_j^T \boldsymbol{\Delta x} = 0 \qquad \text{(gebundene Richtung)} \tag{8.133}$$

oder durch

$$\boldsymbol{a}_j^T \boldsymbol{\Delta x} > 0 \qquad \text{(nicht-gebundene Richtung)}\ . \tag{8.134}$$

Im zweiten Fall ist die Nebenbedingung unwirksam. Im Fall des $=$-Zeichens können die Ungleichungen folgendermaßen geschrieben geschrieben werden, mit einer Matrix $\widehat{\boldsymbol{A}}$ und einem Vektor $\widehat{\boldsymbol{b}}$

$$\widehat{\boldsymbol{A}} \boldsymbol{x} = \widehat{\boldsymbol{b}}\ . \tag{8.135}$$

Dabei wählt man gewisse Zeilen der Matrix $\boldsymbol{A}$ und des Vektors $\boldsymbol{b}$ aus, entsprechend den aktiven Bedingungen.

Aus Gründen analog zum Fall mit linearen Nebenbedingungen in Kapitel 8.5.1 ist ein erlaubter Punkt $\boldsymbol{x}^*$ ein stationärer Punkt in allen Richtungen, wenn

$$\boldsymbol{g}^* = -\widehat{\boldsymbol{A}}^T \boldsymbol{\lambda}^*\ , \tag{8.136}$$

wobei die Zahl der Lagrangeschen Multiplikatoren (Komponenten des Vektors $\boldsymbol{\lambda}^*$) gleich der Zahl der aktiven Bedingungen ist. Jedoch ist der Punkt $\boldsymbol{x}^*$ nicht optimal, wenn es eine Richtung gibt, für die $F(\boldsymbol{x})$ kleiner wird. Deshalb muß für alle $\boldsymbol{\Delta x}$, für die $\widehat{\boldsymbol{A}}\,\boldsymbol{\Delta x} = 0$ gilt, die Bedingung

$$\boldsymbol{g}^{*T}\boldsymbol{\Delta x} > 0 \tag{8.137}$$

erfüllt sein. Andernfalls gäbe es eine Richtung abwärts, welche *nicht* diese Bedingungen verletzt. Wegen $\boldsymbol{g}^* = -\boldsymbol{A}^T\,\boldsymbol{\lambda}^*$ bedeutet dies

$$\boldsymbol{g}^{*T}\boldsymbol{\Delta x} = -\lambda_1^*\widehat{\boldsymbol{a}}_1^T\boldsymbol{\Delta x} - \lambda_2^*\widehat{\boldsymbol{a}}_2^T\boldsymbol{\Delta x} - \ldots > 0 \tag{8.138}$$

mit $\widehat{\boldsymbol{a}}_j^T\,\boldsymbol{\Delta x} > 0$. Daraus folgt, daß die Bedingungen $\lambda_j^* < 0$ erfüllt sein müssen; *alle* Lagrangeschen Multiplikatoren müssen negativ sein.

Der vollständige Satz von *hinreichenden Bedingungen* kann analog zum Fall mit Gleichungen als Nebenbedingungen (Kapitel 8.5) hergeleitet werden mit einer durch $\widehat{\boldsymbol{A}}\,\boldsymbol{Z} = 0$ definierten Matrix $\boldsymbol{Z}$. Die hinreichenden Bedingungen sind:

1. $\boldsymbol{A}\boldsymbol{x}^* \geq \boldsymbol{b}$ mit $\widehat{\boldsymbol{A}}\boldsymbol{x}^* = \widehat{\boldsymbol{b}}$ (Bedingungen erfüllt),
2. $\boldsymbol{g}^* = -\widehat{\boldsymbol{A}}^T\boldsymbol{\lambda}^*$ oder $\boldsymbol{Z}^T\boldsymbol{g}^* = 0$,
3. $\boldsymbol{\lambda}^* < 0$ für alle Lagrangeschen Multiplikatoren,
4. $\boldsymbol{Z}^T\boldsymbol{H}^*\boldsymbol{Z}$ positiv definit.

8.6.2 Schranken für die Variablen

Hier wird die Minimierung einer Funktion für den einfachen Fall von Schranken für die Variablen behandelt

$$\boxed{\begin{array}{ll} \text{minimiere } F(\boldsymbol{x})\,, & \boldsymbol{x} \in R^n \\ x_i^{(l)} \leq x_i \leq x_i^{(u)} & i = 1, 2, \ldots n \end{array}}\,. \tag{8.139}$$

Wenn für alle n Parameter beidseitige Schranken vorgegeben sind, gibt es insgesamt $m = 2n$ Ungleichungen. Für die Berechnung des Schrittes $\boldsymbol{\Delta x}$ müssen aber nur die für die jeweilige Variable aktiven Beschränkungen berücksichtigt werden. Notwendige Bedingungen für lineare Ungleichungen findet man in Kapitel 8.6.1. Wegen der speziellen Struktur des Problems hat die Matrix $\widehat{\boldsymbol{A}}$ der aktiven Bedingungen als Zeilen die (eventuell mit negativen Vorzeichen versehenen) Zeilen der Einheitsmatrix, und die Bedingungen können in der einfachen Form

$$x_i - x_i^{(l)} \geq 0 \qquad\qquad -x_i + x_i^{(u)} \geq 0 \tag{8.140}$$

geschrieben werden. Wenn die Komponente x_i an der unteren Schranke gleich $x_i^{(l)}$ ist, dann ist die entsprechende Zeile der Matrix $\widehat{\boldsymbol{A}}$ die *positive* i-te Zeile der Einheitsmatrix, und wenn die Komponente x_i an der oberen Schranke $x_i^{(u)}$ ist, dann ist die entsprechende Zeile der Matrix $\widehat{\boldsymbol{A}}$ die *negative* i-te Zeile der Einheitsmatrix.

Die Lösung des Problems folgt dem Standardalgorithmus von Kapitel 8.4.1 auf Seite 192. Der Anfangspunkt $\boldsymbol{x}_0$ muß ein erlaubter Punkt sein (er muß innerhalb aller Schranken liegen). Der Vektor $\boldsymbol{\Delta x}$ kann aufgeteilt werden in einen Vektor $\boldsymbol{\Delta x}_{\text{frei}}$ von Variablen, die nicht an einer Schranke liegen, und einen Vektor $\boldsymbol{\Delta x}_{\text{fixiert}}$ von Variablen an einer Schranke. Der Gradient und die Hesse-Matrix werden ebenfalls entsprechend aufgeteilt. In der Praxis bedeutet das in der Regel ein Umordnen der Matrizen. Die Matrix-Gleichung für den Schritt ist dann

$$\begin{pmatrix} \boldsymbol{H}_{\text{frei}} & \boldsymbol{H}_{\text{ff}}^T \\ \boldsymbol{H}_{\text{ff}} & \boldsymbol{H}_{\text{fixiert}} \end{pmatrix} \begin{pmatrix} \boldsymbol{\Delta x}_{\text{frei}} \\ \boldsymbol{\Delta x}_{\text{fixiert}} \end{pmatrix} = \begin{pmatrix} -\boldsymbol{g}_{\text{frei}} \\ -\boldsymbol{g}_{\text{fixiert}} \end{pmatrix}\,. \tag{8.141}$$

Die Matrix $\boldsymbol{H}_{\text{ff}}$ ist Teil der Hesse-Matrix, mit Zeilen, die den fixierten Komponenten, und Spalten, die den freien Komponenten entsprechen. Wenn $\boldsymbol{\Delta x}_{\text{fixiert}} = 0$ ist, kann ein möglicher Schritt $\boldsymbol{\Delta x} = \boldsymbol{\Delta x}_{\text{frei}}$ berechnet werden aus

$$\boldsymbol{H}_{\text{frei}} \boldsymbol{\Delta x}_{\text{frei}} = -\boldsymbol{g}_{\text{frei}} \quad , \tag{8.142}$$

wobei die Hesse-Matrix $\boldsymbol{H}_{\text{frei}}$ und der Gradient $\boldsymbol{g}_{\text{frei}}$ nur den freien Komponenten von $\boldsymbol{x}$ entsprechen.

Ehe dieser Schritt $\boldsymbol{\Delta x}$ im Geraden-Suchalgorithmus verwendet wird, ist zu prüfen, ob eine oder mehrere der festen Variablen frei geworden sind. Wie im vorigen Abschnitt gezeigt, müssen die Lagrangeschen Multiplikatoren negativ sein. Die spezielle Form der Bedingungen (die Matrix $\widehat{\boldsymbol{A}}$ besteht aus negativen bzw. positiven Zeilen der Einheitsmatrix) erlaubt eine einfache Schätzung für die Komponenten von $\boldsymbol{\lambda}$, die den fixierten Komponenten des Gradienten $\boldsymbol{g}_{\text{fixiert}}$ oder des modifizierten Gradienten entsprechen,

$$\boldsymbol{\gamma} = \boldsymbol{g}_{\text{fixiert}} + \boldsymbol{H}_{\text{ff}} \boldsymbol{\Delta x}_{\text{frei}} \quad . \tag{8.143}$$

Wenn die Komponente γ_j des modifizierten Gradienten der (fixierten) Variablen x_i an ihrer unteren Schranke entspricht, dann ist der entsprechende Lagrangesche Multiplikator $\lambda_j = -\gamma_j$, und wenn die Variable x_i an ihrer oberen Schranke ist, dann ist der entsprechende Lagrangesche Multiplikator $\lambda_j = +\gamma_j$. Entsprechend den Bedingungen 8.138 in Kapitel 8.6.1 müssen alle Lagrangeschen Multiplikatoren negativ sein. Jede Komponente, die zu einem positiven Lagrangeschen Multiplikator gehört, muß frei werden und $\boldsymbol{x}_{\text{frei}}$ muß neu berechnet werden.

Bei der Minimierung längs einer Geraden im n-dimensionalen Raum (line search) in Schritt 2 des Standardalgorithmus auf Seite 192 müssen die Schranken der freien Komponenten beachtet werden. Wenn nach der Geradensuche eine freie Komponente an eine Schranke stößt, ist diese Komponente bei der nächsten Iteration als fixiert zu betrachten.

Die Minimierung einer Funktion mit Schranken als Nebenbedingungen erfordert also nur geringfügige Änderungen des normalen Algorithmus für Probleme ohne Nebenbedingungen.

9 Prüfung von Hypothesen

9.1 Prüfung einer einzelnen Hypothese

9.1.1 Allgemeine Begriffe

Dieses Kapitel beschäftigt sich mit der folgenden Fragestellung: Ist eine Menge (Stichprobe) von Daten statistisch vereinbar mit einer gegebenen Annahme (*Hypothese*)? Zum Beispiel könnte man fragen, ob eine Menge von Zahlen den Mittelwert null hat, wie zum Beispiel aufgrund von Symmetriebetrachtungen erwartet werden könnte, oder ob eine Menge von Daten vereinbar mit einer Gaußschen Wahrscheinlichkeitsdichte ist. Tatsächlich kann durch eine Prüfung (einen Test) einer Hypothese deren Gültigkeit nicht bewiesen werden, sehr wohl aber kann sie aufgrund statistischer Betrachtungen verworfen werden. Andererseits steigt die Wahrscheinlichkeit der Gültigkeit einer Hypothese mit zunehmender Genauigkeit der Daten, und man gewinnt Vertrauen in sie.
Der Grad der statistischen Verträglichkeit wird durch die Angabe von Konfidenzniveaus und von Konfidenzgrenzen quantifiziert. Das Konfidenzniveau oder die *Konfidenz* gibt die Wahrscheinlichkeit dafür an, daß statistische Fluktuationen eine Abweichung von den Erwartungen verursacht haben, die kleiner oder gleich der von den Daten gezeigten Abweichung ist (siehe auch Kapitel 6.7).

Beispiel 9.1 Hellsehen.

Für diese Prüfung wird eine Person gebeten, die Augenzahl eines zufällig geworfenen Würfels vorherzusagen. Ein Hellseher sollte in der Lage sein, eine Anzahl korrekter Vorhersagen zu treffen, die deutlich oberhalb der durch puren Zufall erwarteten liegt. Die Hypothese lautet: Hellsichtigkeit gibt es. Man nehme an, daß von 6000 Versuchen der Ausgang von 1050 korrekt vorhergesagt wurde. Welche Aussagen können über die Hypothese getroffen werden?

Ist die Hypothese falsch, so folgt die Anzahl der korrekten Vorhersagen einer Binomialverteilung, die in diesem Fall durch eine Gauß-Verteilung genähert werden kann, mit dem Mittelwert $6000/6 = 1000$ und der Standardabweichung $\sigma = \sqrt{6000/6 \cdot (1-1/6)} = 28.9$. Die Wahrscheinlichkeit, eine Abweichung $\leq s$ vom Mittelwert zu erhalten, ist (siehe Gleichung (4.19))

$$P = \frac{1}{\sqrt{2\pi}\sigma} \int_{-\infty}^{s} \exp(-x^2/2\sigma^2)\,\mathrm{d}x = \frac{1}{2}\left[1 + \mathrm{erf}\left(\frac{s}{\sqrt{2}\sigma}\right)\right] .$$

Mit $s = 1050 - 1000 = 50$ und $\sigma = 28.9$ erhält man ein Konfidenzniveau $P = 0.958$; mit anderen Worten, die Wahrscheinlichkeit, so viele oder noch mehr korrekte Vorhersagen durch Zufall zu treffen, ist 4.2%. Offensichtlich kann man mit dieser Messung nicht *beweisen*, daß es Hellsichtigkeit gibt. Bei welchem Signifikanzniveau würde ein Zweifler ins Grübeln kommen? Die Wahrscheinlichkeit, das Ergebnis durch Zufall zu bekommen, müßte wohl in diesem Fall sehr viel kleiner sein, üblich sind Werte von 1/100 bis 1/1000.

Einseitige und beidseitige Konfidenzgrenzen. In dem vorhergehenden Beispiel 9.1 wurden nur Abweichungen der Daten von den Erwartungen in einer Richtung betrachtet. Dies nennt man einen einseitigen statistischen Test mit einseitigen Konfidenzgrenzen. Häufig wird jedoch nur geprüft, ob Daten mit einer Hypothese verträglich sind, wobei die Richtung der Abweichung keine Rolle spielt. Dies nennt man einen beidseitigen Test mit beidseitigen Konfidenzgrenzen.
Formal kann eine ein- oder mehrdimensionale Testgröße t definiert werden, die anhand der Daten berechnet wird und ein Maß für den Grad der Abweichung der Daten von der Hypothese sein sollte. Sie sollte dem Betrag nach klein sein, wenn die Hypothese zutrifft. In dem

vorhergehenden Beispiel 9.1 ist die Testgröße $t = N_c - N_0$, wobei N_c die Anzahl korrekter Vorhersagen und N_0 die mittlere Anzahl korrekter Vorhersagen bei zufälligem Verhalten ist. Wenn $f(t)$ die Wahrscheinlichkeitsdichte von t ist, so kann man *einseitige* obere (t_+) und untere (t_-) sowie *beidseitige* Konfidenzgrenzen für das Konfidenzniveau (= Wahrscheinlichkeit) P anhand der folgenden Gleichungen definieren:

$$\begin{aligned} &\text{Einseitig} && P(t \le t_+) = \int_{-\infty}^{t_+} f(t)\,\mathrm{d}t \quad \text{bzw.} \\ & && P(t \ge t_-) = \int_{t_-}^{\infty} f(t)\,\mathrm{d}t \\ &\text{Beidseitig} && P(t_- \le t \le t_+) = \int_{t_-}^{t_+} f(t)\,\mathrm{d}t\ . \end{aligned}$$

Die folgenden Werte für das Konfidenzniveau (Wahrscheinlichkeit) sind üblich:

$$\begin{aligned} P &= 68\% \quad (1\,\sigma) \\ &= 95\% \quad (1.96\,\sigma) \quad \text{oder} \quad 95.4\% \quad (2\,\sigma) \\ &= 99\% \quad (2.58\,\sigma) \quad \text{(wenn man sehr sicher sein will).} \end{aligned}$$

Die Werte in Klammern gelten für eine Gaußsche Wahrscheinlichkeitsdichte und für ein beidseitiges symmetrisches Konfidenzniveau. Man sollte stets angeben, ob man sich auf einseitige oder beidseitige Grenzen bezieht. Häufig werden beidseitige Konfidenzgrenzen symmetrisch um den Erwartungswert t_0 gewählt. Diese Grenzen heißen *symmetrische* Konfidenzgrenzen, definiert durch $t_+ - t_0 = t_0 - t_-$.
Abbildung 9.1(a) zeigt einseitige und Abbildung 9.1(b) symmetrische Konfidenzgrenzen.

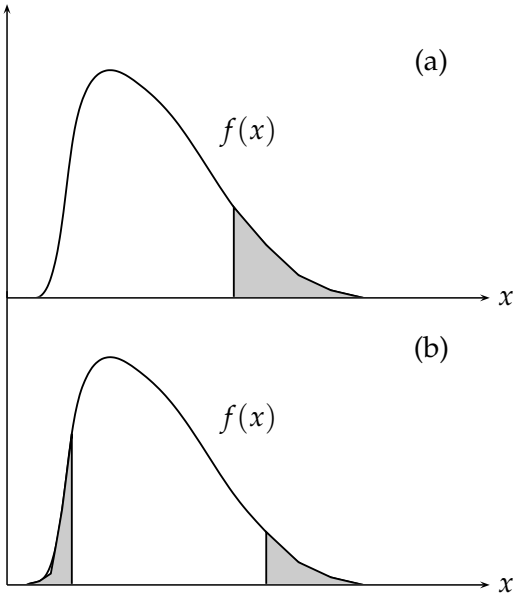


Abbildung 9.1: Eine einseitige Konfidenzgrenze (a) und zweiseitige, symmetrische Konfidenzgrenzen (b).

9.1.2 χ^2-Test

Dies ist ein sehr häufig verwendeter Test, um die Vereinbarkeit von Daten mit einer bestimmten Hypothese zu untersuchen.
Man betrachtet eine Menge von n Meßwerten $y_1, y_2, \ldots y_n$. Die Werte y_i mögen von einer (genau bekannten) Variablen x_i abhängen: $y_i = y_i(x_i)$. Ist diese Menge von n Werten y_i vereinbar mit einer gegebenen 'theoretischen' Wahrscheinlichkeitsdichte $y_t(x)$? Um dies zu überprüfen, sagt man für jede Stelle x_i aufgrund der vorausgesetzten Wahrscheinlichkeitsdichte den Mittelwert

$\langle y_t(x_i)\rangle$ voraus und prüft, ob die Daten y_i an derselben Stelle x_i innerhalb der Fehler verträglich sind.
Die Testgröße ist (siehe Kapitel 4.5.5)

$$t = \chi^2 = \sum_{i=1}^{n} \frac{(y_i - \langle y_t(x_i)\rangle)^2}{\sigma_{t,i}^2}\,, \tag{9.1}$$

wobei $\sigma_{t,i}^2$ die Varianz der theoretischen Verteilung an der Stelle x_i ist, d.h. $\sigma_{t,i}^2 = V\left[y_t(x_i)\right]$. Wenn die Werte y_i normalverteilt sind und eine Stichprobe aus der theoretischen Verteilung darstellen, dann gehorcht $t = \chi^2$ einer χ^2-Verteilung mit $k = n$ Freiheitsgraden (degrees of freedom, dof).
Abbildungen, welche die χ^2-Verteilung zeigen, befinden sich in Abschnitt 4.5.5.
In einigen Spezialfällen stellen die Werte y_i Anzahlen von Ereignissen n_i dar, also $n_i = y_i$. Wenn $n_{t,i}$ die vorhergesagte mittlere Anzahl von Ereignissen an der Stelle x_i ist, ergibt sich die Varianz $\sigma_{t,i}^2 = n_{t,i}$ aufgrund der zugrundeliegenden Poisson-Verteilung, und die χ^2-Testgröße wird

$$t = \chi^2 = \sum_{i=1}^{n} \frac{(n_i - n_{t,i})^2}{n_{t,i}}\,. \tag{9.2}$$

Die Testgröße folgt einer χ^2-Verteilung mit $k = n$ Freiheitsgraden, wenn die Zahlen $n_{t,i}$ hinreichend groß sind (> 10), so daß die gaußsche Näherung hinreichend gut ist.
Der Wert von χ^2 sollte nicht alleiniger Maßstab für den Test sein, sondern es sollten stets auch die Daten mit der theoretischen Funktion direkt verglichen werden.

Eine häufige Anwendung von Gleichung (9.2) betrifft *Histogramme*. Die x-Achse wird dabei in Intervalle eingeteilt und in dem Histogramm die Zahl n_i der Fälle eingetragen, die jeweils in das i-te Intervall (*bin*) fallen. Die zu erwartende Anzahl $n_{t,i}$ in jedem Intervall erhält man durch Integration der Wahrscheinlichkeitsdichte von $y_{t,i}$ über das i-te Intervall. Man kann zeigen [84], daß der Test empfindlicher wird, wenn die Anzahl n der Intervalle (d.h. die Anzahl der Freiheitsgrade) verringert wird (siehe Beispiel 9.3). Dieses Rezept sollte mit Bedacht benutzt werden. Die Unterteilung in Intervalle sollte fein genug sein, um die Strukturen der theoretischen Verteilung zu bewahren. Anhand der n-Abhängigkeit der Empfindlichkeit kann man verstehen, daß häufig ein globalerer Test wie der Kolmogorov-Smirnov-Test (siehe Kapitel 9.1.5) besser ist. Noch besser ist die Test-Statistik von Anderson-Darling [84], allerdings ist diese auch schwieriger anzuwenden.
Meist sind experimentelle und theoretische Verteilung auf dieselbe Gesamtzahl normiert, und in diesem Fall ist die Anzahl der Freiheitsgrade $k = n - 1$. Oft werden zusätzlich zu der Gesamtzahl weitere Parameter der Verteilung durch Minimierung von χ^2 geschätzt. In diesem Fall sinkt die Anzahl der Freiheitsgrade um 1 für jeden geschätzten Parameter.
Der χ^2-Test kann auch auf Verteilungen mit mehr als einer Variablen angewandt werden. In diesem Fall wählt man eine Intervall- (= Gebiets-) Einteilung im mehrdimensionalen Raum; für zwei Variable x, z sind dies zum Beispiel rechteckige Gebiete in x, z. Die Gebiete sollten so groß gewählt werden, daß die Zahl der Einträge in jedem Gebiet ≥ 10 ist. Die erwarteten und gemessenen Anzahlen in jedem Gebiet werden dann durch Berechnung einer χ^2-Summe über alle Gebiete verglichen.

Beispiel 9.2 Prüfung eines Zufallszahlengenerators.

Ein Zufallszahlengenerator habe 1000 Zufallszahlen produziert, die angeblich zwischen 0 und 1 gleichverteilt sind. Sie werden in zehn gleich große Intervalle der Breite 0.1 sortiert, wie in Abbildung 9.2 gezeigt.

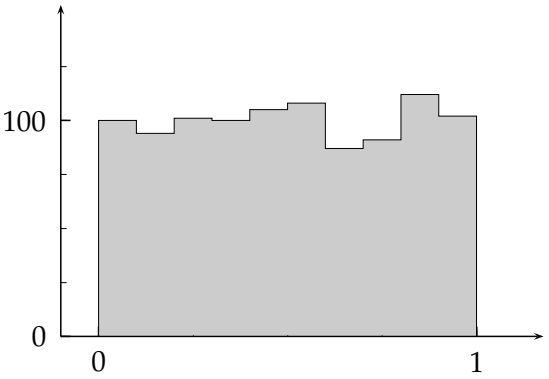


Abbildung 9.2: Prüfung von 1000 Werten eines Zufallszahlengenerators. In den jeweiligen Intervallen ist die Zahl der Zufallszahlen eingetragen, die in dieses Intervall fallen.

Befinden sich n_i Zufallszahlen in dem i-ten Intervall, so ist die Testgröße

$$t = \chi^2 = \sum_{i=1}^{10} \frac{(n_i - n_0)^2}{n_0},$$

wobei $n_0 = 1000/10 = 100$ die mittlere Anzahl von Einträgen in jedem Intervall ist, die für eine Gleichverteilung erwartet wird. Die Testgröße t folgt einer χ^2-Verteilung mit $k = 10 - 1 = 9$ Freiheitsgraden, da ein Parameter, die Gesamtzahl der generierten Zahlen, angepaßt wurde.

Wenn die Zahlen wirklich einer Zufallsverteilung folgen, so ist der Mittelwert von $\chi^2 = 9$. Man nehme an, daß sich für einen bestimmten Durchlauf von 1000 Zahlen $\chi^2 = 20$ ergibt. Welche Schlüsse können dann gezogen werden? Die Abbildung 4.14 zeigt, daß sich die Wahrscheinlichkeit, einen Wert $\chi^2 = 20$ oder einen noch größeren Wert (für $k = 9$) zu erhalten, zu 2% ergibt. Wenn also der Durchlauf von 1000 Zufallszahlen wiederholt wird, ist die Wahrscheinlichkeit, einen χ^2-Wert größer oder gleich dem eben berechneten zu erhalten, nur 2%, wenn die Zufallszahlen tatsächlich gleichverteilt sind. Eine Wahrscheinlichkeit von 2%, ein Ergebnis mit einem χ^2 dieser Größe oder noch größer zu erhalten, legt Zweifel an der Hypothese nahe, daß die Zufallszahlen gleichverteilt sind. Hier würde man wahrscheinlich den Test wiederholen wollen.
Dies ist ein Beispiel, bei dem man eine einseitige Konfidenzgrenze anwendet. Für ein Konfidenzniveau von 95% verlangt man $1 - P = 0.05$ (siehe Abbildung 4.14) und $t = \chi^2 \leq 16.9$. Dieses Beispiel würde daher den Test nicht bestehen. Sollte man einen besonders kleinen Wert von χ^2 erhalten haben, zum Beispiel $\chi^2 = 1$, so wäre dies auch verdächtig. Die Wahrscheinlichkeit, einen so kleinen Wert von χ^2 oder einen noch kleineren zu erhalten, wäre nur 0.1% – das würde eine Korrelation zwischen den Zahlen oder eine fehlerhafte Wahl der Varianz nahelegen.

Beispiel 9.3 Ehrlichkeit einer Verlosung.

In einer Reihe von 10 Verlosungen werden je 1000 Lose gezogen; nach jeder Verlosung registriert man die Anzahl der Gewinne n_i und erhält die folgenden Zahlen: 24, 15, 17, 18, 26, 24, 32, 33, 29, 32.
Die mittlere Anzahl von Gewinnen pro Verlosung ist also $250/10 = 25$. Ist die Verlosung ehrlich, d.h. gibt es bei jeder Verlosung (im Mittel) die gleiche Anzahl an Gewinnlosen? Ein χ^2-Test ergibt

$$t = \chi^2 = \sum_{i=1}^{10} \frac{(n_i - 25)^2}{25} = 14.4;$$

für neun Freiheitsgrade ist die Wahrscheinlichkeit, diesen oder einen noch größeren Wert für χ^2 zu erhalten, 12% (siehe Abbildung 4.14); das ist noch annehmbar.

Es scheint also, daß die Verlosung gerecht ist, dies stimmt aber nicht. Ungefähr in der Mitte der Serie geschieht etwas Böses. Dies läßt sich feststellen, wenn man die ersten fünf und die letzten fünf Zahlen getrennt kombiniert. Das χ^2 ist nun

$$\chi^2 = \frac{(100-125)^2}{125} + \frac{(150-125)^2}{125} = 10 \qquad \text{für} \quad k = 1 \quad \text{Freiheitsgrad}\,.$$

Die χ^2-Wahrscheinlichkeit ist nun 0.2%, was eindeutig unakzeptabel ist. Eine einfache Untersuchung der Zahlen hätte auch aufgezeigt, daß etwas nicht mit rechten Dingen zugeht. Der χ^2-Test sollte also nie blindlings angewandt werden, statt dessen sollten die tatsächlichen Abweichungen aller Zahlen von den erwarteten Werten auf systematische oder vereinzelte, extreme Abweichungen untersucht werden. Das Beispiel unterstreicht auch die allgemeine Regel, daß der χ^2-Test empfindlicher wird, wenn die Zahl der Summanden verringert wird.

Tatsächlich wäre hier der Kolmogorov-Smirnov-Test angebracht gewesen. Aufgrund seines globalen Charakters reagiert er empfindlicher auf Trends in den Daten.

9.1.3 Studentscher t-Test

Dieser Test wird verwendet, um die Vereinbarkeit eines gemessenen (geschätzten) Mittelwertes $\bar{x}$ mit einem gegebenen (theoretischen) Mittelwert μ zu untersuchen, oder aber die Vereinbarkeit zweier gemessener (aus den Daten geschätzter) Mittelwerte für den Fall, daß die Varianzen nicht a priori bekannt sind, sondern aus den Daten mittels Gleichung (4.28) geschätzt werden müssen.
Hat man n Meßwerte x_i, die einer Wahrscheinlichkeitsdichte mit Mittelwert μ und Varianz σ^2 folgen, dann gilt für den Stichproben-Mittelwert $\bar{x} = 1/n \cdot \sum x_i$. Er folgt für große n einer Gauß-Verteilung mit Varianz σ^2/n (Standardabweichung $\sigma/\sqrt{n}$), entsprechend dem Zentralen Grenzwertsatz. Wenn σ^2 nicht bekannt ist, muß eine beste Schätzung s^2 dafür aus den Daten selbst bestimmt werden (siehe Gleichung (4.28)). Um zu prüfen, ob der Mittelwert der Meßwerte $\bar{x}$ mit einem gegebenen theoretischen Mittelwert μ vereinbar ist, wird die Testgröße

$$t = \frac{\bar{x} - \mu}{\sqrt{s^2/n}}$$

gebildet. Sie folgt einer t-Verteilung (siehe Kapitel 4.5.7) mit $n-1$ Freiheitsgraden.

Beispiel 9.4 Vergleich eines gemessenen Mittelwertes mit einem theoretischen Mittelwert.

Man hat drei Meßwerte $x_1 = -1$, $x_2 = 0$, $x_3 = 1$ mit einem Mittelwert $\bar{x} = 0$. Was ist die Wahrscheinlichkeit P', einen solchen Mittelwert oder einen noch größeren durch Zufall zu erhalten, wenn der wahre Mittelwert $\mu = -1$ ist? Antwort: Die geschätzte Varianz ist $s^2 = 1$, gemäß Gleichung (4.28). Die Testgröße ist $t = (\bar{x} - \mu)/(s/\sqrt{n}) = \sqrt{3}$, und die Wahrscheinlichkeit P' ergibt sich unter Verwendung der t-Verteilung (siehe Gleichung (4.26), Tabelle 4.2 und Abbildung 4.19) mit $3 - 1 = 2$ Freiheitsgraden zu

$$P' = \int_{t=\sqrt{3}}^{\infty} f_2(t)\,\mathrm{d}t = 0.11\,.$$

Man beachte, daß Abbildung 4.19 für zweiseitige Konfidenzgrenzen gilt. Das Konfidenzniveau ist $P = 1 - P' = 89\%$.

Vergleich zweier durch das Experiment bestimmter Mittelwerte. Man hat zwei Meßwertreihen mit den Werten x_{1i}, $i = 1, 2, \ldots n_1$ in der einen Meßreihe und den Werten x_{2i}, $i = 1, 2, \ldots n_2$ in der anderen. Sind die Mittelwerte beider Reihen miteinander vereinbar? Die Testgröße ergibt sich zu

$$t = \frac{D}{s} \qquad \text{mit} \qquad D = \overline{x}_1 - \overline{x}_2, \qquad s^2 = s_1^2 + s_2^2\,,$$

wobei $\overline{x}_1$, $\overline{x}_2$ die Mittelwerte der beiden Meßreihen sind und s_1^2, s_2^2 die besten Schätzungen der Varianzen $V[\overline{x}_1]$, $V[\overline{x}_2]$ von $\overline{x}_1$, $\overline{x}_2$, bestimmt aus den Meßreihen gemäß Gleichung (4.28) und Gleichung (4.29). Die Testgröße t folgt einer Studentschen t-Verteilung mit $k = n_1 + n_2 - 2$ Freiheitsgraden.

9.1.4 *F*-Test

Dies ist ein Test auf die Gleichheit zweier Varianzen, die aus zwei Stichproben gemäß Gleichung (4.28) geschätzt wurden. Sind s_1^2 und s_2^2 diese beiden Schätzungen für die Varianzen, so folgt die Variable $F = s_1^2/s_2^2$ einer F-Verteilung, falls die beiden Stichproben aus derselben Wahrscheinlichkeitsdichte stammen. Die F-Verteilung ist in Kapitel 4.5.8 erklärt und tabelliert, wobei nach der gängigen Konvention $s_1^2 \geq s_2^2$ gewählt wird, so daß $F \geq 1$ ist.

9.1.5 Kolmogorov-Smirnov-Test

Dieser Test reagiert empfindlich auf Unterschiede in der globalen Form oder in Tendenzen von Verteilungen. Es sei eine theoretische Wahrscheinlichkeitsdichte $f(x)$ und ihre Verteilungsfunktion $F(x) = \int_{-\infty}^{x} f(x')\,\mathrm{d}x'$ vorgegeben. Um zu prüfen, ob eine Meßreihe mit n Werten x_i, $i = 1, 2, \ldots n$ vereinbar mit der Wahrscheinlichkeitsdichte $f(x)$ ist, ordnet man die x_i-Werte nach ihrer Größe und konstruiert die kumulative Größe

$$F_n(x) = \frac{\text{Anzahl der } x_i\text{-Werte} \leq x}{n} \,.$$

Die Testgröße ist

$$t = \sqrt{n} \cdot \max|F_n(x) - F(x)| \,,$$

d.h. man sucht nach der größten Differenz zwischen den beiden kumulativen Verteilungen. Die Wahrscheinlichkeit P, einen Wert $\leq t_0$ für die Testgröße t zu erhalten, ist

$$P = 1 - 2\sum_{k=1}^{\infty}(-1)^{k-1} \cdot \exp(-2k^2\, t_0^2) \,.$$

Werte für den praktischen Gebrauch sind in Tabelle 9.1 gezeigt.

P	1%	5%	50%	68%	95%	99%	99.9%
t_0	0.44	0.50	0.83	0.96	1.36	1.62	1.95

Tabelle 9.1: Wahrscheinlichkeit P, einen Wert kleiner oder gleich t_0 für die Testgröße t zu erhalten.

9.2 Entscheidung zwischen Hypothesen

9.2.1 Allgemeine Begriffe

Häufig steht man vor dem Problem, zwischen zwei Hypothesen aufgrund experimenteller Daten unterscheiden zu müssen. Einfache Beispiele sind Tests eines neuen Medikaments, wo die zwei folgenden Hypothesen zu überprüfen sind: es wirkt/es wirkt nicht; oder Untersuchungen auf eine Krankheit, wo man die zwei Hypothesen krank/gesund überprüfen muß. Gelegentlich wird ein Problem nicht auf diese Art betrachtet, sondern nur eine (offensichtliche) Hypothese überprüft. Dies kann zu falschen Ergebnissen führen, wie Beispiele zeigen werden. Das allgemeine Schema ist das folgende.

1. Gegeben sind zwei Hypothesen, die Nullhypothese H_0 und die Alternativhypothese H_1, zwischen denen man sich entscheiden muß.
2. Eine Testgröße t wird definiert und anhand der Daten berechnet unter der Annahme, daß H_0 wahr ist. Die Testgröße t ist eine Zufallsvariable.
3. Eine Konfidenzgrenze α für H_0 wird wie folgt bestimmt: Aus der Wahrscheinlichkeitsdichte $f_0(t)$ der Testgröße t wird eine Annahme- und eine Verwurfsregion (auch kritische Region genannt) bestimmt. Die Annahmeregion enthält den großen Bruchteil $(1-\alpha)$ der Fläche unter $f_0(t)$ und die Verwurfsregion den kleinen Bruchteil α. Häufig wird $\alpha = 0.05$ gewählt, dies entspricht etwa zwei Standardabweichungen einer Gauß-Verteilung.

Es gibt einen kritischen Wert t_c für t, der die Annahme- von der Verwurfsregion trennt (siehe Abbildung 9.3).

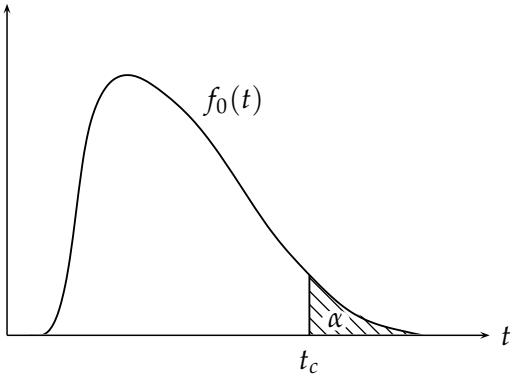


Abbildung 9.3: Die Wahrscheinlichkeitsdichte $f_0(t)$ für die Hypothese H_0 mit der durch t_c gegebenen (einseitigen) kritischen Region. Die kritische Region (Verwurfsregion) bedeckt die Fläche α.

Dementsprechend definiert man Annahme- und Verwurfsregionen für die Hypothese H_1 mit einem Bruchteil β und einer Konfidenz $1-\beta$ durch Berechnung der Wahrscheinlichkeitsdichte $f_1(t)$ mit derselben Testgröße t. Dabei folgt $f_1(t)$ aus der Annahme, daß H_1 wahr ist. Natürlich ist t_c gleich für beide Hypothesen. Abbildung 9.4 verdeutlicht dies für die beiden Hypothesen H_0 und H_1.

Wie man Abbildung 9.4 entnehmen kann, bestimmt die Wahl von t_c die Bruchteile α und β. Man kann α klein wählen auf Kosten von β und umgekehrt. Diese Wahl, die meist auf einen Kompromiß hinausläuft, wird von dem jeweils behandelten Problem abhängen und wird an Beispielen verdeutlicht.

Formal gilt

$$\alpha(t_c) = \int_{V(H_0)} f_0(t)\,\mathrm{d}t\,, \qquad \beta(t_c) = \int_{V(H_1)} f_1(t)\,\mathrm{d}t\,,$$

wobei die Integrale sich über die Verwurfsregionen $V(H_0)$, $V(H_1)$ von H_0 bzw. H_1 erstrecken. Natürlich ist die Annahmeregion von H_0 gleich der Verwurfsregion von H_1 und umgekehrt.

Es sind zwei Arten von Fehlern zu betrachten:
Fehler erster Art: H_0 ist wahr, aber t liegt innerhalb der Verwurfsregion von H_0, wodurch fälschlicherweise H_0 verworfen und H_1 angenommen wird. Die Wahrscheinlichkeit hierfür ist α.
Fehler zweiter Art: H_1 ist wahr, aber t liegt innerhalb der Annahmeregion von H_0, wodurch H_0 zu Unrecht angenommen und H_1 verworfen wird. Die Wahrscheinlichkeit hierfür ist β.
Eine sorgfältige Betrachtung beider Fehlerarten ist wichtig für die Wahl von t_c.
Ein guter Text zu diesem Thema ist [81].

Beispiel 9.5 Aidstest.

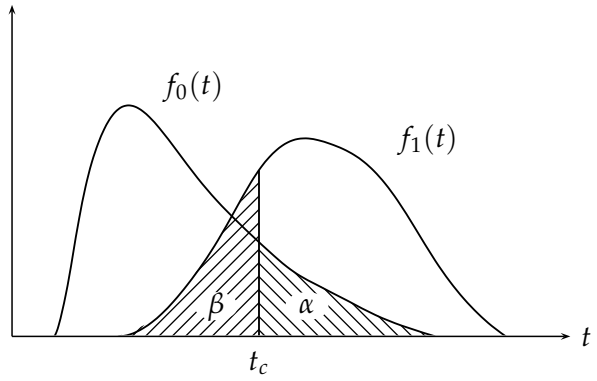


Abbildung 9.4: Die Wahrscheinlichkeitsdichten $f_0(t)$ und $f_1(t)$ für die Hypothesen H_0 und H_1. Der kritische Wert t_c bestimmt die Wahrscheinlichkeiten α und β, die Hypothesen H_0 bzw. H_1 zu verwerfen.

Ein bestimmter Test für Aids hat eine Effektivität von 99%, d.h. in 99% der Fälle wird die korrekte Diagnose gestellt. Andererseits wird für einen Bruchteil von 0.01% der gesunden Personen fälschlicherweise Aids diagnostiziert. Ist die H_0-Hypothese Aids, so ist der Fehler erster Art 1%, der Fehler zweiter Art 0.01%. Dies klingt nach einem guten Test, man stelle sich jedoch vor, daß Aids einen Bruchteil von 10^{-4} der durchschnittlichen Bevölkerung befällt. Werden 100 000 Personen der durchschnittlichen Bevölkerung anhand dieses Tests untersucht, so wird dieser im Mittel zu Unrecht in zehn Fällen Aids diagnostizieren, und in zehn Fällen zu Recht. Daraus folgt, daß in 50% der Fälle die Diagnose Aids falsch ist! Wird also Aids diagnostiziert, sollte unter den Voraussetzungen dieses Beispiels ein zweiter, unabhängiger Test vorgenommen werden.

9.2.2 Strategien zur Wahl der Verwurfsregion

Die Wahl der Verwurfsregion, d.h. die Wahl des kritischen Parameters t_c, wird jeweils vom behandelten Problem abhängen. Es mag zum Beispiel wichtig sein, die korrekte Hypothese mit einer großen Wahrscheinlichkeit anzunehmen und so die Fehler erster Art zu minimieren auf Kosten einer vergrößerten Wahrscheinlichkeit, einen Fehler zweiter Art zu begehen, also die falsche Hypothese anzunehmen. Eine Methode, Krebs zu diagnostizieren, mag diese Eigenschaften verlangen. Wenn die Annahme der falschen Hypothese jedoch katastrophale Konsequenzen nach sich zieht, so gilt es, den Fehler zweiter Art zu minimieren, auch wenn dann manchmal die richtige Hypothese verworfen wird.
Diese Betrachtungen sollen nun formalisiert werden. Man nehme an, daß eine Beobachtung (bzw. eine Reihe von Beobachtungen) gemacht und eine auf die Messungen bezogene Testgröße t bestimmt wurde. Dann bezeichne $f_0(t)$ die Wahrscheinlichkeitsdichte von t unter der Annahme, daß die Hypothese H_0 wahr ist, und $f_1(t)$ die Wahrscheinlichkeitsdichte von t, wenn die Hypothese H_1 wahr ist.

Man mag versucht sein, H_0 anzunehmen, wenn $f_0(t) > f_1(t)$ ist, dies kann jedoch falsch sein, wenn die Hypothese H_0 a priori viel unwahrscheinlicher ist als H_1.

Zum Beispiel mag H_0 die Hypothese sein, daß die Beobachtungen ein Anzeichen für eine wichtige neue Entdeckung sind, aber leider sind solche Entdeckungen recht unwahrscheinlich, und man muß seiner Sache dann schon sehr sicher sein.

Um der Entscheidungsprozedur eine quantitative Grundlage zu verschaffen, muß man den Hypothesen H_0 und H_1 Wahrscheinlichkeiten zuweisen. Im Gegensatz zu der Wahrscheinlichkeitsdefinition in Kapitel 4.2 fußen diese Wahrscheinlichkeiten auf Annahmen, auf der Kenntnis der Problematik und auf dem gesunden Menschenverstand – in der Bayes'schen Statistik werden sie *Prior* genannt (siehe Kapitel 6.8).

Sind die *Wahrscheinlichkeiten*, daß die Hypothese H_0 bzw. H_1 wahr ist, $P(H_0)$ bzw. $P(H_1)$, dann wird H_0 angenommen, wenn

$$P(H_0) \cdot f_0(t) > P(H_1) \cdot f_1(t)\,. \tag{9.3}$$

Beispiel 9.6 Photomultiplier mit Rauschen.

Man wählt als Testgröße die Spannungsamplitude eines Photomultipliers am Ende seiner analogen Auslesekette. Nun möchte man Rauschen (Hypothese H_0) von einem wahren Signal (Hypothese H_1) unterscheiden. Man nehme an, daß das Rauschen normalverteilt ist mit Mittelwert $\mu = 0$ und Standardabweichung σ, und daß das wahre Signal eine Amplitude μ_s bei gleicher Standardabweichung σ hat. Dann ist

$$f_0(t) = \frac{1}{\sqrt{2\,\pi}\sigma} \cdot e^{-t^2/2\sigma^2}, \quad f_1(t) = \frac{1}{\sqrt{2\,\pi}\sigma} \cdot e^{-(t-\mu_s)^2/2\sigma^2}.$$

Die kritische Amplitude t_c zur Unterscheidung von Rauschen und Signal ist dann nach Gleichung (9.3) gegeben durch

$$P(H_0) \cdot e^{-t_c^2/2\sigma^2} = P(H_1) \cdot e^{-(t_c-\mu_s)^2/2\sigma^2}$$

oder, mit $P(H_1) = 1 - P(H_0)$

$$t_c = \frac{\mu_s}{2} + \frac{\sigma^2}{\mu_s} \ln\left(\frac{P(H_0)}{1-P(H_0)}\right). \tag{9.4}$$

Dies ist schön und gut, aber wie läßt sich $P(H_0)$ bestimmen? Eine Methode hierfür wird im Beispiel 9.7 vorgestellt.

Gewichte. Bisher ist der Ansatz noch zu naiv, da die Konsequenzen einer Fehlentscheidung noch nicht einbezogen werden. Man muß diese Konsequenzen gewichten, und es ist üblich, dies dadurch zu machen, daß man einer Fehlentscheidung Kosten zuweist. Es wird eingeführt C_{10} = Kosten für die Annahme der Hypothese H_1, wenn in Wirklichkeit H_0 wahr ist (= Kosten für einen Fehler erster Art), und C_{01} = Kosten für die Wahl von H_0, wenn in Wirklichkeit H_1 wahr ist (= Kosten für einen Fehler der zweiten Art).
Die Kosten, eine jeweils korrekte Hypothese H_0 oder H_1 anzunehmen, werden $= 0$ gesetzt. Eine Entscheidungsprozedur kann nun formuliert werden anhand der Konfidenz C_0, mit der man auf die Hypothese H_0 wetten würde. Diese Konfidenz ist natürlich sehr groß, wenn H_0 a priori sehr wahrscheinlich ist und die Daten mit H_0 vereinbar sind. Wenn jedoch die Kosten C_{01} dafür, H_0 statt der korrekten Hypotehese H_1 zu wählen, sehr groß sind, so muß der Enthusiasmus für H_0 gebremst werden, entsprechend den Kosten, einen Fehler zu machen. Sind also die Kosten für eine falsche Annahme von H_0 hoch, so muß man noch sicherer sein, daß die Annahme von H_0 auch korrekt ist. Also

$$C_0 \approx P(H_0)\, f_0(t)/C_{01}\,.$$

Das gleiche kann für die Hypothese H_1 formuliert werden

$$C_1 \approx P(H_1)\, f_1(t)/C_{10}\,.$$

H_0 wird nun angenommen, wenn $C_0 > C_1$ oder

$$\frac{f_0(t)}{f_1(t)} > \frac{P(H_1) \cdot C_{01}}{P(H_0) \cdot C_{10}} = \frac{(1-P(H_0))\, C_{01}}{P(H_0) \cdot C_{10}}\,. \tag{9.5}$$

Der kritische Wert $t = t_c$ folgt aus dieser Gleichung, wenn man das Gleichheitszeichen setzt.

9.2.3 Minimax-Kriterium

Es ist häufig schwierig, die a priori-Wahrscheinlichkeit $P(H_0)$ der Hypothese H_0 abzuschätzen. Ein Ansatz besteht darin, einen solchen Wert von $P(H_0)$ zu wählen, daß die Kosten für eine Fehlentscheidung bei Anwendung einer optimalen Entscheidungsprozedur, zum Beispiel nach Gleichung (9.5), maximal werden. Die wahren Kosten sind dann immer kleiner.

Beispiel 9.7 Minimax-Kriterium zur Unterscheidung zwischen Rauschen und Signal.

In dem vorangehenden Beispiel wurde eine Entscheidungsprozedur hergeleitet und die kritische Amplitude nach Gleichung (9.4) bestimmt. Sie muß nun durch Einführung von Kosten gemäß Gleichung (9.5) modifiziert werden wie folgt:

$$t_c = \frac{\mu_s}{2} + \frac{\sigma^2}{\mu_s} \cdot \ln\left(\frac{P(H_0)\ \cdot\ C_{10}}{(1-P(H_0))\ \cdot\ C_{01}}\right), \tag{9.6}$$

$$\begin{array}{lcl} \text{wähle } \textit{Signal} & = & H_1, \text{ wenn } t > t_c, \\ \text{wähle } \textit{Rauschen} & = & H_0, \text{ wenn } t < t_c. \end{array}$$

Die Gesamtkosten sind

$$C_g = P(H_0)\ \cdot\ C_{10} \int_{t_c}^{\infty} f_0(t)\,\mathrm{d}t + (1-P(H_0))\ \cdot\ C_{01} \int_{-\infty}^{t_c} f_1(t)\mathrm{d}t$$

mit

$$f_0(t) = \frac{1}{\sqrt{2\,\pi}\sigma}\ \cdot\ e^{-t^2/2\sigma^2}, \quad f_1(t) = \frac{1}{\sqrt{2\,\pi}\sigma}\ \cdot\ e^{-(t-\mu_s)^2/2\sigma^2},$$

wobei die Integrale sich jeweils über die Verwurfsregionen von H_0 und H_1 erstrecken, mit t_c aus Gleichung (9.6). Es ergibt sich

$$C_g = P(H_0)\ \cdot\ C_{10}\,\frac{1}{2}\left(1-\mathrm{erf}\left(\frac{t_c}{\sigma\sqrt{2}}\right)\right) + (1-P(H_0))\ \cdot\ C_{01}\,\frac{1}{2}\left(1+\mathrm{erf}\left(\frac{t_c-\mu_s}{\sqrt{2}\sigma}\right)\right).$$

Setzt man t_c aus Gleichung (9.6) ein, so ergibt sich für C_g eine komplizierte Funktion von $P(H_0)$, die jedoch leicht numerisch berechnet werden kann.

Abbildung 9.5 zeigt dies für zwei Werte für die Kosten, nämlich $C_{10} = 1,\ C_{01} = 2$ und $C_{10} = 2,\ C_{01} = 1$, und mit $\mu_s = 1,\ \sigma = 1$.

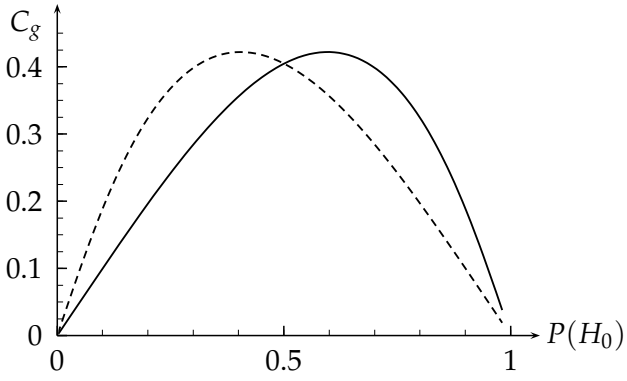


Abbildung 9.5: Die Gesamtkosten C_g als eine Funktion von $P(H_0)$, der Wahrscheinlichkeit von H_0, für zwei Werte der Gewichte, $C_{10} = 1,\ C_{01} = 2$ (durchgezogene Linie) und $C_{10} = 2,\ C_{01} = 1$ (gestrichelte Linie).

Die Kosten sind maximal für $P(H_0) = 0.596$ bzw. $P(H_0) = 0.404$. Dies sind dann die in der Entscheidungsprozedur Gleichung (9.6) zu verwendenden Werte. Sie ergeben obere Grenzen für die Gesamtkosten.

Beispiel 9.8 Negativbeispiel.

Dies ist als abschreckendes Beispiel gedacht.

Man möchte die Effektivität von Vitamin C bei der Behandlung des gewöhnlichen Schnupfens überprüfen. Für das Experiment werden zwei unter Schnupfen leidende Personen untersucht. Eine erhält zur Behandlung Vitamin C, die andere wird mit einem Placebo behandelt. Nach einem Tag wird der Zustand beider Personen verglichen; wenn die mit Vitamin C behandelte Person sich in besserer Verfassung als die andere befindet, wird dies als *Erfolg* gewertet.

Die Nullhypothese lautet: Vitamin C hilft nicht.
Wenn man auf diese Art n Personenpaare untersucht, erwartet man im Mittel $n/2$ 'Erfolge' unter Annahme der Nullhypothese. Um in der Lage zu sein, einen Effekt mit einer Konfidenz von 97.7% (entsprechend 2 Standardabweichungen, einseitig) zu bestimmen, benötigt man (für $n \geq 10$) mehr als etwa

$$n/2 + 2\sigma = n/2 + 2\sqrt{n/4}$$

Erfolge für ein Experiment mit n Paaren (da $\sigma = \sqrt{n\,p\,(1-p)}$ mit $p = 0.5$). Für $n = 20$ ergibt sich

$$n/2 + 2\sqrt{n/4} = 14.5\,.$$

Legt man demnach eine Grenze von 15 fest, so kann anhand der in diesem Fall anzuwendenden Binomialverteilung ein genauerer Wert für P bestimmt werden: P sei die Wahrscheinlichkeit, zu Unrecht einen positiven Einfluß von Vitamin C zu behaupten, wenn in Wirklichkeit nur eine statistische Fluktuation vorliegt. Man erhält

$$P = \sum_{r=15}^{20} P(20, r) = 2.1\%\,,$$

wobei $P(n, r)$ die Wahrscheinlichkeit (4.13) der Binomialverteilung für $n = 20$ und $p = 0.5$ ist. Bis hierher ist alles korrekt.

Nun kommt der abschreckende Teil: Man wählt unter Erfolgsdruck ein anderes Vorgehen. Wenn nämlich nach einer anfänglichen Anzahl von $n_{\min}$ Paaren kein Effekt entsprechend 2 Standardabweichungen gesehen wird, wird die Versuchsreihe bis zu einer maximalen Anzahl $n_{\max}$ fortgeführt. Wenn während der Fortführung ein solcher Effekt festgestellt wird, wird der Versuch abgebrochen und *Erfolg* gemeldet. In diesem Fall ist nun die Wahrscheinlichkeit, eine Wirksamkeit zu behaupten, während in Wirklichkeit die Nullhypothese korrekt ist, viel größer als 2.1%. Die Tabelle 9.2, die anhand von Monte Carlo-Berechnungen bestimmt wurde, zeigt dieses.

$n_{\min}$	20	20	20	20	20
$n_{\max}$	20	22	25	50	100
$P(\%)$	2.1	2.9	4.0	7.4	11.3

Tabelle 9.2: P = Wahrscheinlichkeit, einen falschen Effekt aufgrund einer statistischen Fluktuation zu behaupten, welche die 97.7% Konfidenzgrenze (entspricht etwa 2 σ) übertrifft. Wenn nach $n_{\min}$ Versuchen keine solche Abweichung gesehen wird, wird das Experiment fortgeführt, bis ein Effekt auftritt, und dann gestoppt; man geht dabei bis zu maximal $n_{\max}$ Versuchen.

9.3 Allgemeine Klassifizierungsmethoden

Oft möchte man Dinge, welche durch eine Anzahl von Eigenschaften charakterisiert sind, gewissen Klassen zuordnen. Ein Beispiel ist die Zuordnung von (digitalisierten) Bildern von Gesichtern zu den entsprechenden Namen der Personen. Man kann dies als ein Problem auffassen, bei dem anstelle einer einzigen Testgröße ein Vektor von Testgrößen auftritt, welcher aus den Eigenschaften der Dinge berechnet werden kann. Man muß dann das Entscheidungsproblem in einem mehrdimensionalen Raum lösen. Im folgenden werden zwei gängige Verfahren vorgestellt.

9.3.1 Fishersche Diskriminantenmethode

Dies ist eine allgemeine Methode, um auf einer statistischen Basis verschiedene Ereignisklassen aufgrund ihrer gemessenen Eigenschaften zu unterscheiden [52]. Man nehme zwei Klassen, zwischen denen es zu unterscheiden gilt, Klasse 1 (= *Kaninchen*), und Klasse 2 (= *Hasen*). Ein unbekanntes Ereignis (= *Tier*) soll aufgrund seiner gemessenen Eigenschaften einer der beiden Klassen zugeordnet werden, d.h. es ist die Entscheidung zu treffen, ob es ein Kaninchen oder ein Hase ist.

Die Ereignisse (*Tiere*) werden charakterisiert durch eine Menge von n Variablen (*Eigenschaften*), $X_1^{(1)}, X_2^{(1)}, \ldots X_n^{(1)}$ für Klasse 1 und $X_1^{(2)}, X_2^{(2)}, \ldots X_n^{(2)}$ für Klasse 2, welche zu Vektoren $\boldsymbol{X}^{(1)}$ und $\boldsymbol{X}^{(2)}$ zusammengefaßt werden; $\boldsymbol{X}$ ist der Vektor der Eigenschaften des zu klassifizierenden Gegenstandes. Zum Beispiel sei $X_1 =$ Gewicht des Tieres, $X_2 =$ (Länge der Ohren)$/\sqrt[3]{X_1}$.
Man nehme nun eine Anzahl von Hasen und Kaninchen und trage X_1 gegen X_2 für jede Tierklasse auf. Mit einem Eintrag pro Tier erhält man zwei Mengen von Punkten im zwei- (im allgemeinen n-) dimensionalen Raum. Man bildet nun für jede Klasse den Stichprobenmittelwert der Vektoren $\boldsymbol{X}$, also $\overline{\boldsymbol{X}}^{(1)}$ und $\overline{\boldsymbol{X}}^{(2)}$. Hieraus berechnet man eine Schätzung für die Kovarianzmatrix für die beiden Klassen, zum Beispiel für Klasse 1:

$$V_{mk}^{(1)} = \frac{1}{N} \sum_N (X_m^{(1)} - \overline{X}_m^{(1)})(X_k^{(1)} - \overline{X}_k^{(1)}) \,, \tag{9.7}$$

und entsprechend für die Elemente $V_{mk}^{(2)}$ der Klasse 2. Die Summe geht über die N Ereignisse der Stichprobe. Man bildet dann die mittlere Kovarianzmatrix

$$V_{mk} = \frac{1}{2} \left(V_{mk}^{(1)} + V_{mk}^{(2)} \right) .$$

Um zwischen den zwei Klassen zu diskriminieren, d.h. um herauszufinden, ob ein unbekanntes Ereignis (*Tier*), charakterisiert durch $X_1, X_2, \ldots X_n$, zu Klasse 1 oder Klasse 2 gehört, benötigt man eine Linie (im allgemeinen eine Ebene in $(n-1)$ Dimensionen), um die zwei Punktmengen zu trennen. Dies wird durch Fishers Algorithmus geleistet. Die Testgröße ist

$$t = \sum_{i=1}^{n} f_i \, X_i - \frac{1}{2} \sum_{i=1}^{n} f_i \, (\overline{X}_i^{(1)} + \overline{X}_i^{(2)})$$

mit

$$f_i = \sum_k (V^{-1})_{ik} \, (\overline{X}_k^{(1)} - \overline{X}_k^{(2)}) \,.$$

Da der zweite Term nicht von den das unbekannte Tier charakterisierenden Werten X_i abhängt, muß nur der erste Term berücksichtigt werden.

In einem praktischen Fall trägt man t für je eine Stichprobe der zwei Klassen auf – die Verteilungen sollten sich für eine gute Trennung deutlich unterscheiden. Die Wahl des kritischen Wertes t_c, der zwischen den beiden Klassen diskriminiert, wird von dem gegebenen Problem abhängen, zum Beispiel von der relativen Häufigkeit des Vorkommens der beiden Klassen und von den Konsequenzen einer falschen Zuweisung usw. Es kann gezeigt werden, daß dieser Algorithmus optimal ist, wenn die Variablen X_i normalverteilt sind.

9.3.2 Neuronale Netze

Mit neuronalen Netzen kann man eine allgemeine Klassifizierung durchführen. Ehe man diese Methode benutzt, sollte man einen Versuch mit der Fisherschen Diskriminantenmethode machen. Wenn man die Eingabevariablen so transformieren kann, daß sie angenähert gaußverteilt sind, führt die Fishersche Methode schnell und effektiv zum Ziel. Für unübersichtliche Probleme können neuronale Netze indes der richtige Ansatz sein. Eine umfassende Beschreibung liegt jedoch außerhalb des Rahmens dieses Buches (Literatur siehe zum Beispiel [52, 31, 64, 1]).
Ein häufig verwendeter Ansatz verwendet *feed-forward Netzwerke*. Bei dieser Netzwerkarchitektur wird ein Muster in der Form eines Eingabevektors $\mathbf{X}$ mit Komponenten X_k dem Netz präsentiert. Der einfachste Fall ist ein Netz ohne verborgene Knoten (Abbildung 9.6).

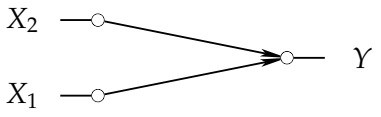


Abbildung 9.6: Ein einfaches neuronales Netzwerk ohne verborgene Knoten.

Die Eingabe besteht aus den beiden Komponenten X_1 und X_2, und die Ausgabe Y ist

$$Y = g(B_1X_1 + B_2X_2 - \alpha)\,.$$

Die Koeffizienten B_i und die Schwelle α bestimmen die Eigenschaften des Netzes. Die Schwellenfunktion $g(x)$ ist im einfachsten Fall die Stufenfunktion

$$g(x) = \begin{cases} 1 & \text{für} \quad x > 0 \\ 0 & \text{für} \quad x \leq 0 \end{cases}$$

Das Argument von g definiert eine Gerade im Raum der X-Variablen, spezifiziert durch B_1, B_2, α. Die Ausgabe Y ist dann 0 oder 1, je nachdem, ob der Punkt (X_1, X_2) auf der einen oder anderen Seite der Geraden liegt (siehe Abbildung 9.7.)
Für den Fall von n Eingangsvariablen ist die Ausgabefunktion

$$Y = g\left(\sum_{k=1}^{n} B_kX_k - \alpha\right)\,.$$

Dies bedeutet eine Trennebene im n-dimensionalen Raum.
Will man mehr als eine Trennebene, so benötigt man eine verborgene Schicht von Knoten; jeder Knoten der verborgenen Schicht kann eine Trennebene definieren. Abbildung 9.8 zeigt ein solches Netzwerk mit einer Schicht H verborgener Knoten und mehreren Ausgabeknoten Y. Es kann zum Beispiel verwendet werden, wie in Abbildung 9.9 gezeigt, um ein Gebiet in zwei Dimensionen zu spezifizieren, wo der Eingabevektor $\boldsymbol{X}$ liegen soll.
Die Ausgabe eines Netzes mit einer verborgenen Schicht ist der Vektor

$$Y_i(\boldsymbol{X}) = g\left(\sum_j A_{ij}\, g\left(\sum_k B_{jk}X_k - \alpha_j\right) - \beta_i\right)\,. \tag{9.8}$$

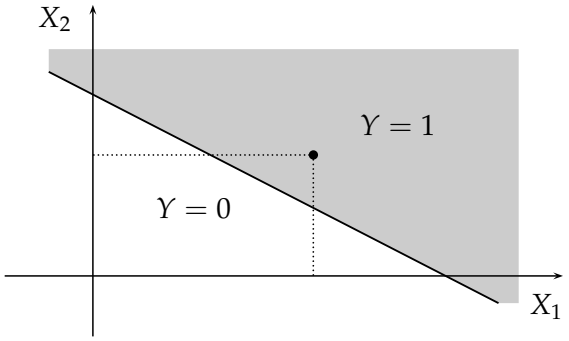


Abbildung 9.7: Einfache Entscheidungsprozedur für zwei Eingangsvariable durch eine Trenngerade.

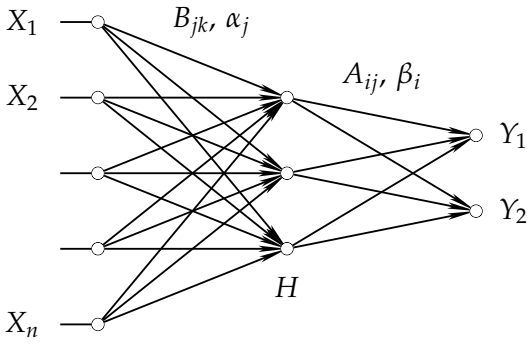


Abbildung 9.8: Ein neuronales Netz mit einer verborgenen Schicht.

Die Gleichung ist für den allgemeinen Fall mehrerer Ergebnis-Ausgänge Y_i formuliert. Dies ist die Form des Netzes, welche häufig in der Praxis verwendet wird. Dabei ist X_k der Eingabevektor, B_{jk} sind seine Gewichte; die verborgene Schicht H hat Schwellenwerte α_j und Gewichte B_{ij}; Y_i ist der Ausgabevektor mit Schwellenwerten β_i. Als Schwellenfunktion $g(x)$ wird statt der Stufenfunktion mit Vorteil ein weicherer Übergang gewählt; gebräuchlich ist

$$g(x) = \frac{1}{2}\,(1 + \tanh \beta x) = \frac{1}{1 + \exp(-2\beta x)}$$

(siehe Abbildung 9.10). Dadurch werden die Übergänge zwischen den Trennebenen abgerundet.

Neuronale Netze können auch logische Funktionen realisieren, zum Beispiel kann die Funktion UND(X_1, X_2) mit 1 = *wahr* und 0 = *falsch* realisiert werden durch das einfache Netz der Abbildung 9.6. Man kann dafür zum Beispiel wählen $B_1 = B_2 = 1$ und $\alpha = 1.5$. Für die Realisierung des exklusiven ODER (XOR(1,1) = XOR(0,0) = 0 ; XOR(1,0) = XOR(0,1) = 1) benötigt man eine verborgene Schicht (siehe Abbildung 9.11).

Wieviele verborgene Schichten sind notwendig oder nützlich? Man kann zeigen, daß zwei verborgene Schichten genügen, und daß für kontinuierliche Funktionen schon mit einer verborgenen Schicht eine beliebig genaue Annäherung möglich ist, wenn die Zahl der verborgenen Knoten groß genug ist [31, 34].

Die Gewichte A_{ij}, B_{jk} und die Schwellen α_j, β_j werden in einer Minimierungsprozedur, genannt Training, bestimmt. Dazu zeigt man dem Netz eine Menge von Fällen, genannt *Muster*, deren Zuordnung zu einer bestimmten Klasse bekannt ist. Die Gewichte und Schwellen werden dann so verändert, daß die Abweichung der Ausgaben Y_i von den erwünschten Ausgaben O_i, gemittelt über alle Muster, minimiert wird. Die zu minimierende Funktion ist

$$E = \frac{1}{2N} \sum_{n=1}^{N} \sum_{i=1}^{I} (Y_i(X_n) - O_i)^2\,,$$

wobei die Summe sich über die I verschiedenen Muster, die es zu unterscheiden gilt, und über die N für das Training des Systems verfügbaren Muster erstreckt. Die Startwerte der Gewichte

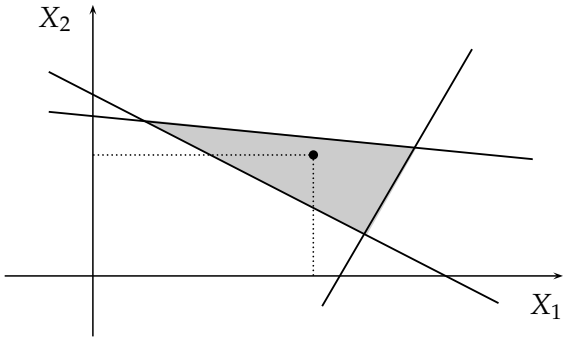


Abbildung 9.9: Definition eines Gebiets durch ein Netz mit einer verborgenen Schicht.

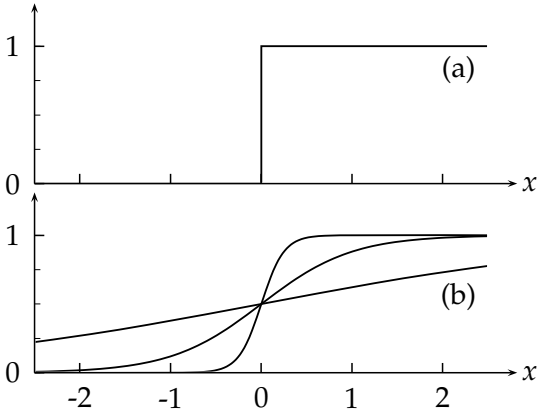


Abbildung 9.10: Schwellenfunktionen: Die Stufenfunktion (a) und die sigmoide Funktion $f(x) = 1/(1 + \exp(-2\beta x))$ für die Werte $\beta = 4$, 1 und 0.25. Die Funktion hat die Eigenschaft $1 - f(x) = f(-x)$.

und Schwellen werden zufällig zwischen $\pm\varepsilon$ gewählt, wobei die Wahl von ε gut überlegt sein will. Durch das Training wird ein optimaler Satz von Gewichten und Schwellen bestimmt. Nach dem Training wird das neuronale Netz eine sehr schnelle Mustererkennung leisten. Wenn die Gewichte und Schwellen feststehen, kann man die Logik des neuronalen Netzes fest in der Hardware verdrahten. Hierfür gibt es kommerzielle Bausteine. Damit sind Entscheidungen im Sub-Mikrosekunden Bereich möglich.

In der Praxis erfordert die Technik neuronaler Netze große Sorgfalt in den Details. Die richtige Wahl der Eingabevariablen ist dabei von zentraler Bedeutung. Es ist vorteilhaft, sie so zu transformieren, daß ihre Werte im großen ganzen alle im selben numerischen Bereich liegen. Dies kann im einfachsten Fall durch eine lineare Transformation geschehen.

Das Verfahren wird umso schwieriger, je mehr Eingabedaten man hat. Es ist deshalb nützlich, festzustellen, welche Variablen überhaupt für die Diskriminierung wichtig sind. Dazu kann man die Eingabevariablen auf Hauptachsen transformieren. Man bildet den Schätzwert ihrer Kovarianzmatrix

$$V_{ik} = \frac{1}{N} \sum (X_i - \overline{X}_i)(X_k - \overline{X}_k) \,,$$

wobei die Summe über alle Muster geht. Man transformiert diese Matrix auf Diagonalform. Die Eigenvektoren sind Linearkombinationen der ursprünglichen Eingabedaten und werden als neue Eingabedaten genommen. Man kann sie einzeln und unabhängig auf ihre Signifikanz für die Diskriminierung zwischen den Klassen prüfen und feststellen, welche wichtig und welche unwichtig sind.

Auch der Optimierungsschritt kann wegen der großen Zahl der anzupassenden Konstanten Probleme erzeugen. Er kann zum Beispiel zu einem Nebenminimum führen. Um dieses Risiko zu verringern, kann eine Temperatur T eingeführt werden, die als Teil der Suchstrategie variiert

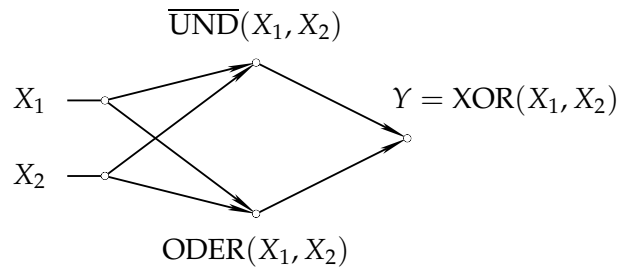


Abbildung 9.11: Ein neuronales Netz zur Realisierung der Funktion XOR(X_1, X_2). Die Ausgabe Y ist das UND der beiden verborgenen Knoten.

wird [73]. Die Netzgleichung (9.8) wird dazu wie folgt modifiziert

$$Y_i(\mathbf{X}) = g\left(\sum_j \frac{1}{T}\left(A_{ij}\, g\left(\sum_k \frac{B_{jk}X_k - \alpha_j}{T}\right) - \beta_i\right)\right) . \tag{9.9}$$

Damit eine Änderung in T nicht durch eine gemeinsame Skalenänderung der Gewichte und Schwellen aufgefangen wird, normiert man die Gewichte zeilenweise: $A_{ik} \to \mathcal{A}_{ik}$ und $B_{ik} \to \mathcal{B}_{ik}$ mit der Normierung

$$\sum_k \mathcal{A}_{ik}^2 = 1 \quad \text{und} \quad \sum_k \mathcal{B}_{ik}^2 = 1\,.$$

Mit diesen Modifizierungen werden wieder durch Minimierung die Gewichte und Schwellen bestimmt, wobei auch die Temperatur in den Minimierungsprozeß eingeschlossen wird. Für $T \to 0$ geht die Schwellenfunktion gegen die Stufenfunktion. Man wählt zunächst einen verhältnismäßig großen Wert von T, etwa $T = 5$, entsprechend einer sanften Schwellenfunktion, und verkleinert T im Verlauf der Minimierung. Im übrigen gilt für die Minimierung das in Kapitel 8 gesagte.

9.3.3 Diskriminanten-Funktionen

Die auf Diskriminanten-Funktionen beruhende Klassifizierungsmethode ist eine Alternative zu neuronalen Netzen. Es kann eine Klassifizierung von Ereignissen anhand einer Zahl n von charakterisierenden Eigenschaften vornehmen, und wie neuronale Netze ist es eine geeignete Methode, wenn mehrere charakterisierende Eigenschaften vorliegen. Im Gegensatz zu neuronalen Netzen ist kein Training erforderlich, wohl aber eine gewisse Datenaufbereitung.

Betrachtet wird als Beispiel die Unterscheidung von zwei Kategorien von Ereignissen, die mit *Signal* und *Untergrund* bezeichnet werden. Die Eigenschaft eines Ereignisses ist durch einen n-Vektor $\boldsymbol{x}$ gegeben, der einen Punkt in einem n-dimensionalen Raum darstellt. Es liegt eine Stichprobe von M Ereignssen vor, aus denen das Signal abzutrennen ist. Der Algorithmus erfordert die Berechnung einer sogenannten Diskriminante D, die anschaulich das Verhältnis von Signal zu Signal plus Untergrund bezeichnet:

$$D(x) = \frac{\rho_s(x)}{\rho_s(x) + \rho_u(x)} , \tag{9.10}$$

wobei $\rho_s(x)$ bzw. $\rho_u(x)$ die jeweiligen Dichten der Signal- und Untergrund-Ereignisse sind. Diese Dichten liegen in der Praxis meist nicht als Funktionen vor, sondern sind durch grosse Menge von mit Monte-Carlo Verfahren erzeugten Signal- und Untergrundereignissen gegeben. Dabei sollte das Verhältnis in der Anzahl von Signal- zu Untergrundereignissen ungefähr dem tatsächlich erwarteten entsprechen.

Die Dichten $\rho_s(x)$ bzw. $\rho_u(x)$ sind angenähert proportional der Zahl von entsprechenden Monte-Carlo Ereignissen in einer passend (nicht zu gross und nicht zu klein) gewählten Umgebung in

dem n-dimensionalen Parameterraum der n Eigenschaften, welche die Ereignisse charakterisieren. Für jedes der M zu diskriminierenden Ereignis wird nun der Wert der Diskriminanten $D(x)$ bestimmt aus den Monte-Carlo Ereignissen in der Nähe des jeweiligen Phasenraumpunktes, wobei insgesamt N Monte-Carlo Ereignisse vorliegen. In der Häufigkeitsverteilung $D(x)$ der Ereignisse bezeichnen Werte nahe an 1 diejenige Wahl von Parametern im n-dimensionalen Parameterraum, welche eine gute Trennung des Signals vom Untergrund erlauben.

Die Anforderungen an Speicherplatz sind proportional zu dem Produkt $n(N + M)$; bei Benutzung eines einfachen sequentiellen Vergleichs wird eine Rechenleistung proportional zu nNM benötigt. Angesichts dauernd steigender Rechnerleistungen ist die Anforderung bzgl. Speicherplatz eher zu erfüllen, aber die erforderliche Rechenleistung kann enorm sein. Es wird daher notwendig, spezielle Suchprogramme zu verwenden, bei denen der Faktor N in der erforderlichen Rechenleisung durch $\log N$ ersetzt werden kann. Ein solches Verfahren ist *range searching* [39, 77, 70], das im Kapitel 2.5 beschrieben wurde (siehe auch Anhang). Damit kann diese Diskriminanten-Methode auch bei sehr großen Datenmengen noch durchgeführt werden[75].

Die Methode der Diskriminaten-Funktionen kann auf allgemeinere Probleme mit mehr als zwei Kategorien angewendet werden[7]. In einer vorbereitenden Datenaufbereitung kann zum Beispiel die Bedeutung des Abstands, durch den die Umgebung definiert wird, zwischen den einzelnen Komponenten des n-Vektors $\boldsymbol{x}$ aneinander angeglichen werden. Es müssen *Schwellenwerte* zur Annahme oder Zurückweisung von Hypothesen bestimmt werden. Die Klassifizierung kann bzgl. verschiedener Anforderungen optimiert werden. Eine mögliche Anforderung ist eine minimale Wahrscheinlichkeit für Misklassifizierung, und eine andere Anforderung kann es sein, z.B. das Risiko des Verlusts einer bestimmten Klasse zu reduzieren. Dazu können a-priori Wahrscheinlichkeiten oder eine Verlust-Matrix eingeführt werden[7], die den Ausdruck von $D(x)$ (9.10) modifizieren.

10 Parametrisierung von Daten

10.1 Einleitung

Eine häufig vorkommende Aufgabe ist die mathematische Beschreibung der empirischen Abhängigkeit einer meßbaren oder in Tabellenform vorliegenden Größe y von einer Variablen x. Gesucht ist die mathematische Funktion $f(x)$, die vorliegende Daten möglichst gut in ihrem Verhalten als Funktion von x repräsentiert. In diesem Kapitel wird angenommen, daß, zum Beispiel durch eine Messung, Daten in der Form

$$x_i,\, y_i,\, \sigma_i \qquad i = 1,\, 2,\, \ldots n \tag{10.1}$$

vorliegen. Dabei kann die Größe y_i als Meßwert der Funktion $f(x)$ an der Stelle x_i angesehen werden. Die Genauigkeit eines Datenpunktes ist durch die Standardabweichung σ_i gegeben. In der Anwendung wird die Standardabweichung in das *Gewicht* $w_i = 1/\sigma_i^2$ des Datenpunktes umgerechnet. Wenn eine Abhängigkeit $y = f(x)$ parametrisiert ist, sind Interpolation, Integration und Differentiation der Abhängigkeit $f(x)$ einfach.

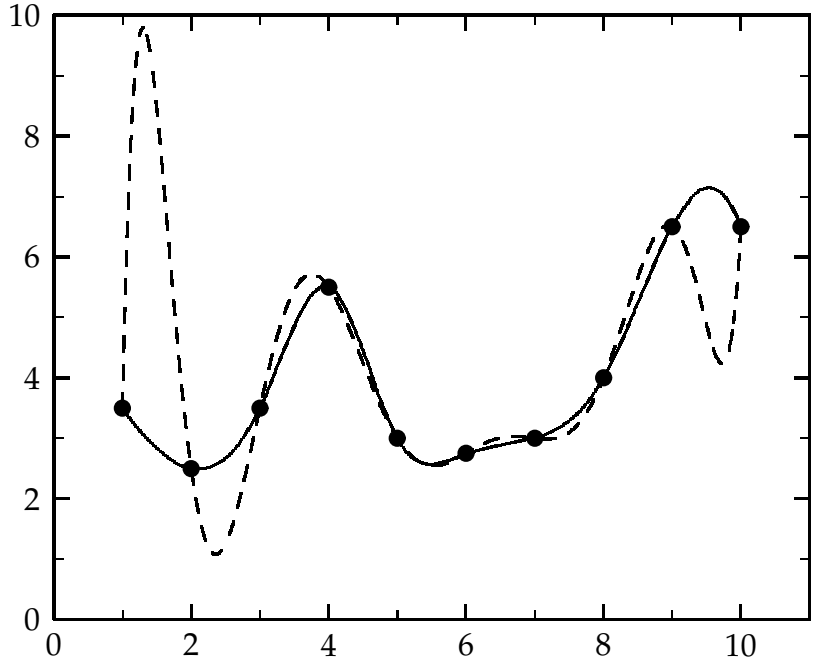


Abbildung 10.1: Interpolation zwischen 10 vorgegebenen Punkten (y_i, x_i) durch eine kubische Spline-Funktion (durchgezogene Kurve) und durch ein Polynom neunten Grades (gestrichelte Kurve).

Exakte Interpolation. Bei genauen Daten (10.1) kann eine *Interpolationsfunktion*, die *exakt* durch die Wertepaare (x_i, y_i) geht, sinnvoll sein. Die Abbildung 10.1 zeigt die exakte Interpolation von 10 Punkten (x_i, y_i) durch ein Polynom neunten Grades (gestrichelte Kurve); die Kurve zeigt die für ein Interpolationspolynom hohen Grades typischen starken Oszillationen. Die durchgezogene Kurve zeigt die Interpolation durch eine kubische Spline-Funktion; die Kur-

ve ist aus Polynomen niedrigen Grades (kubisch) zusammengesetzt, die an Intervallgrenzen bestimmte Stetigkeitsforderungen erfüllen.

Regression. Bei fehlerbehafteten Daten ist die exakte Interpolation nicht mehr sinnvoll. Die Bestimmung einer Abhängigkeit $y = f(x)$ aus fehlerbehafteten Daten wird *Regression* genannt. Wenn für einen Satz Meßdaten eine spezielle Parametrisierung der x-Abhängigkeit in Form einer Funktion $f(x, \boldsymbol{a})$, die von Parametern $\boldsymbol{a}$ abhängt, vorliegt, dann ist eine Anpassung der Parametrisierung an die Daten (10.1) mit Standardmethoden, zum Beispiel mit der Methode der kleinsten Quadrate (Kapitel 7), sinnvoll; die Anpassung besteht in der Bestimmung optimaler Werte der Parameter.

Wenn keine Parametrisierung bekannt ist, muß man von einem allgemeinen Ansatz ausgehen. Dabei ist eine Parametrisierung vorzuziehen, die *linear* von den Parametern abhängt. Zwar können die diskreten Datenpunkte durch die Verwendung von vielen Parametern besser reproduziert werden, aber die wahre Abhängigkeit muß dadurch nicht besser beschrieben werden als mit wenigen Parametern. Es ist also eine optimale Anzahl von Parametern zu finden. In diesem Kapitel werden solche allgemeinen Parametrisierungen mit Hilfe von *Spline-Funktionen* und durch *Systeme orthogonaler Funktionen* behandelt.

Lineare Interpolation und Spline-Funktionen. Die einfachste Form einer Parametrisierung ist eine lineare Interpolation zwischen benachbarten Meßpunkten. Falls die Punkte dicht liegen und genau sind, mag das genügen. Die Formel für eine lineare Interpolation von Daten der Form (10.1) im Intervall $[x_i,\, x_{i+1}]$ ist

$$S_i^{\mathrm{lin}}(x) = \left(\frac{x_{i+1} - x}{x_{i+1} - x_i}\right) y_i + \left(\frac{x - x_i}{x_{i+1} - x_i}\right) y_{i+1} = a_i y_i + b_i y_{i+1} \,, \tag{10.2}$$

der interpolierte Funktionswert ist das gewichtete Mittel benachbarter y-Werte mit x-abhängigen Gewichtsfaktoren a_i und b_i (Summe $a_i + b_i = 1$). Die lineare Interpolation kann als Spezialfall der Interpolation durch eine Spline-Funktion aufgefaßt werden, die in diesem Fall stückweise linear ist und an gewissen Werten von x Unstetigkeiten in der ersten Ableitung aufweist.

Allgemein ist eine eindimensionale Spline-Funktion [17, 18] in jeweils einem x-Intervall durch ein Polynom n-ten Grades gegeben; an den Intervallgrenzen sind die Ableitungen bis zum $(n-1)$-ten Grad stetig. Kubische Spline-Funktionen bestehen stückweise aus kubischen Polynomen, sie haben Unstetigkeiten in der dritten Ableitung und sind stetig bis zur zweiten Ableitung. Solche kubischen Spline-Funktionen sehen glatt aus und eignen sich gut zur Interpolation. Weiteres ist in Abschnitt 10.2 behandelt.

Polynome. Eine einfache *globale* Parametrisierung ist ein Polynom m-ten Grades mit $(m+1)$ Koeffizienten $a_0,\, a_1, \ldots a_m$

$$f(x) = a_0 + a_1 x^1 + a_2 x^2 + \ldots + a_m\, x^m = \sum_{j=0}^{m} a_j x^j \;; \tag{10.3}$$

hierbei beeinflußt jeder Koeffizient das ganze x-Intervall. Der optimale Grad m ist nicht a-priori bekannt, er sollte ausreichend hoch gewählt werden, jedoch auch nicht zu hoch. Man führt daher in der Praxis meist Anpassungen mit verschiedenem Polynomgrad durch und vergleicht die Güte der Anpassung, zum Beipiel mit einem χ^2-Test. Dabei treten zwei Probleme auf. Erstens ändern alle Koeffizienten ihren Wert, wenn der Grad m des Polynoms verändert wird, und zweitens wird die Matrix der Normalgleichungen bei der Methode der kleinsten Quadrate ein ungünstiges numerisches Verhalten haben. Beide Probleme können durch die Benutzung von Systemen orthogonaler Polynome vermieden werden.

Systeme orthogonaler Funktionen. Orthogonale Polynome sind spezielle orthogonale Funktionen. Ein System $\{p_j(x)\}$ von Funktionen $p_j(x)$ mit $j = 1, 2, \ldots$, definiert für ein Intervall $a \leq x \leq b$, ist ein orthogonales System, wenn das innere Produkt von je zwei der Funktionen (p_j, p_k) für $j \neq k$ verschwindet. Das innere Produkt kann durch eine Summe oder ein Integral

$$(p_j, p_k) = \int_a^b w(x) \cdot p_j(x)\, p_k(x)\ \mathrm{d}x \qquad \text{for} \quad j \neq k \tag{10.4}$$

mit einer x-abhängigen Gewichtsfunktion $w(x)$ definiert sein. Dann gilt

$$(p_j, p_k) = \int_a^b w(x) \cdot p_j(x)\, p_k(x)\, \mathrm{d}x = 0 \qquad \text{für} \quad j \neq k\,. \tag{10.5}$$

Orthogonale Funktionen heißen *normiert*, wenn die Werte N_j in dem Ausdruck

$$(p_j, p_j) = \int_a^b w(x) \cdot p_j^2(x)\, \mathrm{d}x = N_j$$

für alle Indizes j gleich eins sind. Ein System orthogonaler Funktionen kann aus einem System linear unabhängiger Funktionen durch den Prozeß der *Orthogonalisierung*[25] konstruiert werden. Eine Parametrisierung $y = f(x)$ kann in einem System orthogonaler und normierter Funktionen in einer Summe

$$f(x) = \sum_{j=1}^{m} a_j p_j(x) \tag{10.6}$$

dargestellt werden; dabei sind die Faktoren a_j, die Koeffizienten der Funktion $p_j(x)$, bestimmt durch

$$a_j = (p_j(x), f(x)) = \int_a^b w(x) \cdot p_j(x) f(x)\, \mathrm{d}x\,; \tag{10.7}$$

sie können also *einzeln* bestimmt werden.

In der Praxis gibt es bei glatten Funktionen $f(x)$ meist nur wenige signifikante Koeffizienten; oberhalb eines Index m_0 werden die Koeffizienten a_j schnell klein und können vernachlässigt werden. Dadurch ist eine Parametrisierung $f(x)$ durch eine Summe weniger Terme möglich, die durch eine Rekursionsrelation zwischen den Funktionen $p_j(x)$ schnell ausgewertet werden kann. Weiteres über orthogonale Polynome findet man in Abschnitt 10.3.

10.2 Spline-Funktionen

Interpolierende kubische Spline-Funktionen. Die exakte Interpolation der Paare (x_i, y_i) (unter Nichtbeachtung eventueller Meßfehler in den y_i) durch kubische Spline-Funktionen wird betrachtet. Die Menge $X = \{x_1, x_2, \ldots x_n\}$ definiert eine Einteilung des Intervalls $[x_1, x_n]$, wobei die Ordnung $x_i < x_{i+1}$ angenommen wird. Die Intervallgrenzen, Knoten genannt, fallen in dieser Anwendung mit den Koordinaten x_i der Datenpunkte zusammen (bei anderen Anwendungen gilt dies nicht).

Eine kubische Spline-Funktion $S(x)$ mit $S(x_i) = y_i,\ i = 1, 2, \ldots n$, ist eine zweimal stetig differenzierbare Funktion in dem Intervall $[x_1, x_n]$. In jedem Unterintervall $[x_i, x_{i+1}],\ i = 1, 2, \ldots (n-1)$ ist sie gegeben durch ein Polynom dritten Grades

$$S_i(x) = a_i + b_i(x - x_i) + c_i(x - x_i)^2 + d_i(x - x_i)^3, \quad i = 1, 2, \ldots (n-1)\,. \tag{10.8}$$

An jedem inneren Knoten x_i stimmen die beiden Polynome der angrenzenden Unterintervalle in den Funktionswerten und den Werten der ersten beiden Ableitungen überein. Eine kubische Spline-Funktion mit insgesamt $4(n-1) = 4n-4$ Koeffizienten für die $(n-1)$ Intervalle ist durch die Interpolations- und Stetigkeitsforderungen noch nicht vollständig festgelegt. Die Forderung $S(x_i) = y_i$ liefert n Gleichungen für die Koeffizienten, und mit den $(n-2)$ Bedingungen an den inneren Stützstellen, jeweils für $S(x)$, $S'(x)$ und $S''(x)$, sind es insgesamt $n + 3(n-2) = 4n-6$ Bedingungen für die $4n-4$ Koeffizienten. Zwei Freiheitsgrade müssen durch zwei zusätzliche Bedingungen fixiert werden. Die folgenden Bedingungen sind üblich:

(1) $S''(x_1) = 0\,, \quad S''(x_n) = 0\,.$ Eine Spline-Funktion mit dieser Bedingung wird *natürliche* Spline-Funktion genannt; diese Bedingung ergibt minimale Krümmung (siehe Gleichung (10.9)).

(2) $S'(x_1) = y'_1\,, \quad S'(x_n) = y'_n\,.$ Schätzwerte y'_1 und y'_n für die ersten Ableitungen an den Intervallgrenzen werden vorgegeben.

(3) $S''(x_1) = y''_1\,, \quad S''(x_n) = y''_n\,.$ Schätzwerte y''_1 und y''_n für die zweiten Ableitungen an den Intervallgrenzen werden vorgegeben.

(4) $S'''(x)$ stetig über x_2 and über x_{n-1}. Beim ersten und beim letzten *inneren* Knoten ist auch die dritte Ableitung stetig; die Bedingung heißt kurz *not-a-knot* (die beiden Knoten führen *nicht* zu einer Unstetigkeit in der dritten Ableitung).

Eine interpolierende Funktion soll möglichst *glatt* sein. Die lokale Krümmung einer Funktion $f(x)$ ist gegeben durch

$$f''(x)/(1+f'^2(x))^{3/2} \qquad \text{(lokale Krümmung)}$$

und kann für $|f'(x)| \ll 1$ durch f'' angenähert werden. Ein gutes Maß für die Glattheit einer Funktion $f(x)$ ist daher das Integral

$$\int_{x_1}^{x_n} \left[f''(x)\right]^2 \,\mathrm{d}x \qquad \text{(totale Krümmung)}\,, \tag{10.9}$$

das als die (totale) Krümmung der Funktion $f(x)$ im Intervall $[x_1, x_n]$ bezeichnet werden kann. Die natürliche Spline-Funktion nach Bedingung (1) ist die glatteste Interpolationsfunktion im Sinn der Krümmung. Die Ungleichung

$$\int_a^b |S''(x)|^2\,\mathrm{d}x \le \int_a^b |f''(x)|^2\,\mathrm{d}x$$

kann bewiesen werden und zeigt, daß die natürliche Spline-Funktion $S(x)$ nach Bedingung (1) unter allen zweifach differenzierbaren, interpolierenden Funktionen minimale totale Krümmung hat. Angenähert ist auch die Biegungsenergie eines elastischen Lineals (spline) minimal, und diese Eigenschaft hat den Funktionen ihren Namen gegeben.

Ein glatter Verlauf der Interpolationsfunktion ist jedoch nicht das einzige Kriterium. Wichtiger ist die Genauigkeit, mit der eine gegebene Funktion $f(x)$ durch die Spline-Funktion $S(x)$ angenähert werden kann. Von diesem Gesichtspunkt aus sind Spline-Funktionen mit Bedingung (1) nicht optimal, außer wenn tatsächlich $f''(x_1) = f''(x_n) = 0$ ist. Wenn die tatsächlichen Steigungen oder die zweiten Ableitungen an den Endpunkten bekannt sind, dann sind die Bedingungen (2) bzw. (3) besser. Wenn nichts über die Ableitungen an den Endpunkten bekannt ist, kann man Bedingung (4) benutzen. Diese allgemeine Bedingung wird im folgenden bevorzugt.

Spline-Funktionen $S(x)$ haben optimale Eigenschaften. Die Differenz $|f(x) - S(x)|$ hat eine obere Schranke, die proportional zur vierten Potenz des Knotenabstands ist. Auch die k-ten Ableitungen der Funktion $f(x)$ bis $(k = 3)$ werden gut angenähert, mit einer Schranke proportional zur $(4 - k)$-ten Potenz des Stützstellenabstands. Die natürliche Spline-Bedingung erzeugt dagegen einen Fehler proportional zum Quadrat des Stützstellenabstands an den Enden, was durch die Bedingung (4) vermieden wird. Sie ist für die Spline-Funktion in Abbildung 10.1 verwendet worden.

Bestimmung der Spline-Koeffizienten. Man geht aus von einer Splinefunktion $S_i(x)$ nach Gleichung (10.8) im Intervall $x_i \leq x \leq x_{i+1}$. Die $(4n - 4)$ Koeffizienten a_i, b_i, c_i, d_i für $i = 1, 2, \ldots (n-1)$ bestimmen sich aus den Interpolations- und Stetigkeitsbedingungen an den Intervallgrenzen

$$\begin{aligned} S_i(x_i) &= y_i & i &= 1,\, 2, \ldots\, n \\ S_i(x_{i+1}) &= y_{i+1} & i &= 1,\, 2, \ldots\, n-1 \\ S_i'(x_{i+1}) &= S_{i+1}'(x_{i+1}) & i &= 1,\, 2, \ldots\, n-2 \\ S_i''(x_{i+1}) &= S_{i+1}''(x_{i+1}) & i &= 1,\, 2, \ldots\, n-2\,. \end{aligned}$$

Zu diesen $(4n - 6)$ Gleichungen für $(4n - 4)$ Koeffizienten kommen noch zwei Gleichungen aus einer der Bedingungen (1) bis (4). Dies führt auf eine lineares Gleichungssystem für die Koeffizienten.

Wesentliche Parameter einer Spline-Funktion sind die sogenannten *Momente* $M_i = S_i''(x_i)$, die zweiten Ableitungen von $S_i(x)$ an den Werten x_i. Diese Momente sind durch ein lineares Gleichungssystem bestimmt und ergeben alle anderen Koeffizienten.

Die zweite Ableitung ist stückweise linear; daher gilt eine lineare Interpolationsformel analog zu Gleichung (10.2)

$$S_i''(x) = a_i\, M_i + b_i\, M_{i+1} \tag{10.10}$$

mit den gleichen x-abhängigen Gewichtsfaktoren a_i und b_i. Für die Funktion $S(x)$ ist zu der linearen Interpolationsformel (10.2) noch ein kubisches Polynom entsprechend den zweiten Ableitungen (10.10) zu addieren, das an den Intervallgrenzen x_i und x_{i+1} verschwindet. Das Ergebnis ist die Gleichung

$$S_i(x) = a_i\, y_i + b_i\, y_{i+1} + c_i\, M_i + d_i\, M_{i+1} \tag{10.11}$$

mit x-abhängigen Gewichten

$$c_i = \frac{1}{6}\left(a_i^3 - a_i\right)\left(x_{i+1} - x_i\right)^2 \qquad d_i = \frac{1}{6}\left(b_i^3 - b_i\right)\left(x_{i+1} - x_i\right)^2\,. \tag{10.12}$$

Die Momente $M_i,\ M_{i+1}$ sind nun durch die Stetigkeitsforderung der ersten Ableitungen

$$S_i'(x) = \frac{y_{i+1} - y_i}{x_{i+1} - x_i} - \frac{3a_i^2 - 1}{6}\left(x_{i+1} - x_i\right)\, M_i + \frac{3b_i^2 - 1}{6}\left(x_{i+1} - x_i\right)\, M_{i+1}$$

bestimmt. Die Stetigkeitsforderung gibt für $j = 2,\, 3,\, \ldots n-1$ die $(n-2)$ linearen Gleichungen

$$M_{i-1}\frac{x_j - x_{j-1}}{6} + M_i\frac{x_{j+1} - x_{j-1}}{3} + M_{i+1}\frac{x_{j+1} - x_j}{6} = \frac{y_{j+1} - y_j}{x_{j+1} - x_j} - \frac{y_j - y_{j-1}}{x_j - x_{j-1}}\,.$$

Hinzu kommen noch die beiden vorher erwähnten weiteren Bedingungen. Das Gleichungssystem kann in einer Matrixgleichung mit einer tridiagonalen Matrix geschrieben werden, die nur in einem Band der Breite 1 um die Diagonale Elemente ungleich null hat.

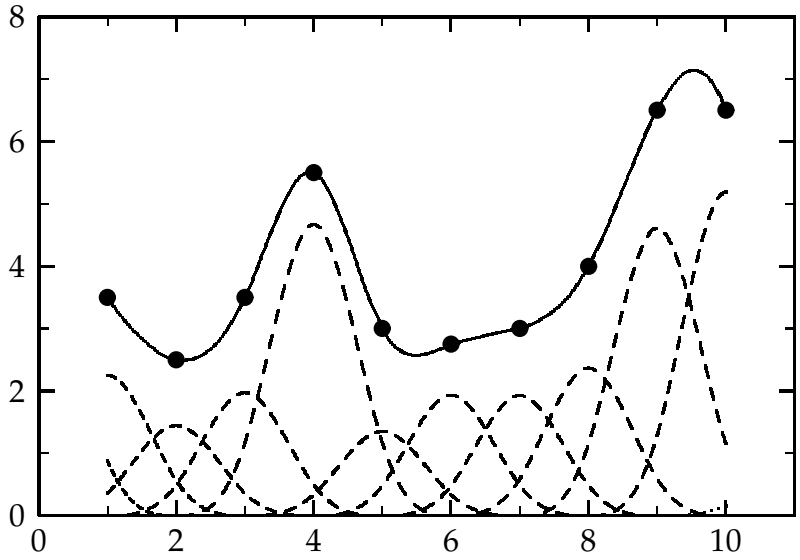


Abbildung 10.2: Die interpolierende Spline-Funktion der vorhergehenden Abbildung und ihre Zerlegung in individuelle B-Splines

Die Lösung des Gleichungssystems läßt sich in einem kompakten, auf der Cholesky-Zerlegung (Kapitel 3.6) beruhenden Algorithmus schreiben, in dem die Elemente der tridiagonalen Matrix erst unmittelbar vor der Verwendung berechnet werden.

Das Unterprogramm `CSPLN` *im Anhang berechnet für Daten der Form* (10.1) *alle Koeffizienten der Spline-Funktion und speichert sie in einem Feld* `C(5,N)` *ab. Die allgemeine Bedingung (4) wird benutzt.*

Die Funktion `CSPF` *im Anhang berechnet den Funktionswert der Spline-Funktion für ein gegebenes Argument* x*. Das Feld* `C(5,N)` *enthält die in Unterprogramm* `CSPLN` *bestimmten Koeffizienten. Wenn* x *außerhalb des Intervalls liegt, erfolgt Extrapolation aus dem ersten bzw. letzten Unterintervall.*

B-Splines. Eine andere Darstellung von Spline-Funktionen als die in Gleichung (10.8) gegebene kann durch die sogenannten *Basis-Spline-Funktionen*, auch kurz B-Splines genannt, erfolgen [17, 18]. Eine einzelne B-Spline-Funktion der Ordnung k ist nur in einem beschränkten Gebiet von x, der *Basis*, von null verschieden. Eine Spline-Funktion $S(x)$ der Ordnung k kann dargestellt werden durch eine Summe von B-Splines $B_{j,k}(x)$

$$S(x) = \sum_j a_j B_{j,k}(x) \ , \tag{10.13}$$

wobei die Funktionen $B_{j,k}(x)$ B-Splines der Ordnung k sind. B-Splines der Ordnung k sind stückweise Polynome des $(k-1)$-ten Grades. Die Abbildung 10.2 zeigt noch einmal die kubische Spline-Funktion $S(x)$ von Abbildung 10.1 als durchgezogene Kurve. Die gestrichelten Kurven zeigen die einzelnen kubischen B-Splines der Ordnung $k = 4$, die durch Linearkombination (10.13) die Funktion $S(x)$ darstellen. Die Darstellung durch B-Splines hat für gewisse Anwendungen Vorteile, zum Beipiel wird eine Anpassung von Daten nach kleinsten Quadraten durch Spline-Funktionen in diesem Fall zu einem linearen Problem (siehe Gleichung (10.13)) und kann in einem Schritt gelöst werden.

B-Splines sind definiert über einer Menge von Knoten $\{t_j\}$. Diese Knoten sind bestimmte x-Werte und müssen nicht notwendig mit den x-Werten von Interpolationspunkten übereinstimmen. In der Abbildung 10.2 fallen die Knoten t_j mit den x-Koordinaten der Datenpunkte zusammen. Die Knoten sollen entsprechend $t_j \leq t_{j+1}$ geordnet sein; zwei oder mehr Knoten können den gleichen Wert annehmen (mehrfache Knoten).

B-Splines lassen sich rekursiv definieren. B-Splines der Ordnung $k = 1$ sind Funktionen mit konstantem Wert 1 in einem Knotenintervall, entsprechend der Definition

$$B_{j,1}(x) = \begin{cases} 1 & t_j \leq x < t_{j+1} \\ 0 & \text{sonst .} \end{cases} \tag{10.14}$$

Durch Linearkombination (10.13) von B-Splines der Ordnung $k = 1$ erhält man eine stückweise konstante Funktion mit Sprungstellen an den Knoten. Ein Histogramm ist eine solche Spline-Funktion. B-Splines höherer Ordnung $B_{j,k}(x)$ (mit $k > 1$) sind durch die Rekursionsbeziehung

$$B_{j,k} = \frac{x - t_j}{t_{j+k-1} - t_j} B_{j,k-1}(x) + \frac{t_{j+k} - x}{t_{j+k} - t_{j+1}} B_{j+1,k-1}(x) \tag{10.15}$$

definiert, die $B_{j,k}(x)$ durch eine Linearkombination von B-Splines der Ordnung $k - 1$ mit positiven Gewichten darstellt.

B-Splines $B_{j,k}(x)$ und die entsprechenden Spline-Funktionen $S(x)$ nach Gleichung (10.13) sind für $k = 2$ stückweise linear, für $k = 3$ stückweise quadratisch und für $k = 4$ stückweise kubisch. Ein einzelner kubischer B-spline $B_{j,4}(x)$ ist in Abbildung 10.3 zusammen mit den Ableitungen im Falle äquidistanter Knotenabstände gezeigt. B-Splines $B_{j,k}(x)$ sind *lokal unterstützt* entsprechend

$$B_{j,k} = \begin{cases} > 0 & t_j \leq x < t_{j+k} \\ 0 & \text{sonst} \end{cases} ,$$

d.h. sie nehmen innerhalb ihres Gebiets $[t_j, t_{j+k}]$ positive Werte an und verschwinden außerhalb ihres Gebiets. In jedem einzelnen Knoten-Intervall $[t_j, t_{j+1}]$ sind von null verschieden nur die k B-splines $B_{j-k+1,k}(x) \ldots B_{j,k}(x)$. B-Splines, wie sie durch die Gleichungen (10.14) und die Rekursionsbeziehung (10.15) definiert sind, sind normiert in dem Sinn, daß ihre Summe gemäß

$$\sum_j B_{j,k}(x) = \sum_{j=l+1-k}^{l} B_{j,k}(x) = 1 \qquad t_l \leq x < t_{l+1} \tag{10.16}$$

an jeder Stelle x gleich 1 ist. Im folgenden werden nur kubische B-Splines betrachtet.

B-Splines mit äquidistanten Knoten. Diese speziellen B-Splines werden mit $b_j(x) = B_{j,4}(x)$ bezeichnet. Mit dem Knotenabstand d gilt für die Knoten $t_{j+1} = t_j + d$. Die expliziten Formeln für die B-Splines $b_j(x)$, ausgedrückt durch die normierte Variable z mit $0 \leq z < 1$, sind

$$b_j(x) = \begin{cases} 1/6\, z^3 & t_j \leq x < t_{j+1} \\ 1/6\,[1 + 3(1 + z(1 - z))z] & t_{j+1} \leq x < t_{j+2} \\ 1/6\,[1 + 3(1 + z(1 - z))(1 - z)] & t_{j+2} \leq x < t_{j+3} \\ 1/6\,(1 - z)^3 & t_{j+3} \leq x < t_{j+4} \\ 0 & \text{sonst .} \end{cases}$$

Die Variable z ist jeweils der durch d dividierte Abstand des x-Wertes von der linken Intervallgrenze; zum Beispiel ist im mittleren Fall $z = (x - t_{j+2})/d$. Ein einzelner B-Spline $b_j(x)$ zusammen mit seinen Ableitungen ist in Abbildung 10.3 dargestellt.

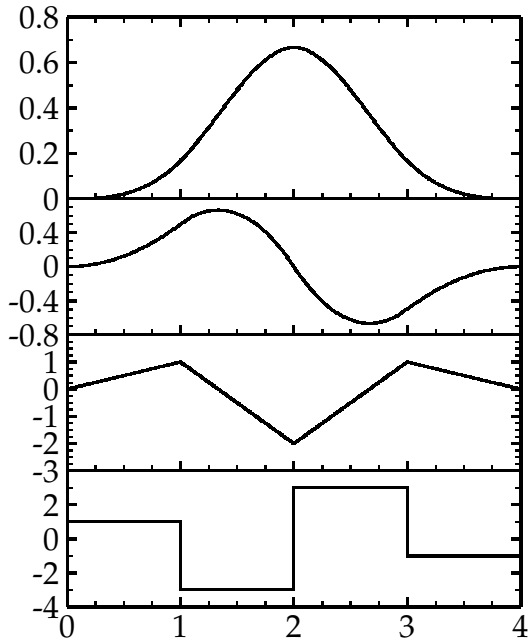


Abbildung 10.3: Ein einzelner B-Spline, oben die Funktion und darunter die erste, zweite und dritte Ableitung für Knoten im Abstand $d = 1$.

Bei der Interpolation von n äquidistanten Datenpunkten (10.1) mit $d = x_{i+1} - x_i$ braucht man $(n + 2)$ B-Splines und die Knoten $t_1 = x_1 - d$, $t_2 = x_1$, ..., $t_{n+1} = x_n$, $t_{n+2} = x_n + d$. Die Bestimmung der Koeffizienten einer Spline-Funktion kann in der Parametrisierung (10.13) mit kubischen B-Splines $b_j(x)$ sehr einfach sein. Es sind lediglich die Koeffizienten a_j der Parametrisierung (10.13) zu bestimmen.

Das Unterprogramm `USPC` *im Anhang berechnet für äquidistante Datenpunkte* `Y(N)` *zwischen den Grenzen* `XA` *und* `XB` *die Koeffizienten* `A(N)`*. Durch Verwendung der not-a-knot Bedingung ist die Zahl der Koeffizienten gleich der Zahl der Datenpunkte.*

Die Funktion `USPF` *im Anhang berechnet einen Funktionswert bei Angabe der Abszisse* `X`*, der Koeffizienten im Feld* `A(N)` *und der Grenzen* `XA` *und* `XB`*. Für Argumente außerhalb des Intervalls erfolgt Extrapolation.*

Allgemeine B-Splines. Durch Anpassung der Knotenpositionen an die Daten kann die Güte der Anpassung wesentlich verbessert werden. Insbesondere bei Daten, die in weiten Bereichen der Variablen x nahezu strukturlos sind, in engen Bereichen jedoch eine große Krümmung aufweisen, ist eine gute Anpassung gerade durch allgemeine B-Splines möglich, die durch die Rekursionsformel (10.15) gegeben sind.

Das Unterprogramm `BSPLNK` *im Anhang berechnet für ein gegebenes Argument* `X` *die Werte der B-Spline-Funktionen* $B_{j,k}$ *für* $j = l, l+1, \ldots l+k-1$*, für einen Index* l *und eine Ordnung* k *der B-Splines* ($k \leq 10$)*, die durch die Argumente* `K` *und* `L` *gegeben sind.*

In einer Parametrisierung (10.13) durch B-Splines sind die Ableitungen $S'(x)$, $S''(x)$, ... und auch das Integral von $S(x)$ über einen x-Bereich jeweils durch Linearkombinationen der Koeffizienten a_j gegeben. Diese Eigenschaft erlaubt eine *lineare* Anpassung von fehlerbehafteten Daten nach kleinsten Quadraten mit B-Splines; die Genauigkeit der einzelnen Daten ist durch ein entsprechendes Gewicht in den Normalgleichungen zu berücksichtigen. Nach der Bestimmung der Koeffizienten a_j und ihrer Kovarianzmatrix sind die Fehler für interpolierte Funktionswerte durch lineare Fehlerfortpflanzung einfach zu berechnen.

Das Unterprogramm `BSKFIT` *im Anhang paßt eine allgemeine Spline-Funktion an Daten der Form* (10.1) *an.*

Nach der Anpassung können Funktionswerte der angepaßten Spline-Funktion mit Hilfe der Funktion `BSKFUN` *im Anhang für einen bestimten Wert der Abszisse* `X` *berechnet werden. Die anderen Argumente der Funktion sind die gleichen wie bei dem Unterprogramm* `BSKFIT`.

Beispiel 10.1 Anpassung einer Spline-Funktion vom Grad $k = 4$ an Daten.

Die Daten (y_i, x_i), $i = 1, 2, \ldots 20$ der Abbildung 10.4 zeigen ein scharfes Maximum um $x = 900$, außerhalb des Maximums ist der Verlauf nahezu konstant. Eine Polynomanpassung (Abbildung 10.6) gibt hier ein unbefriedigendes Ergebnis. Eine allgemeine B-Spline-Funktion mit angepaßten Knotenabständen ist geeigneter; die Abbildung 10.4 zeigt die durch Kreuze markierten Knotenpositionen. Die durchgezogene Kurve gibt die Anpassung wieder, die in allen Bereichen die Daten gut beschreibt, ohne den lokalen statistischen Fluktuationen zu folgen.

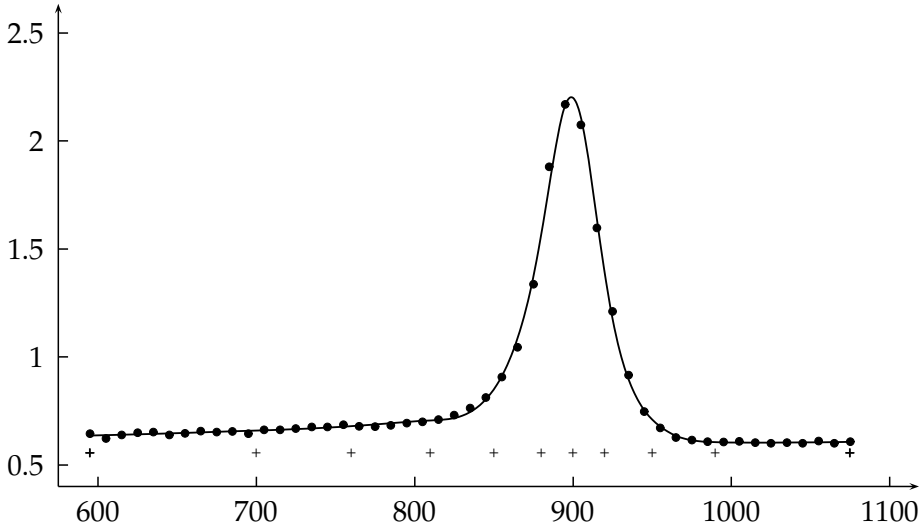


Abbildung 10.4: Anpassung einer kubischen Spline-Funktion an tabellierte Daten. Die nicht-äquidistanten Knoten sind als Kreuze gezeigt; ihre Position ist dem Verlauf der Abhängigkeit $y = f(x)$ angepaßt.

10.3 Orthogonale Polynome

Definition. Ein System orthogonaler Polynome $p_k(x)$ im Bereich $x_1 \leq x \leq x_n$ kann aus den einfachen Potenzfunktionen $1, x, x^2, \ldots x^n$ konstruiert werden durch

$$p_k(x) = \sum_{\nu=0}^{k} b_{k\nu} x^\nu \; . \tag{10.17}$$

Für einen Satz von n Datenpunkten (x_i, y_i) mit den Standardabweichungen σ_i von y_i nach (10.1) sind die zugehörigen orthogonalen Polynome $p_k(x)$ definiert durch

$$\sum_{i=1}^{n} w_i \, p_j(x_i) p_k(x_i) = \begin{cases} 1 & \text{für} \quad j = k \\ 0 & \text{für} \quad j \neq k \end{cases} \tag{10.18}$$

mit den Gewichten $w_i = 1/\sigma_i^2$. Die Summe geht über alle n Datenpunkte. Orthogonale Polynome mit der Normierung auf eins werden als *orthonormale* Polynome bezeichnet.

Die Datenpunkte können nun durch eine Summe über m orthonormale Polynome mit Koeffizienten a_j approximiert werden

$$y_i(x_i) \approx \sum_{j=0}^{m-1} a_j p_j(x_i) \; . \tag{10.19}$$

Eine Schätzung $\widehat{a}_j$ für die Koeffizienten a_j erhält man durch die Methode der kleinsten Quadrate, indem man die Summe der gewichteten Abweichungsquadrate zwischen y_i und $\sum a_j p_j(x_i)$

$$S = \sum_{i=1}^{n} w_i \left(y_i - \sum_{j=0}^{m-1} a_j p_j(x_i) \right)^2 \tag{10.20}$$

minimiert. Der Formalismus dieser Lösung ist bereits in Kapitel 7.4 behandelt worden und führte auf die Gleichung (7.54)

$$\boxed{\begin{array}{rcl} \widehat{\boldsymbol{a}} & = & (\boldsymbol{A}^T \boldsymbol{W} \boldsymbol{A})^{-1} \boldsymbol{A}^T \boldsymbol{W} \boldsymbol{y} \\ \boldsymbol{V}[\widehat{\boldsymbol{a}}] & = & (\boldsymbol{A}^T \boldsymbol{W} \boldsymbol{A})^{-1} \end{array}}$$

für den Vektor der Parameter $\widehat{\boldsymbol{a}}$ und die Kovarianzmatrix $\boldsymbol{V}[\boldsymbol{a}]$ der Parameter. Das Element der Matrix $\left(\boldsymbol{A}^T \boldsymbol{W} \boldsymbol{A}\right)$ mit dem Indexpaar j, k ist mit dem Ansatz Gleichung (10.19) gegeben durch

$$\left(\boldsymbol{A}^T \boldsymbol{W} \boldsymbol{A}\right)_{jk} = \sum_{i=1}^{n} w_i \, p_j(x_i) p_k(x_i) \; ,$$

und der Vergleich mit Gleichung (10.18) zeigt, daß diese Matrix und damit die Kovarianzmatrix der Parameter die Einheitsmatrix ist. Die Parameter sind unkorreliert und haben die Varianz 1 als Folge der Definitionsgleichung der orthonormalen Polynome. Die Schätzwerte der Parameter werden durch die Summe

$$\widehat{a}_j = \sum_{i=1}^{n} w_i \, p_j(x_i) \, y_i \tag{10.21}$$

bestimmt.

Die Eigenschaften der orthogonalen Polynome führt zu einer einfachen Methode zur Bestimmung des Grads des Polynoms, der zu einer Beschreibung der Daten notwendig ist. Wenn ein Koeffizient a_j den wahren Wert null hat, dann folgt sein Schätzwert einer Normalverteilung mit Mittelwert null und Standardabweichung eins. Die Signifikanz des Koeffizienten kann so leicht geprüft werden; zum Beispiel ist bei einer Signifikanz von 95% das Quadrat des Schätzwerts von a_j kleiner als 3.84. Wenn alle a_j für $j > m_0$ mit null verträglich sind, dann können die Daten (y_i, x_i) durch die ersten m_0 Terme der Gleichung (10.19) dargestellt werden.

Konstruktion orthonormaler Polynome. Die ersten zwei orthonormalen Polynome $p_0(x)$ und $p_1(x)$ sind

$$p_0(x) = b_{00} \tag{10.22}$$

$$p_1(x) = b_{10} + b_{11} x = b_{11} \left(x - \langle x \rangle \right) \tag{10.23}$$

mit dem gewichteten Mittelwert der x_i

$$\langle x \rangle = \sum_{i=1}^{n} w_i x_i / \sum_{i=1}^{n} w_i \, .$$

Die Koeffizienten b_{00} und b_{11} sind gegeben durch Summen gemäß

$$b_{00} = \left(\sum_{i=1}^{n} w_i \right)^{-1/2} \qquad b_{11} = \left(\sum_{i=1}^{n} w_i \left(x_i - \langle x \rangle \right)^2 \right)^{-1/2} ,$$

und der Koeffizient b_{10} ist gegeben durch $b_{10} = -\langle x \rangle b_{11}$. Man überzeugt sich leicht, daß diese Polynome orthonormal sind.

Die höheren orthonormalen Polynome erhält man aus der Rekursionsformel[1]

$$\gamma p_j(x) = (x - \alpha)\, p_{j-1}(x) - \beta p_{j-2}(x) , \tag{10.24}$$

wobei die Konstanten α, β und γ noch bestimmt werden müssen. Für die Berechnung von α wird Gleichung (10.24) mit $w_i p_{j-1}(x_i)$ multipliziert und über die x_i summiert. Nach Definition ist die linke Seite null, weil $p_j(x)$ und $p_{j-1}(x)$ zueinander orthogonal sind:

$$0 = \sum_{i=1}^{n} w_i x_i p_{j-1}^2(x_i) - \alpha \sum_{i=1}^{n} w_i p_{j-1}^2(x_i) - \beta \sum_{i=1}^{n} w_i p_{j-1}(x_i) p_{j-2}(x_i) \ . \tag{10.25}$$

Die zweite Summe ist gleich 1, weil die Funktion $p_{j-1}(x)$ normiert ist, und die dritte Summe ist null, weil $p_{j-1}(x)$ und $p_{j-2}(x)$ orthogonal sind. Daher folgt

$$\alpha = \sum_{i=1}^{n} w_i x_i p_{j-1}^2(x_i) \ . \tag{10.26}$$

Ähnlich geht man bei der Bestimmung von β vor. Man multipliziert Gleichung (10.24) mit $w_i p_{j-2}(x_i)$ und bildet die Summen über die x_i

$$0 = \sum_{i=1}^{n} w_i x_i p_{j-1}(x_i) p_{j-2}(x_i) - \alpha \sum_{i=1}^{n} w_i p_{j-1}(x_i) p_{j-2}(x_i) - \beta \sum_{i=1}^{n} w_i p_{j-2}^2(x_i) . \tag{10.27}$$

Die zweite Summe ist null, weil $p_{j-1}(x)$ und $p_{j-2}(x)$ orthogonal sind, die dritte Summe ist 1, weil $p_{j-2}(x)$ normiert ist, und β ist

$$\beta = \sum_{i=1}^{n} w_i x_i p_{j-1}(x_i) p_{j-2}(x_i) . \tag{10.28}$$

Endlich folgt γ aus der Normierungsbedingung (10.18) von $p_j(x)$

$$\sum_{i=1}^{n} w_i p_j^2(x_i) = 1 , \tag{10.29}$$

und man erhält

$$\gamma^2 = \sum_{i=1}^{n} w_i \left[(x_i - \alpha)\, p_{j-1}\left(x_i\right) - \beta p_{j-2}\left(x_i\right) \right]^2 . \tag{10.30}$$

[1] Beweis durch vollständige Induktion: Für $j = 2$ gilt Gleichung (10.24) offensichtlich. Angenommen, diese Gleichung sei für alle $j = 2, 3, \ldots k$ bewiesen. Dann zeigt man, daß für $k+1$ gilt: Die Funktion p_{k+1} läßt sich immer darstellen durch $\gamma p_{k+1} = (x - \alpha) p_k + \sum_{j=0}^{k-1} c_j p_j(x)$. Durch Multiplikation mit $w_i p_k(x_i)$ und Summieren über i zeigt man, daß α identisch mit Gleichung (10.26) ist. Durch Multiplizieren mit $w_i p_l(x_i)$ und Summieren über i und Benutzung der schon bewiesenen Gleichungen (10.24) für $l = k-1, k-2, \ldots 0$ zeigt man, daß die Koeffizienten c_j alle null sind bis auf $c_{k-1} = -\beta$.

i	x_i	y_i	σ_i	i	x_i	y_i	σ_i
1	1.0	5.0935	0.5	11	11.0	-0.7703	0.5
2	2.0	2.1777	0.5	12	12.0	-0.3055	0.5
3	3.0	0.2089	0.5	13	13.0	1.6817	0.5
4	4.0	-2.3949	0.5	14	14.0	1.8728	0.5
5	5.0	-2.4457	0.5	15	15.0	3.6586	0.5
6	6.0	-3.0430	0.5	16	16.0	3.2353	0.5
7	7.0	-2.2731	0.5	17	17.0	4.2520	0.5
8	8.0	-2.0706	0.5	18	18.0	5.2550	0.5
9	9.0	–1.6231	0.5	19	19.0	3.8766	0.5
10	10.0	–2.5605	0.5	20	20.0	4.2890	0.5

Tabelle 10.1: Daten für die Anpassung durch orthogonale Polynome.

Diese Berechnung der Größen α, β und γ wird für alle Indizes j wiederholt.

Die Rekursionsformel kann benutzt werden, um die Funktionswerte an einem beliebigen Wert x zu berechnen. Man kann natürlich auch die orthogonalen Polynome der Gleichung (10.17) in normale Polynome umrechnen.

Das Unterprogramm `ORTHFT` *im Anhang bestimmt die Koeffizienten einer orthogonalen Parametrisierung von Daten gemäß* (10.1).

Interpolierte Werte an beliebigen Werten der Abszisse `X` *können anschließend mit der Funktion* `ORTHFE` *berechnet werden. Zusätzlich zum interpolierten Funktionswert* `Y` *wird auch die Unsicherheit* `DY` *nach Fehlerfortpflanzung berechnet.*

Beispiel 10.2 Anpassung von orthonormalen Polynomen an Daten (1).

Die Daten (y_i, x_i), $i = 1, 2, \ldots 20$ in der Tabelle 10.1 sollten sich durch Polynome niedrigen Grades parametrisieren lassen. Alle Daten haben dieselbe Standardabweichung $\sigma = 0.5$ und daher Gewicht $w = 1/0.5^2 = 4$.

Es werden orthonormale Polynome bis zum 10. Grad angepaßt, das Resultat ist in der Tabelle 10.2 zu sehen. Die erste Spalte der Tabelle gibt den Grad j des Polynoms, die Spalte a_j gibt den Wert des zugehörigen Parameters an. Die statistisch unabhängigen Parameter haben Varianz 1. Die 95% Konfidenzgrenze für den Betrag ist $\sqrt{3.84} = 1.96$. Der höchste signifikante Koeffizient ist demnach a_3; man benötigt Polynome bis zum dritten Grad. In diesem Beispiel ist auch der Koeffizient a_{10} oberhalb der Signifikanzgrenze, wogegen die dazwischenliegenden Koeffizienten alle mit null verträglich sind. In diesem Fall kann man den verhältnismäßig großen Wert von a_{10} als statistische Fluktuation ansehen. Die nächste Spalte gibt die gewichtete Summe der Quadrate der Abweichungen nach Gleichung (10.20) an. Geht man vom Index j zum nächsten Index $j + 1$, so wird die Quadratsumme um a_j^2 vermindert. Dies folgt aus Gleichung (10.20) und Gleichung (10.18). Die Spalte σ_{est} gibt den Faktor, um den die Standardabweichung der Daten vergrößert werden müßte, um das Verhältnis χ^2/Freiheitsgrad zu 1 zu erhalten.

Ein alternativer Test auf insignifikante Terme ist der F-Test. Die Spalte $F(1, n)$ zeigt den F-Testwert für einen und n Freiheitsgrade. Sie müssen mit den Werten in einer F-Tabelle verglichen werden. Für eine Konfidenzgrenze von 95% sollte der Wert von $F(1, n)$ kleiner sein als der Tabellenwert für $F_{0.95}$. Der Test beruht auf Quotienten und ist deshalb unabhängig vom ursprünglich für die Gewichtsberechnung benutzten Wert von σ. Die Schlußfolgerung in diesem Beispiel ist dieselbe wie beim χ^2-Test.

j	a_j	Quadratsumme	σ_{est}	$F(1,n)$	n	$F_{0.95}$
0	8.10	655.7	5.87	1.90	19	4.39
1	13.65	469.4	5.11	7.15	18	4.43
2	16.48	197.9	3.41	23.32	17	4.46
3	-13.23	22.9	1.20	122.4	16	4.50
4	0.64	22.5	1.22	0.27	15	4.54
5	-1.23	21.0	1.22	1.01	14	4.61
6	0.90	20.1	1.24	0.52	13	4.68
7	1.49	17.9	1.22	1.48	12	4.75
8	-1.25	16.4	1.22	1.05	11	4.86
9	0.64	16.0	1.26	0.26	10	4.96
10	2.50	9.7	1.04	5.76	9	5.12

Tabelle 10.2: Ergebnis der Anpassung von orthogonalen Polynomen an die Daten.

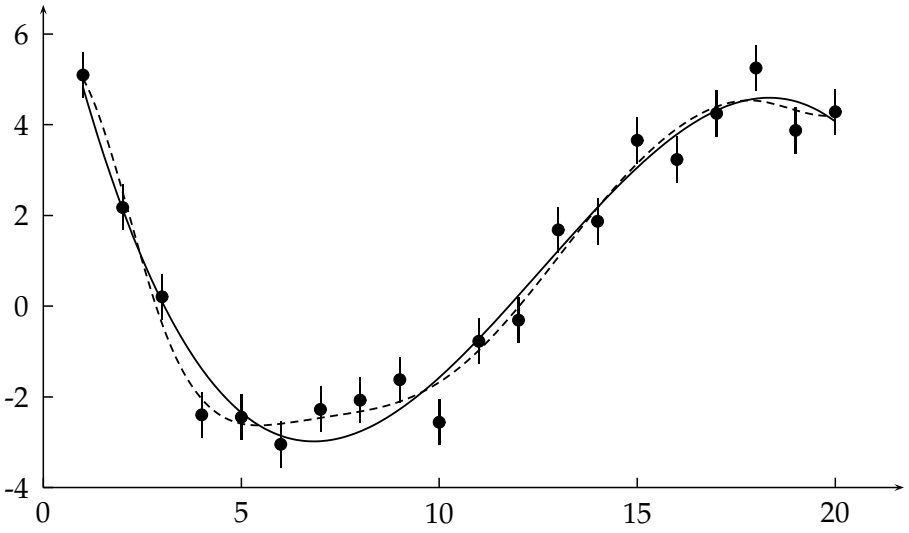


Abbildung 10.5: Anpassung eines Polynoms an Meßdaten. Die durchgezogene Kurve ist ein Polynom dritten Grades, die gestrichelte Kurve ist ein Polynom zehnten Grades.

Die Abbildung 10.5 zeigt die Datenpunkte und die angepaßten Polynome vom Grad drei und zehn. Das Polynom zehnten Grades folgt offenbar den zufälligen statistischen Schwankungen der Datenpunkte, wie es auch die mit null verträglichen Parameterwerte der höheren Grade zeigen. Das Polynom dritten Grades gibt eine befriedigende Beschreibung der Daten.

Beispiel 10.3 Anpassung von orthonormalen Polynome an Daten (2).

An die gleichen Daten (y_i, x_i), $i = 1, 2, \ldots 20$ der Abbildung 10.4, an die dort eine Spline-Funktion angepaßt wurde, werden orthogonale Polynome angepaßt; das Ergebnis ist allerdings nicht befriedigend. Bei kleinem Polynomgrad kann das Maximum nicht gut beschrieben werden, und bei hohem Polynomgrad gibt es auch im nahezu konstanten Bereich eine Wellenstruktur, wie die Abbildung 10.6 für ein Polynom 20. Grades zeigt. In dieser Abbildung ist mit punktierten Kurven auch das durch Fehlerfortpflanzung berechnete Fehlerband entsprechend

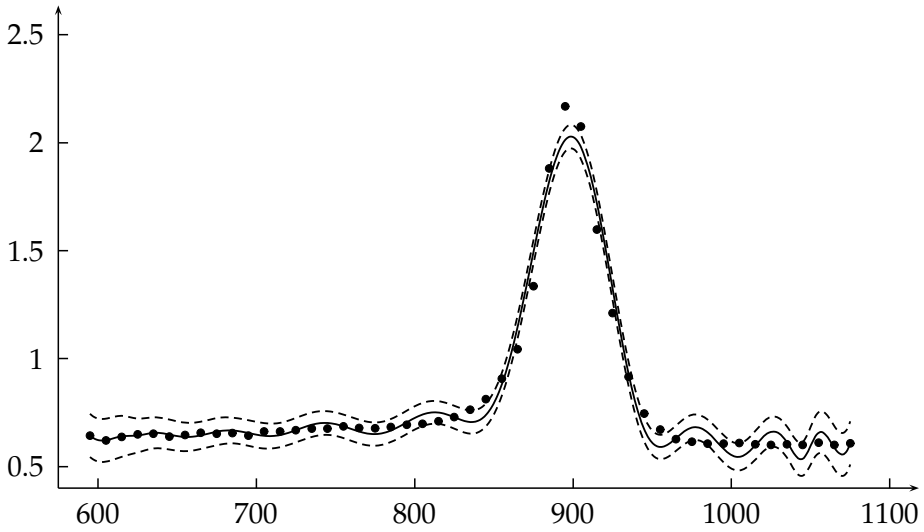


Abbildung 10.6: Anpassung eines Polynoms an Meßdaten. Die durchgezogene Kurve ist ein Polynom 20. Grades. Die punktierten Kurven stellen das Fehlerband entsprechend ± 1 Standardabweichung dar.

± 1 Standardabweichung gezeigt.

10.4 Fourierreihen

Fourierentwicklung. Ein wichtiges Beispiel eines vollständigen orthogonalen Funktionensystems sind die Funktionen 1, $\cos x$, $\sin x$, $\cos 2x$, $\sin 2x$, ..., die im Intervall $0 \leq x \leq 2\pi$ zueinander orthogonal sind. Eine Funktion der Periode 2π kann in eine Fourierreihe entwickelt werden,

$$f(x) = \frac{a_0}{2} + \sum_{\nu=1}^{\infty} (a_\nu \cos \nu x + b_\nu \sin \nu x) \; , \tag{10.31}$$

wobei die Fourierkoeffizienten a_ν und b_ν gegeben sind durch

$$\begin{aligned} a_\nu &= \frac{1}{\pi} \int_0^{2\pi} f(x) \cos \nu x \, \mathrm{d}x \qquad\qquad b_\nu = \frac{1}{\pi} \int_0^{2\pi} f(x) \sin \nu x \, \mathrm{d}x \\ a_0 &= \frac{1}{\pi} \int_0^{2\pi} f(x) \, \mathrm{d}x \; . \end{aligned} \tag{10.32}$$

Für die Darstellung glatter Funktionen reicht eine kleine Zahl von Termen aus.

Das asymptotische Verhalten der Fourierkoeffizienten hängt vom Grad der Glattheit oder Differenzierbarkeit der Funktion $f(x)$ ab. Es gilt das Verhalten

$$c_\nu = \sqrt{a_\nu^2 + b_\nu^2} = O\left(\frac{1}{|\nu|^{r+1}}\right) \tag{10.33}$$

für eine Funktion der Periode 2π mit einer stetigen r-ten Ableitung. Eine andere interessante Eigenschaft der Fourierentwicklung besteht darin, daß jeder Term der Entwicklung einen *unabhängigen* Beitrag zur totalen Krümmung liefert

$$\int_0^{2\pi} \left[f''(x)\right]^2 \, \mathrm{d}x = \pi \sum_{\nu=1}^{\infty} \nu^4 (a_\nu^2 + b_\nu^2). \tag{10.34}$$

Gleichungen (10.33) und (10.34) zeigen, daß für eine glatte Funktion $f(x)$ die höheren Fourierkoeffizienten, die einen großen Beitrag zur totalen Krümmung machen, rasch mit steigendem Index ν kleiner werden. Im Beispiel einer Funktion mit stetiger dritter Ableitung ist der Absolutwert der Koeffizienten für große ν begrenzt durch einen Term proportional $1/|\nu|^4$.

Diskrete Fouriertransformation. Die Fourierentwicklung einer Funktion kann aus einer Reihe von Datenpunkten y_i berechnet werden, wobei die Datenpunkte an n äquidistanten x-Werten $x_i = 2\pi \cdot i/n$ vorgegeben sein sollen. Dann erhält man

$$a_\nu = \frac{2}{n}\sum_{i=1}^{n} y_i \cdot \cos \nu x_i \qquad b_\nu = \frac{2}{n}\sum_{i=1}^{n} y_i \cdot \sin \nu x_i \tag{10.35}$$

$$a_0 = \frac{2}{n}\sum_{i=1}^{n} y_i \,, \tag{10.36}$$

wobei ν von 1 bis $n/2$ geht. Man bildet n Datenpunkte auf n unabhängige Koeffizienten ab.

Eine Methode zur Glättung von Daten (y_i, x_i) mit statistischen Fluktuationen besteht in einer Näherung durch eine Fourierreihe mit Hilfe der Gleichungen (10.35). Die höheren Fourierkoeffizienten mit kleinen Werten sind hauptsächlich auf statistische Fluktuationen zurückzuführen. Man erhält eine geglättete Näherung durch Weglassen dieser Terme aus Gleichung (10.31) [43].

Unter Benutzung der Eigenschaften der trigonometrischen Funktionen sind spezielle schnelle Algorithmen zur Berechnung der Fourierkoeffizienten entwickelt worden und wurden in der Tat seit dem 19. Jahrhundert mehrere Male wiederentdeckt. Die diskrete Fouriertransformation kann in einer Rechenzeit proportional zu $n \lg_2 n$ mit Hilfe der sogenannten *Fast Fourier Transform* (FFT) durchgeführt werden. Dies hat den Weg zu zahlreichen Anwendungen in der Praxis geebnet. Algorithmen für FFT findet man bei [6, 58].

11 Entfaltung

11.1 Problemstellung

Ein häufig auftretendes Problem ist die Bestimmung der Verteilung einer physikalischen Größe x. Ein Satz von Meßgrößen x kann im statistischen Sinn als eine Stichprobe aus einer Wahrscheinlichkeitsdichte $f(x)$ aufgefaßt werden. Die tatsächlich gemessene Verteilung, die oft in Form eines Histogramms vorliegt, unterscheidet sich von der wahren Verteilung wegen der statistischen Fehler und anderer Unvollkommenheiten des Meßprozesses, die es verhindern, daß der wahre Wert von x gemessen wird. Als Beispiel seien zwei wichtige Effekte genannt, *begrenzte Akzeptanz* und *begrenzte Meßgenauigkeit*.

Begrenzte Akzeptanz bedeutet, daß die Wahrscheinlichkeit, einen bestimmten Wert x zu beobachten, kleiner als eins ist. Begrenzte Meßgenauigkeit wird auf Grund der Meßfehler zu einer vom wahren Wert abweichenden Bestimmung der Größe x führen, und durch einen Bias wird auch im Mittel nicht der wahre Wert von x wiedergegeben. Bereits die Einteilung in die Intervalle eines Histogramms erzeugt solche Ungenauigkeiten.

Formal kann dies folgendermaßen ausgedrückt werden: Anstelle der gesuchten Variablen x wird tatsächlich die Variable y gemessen und eine Wahrscheinlichkeitsdichte (Verteilung) $g(y)$ dieser Größe bestimmt. Die gesuchte Verteilung $f(x)$ der physikalischen Variablen x, definiert im Bereich $a \leq x \leq b$, und die gemessene Verteilung $g(y)$ der gemessenen Größe y sind über ein Faltungsintegral

$$g(y) = \int_a^b A(y,x) f(x)\,\mathrm{d}x \tag{11.1}$$

verknüpft. Diese Gleichung heißt *Fredholmsche Integralgleichung der ersten Art.* Der Kern $A(y,x)$ der Integralgleichung bestimmt die Wahrscheinlichkeit, durch eine Messung den Wert y zu erhalten, wenn der wahre Wert der Variablen x ist. In der Praxis ist $A(y,x)$ oft durch eine Monte Carlo-Rechnung bestimmt, die den Meßvorgang $x \to y$ realistisch simuliert. Man sollte die Monte Carlo-Simulation unter allen Umständen überprüfen, zum Beispiel durch Vergleich mit passend angelegten Testmessungen. Weitere Komplikationen sind statistische Fehler $\varepsilon(y)$ und ein Untergrundbeitrag zur gemessenen Verteilung, der von Prozessen kommt, die nichts mit dem gesuchten Prozeß zu tun haben. Man erhält dann für die gemessene Verteilung $\widehat{g}(y)$

$$\widehat{g}(y) = \int_a^b A(y,x) f(x)\,\mathrm{d}x + b(y) + \varepsilon(y)\,. \tag{11.2}$$

Oft ist der Umfang der zur Verfügung stehenden Stichprobe nicht sehr groß, und dann spielen die statistischen Fehler eine große Rolle. Wenn bei präzisen Messungen der Einfluß der Meßgenauigkeit zurücktritt und der Haupteffekt die begrenzte Akzeptanz ist, dann ist die Korrektur nicht sehr schwierig (siehe Kapitel 11.2). Wenn große Meßfehler mit starken Verzerrungen vorhanden sind, gibt es zwei Vorgehensweisen.

Bei Vorliegen einer spezifischen Voraussage $f_{\mathrm{th}}(x)$ für $f(x)$ kann die daraus erwartete gemessene Verteilung mit Hilfe von Gleichung (11.1) berechnet werden. Wenn diese Verteilung $g_{\mathrm{th}}(y)$ mit der gemessenen Verteilung $\widehat{g}(y)$ innerhalb der statistischen Fehler übereinstimmt, dann ist die theoretische Verteilung *kompatibel* mit $f_{\mathrm{th}}(x)$.

Wenn aber eine solche Voraussage nicht mit den Messungen übereinstimmt oder eine solche gar nicht vorliegt, dann ist es notwendig, die gesuchte Verteilung $f(x)$ aus der gemessenen $\widehat{g}(y)$ zu rekonstruieren. Die Rekonstruktion von $f(x)$ aus $\widehat{g}(y)$ heißt *Entfaltung.* Es handelt sich dabei um ein statistisches und im mathematischen Sinn schlecht konditioniertes Problem, welches

wild oszillierende sinnlose Lösungen haben kann [65]. Zu diesem *inversen* Problem gibt es eine ausgedehnte Literatur, zum Beispiel [29, 30, 63, 80, 49, 40].

Diskrete variable. Wenn x und y diskrete Variable sind, muß das Integral in Gleichung (11.2) durch eine Summe ersetzt werden. Man setzt nun voraus, daß die diskreten Variablen x und y (notfalls nach einer Transformation) ganzzahlig im Bereich $1 \ldots n$ bzw. $1 \ldots m$ sind. Statt der Funktionen $f(x)$ und $g(y)$ hat man nun Vektoren mit den Elementen f_j, $j = 1, 2, \ldots n$ und g_i, $i = 1, 2, \ldots m$, und die Faltungsgleichung lautet dann

$$\widehat{g}_i = \sum_{j=1}^{n} A_{ij} f_j + b_i + \varepsilon_i \qquad \text{oder kurz} \qquad \widehat{\boldsymbol{g}} = \boldsymbol{A}\boldsymbol{f} + \boldsymbol{b} + \boldsymbol{\varepsilon} \; ; \tag{11.3}$$

dabei ist $\boldsymbol{f}$ ein n-Vektor, $\widehat{\boldsymbol{g}}$, $\boldsymbol{b}$ und $\boldsymbol{\varepsilon}$ sind m-Vektoren und $\boldsymbol{A}$ ist eine $m \times n$ Matrix, genannt *Responsematrix*. Die Gleichung kann als Diskretisierung der Integralgleichung (11.2) aufgefaßt werden.

Kontinuierliche Variable. Jede numerische Lösung der Integralgleichung erfordert normalerweise eine diskrete Repräsentation. Die einfachste derartige Repräsentation ist eine durch Histogramme $\boldsymbol{f}$ und $\widehat{\boldsymbol{g}}$. Im Kapitel 11.5 wird eine Diskretisierung durch Basisfunktionen behandelt, durch die Nachteile der Histogramme wie Unstetigkeiten and den Intervallgrenzen vermieden werden. Im Normalfall sollte $m > n$ sein und in vielen Fällen empfiehlt sich die Wahl $m \approx 2n$.

Fall $n = m$ und Lösung durch Matrix-Inversion. Wenn $n = m$ ist, dann ist $\boldsymbol{A}$ eine quadratische Matrix, und man erhält mit Standardmethoden der Matrixalgebra die Lösung

$$\widehat{\boldsymbol{f}} = \boldsymbol{A}^{-1} \, (\widehat{\boldsymbol{g}} - \boldsymbol{b}) \; , \tag{11.4}$$

wobei $\boldsymbol{A}^{-1}$ die zu $\boldsymbol{A}$ inverse Matrix ist und $\boldsymbol{b}$ als bekannt angenommen wird. Im folgenden wird zur Vereinfachung vorübergehend $\boldsymbol{b} = 0$ angenommen, d.h. kein Untergrundbeitrag.

Der Erwartungswert $E[\widehat{\boldsymbol{f}}]$ der Schätzung $\widehat{\boldsymbol{f}}$ ist gleich dem wahren $\boldsymbol{f}$, falls die gemessene Verteilung $\widehat{\boldsymbol{g}}$ unverzerrt ist, d.h. falls $E[\widehat{\boldsymbol{g}}] = \boldsymbol{A}\boldsymbol{f}$ ist. Dann gilt

$$E[\widehat{\boldsymbol{f}}] = E[\boldsymbol{A}^{-1}\widehat{\boldsymbol{g}}] = \boldsymbol{A}^{-1} \, E[\widehat{\boldsymbol{g}}] = \boldsymbol{A}^{-1}\boldsymbol{A} \, \boldsymbol{f} = \boldsymbol{f} \; .$$

Eine Schätzung mit dieser Eigenschaft heißt *konsistent*. Standard-Fehlerfortpflanzung ergibt

$$\boldsymbol{V}[\widehat{\boldsymbol{f}}] = \boldsymbol{A}^{-1} \, \boldsymbol{V}[\widehat{\boldsymbol{g}}] \, (\boldsymbol{A}^{-1})^T$$

für die Kovarianzmatrix $\boldsymbol{V}[\widehat{\boldsymbol{f}}]$, berechnet aus der Kovarianzmatrix $\boldsymbol{V}[\widehat{\boldsymbol{g}}]$ der Meßdaten. Leider gibt dieser verhältnismäßig einfache und naheliegende Weg in der Praxis meist enttäuschende Resultate mit oszillierenden Lösungen.

Regularisierungsmethoden. Um akzeptable Resultate zu erhalten, muß man zu *Regularisierungsmethoden* [74, 59] greifen, die die Oszillationen der Lösung unterdrücken. Anschaulich bedeutet Regularisierung, daß man bestimmte Annahmen über den glatten Verlauf der wahren Lösung macht. Da dies eine Verzerrung einführen kann, muß die Regularisierung kontrolliert werden, um ihren Einfluß klein im Vergleich zu den statistischen Fehlern zu halten. Dies führt zu einer Beschränkung der Zahl der Datenpunkte nach der Entfaltung, bedingt durch die Meßgenauigkeit und die verfügbare Statistik. Eine Messung mit beschränkter Genauigkeit bedeutet immer einen Verlust an statistischer Genauigkeit.

In dem im folgenden dargestellten Algorithmus zur Entfaltung mit Regularisierung wird die totale Krümmung der Lösung $f(x)$ nach Gleichung (10.9) benutzt, die durch das Quadrat der zweiten Ableitungen definiert ist. Die Darstellung folgt [8]; die Diskretisierung erfolgt mit

(kontinuierlichen) Basisfunktionen und die Entfaltung entspricht einem Maximum-Likelihood-Fit mit Poisson-Verteilung der gemessenen Daten (Anzahlen pro Bin) und Diagonalisierung der symmetrischen Matrix der zweiten Ableitungen.

Im allgemeinen erfolgt die Diskretisierung durch Histogramm, und die Poisson-Verteilung der Daten wird durch die Normalverteilung angenaehert. Als Orthogonalisierungsverfahren kann dann die Singulärwert-Zerlegung benutzt werden[33]. Weitere Referenzen für Entfaltungsmethoden sind [48, 67, 84].

11.2 Akzeptanzkorrekturen

Die *Akzeptanz* P ist die Wahrscheinlichkeit, daß bei Vorliegen eines wahren Wertes x die zugehörige Variable y tatsächlich gemessen werden kann. Wenn die Effekte der endlichen Meßgenauigkeit vernachlässigt werden können, dann wird bei $n = m$ die Responsematrix diagonal und hat Diagonalelemente $A_{jj} = P_j$. Dies ist der einfachste Fall eines Entfaltungsproblems. Man erhält dann für die gemessene Verteilung

$$g_j = P_j f_j \qquad j = 1, 2, \ldots n \; .$$

Manchmal wird eine eingeschränkte Akzeptanz einfach durch die Geometrie des Problems, zum Beispiel durch Hindernisse, bedingt. In komplizierteren Fällen hilft die Monte Carlo-Methode, die es gestattet, alle Effekte zu berücksichtigen. Die Akzeptanz P_j im Intervall j und ihren statistischen Fehler erhält man durch

$$P_j = \frac{N_{j,\mathrm{rek}}}{N_{j,\mathrm{gen}}} \qquad \sigma_{j,\mathrm{rel}} = \frac{\Delta P_j}{P_j} = \sqrt{\frac{(1-P_j)}{P_j}} \cdot \sqrt{\frac{1}{N_{j,\mathrm{gen}}}} \, . \tag{11.5}$$

Der relative Fehler $\sigma_{j,\mathrm{rel}}$ ist nach der Binomialverteilung berechnet. Dabei ist $N_{j,\mathrm{gen}}$ die Zahl der Fälle, die in der wahren Variablen x im Intervall j generiert wurden, und $N_{j,\mathrm{rek}}$ die Zahl der Fälle, die in der Variablen y im selben Intervall j gemessen wurden.

Für kleine Werte der Akzeptanz P kann der relative Fehler groß werden; in diesem Fall muß man sehr große Anzahlen $N_{j,\mathrm{gen}}$ erzeugen. Die korrigierten Werte der gesuchten Verteilung $\widehat{f}_j$ sind

$$\widehat{f}_j = \frac{g_j}{P_j} \qquad \text{mit Fehler} \qquad \sigma(\widehat{f}_j) = \frac{\sqrt{g_j}}{P_j} \cdot \sqrt{1 + g_j \, \sigma^2_{j,\mathrm{rel}}} \quad .$$

Man sollte die Monte Carlo-Rechnung so anlegen, daß die statistischen Fehler von P klein gegen diejenigen der Messung sind. In Bereichen sehr kleiner Akzeptanz P kann es Schwierigkeiten geben. Es ist unter Umständen besser, derartige Bereiche auszuschließen.

11.3 Entfaltung in zwei Intervallen

Im Fall von nur zwei Intervallen kann man einige der typischen Probleme gut studieren. Dieser Fall entspricht auch dem Fall einer Klassifikation von Ereignissen in zwei Klassen mit bestimmter Wahrscheinlichkeit einer *Fehlklassifikation*. Die Matrixgleichung ist $\boldsymbol{g} = \boldsymbol{A}\boldsymbol{f}$, mit einer Responsematrix $\boldsymbol{A}$ gemäß

$$\boldsymbol{g} = \begin{pmatrix} g_1 \\ g_2 \end{pmatrix} \qquad \boldsymbol{A} = \begin{pmatrix} 1-\varepsilon_{11} & \varepsilon_{12} \\ \varepsilon_{21} & 1-\varepsilon_{22} \end{pmatrix} \qquad \boldsymbol{f} = \begin{pmatrix} f_1 \\ f_2 \end{pmatrix} . \tag{11.6}$$

Der Vektor $\boldsymbol{g}$ ist der Vektor, der die Zahl der gemessenen Einträge in die beiden Intervalle angibt; dies sind ganze Zahlen. Die Kovarianzmatrix dieser gemessenen Zahlen, die der Poisson-Statistik folgen und unabhängig sein sollen, ist die Diagonalmatrix

$$\boldsymbol{V}[\boldsymbol{g}] = \begin{pmatrix} g_1 & 0 \\ 0 & g_2 \end{pmatrix} . \tag{11.7}$$

Das Resultat der Entfaltung ist $\boldsymbol{f} = \boldsymbol{B}\,\boldsymbol{g}$ mit $\boldsymbol{B} = \boldsymbol{A}^{-1}$. Die Kovarianzmatrix von $\boldsymbol{f}$ ist nach der Fehlerfortpflanzung

$$\boldsymbol{V}[\boldsymbol{f}] = \boldsymbol{B}\,\boldsymbol{V}[\boldsymbol{g}]\,\boldsymbol{B}^T , \tag{11.8}$$

wobei die Matrix $\boldsymbol{B}^T$ die transponierte Matrix von $\boldsymbol{B}$ ist.

Man betrachtet nun den Spezialfall, daß alle ε-Faktoren in $\boldsymbol{A}$ gleich sind: $\varepsilon = \varepsilon_{11} = \varepsilon_{12} = \varepsilon_{21} = \varepsilon_{22}$; damit sind die Matrizen $\boldsymbol{A}$ und $\boldsymbol{B}$ symmetrisch. Der Faktor ε ist die Wahrscheinlichkeit, daß eine Messung in das falsche Intervall führt. Die Akzeptanz ist 100%, es gehen keine Messungen verloren. Dann ist die inverse Matrix

$$\boldsymbol{B} = \frac{1}{1-2\varepsilon} \begin{pmatrix} 1-\varepsilon & -\varepsilon \\ -\varepsilon & 1-\varepsilon \end{pmatrix} ,$$

und die korrigierten Zahlen sind

$$f_1 = \frac{1}{1-2\varepsilon} \left[(1-\varepsilon)\ g_1 - \varepsilon\, g_2\right] \tag{11.9}$$

$$f_2 = \frac{1}{1-2\varepsilon} \left[-\varepsilon\, g_1 + (1-\varepsilon)\ g_2\right] \tag{11.10}$$

mit der Kovarianzmatrix

$$\boldsymbol{V}[\boldsymbol{f}] = \frac{1}{(1-2\varepsilon)^2} \begin{pmatrix} (1-\varepsilon)^2\, g_1 + \varepsilon^2\, g_2 & -\varepsilon(1-\varepsilon)\,(g_1+g_2) \\ -\varepsilon(1-\varepsilon)\,(g_1+g_2) & (1-\varepsilon)^2\, g_2 + \varepsilon^2\, g_1 \end{pmatrix} . \tag{11.11}$$

Das Nebendiagonalelement ist negativ: die korrigierten Zahlen sind *negativ* korreliert. Im Grenzfall $\varepsilon \to 0.5$ (dies bedeutet *keine* Messung) wird die Kovarianzmatrix singulär.

Beispiel 11.1 Entfaltung in zwei Intervallen.

Für die beobachteten Zahlen $g_1 = 100$ und $g_2 = 81$ ist der Vektor $\boldsymbol{g}$ mit Fehlern

$$\begin{array}{rcrr} g_1 &=& 100 & \pm 10 \\ g_2 &=& 81 & \pm 9 \end{array} \qquad \boldsymbol{V}[\boldsymbol{g}] = \begin{pmatrix} 100 & 0 \\ 0 & 81 \end{pmatrix} .$$

Die Ergebnisse der Entfaltung für die Vermischungswahrscheinlichkeiten $\varepsilon = 0.1$ und $\varepsilon = 0.4$ sind

$$\begin{array}{lllll} \varepsilon = 0.1 & : & f_1 = 102.4 \pm 11.3 & f_2 = 78.6 \pm 10.2 & \rho_{12} = -0.22 \\ \varepsilon = 0.4 & : & f_1 = 138.0 \pm 35.0 & f_2 = 43.0 \pm 33.6 & \rho_{12} = -0.92 \end{array}$$

(ρ_{12} ist der Korrelationskoeffizient). Bei $\varepsilon = 0.1$ ist die Korrektur der Zahlen und die Korrelation zwischen f_1 und f_2 noch klein. Dagegen führt bei einem Wert von $\varepsilon = 0.4$ die starke negative Korrelation bereits zu erheblich vergrößerten Korrekturen und Fehlern.

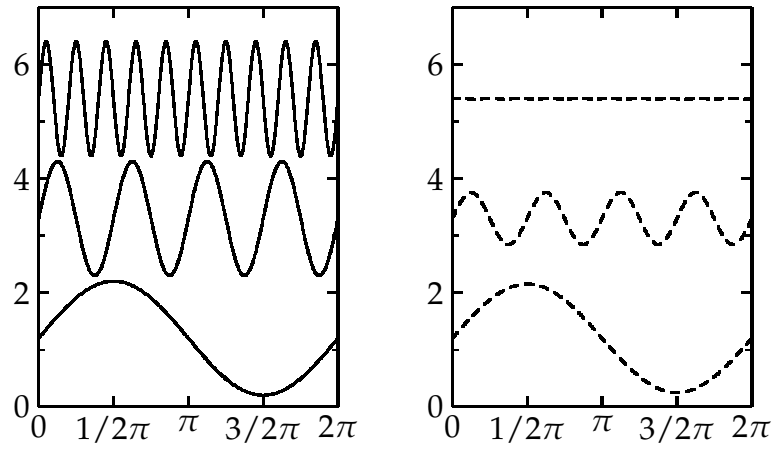


Abbildung 11.1: Effekt der Faltung mit einer Gauß-Verteilung auf die $\sin \nu x$-Terme für $\nu = 1, 4, 10$. Links sind die Terme vor der Faltung und rechts nach der Faltung gezeigt.

11.4 Entfaltung periodischer Verteilungen

In diesem Spezialfall möge die Verteilung periodisch sein mit Periode 2π. Sie ist gegeben durch $f(x)$ im Bereich $0 \leq x \leq 2\pi$. Die Funktion $f(x)$ kann als Fourierreihe

$$f(x) = \frac{a_0}{2} + \sum_{\nu=1}^{\infty} (a_\nu \cos \nu x + b_\nu \sin \nu x) \tag{11.12}$$

geschrieben werden (siehe Gleichung (10.31)). Die Summe ist endlich, wenn die Funktion $f(x)$ durch eine endliche Zahl von Werten gegeben ist; dies ist zum Beispiel bei der Darstellung durch ein Histogramm der Fall. Bei einem *glatten* Verlauf von $f(x)$ reichen bereits wenige Terme für eine genaue Darstellung durch eine Fouriersumme aus.

Die Faltung der Verteilung $f(x)$ mit einer Gauß-Verteilung mit Standardabweichung σ ist gegeben durch das Faltungsintegral

$$g(y) = \int_0^{2\pi} \frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{(y-x)^2}{2\sigma^2}\right) f(x)\ \mathrm{d}x\,. \tag{11.13}$$

Die Faltung ergibt die Verteilung, die man bei Messung mit begrenzter Meßgenauigkeit erhalten würde, wenn das Auflösungsvermögen durch die Gauß-Verteilung gegeben ist.

Setzt man die Fouriersumme (11.12) in das Faltungsintegral (11.13) ein, so erhält man für einen einzelnen $\sin \nu x$ Term

$$\int_0^{2\pi} \frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{(y-x)^2}{2\sigma^2}\right) \sin \nu x\ \mathrm{d}x = \exp\left(-\frac{\nu^2\sigma^2}{2}\right) \sin \nu y\,. \tag{11.14}$$

Ein einzelner Term hat also nach der Faltung dieselbe Form wie vorher, die Amplitude ist jedoch vermindert um den Faktor $\exp(\nu^2\sigma^2/2)$. Dies ist in Abbildung 11.1 für drei Terme gezeigt.

Dieses interessante Resultat, welches für alle $\cos \nu x$ und $\sin \nu x$ Terme gilt, zeigt, wie die Entfaltung in diesem Fall durchzuführen ist. Die gemessene Verteilung wird dargestellt in der Form

$$g(y) = \frac{\alpha_0}{2} + \sum_{\nu=1}^{\infty} (\alpha_\nu \cos \nu y + \beta_\nu \sin \nu y) \tag{11.15}$$

mit Formeln, die äquivalent zu Gleichung (10.32) sind. Die Entfaltung, d.h. die Berechnung der ursprünglichen Funktion, geschieht dann durch einen Koeffizientenvergleich mit den Formeln

$$a_\nu = \exp\left(\frac{\nu^2\sigma^2}{2}\right) \cdot \alpha_\nu \qquad b_\nu = \exp\left(\frac{\nu^2\sigma^2}{2}\right) \cdot \beta_\nu\,. \tag{11.16}$$

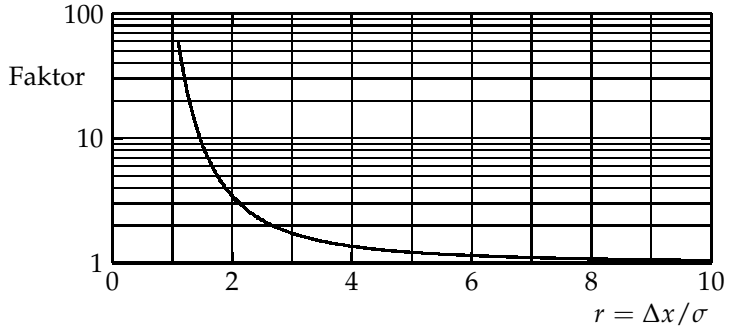


Abbildung 11.2: Faktor der Vergrößerung des statistischen Fehlers bei einer Messung mit Gaußscher Auflösung als Funktion des Verhältnisses von Intervallbreite Δx zur Standardabweichung σ.

Diese exakten Formeln zeigen exemplarisch die prinzipiellen Schwierigkeiten *jeder* Entfaltung. Die Koeffizienten α_ν und β_ν können nur mit gewissen statistischen Fehlern bestimmt werden, wobei die Werte selbst mit wachsendem Index ν in der Regel abnehmen. Die Gleichung (11.16) bedeutet dann Multiplikation mit einem oberhalb einer Schwelle mit ν stark anwachsenden Faktor, und das entfaltete Resultat wird bei größeren Werten von ν dann durch statistische Fluktuationen dominiert. Um große unsinnige Fluktuationen in der Lösung zu vermeiden, muß man also die Zahl der Fourierterme begrenzen und nur einen maximalen Wert $\nu_{\max}$ für ν zulassen. Diese notwendige Begrenzung verhindert natürlich die Auflösung feiner Details.

Die Faktoren in Gleichung (11.16) liefern eine Beziehung zwischen der Meßgenauigkeit σ und dem Korrekturfaktor. Wenn $\nu_{\max}$ der maximale Wert von ν ist, dann ist der zugehörige Faktor $\exp\left(\nu_{\max}^2\sigma^2/2\right)$. Üblicherweise wird ein Meßergebnis nicht durch Fourierkoeffizienten, sondern durch Datenpunkte mit einer bestimmten Intervallbreite Δx dargestellt. Der Mittelwert von $f(x)$ in einem Intervall wird durch Mittelung (mit Integration) über das Intervall der Darstellung (11.12) bestimmt. Mit einer Intervallbreite von $\Delta x = 2\pi/2\nu_{\max}$ ist der fragliche Faktor

$$\exp\left(\frac{1}{2}\nu_{\max}^2\sigma^2\right) = \exp\left(\frac{1}{2}\pi^2\left(\frac{\sigma}{\Delta x}\right)^2\right) = \exp\left(\frac{1}{2}\left(\frac{\pi}{r}\right)^2\right)$$

mit $r = \Delta x/\sigma$. Dies ist in Abbildung (11.2) dargestellt; der Faktor ist bereits 3.5 für $r = \Delta x/\sigma = 2$ und nimmt für kleinere Intervallbreiten schnell zu. Schlußfolgerung ist, daß bei einer Gaußschen Auflösung mit Standardabweichung σ eine Intervallbreite für die entfaltete Verteilung von der Größenordnung von etwa $\Delta x = 2\sigma$ erreicht werden kann; bei einer kleineren Intervallbreite wachsen die Fehlerfaktoren stark an.

11.5 Diskretisierung

Um $f(x)$ aus den Daten zu bestimmen, muß die Integral-Gleichung (11.2) diskretisiert werden wie Gleichung (11.3). Die oft benutzte Methode, die Funktionen $f(x)$, $A(y,x)$ und $g(y)$ durch Histogramme darzustellen, kann zu einem weiteren Verlust an Genauigkeit führen. Hier wird eine alternative Methode dargestellt.

Die Diskretisierung führt auf lineare Gleichungen. In einem ersten Schritt wird $f(x)$ durch eine Linearkombination

$$f(x) = \sum_{j=1}^{n} a_j p_j(x) \tag{11.17}$$

parametrisiert. Die $p_j(x)$ sind Basisfunktionen, die später spezifiziert werden. Die Parametrisierung der Gleichung (11.17) erlaubt die Integration

$$\int_a^b A(y,x)f(x)\,\mathrm{d}x = \sum_{j=1}^{n} a_j\left[\int_a^b A(y,x)p_j(x)\,\mathrm{d}x\right] = \sum_{j=1}^{n} a_j A_j(y)\ ,$$

wobei die Funktionen $A_j(y)$ durch Integrale

$$A_j(y) = \int_a^b A(y,x) p_j(x)\,\mathrm{d}x$$

gegeben sind. Nun kann Gleichung (11.2) geschrieben werden in der Form

$$g(y) = \sum_{j=1}^{n} a_j A_j(y) + b(y) + \varepsilon(y)\ , \tag{11.18}$$

wobei $A_j(y)$ jeweils einen Term $p_j(x)$ der Gleichung (11.17) repräsentiert. Danach werden alle y-abhängigen Funktionen (Gleichung (11.18)) durch Histogramme dargestellt in den Intervallen mit Grenzen $y_0, y_1, \ldots y_m$ gemäß

$$g_i = \int_{y_{i-1}}^{y_i} g(y)\ \mathrm{d}y \qquad A_{ij} = \int_{y_{i-1}}^{y_i} A_j(y)\ \mathrm{d}y \qquad b_i = \int_{y_{i-1}}^{y_i} b(y)\ \mathrm{d}y\,. \tag{11.19}$$

Die gemessenen Zahlen in den Intervallen seien g_i. Damit kann Gleichung (11.18) als Matrixgleichung

$$\boldsymbol{g} = \boldsymbol{A}\boldsymbol{a} + \boldsymbol{b} \tag{11.20}$$

geschrieben werden mit den n-Vektoren $\boldsymbol{g}$ und $\boldsymbol{b}$, die die Verteilungen der gemessenen Größe und des Untergrunds beschreiben. Der Parameter-Vektor $\boldsymbol{a}$ ist ein n-Vektor der Koeffizienten a_j, und $\boldsymbol{A}$ ist eine $m \times n$ Matrix. Die Spalte $\boldsymbol{A}_j$ von $\boldsymbol{A}$ repräsentiert das Histogramm in y für $f(x) = p_j(x)$.

Die Elemente A_{ij} von $\boldsymbol{A}$ können mit einer Stichprobe aus Monte Carlo-Verfahren bestimmt werden, indem einzelne Meßprozesse simuliert werden. Ausgehend von einer angenommenen Verteilung $f_0(x)$ der wahren Meßwerte x wird jeweils y berechnet und ein Gewicht proportional zu $p_j(x)$ zu Histogramm $\boldsymbol{A}_j(y)$ addiert, wobei man Gleichung (11.17) benutzt. Für die Basisfunktionen sollte möglichst $p_j(x) \geq 0$ gelten, da ihre Linearkombination eine (nichtnegative) Dichte beschreibt. Wenn B-Splines als Basisfunktionen $p_j(x)$ benutzt werden (siehe Kapitel 10.2), haben die Funktionen $p_j(x)$ die Eigenschaften

$$p_j(x) \geq 0 \qquad \sum_{j=1}^{n} p_j(x) = 1\ .$$

Auch die Koeffizienten a_j sollten dann nicht-negativ sein. Sind die Koeffizienten a_j bestimmt, so kann man für die Messung von $f(x)$ eine Darstellung durch Datenpunkte f_k mit Fehlern erhalten durch die Funktionswerte entweder bei festen Werten x_k oder durch Mittelung über das jeweilige Intervall in x gemäß

$$f_k = \frac{\left(\int_{x_{k-1}}^{x_k} f(x)\,\mathrm{d}x\right)}{(x_k - x_{k-1})} = \frac{\left(\sum_{j=1}^{n} a_j \int_{x_{k-1}}^{x_k} p_j(x)\,\mathrm{d}x\right)}{(x_k - x_{k-1})}\ ;$$

dabei bedeuten die Werte f_k die Mittelwerte der Funktion $f(x)$ im Intervall $[x_{k-1}, x_k]$. Da die Werte f_k lineare Funktionen der Koeffizienten a_j sind, folgt die Berechnung der Fehler nach dem Standardverfahren.

Es ist sinnvoll, bei der Monte Carlo-Simulation zur Berechnung der Matrix $\boldsymbol{A}(y,x)$ bereits von einer Verteilung $f_0(x)$ auszugehen, die nahe an der wahren Lösung $f(x)$ liegt. Dies entspricht einem Produktansatz $f(x) = f_0(x) \cdot f_1(x)$, wobei $f_1(x)$ nur schwach von x abhängt. Gleichung (11.1) läßt sich in der Form

$$g(y) = \int A(y,x) f_0(x) f_1(x)\,\mathrm{d}x = \int A_0(y,x) f_1(x)\,\mathrm{d}x$$

mit $A_0(y,x) = A(y,x) \cdot f_0(x)$ schreiben.

11.6 Entfaltung ohne Regularisierung

Die Lösung $f(x)$ ist durch die Linearkombination (11.17) parametrisiert. Mit der Diskretisierung im vorhergehenden Abschnitt erfolgt die Entfaltung mit der Bestimmung der Parameter $\boldsymbol{a}$ der Linearkombination durch eine Anpassung des linearen Ausdrucks $\boldsymbol{g} = \boldsymbol{A}\boldsymbol{a} + \boldsymbol{b}$ an die Daten. Die gemessenen Anzahlen g_i^{m} in den einzelnen Intervallen folgen einer Poisson-Verteilung mit Mittelwert $g_i(\boldsymbol{a})$. Deshalb können die Parameter $\boldsymbol{a}$ nach der Maximum-Likelihood-Methode mit der negativen Log-Likelihood-Funktion (siehe Gleichung (6.16))

$$S(\boldsymbol{a}) = \sum_i \left(g_i\left(\boldsymbol{a}\right) - g_i^{\mathrm{m}} \cdot \ln g_i(\boldsymbol{a})\right) \tag{11.21}$$

bestimmt werden; dabei wurde die Poisson-Verteilung zugrundegelegt. Die Bestimmung von $\boldsymbol{a}$ muß iterativ erfolgen, jeweils ausgehend von einer Näherung $\tilde{\boldsymbol{a}}$. Die Funktion $S(\boldsymbol{a})$ ist dann in der quadratischen Näherung

$$S(\boldsymbol{a}) = S(\tilde{\boldsymbol{a}}) + \left(\boldsymbol{a} - \tilde{\boldsymbol{a}}\right)^T \boldsymbol{h} + \frac{1}{2}\left(\boldsymbol{a} - \tilde{\boldsymbol{a}}\right)^T \boldsymbol{H}\left(\boldsymbol{a} - \tilde{\boldsymbol{a}}\right) \tag{11.22}$$

mit dem Gradienten $\boldsymbol{h}$ und der Hesse-Matrix $\boldsymbol{H}$. Der Standard-Lösungsweg ergibt eine Korrektur $\boldsymbol{\Delta a} = -\boldsymbol{H}^{-1}\,\boldsymbol{h}$ mit dem Endergebnis (nach Konvergenz) von $\widehat{\boldsymbol{a}} = \tilde{\boldsymbol{a}} + \boldsymbol{\Delta a}$ und der Kovarianzmatrix

$$\boldsymbol{V}[\widehat{\boldsymbol{a}}] = \boldsymbol{H}^{-1}\,. \tag{11.23}$$

Diese iterative bestimmte Lösung $\widehat{\boldsymbol{a}}$, die dem Minimum des Ausdrucks (11.21) bzw. (11.22) entspricht, ist oft unbrauchbar, da die Verteilung entsprechend dem Ansatz (11.17)

$$\widehat{f}(x) = \sum_{j=1}^{n} \widehat{a}_j p_j(x) \tag{11.24}$$

ein unplausibles oszillierendes Verhalten zeigen kann.

Orthogonalisierungsverfahren mit der Diagonalisierung der Hesse-Matrix erlauben eine Analyse dieses unerwünschten Verhaltens. Statt des Vektors $\boldsymbol{a}$ wird der transformierte Vektor $\boldsymbol{b}$ mit der Transformation

$$\boldsymbol{a} = \boldsymbol{M}_b\boldsymbol{b} \qquad \text{mit} \quad \boldsymbol{M}_b = \boldsymbol{U}_H\boldsymbol{D}^{-1/2} \tag{11.25}$$

benutzt, wobei die Matrizen $\boldsymbol{U}_H$ und $\boldsymbol{D}$ bei der Diagonalisierung der Hesse-Matrix entstehen

$$\boldsymbol{H} = \boldsymbol{U}_H\,\boldsymbol{D}\,\boldsymbol{U}_H^T \qquad \boldsymbol{D} = \boldsymbol{U}_H^T\,\boldsymbol{H}\,\boldsymbol{U}_H\,.$$

Die Matrix $\boldsymbol{D}$ enthält als Diagonalelemente die (reellen) Eigenwerte von $\boldsymbol{H}$, die positiv sind, wenn $\boldsymbol{H}$ positiv definit ist. Die Matrix $\boldsymbol{U}_H$ ist orthogonal mit der Eigenschaft $\boldsymbol{U}_H^T\boldsymbol{U}_H = \boldsymbol{I}$ und enthält die Eigenvektoren $\boldsymbol{u}_j$ zu den Eigenwerten als Spaltenvektoren. Die Eigenwerte D_{jj} können in absteigender Folge angeordnet werden; in typischen Anwendungen nehmen sie um einige Größenordnungen ab. Dann kann eine Diagonalmatrix $\boldsymbol{D}^{1/2}$ definiert werden mit der Eigenschaft $\boldsymbol{D}^{1/2}\boldsymbol{D}^{1/2} = \boldsymbol{D}$, die die positiven Quadratwurzeln der Eigenwerte in der Diagonalen hat. Entsprechend definiert man auch die inverse Matrix $\boldsymbol{D}^{-1/2}$, die in der Transformationsformel (11.25) steht.

Ausgedrückt durch den Vektor $\boldsymbol{b}$ lautet die zu minimierende Funktion (11.22) nun

$$S(\boldsymbol{b}) = S_0 + \boldsymbol{b}^T\boldsymbol{h}_b + \frac{1}{2}\boldsymbol{b}^T\boldsymbol{b} \qquad \text{mit} \quad \boldsymbol{h}_b = \boldsymbol{D}^{-1/2}\boldsymbol{U}_H^T\left(-\boldsymbol{H}\tilde{\boldsymbol{a}} + \boldsymbol{h}\right)\,,$$

wobei S_0 alle konstanten Beiträge zusammenfaßt und die Hesse-Matrix bezüglich $\boldsymbol{b}$ die Einheitsmatrix ist. Die Lösung für den transformierten Vektor und seine Kovarianzmatrix ist nun einfach gegeben durch

$$\widehat{\boldsymbol{b}} = -\boldsymbol{h}_b \qquad \text{und} \qquad \boldsymbol{V}[\widehat{\boldsymbol{b}}] = \boldsymbol{I} \; .$$

Die Komponenten von $\boldsymbol{b}$ haben Varianz 1 und sind unkorreliert. Analog zur Darstellung (11.24) kann für die transformierten Koeffizienten das Ergebnis dargestellt werden durch

$$\widehat{f}(x) = \sum_{j=1}^{n} \widehat{b}_j \, p_j^{'}(x) \tag{11.26}$$

mit den Koeffizienten $\widehat{b}_j$. Die Funktionen $p_j^{'}(x)$ sind durch Linearkombination der Funktionen $p_j(x)$ entsprechend der Transformation (11.25) entstanden und bilden ein orthonormales Funktionensystem, das genau dem speziellen Problem mit seinen statistischen Fehlern und der Auflösung angepaßt ist. Jeder einzelne Koeffizient wird mit dem gleichen statistischen Fehler gemessen. Wegen der Sortierung der Eigenwerte D_{jj} nach absteigender Folge werden die Funktionen $p_j^{'}(x)$ mit wachsendem Wert des Index mehr und mehr Oszillationen zeigen, ähnlich zum Beispiel den Legendre-Polynomen. In der Regel sind die wahren Werte der Koeffizienten b_j für größere Werte des Index sehr klein, und die Meßwerte sind dann mit null verträglich.

Das Ergebnis kann nun wieder zum Vektor $\boldsymbol{a}$ zurücktransformiert werden nach Gleichung (11.25). Bei der Rücktransformation der Kovarianzmatrix erhält man natürlich wieder das Ergebnis (11.23). Die Rücktransformation kann auch in der Form

$$\widehat{\boldsymbol{a}} = \boldsymbol{U}_H \boldsymbol{D}^{-1/2} \boldsymbol{b} = \sum_{j=1}^{n} \left(\frac{1}{D_{jj}} \right)^{1/2} \widehat{b}_j \boldsymbol{u}_j$$

geschrieben werden. Dieser Ausdruck stellt den Vektor $\widehat{\boldsymbol{a}}$ als Linearkombinationen der Eigenvektoren $\boldsymbol{u}_j$ dar mit zwei Faktoren, dem Koeffizienten $\widehat{b}_j$, der Varianz 1 hat, und der Wurzel von $1/D_{jj}$. Hieraus kann man das oszillierende Verhalten der Lösungen verstehen. Die Beiträge der Eigenvektoren mit Koeffizienten $\widehat{b}_j$ mit einem großen Wert des Index j, die mit null verträglich sind, werden mit großen Faktoren multipliziert aufgrund des kleinen Eigenwertes, und dies beeinflußt dann die Lösung stark. Um eine sinnvolle Lösung zu erhalten, muß der Beitrag dieser Terme ausgeschlossen werden, indem man die Summe nur bis zu einem Index $n_0 \; (, n)$ bildet. Ein scharfes Abschneiden bei einem bestimmten Index n_0 kann in der Lösung bestimmte Fluktuationen erzeugen, bekannt als *Gibbs'sches Phänomen*. Ein sanftes Abschneiden erreicht man durch die nachfolgend beschriebene Regularisierungsmethode.

11.7 Entfaltung mit Regularisierung

Die Größe der Fluktuationen in einer Lösung kann auf verschiedene Weise gemessen werden. Eine Möglichkeit ist die Benutzung des Quadrats der ersten Ableitung von $f(x)$. Hier wird das Quadrat der zweiten Ableitung benutzt, die bereits bei den Spline-Funktionen erwähnte totale Krümmung der Gleichung (10.9).

Wenn $f(x)$ nach Gleichung (11.17) durch eine Summe von B-Splines (siehe Kapitel 10.2) der vierten Ordnung parametrisiert wird (kubische B-Splines), dann hat die Krümmung die einfache Form eines quadratischen Ausdrucks $r(\boldsymbol{a}) = \boldsymbol{a}^T \boldsymbol{C} \boldsymbol{a}$ mit einer (konstanten) symmetrischen, positiv-semidefiniten Matrix $\boldsymbol{C}$ gemäß

$$r(\boldsymbol{a}) = \int \left(f''(x) \right)^2 \, \mathrm{d}x = \boldsymbol{a}^T \boldsymbol{C} \boldsymbol{a} \tag{11.27}$$

mit einer Matrix $\boldsymbol{C}$, die (bis auf einen unwichtigen konstanten Faktor) gegeben ist durch

$$\boldsymbol{C} = \begin{pmatrix} 2 & -3 & 0 & 1 & 0 & 0 & \dots \\ -3 & 8 & -6 & 0 & 1 & 0 & \\ 0 & -6 & 14 & -9 & 0 & 1 & \\ 1 & 0 & -9 & 16 & -9 & 0 & \\ 0 & 1 & 0 & -9 & 16 & -9 & \\ 0 & 0 & 1 & 0 & -9 & 16 & \\ \vdots & & & & & & \ddots \end{pmatrix} \tag{11.28}$$

für kubische B-Splines bei äquidistanten Stützstellen. Für eine stark fluktuierende Lösung nimmt die Krümmung große Werte an.

Man führt nun, entsprechend dem Konzept der Regularisierung, eine neue Funktion $R(\boldsymbol{a})$ ein, indem man zur negativen Log-Likelihood-Funktion $S(\boldsymbol{a})$ (Gleichung (11.21)) die totale Krümmung $r(\boldsymbol{a})$ mit einem Faktor $\tau/2$ addiert

$$\begin{aligned} R(\boldsymbol{a}) &= S(\boldsymbol{a}) + \frac{1}{2}\tau \cdot r(\boldsymbol{a}) \qquad (11.29) \\ &= S(\tilde{\boldsymbol{a}}) + (\boldsymbol{a}-\tilde{\boldsymbol{a}})^T \boldsymbol{h} + \frac{1}{2}(\boldsymbol{a}-\tilde{\boldsymbol{a}})^T \boldsymbol{H} (\boldsymbol{a}-\tilde{\boldsymbol{a}}) \\ &\quad + \frac{1}{2}\tau \cdot \boldsymbol{a}^T \boldsymbol{C} \boldsymbol{a} \ , \end{aligned}$$

und das minimum dieser Funktion bezüglich $\boldsymbol{a}$ bestimmt für einen vorgegebenen Wert des Regularisierungsparameters τ.

Offensichtlich kann die Regularisierung eine Verzerrung des Ergebnisses verursachen, abhängig von der Größe von τ. Für $\tau \to 0$ verschwindet der Einfluß der Regularisierung, und im Grenzfall $\tau \to \infty$ erzwingt der Regularisierungsterm nach Gleichung (11.27) eine *lineare* Lösung $f(x)$, bei der die Krümmung verschwindet.

Wie im folgenden gezeigt wird, kann man τ so wählen, daß der Einfluß der Verzerrung klein ist im Vergleich zu den statistischen Fehlern. Die Regularisierung bewirkt ein sanftes Abschneiden der stärker fluktuierenden Beiträge.

Um zu sehen, welche Beiträge statistisch relevant sind, transformiert man die Hesse-Matrix auf Diagonalform in einem Verfahren analog zu Gleichung (11.25). Die Hesse-Matrix wird zur Einheitsmatrix, d.h. die neuen Variablen sind unkorreliert und haben Varianz 1. Dies macht es leicht, ihre statistische Signifikanz festzustellen. In dem hier vorliegenden Fall muß allerdings die Matrix $\boldsymbol{C}$ simultan diagonalisiert werden. Ein allgemeines Verfahren für dieses Problem ist in Kapitel 3.8 beschrieben. Man transformiert zunächst wie im Falle ohne Regularisierung (Gleichung (11.25)) mit einer symmetrischen Matrix $\boldsymbol{M}_b = \boldsymbol{U}_H \boldsymbol{D}^{-1/2} \boldsymbol{U}_H^T$; dabei entsteht die Matrix $\boldsymbol{M}_b \boldsymbol{C} \boldsymbol{M}_b$, die diagonalisiert wird,

$$\boldsymbol{M}_b \boldsymbol{C} \boldsymbol{M}_b = \boldsymbol{U}_C \, \boldsymbol{S} \, \boldsymbol{U}_C^T \qquad \boldsymbol{S} = \boldsymbol{U}_C^T \, \boldsymbol{M}_b \boldsymbol{C} \boldsymbol{M}_b \, \boldsymbol{U}_C \ .$$

$\boldsymbol{S}$ ist eine Diagonalmatrix, bei der die Eigenwerte S_{jj} in aufsteigender Folge (entsprechend einer mit j anwachsenden Krümmung) angeordnet werden. Der Vektor $\boldsymbol{a}$ wird mit der Transformationsformel

$$\boldsymbol{a} = \boldsymbol{M}_c \, \boldsymbol{c} \qquad \text{mit} \quad \boldsymbol{M}_c = \boldsymbol{U}_H \boldsymbol{D}^{-1/2} \boldsymbol{U}_H^T \boldsymbol{U}_C \tag{11.30}$$

in einen Vektor $\boldsymbol{c}$ transformiert; die zu minimierende Funktion wird damit

$$R(\boldsymbol{c}) = R_0 + \boldsymbol{c}^T \boldsymbol{h}_c + \frac{1}{2}\boldsymbol{c}^T \boldsymbol{c} + \frac{1}{2}\tau \cdot \boldsymbol{c}^T \boldsymbol{S} \boldsymbol{c} \ .$$

Die beiden den Vektor $\boldsymbol{c}$ enthaltenden quadratischen Formen haben nun Diagonalgestalt. Das Minimum findet man durch Differenzieren und erhält die Bedingung

$$\boldsymbol{h}_c + (\boldsymbol{I} + \tau \cdot \boldsymbol{S})\, \boldsymbol{c} = 0 \quad \text{und} \quad \boldsymbol{c} = -\,(\boldsymbol{I} + \tau \cdot \boldsymbol{S})^{-1}\, \boldsymbol{h}_c$$

für den transformierten Vektor $\boldsymbol{c}$. Die Lösung mit $\tau = 0$ ist die unregularisierte Lösung $\boldsymbol{b}$, sodaß man

$$\boldsymbol{c} = (\boldsymbol{I} + \tau \cdot \boldsymbol{S})^{-1}\, \boldsymbol{b}$$

schreiben kann. Weil die Matrizen diagonal sind, sind die Koeffizienten c_j der regularisierten Lösung ($\tau \neq 0$) leicht zu berechnen aus den Koeffizienten b_j der unregularisierten Lösung ($\tau = 0$) durch

$$c_j = \frac{1}{1 + \tau \cdot S_{jj}}\, b_j \; . \tag{11.31}$$

Die Koeffizienten der regularisierten Lösung erhält man also durch Multiplikation der Koeffizienten der unregularisierten Lösung mit dem Faktor $1/(1 + \tau S_{jj})$, der nahe eins ist für alle Indizes j mit $S_{jj} \ll \tau^{-1}$. Da die Werte von S_{jj} mit j anwachsen, geht der Abschwächungsfaktor gegen null für $S_{jj} \gg \tau^{-1}$ nach einem Übergangsgebiet, in dem $S_{jj} \approx \tau^{-1}$ ist. Die Regularisierung erzeugt also ein allmähliches Abschneiden der höheren Terme. Wo soll abgeschnitten werden? Dort, wo die Koeffizienten b_j ab einem bestimmten Index nicht mehr signifikant von null verschieden sind; dies möge bei dem Index n_1 der Fall sein. Dann sollte der Regularisierungsparameters τ von gleicher Größenordnung sein wie $1/S_{n_1,n_1}$. Die *effektive* Zahl der Freiheitsgrade wird durch die Dämpfung nach Gleichung (11.31) kleiner als die Gesamtzahl n der Koeffizienten; man kann diese effektive Zahl für einen gegebenen Wert von τ gleich der Summe der Faktoren in Gleichung (11.31) setzen.

Dies erlaubt nun eine Bestimmung von τ zu einer vorgegebenen Zahl von Freiheitsgraden n_0. Man wählt einen Wert n_0 knapp oberhalb von n_1 und bestimmt τ entsprechend dem Ausdruck

$$m_0 = \sum_{j=1}^{m} \frac{1}{1 + \tau S_{jj}} \; . \tag{11.32}$$

Da die Kovarianzmatrix von $\boldsymbol{b}$ die Einheitsmatrix ist, erhält man für die Kovarianzmatrix des Vektors $\boldsymbol{c}$

$$\boldsymbol{V}[\boldsymbol{c}] = (\boldsymbol{I} + \tau \cdot \boldsymbol{S})^{-2} \; .$$

Das Resultat kann entsprechend der Transformation (11.30) zurücktransformiert werden in den Koeffizientenvektor $\boldsymbol{a}$.

Die resultierenden Koeffizienten a_j sollten schließlich in etwa n_0 Datenpunkte $\widehat{f}_k$ verwandelt werden, indem man $\widehat{f}(x)$ über die entsprechenden Intervalle von x integriert.[1] Die Wahl dieser Intervalle bestimmt die Korrelationen zwischen den Datenpunkten, die möglichst klein sein sollten. Bei annähernd gleicher Genauigkeit für den ganzen x-Bereich kann man Intervalle gleicher Breite nehmen. Bei unterschiedlicher Genauigkeit kann man wie folgt vorgehen:
Da die Funktion $p'_{n_0+1}(x)$ (Gleichung (11.26)) gerade n_0 Nullstellen hat, erscheint es optimal, die n_0 Intervalle um diese Nullstellen herumzulegen, mit den $(n_0 - 1)$ x-Werten, an denen

[1] Man könnte zwar eine größere Zahl von Datenpunkten wählen, aber der Rang der Kovarianzmatrix bliebe unverändert n_0.

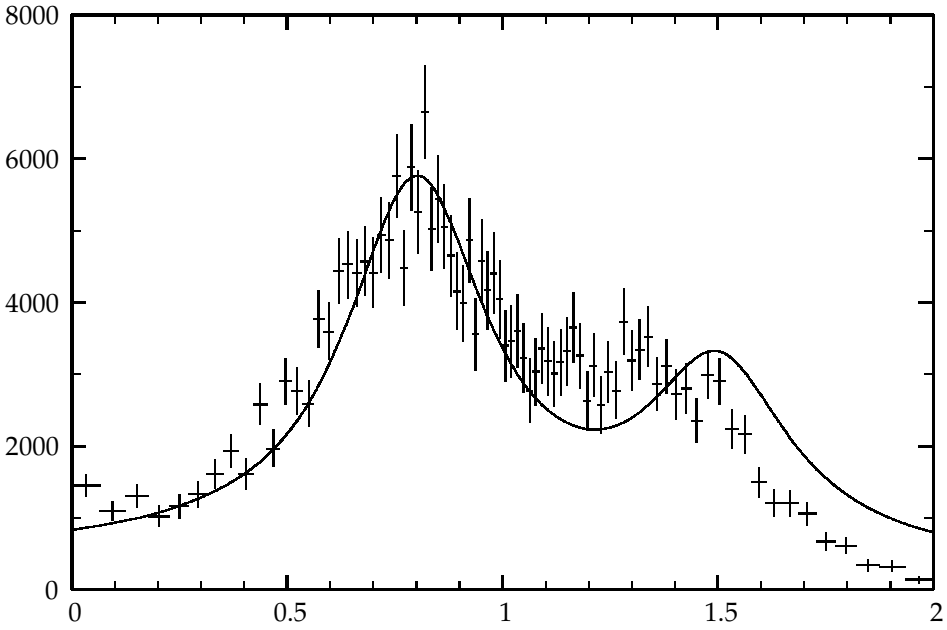


Abbildung 11.3: Die Kurve zeigt die Testverteilung (Anzahl der Einträge dividiert durch die Intervallbreite), die aus der Summe von drei Cauchy-Verteilungen gebildet ist. Das Histogramm mit Fehlerbalken zeigt das simulierte Ergebnis einer Messung von 10 000 Einzelwerten. Durch Detektoreffekte wird das Tal zwischen den beiden Maxima aufgefüllt.

die Funktion $p'_{n_0+1}(x)$ Extremwerte hat, als Intervallgrenzen. Dies wird den (n_0+1)-ten Beitrag unterdrücken, und berücksichtigt die statistische Genauigkeit und die Meßgenauigkeit der x-Bereiche. Jeder Wert ist eine Linearkombination von Koeffizienten, und die Fehlerfortpflanzung wird nach den üblichen Regeln durchgeführt.

Benutzt man das entfaltete Resultat $\hat{f}(x)$ als Gewichtsfaktor in einer Monte Carlo-Rechnung, so kann man durch Vergleich der vorhergesagten Verteilung mit der direkt gemessenen Verteilung die Simulation kontrollieren. Dieser Test kann auch auf Meßgrößen ausgedehnt werden, die nicht direkt in der Entfaltung enthalten waren. Diese Kontrolle kann durch Umgewichtung sogar mit der ursprünglichen Simulation, die zur Bestimmung der Response-Matrix benutzt wurde, durchgeführt werden. Dabei ist es von Vorteil, daß das entfaltete Resultat $\hat{f}(x)$ nicht durch ein Histogramm dargstellt wird, sondern durch die Verwendung der Basisfunctionen einen glatten Verlauf hat.

Beispiel 11.2 Entfaltung mit Regularisierung.

In diesem Beispiel wird in einer Monte Carlo-Rechnung die Messung einer Variablen x im Bereich $0 \leq x \leq 2$ simuliert mit den folgenden Eigenschaften: Die Akzeptanz wird angenommen als $P(x) = 1 - 1/2\,(x-1)^2$. Die wahren Werte von x werden transformiert in die Variable y_{tr} durch $y_{tr} = x - 0.05x^2$, um eine Nichtlinearität des Detektorverhaltens zu simulieren. Die Variable y_{tr} wird mit einem gauß-verteilten Fehler mit Standardabweichung $\sigma = 0.1$ *verschmiert*, und daraus resultiert die gemessene Variable y. Als wahre Funktion wird eine gewichtete Sum-

me aus drei Cauchy-Verteilungen

$$f(x) = \sum_{k=1}^{3} b_k \frac{g_k^2}{(x - x_k)^2 + g_k^2}$$

k	b_k	x_k	g_k
1	1.0	0.4	2.0
2	10.0	0.8	0.2
3	5.0	1.5	0.2

im Bereich $0 \leq x \leq 2$ angenommen mit den angegebenen Parameterwerten. Insgesamt 10 000 Monte Carlo-Werte von x werden gemäß der Funktion $f(x)$ erzeugt. Nach Anbringung von Akzeptanz-, Transformations- und Meßfehlereinflüssen entsprechend den oben gegebenen Annahmen bleiben 8950 y-Werte.

j	S_{jj}	$\widehat{b}_j$	$\widehat{c}_j$	Faktor
1	0	46.42	46.42	1.000
2	0	36.45	36.45	1.000
3	0.71×10^{-6}	23.03	23.03	1.000
4	0.74×10^{-5}	8.92	8.92	1.000
5	0.38×10^{-4}	4.56	4.56	1.000
6	0.15×10^{-3}	9.60	9.59	0.999
7	0.48×10^{-3}	0.25	0.25	0.998
8	0.15×10^{-2}	0.38	0.38	0.993
9	0.43×10^{-2}	0.29	0.29	0.981
10	0.13×10^{-2}	0.91	0.86	0.945
11	0.36×10^{-1}	1.74	1.49	0.858
12	0.118	1.64	1.06	0.647
13	0.393	0.19	0.07	0.355
14	1.595	0.96	0.11	0.012
15	4.635	0.33	0.01	0.045
16	7.393	1.06	0.03	0.028
17	10.963	1.09	0.02	0.019
18	16.590	1.51	0.02	0.013

Tabelle 11.1: Unregularisierte Koeffizienten $\widehat{b}_j$ und regularisierte Koeffizienten $\widehat{c}_j$ der Lösung, zusammen mit den Eigenwerten S_{jj} und dem Regularisierungsfaktor.

Die Transformation beschränkt y_{tr} auf $0 \leq y_{\text{tr}} \leq 1.8$, mit dem zusätzlichen Effekt, daß die zwei Maxima etwas schmaler werden. Die endliche Auflösung macht die Maxima wieder breiter und füllt das Tal zwischen ihnen auf. In Abbildung 11.3 ist die simulierte Messung als Histogramm mit Fehlerbalken gezeigt. Die durchgezogene Kurve stellt die ursprüngliche Funktion dar.

Nun wird die Entfaltung mit Regularisierung angewandt. Zur Diskretisierung werden Spline-Funktionen mit äquidistanten Knotenabständen benutzt. Der Regularisierungsparameter τ wird für $n_0 = 12$ bestimmt. Die Koeffizienten $\widehat{c}_j$ der regularisierten Lösung erhält man durch Multiplikation der Koeffizienten $\widehat{b}_j$ mit den Faktoren der Regularisierung; die Parameter der Entfaltung sind in der Tabelle 11.1 gezeigt. Nur die ersten sechs Koeffizienten sind statistisch signifikant.

Schließlich werden aus den Koeffizienten Datenpunkte berechnet, die Mittelwerte für die entfaltete Funktion $f(x)$ in den Intervallen repräsentieren. Das Endresultat ist in Abbildung 11.4 zu sehen, zusammen mit der ursprünglichen Funktion $f(x)$. Die waagerechten Balken geben

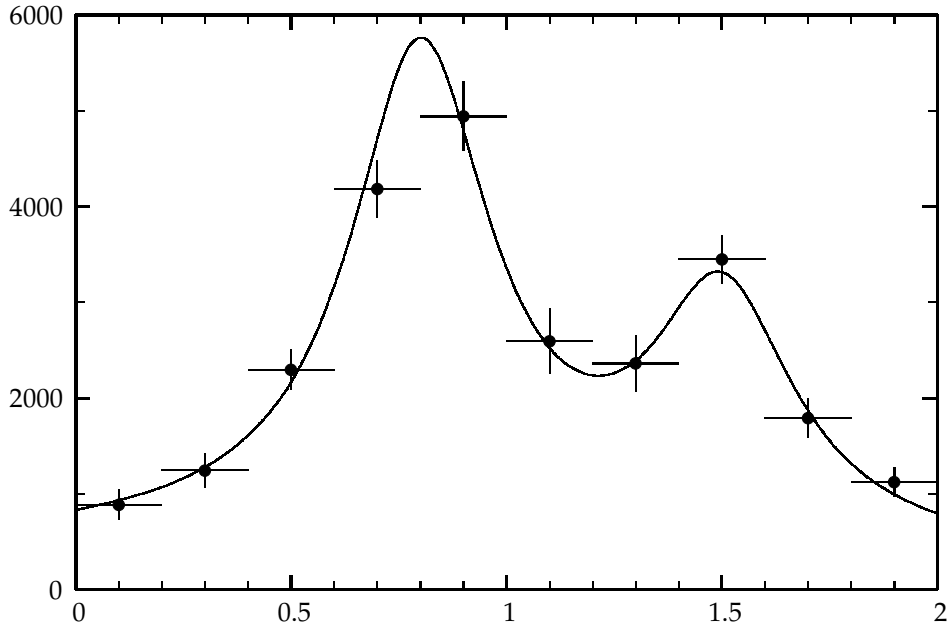


Abbildung 11.4: Das Ergebnis der Entfaltung, dargestellt durch zehn Datenpunkte mit Fehlerbalken. Zum Vergleich ist die ursprüngliche (wahre) Verteilung als Kurve gezeigt.

den x-Bereich an, über den gemittelt wurde. Die entfalteten Daten stimmen innerhalb der statistischen Fehler mit der ursprünglichen Funktion überein, und auch das Tal zwischen den Maxima, in dem *gemessenen* Histogramm kaum zu sehen (Abbildung 11.3), wird reproduziert. Natürlich sind die statistischen Fehler viel größer als in dem Histogramm der erzeugten Daten in Abbildung 11.3. Dieser Verlust statistischer Genauigkeit ist auf die endliche Meßgenauigkeit zurückzuführen.

Die Regularisierung bewirkt ein allmähliches Abschneiden der insignifikanten höheren Terme der Entwicklung nach orthogonalen Funktionen. Würde man die höheren Terme mitnehmen, so erhielte man unphysikalische oszillierende Beiträge. Zur Bestimmung des Regularisierungsgewichts dienen Standardtests der Statistik; sie gewährleisten, daß Verzerrungen durch die Regularisierung unterhalb der statistischen Fehler bleiben. Die Methode liefert mittlere Funktionswerte $\widehat{f_k}$ der entfalteten Lösung, die untereinander nur schwach korreliert sind.

A Fortran-Programme

A.1 Sortierprogramme

Man hat n Datensätze A_i, $i = 1, 2, \ldots n$ zu sortieren, wobei jeder Datensatz A_i einen Sortierschlüssel t_i enthält. Vertauschoperationen für die Datensätze sind im allgemeinen sehr aufwendig. Falls die Tabelle $A(1:N)$ aus großen Datensätzen besteht, ist es daher sinnvoll, einen Indexvektor einzuführen, welcher auf die Sortierschlüssel t_i der Datensätze zeigt. Der Indexvektor hat die Elemente I_j, $j = 1, 2, \ldots n$. Die Algorithmen werden dann so abgeändert, daß sie auf den Indexvektor zugreifen und Elemente des Indexvektors statt ganzer Datensätze vertauschen. Das Resultat ist ein Algorithmus, der den Indexvektor sortiert, so daß der kleinste Schlüssel t_i den Index I_1, der nächste Schlüssel den Index I_2 hat usw.

Shell's Sortieralgorithmus. Das Unterprogramm `ISORTS` ist eine auf diese Weise modifizierte Implementation von Shell's Sortieralgorithmus. Ehe diese Prozedur aufgerufen wird, muß der Indexvektor $\boldsymbol{I}$, im Programm der Vektor `IDX(1:N)`, mit Indizes gefüllt sein, welche auf die Tabelle `A` zeigen, so daß das Element (integer) `A(IDX(I))` der Sortierschlüssel t_i von A_i ist. Die Prozedur `ISORTS` kann auch benutzt werden, um einen Vektor `A(1:N)` von einfachen Zahlen zu sortieren. In diesem Fall setzt man `IDX(I)=I` für alle `I`. Für spezielle Anwendungen (reelle Zahlen statt ganzer Zahlen, mehrfacher Schlüssel) kann das Unterprogram leicht abgeändert werden. im Anhang auf Seite 255 ...

Subprogram `isorts.F`

Listing A.1: Sortierung – Shell's Algorithmus

```
      SUBROUTINE ISORTS(A,IDX,N)  ! Shell sort (integer)
*     integer keys A(*), unchanged at return
*     N indices IDX(1) ... IDX(N), changed at return such that
*        A(IDX(L)) <= A(IDX(L+1))
      INTEGER I,J,M,IDXT,N,IDX(*)
      INTEGER A(*)                ! array of keys
      M=4
 10   IF(3*M+1.LT.N) THEN         ! determine initial increment
        M=3*M+1
        GOTO 10
      END IF
 20   M=M/3                       ! next increment
      DO I=1+M,N                  ! ... used in insert sort
       IDXT=IDX(I)
       J=I
 30    IF(A(IDX(J-M)).GT.A(IDXT)) THEN
         IDX(J)=IDX(J-M)
         J=J-M
         IF(J.GT.M) GOTO 30
       END IF
       IDX(J)=IDXT
      END DO
      IF(M.GT.1) GOTO 20
      END
```

Sortieren eines Feldes von ganzen Zahlen. Vor dem Aufruf des Unterprogramms `ISORTS` muß der Vektor `IDX(1:N)` mit den Indizes der zu sortierenden Elemente des Feldes `A(*)` definiert werden. In dem folgenden kurzen Programmbeispiel werden die Elemente des Vektors auf die Werte 1 bis `N` gesetzt. Nach dem Aufruf von `ISORTS` ist das Feld `A(*)` unverändert, und der Vektor `IDX(1:N)` enthält die Indizes 1 bis `N` in der Sortierreihenfolge.

```
      INTEGER A(N),IDX(N)
      DO I=1,N
```

```
      IDX(I)=I
    END DO
    CALL ISORTS(A,IDX,N)
```

Quicksort. Die folgende Liste enthält eine Implementierung des Algorithmus mit einem sogen. stack (d.h. kein rekursiver Aufruf). Der Aufruf ist identisch mit dem Aufruf bei dem Unterprogramm `ISORTS`.

Subprogram `isortq.F`

Listing A.2: Sortierung – Quicksort Algorithmus

```
      SUBROUTINE ISORTQ(A,IDX,N)     ! quick sort (integer)
*     integer keys A(*), unchanged at return
*     N indices IDX(1) ... IDX(N), changed at return such that
*        A(IDX(L)) <= A(IDX(L+1))
      INTEGER NLEV            ! stack size
      PARAMETER (NLEV=2*32) ! ... for N = 2**32 = 4.3 10**9
      INTEGER I,J,L,R,IDXT,N,IDX(N),LEV,LR(NLEV)
      INTEGER A(*),AT         ! array of keys and pivot key
      LEV=0
      L=1
      R=N
 10   IF(R-L.EQ.1) THEN       ! sort two elements
        IF(A(IDX(L)).GT.A(IDX(R))) THEN
          IDXT   = IDX(L) ! exchange
          IDX(L) = IDX(R)
          IDX(R) = IDXT
        END IF
        R=L
      END IF
      IF(R.EQ.L) THEN
        IF(LEV.LE.0) RETURN
        LEV=LEV-2
        L=LR(LEV+1)
        R=LR(LEV+2)
      ELSE
        AT=A(IDX((L+R)/2))
        I=L-1         ! find limits [J,I] with [L,R]
        J=R+1
 20     I=I+1
        IF(A(IDX(I)).LT.AT) GOTO 20
 30     J=J-1
        IF(A(IDX(J)).GT.AT) GOTO 30
        IF(I.LE.J) THEN
          IDXT   = IDX(I)
          IDX(I) = IDX(J)
          IDX(J) = IDXT
          GO TO 20
        END IF
        IF(LEV+2.GT.NLEV) STOP ' SORTQS: stack overflow '
        IF(R-I.LT.J-L) THEN
          LR(LEV+1)=L
          LR(LEV+2)=J
          L=I
        ELSE
          LR(LEV+1)=I
          LR(LEV+2)=R
          R=J
        END IF
        LEV=LEV+2
      END IF
      GO TO 10
```

```
      END
```

Heapsort. Das Unterprogramm FSORTH ist eine durch die Einführung des Indexvektors weiterentwickelte Version einer Routine bei [58]. Der Aufruf ist identisch mit dem Aufruf bei dem Unterprogramm ISORTS, jedoch sind hier die Sortierschlüssel vom Typ REAL.

Subprogram `fsorth.F`

Listing A.3: Sortierung – Heapsort Algorithmus

```
      SUBROUTINE FSORTH(A,IDX,N) ! heap sort (real)
*     real keys A(*), unchanged at return
*     N indices IDX(1) ... IDX(N), changed at return such that
*        A(IDX(L)) <= A(IDX(L+1))
      INTEGER I,J,L,R,IDXT,N,IDX(N)
      REAL A(*),AT    ! array of keys and pivot key value
      L=N/2+1
      R=N
   10 IF(L.GT.1) THEN
         L=L-1
         IDXT=IDX(L)
         AT  =A(IDXT)
      ELSE
         IDXT=IDX(R)
         AT  =A(IDXT)
         IDX(R)=IDX(1)
         R=R-1
         IF(R.EQ.1) THEN
            IDX(1)=IDXT
            RETURN
         END IF
      END IF
      I=L
      J=L+L
   20 IF(J.LE.R) THEN
         IF(J.LT.R) THEN
            IF(A(IDX(J)).LT.A(IDX(J+1))) J=J+1
         END IF
         IF(AT.LT.A(IDX(J))) THEN
            IDX(I)=IDX(J)
            I=J
            J=J+J
         ELSE
            J=R+1
         END IF
         GOTO 20
      END IF
      IDX(I)=IDXT
      GOTO 10
      END
```

A.2 Suchalgorithmen

Binäre Suche. In der Funktion LEFT wird die binäre Suche angewandt, um in einer geordneten Liste t_i, i=1,2..n das Intervall $[t_l, t_{l+1}]$ zu finden, welches einen vorgegebenen Wert x einschliesst.

Subprogram `left.F`

Listing A.4: Binäre Suche

```
      FUNCTION LEFT(X,TAB,N) ! binary search in ordered table TAB(N)
*     (multiple table values allowed, e.g. for B-splines)
*     returned function value L with
*        TAB(L) <= X < TAB(L+1), i.e. outlow: L=0; outhigh: L=N
*     exception if X =TAB(N): TAB(L) <  X <= TAB(L+1)
      INTEGER L,R,N,LR
      REAL TAB(N),X
      L=0
      R=N+1
   10 LR=(L+R)/2     ! middle index
      IF(LR.NE.L) THEN
         IF(X.GT.TAB(LR)) THEN
            L=LR      ! new left limit
         ELSE
            R=LR      ! new right limit
         END IF
         GOTO 10
      END IF
   20 IF(LR.LT.N) THEN ! check multiple keys
         IF(X.EQ.TAB(LR+1)) THEN
            LR=LR+1
            GOTO 20
         END IF
      ELSE
         LR=N        ! check exception X=TAB(N)
   30    IF(X.LE.TAB(LR)) THEN ! multiple keys
            LR=LR-1
            GOTO 30
       END IF
      END IF
      LEFT=LR
      END
```

Äquivalenzklassen.

Subprogram `equicl.F`

Listing A.5: Äquivalenzklassen

```
      SUBROUTINE EQUICL(NC,N,LISTJK,M) ! equivalence classes
*     Elements are numbered 1 ... N and result is in array NC(N).
*     Input is LISTJK(2,M) with M pairs of equivalent elements,
*     for example: LISTJK(1,7)=1  LISTJK(2,7)=3 means:
*     elements 1 and 3 are equivalent.
*     Result is in array NC(N): NC(I) is the equivalence class number
*     for element I, and all equivalent elements will have the same
*     equivalence class number.
      INTEGER NC(N),LISTJK(2,M),J,K,L,M,N
      DO K=1,N
       NC(K)=K              ! initialize classes
      END DO
      DO L=1,M              ! loop on pairs
        J=LISTJK(1,L)
   10   IF(NC(J).NE.J) THEN
           J=NC(J)
           GOTO 10
        END IF
        K=LISTJK(2,L)
   20   IF(NC(K).NE.K) THEN
           K=NC(K)
           GOTO 20
        END IF
        IF(J.NE.K) NC(J)=K
```

```
      END DO
      DO J=1,N
30     IF(NC(J).NE.NC(NC(J))) THEN
         NC(J)=NC(NC(J))
         GOTO 30
       END IF
      END DO
      END
```

Minimum spanning tree. Der Algorithmus ist in dem Unterprogramm `MSTREE` programmiert. Die Abstände oder Gewichte w_k (`DISTJK` im Programm genannt) werden als nicht-negativ angenommen. Im Unterprogramm werden alle im *minimum spanning tree* nicht benutzten Gewichte negativ gesetzt. Das Datenfeld `NC` hat die gleiche Bedeutung wie in dem vorhergehenden Unterprogramm `EQUICL`.

Subprogram `mstree.F`

Listing A.6: Minimum spanning tree

```
      SUBROUTINE MSTREE(NC,N,LISTJK,DISTJK,IDX,M) ! minimum spanning tree
*     Elements are numbered 1 ... N and result is in array NC(N).
*     Input is LISTJK(2,M) with M pairs of equivalent elements,
*     and DISTJK(M), the non-negative distances between the elements.
*     IDX(M) is an auxiliary array used for sorting the distances.
*     Result is in array NC(N): NC(I) is the equivalence class number
*     for element I, and all equivalent elements will have the same
*     equivalence class number.
      INTEGER NC(N),IDX(M),LISTJK(2,M) ! pairs of items
      REAL DISTJK(M) ! non-negative distances between pairs
      INTEGER I,J,K,M,N
*     ...
      DO I=1,M
       IDX(I)=I                  ! prepare index vector
      END DO
      CALL FSORTH(DISTJK,IDX,M) ! real version of heapsort
      DO J=1,N
       NC(J)=J                   ! initialize classes
      END DO
      DO I=1,M       ! loop in order of increasing distance
       J=LISTJK(1,IDX(I))
10     IF(NC(J).NE.J) THEN
         J=NC(J)
         GOTO 10
       END IF
       K=LISTJK(2,IDX(I))
20     IF(NC(K).NE.K) THEN
         K=NC(K)
         GOTO 20
       END IF
       IF(J.NE.K) THEN
         NC(J)=K                 ! J and K now related
       ELSE
         DISTJK(I)=-DISTJK(I)-1.0  ! flag as removed
       END IF
      END DO
      DO J=1,N
30      IF(NC(J).NE.NC(NC(J))) THEN
         NC(J)=NC(NC(J))
        GOTO 30
       END IF
      END DO
      END
```

A.3 Bereichsuche (Range Search)

Das Problem bei der Bereichsuche ist, unter einer eventuell großen Zahl von Daten alle Elemente zu finden, die in einen bestimmten Größenbereich fallen. Beispiel sind Punkte in einer Ebene, von denen alle in einem gegebenen Rechteck liegenden Punkte auszuwählen sind. Der *tree*-Algorithmus erfordert in der Vorbereitung (`DFTREE`) den Aufbau einer Indexstruktur. Die Daten sind gegeben durch `NVEC` Vektoren der Dimension `NDIM`, die in dem Feld `VECT(NDIM,NVEC)` gespeichert sind. Von den `NDIM` Elementen eines Vektors werden `NX` bei der Bereichsuche benutzt ($\texttt{NX} \leq \texttt{NDIM}$); die Indizes der zu benutzenden Elemente werden in dem Feld `INX(NX)` angegeben, die Grenzen sind durch $\texttt{X(K)} \pm \texttt{DX(K)}$ gegeben. Bei der Vorbereitung der Bereichsuche durch `DFTREE` wird ein (integer) Indexarray `INDLR(2,NVEC)` definiert; der zweite Dimensionsparameter muss mindestens so groß sein wie die Anzahl `NVEC` der Vektoren.

Das Indexarray `INDLR(2,NVEC)` wird in der Bereichsuche benutzt, die entweder mit dem Programm `NCTREE` oder mit dem Programm `ITREE` erfolgen kann, mit denen bei einem Aufruf alle Vektoren oder jeweils ein Vektor (in einer Schleife) bestimmt wird.

Subprogram `dftree.F`

Listing A.7: Vorbereitung der Bereichsuche

```
      SUBROUTINE DFTREE(INX,NX,VECT,NDIM,NVEC,INDLR,NCDIM)
*
*     Pre-processing for multi-dimensional range search (INDLR(2,...))
*
*     The array  VECT(NDIM,NVEC)  is assumed to contain NVEC NDIM-dimensional
*     single precision key vectors. For example for NDIM=3 the first vector
*     is stored in elements VECT(1,1),VECT(2,1),VECT(3,1).
*
*     The integer array INX(NX) contains the indices of the NX used
*     vector elements.
*
*     The integer array INDLR(2,NVEC) (i.e. second dimension equal to
*     the number of  vectors) contains at return pointer information necessary
*     and to be used in the functions NCTREE and ITREE.
*
*     The range search results of the function NCTREE is stored in the
*     integer array INDC(.) in the common /CINDEX/. The default dimension
*     if the array INDC(.) is 100, sufficient for up to 100 indices found
*     in a range search. The user may use a larger INDC(.) array dimension
*     NCDIM, which has to be specified as the last argument in the call.
*
      INTEGER INX(NX),INDLR(2,NVEC)          ! index vector, tree pointer
      REAL VECT(NDIM,NVEC)                   ! table of keys
      LOGICAL LOWER
      PARAMETER (MINDC=100)                  ! default dimension of
      COMMON/CINDEX/NINDC,NTRY,ISTMAX,INDC(MINDC)  ! result index common
*     ...
      NINDC=MAX(NCDIM,MINDC)                 ! actual dimension
      DO N=1,NVEC                            ! reset
        INDLR(1,N)=0
        INDLR(2,N)=0
      END DO
      I=1                                    ! initialize tree
      DO N=2,NVEC                            ! loop N start
        L=1                                  ! 1. component first
        I=1                                  ! treeinsert
   10   J=I                                  ! parent
        LOWER=VECT(INX(L),N).LT.VECT(INX(L),I) ! compare keys
        IF (LOWER) THEN
          I=INDLR(1,I)
```

```
      ELSE
         I=INDLR(2,I)
      END IF
      L=MOD(L,NX)+1                    ! index of next component
      IF(I.NE.0) GOTO 10
      I=N                               ! add new key
      INDLR(1,I)=0
      INDLR(2,I)=0
      IF(LOWER) THEN
         INDLR(1,J)=I
      ELSE
         INDLR(2,J)=I
      END IF
     END DO                             ! end loop N
     END
```

Subprogram `nctree.F`

Listing A.8: Bereichsuche – Satz von Elementen

```
      FUNCTION NCTREE(X,DX, INX,NX,VECT,NDIM,NVEC,INDLR)
*     binary search tree - returns number NC, and the indices in common
*
*     A search is made for key vectors, which satisfy the conditions.
*     The arguments INX,NX, VECT,NDIM,NVEC, INDLR are as before in DFTREE.
*     The user has to specify the limits in the arrays X(NDIM),DX(NDIM).
*
*     The returned function value NC is the number of key vectors found,
*     satisfying the conditions given by the limits.
*     The range search results of the function NCTREE is stored in the
*     integer array INDC(.) in the common /CINDEX/. The default dimension
*     of the array INDC(.) is 100, sufficient for up to 100 indices found
*     in a range search. The user may define a larger INDC(.) array dimension
*     NCDIM in the common, which has to be specified as the last argument
*     in the call.
*     The indices of the found key vectors are in the array
*                         INDC(J), J = 1, NC
*     in the common /CINDC/.
*     Additional information on the range search in the common is:
*        NTRY   = number of test attempts (measures the efficiency)
*        ISTMAX = maximum length of the stack, used in the range search
*     The default dimension of the stack is 100, which should be sufficient
*     for almost all cases. The programs stops in case of a stack overflow.
*
      REAL X(*),DX(*),VECT(NDIM,NVEC)
      INTEGER INX(NX),INDLR(2,NVEC)          ! index vector, tree pointer
      PARAMETER                 (MINDC=100)  ! default dimension of
     COMMON/CINDEX/NINDC,NTRY,ISTMAX,INDC(MINDC)  ! result index common
      PARAMETER  (NSTACK=100)
      INTEGER KST(NSTACK),LST(NSTACK)        ! stack
      LOGICAL TXL,TXR
*     ...
      NTRY=0
      NC=0
      IST=1
      KST(IST)=1                             ! push
      LST(IST)=1                             ! 1. component
      ISTMAX=0
   10 I=KST(IST)                             ! pop element
      L=LST(IST)                             ! index
      IST=IST-1
      IF(I.EQ.0) STOP 'I is zero '
```

```
      TXL=    VECT(INX(L),I).GE.X(L)-DX(L)
      TXR=    VECT(INX(L),I).LE.X(L)+DX(L)
      NTRY=NTRY+1                        ! count number of attempts
      IF(TXL.AND.TXR) THEN
         DO K=1,NX                         ! check rectangle
           IF(ABS(VECT(INX(K),I)-X(K)).GT.DX(K)) GOTO 20
         END DO
         NC=NC+1                           ! count
         IF(NC.LE.NINDC) INDC(NC)=I        ! save index
      END IF
   20 IF(INDLR(2,I).NE.0.AND.TXR) THEN
         IST=IST+1
         IF(IST.GT.NSTACK) STOP 'NCTREE stack overflow'
         KST(IST)=INDLR(2,I)               ! push r
         LST(IST)=MOD(L,NX)+1              ! next component
         ISTMAX=MAX(ISTMAX,IST)
      END IF
      IF(INDLR(1,I).NE.0.AND.TXL) THEN
         IST=IST+1
         IF(IST.GT.NSTACK) STOP 'NCTREE stack overflow'
         KST(IST)=INDLR(1,I)               ! push l
         LST(IST)=MOD(L,NX)+1              ! next component
         ISTMAX=MAX(ISTMAX,IST)
      END IF
      IF(IST.NE.0) GOTO 10
      NCTREE=NC
      END
```

Subprogram `itree.F`

Listing A.9: Bereichsuche – ein Element

```
      INTEGER FUNCTION ITREE(X,DX,  INX,NX, VECT,NDIM,NVEC, INDLR)
*     binary search tree
*
*     A search is made for key vectors, which satisfy the conditions.
*     The arguments INX,NX, VECT,NDIM,NVEC, INDLR are as before in DFTREE.
*     The user has to specify the limits in the arrays X(NDIM),DX(NDIM).
*
*     The function returns the index for each found key vector.
*     The value zero is returned, if no further key vector is found.
*
      IMPLICIT NONE
      INTEGER NX,NDIM,NVEC,NSTACK,I,IST,ISTMAX,K,L
      REAL X(*),DX(*),VECT(NDIM,NVEC)
      INTEGER INX(NX),INDLR(2,NVEC)          ! index vector, tree pointer
      PARAMETER  (NSTACK=100)
      INTEGER KST(NSTACK),LST(NSTACK)        ! stack
      LOGICAL TXL,TXR
      DATA I/0/
      SAVE
*     ...
      IF(I.NE.0) GOTO 20
      IST=1
      KST(IST)=1                             ! push
      LST(IST)=1                             ! 1. component
      ISTMAX=0
   10 I=KST(IST)                             ! pop element
      L=LST(IST)                             ! index
      IST=IST-1
      IF(I.EQ.0) STOP 'I is zero '
      TXL=VECT(INX(L),I).GE.X(L)-DX(L)
```

```
      TXR=VECT(INX(L),I).LE.X(L)+DX(L)
      IF(TXL.AND.TXR) THEN
         DO K=1,NX                    ! check rectangle
           IF(ABS(VECT(INX(K),I)-X(K)).GT.DX(K)) GOTO 20
         END DO
         GOTO 30
      END IF
   20 IF(INDLR(2,I).NE.0.AND.TXR) THEN
         IST=IST+1
         IF(IST.GT.NSTACK) STOP 'ITREE stack overflow'
         KST(IST)=INDLR(2,I)          ! push r
         LST(IST)=MOD(L,NX)+1         ! next component
      END IF
      IF(INDLR(1,I).NE.0.AND.TXL) THEN
         IST=IST+1
         IF(IST.GT.NSTACK) STOP 'ITREE stack overflow'
         KST(IST)=INDLR(1,I)          ! push l
         LST(IST)=MOD(L,NX)+1         ! next component
      END IF
      IF(IST.NE.0) GOTO 10
      I=0
   30 ITREE=I                         ! no further elements
      END
```

Beispiel einer Bereichsuche. Ein Feld `VECT(3,1000000)` von Vektoren mit jeweils drei Komponenten wird mit $U(0,1)$-Zufallszahlen gefüllt. Gesucht werden sollen solche Vektoren, deren *erste* und *dritte* Komponenten in einem bestimmten kleinen Bereich liegen. Die Suche erfolgt zum einen mit dem Program `nctree`, bei dem alle Vektoren mit einem Aufruf gefunden werden, und zum anderen mit dem Program `itree` in einer Schleife.

In dem Beispiel werden 14 (von insgesamt 10^6) Vektoren gefunden, dazu werden nur etwa 50 Vergleiche (statt 10^6) gemacht. Es werden nur vier (von 100 vorgesehenen) Elementen des Stacks benutzt.

```
      REAL VECT(3,1000000),X(2),DX(2)
      INTEGER INX(2),INDLR(2,1000000)
      PARAMETER                                (MINDC=100)
      COMMON/CINDEX/NINDC,NTRY,ISTMAX,INDC(MINDC)
*
      CALL RANSHI(3000000,VECT)
      INX(1)=1
      INX(2)=3
      CALL DFTREE(INX,2,VECT,3,1000000,INDLR,100)

      X(1)=0.8
      DX(1)=0.002
      X(2)=0.5
      DX(2)=0.002
      NC=NCTREE(X,DX,INX,2, VECT,3,1000000, INDLR)
      WRITE(*,*) 'Using nctree:'
      WRITE(*,101) ' Range search in area',X(1)-DX(1),X(1)+DX(1)
      WRITE(*,101) '                    ',X(2)-DX(2),X(2)+DX(2)
  101 FORMAT(A,F9.5,' ... ',F9.5)
      WRITE(*,*) NC,' number of keys found'
      WRITE(*,*) NINDC,' dimension of index array INDC'
      WRITE(*,*) NTRY,' number of tests'
      WRITE(*,*) ISTMAX,' length of stack used'
      WRITE(*,*) '              ... in last search'
      WRITE(*,*) ' '
      DO I=1,NC
        WRITE(*,102) (VECT(K,INDC(I)),K=1,3)
```

```
      END DO
  102 FORMAT(10X,3F10.5)

      WRITE(*,*) '␣'
      WRITE(*,*) 'Using␣itree:'
   10 I=ITREE(X,DX, INX,2, VECT,3,1000000, INDLR)
      IF(I.EQ.0) STOP
      WRITE(*,102) (VECT(K,I),K=1,3)
      GOTO 10
      END
```

A.4 Programme zur linearen Algebra

Speicherung von Matrizen in FORTRAN. Das *spaltenweise* Speichern von Matrizen und Tabellen wird verwendet, also werden die Elemente `A(i,j)` des Beispiels auf Seite 38 oben in der Reihenfolge
`A(1,1)`, `A(2,1)`, `A(3,1)`, `A(1,2)`, `A(2,2)` und `A(3,2)` gespeichert (wobei sich der erste Index am schnellsten ändert); zum Beispiel ist das Element A_{22} im 5. Speicherplatz, wenn die Tabellen-Deklaration `REAL A(3,2)` ist. Wenn der erste Dimensionsparameter der Deklaration nicht mit der tatsächlichen Zeilenzahl der Matrix übereinstimmt, dann nehmen die Matrixelemente nicht aufeinanderfolgende Plätze im Speicher ein. Das ist wichtig zu wissen, wenn eine Matrix an ein Unterprogramm weitergegeben wird: Innerhalb der Subroutine müssen die Parameter der tatsächlichen (logischen) Dimensionen *und* wenigstens der erste Dimensionsparameter der Tabellendeklaration bekannt sein.

Speichern symmetrischer Matrizen. Für symmetrische Matrizen gibt es besondere Schemata, die die numerische Behandlung von Matrixoperationen effizienter gestalten. In dem folgenden Speicherschema für symmetrische Matrizen verwendet man eine eindimensionale Tabelle `V`, beschrieben durch `REAL V(NDIM)` mit `NDIM =` $(n^2+n)/2$, in der die Elemente der symmetrischen Matrix gespeichert werden; beispielsweise wird eine symmetrische 3×3 Matrix

$$\begin{pmatrix} V_{11} & V_{12} & V_{13} \\ V_{21} & V_{22} & V_{23} \\ V_{31} & V_{32} & V_{33} \end{pmatrix} \quad \text{gespeichert als} \quad \begin{pmatrix} \texttt{V(1)} & & \\ \texttt{V(2)} & \texttt{V(3)} & \\ \texttt{V(4)} & \texttt{V(5)} & \texttt{V(6)} \end{pmatrix} . \tag{A.1}$$

Das Element A_{ij} einer symmetrischen $n\times n$ Matrix ist das q-te Element der eindimensionalen Tabelle mit

$$q = \begin{cases} i + j(j-1)/2 & \text{für} \quad i \le j \\ j + i(i-1)/2 & \text{für} \quad i \ge j \end{cases} \tag{A.2}$$

oder $q = \min(i,j) + [\max(i,j)\,(\max(i,j)-1)]/2$. Man beachte, daß die Ausdrücke nicht von dem Dimensionsparameter n abhängen, infolgedessen ist es einfach, eine weitere Zeile und Spalte hinzuzufügen, und kein besonderes Argument ist notwendig, wenn man Matrix-Unterprogramme aufruft. Eine symmetrische $n\times n$ Matrix hat insgesamt $(n^2+n)/2$ gespeicherte Elemente.

Speichern von Bandmatrizen. Symmetrische Bandmatrizen speichert man auf eine andere Art. Bei einer symmetrischen $n\times n$ Bandmatrix $\boldsymbol{V}$ mit Bandbreite m gilt $V_{ij}=0$ für all i und j mit $|i-j|\ge m$. Eine solche symmetrischen Bandmatrix hat $(m+1)\cdot n - m\cdot(m+1)/2$ signifikante Elemente; sie kann in einer zweidimensionalen Anordnung `V` gespeichert werden, deklariert als `REAL V(M+1,N)` mit $(m+1)n$ Elementen. Das Element V_{ij} entspricht `V(1+|I-J|,I)`

(Diagonalelemente haben als ersten Index 1). Mit dieser Konvention wird eine symmetrische 4×4 Bandmatrix der Bandbreite $m = 1$

$$\begin{pmatrix} V_{11} & V_{12} & 0 & 0 \\ V_{21} & V_{22} & V_{23} & 0 \\ 0 & V_{32} & V_{33} & V_{34} \\ 0 & 0 & V_{43} & V_{44} \end{pmatrix}$$

gespeichert an den Stellen mit den Indizes

$$\begin{pmatrix} \texttt{V(1,1)} & & & \\ \texttt{V(2,1)} & \texttt{V(1,2)} & & \\ & \texttt{V(2,2)} & \texttt{V(1,3)} & \\ & & \texttt{V(2,3)} & \texttt{V(1,4)} \end{pmatrix}$$

Austauschschritt. Das folgende Programmstück führt einen Austauschschritt an einer $n \times m$ Matrix aus, wobei das Element A_{pq} als Pivotelement benutzt wird.

```
*     exchange with pivot element (p,q)
      PIVOTI=1.0/A(P,Q)
      DO I=1,N          ! rectangular rule
       IF(I.NE.P) THEN
         DO J=1,M
          IF(J.NE.Q) THEN
            A(I,J)=A(I,J)-PIVOTI*A(I,Q)*A(P,J)
          END IF
         END DO
       END IF
      END DO
      DO I=1,N          ! column
       IF(I.NE.P) A(I,Q)=PIVOTI*A(I,Q)
      END DO
      DO J=1,M  ! row
       IF(J.NE.Q) A(P,J)=-PIVOTI*A(P,J)
      END DO
      A(P,Q)=PIVOTI     ! pivot element
```

Inversion einer symmetrischen Matrix durch Austauschschritte.

Subprogram `smsinv.F`

Listing A.10: Inversion einer symmetrischen Matrix

```
      SUBROUTINE SMSINV(V,B,N,MFLAG,NRANK)
      IMPLICIT NONE
*     Inverse symmetric matrix and solution of system of linear  equations
*     V(*) = symmetric matrix of N cols and rows, replaced by inverse
*     B(N) = right hand side, for MFLAG .ne. 0 replaced by solution
*     B is a dummy argument for MFLAG .eq. 0 (inversion only)
*     NRANK is returned rank of matrix; only part of matrix is inverted
*     for NRANK < N (remaining elements set to 0)
*     Limitation to N <= NDIMLD!
      INTEGER NDIMLD
      REAL EPS
      PARAMETER (NDIMLD=100,EPS=1.E-6)
      INTEGER I,J,K,L,M,N,II,JJ,KK,JK,MFLAG,LK,NRANK
      REAL V(*),B(*),VKK,VJK,PIVOT,DIAG(NDIMLD)
      LOGICAL DONE(NDIMLD)
*     ...
      IF(N.GT.NDIMLD) STOP ' SMSINV: argument N too large '
```

```
      II=0
      DO I=1,N                ! reset flags
       DONE(I)=.FALSE.
        II=II+I
       DIAG(I)=ABS(V(II))
      END DO
*     loop
      NRANK=0
      DO I=1,N
*       search for pivot and test for linearity and zero matrix
       K  =0
       VKK=0.0
       JJ=0
       DO J=1,N
         JJ=JJ+J
        IF(.NOT.DONE(J)) THEN
            IF(K.EQ.0) K=J
            IF(ABS(V(JJ)).GE.MAX(VKK,EPS*DIAG(J))) THEN
               VKK=ABS(V(JJ))
               K  =J
            END IF
         END IF
       END DO
       DONE(K)=.TRUE.
*      KK is previous diagonal element
       KK=(K*K-K)/2
       IF(VKK.NE.0.0) THEN
*         preparation for exchange step
          NRANK=NRANK+1
          PIVOT=1.0/V(KK+K)
          V(KK+K)=-PIVOT
          IF(MFLAG.NE.0) B(K)=PIVOT*B(K)
          JJ=0
*         exchange step
          JK=KK
          DO J=1,N
            IF(J.LE.K) JK=JK+1
            IF(J.GT.K) JK=JK+J-1
            IF(J.NE.K) THEN
               VJK  =V(JK)
               V(JK)=PIVOT*VJK
               IF(MFLAG.NE.0) B(J) =B(J)-B(K)*VJK
*              index of pivot row/col
               LK=KK
               DO L=1,J
                 IF(L.LE.K) LK=LK+1
                 IF(L.GT.K) LK=LK+L-1
                 IF(L.NE.K) THEN
                    V(JJ+L)=V(JJ+L)-V(LK)*VJK
                 END IF
               END DO
             END IF
             JJ=JJ+J
          END DO
       ELSE
*         no pivot found, clear
          JK=KK
          DO J=1,N
            IF(J.LE.K) JK=JK+1
            IF(J.GT.K) JK=JK+J-1
           V(JK)=0.0
          END DO
```

```
      END IF
     END DO
*    change sign
     DO I=1,(N*N+N)/2
      V(I)=-V(I)
     END DO
     END
```

LR-Zerlegung einer quadratischen $N \times N$ Matrix.

Subprogram `gnlrd.F`

Listing A.11: LR-Matrix-Zerlegung

```
      SUBROUTINE GNLRD(A,N,NDIM1,IDX)
*     LR decomposition of general N*N quadratic matrix in A(NDIM1,N)
*     with result overwriting matrix A.
*     IDX(N) is an array, returned with information about the order,
*     with IDX(1)=0 in case of singular matrix.
*     Limitation to N <= NDIMP!
      INTEGER N,IDX(N),NDIM1,NDIMP,I,J,K,IMAX
      PARAMETER (NDIMP=100)
      REAL A(NDIM1,N),SCALE(NDIMP),ELMAX,TMP,SUM
      IF(N.GT.NDIMLD) STOP ' GNLRD: argument N too large '
     DO I=1,N
      ELMAX=0.0
      DO J=1,N
       IF(ABS(A(I,J)).GT.ELMAX) ELMAX=ABS(A(I,I))
      END DO
      IF(ELMAX.EQ.0.0) THEN
          IDX(1)=0 ! matrix singular
          RETURN
      END IF
      SCALE(I)=1.0/ELMAX
     END DO
     DO J=1,N
      DO I=1,J-1
       SUM=A(I,J)
       DO K=1,I-1
        SUM=SUM-A(I,K)*A(K,J)
       END DO
       A(I,J)=SUM
      END DO
      ELMAX=0.0
      DO I=J,N
       SUM=A(I,J)
       DO K=1,J-1
        SUM=SUM-A(I,K)*A(K,J)
       END DO
       A(I,J)=SUM
       TMP=SCALE(I)*ABS(SUM)
       IF(TMP.GE.ELMAX) THEN
           IMAX=I
           ELMAX=TMP
       END IF
      END DO
      IF(J.NE.IMAX) THEN
          DO K=1,N
           TMP=A(IMAX,K)
           A(IMAX,K)=A(J,K)
           A(J,K)=TMP
          END DO
          SCALE(IMAX)=SCALE(J)
```

```
      END IF
      IDX(J)=IMAX
      IF(J.NE.N) THEN
        TMP=1.0/A(J,J)
        DO I=J+1,N
          A(I,J)=A(I,J)*TMP
        END DO
      END IF
    END DO
    END
```

Subprogram `gnsol.F`

Listing A.12: Lösung eines Gleichungssystems durch LR-Zerlegung

```
      SUBROUTINE GNSOL(A,N,NDIM1,IDX,B)
*     Solution of matrix equation using LR decomposition, with
*     arrays A and IDX from previous call to GNLRD.
*     B(N) is array of right-hand side, replaced by result.
      REAL A(NDIM1,N),B(N),SUM
      INTEGER IDX(N),NDIM1,N,I,J,K
      DO I=1,N
        K=IDX(I)
        SUM=B(K)
        B(K)=B(I)
        DO J=1,I-1
          SUM=SUM-A(I,J)*B(J)
        END DO
        B(I)=SUM
      END DO
      DO I=N,1,-1
        SUM=B(I)
        DO J=I+1,N
          SUM=SUM-A(I,J)*B(J)
        END DO
        B(I)=SUM/A(I,I)
      END DO
      END
```

Subprogram `gninv.F`

Listing A.13: Matrixinversion durch LR-Zerlegung

```
      SUBROUTINE GNINV(A,AINV,N,NDIM1,IDX)
*     Inverse of matrix AINV returned, using LR decomposition with
*     arrays A and IDX from previous call to GNLRD.
      REAL A(NDIM1,N),AINV(NDIM1,N)
      INTEGER IDX(N),I,J,N,NDIM1
      DO J=1,N
        DO I=1,N
          AINV(I,J)=0.0
        END DO
        AINV(J,J)=1.0
      END DO
      DO J=1,N
        CALL GNSOL(A,N,NDIM1,IDX,AINV(1,J))
      END DO
      END
```

Beispiel A.1 Matrixinversion einer 3×3 Matrix.

Das Programmstück zeigt eine 3×3 Matrix; es invertiert die Matrix unter Anwendung der ***LR***-Zerlegung und unter Benutzung der Subroutinen GNLRD und GNINV.

```
      REAL A(3,3),B(3,3)
      INTEGER IDX(3)
      DATA A/0.0,3.0,5.0,1.0,0.0,3.0,2.0,2.0,1.0/
      CALL GNLRD(A,3,3,IDX)
      CALL GNINV(A,B,3,3,IDX)
```

Die ursprüngliche Matrix und ihre inverse Matrix sind demnach:

$$\boldsymbol{A} = \begin{pmatrix} 0 & 1 & 2 \\ 3 & 0 & 2 \\ 5 & 3 & 1 \end{pmatrix} \qquad \boldsymbol{A}^{-1} = \begin{pmatrix} -0.24 & 0.2 & 0.08 \\ 0.28 & -0.4 & 0.24 \\ 0.36 & 0.2 & -0.12 \end{pmatrix}$$

Die ersten beiden Diagonalelemente der Matrix $\boldsymbol{A}$ sind null. Damit ist ein Umordnen in der Subroutine `GNLRD` nötig.

Bandmatrix. Das folgende Programm löst die Gleichung $Wx = b$, wobei W eine symmetrische positiv-semidefinite Bandmatrix der Breite NBAND ist. Die LR-Zerlegung, bei der die ursprüngliche Matrix überschrieben wird, wird zur Berechnung der Lösung benutzt, die inverse Matrix (volle Matrix) wird nicht berechnet. Das Pivotelement wird fortlaufend auf der Diagonalen gewählt. Bei zu kleinem Pivotelement werden die entsprechenden Elemente der Lösung zu null gesetzt.

Subprogram `bmsol.F`

Listing A.14: Lösung eines Gleichungssystems mit Bandmatrix

```
      SUBROUTINE BMSOL(W,B,NBAND,NROW)
*     Solution of band matrix equation W * X = B
*     with symmetric band matrix in special storage mode in array W.
*     Array B(NROW) is right-hand side, replaced by result.
      INTEGER I,J,N,NN,IMAX,JMAX,NDIM,NBAND,NROW
      PARAMETER (NDIM=100)
      REAL W(NBAND,NROW),B(NROW),DIAG(NDIM),RATIO
      DO N=1,NROW
       DIAG(N)=W(1,N)
      END DO
*     decomposition
      DO N=1,NROW
       IF (W(1,N)+DIAG(N).LE.DIAG(N)) THEN
*         singular case
          DO J=1,NBAND
           W(J,N)=0.0
          END DO
       ELSE
          W(1,N)=1.0/W(1,N)
          IMAX=MIN(NBAND-1,NROW-N)
          JMAX=IMAX
          DO I=1,IMAX
           RATIO=W(I+1,N)*W(1,N)
           DO J=1,JMAX
            W(J,N+I)=W(J,N+I)-W(J+I,N)*RATIO
           END DO
           JMAX=JMAX-1
           W(I+1,N)=RATIO
          END DO
       END IF
      END DO
*     solution by forward ...
```

```
      DO N=1,NROW
       DO J=1,MIN(NBAND-1,NROW-N)
        B(J+N)=B(J+N)-W(J+1,N)*B(N)
       END DO
      END DO
*     ... and backward substitution
      DO NN=1,NROW
       N=NROW+1-NN
       B(N)=B(N)*W(1,N)
       DO J=1,MIN(NBAND-1,NROW-N)
        B(N)=B(N)-W(J+1,N)*B(J+N)
       END DO
      END DO
      END
```

Cholesky-Zerlegung. In den folgenden Unterprogrammen für eine positiv-definite symmetrische Matrix ist die Cholesky-Zerlegung implementiert. Die Berechnung wird bei Auftreten eines verschwindenden Pivotelements abgebrochen.

Subprogram `chlrd.F`

Listing A.15: Cholesky-Zerlegung einer symmetrischen Matrix

```
      SUBROUTINE CHLRD(W,N)
*     Cholesky decomposition of symmetric N*N matrix W,
*     overwritten by result of decomposition.
*     W(1)=0 returned in case of non-positive diagonal element
      REAL W(*)
      II=0
      DO I=1,N
       II=II+I
       IF(W(II).LE.0.0) THEN
*         non-positive diagonal element
          W(1)=0.0
          RETURN
       END IF
       W(II)=SQRT(W(II))        ! diagonal element of R
       DO J=I+1,N
        IJ=I+(J*J-J)/2
        W(IJ)=W(IJ)/W(II)       ! other elements of R
       END DO
       DO J=I+1,N
        IJ=I+(J*J-J)/2
        DO K=J,N
         JK=J+(K*K-K)/2
         IK=I+(K*K-K)/2
         W(JK)=W(JK)-W(IJ)*W(IK)
        END DO
       END DO
      END DO
      END
```

Subprogram `chsol.F`

Listing A.16: Lösung eines Gleichungssystems mit Cholesky-Zerlegung

```
      SUBROUTINE CHSOL(W,N,B)
*     Solution of matrix equation using LR decomposition, with
*     arrays W from previous call to CHLRD.
*     B(N) is array of right-hand side, replaced by result.
      REAL W(*),B(N)
```

```
      DO I=1,N        ! forward solution
       SUM=B(I)
       DO J=1,I-1
        SUM=SUM-W(J+(I*I-I)/2)*B(J)
       END DO
       B(I)=SUM/W((I*I+I)/2)
      END DO
      DO I=N,1,-1     ! backward solution
       SUM=B(I)
       DO J=I+1,N
        SUM=SUM-W(I+(J*J-J)/2)*B(J)
       END DO
       B(I)=SUM/W((I*I+I)/2)
      END DO
      END
```

Bandmatrix-Operationen.

Die Cholesky-Zerlegung bietet insbesondere für Operationen mit Band-Matrizen große Vorteile, sowohl in Hinblick auf die Rechenzeit als auch beim Platzbedarf zur Speicherung. Bei der Cholesky-Zerlegung bleibt die Bandstruktur erhalten, und die Zerlegung kann den gleichen Platz wie die ursprüngliche Matrix einnehmen. Die Zahl der Rechenoperationen für die Zerlegung ist proportional zu $m^2 \cdot n$ and damit (bei festem m) *linear* in n (dagegen proportional zu n^3 für allgemeine Matrizen). Zum Beispiel für eine Bandmatrix mit $n = 100$ und $m = 2$ ist die Cholesky-Zerlegung um den Faktor 1000 schneller als Standard-Methoden.

Die Inverse einer Bandmatrix ist eine volle Matrix, deren Berechnung entsprechend der Zahl der Elemente aufwändig ist. Oft werden nur die Elemente der Inversen in und um die Diagonale herum benötigt. Diese Elemente der Inversen entsprechend der Bandstruktur lassen sich nach der Zerlegung mit einem Aufwand proportional zu n berechnen[35, 46, 2].

Die Zerlegung der Bandmatrix $\boldsymbol{W}$ in `DBCDEC` entspricht dem Produkt $\boldsymbol{R}^{\mathrm{T}}\boldsymbol{D}\boldsymbol{R}$; dabei werden keine Quadratwurzeln benötigt. Die Elemente der Diagonalmatrix $\boldsymbol{D}$ werden in den Diagonalelementen der Matrix $\boldsymbol{W}$ gespeichert. Die Diagonalelemente der Matrix $\boldsymbol{R}$ sind gleich 1 und werden nicht gespeichert. Matrizen und Vektoren sind in `DBCDEC` als `DOUBLE PRECISION` vereinbart.

Das Program `DBCDEC` zur Cholesky-Zerlegung bietet auch Operationen (1) zur Lösung eines Gleichungssystems, (2) zur Bestimmung der Elemente der Inversen im Band der Matrix, und (3) die Bestimmung der vollen Inversen. Matrizen und Vektoren sind in `DBCDEC` als `DOUBLE PRECISION` vereinbart.

Subprogram `dbcdec.F`

Listing A.17: Bandmatrix-Operationen

```
      SUBROUTINE DBCDEC(W,MP1,N, AUX)   ! decomposition, bandwidth M
*     Symmetric band matrix: decomposition, solution, complete
*                                 inverse and band part of the inverse
*     M=MP1-1                                        N*M(M-1) dot operations
      DOUBLE PRECISION W(MP1,N),V(MP1,N),B(N),X(N),AUX(N),VS(*),RXW
*     ...
      DO I=1,N
       AUX(I)=16.0D0*W(1,I) ! save diagonal elements
      END DO
      DO I=1,N
       IF(W(1,I)+AUX(I).NE.AUX(I)) THEN
          W(1,I)=1.0/W(1,I)
       ELSE
          W(1,I)=0.0D0  ! singular
       END IF
```

```
      DO J=1,MIN(MP1-1,N-I)
       RXW=W(J+1,I)*W(1,I)
       DO K=1,MIN(MP1-1,N-I)+1-J
        W(K,I+J)=W(K,I+J)-W(K+J,I)*RXW
       END DO
       W(J+1,I)=RXW
      END DO
     END DO
     RETURN

     ENTRY DBCSLV(W,MP1,N,B, X)  ! solution B -> X
*                                          N*(2M-1) dot operations
     DO I=1,N
      X(I)=B(I)
     END DO
     DO I=1,N ! forward substitution
      DO J=1,MIN(MP1-1,N-I)
       X(J+I)=X(J+I)-W(J+1,I)*X(I)
      END DO
     END DO
     DO I=N,1,-1 ! backward substitution
      RXW=X(I)*W(1,I)
      DO J=1,MIN(MP1-1,N-I)
       RXW=RXW-W(J+1,I)*X(J+I)
      END DO
      X(I)=RXW
     END DO
     RETURN

     ENTRY DBCIEL(W,MP1,N, V)    ! V = inverse band matrix elements
*                                        N*M*(M-1) dot operations
     DO I=N,1,-1
      RXW=W(1,I)
      DO J=I,MAX(1,I-MP1+1),-1
       DO K=J+1,MIN(N,J+MP1-1)
        RXW=RXW-V(1+ABS(I-K),MIN(I,K))*W(1+K-J,J)
       END DO
       V(1+I-J,J)=RXW
       RXW=0.0D0
      END DO
     END DO
     RETURN

     ENTRY DBCINV(W,MP1,N, VS)    ! V = inverse symmetric matrix
*                                       N*N/2*(M-1) dot operations
     DO I=N,1,-1
      RXW=W(1,I)
      DO J=I,1,-1
       DO K=J+1,MIN(N,J+MP1-1)
        RXW=RXW-VS((MAX(I,K)*(MAX(I,K)-1))/2+MIN(I,K))*W(1+K-J,J)
       END DO
       VS((I*I-I)/2+J)=RXW
       RXW=0.0D0
      END DO
     END DO
     END
```

Bestimmung der Eigenwerte und der Eigenvektoren einer symmetrischen $N \times N$-Matrix. Subprogram `smjrot.F`

Listing A.18: Eigenwerte und der Eigenvektoren einer symmetrischen Matrix

```
      SUBROUTINE SMJROT(V,A,N,NDIM1,DIAG,NROT)
*     Eigenvalues and eigenvectors of symmetric N*N matrix V(*), which
*     is destroyed during computation.
*     Returned: the N*N matrix of eigenvectors in array A(NDIM1,N)
*     and the diagonal elements (eigenvalues) in array DIAG(N).
*     NROT is the performed number of rotations
*     Limitation to N <= NDD!
      INTEGER NDD
      PARAMETER (NDD=100)
      REAL V(*),A(NDIM1,N),DIAG(N),DDAG(NDD)
      REAL AIP,AIQ,VIP,VIQ,AM,SM,DIFF,EPSFAC,TEST
      REAL COSPHI,SINPHI,TANPHI,TANPH2,THETA
      INTEGER I,J,N,II,IJ,P,Q,PQ,NDIM1,NROT,MSY,MDI,ITER
      PARAMETER (EPSFAC=100.0)
*     indices in symmetric matrix
      MSY(I,J)=MIN(I,J)+(MAX(I,J)*MAX(I,J)-MAX(I,J))/2
      MDI(I)  =(I*I+I)/2
      NROT=0
      DO J=1,N
       DO I=1,N
        A(I,J)=0.0
       END DO
       A(J,J)=1.0 ! A = unit matrix
      END DO
      II=0
      DO I=1,N
       II=II+I
       DIAG(I)=V(II) ! copy diagonal elements to DIAG
       IF(I.LE.NDD) DDAG(I)=0.0
      END DO
      DO ITER=-N,50    ! iterations
       SM=0.0
       AM=0.0
       IJ=0
       DO I=1,N
        DO J=1,I-1
         IJ=IJ+1
         SM=SM+ABS(V(IJ))                    ! sum
         IF(ABS(V(IJ)).GT.AM) AM=ABS(V(IJ)) ! maximum
        END DO
        IJ=IJ+1
       END DO
       IF(SM.EQ.0.0) RETURN  ! test convergence
       PQ=0
       DO Q=1,N
        DO P=1,Q-1
         PQ=PQ+1
         IF(ITER.GE.0.OR.ABS(V(PQ)).EQ.AM) THEN
            TEST=EPSFAC*ABS(V(PQ))
            IF(ITER.LT.0.OR.
     +         ABS(DIAG(P)).NE.ABS(DIAG(P))+TEST.OR.
     +         ABS(DIAG(Q)).NE.ABS(DIAG(Q))+TEST) THEN
              NROT=NROT+1          ! begin/Jacobi rotation
              DIFF=DIAG(Q)-DIAG(P)
              IF(ABS(DIFF).EQ.ABS(DIFF)+TEST) THEN
                 TANPHI=V(PQ)/DIFF
              ELSE
                 THETA =0.5*DIFF/V(PQ)
                 TANPHI=1.0/(THETA+SIGN(SQRT(1.0+THETA*THETA),THETA))
              END IF
              COSPHI=1.0/SQRT(1.0+TANPHI*TANPHI)
              SINPHI=COSPHI*TANPHI
```

```
            TANPH2=SINPHI/(1.0+COSPHI)
            DO I=1,N
              IF(I.NE.Q.AND.I.NE.P) THEN
                 VIP=V(MSY(I,P)) ! transform symmetric
                 VIQ=V(MSY(I,Q)) !  matrix
                 V(MSY(I,P))=VIP-SINPHI*(VIQ+TANPH2*VIP)
                 V(MSY(I,Q))=VIQ+SINPHI*(VIP-TANPH2*VIQ)
              END IF
              AIP=A(I,P) ! update rotation
              AIQ=A(I,Q) !  matrix
              A(I,P)=AIP-SINPHI*(AIQ+TANPH2*AIP)
              A(I,Q)=AIQ+SINPHI*(AIP-TANPH2*AIQ)
            END DO
            DIAG(P)=DIAG(P)-TANPHI*V(PQ) ! update special elements
            DIAG(Q)=DIAG(Q)+TANPHI*V(PQ) !  of symmetric matrix
            IF(P.LE.NDD) DDAG(P)=DDAG(P)-TANPHI*V(PQ)
            IF(Q.LE.NDD) DDAG(Q)=DDAG(Q)+TANPHI*V(PQ)
          END IF
          V(PQ)=0.0   ! off-diagonal element becomes zero
        END IF        ! end/Jacobi rotation
      END DO
      PQ=PQ+1
    END DO
    II=0
    DO I=1,MIN(N,NDD)
      II=II+I
      V(II)=V(II)+DDAG(I) ! improve diagonal elements
      DIAG(I)=V(II)
      DDAG(I)=0.0
    END DO
  END DO
  WRITE(*,*) 'SMJROT: no convergence - stop'
  STOP
  END
```

Beispiel: Diagonalisierung einer 4×4 Matrix.

Das folgende Programmstück definiert eine 4×4 Matrix, und führt die Diagonalisierung durch (siehe Kapitel 3.2).

```
REAL V(10),DIAG(4),U(4,4)
DATA V/4.0,1.0,2.0,-2.0,0.0,3.0,2.0,1.0,-2.0,-1.0/
CALL SMJROT(V,U,4,4,DIAG,NROT)
```

Die Eingabematrix $\boldsymbol{V}$ und die Rotationsmatrix $\boldsymbol{U}$ sind die folgenden:

$$\boldsymbol{V} = \begin{pmatrix} 4 & 1 & -2 & 2 \\ 1 & 2 & 0 & 1 \\ -2 & 0 & 3 & -2 \\ 2 & 1 & -2 & -1 \end{pmatrix} \qquad \boldsymbol{U} = \begin{pmatrix} 0.7180 & 0.2212 & -0.5574 & 0.3534 \\ 0.2017 & 0.7894 & 0.5796 & 0.0103 \\ 0.6423 & -0.5442 & 0.5202 & -0.1440 \\ -0.1767 & -0.1781 & 0.2877 & 0.9243 \end{pmatrix}$$

und die Eigenwerte sind

$$6.844621 \quad 2.268531 \quad 1.084364 \quad -2.197519\,.$$

A.5 Generatoren für Zufallszahlen

Ein einfacher Zufallszahlengenerator. Nachfolgend ist ein einfacher Zufallszahlengenerator abgedruckt. Er erzeugt eine reelle Zahl zwischen 0 und 1 (ausschließlich). Die ganze Zahl LAST muss mit der Saatzahl initialisiert und gespeichert werden.

Subprogram `urand.F`

Listing A.19: Einfacher $U(0,1)$ Zufallszahlengenerator

```
      FUNCTION URAND()   ! return random number U(0,1)
*     (simple generator, showing principle)
      PARAMETER (IA=205,IC=29573,IM=139968)
      DATA LAST/4711/
      LAST=MOD(IA*LAST+IC,IM)
      IF(LAST.EQ.0) LAST=MOD(IA*LAST+IC,IM)
      URAND=FLOAT(LAST)/FLOAT(IM)
      END
```

Gutbrod's Generator. Die folgende FORTRAN Programmversion von `RANSHI`, die gegenüber der Originalversion von Gutbrod leicht verändert ist, enthält auch die notwendige Initialisierung (`RANVIN`). Das Unterprogramm erzeugt bei einem Aufruf `N` Zufallszahlen, die in einem Feld `A(N)` zurückgegeben werden; die Argumente von `RANSHI` sind die Anzahl `N` und der Feldname `A`. Ein Aufruf `CALL RANVIN(JSEED,JWARM)` gestattet es, den Generator (*vor* dem ersten Aufruf von `RANSHI`) mit einer definierten Saatzahl `JSEED` zu initialisieren. Dies ist zum Beispiel notwendig, wenn man viele unabhängig und parallel laufende Monte Carlo-Prozesse hat. Für eine gute Anfangsdurchmischung werden in der Initialisierung `JWARM` Durchläufe gemacht. Wenn `RANVIN` nicht gerufen wird, führt das Programm automatisch eine Initialisierung aus. Soll eine Rechung in einem nachfolgenden Programmlauf fortgesetzt werden, so sollte erst `RANVIN` gerufen werden und dann der Inhalt des Commons `/RANBUF/` vom vorhergehenden Programmlauf eingesetzt werden.

Subprogram `ranshi.F`

Listing A.20: Generatoren für Feld von $U(0,1)$ Zufallszahlen

```
      SUBROUTINE RANSHI(N,A)
*     return N random numbers U(0,1) in array A(N)
      PARAMETER (NB=511)
      PARAMETER (IA=16807,IM=2147483647,IQ=127773,IR=2836)
      PARAMETER (AEPS=1.0E-10,SCALIN=4.6566125E-10)
      COMMON/RANBUF/MBUFF(0:NB),IAN,IC,IBOOST
      REAL A(*)
      INTEGER ISTART
      IROTOR(M,N)=IEOR(IOR(ISHFT(M,17),ISHFT(M,-15)),N)
      DATA ISTART/0/,IWARM/10/,ISEED/4711/
      IF(ISTART.NE.0) GOTO 20
      WRITE(*,*) ' Automatic RANSHI initialization using:'
*     initialize buffer
 10   IDUM=ISEED+9876543          ! prevent damage, if iseed=0
      WRITE(*,*) '           ISEED=',ISEED,'   IWARM=',IWARM
      DO J=0,NB+1                 ! fill buffer
        K=IDUM/IQ                 ! minimal standard generator
        IDUM=IA*(IDUM-K*IQ)-IR*K  !    with Schrages method
        IF(IDUM.LT.0) IDUM=IDUM+IM !
        MBUFF(J)=ISHFT(IDUM,1)    ! fill in leading bit
      END DO
      IAN=IAND(IAN,NB)            ! mask angle
      IC=1                        ! set pointer
      IBOOST=0
      DO J=1,IWARM*NB             ! warm up a few times
        IT=MBUFF(IAN)             ! hit ball angle
        MBUFF(IAN)=IROTOR(IT,IC)  ! new spin
        IC=IT                     ! replace red spin
        IAN=IAND(IT+IBOOST,NB)    ! boost and mask angle
        IBOOST=IBOOST+1           ! increment boost
      END DO
      IF(ISTART.LT.0) RETURN      ! return for RBNVIN
      ISTART=1                    ! set done-flag
```

```
*     generate array of r.n.
 20   DO I=1,N
       IT=MBUFF(IAN)                ! hit ball angle
       MBUFF(IAN)=IROTOR(IT,IC)     ! new spin
       IC=IT                        ! replace red spin
       IAN=IAND(IT+IBOOST,NB)       ! boost and mask angle
       A(I)=FLOAT(ISHFT(IT,-1))*SCALIN+AEPS ! avoid zero output
       IBOOST=IBOOST+1              ! increment boost
      END DO
      IBOOST=IAND(IBOOST,NB)
      RETURN
      ENTRY RANVIN(JSEED,JWARM)    ! initialize, but only once
      IF(ISTART.EQ.0) THEN
         WRITE(*,*) ' RANSHI initialization by RANVIN-call using:'
         ISEED=JSEED               ! copy seed and
         IWARM=JWARM               !    warm-up parameter
         ISTART=-1                 ! start flag
         GOTO 10
      END IF
      END
```

Im Gegensatz zu üblichen Zufallszahlengeneratoren, die als Funktion geschrieben sind und bei einem Aufruf als Funktionswert *eine* Zufallszahl ergeben, gibt das Unterprogramm `RANSHI` bei einem Aufruf viele Zufallszahlen zur Vermeidung der Zeitverluste durch vielfachen *call/return*. Die folgende kurze Funktion `RANU` wird in nachfolgenden Beispielen gerufen und benutzt intern das Unterprogramm `RANSHI`. Diese Funktion vermeidet natürlich nicht den erwähnten Zeitverlust, ist aber einfach zu benutzen.

Subprogram `ranu.F`

Listing A.21: $U(0,1)$ Zufallszahlen

```
      FUNCTION RANU()
*     random number U(0,1) using RANSHI
      PARAMETER (NDIM=100)
      REAL BUFFER(NDIM)
      DATA INDEX/NDIM/
      SAVE INDEX,BUFFER
      INDEX=MOD(INDEX,NDIM)+1
      IF(INDEX.EQ.1) CALL RANSHI(NDIM,BUFFER)
      RANU=BUFFER(INDEX)
      END
```

Zufallszahlen nach einer Poissonverteilung.

Subprogram `mcpoi.F`

Listing A.22: Zufallszahlen der Poisson-Verteilung

```
      FUNCTION MCPOI(XMU) ! return random integer 0 ...
*     from Poisson distribution with mean XMU
*     (for large XMU slow,  use normal approximation)
      DATA ARG/0.0/
      SAVE EXPM,ARG
      IF(XMU.NE.ARG) THEN
         ARG =XMU  ! mean value
         EXPM=EXP(-ARG)
      END IF
      K=0
      A=1.0
   10 U=RANU()    ! U(0,1) random number
      A=A*U
```

```
      IF(A.GE.EXPM) THEN
        K=K+1
        GOTO 10
      END IF
      MCPOI=K ! K follows Poisson distribution
      END
```

Zufallszahlen nach einer Binominalverteilung. Die folgende Routine erzeugt Zufallszahlen k gemäß einer Binomialverteilung mit n Versuchen, wobei das Ereignis mit Einzelwahrscheinlichkeit p genau k mal auftritt.

Subprogram `mcbin.F`

Listing A.23: Zufallszahlen der Binomial-Verteilung

```
      FUNCTION MCBIN(N,P) ! return random integer 0 ... N
*     from Binomial distribution (N,p)
      PP=MAX(P,1.0-P) ! P is probability
      K=0
      DO I=1,N          ! N is number of trials
       IF(RANU().LT.PP) K=K+1
      END DO
      MCBIN=K
      IF(P.LT.0.5) MCBIN=N-K
      END
```

A.6 Programme für Spline-Funktionen

Koeffizienten für Spline-Funktion. Das Unterprogramm CSPLN berechnet für Daten der Form (10.1) alle Koeffizienten der Spline-Funktion und speichert sie in einem Feld `C(5,N)` ab. Die allgemeine Bedingung (4) von Kapitel 10 wird benutzt.

Subprogram `cspln.F`

Listing A.24: Koeffizienten für interpolierende Spline-Funktion

```
      SUBROUTINE CSPLN(X,Y,C,N)
*     Calculation of spline coefficients from N pairs x,y in
*     arrays X(N) and Y(N) (array X in increasing order), using the
*     not-a-knot end conditions.
*     The array C, dimensioned C(5,N), contains after return
*        C(1,I) = F(X(I)) = Y(I)
*        C(2,I) = first derivative F'(X(I))
*        C(3,I) = second derivative F''(X(I))/2.0
*        C(4,I) = third derivative F'''(X(I))/6.0
*        C(5,I) = X(I)
*     and can be used for interpolation.
*     Calculation of interpolated value for given X
*        with X(I) =< X =< X(I+1):
*     H=X-C(5,I)
*     F=C(1,I)+H*(C(2,I)+H*(C(3,I)+H*C(4,I)))
*
      REAL C(5,*),X(*),Y(*)
      IF(N.LT.3) RETURN
      DO I=1,N
       C(1,I)=Y(I)
       C(5,I)=X(I)
      END DO
*     first differences of x and first divided differences of data
      DO I=2,N
       C(3,I)= C(5,I)-C(5,I-1)
```

```
      C(4,I)=(C(1,I)-C(1,I-1))/C(3,I)
      END DO
*     not-a-knot at left side
      C(4,1)=C(3,3)
      C(3,1)=C(3,2)+C(3,3)
      C(2,1)=((C(3,2)+2.0*C(3,1))*C(4,2)*C(3,3)+C(3,2)**2*C(4,3))/C(3,1)
*     generate equations and carry out forward pass
      DO I=2,N-1
       G=-C(3,I+1)/C(4,I-1)
       C(2,I)=G*C(2,I-1)+3.0*(C(3,I)*C(4,I+1)+C(3,I+1)*C(4,I))
       C(4,I)=G*C(3,I-1)+2.0*(C(3,I)+C(3,I+1))
      END DO
*     not-a-knot at right side
      IF(N.EQ.3) THEN
         C(2,N)=2.0*C(4,N)
         C(4,N)=1.0
         G=-1.0/C(4,N-1)
      ELSE
         G=C(3,N-1)+C(3,N)
         C(2,N)=((C(3,N)+2.0*G)*C(4,N)*C(3,N-1)+C(3,N)**2*(C(1,N-1)
     +           -C(1,N-2))/C(3,N-1))/G
         G=-G/C(4,N-1)
         C(4,N)=C(3,N-1)
      END IF
*     complete forward pass
      C(4,N)=C(4,N)+G*C(3,N-1)
      C(2,N)=(G*C(2,N-1)+C(2,N))/C(4,N)
*     backward substitution
      DO I=N-1,1,-1
       C(2,I)=(C(2,I)-C(3,I)*C(2,I+1))/C(4,I)
      END DO
*     generate coefficients
      DO I=2,N
       DX=C(3,I)
       DVD1=(C(1,I)-C(1,I-1))/DX
       DVD3=C(2,I-1)+C(2,I)-2.0*DVD1
       C(3,I-1)=(DVD1-C(2,I-1)-DVD3)/DX
       C(4,I-1)=(DVD3/DX)/DX
      END DO
*     calculation of coefficients at x(n) (for completeness only)
      DX=C(5,N)-C(5,N-1)
      C(2,N)=C(2,N-1)+DX*(2.0*C(3,N-1)+DX*3.0*C(4,N-1))
      C(3,N)=C(3,N-1)+DX*3.0*C(4,N-1)
      C(4,N)=C(4,N-1)
      END
```

Funktionswerte. Die folgende Funktion berechnet den Funktionswert der Spline-Funktion für ein gegebenes Argument x. Das Feld `C(5,N)` enthält die in Unterprogramm `CSPLN` bestimmten Koeffizienten. Wenn x außerhalb des Intervalls liegt, erfolgt Extrapolation aus dem ersten bzw. letzten Unterintervall.

Subprogram `cspf.F`

Listing A.25: Funktionswert der interpolierenden Spline-Funktion

```
      FUNCTION CSPF(X,C,N)
*     Calculate function value for given argument X, for spline
*     function in array C, dimensioned C(5,N), obtained by previous
*     call of CSPLN.
      REAL C(5,N)
      DATA I/1/
      IF(I.GE.N) I=(N+1)/2
```

```
      IF(X.GE.C(5,I)) THEN    ! make use of previous index
         IH=MIN(I+2,N-1)
         IF(X.GE.C(5,IH)) IH=N-1
         IL=I
      ELSE
         IL=MAX(I-2,1)
         IF(X.LT.C(5,IL)) IL=1
         IH=I
      END IF
   10 I=(IL+IH)/2       ! binary search
      IF(I.NE.IL) THEN
         IF(X.GE.C(5,I)) THEN
            IL=I
         ELSE
            IH=I
         END IF
         GOTO 10
      END IF
      H=X-C(5,I)
      CSPF=C(1,I)+H*(C(2,I)+H*(C(3,I)+H*C(4,I)))
      END
```

Spline-Funktion zur Interpolation. Das Unterprogramm `USPC` berechnet für äquidistante Datenpunkte `Y(N)` zwischen den Grenzen `XA` und `XB` die Koeffizienten `A(N)`. Durch Verwendung der not-a-knot Bedingung ist die Zahl der Koeffizienten gleich der Zahl der Datenpunkte. Das Programm braucht zwei Hilfsfelder `G(N)` und `H(N)` zur Berechnung.

Subprogram `uspcn.F`

Listing A.26: Interpolation mit äquidistanten B-spline Funktionen

```
      SUBROUTINE USPCN(Y,A,N,G,H)
*     Calculation of B-spline coefficients A(N) from equidistant function
*     values in array Y(N), using the not-a-knot condition.
*     Arrays Y(*) and A(*) may be identical.
*     G(N) and H(N) are auxiliary arrays used during computation.
      REAL Y(N),A(N),G(N),H(N)
      DO I=2,N
       G(I)=Y(I)-Y(I-1)
      END DO
      H(1)=(5.0*G(2)+G(3))/2.0
      H(2)=-H(1)+3.0*(G(3)+G(2))
      G(2)=2.0
      DO I=3,N-1
       H(I)=-H(I-1)/G(I-1)+3.0*(G(I+1)+G(I))
       G(I)=4.0-1.0/G(I-1)
      END DO
      H(N)=(5.0*G(N)+Y(N-1)-Y(N-2))/2.0
      G(N)=1.0-2.0/G(N-1)
*     backward ...
      H(N)=(-2.0*H(N-1)/G(N-1)+H(N))/G(N)
      DO I=N-1,2,-1
       H(I)=(H(I)-H(I+1))/G(I)
      END DO
      H(1)=(H(1)-2.0*H(2))
*     ... and forward solution
      ALN=Y(N)+Y(N)-Y(N-1)-(2.0*H(N)+H(N-1))/3.0
      DO I=1,N-1
       A(I)=Y(I)+Y(I)-Y(I+1)+(2.0*H(I)+H(I+1))/3.0
      END DO
      A(N)=ALN
      END
```

Interpolation. Die Funktion `USPF` berechnet einen Funktionswert bei Angabe der Abszisse `X`, der Koeffizienten im Feld `A(N)` und der Grenzen `XA` und `XB`. Für Argumente außerhalb des Intervalls erfolgt Extrapolation.

Subprogram `uspf.F`

Listing A.27: Funktionswert von äquidistanten B-spline Funktionen

```
      FUNCTION USPF(X,A,N,XA,XB)
*     Returned is the function value at X of B-spline function.
*     A(N) is array of coefficients, obtained from USPCN.
*     XA and  XB are X-values of first and last function value
*     Extrapolation is done for X outside the range XA ... XB.
      REAL A(N)
      R=FLOAT(N-1)*(X-XA)/(XB-XA)
      I=MAX(1,MIN(INT(R),N-3))
      Z=R-FLOAT(I)
      USPF
     +     =(A(I)+4.0*A(I+1)+A(I+2)+Z*(3.0*(-A(I)+A(I+2))
     +      +Z*(3.0*(A(I)-2.0*A(I+1)+A(I+2))
     +      +Z*(-A(I)+3.0*A(I+1)-3.0*A(I+2)+A(I+3)))))/6.0
      END
```

Allgemeine Spline-Funktion. Das folgende Unterprogramm `BSPLNK` berechnet für ein gegebenes Argument `X` die Werte der B-Spline-Funktionen $B_{j,k}$ für $j = l, l+1, \ldots l+k-1$, für einen Index l und eine Ordnung k der B-Splines ($k \leq 10$), die durch die Argumente `K` und `L` gegeben sind. Das Feld `T(M)` muß die Knotenkordinaten $\{t_j\}$ mit der Ordnung $t_j \leq t_{j+1}$ enthalten; innerhalb des Bereiches t_{k-1} bis t_{m+1-k} addieren sich die B-Splines entsprechend Formel (10.16) jeweils zu 1. Das Unterprogramm läßt auch mehrfache Knoten zu; bei einer Knoten-Multiplizität von ν sind die $(k-\nu)$-ten Ableitungen an den Knotengrenzen unstetig. Dies erlaubt auch die Anpassung von Daten mit einem diskontinuierlichen Verlauf. In Anwendungen werden der erste und der letzte Knoten t_1 und t_m oft als $(k-1)$-fache Knoten definiert; der in einer Anpassung nutzbare Bereich in x ist dann t_1 bis t_m.

Subprogram `bsplnk.F`

Listing A.28: B-spline-Funktion mit nicht-äquidistanten Knoten

```
      SUBROUTINE BSPLNK(X,K,B,L,T)
*     Returned are B(1)...B(K), the B-spline values for given X and order K
*     K = order of spline (1=const, 2=linear...) up to limit K=KMAX=10
*     The array T(KNK) contains the knot positions.
*     In T(K) =<  x  < T(KNK+1-K) the K B-spline values add up to 1.
*     Multiple knots allowed: (K - mult)-th derivative is discontinuous.
*     Multiple knots allow to define a B-spline sum with certain
*     discontinuities.
*
*     The B-spline values  B(1) ... B(K)      correspond to the B-splines
*                          B(L) ... B(L-1+K)
*     In total there are  NKN-K  B-splines. The index L can be determined
*     by the call
*        L=MAX(K,MIN(LEFT(X,T,NKN),NKN-K))
*     before the BSPLNK call.
*
*     Example: K = 4 = cubic B-splines
*     X-limits A = T(4) and B = T(KNK-3)
*     Knots T(1) ... T(3) are <= T(4), often defined with equidistant
*     knot distances; Knots T(KNK-2),T(KNK-1),T(KNK) are >= T(KNK-3),
*     often with equidistant knot distances.
*     The knot multiplicity at knot T(I) is 1, if T(I-1) < T(I) < T(I+1);
*     derivatives 1, 2 and 3 (= K-1) agree at a knot of multiplicity 1.
```

```
      REAL B(K),T(*)
      PARAMETER (KMAX=10)
      DOUBLE PRECISION DL(KMAX-1),DR(KMAX-1),SV,TR
*     ...
      IF(K.GT.KMAX) STOP ' BSPLNK: K too large '
      B(1)=1.0
      DO J=1,K-1
       DR(J)=T(L+J)-X
       DL(J)=X-T(L-J+1)
       SV=0.0D0
       DO I=1,J
        TR  =B(I)/(DR(I)+DL(J-I+1))
        B(I)=SV+TR*DR(I)
        SV  =  TR*DL(J-I+1)
       END DO
       B(J+1)=SV
       IF(B(  1)+1.0.EQ.1.0) B(  1)=0.0 ! avoid underflows
       IF(B(J+1)+1.0.EQ.1.0) B(J+1)=0.0
      END DO
      END
```

Fit von Spline-Funktion. Das folgende Unterprogramm `BSKFIT` paßt eine allgemeine Spline-Funktion an Daten der Form (10.1) an; die B-Splines der Ordnung `KS` werden durch das Feld `T(NKN)` von Knoten vorgegeben. Das Ergebnis der Anpassung sind die Parameter `(NKN-K)` in dem Feld `C(*)`. In dem Programm werden die Normalgleichungen aufgestellt, die wegen der Eigenschaften der B-Splines eine Bandstruktur haben. Die Matrixgleichung wird mit dem Unterprogramm `BMSOL` gelöst.

Subprogram `bskfit.F`

Listing A.29: Anpassung von B-spline-Funktion mit Methode der kleinsten Quadrate

```
      SUBROUTINE BSKFIT(X,Y,W,N,K,T,NKN,C)
*     Least squares spline fit of B-spline of order K to
*     data Y in array Y(N), with abscissa values X in arrays X(N)
*     and weights W in array W(N).
*     The array T(NKN) contains NKN (ordered) knot positions.
*     The resulting B-spline coefficients are in an array of dimension
*     NDIM = NKN-K, i.e. C(NKN-K).
*     Limitation: NDIM <= ND=100 ; K <= 10
      REAL X(N),Y(N),W(N), T(NKN),C(*)    ! C(NKN-K)
      PARAMETER (ND=100)
      REAL Q(4*ND),B(10)
*     ...
      DO I=1,NKN-K                        ! reset normal equation
       C(I)=0.0
      END DO
      DO I=1,K*(NKN-K)
       Q(I)=0.0
      END DO
      DO M=1,N
       L=MAX(K,MIN(LEFT(X(M),T,NKN),NKN-K)) !index within limits
       CALL BSPLNK(X(M),K,B,L,T)          ! calculate B-splines
       DO I=1,K                           ! add to normal equation
        C(I+L-K)=C(I+L-K)+B(I)*Y(M)*W(M)
        IQ=1+K*(I+L-K)-I-K
        DO J=I,K
         Q(J+IQ)=Q(J+IQ)+B(I)*B(J)*W(M)
        END DO
       END DO
      END DO
      CALL BMSOL(Q,C,K,NKN-K)             ! solve band matrix equation
```

```
      END
```

Funktionswerte der Spline-Funktion. Nach der Anpassung können Funktionswerte der angepaßten Spline-Funktion mit Hilfe der Funktion `BSKFUN` für einen bestimten Wert der Abszisse `X` berechnet werden. Die anderen Argumente der Funktion sind die gleichen wie bei dem Unterprogramm `BSKFIT`.

Subprogram `bskfun.F`

Listing A.30: Funktionswerte der B-spline-Funktion

```
      FUNCTION BSKFUN(X,K,T,NKN,C)
*     Returned is the function value at X of B-spline function of
*     order K within the knot range.
*     C(*) is the array of coefficients, and T(NKN) is the knot array.
      PARAMETER (KMAX=10)
      REAL T(NKN),B(KMAX),C(*)
      DOUBLE PRECISION SUM
*     ...
      L=MAX(K,MIN(LEFT(X,T,NKN),NKN-K))    ! index within limits
      CALL BSPLNK(X,K,B,L,T)                 ! calculate K B-splines
      SUM=B(1)*C(L-K+1)
      DO I=2,K
       SUM=SUM+B(I)*C(I+L-K)
      END DO
      BSKFUN=SUM
      END
```

A.7 Orthogonale Polynome

Konstruktion eines orthonormalen Polynoms. Das Unterprogramm `ORTHFT` bestimmt die Koeffizienten einer orthogonalen Parametrisierung von Daten gemäß (10.1) in Feldern `X(N)`, `Y(N)` und `W(N)` für `NPAR` Parameter, also bis zur Ordnung `NPAR-1`. Ein Hilfsfeld `Q(2,N)` wird benötigt. Das Ergebnis der Anpassung wird in einem Feld `P(5,NPAR)` gespeichert.

Subprogram `orthft.F`

Listing A.31: Konstruktion eines orthonormalen Polynoms

```
      SUBROUTINE ORTHFT(X,Y,W,N,NPAR,Q,P)
*     Construction of orthogonal polynomials from
*     data Y in array Y(N), with abscissa values X in arrays X(N)
*     and weights W in array W(N) with NPAR parameters (up to order
*     NPAR-1 of the polynomial.
*     Q(2,N) is an auxiliary array, used during computation.
*     The array P, dimensioned P(5,NPAR), contains after return
*     the result of the fit, to be used in subroutine ORTHFE later.
*        P(1,i) = alpha                P(1,1) = 100*N + NPAR
*        P(2,i) = beta                 P(2,1) = variance estimate
*        P(3,i) = gamma                P(3,1) = p0(x) (constant)
*        P(4,i) = coefficient
*        P(5,i) = remaining chi**2
      REAL X(N),Y(N),W(N),Q(2,N),P(5,NPAR)
*     Double precision for the accumulation of sums
      DOUBLE PRECISION ABGDS(5),AL,BE,GA,DE,SQSUM
      EQUIVALENCE (ABGDS(1),AL),(ABGDS(2),BE),(ABGDS(3),GA),
     +            (ABGDS(4),DE),(ABGDS(5),SQSUM)
      NP=NPAR
      DO J=1,5
       ABGDS(J)=0.0D0
```

```
      END DO
*     determination of constant term
      DO I=1,N
       GA=GA+W(I)          ! sum of weights
       DE=DE+W(I)*Y(I)     ! weighted y sum
      END DO
      IF(GA.NE.0.0) GA=1.0D0/SQRT(GA)
      DE =GA*DE
      DO I=1,N
       SQSUM =SQSUM+W(I)*(Y(I)-GA*DE)**2 ! sum of squared residuals
       Q(1,I)=GA           ! initialize Q array
       Q(2,I)=0.0
      END DO
      DO J=1,5
       P(J,1)=ABGDS(J)     ! store parameters
      END DO
*     recursive loop for higher terms
      IAM=1
      DO K=2,NP
       IAM=3-IAM
       DO J=1,4
        ABGDS(J)=0.0D0
       END DO
       DO I=1,N
        AL=AL+W(I)*X(I)*Q(3-IAM,I)**2          ! sum x*f(j-1)**2
        BE=BE+W(i)*X(I)*Q(IAM,I)*Q(3-IAM,I)  ! sum f(j-2)*f(j-2)
       END DO
       DO I=1,N
        Q(IAM,I)=(X(I)-AL)*Q(3-IAM,I)-BE*Q(IAM,I) ! unnormalized f(j)
        GA=GA+W(I)*Q(IAM,I)*Q(IAM,I) ! sum for normalization
       END DO
       IF(GA.NE.0.0) GA=1.0/DSQRT(GA)
       DO I=1,N
        Q(IAM,I)=GA*Q(IAM,I)    ! normalize f(j)
        DE=DE+W(I)*Q(IAM,I)*Y(I) ! sum for coefficient
       END DO
       SQSUM=MAX(0.0D0,SQSUM-DE*DE) ! update sum of squares
       DO J=1,5
        P(J,K)=ABGDS(J)         ! store parameters
       END DO
      END DO
      P(1,1)=FLOAT(100*N+NP)    ! dimension parameter
      P(2,1)=1.0                ! factor for error calculation
      END
```

Interpolation. Interpolierte Werte an beliebigen Werten der Abszisse `X` können anschließend mit der Funktion `ORTHFE` berechnet werden. Zusätzlich zum interpolierten Funktionswert `Y` wird auch der Fehler `DY` nach Fehlerfortpflanzung berechnet.

Subprogram `orthfe.F`

Listing A.32: Interpolation mit orthonormalem Polynom

```
      SUBROUTINE ORTHFE(X,P,Y,DY)
*     Return function value Y and error DY of the orthogonal polynomial
*     at the abscissa value X, using the array P (see ORTHFT)
      REAL X,Y,DY,P(5,*)
      DOUBLE PRECISION Q(2),F,DF
      NP=MOD(P(1,1),100.0)+0.5
      Q(1)=P(3,1)         ! constant term
      Q(2)=0.0
      F =P(3,1)*P(4,1)
```

```
DF=P(3,1)*P(3,1)
IAM=1
DO K=2,NP
 IAM=3-IAM  ! recurrence relation for higher terms
 Q(IAM)=((X-P(1,K))*Q(3-IAM)-P(2,K)*Q(IAM))*P(3,K)
 F =F +P(4,K)*Q(IAM)
 DF=DF+Q(IAM)*Q(IAM)
END DO
Y =F                ! function value
DY=SQRT(DF*P(2,1)) ! standard deviation
END
```

A.8 Übersicht über die Fortran-Programme

Listings

Literaturverzeichnis

[1] Igor Alexander and Helen Morton. *An Introduction to Neural Computing*. Chapman and Hall, London, 1990.

[2] A.M.Erisman and W.F.Tinney. On computing certain elements of the inverse of a sparse matrix. *CACM*, 18:177.

[3] Roger Barlow. *Statistics: A Guide to the Use of Statistical Methods in the Physical Sciences*. The Manchester Physics Series. John Wiley & Sons, Chichester, 1989.

[4] Roger Barlow. Extended maximum likelihood. *Nuclear Instr. and Methods in Phys. Res. A*, 297:496, 1990.

[5] J. M. Bernardo and A. F. M. Smith. *Bayesian Theory*. John Wiley and Sons Ltd, 1994.

[6] Thomas Beth. *Verfahren der schnellen Fourier-Transformation*. Teubner, 1984.

[7] Christopher M. Bishop. *Neural Networks for Pattern Recognition*. Clarendon Press, Oxford, 1995.

[8] Volker Blobel. Unfolding methods in high-energy physics experiments. In *CERN School of Computing*, 1985. Bericht CERN 85-02.

[9] Siegmund Brandt. *Statistical and Computational Methods in Data Analysis*. North Holland, 1976.

[10] Siegmund Brandt. *Datenanalyse*. Wissenschaftsverlag, Mannheim/Leipzig/Wien/Zürich, 1992.

[11] F. P. Brooks Jr. *The Mythical Man-Month*. Addison Wesley, 1995.

[12] C. G. Broydon. The convergence of a class of double-rank minimization algorithms. *Inst. Maths. Applics.*, 6, 1970.

[13] A. Compagner. Operational conditions for random-number generation. *Physical Review* **E**, 52:5634, 1995.

[14] Harald Cramér. *Mathematical Theory of Statistics*. Princeton University Press, Princeton, 1954.

[15] Giulio D'Agostini. Probability and measurement uncertainty in physics – a Bayesian primer. Bericht DESY 95-242, 1995.

[16] W. C. Davidon. Variable metric methods for minimization. *A.E.C. Res. and Dev. Rep. ANL-5990, ANL*, 1959.

[17] Carl de Boor. *A Practical Guide to Splines*. Applied Mathematical Sciences. Springer-Verlag, 2001.

[18] P. Dierckx. *Curve and Surface Fitting with Splines*. Applied Mathematical Sciences. Clarendon, Oxford, 1993.

[19] Edsger W. Dijkstra. A note on two problems in connexion with graphs. *Numerische Mathematik*, 1:269–271, 1959.

[20] W. T. Eadie, D. Drijard, F. E. James, M. Roos, and B. Sadoulet. *Statistical Methods in Experimental Physics*. North Holland, 1971.

[21] Gary J. Feldman and Robert D. Cousins. Unified approach to the classical statistical analysis of small signals. *Phys. Rev. D*, 57:3873–3889, Apr 1998.

[22] R. Fletcher. A new approach to variable metric algorithms. *Comp. J.*, 13, 1970.

[23] Philip E. Gill, Walter Murray, and Margarete H. Wright. *Practical Optimization*. Academic Press, New York and London, 1981.

[24] D. Goldfarb. A family of variable metric methods derived by variational means. *Mathematics of Computation*, 24, 1970.

[25] Gene H. Golub and Gérard Meurant. *Matrices, Moments and Quadarture with Applications*. Princeton University Press, Princeton and Oxford, 2010.

[26] Particle Data Group. Review of particle physics. *Physical Review* **D** *Particles and Fields*, 54:1, 1996.

[27] Fritz Gutbrod. A fast random number generator for the intel paragon supercomputer. *Computer Physics Communications*, 87:291, 1995.

[28] Fritz Gutbrod. On the periods of the ranshi random number generator. *Int. Journal of Modern Physics C*, 7:909, 1996.

[29] Per Christian Hansen. *Rank-Deficient and Discrete Ill-Posed Problems – Numerical Aspects of Linear Inversion*. Society for Industrial and Applied Mathematics, Philadelphia, 1998.

[30] Per Christian Hansen. *Discrete Inverse Problems – Insight and Algorithms*. Society for Industrial and Applied Mathematics, Philadelphia, 2010.

[31] J. Hertz, A. Krogh, and R. G. Palmer. *Introduction to the Theory of Neural Computation*. Addison-Wesley, 1991.

[32] C. A. R. Hoare. Quicksort. *Computer Journal*, 5(1):10–15, 1962.

[33] A. Höcker and V. Kartvelishvili. SVD approach to data unfolding. *Nucl. Instr. and Methods A*, 372:469 – 481, 1996.

[34] Kurt Hornik. Approximation capabilities of multilayer feedforward networks. *Neural Networks*, 4:251, 1991.

[35] I.S.Duff, A.M.Erisman, and J.K.Reid. *Direct Methods for Sparse Matrices,*. Oxford Science Publications, Oxford, 1986.

[36] F. James. RANLUX: A Fortran implementation of the high-quality pseudorandom number generator of Lüscher. *Comp. Phys. Comm.*, 79:111, 1994.

[37] F. James and M. Roos. Minuit - a system for function minimization and analysis of parameter errors and correlations. *Comp. Phys. Comm.*, 10, 1972.

[38] Frederick James. *Statistical Methods in Experimental Physics (2nd Edition)*. World Scientific, 2006.

[39] J.L.Bentley. *Comm. of the ACM*, 18:9, 1975.

[40] Jari Kaipio and Erkki Somersalo. *Statistical and Computational Inverse Problems*. Springer, 2005.

[41] Brian W. Kernighan and Dennis M. Ritchie. *Programmieren in C*. Hanser und Prentice-Hall Internat., 1990.

[42] D. E. Knuth. *The Art of Computer Programming, Volume II: Seminumerical Algorithms*. Computer Science and Information Processing. Addison Wesley, Reading, Mass., 1981.

[43] E. L. Kosarev and E. Pantos. Optimal smoothing of noisy data by fast Fourier transform. *Scientific Instruments*, 16:537–543, 1983.

[44] Walter Kraemer. *Denkste*. Piper, 1998.

[45] J. B. Kruskal, Jr. On the shortest spanning subtree of a graph and the travelling salesman problem. *Proceedings of the AMS*, 7(1):48–50, 1956.

[46] K.Takahashi, J.Fagan, and M.Chin. Formation of a sparse bus impedance matrix and its application to short circuit study. *Proceedings 8th PICA Conference*, 1973.

[47] C. L. Lawson and R. J. Hanson. *Solving Least Squares Problems*. Prentice Hall, Englewood Cliffs, 1974.

[48] Ludger Lindemann and G. Zech. Unfolding by weighting Monte Carlo events. *Nucl.Instr. and Methods* **A**, 354:516, 1994.

[49] A. K. Louis. *Inverse und schlecht gestellte Probleme*. Teubner, Stuttgart, 1989.

[50] Martin Luescher. A portable high-quality random number generator for lattice field theory simulations. Bericht DESY 93-133, 1993.

[51] Louis Lyons. *Statistics for Nuclear and Particle Physicists*. Cambridge University Press, Cambridge, 1986.

[52] P. Mazzanti and R. Odorico. Bottom jet recognition by neural networks and statistical discriminants. *Zeitschrift für Physik C*, 59:273, 1993.

[53] Alexander M. Mood and Franklin A. Graybill. *Introduction to the Theory of Statistics*. Series in Probability and Statistics. Mc Graw-Hill, 1963.

[54] Glenford J. Myers. *The Art of Software Testing*. John Wiley and Sons, 1979.

[55] Glenford J. Myers. *Methodisches Testen von Programmen*. Oldenbourg, 1991.

[56] J. A. Nelder and R. Mead. *Computer Journal*, 7:308, 1965.

[57] S. J. Press. *Bayesian Statistics: Principles, Models, and Applications*. John Wiley and Sons, 1989.

[58] William H. Press, Saul A. Teukolsky, William T. Vetterling, and Brian P. Flannery. *Numerical Recipes in FORTRAN. The Art of Scientific Computing*. Cambridge University Press, Cambridge, 1992.

[59] S. W. Provencher. A constrained regularization method for inverting data represented by a linear algebraic or integral equation. *Comp. Phys. Comm.*, 27:213–227, 1982.

[60] R.Barlow. Asymmetric systematic errors. arXiv:physics/0306138, 2003.

[61] R.Barlow. Asymmetric statistical errors. arXiv:physics/0406120, 2004.

[62] R.Barlow. A note on delta lnl = -1/2 errors. arXiv:physics/0403046, 2004.

[63] Andreas Rieder. *Keine Probleme mit Inversen Problemen*. Vieweg, Wiesbaden, 2003.

[64] R. Rojas. *Theorie der neuronalen Netze*. Springer, 1993.

[65] B. W. Rust and W. R. Burrus. *Mathematical Programming and the Numerical Solution of Linear Equations*. American Elsevier Publishing Company, New York, 1972.

[66] R.v.Mises. *Probability,Statistics and Truth*. Dover, 1981.

[67] M. Schmelling. The method of reduced cross-entropy. A general approach to unfold probability distributions. *Nuclear Instr. and Methods in Phys. Res. A*, 340:400, 1990.

[68] H. R. Schwarz. *Numerische Mathematik*. Teubner, Stuttgart, 1993.

[69] H. R. Schwarz, H. Rutishauser, and E. Stiefel. *Numerik symmetrischer Matrizen*. Teubner, Stuttgart, 1968.

[70] R. Sedgewick. *Algorithms in C++*. Addison Wesley, Boston, USA, 1992.

[71] D. F. Shanno. Conditions of quasi-newton methods for function minimization. *Mathematics of Computation*, 24, 1970.

[72] Donald L. Shell. A high-speed sorting procedure. *Communications of the ACM*, 2(7):30–32, 1959.

[73] Ralph Sinkus. A novel approach to error function minimization for feed forward neural networks. *Nucl. Instr. and Meth. in Phys. Res. A*, 361:290, 1995.

[74] A. M. Tikhonov and V. Y. Arsenin. *Metody resheniya nekorrektnyh zadach (Methods for Solution of ill-posed Problems)*. Nauka, Moskau, 1979.

[75] T.Carli und B.Koblitz. A multi-variate discrimination technique based on range-searching. In *Adv. Comp. Analysis Techniques in Phys.Research (Batavia, USA)*, 2000. hep-ph 0011224.

[76] H.P.Beck-Bornholdt und H.H.Dubben. *Der Hund der Eier legt*. rororo Serie Sachbuch, 1997.

[77] J.L.Bentley und J.H.Friedman. *Computing Surveys*, 11:4, 1979.

[78] I. Vattulainen, T. Ala-Nissila, and K. Kankaala. Physical tests for random numbers in simulation. *Physical Review Letters*, 73:2513, 1994.

[79] I. Vattulainen, K. Kankaala, J. Saarinen, and T. Ala-Nissila. A comparative study of some pseudorandom number generators. *Comp. Phys. Comm.*, 86:209, 1995.

[80] Curtis R. Vogel. *Computational Methods for Inverse Problems*. Society for Industrial and Applied Mathematics, Philadelphia, 2002.

[81] A. D. Whalen. *Detection of Signals in Noise*. Academic Press, 1990.

[82] J. W. J. Williams. Algorithm 232: Heapsort. *Communications of the ACM*, 7(6):347–348, 1964.

[83] R. L. Winkler. *An introduction to Bayesian inference and decision*. Holt, Rinehart and Winston,Inc., 1972.

[84] G. Zech. Comparing statistical data to Monte Carlo simulation – parameter fitting and unfolding. Bericht DESY 95-113, 1995.

Index