The Astrophysical Journal Supplement Series, 247:60 (11pp), 2020 April https://doi.org/10.3847/1538-4365/ab72f2
© 2020. The American Astronomical Society. All rights reserved.

Enabling Catalog Simulations of Transient and Variable Sources Based on LSST Cadence Strategies

Rahul Biswas[1,2,3], Scott F. Daniel[3], R Hložek[4,5], A. G. Kim[6], and Peter Yoachim[3]
(LSST Dark Energy Science Collaboration)

[1] The Oskar Klein Centre for Cosmoparticle Physics, Department of Physics, Stockholm University, AlbaNova, Stockholm SE-10691, Sweden
[2] The eScience Institute, University of Washington, Seattle, WA 98195, USA
[3] Department of Astronomy, University of Washington, Seattle, WA 98195, USA
[4] David A. Dunlap Department of Astronomy and Astrophysics, University of Toronto, 50 St. George Street, Toronto, ON M5S3H4, Canada
[5] Dunlap Institute for Astronomy and Astrophysics, University of Toronto, 50 St. George Street, Toronto, ON M5S3H4, Canada
[6] Physics Division, Lawrence Berkeley National Laboratory, 1 Cyclotron Road, Berkeley, CA 94720, USA
Received 2019 June 3; revised 2020 January 15; accepted 2020 January 17; published 2020 April 1

Abstract

The Large Synoptic Survey Telescope (LSST) project will conduct a 10 year multi-band survey starting in 2022. Observing strategies for this survey are being actively investigated, and the science capabilities can be best forecasted on the basis of simulated strategies from the LSST Operations Simulator (`OpSim`). `OpSim` simulates a stochastic realization of the sequence of LSST pointings over the survey duration, and is based on a model of the observatory (including telescope) and historical data of observational conditions. `OpSim` outputs contain a record of each simulated pointing of the survey along with a complete characterization of the pointing in terms of observing conditions, and some useful quantities derived from the characteristics of the pointing. Thus, each record can be efficiently used to derive the properties of observations of all astrophysical sources found in that pointing. However, in order to obtain the time series of observations (light curves) of a set of sources, it is often more convenient to compute all observations of an astrophysical source, and iterate over sources. In this document, we describe the open source python package `OpSimSummary`, which allows for a convenient reordering. The objectives of this package are to provide users with an Application Programming Interface for accessing all such observations and summarizing this information in the form of intermediate data products usable by third party software such as `SNANA`, thereby also bridging the gap between official LSST products and preexisting simulation codes.

Unified Astronomy Thesaurus concepts: Astronomical methods (1043); Sky surveys (1464); Time domain astronomy (2109); Type Ia supernovae (1728); Cosmology (343)

1. Introduction

The Large Synoptic Survey Telescope (LSST) project will conduct a multi-band imaging survey (LSST Science Collaboration et al. 2009) of the sky with a 3.2 gigapixel camera on a 8 m class ground-based telescope at Cerro Pachon, Chile with a field of view of about 10 square degrees. The survey is scheduled to start taking data for science operations in 2022, and cover most of the Southern sky to median single-visit depths of $r \sim 24.3$, revisiting each location frequently. The combination of large sky-coverage, high depth and repeated visits enables several major scientific goals such as studying the solar system, astrophysical transients and variables, the Milky Way, and the physics of dark matter and dark energy (Ivezić et al. 2019). The efficacy of such investigations, particularly the Time Domain Astronomy programs involving observations of Time Domain Astronomical Sources (TDAS) such as transients, variable stars, active galactic nucleus, as well as solar system objects, depends critically on the observing strategy used to determine the sequence of pointings of the telescope.

Forecasting the performance of a science program with LSST survey strategies through the analysis of mock catalogs of observations of sources relevant to the science program is important and timely. Such forecasts are essential for the study of the impact of survey design and strategy. They are also instrumental for developing and testing appropriate analysis methods. Simulations of such mock catalogs requires models of the astrophysical sources, models of the observing instrument, and the analysis methods used to reduce the real data to such catalogs, and a model of the survey strategy along with a model of the observing conditions.

During the survey, the LSST project will make observations of the sky by pointing in different directions, recording the image for a certain amount of time and then processing the image. This procedure of procuring an image of a sky location for processing is referred to as a "visit" in the LSST literature, and the visit itself may involve two "snaps" separated by the shutter closing (current baseline strategies have two snaps of 15 s each resulting in a visit of exposure of 30 s). A visit will be followed by a possible slew of the telescope to a different location, after which a new visit starts again to repeat the cycle. As each visit is short, the observing conditions determined by the atmospheric and sky conditions can be approximated as constant during a visit. Currently, the LSST project simulates observations during its survey period using the Operations Simulator (`OpSim`) (Delgado et al. 2014; Delgado & Reuter 2016; Reuter et al. 2016). This is done with a prototype scheduler queuing visits according to a strategy designed to optimize science using a high-fidelity model of the telescope to calculate times required for telescope slews, and realtime observing conditions simulated using an empirical model of the sky and atmosphere. The output of such an `OpSim` simulation is a sequence of all the visits during the survey, and includes quantities required to describe the state of the telescope after each visit, and

the observing conditions during the pointing. Such OpSim outputs may be considered realized forecasts of LSST.

Such forecasts of science performance can be done in several ways representing different tradeoffs between computation/storage costs and the level of accuracy. On the low-resource end, the Metric Analysis Framework (Jones et al. 2014, MAF) uses "metrics" that are proxies of the scientific performance of the survey. Such proxies are built as functions of quantities related to observational conditions, and are usually designed by scientists on the basis of past experiences and intuition. Such metrics are extremely useful for studying the impact of survey strategy. On the resource-intensive end, there are image simulation codes (PhoSim (Peterson et al. 2015) and ImSim[7]) capable of using the OpSim outputs and producing detailed realistic simulations of LSST images, but are computationally expensive in terms of generation and storage. Furthermore, analysis of these images follows the expected LSST image processing using the LSST software stack (Jurić et al. 2017) and therefore best represents the scientific performance of LSST. The computational expense for forecasts using image simulations is much higher than the expenses for image simulation-aided efforts on current survey data, because the forecasts require both images of the sky, including galaxies and point sources, to be simulated. On the other hand, the use of simulated point sources on image stamps in characterizing photometry pipelines of survey data, as seen in the Supernova Legacy Survey (Astier et al. 2013), the Dark Energy Survey (Brout et al. 2019), and the Palomar Transient Factory (Frohmaier et al. 2019) underscores the importance of the image-processing steps in the accuracy of final results. This leads to the conclusion that such end-to-end explorations are hard, and therefore can only be used in a limited number of cases. An interesting middle ground is provided by catalog simulations, which utilize the OpSim outputs to obtain properties of visits, models of the astrophysical sources obtained from previous data or theoretical calculations, and models of aspects of the image-processing procedure in the LSST analyses to produce simulated catalogs directly. These simulated catalogs are mock realizations of the LSST Data Release Products (DRP) containing information on TDAS. LSST DRP catalogs are expected to contain forced photometry of all TDAS detected by LSST, and are expected to be released through a (nearly) annual frequency (Jurić et al. 2017). Thus, such catalog simulations replace the steps of image analysis and reduction to a catalog by an assumed model (which can in turn be improved through studies involving reprocessing older data and image simulations).

For the more abundant categories of TDAS such as Type Ia supernova (SN Ia), it is critical for catalog simulations to use distributed computing to speed up the simulations. There are at least two natural paradigms of organizing the distribution of computing resources. The first alternative (a) is to calculate the observed quantities corresponding to each telescope visit at a particular instance of time, which may be further split into smaller spatial regions. Indeed, this is almost essential for any image simulation, and is an approach utilized in generating "Instance Catalogs" by the LSST Catalog Simulations (CatSim; Connolly et al. 2010, 2014) that are used as intermediate data products by Image Simulation software like PhoSim and ImSim. These Instance Catalogs are catalogs of astrophysical objects in the simulated universe whose light are expected to impinge on the LSST CCDs on that particular visit, along with a complete description of their astrophysical properties at that instance of time. In this method, obtaining the visit information is simple; however, the state of the transient objects needs to be persisted from one visit to another, and the output of several visits has to be serialized before the light curves of the transients can be built. In the second approach (b) popular in the transient world, the paradigm involves distributing each astrophysical source (or groups thereof) to different resources, and simulating all of the observations of the source over a sequence of times. While this automatically leads to outputs with light curves for different objects in exactly the format useful for analysis, this calls for collecting the correct sequence of visits at a particular sky location, which is the only non-trivial step remaining.

Our objective in this work is to provide a solution to the collection of the correct sequence of visits for a transient or variable source to make alternative (b) simple. As described, we do this by providing an open source package with a simple public Application Programming Interface (API) that users can use to obtain such sequences of visits. We also recognize that there are useful and often-used codes like SNANA (Kessler et al. 2009, 2019a) that are used to produce catalog simulations of TDAS and demand specific forms of inputs aggregating this information. To enable the use of this code, we also provide a script that produces an intermediate data product (an observation library file in the SNANA terminology) in exactly the input form desired, so that this can work out of the box with SNANA simulations.

2. Methods

While we will not discuss TDAS simulations here, note that the observation information necessary for such simulations is available from OpSim outputs, while a separate code (not provided in this work) is necessary to model the population of astrophysical objects themselves to get simulated observations. In order to simulate catalogs of TDAS, one needs to simulate the observed "flux" or photon counts of a source of known apparent brightness, as parameterized by the specific flux $F_\nu(\lambda)$ at the top of the earth's atmosphere, and the uncertainty in the measured flux. The measured flux, or rather the counts of photons received from an astrophysical point source or the sky are modeled as random variables that follow a Poisson distribution, where the expected counts from the source and the sky can be calculated from the physical parameters of the telescope and instruments, knowledge of the effective point-spread function (PSF), and the specific flux per unit area of the sky. (See Appendix A or Ivezić et al. 2010 for a more comprehensive discussion). The expected counts of photons from astrophysical sources and the sky may be written (please see the Appendix for a derivation; here we only use a summary of the results) in terms of the source magnitude m and the sky brightness $m_{\rm sky}$:

$$c_{\rm source} = \kappa 10^{-0.4m}, \qquad c_{\rm sky} = \alpha 10^{-0.4m_{\rm sky}}, \tag{1}$$

where κ, α are quantities that can be written in terms of physical constants, physical parameters of the optical system, and noise-equivalent area of the effective PSF (FWHMeff as listed in OpSim outputs) of the visit, all of which are known or measured quantities. Additionally, κ depends on the total transmission function (optical system and atmosphere) through the throughput integral T_b (defined in Equation (12)), which changes from observation to observation, mostly driven by

[7] https://github.com/LSSTDESC/imSim

airmass and clouds), while α depends on the system transmission function through the system throughput integral Σ_b (defined in Equation (16)), which is constant except for tiny differences caused by flexure of the system, or slowly over the years through the evolution of the system. The signal-to-noise ratio of the flux measurement is driven by the Poisson error due to both the source and sky counts. Since OpSim outputs do not contain κ or α, but an equivalent set of variables, it is convenient to eliminate some of them in terms of quantities that are measured in a survey or available as simulated quantities in the OpSim outputs like the 5σ depth m_5, the sky brightness $m_{\rm sky}$, and the PSF width provided in OpSim in terms of FWHMeff. The general expression is

$$\kappa = \frac{25 \times 10^{0.4m_5}}{2}\left(1 + \sqrt{\left(1 + \frac{4\alpha}{25}10^{-0.4m_{\rm sky}}\right)}\right), \quad (2)$$

which reduces to the familiar background-dominated limit of $5\sqrt{\alpha} \times 10^{0.2(2m_5 - m_{\rm sky})}$ in the limit where $\sqrt{c_{\rm sky}} \gg 1$. This is similar in spirit to how $\sigma_{\rm rand}$ is calculated in Ivezić et al. (2019). These expressions relate κ to physical constants, physical parameters of the optical system that are constant in time through α, and the quantities $m_{\rm sky}$, m_5, and FWHMeff, available in OpSim. It should be remembered that all of these quantities α, m_5, and $m_{\rm sky}$ are not independent, therefore Equation (2) does not imply that changing α by changing the PSF would change κ. On the other hand, if the small difference between T_b and Σ_b is ignored such that $\frac{\alpha}{\kappa}$ is considered to a measured quantity from the measured PSF, one can find an expression for κ in terms of the OpSim quantities $m_{\rm sky}$, m_5, FWHMeff, and the pixel size

$$\kappa = \frac{25\alpha}{\kappa}10^{0.4(2m_5 - m_{\rm sky})}\left(1 + \frac{\kappa}{\alpha}10^{-0.4(m_5 - m_{\rm sky})}\right), \quad (3)$$

without worrying about the physical characteristics of the optical system. It should be noted that this formalism assumes that the sky background used in these equations is the truth. In the case of analysis of real data, these are estimated quantities, and could be biased high due to the low-surface-brightness wings of galaxies, as found, for example, in the Hyper-SuprimeCam data sets or simulations similar to the Deep Lens Survey (Ji et al. 2018). Since the OpSim values do not include such corrections, there is no way to emulate this directly in catalog simulations. However, supernova simulations including SNANA simulations can be performed with an additional attribute of location with respect to host galaxies. If such corrections are estimated from image simulations as a function of local host-galaxy properties, (e.g., surface brightness at the location of the supernova) estimated biases can be added to the results of catalog simulations.

Our goal is to obtain these terms for each visit in a transient light curve from the OpSim output. This is explained in a step-by-step procedure in Section 2.2

2.1. Input Data: Operation Simulator Outputs

To summarize the methodology used, we start by describing the input data product, namely the outputs from OpSim. The LSST project simulates observing strategies; using the OpSim and the resulting sequence of pointings, properties of observations are disseminated in the form of a sqlite database. The database contains multiple tables, and the most important ones for our purpose are the "summaryAllProps" and "proposal"[8] The "proposal" table is a table of scientific surveys or proposals, each of which have their own requirements in terms of desired visits and survey properties, along with a unique integer identifier "proposalId." Currently, LSST has a Wide FastDeep survey (WFD), a Deep Drilling Field survey (DDF), a Southern Galactic Cap Survey, a Milky Way Survey, and a Northern Ecliptic Spur survey in different geographical regions as shown in Figure 1 and different survey strategies applied to each of them.

The "summaryAllProps" table is the sequence of simulated observations based on the simulated conditions throughout the 10 year period. Each row of the table is an observation or a telescope pointing that we will refer to as "visits." The row for a visit is identified by an integer "observationId" with important properties characterizing the observation as well as the "proposalId" whose criteria it satisfies. The characteristics of the observations include the pointing location, the time of observation, the bandpass in which the observation is made, the airmass, the seeing and the PSF, the sky brightness, and the 5σ depth. The seeing is based on historical data, while the sky brightnesses are computed using a data-driven model (Yoachim et al. 2016). Together, these two tables tell us about all of the simulated observations, and the scientific proposal or survey that they were taken to satisfy. These represent the sum total of information available about the simulated strategies and are sufficient to generate catalog simulations. Complete details on such quantities are available from the schema of the output in the relevant version.[9,10] In the current versions, the pointings are located on a discrete grid with an integer (fieldID) identifying each point on the discrete grid. There is no fundamental requirement that an observing strategy use such a grid, and it is likely (and already true in some alternative simulators) that this grid does not exist; consequently the methodology we will describe below does not use this feature. To give an idea of the sizes involved, a typical operations simulator output contains about 2.5 million visits, while typical OpSim databases have a size of about 4.6 GB. There are some very specific details of OpSim outputs that are not obvious in a first encounter. We attempt to list them here:

1. Most of the proposals in the current baselines are non-overlapping. If there was a spatial location that was observed by survey WFD, it is not observed by a survey like Southern Celestial Pole or the Northern Ecliptic Spur. However, this is not true for WFD and DDF, and DDF fields can be observed by WFD as well. There is no reason that future mini-surveys will not have such overlapping properties.
2. For a small fraction of cases, there can be multiple (actually two) rows of the summary table that point to the same visit. This happens whenever a particular visit satisfies the requirements of two different proposals or surveys. Currently, this is seen in the overlapping area of

[8] In version 3, the "summaryAllProps" table was called the "summary" table.
[9] https://www.lsst.org/scientists/simulations/opsim/summary-table-column-descriptions-v335
[10] https://lsst-sims.github.io/sims_ocs/tables/summaryallprops.html

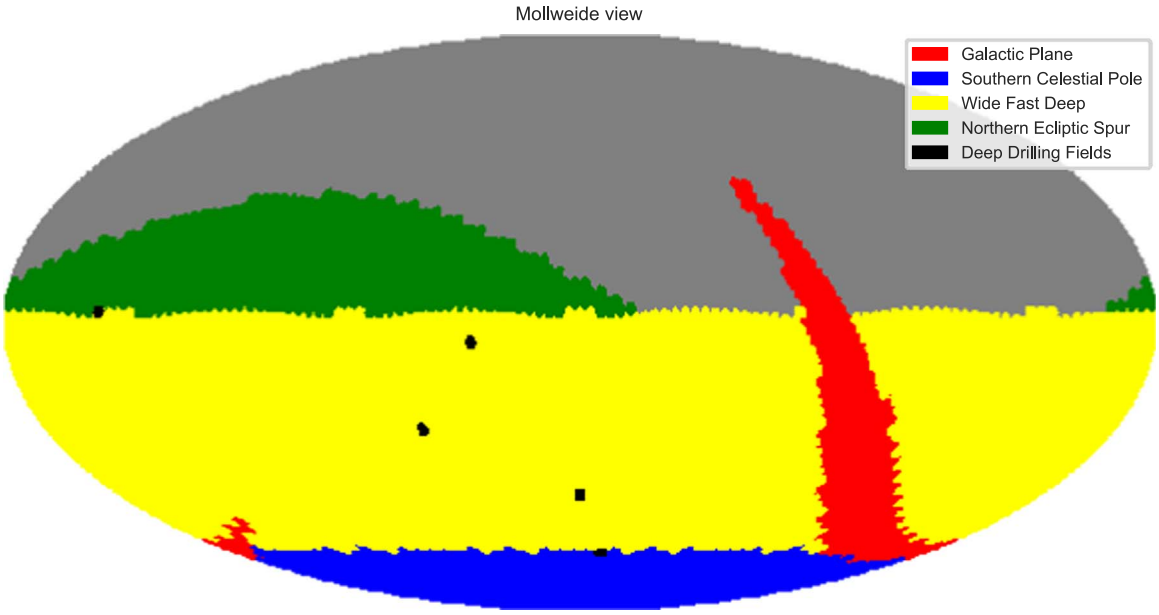

Figure 1. An all-sky representation in Celestial coordinates in the Mollweide projection of the different LSST proposals or surveys.

the Wide Fast Deep/Deep Drilling Field due to the previous point.
3. While some outputs of the `OpSim` come with a column of `ditheredRA` and `ditheredDec`, these are added post facto to the operation simulator output. Discussion of what the dithers should be is still ongoing, but it is useful to have the capability to replace these dithered observations with other dithers obtained from external sources.

2.2. Objectives

To further detail our objectives, we first define some terms that we will use in this paper. For any particular visit in LSST, a sky location within an angular radius of $1\overset{\circ}{.}75$ (the radius of the LSST focal plane) will be said to be "observed by LSST during this visit." In reality, this is an approximation: there are parts of the circular disk that are not covered by the rectangular geometry of the LSST chips, as well as gaps between the chips. Thus, the set of points observed by LSST during a visit according to the above definition is a superset of the points actually observed by the visit. We will ignore this distinction, except to note that the fill factor of chips is about 90%.[11] Given a sequence of visits (or rows of LSST `OpSim` output) and a sky location, one can find the sequence of visits that will observe the sky location according to this definition. As this quantity will be used repeatedly in this paper, we will refer to such a subset of all of the visits in an `OpSim` output as the "visit set" associated with a point on the sky.

In terms of the terminology defined above, our objectives are quite simple:

1. Given an `OpSim` output, and a sky location in terms of R.A. and decl., we need a simple API to obtain the visit set of this location, i.e., the sequence of visits in the `OpSim` output that observe this location.
2. Since the `OpSim` outputs are large (∼2.5 million visits) and the number of transients in LSST simulation volumes can be large (∼millions) for abundant and bright transients like SN Ia, this could lead to $\mathcal{O}(10^{12})$ simple computations if done by brute force in a naive way. We would like the process to be reasonably fast and not be a huge load on the memory requirements. Note that while the number of cosmologically useful SN in LSST will be smaller than the number of supernovae exploding in the observable volume, simulations have to simulate all of the supernovae before applying selection cuts to identify cosmologically useful supernovae.
3. Precompute this information on a dense grid and serialize to `SNANA` observation library formats to enable fast computations.
4. Since the Operations Simulation schema changes from version to version in terms of names, even though the conceptual setup remains the same, we would like to account for these changes and provide a stable interface for a catalog simulator.

3. Results

We present a simple, open source modular python package `OpSimSummary` based on other open source libraries, particularly the package `Scikit-learn` (Pedregosa et al. 2011) to meet each of our objectives. The code (Biswas et al. 2019) is available online,[12] while the particular release described in this paper is provided as a link. While the actual implementations are somewhat different in terms of packages used, some of the key ideas are inspired by those used in `MAF`. We first explain how this code meets each of our objectives:

3.1. Objective 1: API to Collect Visits Observing a Transient

This package achieves our objective of collecting visits observing a transient. It takes the publicly available LSST project provided `OpSim` outputs (in `OpSim` versions 3 and 4, as well as the two other schedulers that were used: the Feature

[11] https://www.lsst.org/about/camera/features

[12] https://www.github.com/lsstdesc/OpSimSummary

Based Scheduler (Naghib et al. 2019) and AltSched (Rothchild et al. 2019) as input, and provides an API for obtaining the visits for a point source at a sequence of arbitrary locations (defined by R.A. and decl. values). The code structure and examples for doing this are in the Appendix of this paper, and available with the source code itself. It also allows for the usage of an additional set of dithers input as the file name of a file in comma separated values format. If the sources to be simulated can be simulated independently, distribution is trivial to achieve by splitting their locations into arrays and using these arrays independently.

3.2. Objective 2: Computational Efficiency

While the problem of enumerating all the transients, and the visits that observe each one of them is naively $\mathcal{O}(N_{\rm visits}) \times \mathcal{O}(N_{\rm transient})$, it is intuitively clear that an easier computation should be possible. Since one does not require the computation of distances to visit centers that are too far away, the computation could take advantage of this.

There are different ways of implementing this intuitive idea of locality of visits. For example, a simple approach is choosing a convenient set of sky locations *pl* at which the visit sets are actually computed and approximating the visit set of an arbitrary point (for example, the set of point-source locations *tl*) by the visit set of a deterministically selected grid point. Thus, such schemes are defined by two components:

i. A set *pl* of selected points p, at which the visit sets v will be precomputed with no approximation. For the approximation to make a computing time difference, it would be nice for the size of *pl* to be significantly smaller than the size of the set *tv* of transients.
ii. A mapping from the visit sets $v(x)$ for any point x in *tl* to the visit sets $v(y)$ of points y in *pl*.

$$v(x) = v(\{v(y)\}), \qquad x \in tl,\, y \in pl. \tag{4}$$

A very simple algorithm along these lines would be nearest-neighbor-interpolation, where the component (ii) would be defined by assigning to an arbitrary point $x \in tl$, the visit sets of the point in *pl* closest to x. Interpolation techniques exploit the smoothness of the function being interpolated. Here the "function" under consideration is a map that returns the visit set of a point. While observing conditions in the sky vary reasonably smoothly with location and time, the set of points being observed by a visit is determined by a hard boundary (edge of the focal plane). Any time such an edge falls between two points, one of the two points will be observed and the other will not. As the distance between two points decreases, the probability of such a visit also decreases, but for a large number of true visits in a visit set (in the WFD survey of LSST, this is $\sim$1000), this would still be expected to happen. This implies that despite the smoothness of observing conditions with spatial locations, the visit set associated with points would not be "interpolated" as well as quantities like sky conditions. For a dense enough set of points, such a strategy could still provide an excellent approximation to the true visit sets. Of course, precomputation of the quantities in a dense set and their storage could itself be challenging, particularly if several versions of survey strategies are analyzed.

An elegant way to exploit the locality of visits without using the smoothness of the visit set is the use of a Tree data structure to partition the data based on spatial positions, so that we should expect a scaling of $\mathcal{O}(N_{\rm transient}) \times \mathcal{O}(\log(N_{\rm visits}))$. As far as the distance computations are concerned, i.e., if we ignore the position of the chips etc., then this calculation does not involve any additional approximation, and the speed attained is simply due to an organization of the calculation.

Here, we use a Tree implementation to exploit the locality of visits and provide a simple API to compute the visit set associated with individual visits. This should be easy to use for a simulator in the sense described above. This is done using an implementation within the package "Scikit-learn" (Pedregosa et al. 2011) called "BallTree" (Buitinck et al. 2013). We also use the API to precompute visit sets for a particular set of points to obtain approximate visit sets for each point, through an interpolation scheme for the well known `SNANA` code as described in the next subsection.

3.3. Objective 3: `SNANA` Observation Libraries

For transient simulations, `SNANA` has historically utilized the idea of splitting the sky into a relatively small set of predetermined points. `SNANA` simulates transients at only these locations. The abundance of transients simulated at each of these locations is tuned so that the expected number of transients (based on rates, survey volumes, etc.) starting within any period of time over the total survey footprint is the sum of the number of transients during the same time period at these locations. To do such simulations, `SNANA` reads in a precomputed set of telescope pointings of a survey called "simlib fields" and the observing conditions associated with each pointing observing each of the simlib fields from an ASCII file known as a `SNANA` observation library, with a specific format. An important objective of the `OpSimSummary` codebase is to provide precomputed observation libraries for `SNANA` to enable simulations of LSST. Previous versions of this codebase have been used to generate observation libraries used for `SNANA` simulations and analyses in the LSST DESC Science Requirement Document (The LSST Dark Energy Science Collaboration et al. 2018), while the code and features described here were primarily for the data generation of the PLAsTiCC challenge (The PLAsTiCC team et al. 2018), as described in the PLAsTiCC model and simulations paper (Kessler et al. 2019b). We therefore include a script to use the more general API of 3.1 to produce observation library files, which we are using for `SNANA` simulations of LSST. We proceed to describe the method by which such files were generated by first describing the quantities being used by `SNANA` and how they are related to `OpSim` quantities. We then describe the procedure we follow (in the script) to generate these observation library files: this includes the selection of footprints, selection of simlib fields, and then computing the quantities and writing them out.

First we tie the quantities in the observation library file to `OpSim` outputs, with a brief justification of the procedure. We then discuss the process of choosing the discrete locations at which these evaluations take place. The `SNANA` observation library quantities (bold-faced on the left side of Equation (5)) are related to the `OpSim` quantities (bold-faced on the right

side of Equation (5)) through simple transformations as

$$\begin{aligned}
\mathbf{PSF1} &= \mathbf{FWHMeff}/2.35/\mathrm{pixelSize} \\
A &= 1.51 \times \mathbf{FWHMeff} \\
\mathbf{ZPTAVG} &= \mathrm{ZPTApprox} + \mathrm{ZPTCorr} \\
\mathrm{ZPPTApprox} &= 2.5\log_{10}(25A) + (2\boldsymbol{m}_5 - \boldsymbol{m}_{\mathbf{sky}}) \\
\mathrm{ZPTCorr} &= 2.5\log_{10}\left(1.0 + \frac{10^{-0.4(\boldsymbol{m}_5 - \boldsymbol{m}_{\mathbf{sky}})}}{A}\right) \\
\mathbf{skySig} &= \mathrm{pixelSize} \times 10^{-0.4(\boldsymbol{m}_{\mathbf{sky}} - \boldsymbol{m}_5)} \\
\mathbf{PSF2} &= 0 \\
\mathbf{PSF2/1} &= 0. \qquad (5)
\end{aligned}$$

In particular, the variables PSF1 and PSF2 of SNANA meant to describe the PSF shape are represented by a simple two-dimensional isotropic Gaussian profile with a radial standard deviation. Since FWHMeff is related to the effective PSF area in OpSim, we set PSF2 and PSF2/1 to zero, and then the quantity PSF1 is simply related to FWHMeff of OpSim through the first equation of Equation (5). The quantity skySig of SNANA is related to $\frac{\alpha}{\kappa}$ of Equation (20). Finally, SNANA uses the approximation of $\alpha/\kappa = 1$ in Equation (3). Thus, one can see that ZPTAVG $= 2.5\log_{10}(\kappa)$ from Equation (3), with α/κ set to unity.

Next, we discuss the selection of discrete points, which we will refer to as "simlib fields," where the visit sets, and the quantities above are calculated for each visit in the visit sets using the API of 3.1. This is done by first selecting the DDF and WFD footprints, followed by uniformly selecting points from each of these footprints. The first step in this procedure is the selection of the DDF footprint. The footprint selection is done by going over all the visits in the LSST DDF minisurvey, which are identified by a "proposalId" index in OpSim. We tessellate the sky into small HEALPix[13] pixels (Górski et al. 2005), which we shall refer to as healpixels. The size of healpixels is chosen as NSIDE = 256, equivalent to pixel area ≈0.05 square degrees, which is roughly the size of a LSST chip), and find the healpixels that contain at least one point that is observed by at least a threshold number of visits. These healpixels together make up the footprint of the DDF minisurvey. The provided script arguments allow the user to set the threshold, but the default is 500 visits over 10 years. As the healpixels have equal area, the total area of the DDF footprint is the number of healpixels multiplied by the area of each healpixel.

The footprint of the WFD survey is also found in a similar way. First healpixels belonging to the DDF are removed, and any other healpixel that contains at least a single point that has a visit set with a number of visits over a threshold is taken to be the WFD footprint. This gives us the footprint of the WFD without any point that has been observed by DDF (i.e., with holes around the DDF location) as shown in the top panel of Figure 2. The color shows the healpix id in the HEALPix nest scheme. Again, the threshold can be defined by the user, but the default value used is 500. It should be noted that estimates of the WFD area in the LSST literature follow a different convention: the DDF area is not removed from the WFD footprint as we have done, and areas for the WFD footprint are often quoted with a threshold of 825 visits, the median requirement (LSST SRD) of WFD visits in a field.

[13] https://healpix.sourceforge.io

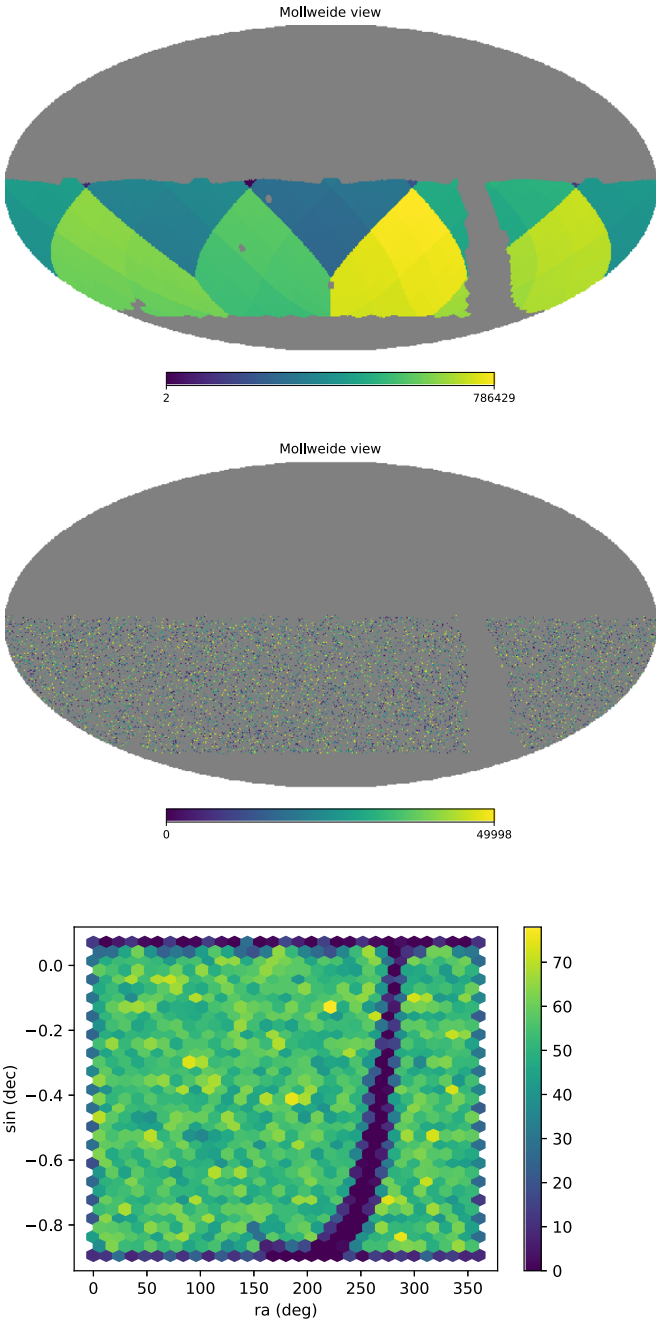

Figure 2. Observation library fixed position choices. (Top panel) healpixels from NSIDE = 256 (nest scheme) filled by the WFD survey used in observation libraries, which excludes the DDF areas as holes. The colors show the healpixel id, which for the particular NSIDE and scheme uniquely identify the healpixel. The solid color pattern shows that the healpixel ids have been written out in increasing order of healpixel ids. (Middle panel) a sample of 50,000 healpixel positions, rather than all of the healpixel positions in a random order that is written to observation library files. Here the color represents the order in which the healpixels are written to the observation library, and the lack of a solid color pattern shows that it is random rather than in increasing order of healpixel ids. (Lower panel). A hex-binned 2D histogram of the number of selected simlib fields in R.A. and sin(decl.). Since hex bins in these transformed coordinates are of equal area, the color uniformity away from the footprint edges demonstrates uniform sampling to obtain the points.

We treat these two geographical areas as different surveys whose areas have been measured. In order to simulate observation library files for each of these surveys, we first choose a fixed number of simlib fields within the survey

footprint. This fixed number can be chosen by the user, and the default number is 50,000 points for the WFD. The default number for the DDF is 150. The numbers are roughly proportional to the survey area footprints found above. The simlib fields are chosen randomly from the footprint. Here, random implies that any area of a fixed size within the footprint has the same probability of having a certain number of points. However, uniformly sampling an odd-shaped footprint is somewhat complicated. While the codebase can uniformly sample healpixels (using rejection sampling), and therefore the footprint that is made of healpixels, this is inherently slow. Thus, we choose to not worry about it. Instead we pick a random integer from the set of healpixel IDs for the healpixels in the footprint, giving us different healpixels with equal probability. This is sufficient for our requirements of uniform sampling, but would not pass specific tests of isotropy, as the points are chosen to be healpixel centers. The uniformity of sampling is shown in the lower panel of Figure 2 where a hex-binned plot of the selected points against R.A. and sin(decl.), where the color scale shows the number of selected points in each bin is shown to be roughly uniform, excluding the location of the Milky Way, which is not observed in detail by the WFD and DDF surveys in the current strategy.

For each of the selected simlib fields, we use the API of Section 3.1 to obtain the visit sets observing these points from both WFD and DDF proposals of the `OpSim` output. By construction, the visit set of points in the DDF footprint includes visits from both WFD and DDF proposals of `OpSim`, which is the correct way to simulate transients. Points on the WFD footprint have visit sets that contain visits only from the `OpSim` WFD proposal. We calculate the derived quantities required for `SNANA` using the quantities available from `OpSim` through Equations (5) as described above, and write out the information in the format required by `SNANA` to simlib files. We have found that when used with `SNANA` for rare transients, it is important to randomize the order in which these healpixel positions is read (i.e., not according to increasing healpixel id) and randomize the order in which the selected points are written out. An example of such a selected sample, with the colors showing the serial ordering of these points, is shown in the middle panel of Figure 2. Often, to speed up the simulation process, and control sizes of outputs, these simlibs are coadded over nights by `SNANA`, before the simulation.

In the bottom panel, we check that the distribution of points is truly uniform (barring anomalous regions like the Milky Way where there are no WFD/DDF visits) by checking the rough uniformity of the hexbin plot in the bottom panel of Figure 2.

We end this subsection on `SNANA` simlibs with a description of a few generic features of the simlibs for the current baseline cadences of LSST. The sky area of the WFD and the DDF footprints calculated in the method described above are $\sim 18.0 \times 10^3$ and ~ 47.6 sq. deg., respectively. These footprints are modeled by 50,000 and 150 simlib fields respectively, resulting in an average area per simlib field of 0.36 and 0.32 sq. deg., respectively. Without compression, these ASCII files have sizes of about 4.6 and 0.3 GB, respectively.

3.4. Validation, Performance, and Accuracy

First, as part of a standard test, we check that the visit sets from the API match the values with the brute-force solution. To give an idea of the time required in the current setup, (after a common initialization for all sources, which mostly involves reading in the database) the code required 37.2 ± 3.1 s to obtain visit sets for 50,000 sky locations spread over the same 100 sq. deg patch of the sky as shown in Figure 3.

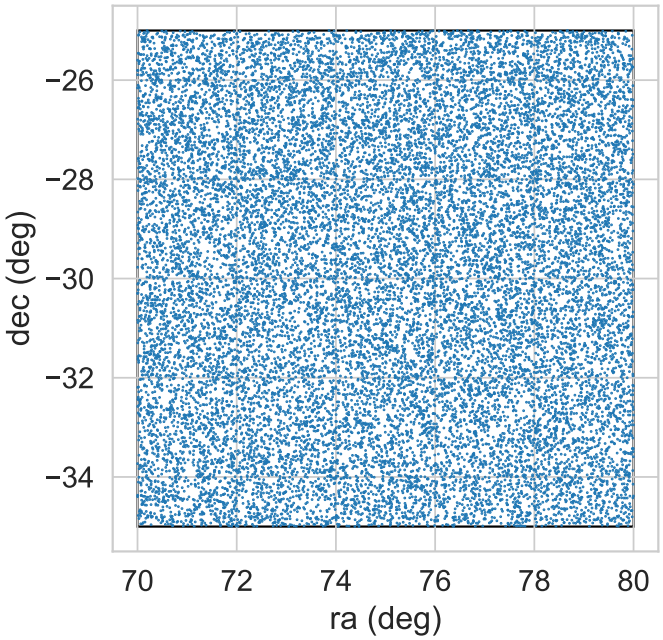

Figure 3. Location of the sky patch and a scatter plot of the points chosen uniformly in the area to evaluate the accuracy of discretization schemes.

As we have seen, approximating the visit list by the visit list of a nearby pre-computed point is a useful approach, not only because it enables the use of other software, but also because during the actual simulation obtaining the visit list is almost instantaneous. We have noted that this will inevitably result in differences with the correct calculations, but the approximation approaches the correct results, as the set of points where the precomputation is performed is made denser. Making the set of precomputed points arbitrarily dense requires precomputations and storage. So, quantifying a relationship between the denseness of points (e.g., simlib fields) and the accuracy of a precomputations at those points helps explain the tradeoffs. With the tools available here, we can quantify this accuracy with density.

To quantify the accuracy of visit sets in different approximation schemes with precomputed points, we choose a discretization scheme: following the discussion of such schemes, this can be described by the two components (a) a predetermined set pl of points p at which visit sets $v(p)$ are actually computed and (b) a prescription to assign visit sets to each point x in the set of transients, and the visit sets of a corresponding point p in pl. The approximation is perfectly accurate if the true visit sets $v(x)$ are identical to approximate visit sets $v(p)$, but when we study accuracy, we will limit ourselves to a weaker definition of accuracy defined as the difference in cardinality of the two visit sets $v(x)$ and $v(p)$. This is the difference in the number of epochs over the survey time, which is a good proxy for how the approximation affects studies of TDAS.

Since we expect the accuracy of a discretization scheme to depend strongly on the density of points pl at which the visit sets were actually computed, and perhaps weakly, on the actual discretization scheme, we choose a particular, convenient and relevant (as the use of such schemes are already prevalent) kind

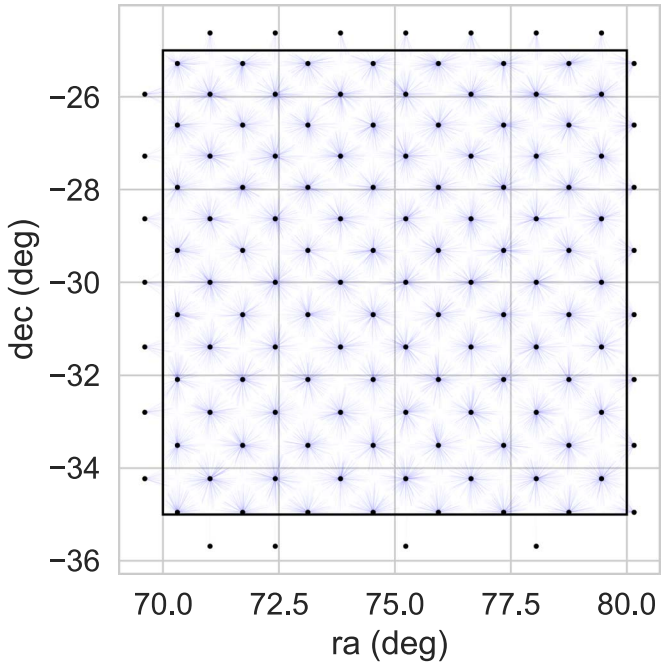

Figure 4. Scatter plot of the points (black dots) at which the visit sets are computed for a scheme where NSIDE = 64. Since the scheme results in assigning to any point x in Figure 3 the visit sets of exactly one black point y of this scatter plot, we use the blue lines to connect the pair of points ($\{x, y\}$ to show the mapping operation.

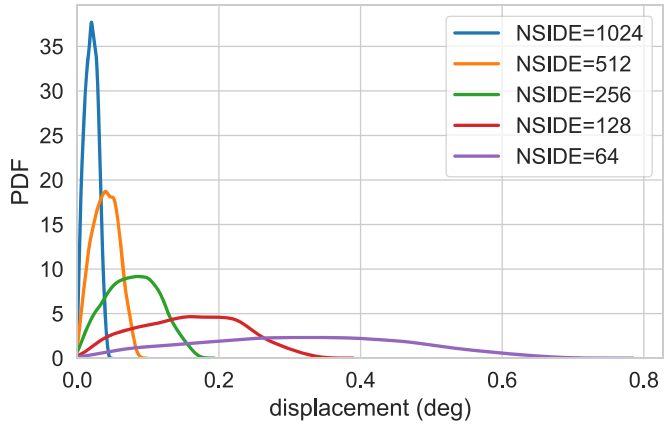

Figure 5. Distribution of lengths of the blue lines in Figure 4 in degrees connecting a point x in Figure 3 and a black point in Figure 4 where its visit set is evaluated. Shorter displacements indicate better approximations.

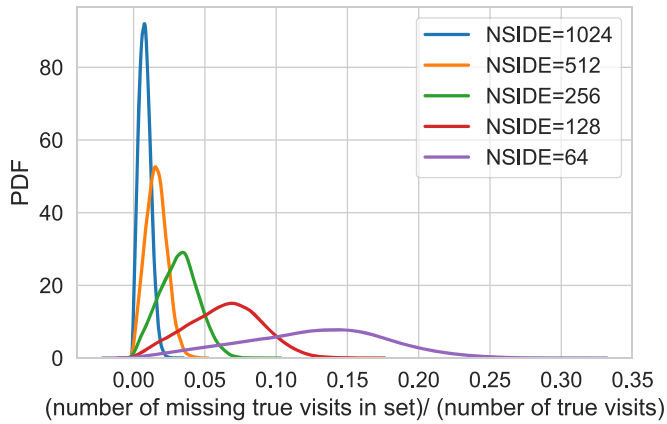

Figure 6. Distribution of the ratio of the number of visits missing in the approximate computation of visit sets to the true number of visits computed at points in Figure 3.

of discretization scheme and vary the density. To that end, we set up the following exercise: we choose a patch of the sky in the R.A. range of (70°, 80°), and the decl. range of (−35°, −25°), and select a sample *tl* of size = 20,000 points uniformly within this area. This patch of the sky and a scatter plot of these points is shown in Figure 3. The points chosen for precomputation *pl* are the positions of healpixels near the sky patch at different values of NSIDE = {64, 128, 256, 512, 1024}. We remind the reader that healpixels of a particular `NSIDE` all have equal area, and 12 × NSIDE^2 healpixels tile the surface of the entire sphere. This allows the calculation of the healpixel areas in square degrees, or their "resolution," which is simply the square root of the pixel area. Accordingly, these pixels have areas of 0.84, 0.21, 0.05, 0.01, 0.003 square degrees and resolutions of 55′.0, 27′.4, 13′.7, 6′.9, 3′.4. This is explained in Figure 4, where we show the setup for the choice of NSIDE = 64. Here, the black points show the *pl*, the healpixel centers at which we actually compute the visit sets for this scheme, while the blue lines connect pairs of points $\{x, y\}$, where $x \in tl$ is a point in Figure 3, and $y \in pl$ is a black point in Figure 4 where the visit sets assigned to x are the visit sets computed at y. Before looking at the results, we recall from the discussion that we expect the accuracy of visit sets to be related to the length of these blue lines in Figure 4 connecting the points x, y, and therefore show the distribution of such lengths in Figure 5.

We compute the true visit sets for each of the points in *tl* shown in Figure 3. We then use our discretization scheme for different values of `NSIDE`, to compute only the visit sets at the precomputed points *pl*, and assign to each point in *tl* the visit sets of the healpix position of the healpixel where the point lies. We refer to this as the approximated visit set under the specific discretization scheme. Following a previous discussion, we know that the inaccuracies will result in (a) visits in the original visit set missing in the approximated visit set, and (b) new visit sets in the approximate visit set that do not exist in the true visit set. While we would like to keep both of these quantities small, we also recognize that they will grow with the size of the true visit set. Hence, the appropriate quantity to monitor is the ratio of the number of missing visits to the number of true visits, and the ratio of the number of new visits to the number of true visits. In Figure 6, we show the distribution of the number of missing visits to the number of true visits for different values of NSIDE. The plot shows that the distribution is quite broad for NSIDE = 64, which has a resolution of ∼54′, and peaks at about 15% of the visits missing, while for NSIDE = 1024, which has a resolution of ∼3′, this distribution is very narrow, and peaks at slightly lower than a percent of missing visits, with the values in between following the trend in both the width of the distribution and the location of the peak. The distribution of the ratio of new visits to the true visit sets is also shown in Figure 7. The distribution of these visits is quantitatively very similar to the visits in Figure 6.

We should note that the difference between the total number of visits in the approximated visit set and the true visit set is statistically smaller, as these two errors affect the size in the opposite direction. However, replacing the visits by a different set of visits does not preserve the time of observation and can have significant differences due to, for example, differences of

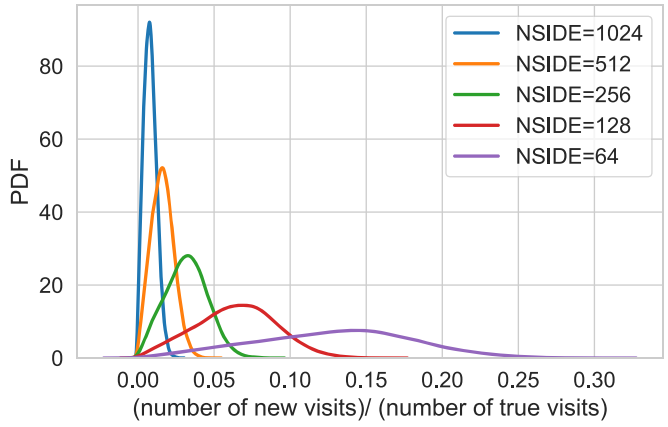

Figure 7. Distribution of the ratio of the number of visits in the approximate computation that are not in the true visit sets to the number of visits in the true visit sets.

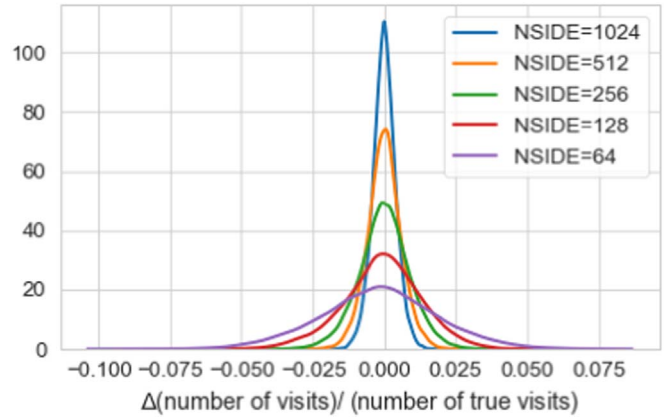

Figure 8. Distribution of the ratio of the difference in the number of visits in the approximate computation and the number of visits in the true visit set to the number of visits in the true visit sets.

bright and dark times. It also does not necessarily preserve the observation bandpass. We know that the distance estimates of SNe Ia are closely linked to band coverage, and thus cases where the missing visits correspond to the less frequently observed bands in LSST; it is likely that the additional visits are going to be in the more frequently observed bands. Such differences might be important for science programs like supernova cosmology, even though investigating these details is beyond the scope of this work. With this cautionary note, in Figure 8 we show the ratio of differences in sizes between the approximate visit set and the true visit set. Given the quantitative similarity between Figures 7 and 6, it is not surprising to see that the distribution of the difference in the number of visits normalized by the number of visits in the true visit set is centered at zero, with a width decreasing with NSIDE from a few percent to $\lesssim$1%.

4. Summary and Discussions

In this paper, we discuss the importance of catalog simulations of Time Domain Sources for the study of analysis methods and survey strategy of LSST. Survey strategies of LSST are currently simulated by the LSST project using `OpSim`; such simulated survey strategies are made public in the form of sqlite databases that are outputs of `OpSim`. We discuss the transformations of the set of quantities in `OpSim` that are required for catalog simulations. We also discuss the usefulness of reordering the outputs in terms of `OpSim` visits observing a particular sky location, delineating the necessity of such an API. While conceptually simple, we discuss why a naive solution is inefficient, particularly during the simulation of abundant sources. As strategies to address this issue, we discuss exploiting the locality of visits using a Tree data structure; and approximating the problem by serializing precomputed results for use with a simulator. This strategy makes the step during simulations essentially instantaneous, but inevitably results in errors that can be minimized by choosing a very dense set of predetermined points at the cost of large file sizes.

We present an open source modular python source software package for such operations, which contains an API for reading in `OpSim` outputs and reordering them to obtain the visits for each point. Thus, a simulation code can directly use this API to obtain the important quantities. A Tree is used to speed up the calculations. We also use the obtained visits, along with simple transformations of `OpSim` quantities to serialize the results for a set of points in the form of an `SNANA` simlib. The script to perform this was also made available as part of the `OpSimSummary` package. Currently, `OpSimSummary` works with `OpSim` outputs of version 3, and 4, along with outputs of Feature Based Scheduler and AltSched.

We study the accuracy of the approximate precomputed visit sets as a function of the density (or average separation) of the points at which the visit sets are actually computed, and show that at large average separations between these points, the visit set of sky locations has several visits missing, while several new visits not originally in the visit set are inserted. According to the numbers calculated for the current strategies, we would expect the current method to include $\sim$10% visits as missing while a similar number of $\sim$10% visits that were not in the true visit set were added.

This code has been used through the direct use of API in the study of serendipitous discoveries of kilonovae using the LSST (Setzer et al. 2019), which also formed part of a LSST DESC survey strategy white paper for Wide Fast Deep Fields in LSST (Lochner et al. 2018). `SNANA` observation library files (Biswas et al. 2017) generated through previous versions of `OpSimSummary` (and distributed publicly with the `SNANA` code) have been used in the study of serendipitous detection of kilonovae (Scolnic et al. 2018a) and the LSST DESC Science Requirement Document (The LSST Dark Energy Science Collaboration et al. 2018). This paper describes the improved versions of `SNANA` observation library files (simlibs) currently available, developed primarily for the data generation of PLAsTiCC (The PLAsTiCC team et al. 2018), as described in the PLAsTiCC models and simulations paper (Kessler et al. 2019b). These observation library files have also been used in the supernova simulations using `SNANA` used for the supernova cosmology analyses in the LSST DESC Survey Strategy white papers (Lochner et al. 2018; Scolnic et al. 2018b).

R.B. would like to thank David Cinabro for sharing his DACG project at the beginning of this work, Rick Kessler for stimulating discussions, particularly on the use of DACG with `SNANA` and Lynne Jones for help in understanding MAF.

This paper has undergone internal review in the LSST Dark Energy Science Collaboration. The internal reviewers were Philippe Gris and Isobel Hook, and the authors would like to thank them for their comments. Author contributions are as

follows: R.B. initiated and led project, wrote the oss package, drafted paper, derived results; S.F.D. provided support for middleware connecting OpSim with simulations of astrophysical sources; R.H. performed code beta testing and discussed results; A.G.K. motivated the project and consulted with R.B., particularly in its initial phases; and P.Y. supported OpSim and MAF usage.

During this work, R.B. was partially supported by the Washington Research Foundation Fund for Innovation in Data-Intensive Discovery and the Moore/Sloan Data Science Environments Project at the University of Washington and the Swedish Research Council (VR) through the Oskar Klein Centre. R.B. was further supported by the research environment grant "Gravitational Radiation and Electromagnetic Astrophysical Transients (GREAT)" funded by the Swedish Research council (VR) under Dnr 2016-06012.

The DESC acknowledges ongoing support from the Institut National de Physique Nucléaire et de Physique des Particules in France; the Science & Technology Facilities Council in the United Kingdom; and the Department of Energy, the National Science Foundation, and the LSST Corporation in the United States. DESC uses resources of the IN2P3 Computing Center (CC-IN2P3–Lyon/Villeurbanne—France) funded by the Centre National de la Recherche Scientifique; the National Energy Research Scientific Computing Center, a DOE Office of Science User Facility supported by the Office of Science of the U.S. Department of Energy under Contract No. DE-AC02-05CH11231; STFC DiRAC HPC Facilities, funded by UK BIS National E-infrastructure capital grants; and the UK particle physics grid, supported by the GridPP Collaboration. This work was performed in part under DOE Contract DE-AC02-76SF00515.

Software: Aside from the standard python package, this work used the following software packages: numpy (van der Walt et al. 2011), healpy (Zonca et al. 2019) and `HEALPix` packages (Górski et al. 2005), pandas (McKinney 2010), sqlalchemy, scikitlearn (Pedregosa et al. 2011; Buitinck et al. 2013). The examples use Jupyter Notebooks (Kluyver et al. 2016).

Appendix
Point Sources and S/N

Given the physical parameters describing a telescope, and a description of the sky and astrophysical sources, one can calculate quantities like the expected number of photons collected from a point source in the sky (i.e., no background galaxy), or the sky. Combining this with observing conditions based on seeing, airmass, etc., one can calculate a good estimate of the expected signal-to-noise ratio of an observation. We follow the discussion in Ivezić et al. (2010), keeping the gain $g = 1$ in our calculation. (For an extensive discussion including latest updates to LSST values, see Jones 2016).

For a point source with intensity $\epsilon(\lambda)$ as a function of its wavelength λ, the number of photons collected with an exposure time T in a telescope with collecting area A is given by

$$c_{\rm source} = \frac{AT}{h}\int_0^\infty F_\nu(\lambda)\lambda^{-1}d\lambda S^{\rm tot}(\lambda), \tag{6}$$

where the flux density $F_\nu(\lambda)$ is the frequency derivative of the intensity $F_\nu(\lambda) \equiv \frac{d\epsilon(\lambda)}{d\nu}$, while $S^{\rm tot}(\lambda)$ is the total transmission probability due to the atmosphere, the telescope system, and h is the Planck constant. We note that $S^{\rm tot}(\lambda)$ is also a function of time through the dependence of the atmospheric transmission functions on airmass, and atmospheric conditions.

Similarly, using the intensity per unit area of the sky $b_\nu(\lambda)$, one can calculate the time-averaged number of photons collected in $n_{\rm eff}$ pixels as

$$\frac{c_{\rm sky}}{n_{\rm eff}} = \frac{AT}{h}\int_0^\infty b_\nu(\lambda)\lambda^{-1}d\lambda S^{\rm sys}(\lambda)pa, \tag{7}$$

where pa is the area of a pixel. In order to estimate the number of photons collected from the source and sky during a particular exposure from he observed pixel counts, one uses estimators such as "aperture photometry" and "psf photometry." In each of these, one can use a value of $n_{\rm eff}$ pixels based on the observing conditions. For the estimator used in PSF photometry, this is given by

$$n_{\rm eff} = 2.27\frac{\rm FWHM}{\rm pixelScale}^2 \tag{8}$$

if the PSF profile is assumed to be a single radial Gaussian.

These counts obviously depend on the flux densities in exactly the same way as magnitudes in the bands, and can be calculated just by knowing the source magnitudes and the sky brightness in mags arcsec^{-2}, without requiring complete information on the flux densities:

$$m_{\rm source} = -2.5\log_{10}\left(\frac{\int_0^\infty F_\nu(\lambda)\lambda^{-1}d\lambda S^{\rm Total}(\lambda)}{\int_0^\infty F_\nu^{\rm std}(\lambda)\lambda^{-1}d\lambda S^{\rm Total}(\lambda)}\right), \tag{9}$$

$$c_{\rm source} = 10^{-0.4m_{\rm source}}\int_0^\infty F_\nu^{\rm std}(\lambda)\lambda^{-1}d\lambda S^{\rm Total}(\lambda)\frac{AT}{h} \tag{10}$$

$$= 5328(D/6.43m)^2(T/30{\rm s})T_b \times 10^{-0.4m_{\rm source}} \tag{11}$$

where the numerical values in the last line assume that the magnitude is in the AB system (i.e., $F_\nu^{\rm std} = 3631$ Jy), and that the area is a circular disk of diameter D, and the throughput integral T_b is

$$T_b \equiv \int_0^\infty S^{\rm tot}(\lambda)\lambda^{-1}d\lambda. \tag{12}$$

One can do a similar calculation for the counts of sky photons:

$$m_{\rm sky} = -2.5\log_{10}\left(\frac{\int_0^\infty b_\nu(\lambda)\lambda^{-1}d\lambda S^{\rm sys}(\lambda)}{\int_0^\infty F_\nu^{\rm std}(\lambda)\lambda^{-1}d\lambda S^{\rm sys}(\lambda)}\right) \tag{13}$$

$$c_{\rm sky} = 10^{-0.4m_{\rm sky}}\int_0^\infty F_\nu^{\rm std}(\lambda)\lambda^{-1}d\lambda S^{\rm Sys}(\lambda)\frac{AT\times pa\times n_{\rm eff}}{h} \tag{14}$$

$$= 5328(D/6.43m)^2(T/30{\rm s})\left(\frac{pa\times n_{\rm eff}}{0.04}\right)\sum_b \times 10^{-0.4m_{\rm sky}}, \tag{15}$$

where the system throughput integral Σ_b is

$$\sum_b \equiv \int_0^\infty S^{\rm sys}(\lambda)\lambda^{-1}d\lambda. \tag{16}$$

Hence, we see that we can write the photon counts as

$$c_{\rm source} = \kappa 10^{-0.4m_{\rm source}} \qquad c_{\rm sky} = \alpha 10^{-0.4m_{\rm sky}}, \tag{17}$$

where we can write κ, α in terms of physical quantities emphasizing the fact that $T_b(t)$ changes with time, as does $n_{\rm eff}$,

but is related directly to quantities that are supplied by most surveys (and in `OpSim`).

$$\kappa = 5328(D/6.43m)^2(T/30\mathrm{s})T_b(t) \tag{18}$$

$$\alpha = 5328(D/6.43m)^2(T/30\mathrm{s})\left(\frac{pa \times n_{\mathrm{eff}}(t)}{0.04}\right)\sum_b. \tag{19}$$

We then get

$$\frac{\alpha}{\kappa} = \frac{pa \times n_{\mathrm{eff}}}{0.04}\left(\frac{\sum_b}{T_b}\right). \tag{20}$$

The signal-to-moise ratio (S/N) of a measured source can be found from Poisson statistics:

$$\mathrm{S/N} = \frac{c_{\mathrm{source}}}{(c_{\mathrm{source}} + c_{\mathrm{sky}})^{1/2}}. \tag{21}$$

In practice, there may be other small sources of uncertainty such as read noise or other systematic errors that could in principle be grouped together with the Poisson noise in the denominator of Equation (21). Plugging Equation (17) into Equation (21), we can get

$$\mathrm{S/N} = \frac{\kappa 10^{-0.4m_{\mathrm{source}}}}{(\kappa 10^{-0.4m_{\mathrm{source}}} + \alpha 10^{-0.4m_{\mathrm{sky}}})^{1/2}}. \tag{22}$$

If values of m_{sky} and m_5 are supplied for an observation for a survey (as they often are), one can solve this to obtain Equation (2).

ORCID iDs

Rahul Biswas https://orcid.org/0000-0002-5741-7195
R Hložek https://orcid.org/0000-0002-0965-7864
A. G. Kim https://orcid.org/0000-0001-6315-8743
Peter Yoachim https://orcid.org/0000-0003-2874-6464

References

Astier, P., El Hage, P., & Guy, J. 2013, A&A, 557, A55
Biswas, R., Cinabro, D., & Kessler, R. 2017, simlib_minion, Zenodo, doi:10.5281/zenodo.1006719
Biswas, R., Setzer, C., & Azfar, F. 2019, LSSTDESC/OpSimSummary v2.0.0, Zenodo, doi:10.5281/zenodo.2671955
Brout, D., Scolnic, D., Kessler, R., et al. 2019, ApJ, 874, 150
Buitinck, L., Louppe, G., Blondel, M., et al. 2013, in Proc. European Conf. on Machine Learning and Principles and Practice of Knowledge Discovery in Databases, ed. H. Blockeel et al. (Berlin: Springer), 108
Connolly, A. J., Angeli, G. Z., Chandrasekharan, S., et al. 2014, Proc. SPIE, 9150, 14
Connolly, A. J., Peterson, J., Jernigan, J. G., et al. 2010, Proc. SPIE, 7738, 77381O
Delgado, F., & Reuter, M. A. 2016, Proc. SPIE, 9910, 991013
Delgado, F., Saha, A., Chandrasekharan, S., et al. 2014, Proc. SPIE, 9150, 915015
Frohmaier, C., Sullivan, M., Nugent, P. E., et al. 2019, MNRAS, 486, 2308
Górski, K. M., Hivon, E., Banday, A. J., et al. 2005, ApJ, 622, 759
Ivezić, Ž, Jones, R. L., & Lupton, R. 2010, The LSST Photon Rates and SNR Calculations (Tucson, AZ: LSST)
Ivezić, Ž, Kahn, S. M., Tyson, J. A., et al. 2019, ApJ, 873, 111
Ji, I., Hasan, I., Schmidt, S. J., & Tyson, J. A. 2018, PASP, 130, 084504
Jones, L. 2016, Calculating LSST Limiting Magnitudes and SNR Tech. Rep. SMTN-002, Zenodo, doi:10.5281/zenodo.192828
Jones, R. L., Yoachim, P., Chandrasekharan, S., et al. 2014, Proc. SPIE, 9149, 91490B
Jurić, M., Kantor, J., Lim, K., et al. 2017, in ASP Conf. Ser. 512, Astronomical Data Analysis Software and Systems XXV, ed. N. P. F. Lorente, K. Shortridge, & R. Wayth (San Francisco, CA: ASP), 279
Kessler, R., Bernstein, J. P., Cinabro, D., et al. 2009, PASP, 121, 1028
Kessler, R., Brout, D., D'Andrea, C. B., et al. 2019a, MNRAS, 485, 1171
Kessler, R., Narayan, G., Avelino, A., et al. 2019b, PASP, 131, 094501
Kluyver, T., Ragan-Kelley, B., Pérez, F., et al. 2016, Positioning and Power in Academic Publishing: Players, Agents and Agendas (Amsterdam: IOS Press), 87
Lochner, M., Scolnic, D. M., Awan, H., et al. 2018, arXiv:1812.00515
LSST Science Collaboration, Abell, P. A., Allison, J., et al. 2009, arXiv:0912.0201
McKinney, W. 2010, in Proc. 9th Python in Science Conf., ed. S. van der Walt & J. Millman (Austin, TX: SciPy), 51
Naghib, E., Yoachim, P., Vanderbei, R. J., Connolly, A. J., & Jones, R. L. 2019, AJ, 157, 151
Pedregosa, F., Varoquaux, G., Gramfort, A., et al. 2011, J. Mach. Learn. Res., 12, 2825
Peterson, J. R., Jernigan, J. G., Kahn, S. M., et al. 2015, ApJS, 218, 14
Reuter, M. A., Cook, K. H., Delgado, F., Petry, C. E., & Ridgway, S. T. 2016, Proc. SPIE, 9911, 991125
Rothchild, D., Stubbs, C., & Yoachim, P. 2019, PASP, 131, 115002
Scolnic, D., Kessler, R., Brout, D., et al. 2018a, ApJL, 852, L3
Scolnic, D. M., Lochner, M., Gris, P., et al. 2018b, arXiv:1812.00516
Setzer, C. N, Biswas, R., Peiris, H. V., et al. 2019, MNRAS, 485, 4260
The LSST Dark Energy Science Collaboration, Mandelbaum, R., Eifler, T., et al. 2018, arXiv:1809.01669
The PLAsTiCC teamAllam, Tarek, J., et al. 2018, arXiv:1810.00001
van der Walt, S., Colbert, S. C., & Varoquaux, G. 2011, CSE, 13, 22
Yoachim, P., Coughlin, M., Angeli, G. Z., et al. 2016, Proc. SPIE, 9910, 99101A
Zonca, A., Singer, L., Lenz, D., et al. 2019, JOSS, 4, 1298