The Astronomical Journal, 170:100 (41pp), 2025 August https://doi.org/10.3847/1538-3881/add3ec

Sorcha: A Solar System Survey Simulator for the Legacy Survey of Space and Time

Stephanie R. Merritt[1], Grigori Fedorets[1,2,3], Megan E. Schwamb[1], Samuel Cornwall[4], Pedro H. Bernardinelli[5], Mario Jurić[5], Matthew J. Holman[6], Jacob A. Kurlander[5], Siegfried Eggl[4,7,8], Drew Oldag[5,9], Maxine West[5,9], Jeremy Kubica[9,10], Joseph Murtagh[1], R. Lynne Jones[11,12], Peter Yoachim[5], Ryan R. Lyttle[1], Michael S. P. Kelley[13], Joachim Moeyens[5,14], Kathleen Kiker[14], Shantanu P. Naidu[15], Colin Snodgrass[16], Shannon M. Matthews[1], and Colin Orion Chandler[5,9]

[1] Astrophysics Research Centre, School of Mathematics and Physics, Queen's University Belfast, Belfast BT7 1NN, UK; m.schwamb@@qub.ac.uk
[2] Finnish Centre for Astronomy with ESO, University of Turku, FI-20014 Turku, Finland
[3] Department of Physics, University of Helsinki, P.O. Box 64, 00014 Helsinki, Finland
[4] Department of Aerospace Engineering, Grainger College of Engineering, University of Illinois at Urbana-Champaign, Urbana, IL 61801, USA
[5] DiRAC Institute and the Department of Astronomy, University of Washington, 3910 15th Avenue NE, Seattle, WA 98195, USA
[6] Center for Astrophysics | Harvard & Smithsonian, 60 Garden Street, MS 51, Cambridge, MA 02138, USA
[7] Department of Astronomy, University of Illinois at Urbana-Champaign, Urbana, IL 61801, USA
[8] National Center for Supercomputing Applications, University of Illinois at Urbana-Champaign, Urbana, IL 61801, USA
[9] LSST Interdisciplinary Network for Collaboration and Computing Frameworks, 933 N. Cherry Avenue, Tucson AZ 85721, USA
[10] McWilliams Center for Cosmology, Department of Physics, Carnegie Mellon University, Pittsburgh, PA 15213, USA
[11] Rubin Observatory, 950 N. Cherry Avenue, Tucson, AZ 85719, USA
[12] Aston Carter, Suite 150, 4321 Still Creek Drive, Burnaby, BC V5C6S, Canada
[13] Department of Astronomy, University of Maryland, College Park, MD 20742-0001, USA
[14] Asteroid Institute, 20 Sunnyside Avenue, Suite 427, Mill Valley, CA 94941, USA
[15] Jet Propulsion Laboratory, California Institute of Technology, Pasadena, CA, USA
[16] Institute for Astronomy, University of Edinburgh, Royal Observatory, Edinburgh, EH9 3HJ, UK
Received 2025 January 28; revised 2025 April 25; accepted 2025 April 28; published 2025 July 21

Abstract

The upcoming Legacy Survey of Space and Time (LSST) at the Vera C. Rubin Observatory is expected to revolutionize solar system astronomy. Unprecedented in scale, this 10 yr wide-field survey will collect billions of observations and discover a predicted ∼5 million new solar system objects. Like all astronomical surveys, its results will be affected by a complex system of intertwined detection biases. Survey simulators have long been used to forward-model the effects of these biases on a given population, allowing for a direct comparison to real discoveries. However, the scale and tremendous scope of the LSST requires the development of new tools. In this paper we present `Sorcha`, an open-source survey simulator written in `Python`. Designed with the scale of LSST in mind, `Sorcha` is a comprehensive survey simulator to cover all solar system small-body populations. Its flexible, modular design allows `Sorcha` to be easily adapted to other surveys by the user. The simulator is built to run both locally and on high-performance computing clusters, allowing for repeated simulation of millions to billions of objects (both real and synthetic).

Unified Astronomy Thesaurus concepts: Solar system astronomy (1529); Small Solar System bodies (1469); Astronomy software (1855); Astronomical simulations (1857)

Software reviewed by the Journal of Open Source Software JOSS

1. Introduction

The upcoming Legacy Survey of Space and Time (LSST; LSST Science Collaboration et al. 2009; Ž. Ivezić et al. 2019; F. B. Bianco et al. 2022) using the Vera C. Rubin Observatory will revolutionize solar system astronomy. This 10 yr synoptic ground-based survey boasts an unprecedented combination of six broadband optical filters, a 9.6 deg^2 field of view (FOV), and a survey depth of ∼24.5 magnitude (in the *r*-filter). Moreover, the main Wide-Fast-Deep component of the survey will provide uniform sky coverage of ∼18,000 deg^2 of the southern sky (Ž. Ivezić & the LSST Science Collaboration 2013). With cataloging the solar system as one of its four main science goals, the LSST is expected to collect on the order of a billion individual observations of solar system objects, while discovering approximately 5 million new solar system objects across all populations (LSST Science Collaboration et al. 2009). This wealth of new information surpasses any solar system survey to date in its combination of depth, sky coverage, and sheer number of observations, and this will allow us to probe the dynamics and formation history of the solar system on a scale never-before attempted. To highlight just some of the specific discoveries awaited: the LSST is expected to update the near-Earth object (NEO) inventory of objects with diameters larger than 140 m to 66%–86% from the 42% estimated to have been discovered by 2022 (P. Vereš & S. R. Chesley 2017a; R. L. Jones et al. 2018); the discovery and classification of many more active asteroids in the main asteroid belt will enable their population-level studies (L. Denneau et al. 2015; E. McLoughlin et al. 2015); additional targets may be found for NASA's (National Aeronautics and Space Administration's) ongoing Lucy mission (M. E. Schwamb et al. 2018) and for ESA's (European Space Agency's) upcoming lay-in-wait Comet Interceptor mission (C. Snodgrass & G. H. Jones 2019); and a handful of new interstellar objects are also predicted (A. Moro-Martín et al. 2009; N. V. Cook et al. 2016; T. Engelhardt et al. 2017; D. E. Trilling et al. 2017; D. Seligman & G. Laughlin 2018; W. G. Levine et al. 2021; D. J. Hoover et al. 2022).

Finally, it is expected that LSST will be paramount in settling the debate considering the existence of Planet Nine (e.g., C. A. Trujillo & S. S. Sheppard 2014; K. Batygin & M. E. Brown 2016; M. Belyakov et al. 2022; M. E. Brown et al. 2024), by discovering a sufficient number of extreme trans-Neptunian objects (eTNOs) to definitely wield support for or against the clustering of eTNOs, or even discovering the putative planet itself (D. E. Trilling et al. 2018; M. Belyakov et al. 2022; M. E. Brown et al. 2024).

While the size and scope of the LSST is unprecedented, astronomical surveys have a long history in the exploration and characterization of solar system small body populations in both the inner and outer solar system. All astronomical surveys are affected by a complex set of intertwined observational biases caused by a number of factors, including observational strategy and cadence, limiting magnitude, instrumentation effects, and poor weather/seeing. The detections from an astronomical survey therefore provide a distorted view of the actual underlying population. Survey simulators have emerged as powerful tools for assessing the impact of observational biases and aiding in the study of the target population. As long as a survey is "well characterized," in terms of having a known pointing history and calibrated image limiting magnitudes and detection efficiencies, a survey simulator can be used to replicate the varied biases of the survey upon a small body model population. This allows forward-biased model populations (objects from the input model that the simulator deemed detectable/discoverable by the survey) to be compared to survey's real discoveries.

Survey simulators have contributed to solar system science across many surveys, including the Canada-France Ecliptic Plane Survey (CFEPS; R. L. Jones et al. 2006; J. J. Kavelaars et al. 2009; J. M. Petit et al. 2011), the Palomar Distant Solar System Survey (M. E. Schwamb et al. 2010), the Outer Solar System Origins Survey (OSSOS; M. T. Bannister et al. 2016, 2018; S. M. Lawler et al. 2018a), the Dark Energy Survey (DES; P. H. Bernardinelli et al. 2022), and the DECam Ecliptic Exploration Project (DEEP; P. H. Bernardinelli et al. 2024; D. E. Trilling et al. 2024). In the outer solar system, survey simulators have been used to determine the structure of the Kuiper Belt and Centaur region (J. M. Petit et al. 2011; B. Gladman et al. 2012; M. T. Bannister et al. 2017, 2018; J. J. Kavelaars et al. 2021; J. A. Kurlander et al. 2025); explore size distributions of scattering trans-Neptunian objects (TNOs; S. M. Lawler et al. 2018b); place constraints on formation models for distant Sedna-like objects (M. E. Schwamb et al. 2010); probe the location of a color transition in the primordial TNO population (L. E. Buchanan et al. 2022); test for clustering in the orbits of high-perihelion TNOs (C. Shankman et al. 2017; P. H. Bernardinelli et al. 2020; K. J. Napier et al. 2021); and place limitations on the discoverability of Planet Nine (S. S. Sheppard & C. Trujillo 2016; D. E. Trilling et al. 2018; M. Belyakov et al. 2022; M. E. Brown et al. 2024). In the inner solar system, survey simulators have found previously unreported detections of known NEOs (J. R. Masiero et al. 2023), predicted the discovery yields of future planned space missions (J. R. Masiero et al. 2024), estimated survey effectiveness at anticipating NEO impacts as a function of impact energy (O. Lay et al. 2024), explored the size distribution, composition, and population size for NEOs and potentially hazardous asteroids (J. Luu & D. Jewitt 1989; W. F. Bottke et al. 2002; J. S. Stuart & R. P. Binzel 2004), and probed the population of asteroids interior to Earth and Venus (P. Pokorný et al. 2020; S. S. Sheppard et al. 2022).

The LSST's impact on the investigation of all solar system small body populations will be immense: the potential utility of a survey simulator for use with the LSST is clear. However, previous survey simulators have been built with a specific survey and solar system small body population in mind, and are often tailored for a particular scientific objective. Additionally, they have universally been built to manage far fewer objects than the LSST is expected to discover—OSSOS, for instance, tracks only about 800 objects (M. T. Bannister et al. 2018). In comparison, a survey simulator for the LSST must be able to calculate the on-sky positions and magnitudes for input objects of all populations, and it must do this for approximately 2 million survey exposures/pointings, in six filters, taken over 10 yr. This presents a computational challenge of much greater scale than those faced by previous survey simulators. It is evident that previous survey simulators are inadequate for the LSST's needs and cannot simply be adapted.

Considering the LSST's vast potential for advancing solar system research, there is an urgent need for a versatile and scalable survey simulator that can handle all solar system small body populations. For this purpose, we present `Sorcha`, our multipurpose, open-source solar system survey simulator for the LSST. `Sorcha` is built to be flexible, easy-to-use, and applicable to all solar system small body subpopulations; our survey simulator's modular design and configuration file allow each simulation to be highly customizable by the user for their specific needs. While `Sorcha` was built with the LSST in mind, this inherent customizability also allows it to be easily adapted to other surveys. Additionally, the simulator was designed to work at the large scale demanded by the billions of upcoming LSST observations, and simulations can be run both locally and on high-performance computing clusters. We have designed `Sorcha` to be a key community tool for solar system science with the LSST. `Sorcha` is `pip` or `conda/mamba` installable, and the code is hosted in an open repository.[17] Additional documentation is also available.[18] The software is reviewed by the Journal of Open Source Software (S. Merritt et al. 2025).

In Section 2, we discuss the particular challenges of the LSST pertaining to solar system science. Sections 3 and 4 give a general overview of `Sorcha`'s software design and documentation. How to interface with `Sorcha` is discussed in Section 5. Section 6 describes `Sorcha`'s internal ephemeris generator, while Section 7 outlines the various calculations and "filters" within `Sorcha`'s postprocessing stage designed to model the effects of the observational biases of a survey. Section 8 describes the outputs of the survey simulator. The validation tests performed to verify the output of `Sorcha` are described in Section 9, and the benchmarking and information on `Sorcha`'s performance are explained in Section 10. Utility scripts designed to perform various secondary functions useful to the user are explained in Section 11. Section 12 discusses limitations and caveats of the software. Finally, Section 13 outlines our conclusions and plans for the future of `Sorcha`.

[17] https://github.com/dirac-institute/sorcha
[18] https://sorcha.readthedocs.io

2. The LSST and the Solar System

Building a survey simulator for the LSST comes with many unique challenges that were not faced by previous solar system survey simulators. Perhaps most obvious of these challenges is that the unprecedented number of LSST exposures (a total of approximately 2 million images), for example, will ultimately lead to an equally unprecedented number of discoveries. Consequently, the number of known members of each of the solar system small body subpopulations is expected to increase by an order of magnitude over the number currently reported in the Minor Planet Center (MPC; R. L. Jones et al. 2009; LSST Science Collaboration et al. 2009; M. Solontoi et al. 2010; A. Shannon et al. 2015; T. Grav et al. 2016; K. Silsbee & S. Tremaine 2016; P. Vereš & S. R. Chesley 2017b; R. L. Jones et al. 2018; M. E. Schwamb et al. 2018; Ž. Ivezić et al. 2019; G. Fedorets et al. 2020). The LSST will also integrate multiple surveys into its observational strategy. As of early 2025, the LSST observational cadence has been mostly finalized (Ž. Ivezić & the SCOC 2021; F. Bianco & the SCOC 2022, 2024), but there are a few remaining decisions to be made in the next year. There are expected to be some final tweaks on the baseline observing strategy and rolling cadence plans, in addition to small changes necessary for optimizing science returns in the first year of operations. However, it has been decided that around 80%–90% of the observing time will be devoted to the Wide-Fast-Deep survey, a systematic survey covering 18,000 deg^2 of the southern sky with a consistent, universal observing strategy. The remaining time will be dedicated to mapping Deep Drilling Fields (small pre-selected patches of the sky which will receive significantly more observations compared to the average number of Wide-Fast-Deep visits) as well as surveys of the southern polar cap, the galactic plane, and the northern ecliptic spur (which covers the sky 10° above the ecliptic in the northern celestial hemisphere (LSST Science Collaboration et al. 2009; M. E. Schwamb et al. 2018; Ž. Ivezić & the SCOC 2021; F. Bianco & the SCOC 2022, 2024). Any survey simulator built for the LSST must thus be capable of processing tens of millions of synthetic input small bodies to identify which of the millions of exposures they each land in while tracking these orbits over 10 yr across multiple subsurveys within the LSST. This is in itself a difficult task, and to the best of our knowledge, no survey simulators built previously were required to work on such an immense computational scale required for the LSST's unprecedented combination of sky coverage and depth.

The LSST is not only focused on discovering solar system objects, however. The survey will track 5 million new solar system objects over its 10 yr duration, with each object receiving up to hundreds of observations across six broadband *ugrizy* filters; moreover, each pointing will be observed at least twice a night in two different filters. LSST's built-in follow-up will enable the study of activity throughout the solar system, such as cometary outbursts, outgassing, asteroid collisions, and rotational breakup events (R. L. Jones et al. 2009; LSST Science Collaboration et al. 2009; M. E. Schwamb et al. 2018, 2021). Furthermore, the large volume of observations for each object will facilitate the determination of rotational light curves, phase curves, and color information, significantly enhancing our understanding of the shape, rotation rate, and composition of small solar system bodies (R. L. Jones et al. 2009; LSST Science Collaboration et al. 2009; M. E. Schwamb et al. 2018). This built-in ability to monitor and follow up on discoveries is vital for LSST science, and thus, needs to be accurately modeled in a survey simulator. At its most basic level, a simulator should be capable of computing apparent magnitudes of solar system objects in all six LSST filters, accounting for phase effects on brightness to accurately simulate phase curves using the user's preferred phase function model; it would furthermore be useful to apply the effects of activity or rotation to the light curves generated in survey simulations.

In addition to the expected instrumental and observational biases linked to the optics and observational strategy of the survey, the solar system object detection and discovery pipeline is also to be considered. The Rubin Solar System Processing (SSP) pipeline (J. Myers et al. 2013; M. Jurić et al. 2020), currently under development, is a fully automated discovery pipeline that works to link observations of transient sources to discover previously unknown moving solar system objects. In this technique, two or more observations of point sources in a single night, suspected to be the same unknown object moving across the FOV, are linked into a "tracklet"; these tracklets are then further linked into a "track" spanning multiple nights, and a preliminary orbit can thus be calculated (J. Kubica et al. 2007). This process is described in more detail in Section 7.10. In order to mimic the operation of the LSST, software such as `Sorcha` must have the option to simulate the effects of this linking procedure, to determine which objects of the input population of interest are discovered by the SSP pipeline. In addition, the SSP pipeline is only expected to link objects out to a maximum of ∼200 au (M. E. Schwamb et al. 2023). Users of `Sorcha` and the LSST interested in, for example, Planet Nine, may be using their own bespoke linking software to detect distant objects beyond the SSP's ∼200 au limit. Ideally, `Sorcha` should be able to simulate both the SSP pipeline and more user-specific linking software.

The nature of data release for the LSST also provides some interesting considerations for a survey simulator. The distribution of solar system data for LSST will occur through two main avenues, prompt data processing and annual releases, as described in the Rubin Data Products Definition Document (M. Jurić et al. 2021). The prompt data products will include all data observed on a given night, and will be released on a daily basis, with the MPC being one of the distribution channels for solar system objects. During the night, the Rubin prompt processing pipelines will also send out alerts to the Rubin alert stream when known solar system objects are matched to transient sources identified in LSSTCam exposures. This is expected to occur within 60 s of image readout. Annual data releases, meanwhile, will provide an annual catalog of solar system discoveries, regenerated and updated using only LSST data. A fast, easy-to-use survey simulator will allow users to predict which prompt products will contain their objects of interest or compare the results of many simulations of many different models to the annual data release catalogs.

The LSST's ambitious goals, from tracking millions of new solar system objects to integrating diverse observational strategies, require a simulator that not only handles the vast computational load but also offers flexibility and precision in modeling the intricate dynamics of small body populations. `Sorcha` must therefore rise to meet these demands by providing detailed, customizable simulations that can account for the LSST's unique capabilities, including its frequent

observations, varied filters, and the sophisticated linking procedures of the SSP pipeline.

3. Software Design Overview and Implementation Choices

High-level modules of Sorcha are written in Python, making use of a C/C++ backend where possible. Our focus throughout development of Sorcha has been to make the survey simulator as flexible and easy-to-use as possible. Sorcha's basic structure consists of two main halves, illustrated in Figure 1: ephemeris generation and postprocessing. Ephemeris generation, described further in Section 6, is the process by which Sorcha calculates the on-sky position of an input set of small bodies and matches them to the survey observations in which they appear. Postprocessing, covered in Section 7, applies a number of sequential steps designed to calculate the brightness of these input small bodies and apply the optical and instrumental effects of the survey to each individual observation. Sorcha allows either of these stages to be run in stand-alone mode. One might, for instance, use the built-in ephemeris generator by itself to simply calculate which survey observations a simulated small body population will appear in. Similarly, it is possible to only use the components of postprocessing that will calculate the predicted brightness of those objects; if a user prefers pre-generated ephemerides from other software or a previous Sorcha run, only the postprocessing steps may be desirable and can be performed independently. Together, these two steps of ephemeris generation and postprocessing complete the full survey simulation for a target small body population.

Details about Sorcha's repository setup, software testing, and package architecture and deployment are described in Appendix A. Sorcha has been designed to facilitate easy customization both for new and advanced users. For the majority of users, Sorcha's configuration file offers everything needed to tailor the simulation to their particular science case. In this file, described further in Section 5.1, users can easily customize the parameters of every aspect of the simulation and switch off or on the available functions. This configuration file is also copied in its entirety to the log file of every Sorcha run, so that the user may easily locate and even copy out the specific configuration for that run. For users who wish to apply the effects of rotation or activity to their objects, we provide an additional package, sorcha-add-ons, designed to interface easily with Sorcha. This package, described in Section 7.3, contains simple models to simulate activity or rotational light curves, and also allows the user to simply add their own custom classes modeling these effects. Finally, Sorcha was written to be highly modular and human-readable. The more advanced user should therefore find it simple to edit or replace any function in the simulation to suit their own needs.

Attention was also paid to the many ways in which a user might wish to run Sorcha. Getting started with Sorcha on a local machine or laptop is straightforward. Sorcha comes bundled with demonstration input files and a demo command, both easily accessible via the command line interface (CLI), and example configuration files that are designed for simulating the LSST are also available via the same method. A quick-start guide is presented in Appendix B. However, it is understood that many users will be using HPC clusters to perform multiple runs of Sorcha on their simulated populations. Sorcha has thus been built and extensively tested for HPC, and comes with useful utility scripts designed to aid the user in collating and exploring the results of multiple runs. These are explained further in Section 11. Additionally, all inputs to Sorcha can be "chunked," where Sorcha reads in and iterates over segments of the input files in sequence, with a chunk size controlled by the configuration file to enable the user to tailor Sorcha runs for their own specific setup. This feature—especially useful in the case of very large input files—reduces the memory footprint of Sorcha simulations.

4. Documentation

Extensive documentation is available (see footnote 18). The documentation is automatically compiled in the Sorcha Github (see footnote 17) repository and published on ReadTheDocs using the Sphinx[19] generator. The documentation includes guides on installation, getting started with Sorcha, and examples on how to run the software. The docstrings[20] within the Sorcha code base are also extracted automatically in the API Reference section.[21]

5. Inputs

Sorcha is executed as a stand-alone Python package requiring a collection of mandatory and optional input files. Behavior specific to the execution and fine-tuning of the simulation is controlled through the configuration file, whereas more general parameters, such as paths to the input small body population files and output locations, are controlled through the CLI. This design allows for efficient simulations using HPC facilities: a large number of differing input small body populations can thus be run concurrently with only minor changes to the command line call, while the overall simulation-level behavior can be easily maintained using a single configuration file.

Sorcha's input files are divided into three categories: the aforementioned configuration file, which sets aspects of Sorcha's behavior; the files defining the input synthetic small body population to be run through Sorcha; and the files defining the survey, camera, and telescope. To enable ephemeris generation, Sorcha also requires auxiliary data and resource files to be downloaded once per installation: these are described further in Section A.4. An overview of the input files can be seen in Figure 2.

5.1. Configuration File

The configuration file acts as a main control file for Sorcha, allowing the user to customize how Sorcha runs. This configuration file is divided into sections, each containing a number of customizable parameters. Example configuration files can be found in Appendix C. Throughout Sections 5, 6, 7, and 8 of this paper, the configuration parameters that are used to control the aspect of Sorcha under discussion will be noted under a "relevant configuration file parameters" heading as an easy reference for the user.

[19] https://www.sphinx-doc.org/en/master/index.html

[20] We use the NumPy(C. R. Harris et al. 2020) style for docstrings: see https://numpydoc.readthedocs.io/en/latest/format.html.

[21] https://sorcha.readthedocs.io/en/latest/autoapi/index.html

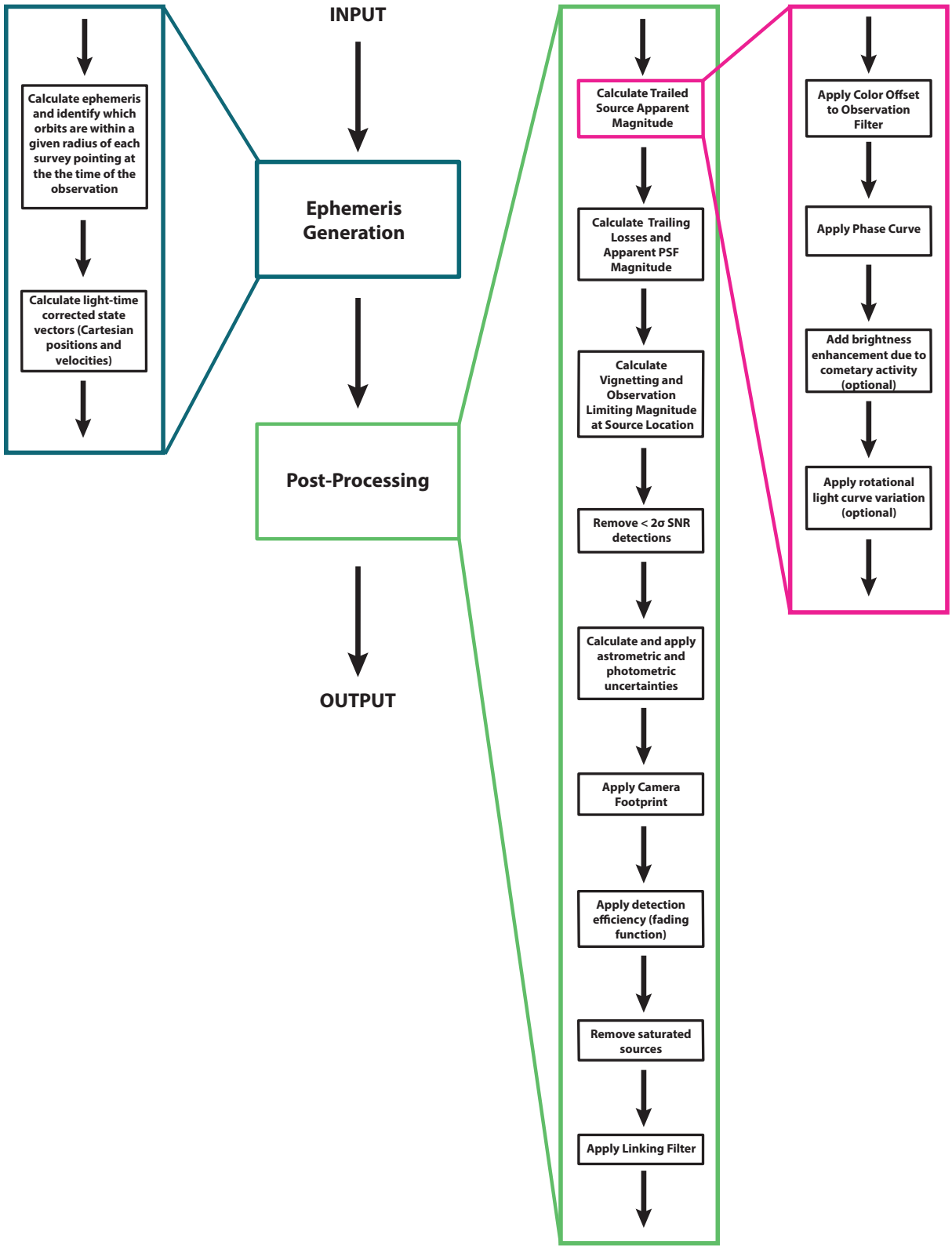

Figure 1. The main `Sorcha` workflow. `Sorcha` is split into two stages. The ephemeris generation stage (described in Section 6) is in gray, and the postprocessing stage (described in Section 7) is in green and magenta. We note for the reader that this workflow is for simulating what the LSST is expected to discover. We note there are a small number of additional functions/filters available (described in Section 7.11) that can be applied to produce different desired outputs of the simulated small body detections and discoveries based on the user's desired science case.

5.2. Population Input Files

The population input files describe the synthetic solar system object population to be run through `Sorcha`, providing the information needed to compute the on-sky location/state vector and apparent magnitude of the objects at any given time. These should be supplied as text files, with columns separated by either white spaces or commas, and with the expected file format given

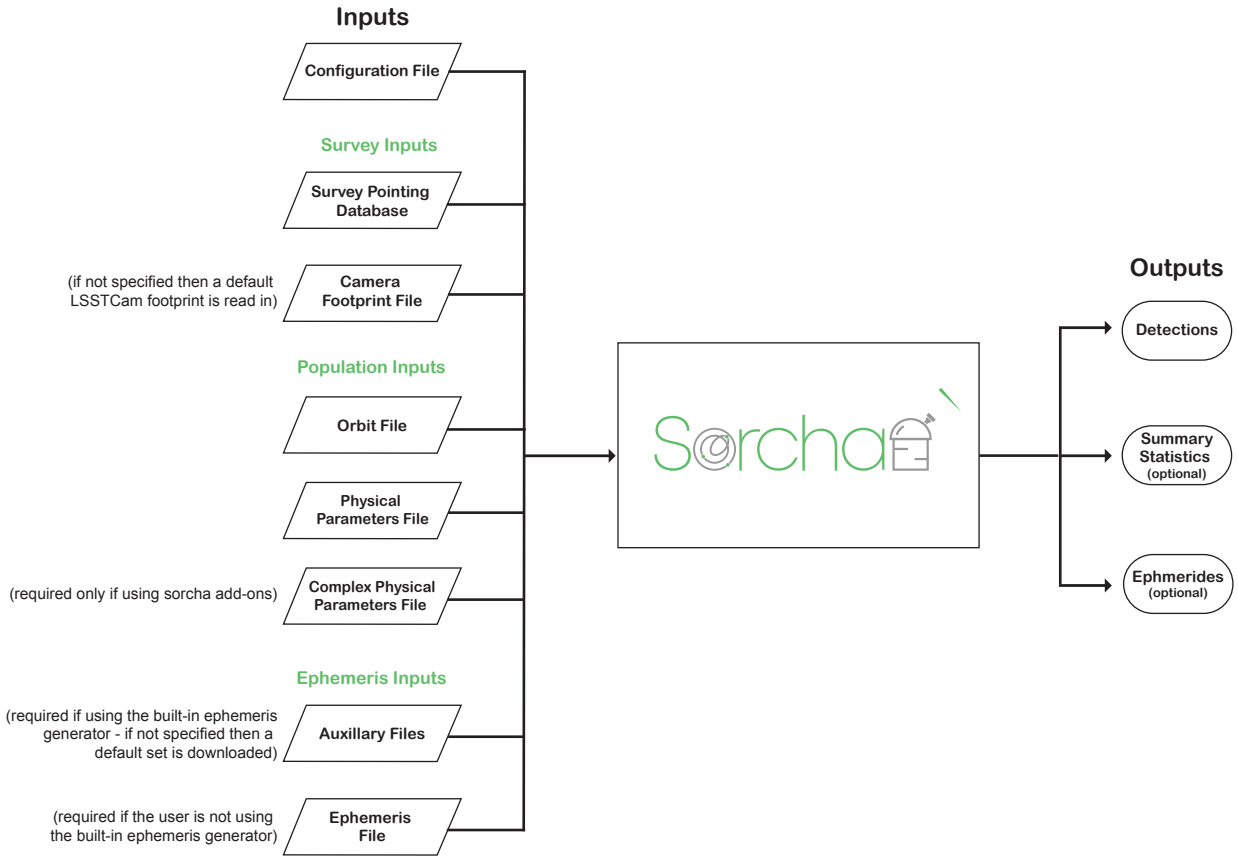

Figure 2. `Sorcha` inputs and outputs. This figure provides an overview of the input files that are ingested by `Sorcha` and the possible resulting simulation outputs.

in the configuration file. The files should contain one row for each synthetic object, and the first column must be `ObjID`, which contains a unique string identifier for each object: the remaining columns can be in any order.

5.2.1. Orbit File

The orbit file defines the orbital parameters of each of the simulated objects and is used by `Sorcha` to compute their position on-sky at any given time. The required column headings and input units for each format can be seen in Table 1. These orbital parameters can be supplied as heliocentric or barycentric defined orbits. The six orbital elements can be defined in Keplerian, cometary, or Cartesian format. An epoch column must also be included that defines the moment in time used as the reference point for the orbital elements.

Relevant configuration file parameters: `aux_format`

5.2.2. Physical Parameters File

The physical parameters file is used to calculate the apparent magnitudes of the input small body population in the filter of each observation, and thus contains the absolute magnitude in a set filter (which we refer to as the *main filter*), photometric colors (provided in the form of offsets with regards to main filter), and phase function parameters for each of the input objects. `Sorcha` enables the calculation of the phase function using several different models, as described in Section 7.2, and additionally allows the user to specify filter-dependent phase function parameters. Due to this flexibility, the required columns for this file differ depending on the selected phase function. The column headings for the physical parameters file are detailed in Table 2.

Relevant configuration file parameters: `aux_format`, `phase_function`, `observing_filters`

5.2.3. Complex Physical Parameters File

`Sorcha` has the ability to implement cometary activity and rotational light curves. Any additional parameters required for these models should be supplied in the optional complex physical parameters file. This behavior is described further in Section 7.3.

Relevant configuration file parameters: `aux_format`

5.2.4. Ephemeris File

The first step in a `Sorcha` simulation is the determination of which of the input small body population will appear in or near the camera FOV for any given exposure. This process of ephemeris generation, as described in Section 6, requires the numerical N-body integration of each object's orbit over the course of the survey, and is thus the most computationally intensive part of `Sorcha`. For this reason, the user is given the option to supply their own ephemeris file, either from a previous `Sorcha` run or from an external ephemeris generator. This is useful in situations in which the user wishes to re-run `Sorcha` using the same object ephemerides but with a differing configuration or new physical parameters for the synthetic objects. The columns required in the ephemeris file are given in Table 3. Given the very large sizes of the ephemeris output, `Sorcha` has the capability of ingesting

Table 1
Orbit File Column Headings

Heading	Description
	Cometary Format
ObjID	Unique object identifier for each input object (string)
FORMAT	Orbit format string (COM for heliocentric or BCOM for barycentric)
q	Perihelion distance (au)
e	Eccentricity
inc	Inclination (degrees)
node	Longitude of the ascending node (degrees)
argPeri	Argument of perihelion (degrees)
t_p_MJD_TDB[a]	Time of periapsis passage (MJD)
epochMJD_TDB[a]	Epoch (MJD)
	Keplerian Format
ObjID	Unique object identifier for each input object (string)
FORMAT	Orbit format string (KEP for heliocentric or BKEP for barycentric)
a	Semimajor axis (au)
e	Eccentricity
inc	Inclination (degrees)
node	Longitude of the ascending node (degrees)
argPeri	Argument of perihelion (degrees)
ma	Mean anomaly (degrees)
epochMJD_TDB[a]	Epoch (MJD)
	Cartesian Format
ObjID	Unique object identifier for each input object (string)
FORMAT	Orbit format string (CART for heliocentric or BCART for barycentric)
x[b]	The x-component of the heliocentric/barycentric position vector (au)
y[b]	The y-component of the heliocentric/barycentric position vector (au)
z[b]	The z-component of the heliocentric/barycentric position vector (au)
xdot[b]	The x-component of the heliocentric/barycentric velocity vector (au day^{-1})
ydot[b]	The y-component of the heliocentric/barycentric velocity vector (au day^{-1})
zdot[b]	The z-component of the heliocentric/barycentric velocity vector (au day^{-1})
epochMJD_TDB[a]	Epoch (MJD)

Notes.
Orbital elements can either be heliocentric or barycentric, independent of format.
Sorcha requires that the input orbit file have the same format/orbit type throughout.
[a] All times/epochs specified in the orbit file are expected to be defined as MJD (Modified Julian Date) in TDB (Barycentric Dynamical Time).
[b] All Cartesian coordinates and velocities are defined in the ecliptic frame.

Table 2
Physical Parameters File Column Headings

Heading(s)	Description
ObjID	Unique object identifier for each input object (string)
H_x	Absolute magnitude of the object in the main filter, where the main filter is x
u-x, g-x, etc	Photometric colors in the relevant survey filters
GS **OR** GS_u, GS_g, etc	Phase parameter for HG model
G1 **OR** G1_u, G1_g, etc	First phase parameter for HG_1G_2 model
G2 **OR** G2_u, G2_g, etc	Second phase parameter for HG_1G_2 model
G12 **OR** G12_u, G12_g, etc	Phase parameter for HG_{12} model
S **OR** S_u, S_g, etc	Phase curve parameter for linear model

Note. The letter x here represents the (main) filter in which H is defined. The main filter and the phase function are required to be specified in the Sorcha configuration file. The user also must specify in the configuration file whether all survey observations or a subset of observations in a smaller set of observing filters should be used in the Sorcha simulation. The phase parameter columns required will depend both on the phase function model the user has selected, and whether they are using filter-dependent phase parameters.

Table 3
Ephemeris File Column Headings

Heading	Description
ObjID	Object identifier for each input small body simulated (string)
FieldID	Observation pointing field identifier
fieldMJD_TAI[a]	Observation Mean Julian Date (MJD) in TAI (International Atomic Time)
fieldJD_TDB[a]	Observation Julian Date in TDB (Barycentric Dynamical Time)
Range_LTC_km	Light-time-corrected object-observer distance (km)
RangeRate_LTC_km_s	Light-time-corrected rate of change of the object-observer distance (km s^{-1})
RA_deg	Object R.A. (degrees)
RARateCosDec_deg_day	Object R.A. rate of motion multiplied by cos(Dec) (deg day^{-1})
Dec_deg	Object decl. (degrees)
DecRate_deg_day	Object decl. rate of motion (deg day^{-1})
Obj_Sun_x_LTC_km[b]	Light-time-corrected Cartesian *X*-component of the object's heliocentric position (km)
Obj_Sun_y_LTC_km[b]	Light-time-corrected Cartesian *Y*-component of the object's heliocentric position (km)
Obj_Sun_z_LTC_km[b]	Light-time-corrected Cartesian *Z*-component of the object's heliocentric position (km)
Obj_Sun_vx_LTC_km_s[b]	Light-time-corrected Cartesian *X*-component of the object's heliocentric velocity (km s^{-1})
Obj_Sun_vy_LTC_km_s[b]	Light-time-corrected Cartesian *Y*-component of the object's heliocentric velocity (km s^{-1})
Obj_Sun_vz_LTC_km_s[b]	Light-time-corrected Cartesian *Z*-component of the object's heliocentric velocity (km s^{-1})
Obs_Sun_x_km[b]	Cartesian *X*-component of the observer's heliocentric position (km)
Obs_Sun_y_km[b]	Cartesian *Y*-component of the observer's heliocentric position (km)
Obs_Sun_z_km[b]	Cartesian *Z*-component of the observer's heliocentric position (km)
Obs_Sun_vx_km_s[b]	Cartesian *X*-component of the observer's heliocentric velocity (km s^{-1})
Obs_Sun_vy_km_s[b]	Cartesian *Y*-component of the observer's heliocentric velocity (km s^{-1})
Obs_Sun_vz_km_s[b]	Cartesian *Z*-component of the observer's heliocentric velocity (km s^{-1})
phase_deg	Phase angle between the Sun, object, and observer (degrees)

Notes. Light-time corrected distances, velocities, and vectors account for the finite speed of light between the light being reflected from the object's surface and observed by the telescope.
[a] All times are reported for the midpoint of the simulated survey observation.
[b] Defined in the ecliptic frame.

ephemerides in HDF5 (Hierarchical Data Format version 5) format.

Relevant configuration file parameters: ephemerides_type, eph_format

5.3. Survey and Telescope Input Files

The survey and telescope input files describe the observational schedule and camera architecture of the survey to be simulated, independent of the input small body population. While Sorcha was built with the LSST in mind, these files allow any survey to be simulated if its pointing history is known or simulated and its camera architecture can be described.

5.3.1. The Pointing Database

The pointing database is a SQLite database containing metadata about the survey pointing history, image depth, and observing conditions. Sorcha uses the observation mid-time, center-of-field R.A., center-of-field decl., rotation angle of the camera, observation 5σ limiting magnitude at the center of the FOV, filter, and seeing information to evaluate the effects of various observational biases.

Sorcha is currently designed to work with the simulated pointing databases generated by the rubin_sim package (A. J. Connolly et al. 2014; P. Yoachim et al. 2023) and the Rubin Observatory scheduler, rubin_scheduler (F. Delgado et al. 2014; P. Yoachim et al. 2024).[22] This software is being used to simulate the cadence and on-sky coverage of differing survey strategies. A description of an early version of this Python software can be found in A. J. Connolly et al. (2014), F. Delgado et al. (2014), and R. L. Jones et al. (2018); further details can be found in M. E. Schwamb et al. (2023). Once the LSST has begun operations, future versions of Sorcha will be modified to read the actual pointing history.

Relevant configuration file parameters: pointing_sql_query

5.3.2. The Camera Footprint File

The architecture of the survey camera, including the shape of its footprint and any gaps between its detectors, is relevant for object detectability. By ingesting a map of the camera detector layout and taking into account field rotation from the pointing database, Sorcha can track where the input objects land on the detectors and remove those that have fallen into gaps between the detectors and those fall on the very edges of the CCD, where they may not be detected (see Section 7.7 for more details). Sorcha comes with a representation of the LSST Camera (LSSTCam) architecture already installed: this is used if the full camera footprint is requested in the configuration file and no alternative camera footprint file location is supplied. However, the user can also supply their own comma-separated text file describing the survey camera footprint in the configuration file. The expected columns for this file are given in Table 4.

Relevant configuration file parameters: camera_model, footprint_path

[22] The most up-to-date versions of the rubin_sim/rubin_scheduler cadence simulations can be found at https://s3df.slac.stanford.edu/data/rubin/sim-data/.

Table 4
Camera Footprint File Column Headings

Heading	Description
`detector`	Detector identifier (integer)
x	*x*-position of the detector corner on the focal plane (float)
y	*y*-position of the detector corner on the focal plane (float)

Note. The *x*- and *y*-values are unitless and are, respectively, equal to tan(ra) and tan(dec), where "ra" and "dec" are the vertical and horizontal angles of the points from the center of the sphere tangent to the origin in the focal plane, that is, whose origin is at the nominal boresight angle of the focal plane. For each detector, all four corners must be specified in the camera footprint file.

6. Ephemeris Generation

After ingestion and validation checks of the input files, `Sorcha` proceeds to ephemeris generation (if requested by the user). The generation of ephemerides was one of the most challenging steps within `Sorcha` to optimize as the time span of the survey, limiting magnitude, and sky coverage of the LSST are unprecedented. Traditional brute force methods of calculating the on-sky positions for every input orbit at every survey observation are too slow when scaled to LSST discovery rates (even when deploying on high-performance computing clusters). The LSST is estimated to generate approximately 1 billion photometric/astrometric detections of approximately 5 million solar system objects (LSST Science Collaboration et al. 2009). To simulate the LSST survey detections (especially the main-belt asteroids, MBAs) requires simulating a significantly larger population of synthetic objects than will be detected, with all of those orbits needing to be run through `Sorcha` with ephemerides predicted. In `Sorcha`, this barrier has been overcome by a combination of predicting positions at set times per night, on-sky grids, and interpolation. Full details about the implemented algorithm are described in M. J. Holman et al. (2025); we provide a brief summary below. Additional details on how the auxiliary data files required by the ephemeris generator are downloaded and stored can be found in Appendix A.4.

`Sorcha`'s internal ephemeris generator iterates through each survey observation ingested from the pointing database, determining the heliocentric Cartesian positions for every model orbit at the time of the observation, converting these positions to R.A. and decl. on-sky at the observatory's location, and identifying which of the input synthetic small bodies are within a given radius of the observation's pointing center. This radius is specified by the user in the configuration file (the `ar_ang_fov` + `ar_fov_buffer`) and should be larger than the survey's camera FOV, as later on in the simulation, `Sorcha` can perform a more refined filtering using a model or approximation for the camera footprint (see Section 7.7). The user also provides the relevant survey's MPC observatory code in the configuration file. The R.A., decl., and heliocentric state vector (Cartesian position and velocity) are returned for each synthetic small body within the search radius of the observation pointing. Using the location of the observatory, the phase angle and the light-time-corrected state vector are also computed and returned, as these will be later used when calculating the apparent magnitudes as described in Section 7.2. All of the values that are calculated by the ephemeris generator and passed on to the postprocessing stage can be found in Table 3.

`Sorcha`'s ephemeris generator is powered by ASSIST (M. J. Holman et al. 2023), an open-source `Python` and C99 software package for producing ephemeris-quality integrations of solar system test particles. ASSIST is an extension of the REBOUND *N*-body package (H. Rein & S. F. Liu 2012) and uses its IAS15 (Gauss-Radau integrator with Adaptive Step-size control, 15th order; H. Rein & D. S. Spiegel 2015) integrator to integrate test particle trajectories in the field of the Sun, Moon, planets, and 16 massive asteroids and dwarf planets (listed in Table 5). The positions of the masses come from the JPL (Jet Propulsion Laboratory) DE440/441 ephemeris and its associated asteroid perturber file (see Section A.4). ASSIST includes the most significant gravitational harmonics and general relativistic corrections, and it also accounts for position- and velocity-dependent nongravitational effects. With this level of detail, ASSIST closely matches the results of JPL's small bodies integrator, a current standard of reference. Note that including any of the 16 bodies in a `Sorcha` simulation will yield inaccurate results due to gravitational self-perturbation. A user-calculated external ephemeris (see Section 5.2.4) could be used as input into a `Sorcha` simulation if these bodies do need to be included. Straightforward modifications to `Sorcha`'s internal ephemeris generator would be needed to initialize ASSIST with an additional massive perturber such as some of the recently proposed theories of a planetary-mass beyond Neptune (C. A. Trujillo & S. S. Sheppard 2014; K. Batygin & M. E. Brown 2016; S. S. Sheppard & C. Trujillo 2016; K. Batygin et al. 2019; M. E. Brown & K. Batygin 2019; M. E. Brown & K. Batygin 2021; W. J. Oldroyd & C. A. Trujillo 2021; P. S. Lykawka & T. Ito 2023). We note that we have designed this part of the software such that it would be straightforward in future versions of `Sorcha` to potentially incorporate other *N*-body integrators as additional engines for the generator.

Relevant configuration file parameters: `ephemeri-des_type`, `ar_ang_fov`, `ar_fov_buffer`, `ar_picket`, `ar_obs_code`, `ar_healpix_order`, `ar_n_sub_intervals`

7. Postprocessing

Once the ephemerides have been generated or read in from an external file, `Sorcha` moves on to the second phase, which we call postprocessing. For each of the input objects, `Sorcha` goes through the potential detections identified in the ephemeris generation step and performs a series of calculations and assessments in the postprocessing stage to determine whether the objects would have been detectable as a source in the survey images and would have later been identified as a moving solar system object. `Sorcha` combines the calculated ephemerides with the physical parameters of the input objects and the effects originating from the telescope optics and automated data processing pipelines. It then uses that to predict the apparent magnitude and eventual discoverability of the input small body population. This is performed by applying a

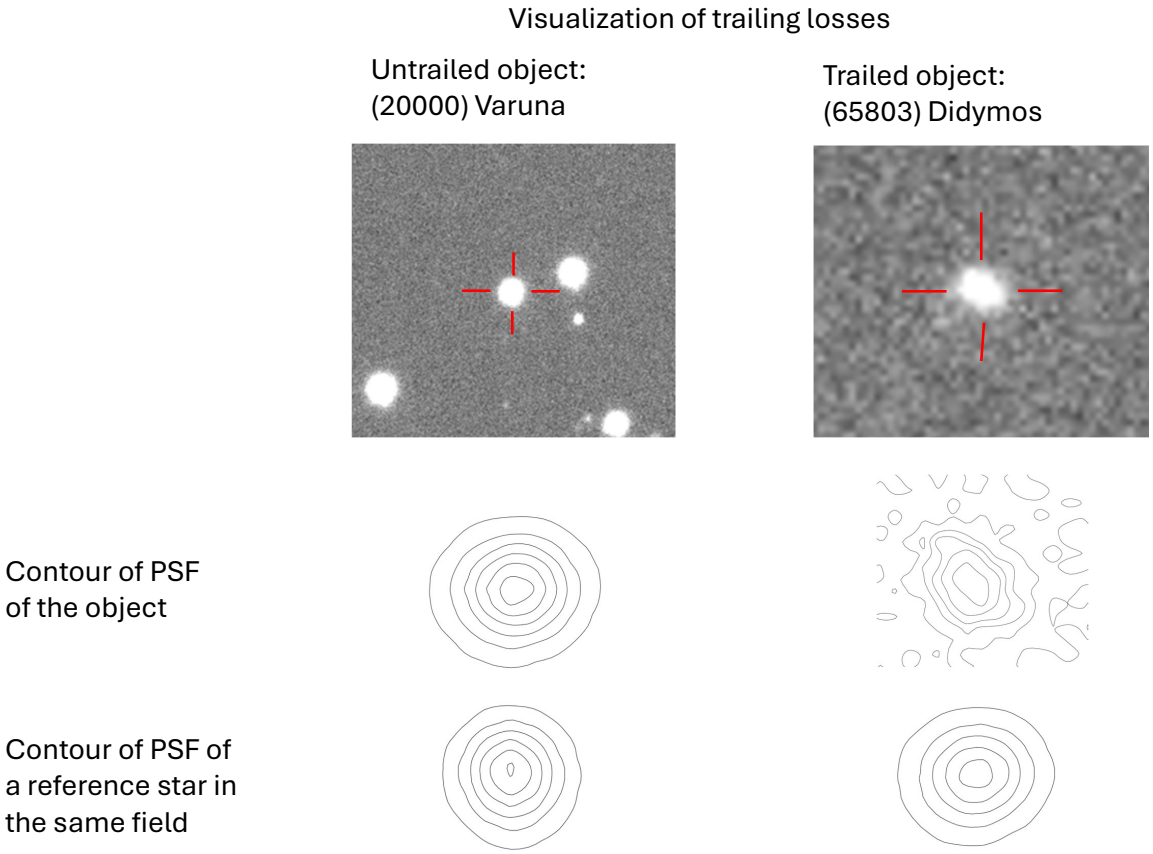

Figure 3. The trailing of moving solar system objects visualized with Dark Energy Survey (DES; T. M. C. Abbott et al. 2021) observations. The top two rows show the image cutouts and PSF (point-spread function) contours of example solar system objects (identified by the red crosshairs in the cutouts). The last row shows the PSF contours for representative reference stars within the same images. Slow-moving TNO (20000) Varuna has a stellar-like PSF (shown on the left) and fast-moving NEO (65803) Didymos has an extended significantly trailed PSF, as shown on the right. Dark Energy Camera (DECam; B. Flaugher et al. 2015) data was selected because of the expected similarity to the LSSTCam images (Ž. Ivezić et al. 2019; F. B. Bianco et al. 2022).

Table 5
List of MBA and Dwarf Planet Perturbers Included within `Sorcha`'s Ephemeris Generation Stage

Asteroid MPC Designation	Asteroid MPC Designation
(107) Camilla	(704) Interamnia
(1) Ceres	(7) Iris
(65) Cybele	(3) Juno
(511) Davida	(2) Pallas
(15) Eunomia	(16) Psyche
(31) Euphrosyne	(87) Sylvia
(52) Europa	(88) Thisbe
(10) Hygiea	(4) Vesta

Note. As these objects and their masses are included with the ASSIST integrations performed during the ephemeris generation stage. Poor predictions will result if these specific main-belt asteroids (MBAs) and dwarf planets are simulated within `Sorcha` when using the simulator's internal ephemeris engine.

sequential series of modular steps to every potential detection of the input objects, each of which calculates a particular attribute (such as apparent brightness) or simulates a specific optical/structural/software phenomenon/characteristic of the astronomical survey. Each of these steps can be controlled or modified to suit the user's needs via the configuration file, facilitating a high level of flexibility. An overview of the postprocessing workflow for simulating what the LSST should find is shown in Figure 1 in green and magenta; the sections below describe each stage of postprocessing in detail, including the relevant configuration file parameters and code functions. Each of these functions has been separately validated to ensure correct operation: validation notebooks for each function can be found in the additional documentation (see footnote 18) and within the main `Sorcha` repository on `GitHub` (see footnote 17).

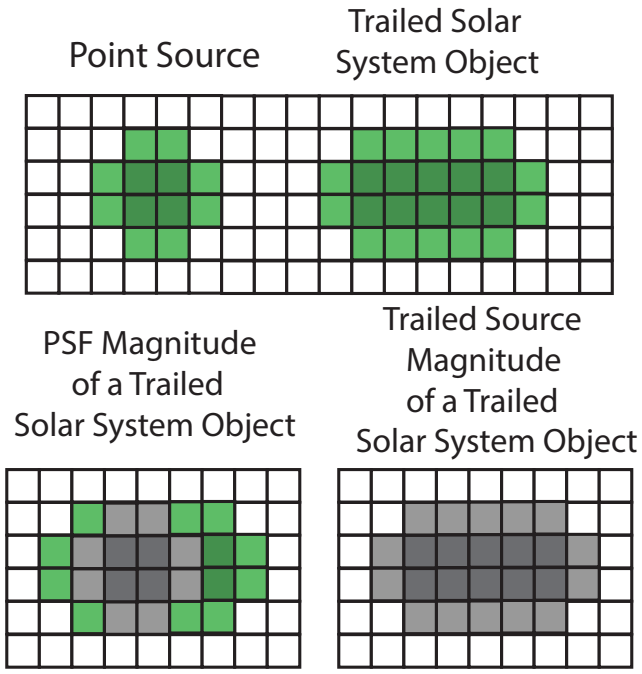

Figure 4. A cartoon schematic depicting the difference between how the trailed source magnitude and the PSF magnitude for a moving solar system object observed on an LSST image are estimated. The Rubin Data Management source detection pipeline, the Difference Image Analysis (DIA) pipeline, uses PSF filter matching to find sources on the image. This pipeline will measure the PSF magnitude. Only transient sources identified by the DIA pipeline will be sent on to the Rubin Solar System Processing (SSP) pipelines to search for moving objects. The SSP pipelines will report the trailed source magnitude.

7.1. Rubin Apparent Magnitudes

In order to assess the detectability of the input synthetic small body population in LSST observations, `Sorcha` calculates two apparent magnitudes (which we refer to as the *trailed source magnitude* and the *PSF, point-spread function, magnitude*). Unlike background stars, a moving solar system object may leave a streak on the detector, depending on the object's on-sky velocity, the image exposure time, the camera's pixel resolution, and the image quality, resulting in an extended PSF ("trail") spanning more pixels than a point source would. This effect is shown in Figure 3. The *trailed source magnitude* (described in Section 7.2) is the true apparent magnitude of the object (and the apparent magnitude that will be eventually calculated by the Rubin SSP pipelines), whereas the *PSF magnitude* (discussed in Section 7.4) is the effective brightness of the solar system object measured by the Rubin source detection algorithm (Difference Image Analysis; DIA). The PSF magnitude accounts for the loss in flux from the Rubin source detection algorithms, which use stellar PSF-like matched filter to identify transient sources in the survey's difference images. The difference between these two magnitudes is illustrated in Figure 4. For the LSST and its expected $\sim$30 s exposures, objects whose rates of motion are 0.16 deg day^{-1} will trail 0.″2, the size of an LSSTCam pixel. With a median expected seeing of 0.″7 $\approx$ 3.5 pixels (Ž. Ivezić et al. 2019), we then expect that objects with rates of motion $\gtrsim$0.4 deg day^{-1}, such as the closest MBAs and NEOs, will have trails roughly comparable to their PSF FWHM, whereas the most distant objects (e.g., TNOs) will mostly appear as point sources (see Figure 3). The longer the trail, the larger the difference between the trailed source magnitude and the PSF magnitude. For TNOs and more distant objects, the PSF magnitude and the trailed source magnitude will be nearly identical because of their slow rates.

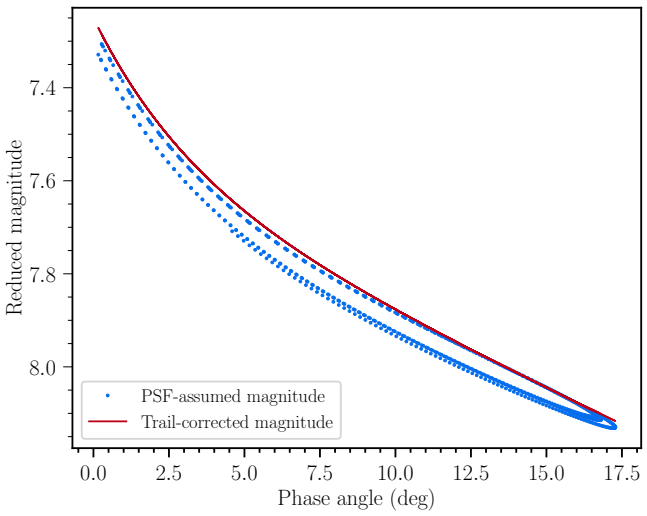

Figure 5. Demonstration of the difference between the trailed source magnitude and the PSF magnitude. We plot the reduced magnitude derived from the PSF magnitude (blue dots) and the trailed source magnitude (red curve), simulated for asteroid (24) Themis. The PSF magnitude measures less flux because it is not sampling using the full image trail from the simulated detection, and therefore, a clear flux loss can be seen in the reduced PSF magnitude, which varies with the on-sky velocity of Themis, compared to the reduced magnitude calculated from the trailed source magnitude. For any photometric analyses of simulated detections/discoveries output by `Sorcha`, the user should use the reported trailed source magnitude.

When analyzing the detections and discoveries output from a `Sorcha` simulation, we caution the reader to only use the trailed source magnitude. Using the PSF magnitude will give incorrect results because it is missing some of the object's flux on the LSST image. To demonstrate this effect, we have simulated a toy scenario with 2 yr of daily (between 2021 and 2023), single band, noiseless observations of asteroid (24) Themis, from the LSST site. We use the *HG* photometric model (E. Bowell et al. 1989; as described in Section 7.2), assuming $H = 7.25$ and $G = 0.19$. We then estimate the PSF magnitude from the prescription in Section 7.4. The reduced magnitude (distance-corrected magnitude) is calculated for each scenario, as shown in Figure 5. The reduced magnitudes estimated from the PSF magnitudes do not follow the shape of the absolute magnitude as a function of phase angle expected from the *HG* model. To quantify this trend, we have performed a simple least-squares fit to this model, using an independent (i.e., not the one used by `Sorcha`) implementation of the *HG* model. We recover, in the case of the PSF magnitude, $H = 7.29$, $G = 0.22$, demonstrating a bias toward fainter magnitudes, as expected. The same procedure on the reduced magnitudes estimated from the trailed source magnitude leads to the expected values of H and G to high precision.

7.2. Calculating the Trailed Source Magnitude

`Sorcha` first computes the trailed source magnitude and then later calculates the discovery trailing losses and the resulting PSF magnitude, described later in Section 7.4. The trailed source magnitude is the measured apparent magnitude when all pixels in the trail are included. When simulating a nonactive body (e.g., no cometary activity or production of a

dust tail/coma), we calculate the trailed source magnitude $m_{\mathrm{trailedsource},i}$ in a given filter i for a small body at a time t as:

$$m_{\mathrm{trailedsource},i} = H_x + 5\log_{10}(r\Delta) + \Phi(\alpha, i) + C(i - x) + \delta(i, t) \quad (1)$$

where:

1. r and Δ, respectively, are the heliocentric and geocentric distance to the object;
2. α is the phase angle (the angle between the Sun, the object, and the observer);
3. H_x is the absolute magnitude of the object in photometric filter x (referred to in `Sorcha` as the *main filter*), defined as the trailed source magnitude at opposition at the distance of 1 au both from the Sun and the observer at $\alpha = 0$;
4. $\Phi(\alpha, i)$ is the disk-integrated phase function evaluated at phase angle α for filter i;
5. $C(i - x)$ is the $(i - x)$ color of the object to convert the apparent magnitude from the main filter (x) to the observed filter (i);
6. $\delta(i, t)$ is an optional contribution from the rotational light curve in filter i at time t.

The phase function ($\Phi(\alpha, i)$) model applied by `Sorcha` is specified by the user in the configuration file, and we use the implementations available in the small body `Python` package `sbpy` (M. Mommert et al. 2019). The models currently available for estimating the phase curve contribution are:

1. none (no phase function applied)
2. *HG* (E. Bowell et al. 1989)
3. HG_1G_2 (K. Muinonen et al. 2010)
4. Modified HG_{12} (A. Penttilä et al. 2016)
5. linear

The relevant phase curve parameters are specified per object in the physical parameters file. The phase function is filter-dependent (e.g., A. Alvarez-Candal et al. 2022; M. M. Dobson et al. 2023; J. E. Robinson et al. 2024). The user can either provide filter-specific phase curve parameters or single values applied to all survey filters evaluated (see Section 5.2.2). We allow the user to optionally supply their own self-developed models for rotation effects, $\delta(i, t)$. This functionality is described further in Section 7.3.

In the case of simulating active objects, `Sorcha` performs one more step. It assumes that the trailed source magnitude calculated in Equation (1) is the apparent magnitude of the nucleus ($m_{\mathrm{nucleustrailedsource},i}$) and the trailed source magnitude of the nucleus and coma/tail together is estimated as

$$m_{\mathrm{trailedsource},i} = A(r, \Delta, \alpha, i, t, m_{\mathrm{nucleustrailedsource},i}) \quad (2)$$

where $A(r, \Delta, \alpha, i, t, m_{\mathrm{nucleustrailedsource},i})$ is provided by the user. The user's self-developed models for cometary activity are expected to compute the flux contribution from any coma/tail and combine this flux with the contribution from the nucleus. This functionality is described further in the next section (Section 7.3).

Relevant configuration file parameters: `phase_function`, `observing_filters`

Code function: `PPCalculateApparentMagnitude`

7.3. Optional Activity and Light-curve Models

To support future extensions without opening the original source code to external contribution, `Sorcha` makes use of a technique that allows users to use their own models for comet activity or rotational light-curve effects by providing abstract base classes (`AbstractCometaryActivity` and `AbstractLightCurve`) from which custom implementations can inherit. At runtime, `Sorcha` automatically registers all classes found in the working environment that inherit from these abstract classes and makes those available for use within the code. Thus, any user-defined subclass can be used within `Sorcha`, without having to modify `Sorcha`'s source code. To make it easier for new users to develop their own models, we provide example subclasses for both comet activity and light-curve models (`LSSTCometActivity` and `SinusoidalLightCurve`) that demonstrate the bare minimum required implementation: these can be found in the `Sorcha add-ons` package.[23] To facilitate the use of these add-ons, `Sorcha` allows the ingestion of an optional complex physical parameters file, in which the user may include any necessary parameters for their models. These depend on the chosen model of variability and activity, and must reflect the necessary columns for the user-specified model.

7.3.1. Comet Activity

At Earth, most of what is observed from an active solar system object is due to mass loss. The measured flux from a comet will have a contribution from the ejected dust. The dust particles injected into a coma around the nucleus or shaped into a tail will scatter and reflect sunlight toward the observer (see J. Agarwal et al. (2024), D. Jewitt & H. H. Hsieh (2024), and references therein). In `Sorcha`, the input absolute magnitude is assumed to be the value of the nucleus alone, but we have developed the framework to allow the calculated apparent magnitude to be modified to mimic the effect of cometary activity or a collision using the cometary activity class within the `Sorcha` add-ons package. `Sorcha` assumes that the active object is still a point source with its apparent brightness modified by the activity. Given the complexities in tail/comae shapes, `Sorcha` does not adjust the PSF of the simulated small body, but the detectability of the active object will change due to the flux enhancement as a result of the activity. Adapting from the base class, a user can develop or implement their own function to take the nucleus' measured flux (which can include a contribution from a rotational light curve as described in Section 7.3.2) and add on an additional contribution for an unresolved coma/tail. This may be more representative of weak cometary activity (e.g., A. L. Cochran et al. 1989) or where the coma is bound to the object like in the case of the recent epoch of cometary activity on Centaur Chiron (M. M. Dobson et al. 2021, 2023; N. Pinilla-Alonso et al. 2024). The input parameters needed to calculate the contribution from the cometary activity should be provided in the complex physical parameters file (see Section 5.2.4).

In `Sorcha add-ons`, we have implemented the `LSSTCometActivity` cometary activity subclass. Dust production is typically parameterized by a comet's $Af\rho$ value (M. F. A'Hearn et al. 1984), and can be calculated from observed quantities

[23] https://github.com/dirac-institute/sorcha-addons

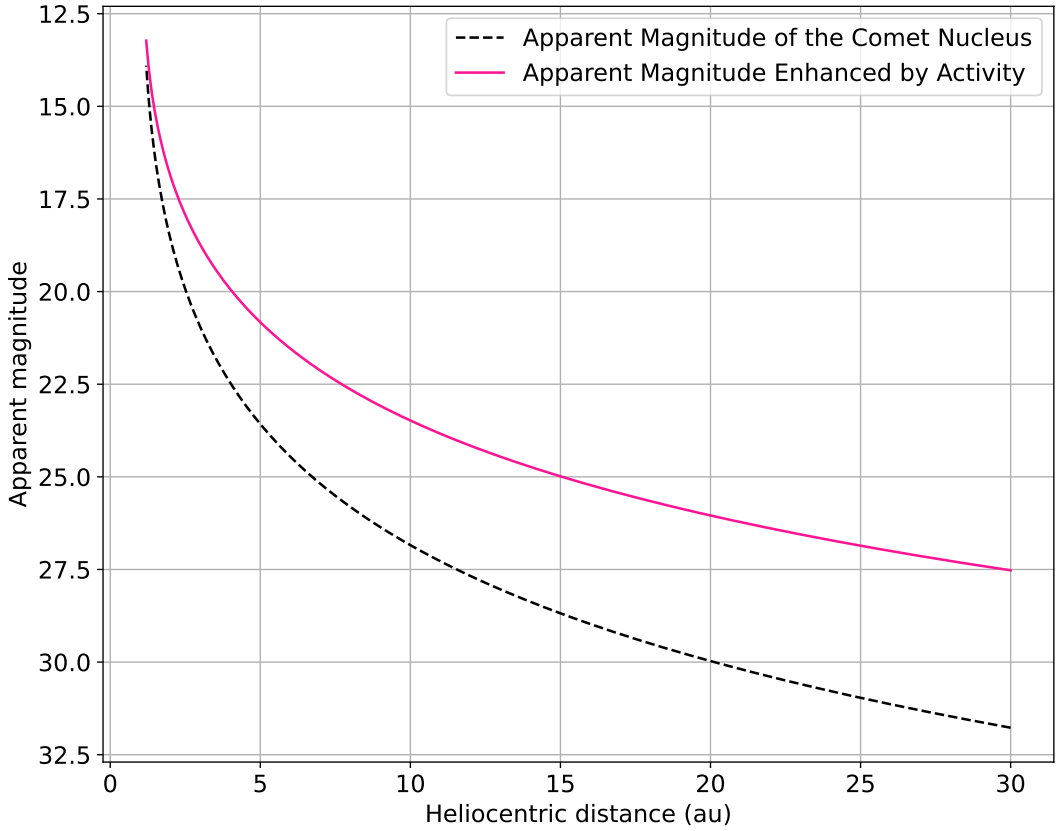

Figure 6. The `LSSTCometActivity` class's cometary activity model included in the `Sorcha add-ons` package is demonstrated here. The apparent magnitude (trailed source magnitude) in the r band as a function of heliocentric distance is plotted for an inactive comet nucleus (black) and an active comet ($Af\rho_1 = 150$ cm and $k = -0.3$.). Both simulated objects have a nucleus of $H_r = 17$. No phase function effects are included in order to highlight the impact from cometary activity ($\alpha = 0$ is assumed at all distances).

within a fixed physical photometric aperture:

$$Af\rho = \frac{4r^2\Delta^2 F_{\rm comet}}{\rho F_\odot} \quad (3)$$

where r is the heliocentric distance of the comet measured in astronomical unit, Δ is the geocentric distance of the comet in centimeters, $F_{\rm comet}$ is the measured flux of the comet in the observing band, and $F_\odot$ is the flux of the Sun at 1 au in the observing band, and ρ is the physical size of the photometric aperture in centimeters (typically selected to be 10,000 km). The activity/dust production varies as a function of heliocentric distance. The user provides the V-band $Af\rho$ value of the comet at 1 au ($Af\rho_1$) and the power-law slope that describes how the activity varies with heliocentric distance (k). The $Af\rho$ at the given observation is estimated as:

$$Af\rho(r) = Af\rho_1 r^k \mathrm{f}(\alpha) \quad (4)$$

where $Af\rho_1$ is the V-band $Af\rho$ value at 1 au, k is the activity power-law slope, r is the heliocentric distance in astronomical unit, and f(α) is the Halley–Marcus phase function at phase angle (α) from Schleicher.[24] The estimated $Af\rho(r)$ is calculated and transformed into an apparent magnitude in the observing filter (i) for the coma assuming a 1″ aperture and solar color for the dust, from C. N. A. Willmer (2018). Because of the uncertainty in what the profile of a coma should look like as a function of heliocentric distance and dust/gas production, the `LSSTCometActivity` class only applies an enhancement to the photometry. The coma/tail is treated as unresolved, meaning the active body has the same PSF as a nonactive object of the same apparent brightness. This approximation will be valid for very distant active objects and those bodies exhibiting low level activity where the contribution to the apparent magnitude from the coma will dominate the apparent magnitude (see Figure 6).

Relevant configuration file parameters: `comet_activity`

7.3.2. Light Curves

The light curves of small bodies are typically caused by rotation effects, where the variability is caused by a combination of effects such as the nonsphericity of the object, inhomogeneity in surface material, or the presence of an (unresolved) companion (e.g., J. Ďurech et al. 2010; M. Lazzarin et al. 2010; M. R. Showalter et al. 2021; C.-K. Chang et al. 2021; J. Ďurech et al. 2022; E. Ashton et al. 2023; S. Hasegawa et al. 2024). Further, with longer timescales (comparable to the expected 10 yr of operations of LSST), the change in the aspect angle of the object can also lead to a different shape of the light curve. To be able to reproduce this wide variety of variability, and its effect on object discoverability, all of the information associated with the object and the potential detections's survey observation are exposed to the user-supplied light-curve calculation, which includes any input

[24] https://asteroid.lowell.edu/comet/dustphase/

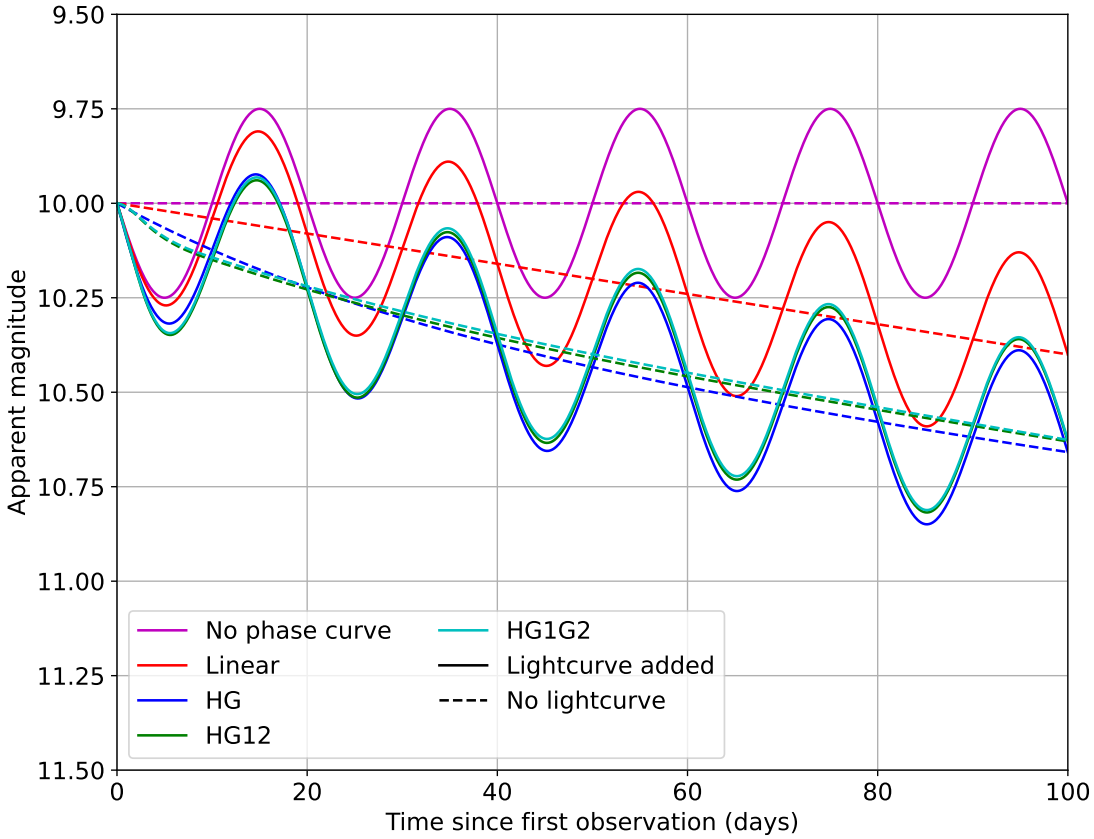

Figure 7. The effects of the simple sinusoidal light-curve model included in the `Sorcha add-ons` package for different phase function models. The dashed lines show the unmodified apparent magnitude (trailed source magnitude); the solid lines show the apparent magnitude after the model has been applied.

light-curve parameters, as well as the additional information computed in `Sorcha` (such as the full state vector of the object during the visit). Currently, the `Sorcha add-ons` package includes only a simple sinusoidal change in magnitude (the `SinusoidalLightCurve` subclass): the effect of this model is shown in Figure 7. However, the modular nature of the package and the access to the entire physical state of the object enable general calculations of the light curve in both magnitude and flux space.

Relevant configuration file parameters: `lc_model`

7.4. Calculating Trailing Losses and the PSF Magnitude

As mentioned in Section 7.1, depending on the camera pixel scale, object on-sky velocity and observation exposure time, solar system objects can produce elongated or trailed PSFs in survey images. In order to estimate the astrometric and photometric uncertainties and determine the PSF magnitude, `Sorcha` calculates for each potential detection the equivalent photometric losses caused by the elongated PSFs. `Sorcha` calculates these losses as magnitude offsets to the trailed source magnitude. There are two different trailing losses that must be calculated. The first is the trailing loss due to the smearing of the photometric signal over a larger number of pixels than for a point-source PSF, Δm(PSF) (the *PSF trailing loss*). The second is the trailing loss due to the Rubin Data Management software (M. Jurić et al. 2017) detection algorithm attempting to identify sources on the image using a stellar PSF-like matched filter. We refer to this as the *detection trailing loss*, Δm(PSF + detection), as it accounts for both the matched filter excluding part of the trail and the signal-to-noise ratio (SNR) losses due to the object's flux being distributed differently than a point source for the pixels that are included by the detection algorithm.

The PSF magnitude ($m_{\rm PSF}$) and the trailed source magnitude ($m_{\rm trailedsource}$) are related by:

$$m_{\rm PSF} = m_{\rm trailedsource} + \Delta m({\rm PSF + detection}) \quad (5)$$

The PSF trailing loss will be used later to calculate the uncertainty of the trailed source magnitude (detailed in Section 7.5) as $m_{\rm trailedsource} + \Delta m$(PSF) provides the apparent magnitude of a point source with the equivalent SNR as what would be measured for the trailed PSF. In `Sorcha`, the PSF magnitude is calculated and stored. Δm(PSF) is then calculated separately and is temporarily stored as an auxiliary value. Δm(PSF) is later used to properly calculate the overall SNR of the potential detection for an assessment of astrometric and photometric errors of the trailed source magnitude downstream (Section 7.5).

We use the prescription developed by R. L. Jones et al. (2018) to calculate both trailing losses for moving solar system objects based on the expected LSST photometric performance. The functions calculating these values in `Sorcha` were adapted from `rubin_sim` (A. J. Connolly et al. 2014; P. Yoachim et al. 2023). Both components of the trailing loss can be described by the empirical formula:

$$\Delta m = -1.25 \log_{10}\left(1 + \frac{ax^2}{1 + bx}\right) \quad (6)$$

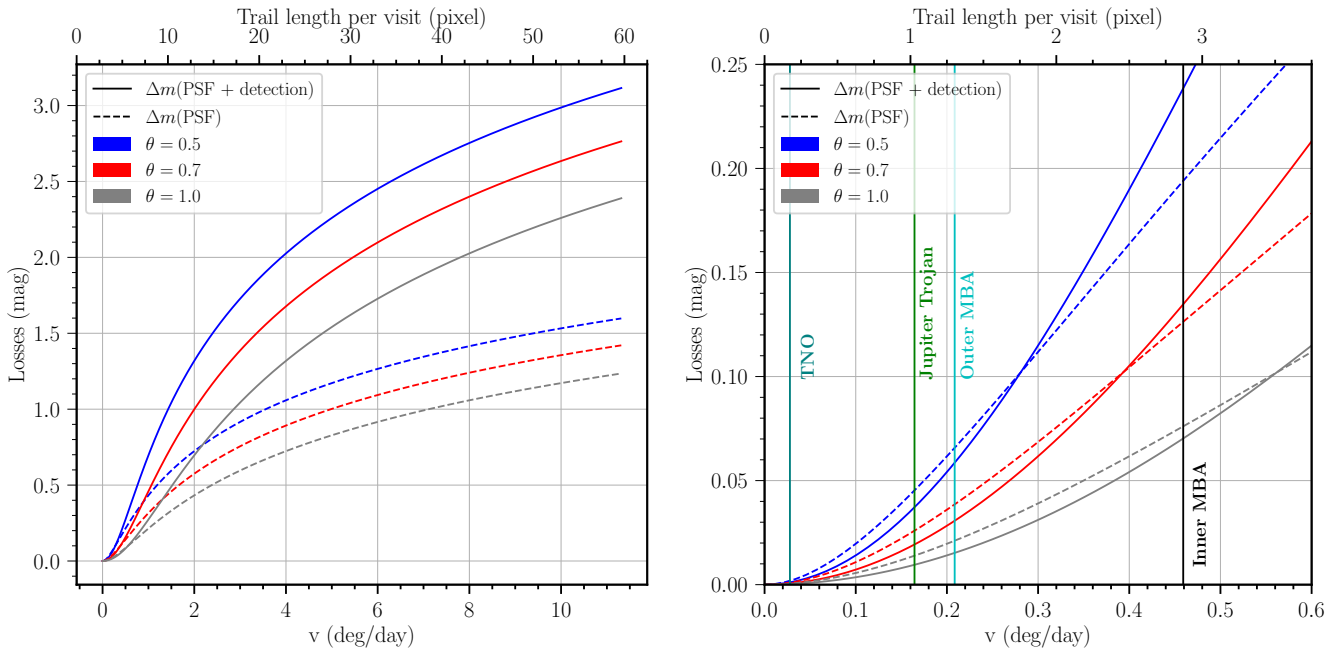

Figure 8. Left panel: the trailing losses for different values of the seeing θ, shown as a function of the object's on-sky velocity v, given in degrees per day on the bottom axis and pixels per 30 s visit on the upper axis. Right panel: a zoomed-in version of the left panel for low v. Vertical lines represent the thresholds for typical on-sky motions of a TNO, a Jupiter Trojan, and inner and outer main-belt asteroids (MBAs; (J. X. Luu & D. Jewitt 1988, Equation (1)).

where

$$x = \frac{vT_{exp}}{24\theta} \tag{7}$$

where v is the on-sky velocity (deg/day), $T_{\rm exp}$ is the exposure time (seconds), and θ is the seeing FWHM (arcseconds). a and b are different combinations of trailing loss components. We use the values fit by R. L. Jones et al. (2018) for the predicted sensitivity and performance of the Rubin Observatory and the LSST, where $a = 0.67$ and $b = 1.16$ for computing Δm(PSF), and $a = 0.42$ and $b = 0$ is used to estimate Δm(PSF + detection). This is illustrated in Figure 8. The calculated Δm(PSF) and Δm(PSF + detection) will always be greater than or equal to zero.

Relevant configuration file parameters: `trailing_losses_on`

Code function: `PPTrailingLoss`

7.5. Photometric and Astrometric Uncertainty Estimation and Randomization

In the interest of estimating the quality of orbit fits and characterization of the objects LSST will find, and to model lost and opportunistic detections near the limiting magnitude of an image, the capability to inject the uncertainty on astrometric and photometric measurements by means of random draws is provided. We compute the uncertainties for each potential detection of the input population and use them to characterize a normal distribution with a mean equal to the true value. A random draw from this distribution is then taken to generate a measurement with simulated noise. True values are retained in the full `Sorcha` output for comparison to the resulting fits (see Section 8). We note that the functions calculating these values in `Sorcha` were adapted from versions developed for `rubin_sim` (A. J. Connolly et al. 2014; P. Yoachim et al. 2023).

The models for these uncertainties are primarily driven by the SNR for a point source in an image, following the methods in Ž. Ivezić et al. (2019). Derivations of these relations are given in Appendix D. Uncertainties are computed individually for the PSF magnitude and the trailed source magnitude. The photometric error ($\sigma_{\rm mPSF}$) for the PSF magnitude ($m_{\rm PSF}$) is given in magnitudes by

$$\sigma^2_{\rm mPSF} = (0.04 - \gamma) \times 10^{0.4(m_{\rm PSF} - m_{5\sigma})} + \gamma \times 10^{2*0.4(m_{\rm PSF} - m_{5\sigma})}. \tag{8}$$

The photometric error ($\sigma_{\rm mtrailed}$) for the trailed source magnitude ($m_{\rm trailedsource}$) is given in magnitudes by

$$\sigma^2_{\rm mtrailed} = (0.04 - \gamma) \times 10^{0.4(m_{\rm trailedsource} + \Delta m({\rm PSF}) - m_{5\sigma})} + \gamma \times 10^{2*0.4(m_{\rm trailedsource} + \Delta m({\rm PSF}) - m_{5\sigma})}. \tag{9}$$

Δm(PSF) is the PSF trailing loss (as discussed in Section 7.4), and $m_{\rm trailedsource} + \Delta m$(PSF) provides the apparent magnitude of a point source with the equivalent SNR as what would be measured from a moving object's extended PSF. $m_{5\sigma}$ is the observation's 5σ limiting magnitude at the object's location on the FOV, and $\gamma = 0.039$ is a blended single-value parameter gathering together, for example, the effects of sky background and readout noise. This is in principle a filter-specific value: however, analysis by Ž. Ivezić et al. (2019) (Table 2) indicates close agreement in all bands except u, where $\gamma = 0.038$. As u-filter observations are expected only for relatively bright solar system bodies where the photometric errors are small, we find the introduced error negligible, and for simplicity, we use $\gamma = 0.039$ for all LSST bands.

The single coordinate astrometric uncertainty is then given by:

$$\sigma_{\rm astr} = k\frac{\theta}{\rm SNR} \tag{10}$$

with the SNR being well approximated in the regime of SNR $\geqslant$2 by

$$\rm SNR = \frac{1}{\sigma_{mtrailed}} \tag{11}$$

where θ is the seeing, and k is a model-dependent coefficient, which is 0.6 for a Gaussian PSF. The trailed source magnitude uncertainty is used here because the astrometic uncertainty is correlated with the trailed length. This error is then added in quadrature with 10 mas noise floor, the LSST design specification on astrometric precision (Ž. Ivezić & the LSST Science Collaboration 2013), such that the best positional accuracy achievable for the brightest image sources will never be better than the survey's reported precision. To avoid complicated conversion to uncertainty in R.A. and decl., the random perturbation applied to the astrometric measurement is computed in the local tangent plane, which is then translated into the reported values.

Since the magnitude error approximation is inversely proportional to SNR, at very low signal levels, the photometric errors increase significantly. This means that a low-SNR source with a PSF magnitude significantly fainter than the observation's 5σ limiting magnitude will have very large error bars. If there are very high photometric uncertainties, applying a random perturbation on the computed apparent magnitudes will incorrectly generate a subset of "bright sources" well above the observation limiting magnitude from objects that are too dim to be detected on the real survey images. As a compromise between low-probability detections and unrealistic magnitude uncertainties producing "fake detections," by default `Sorcha` removes all potential detections of the input population with SNR < 2 after calculating the astrometric and photometric uncertainties.

Relevant configuration file parameters: `randomization_on`

Code functions: `PPAddUncertainties`, `PPRandomizeMeasurements`

7.6. Calculating the Limiting Magnitude at the Source FOV Location

As with all telescopes, the LSSTCam FOV will be unevenly illuminated due to vignetting with the edges of the focal plane receiving less light/photonslight/fewer photons than the center. The effect of this is to decrease the 5σ limiting magnitude—the apparent magnitude where a detected point source has exactly a 50% probability of detection—at the edges of the LSSTCam FOV. `Sorcha` accommodates this by calculating the effects of vignetting at the source's location on the focal plane and adjusting the 5σ limiting magnitude accordingly for each potential detection. This modified limiting magnitude will be used when applying the survey detection efficiency (described in Section 7.8). Around 7% of LSSTCam's $2\overset{\circ}{.}06$ radius circular focal plane is expected to be vignetted by more than a 0.1 mag (P. Vereš & S. R. Chesley 2017a), but only the outermost CCDs at the corners of the cross-shaped LSSTCam detector layout are far enough from the center to be significantly impacted by vignetting, as shown in Figure 9. `Sorcha`'s built-in vignetting function is based on the functionality developed for `rubin_sim` (A. J. Connolly et al. 2014; P. Yoachim et al. 2023) based on the parameterization from C. Araujo-Hauck et al. (2016). It is tailored for the specific case of LSST and is therefore nonconfigurable.

Relevant configuration file parameters: `vignetting_on`

Code function: `PPVignetting`

7.7. Camera Footprint Filter

The footprint filter models the effects of the camera detector layout within the footprint on discoverability. This is important for the LSST, where the CCDs are arranged in a cross pattern across the circular focal plane, as shown in Figure 10. Note that 189 CCDs are mounted in 3 $\times$ 3 "rafts," where both the CCDs and the rafts are separated by narrow gaps (LSST Science Collaboration et al. 2009; Ž. Ivezić et al. 2019). `Sorcha` sifts through the possible moving object detections and picks out those that actually land within the associated observation's camera footprint and therefore could be detectable if bright enough. Solar system objects may move into one of these chip gaps (or chip gaps and raft gaps in the case of the LSST) between nightly observations, affecting their detectability; this effect is most significant in the case of slow-moving objects such as TNOs, which move only a small amount over the course of a night. However, the determination of which objects lie on the full camera footprint of the LSST can be computationally expensive, and is less concerning for faster-moving objects such as MBAs or NEOs. As a result, we provide the user with two mutually exclusive methods of applying the camera footprint, which can be selected using the configuration file: the circular footprint or the full camera footprint.

Relevant configuration file parameters: `camera_model`

Code function: `PPApplyFOVFilter`

7.7.1. Circular Footprint

This filter applies a simple circular camera footprint of a user-determined radius. For the LSST, the minimal radius to include all observations (with a certain overlap) is $2\overset{\circ}{.}06$, and the circle containing 90% of observations has a radius of $1\overset{\circ}{.}75$ (R. L. Jones et al. 2018). These radii are illustrated in Figure 10. If using the circle footprint filter, `Sorcha` can also remove a user-defined fraction of random input objects that land within the circle footprint. This is intended to quickly mimic the effect of the gaps/areas without CCDs without performing the computationally expensive procedure of explicitly detecting and removing specific detections, which do not lie upon a detector. The "fill factor" (the fraction of observations to retain) is set in the configuration file. A circular radius of $1\overset{\circ}{.}7$ and a fill factor of 90% mimics the LSSTCam detector area. Note that as described in Section 6, the in-built ephemeris generator already utilizes a circular "search field" analogous to this circular footprint with the radius set by the user in the configuration file. Thus, setting the radius of the circular footprint filter to be equal to or larger than that set by the user for the ephemeris generation search radius (plus buffer) will have little effect.

Relevant configuration file parameters: `circle_radius`, `fill_factor`

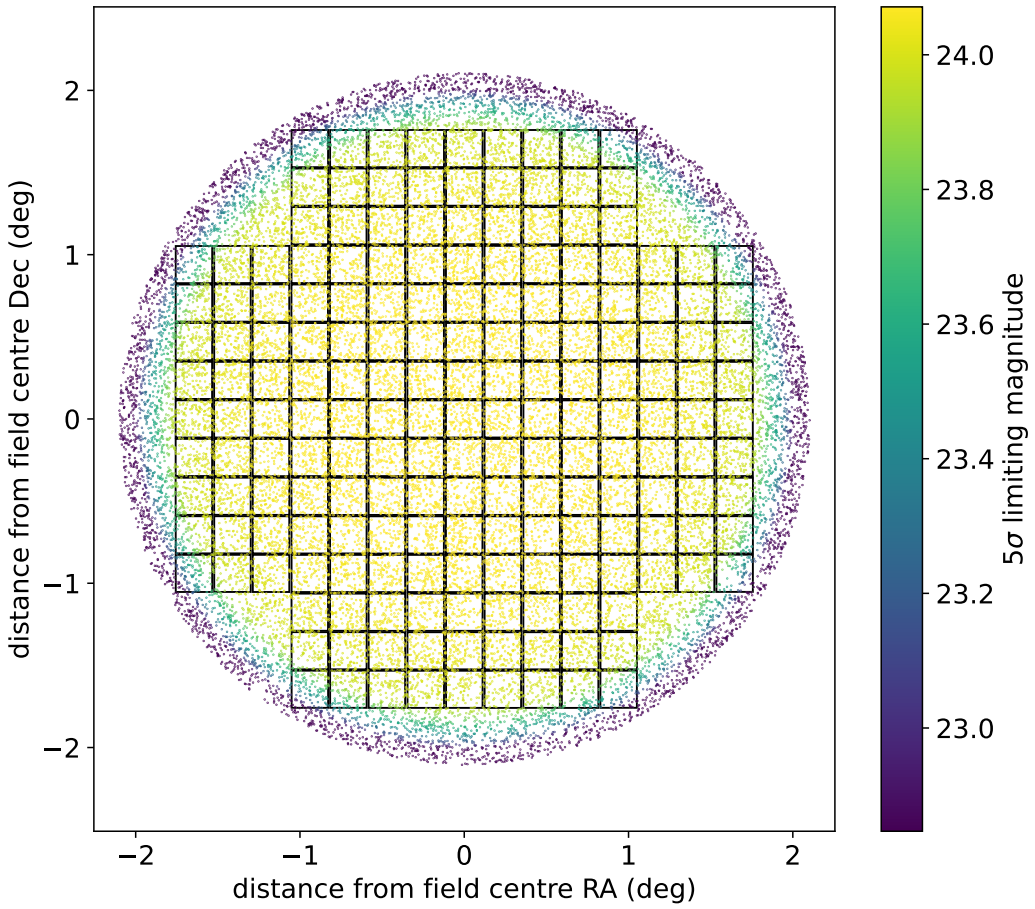

Figure 9. The effects of vignetting on the 5σ limiting magnitude for a randomized series of points on a circular FOV in the LSSTCam focal plane. Locations farther from the center of the FOV have shallower depths. The LSSTCam detector footprint is also plotted. The center of this example pointing has a limiting magnitude of 24.071.

Code function: `PPCircleFootprint`, `PPSimpleSensorArea`

7.7.2. Full Camera Footprint

The camera footprint filter is the most accurate and computationally most expensive version of the footprint filter. The function was adapted from a version utilized in `rubin_sim` (A. J. Connolly et al. 2014; P. Yoachim et al. 2023). It checks every object against the full detector map of the instrument, taking into account the field rotation from the pointing database, and removes all objects that do not lie upon a detector. The effect of this is illustrated in Figure 11. By default, the code will use an in-built detector map of LSSTCam. The user may also supply a detector map of their own devising, as described in Section 5.3.2. Additionally, it is expected that sources that lie on the very edges of the detectors will not be correctly extracted. The user can, via the configuration file, parameterize the distance from the edge of a detector (in arcseconds on the focal plane) at which an object will not be detected. In `Sorcha`'s example configuration files, this value is set to 2″ (10 pixels) following recommendations made by the Rubin Data Management Team.

Relevant configuration file parameters: `footprint_path`, `footprint_edge_threshold`

Code function: `footprint.applyFootprint`

7.8. Source Detection Efficiency Filter

The detection efficiency filter serves to pick out which of the input small bodies that land within the FOV footprint of a given survey's observation would be detected by the survey's source extraction algorithm (for the LSST, this is the Rubin Difference Imaging Analysis; M. Jurić et al. 2021). For each observation/pointing in the survey, we model the source detection efficiency as a fading function, the shape of which is twofold: the detection efficiency remains constant when substantially above the 5σ limiting magnitude of the observation, then drops swiftly when approaching the 5σ limiting magnitude (shown in Figure 12). Following P. Vereš & S. R. Chesley (2017a), `Sorcha` utilizes the following formulation of the fading function (B. Gladman et al. 1998; M. Zavodny et al. 2008; B. J. Gladman et al. 2009) to represent the per image source detection efficiency:

$$\epsilon\left(m_{\mathrm{PSF}}\right) = \frac{F}{1 + e^{\frac{m_{\mathrm{PSF}} - m_{5\sigma}}{w}}} \tag{12}$$

where $\epsilon(m_{\mathrm{PSF}})$ is the probability of detection, F is the peak detection efficiency, m_{PSF} and $m_{5\sigma}$ are, respectively, the object's PSF magnitude and 5σ limiting magnitude of the observation at the source's location on the camera focal plane (see Section 7.6), and w is the width of the fading function. In this formulation, by definition $\epsilon(m_{5\sigma}) = F/2$. The shape of the function and the variation of parameters F and w are visualized

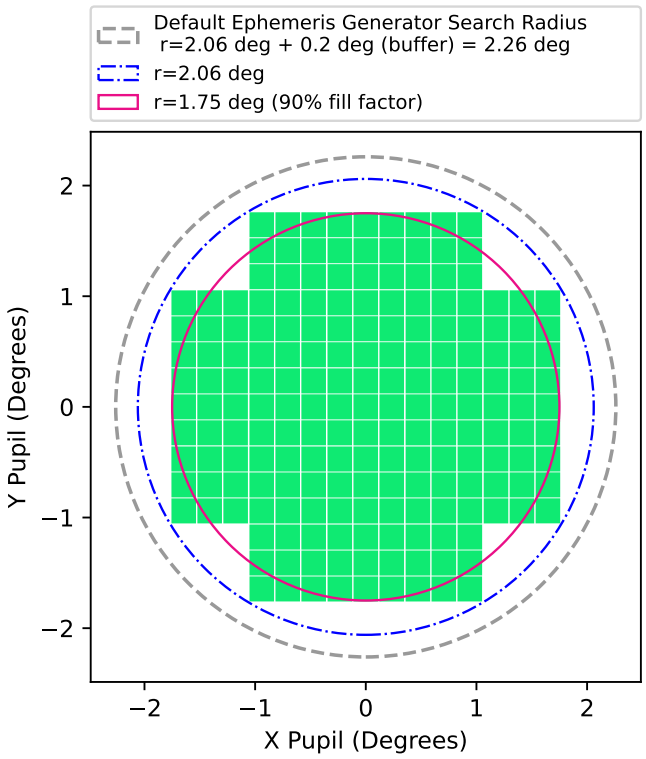

Figure 10. The LSSTCam field of view (FOV). The overplotted circles show the recommended default search area radius (including buffer) used in `Sorcha`'s ephemeris generation stage (gray dashed line); the $2.^{\circ}06$ radius circle encompasses all of the CCD detectors (blue dashed–dotted line); and the $1.^{\circ}75$ radius circles contains 90% of LSSTCam detector area (pink solid line).

in Figure 12. In our implementation, the detection efficiency $\epsilon(m_{\rm PSF})$ is calculated at the PSF magnitude for each potential detection of an input synthetic small body, and compared to a random number selected for each detection opportunity from a uniform distribution. The PSF magnitude is used here because the Rubin DIA pipeline uses PSF filter matching as described in Section 7.1. Those potential detections whose drawn random number is less than or equal to $\epsilon(m_{\rm PSF})$ will be deemed "detected" as an astronomical source on the relevant survey observation/pointing and will continue to be passed on to later stages of postprocessing.

The values of w and F can be controlled via the user through the configuration file and are fixed for all pointings of the simulated survey. Both parameters are survey-specific, and the actual values for the LSST can only be estimated when on-sky data becomes available. For w, we adopt 0.1 as a baseline following the Sloan Digital Sky Survey (J. Annis et al. 2014). For F, we adopt a default of 1: source detection for the LSST has been well-tuned for bright sources, and the true value will likely be very close to 1 (M. Jurić et al. 2021).

We acknowledge that the currently implemented method for accounting for the survey source detection efficiency within `Sorcha` uses median values measured/estimated for the entire survey being simulated. This strategy works well given the values provided within the pointing database generated by the `rubin_sim` LSST cadence simulations and the information expected to be available early on at the start of the LSST. Other representations, such as including per observation per CCD detector source detection efficiencies, may be a better fit once data is flowing from the LSST. We are planning for future releases of `Sorcha` to include additional options for representing realistic LSST source detection efficiency estimates.

Relevant configuration file parameters: fading_function_-width, fading_function_peak_efficiency

Code functions: `PPFadingFunctionFilter`

7.9. Saturation Limit Filter

While saturated sources can often still be detected in astronomical images, their brightness cannot be reliably measured because either the pixels have such a high number of electrons that the response significantly deviates from linear or the pixel well has overflowed with electrons spilling into neighboring pixels. Moving solar system objects that saturate in LSSTCam exposures will not be identified. This is because the DIA algorithms employed by the Rubin Observatory Data Management Team for daily prompt processing and annual data releases will mask saturated pixels (LSST Science Collaboration et al. 2009; Ž. Ivezić et al. 2019). Ž. Ivezić et al. (2019) estimated that the saturation limit for LSST will be the 16th magnitude in the r-filter. `Sorcha` implements a simple magnitude cutoff. This cutoff threshold can be provided by the user in the configuration file as either a single value that will apply to all observational filters, or as a comma-separated list of filter-specific values.

Relevant configuration file parameters: `bright_limit`

Code function: `PPBrightLimit`

7.10. Linking Filter

It is not enough for a moving object to be found by the source detection pipelines within a wide-field survey, it must also be identified as a moving solar system object. In practice, automated algorithms sift through a survey's source catalogs "linking" transient detections together in order to identify potential tracks from objects on heliocentric orbits. This step oftentimes involves orbit fitting from the linked astrometric measurements. The linking filter within `Sorcha` mimics the selection function from automated discovery algorithms used to discover the solar system moving objects in wide-field surveys to pick out which members of the synthetic input population are discovered by the simulated survey and when the discovery happens. For `Sorcha`'s v1.0 release, we primarily focused on replicating the Rubin SSP requirements (see requirement `OSS-REQ-0159` in https://ls.st/oss, and also M. Jurić et al. 2020), which builds tracklets (linkages of observations within a given night) and attempts to link three separate nights of tracklets over 14 days (see Figure 13) onto heliocentric orbits. SSP relies on an upgraded version of the HelioLinC algorithm (M. J. Holman et al. 2018), but the linking filter within `Sorcha` simply counts the number of detections per night to make tracklets, retaining detections from objects that satisfy a certain number of tracklet nights within a specific window while also applying a discovery efficiency. For SSP, we set the discovery efficiency in our default configuration files to 95%, the minimum efficiency required specified in the LSST Observatory System Specifications (`OSS-REQ-0159` in https://ls.st/oss). The software implementation of this algorithm—named `miniDifi`—is highly vectorized for performance reasons and described in detail in the code comments.

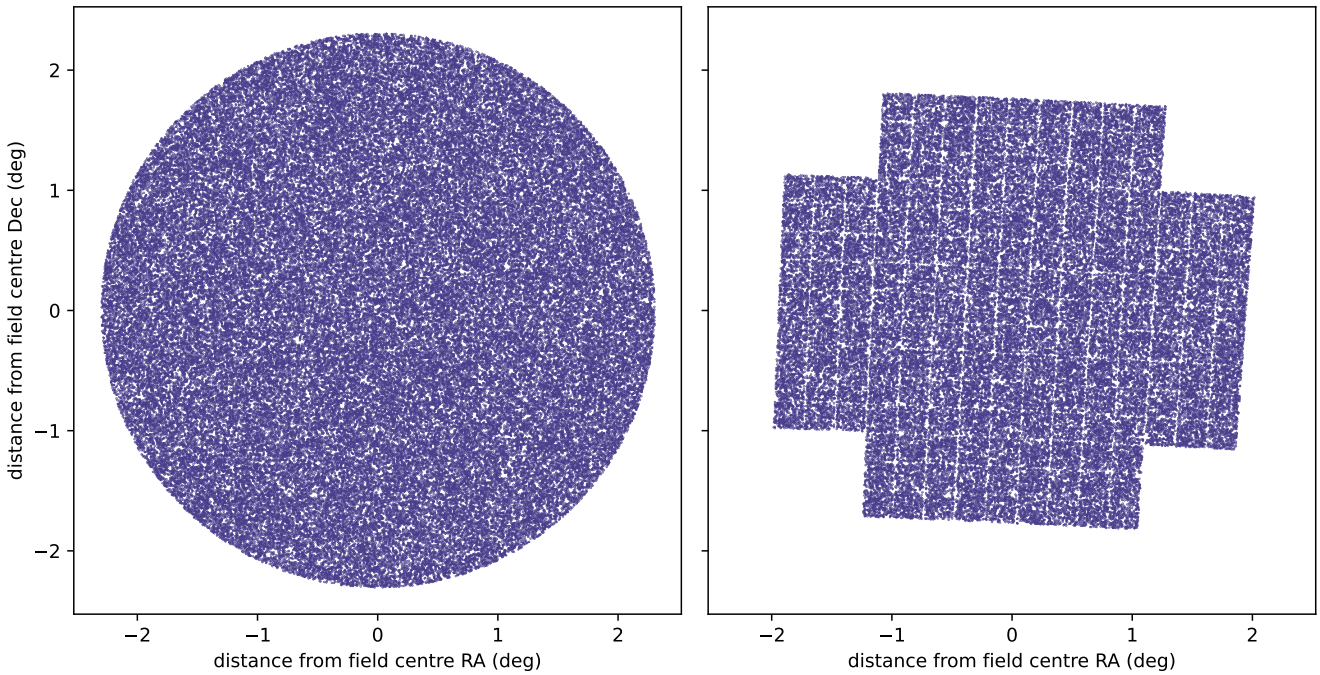

Figure 11. The effect of the full camera footprint filter on a selection of 100,000 random synthetic sources. Left panel: original sources, distributed over a circular FOV of radius 2.°1. Right panel: the same sources after running `Sorcha`'s full camera footprint filter. The shape of the LSSTCam detector footprint can be seen with the loss of detections in the raft and chip gaps.

We assume perfect precovery and recovery from within the survey observation such that if the input object has sufficient observations to be linked at some point in the survey, then the linking filter keeps all of these observations associated with that object. Detections from objects not linked are removed by default and not passed to the next stages of postprocessing. For the same `Sorcha` run, a user may want to compare the parts of the input population that would have been correctly linked and identified as a moving solar system object to the parts that were not linked but would be present in the survey source catalogs. We have made the linking filter configurable to either remove all observations from unlinked input objects or keep all detections even if the object is not linked with a linking flag column to identify which input objects were linked.

Although SSP will discover the majority of the solar system objects found during the LSST, SSP requires that there be motion between each observation that comprises a tracklet: this means that slow objects moving at distances beyond about 200 au will be undetectable by the SSP unless it is implemented in a way that exceeds Rubin requirements. Other bespoke algorithms targeting very slow/distant objects may have to be developed by the community, and applied later on during the lifetime of the LSST. We have made the linking filter flexible in that the minimum on-sky distance between nightly pairs within a tracklet, the number of observations required to make a tracklet, the number of tracklets that must be linked, and the linking window are all configurable by the user.

Relevant configuration file parameters: `SSP_detection_efficiency`, `SSP_number_observations`, `SSP_separation_threshold`, `SSP_maximum_time`, `SSP_number_tracklets`, `SSP_track_window`, `SSP_night_start_utc`

Code function: `PPLinkingFilter`

7.11. Additional Advanced Filters

We have also developed some additional filters to assist with specific science use cases and scenarios that could speed up `Sorcha`'s performance. These filters are intended for the advanced user and are turned off by default within `Sorcha`. These filters can be activated by simply including the relevant keyword in the configuration file.

7.11.1. SNR/Magnitude Limit Filters

We have also included the option for the user to create a magnitude-limited or SNR-limited subsample of the synthetic survey discoveries through the magnitude limit and SNR filter functions. The user may either implement the SNR limit, to remove all observations of objects below a user-defined SNR threshold, or the magnitude limit, to remove all observations of objects above a user-defined trailed source magnitude. This may be useful for estimating/predicting follow-up sample sizes, checking whether a known object is predicted to have been observed by the survey with a certain SNR or trailed source apparent magnitude, or analyzing population statistics related to follow-up observing programs. One such use case we envision with `Sorcha` is for developing/testing the results from an observing program similar to the Colours of the Outer Solar System Origins Survey (Col-OSSOS; M. E. Schwamb et al. 2019; W. C. Fraser et al. 2023). Col-OSSOS observed 102 Centaurs and TNOs discovered by OSSOS; M. T. Bannister et al. 2016, 2018), with discovery triplet brightness $r < 23.6$ mag. This brightness limited sample enabled cosmogonical composition studies of the Kuiper Belt (W. C. Fraser et al. 2021; L. E. Buchanan et al. 2022; R. E. Pike et al. 2023) by combining the optical–near-infrared colors from Col-OSSOS and the well-characterized OSSOS detection efficiency and pointing history through the OSSOS survey simulator

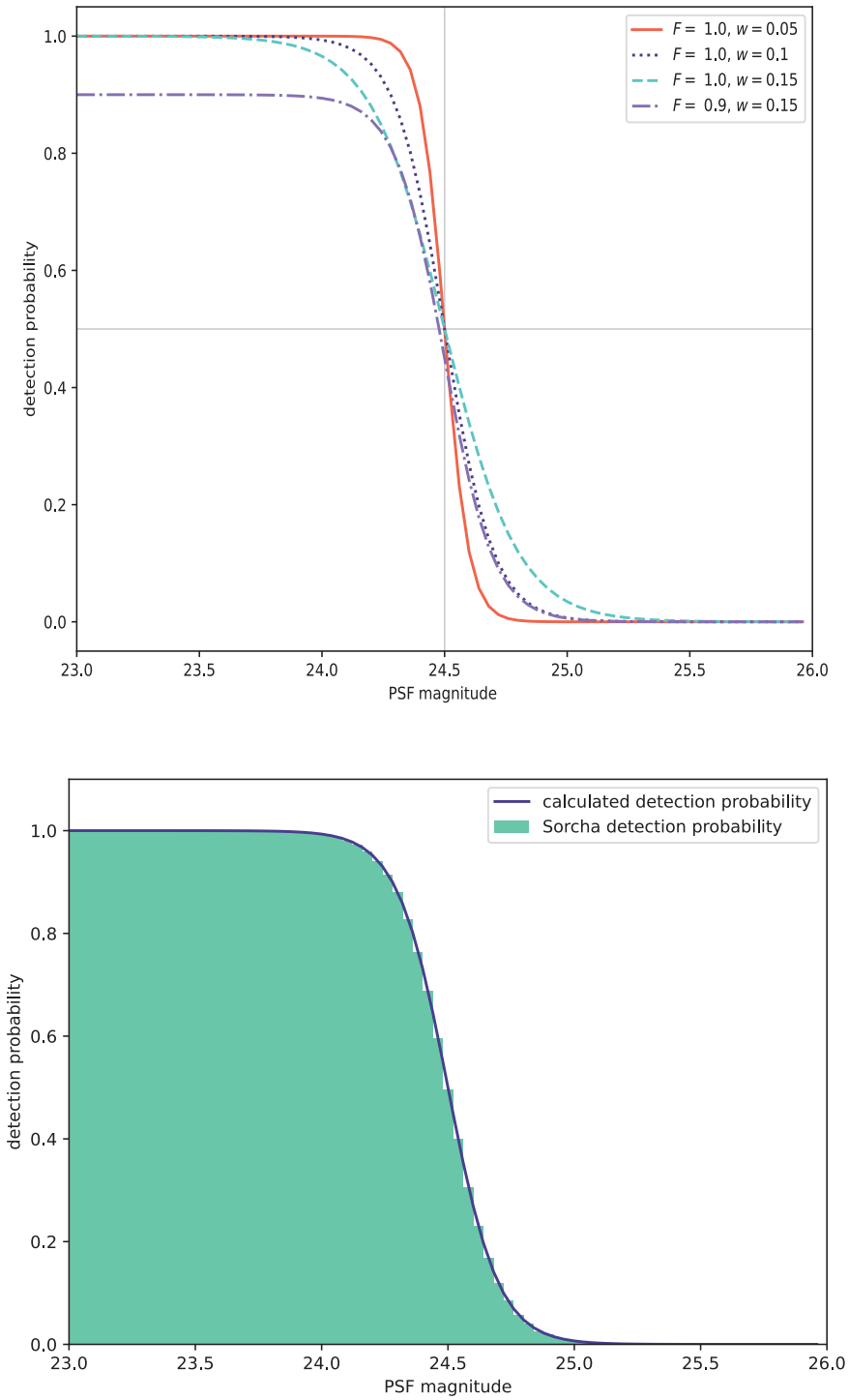

Figure 12. Top panel: the fading function as defined in Equation (12), representing the fraction of observed point sources as a function of PSF magnitude. The different lines represent the effect of the variation of the peak detection efficiency F and the width parameter w on the shape of the function. The 5σ limiting magnitude at the source location is marked in gray ($m_{5\sigma} = 24.5$). Bottom panel: a histogram showing detection probability of 10,000 point sources passed through Sorcha's fading function filter, with the actual calculated detection probability from Equation (12) overplotted as a solid line. Here, $F = 1.0$, $w = 0.1$, and $m_{5\sigma} = 24.5$, and the binsize is 0.04 mag.

(S. M. Lawler et al. 2018a) and other simulations. We note that using these filters is not the appropriate way of handling the source detection efficiency of the simulated survey (see Section 7.8). This filter is applied before the source detection efficiency and linking effects are modeled within Sorcha. Therefore, the user must take care when combining the SNR/magnitude limit filters with Sorcha's source detection efficiency and/or the linking filter to ensure that the output from the Sorcha simulation satisfies the user's science case as the generated Sorcha detections will not represent all of the discoveries that the LSST should have found for the input population.

Relevant configuration file parameters: `SNR_limit`, `magnitude_limit`

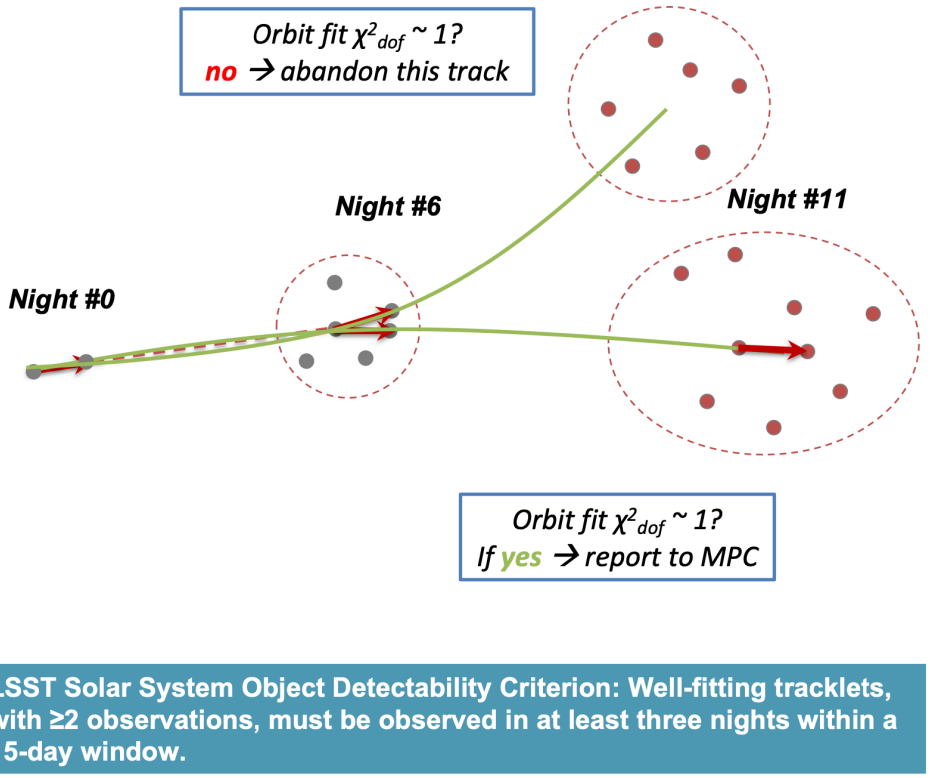

Figure 13. A cartoon schematic of tracklet creation and the linking process for the Rubin Solar System Processing Pipeline (SSP).

Code functions: `PPSNRLimit, PPMagnitudeLimit`

7.11.2. Faint Object Culling Filter

By default, `Sorcha` passes all input orbits to its built-in ephemeris generator as described in Section 6. Applying the ephemeris generator's integrations and calculations to the entire input population can be costly given that small objects on moderately to highly eccentric objects will be preferentially discovered only near perihelion, but spend a large fraction of their time near aphelion undetectable by the simulated survey with apparent magnitudes well beyond the survey's brightness limit. In most cases with the power-law absolute magnitude/size distributions, the majority of the input population that `Sorcha` will process will not end up detected by the simulated survey. We provide the expert user the option to apply the faint object culling filter, which estimates the brightest trailed source magnitude any input object will potentially reach and removes those input objects that will never be bright enough to be detected by the simulated survey before moving on to the ephemeris generation stage.

As a fast first pass, this filter takes the orbits of the input model and calculates a maximum trailed source magnitude in each filter for every object to be simulated as described in Section 7.2 (for phase angle α of 0) at the object's perihelion. The geocentric distances are approximated as $(q - 1)$ in order to calculate an estimate for the brightest possible trailed source magnitude. The routines to calculate the perihelion if needed from the input parameters are described in M. J. Holman et al. (2025). Modifications to this maximum brightness due to any activity or light-curve models are incorporated at this stage according to the user's implementation of the `maxBrightness` method within each user-supplied activity and light-curve class (see Section 7.3). Finally, for each object, a check is made if all of these calculated maximum trailed source magnitudes are fainter than 2 + the deepest survey observation per filter (as obtained from the survey pointing database)—if the object is fainter in all filters than this criteria, it is dropped and not sent on to the ephemeris generation stage. This approximation is only applied to those objects with a calculated perihelia >2 au.

This optional filter within `Sorcha` substantially decreases the compute time for a simulation. For the model of $\sim 6.9 \times 10^6$ Centaurs from J. Murtagh et al. (2025), split across 128 cores on Queen's University Belfast's HPC Kelvin2 with a chunk size of 5000, a full simulation of all objects takes ~29 hr to run (or ~3700 core-hours)—for the exact same model and simulation setup but with the filter applied, this is reduced to only ~1.25 hr (or ~225 core-hours).

Relevant configuration file parameters: `brute_force`

Code functions: `PPFaintObjectCullingFilter, PPEstimatePerihelion`

8. Outputs

The main output from `Sorcha` is a file listing all predicted observations of objects from the input population. Additionally, `Sorcha` can be configured to output the generated ASSIST+REBOUND ephemeris and a statistics file summarizing the main results. `Sorcha` also generates two log files (one with a ".log" extension for general information, and one with a ".err" extension for error messages) each time `Sorcha` is run. These log files should contain helpful information for the user wishing to check if a `Sorcha` run has completed successfully and understand why a run has failed or produced unexpected results. We provide further details about the

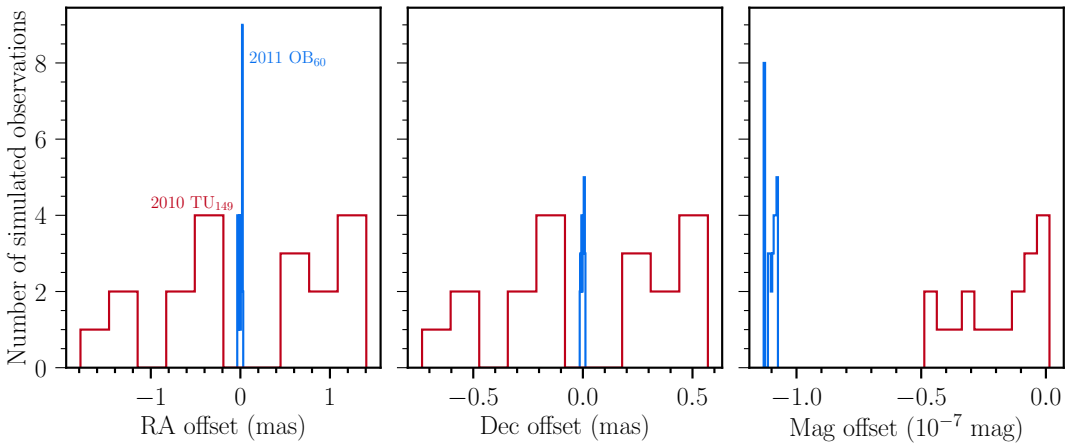

Figure 14. Histogram of differences in R.A. (left panel), decl. (center panel), and trailed source magnitudes (right panel) between `Sorcha`-derived values and an independent verification using JPL Horizons. The red curves show 2010 TU_{149}, an MBA, while the blue curves show 2011 OB_{60}, a TNO.

contents and formats of `Sorcha`'s outputs below, but we recommend the reader review any updates to the output format by reading the additional documentation.[25]

8.1. Detections File

`Sorcha`'s main output file is the detections file, which contains all of the predicted survey observations of input objects that are found by the survey. By default, only the information associated with "detected and linked" objects from the input population are included in detections file. If the science cases require it, a configuration file variable can be set to allow successful detections of unlinked objects to be included in the output. The resulting file will contain an additional column, a flag identifying which detections are associated with input objects that have passed both the source detection and linking filters in postprocessing (see Sections 7.8 and 7.10, respectively).

The user also has complete control over how much information is provided in each row through the configuration file. By default, "basic" output includes column headers describing the pointing information for the observation and the object's predicted position and calculated brightness in the observational filter. Alternatively, the user may ask for "all" columns, which includes all columns from the input files and the full ephemeris information; this is useful in cases where the user wants all of the information accessible in a single file and does not need to worry about the larger file size. Finally, the advanced user can also specify a comma-separated list of column headings for their output file, choosing from the available headings to create a customized output. The potential column headings for the detections file are given in Table 6. The main output file can also be written out in multiple formats: comma-separated and whitespace-separated text files, which are useful for readability when output file sizes are smaller; and HDF5 and SQLite3 databases, which perform better when output sizes become especially large and are generally recommended for HPC runs. The advanced user may also specify if they wish for the values to be rounded to a select number of decimals; by default, this option is off.

Relevant configuration file parameters: `output_format`, `output_columns`, `position_decimals`, `magnitude_decimals`

8.2. Output Ephemeris File

The user can output the ephemeris calculated by the ASSIST+REBOUND ephemeris generator (described in Section 6), using a flag on the command line. The most common use case for this ephemeris output is to create a file that can be read back in by `Sorcha` in subsequent runs, thus skipping the computationally expensive process of ephemeris generation in cases where the user wishes to re-run `Sorcha` with the same orbital information but with changes to the simulation configuration or the physical parameters of the input population. The contents of this file are shown in Table 3. This ephemeris file can be output in several formats: comma-separated, whitespace-separated, or HDF5. Of the three, HDF5 format allows for the fastest read-in.

Relevant configuration file parameters: `eph_format`

8.3. Statistics (Tally) File

`Sorcha` has the option to output a statistics or "tally" file, again using a flag on the command line. This file is designed to be a quick, easy-to-access overview of the output of a `Sorcha` run providing information per detected object per filter. The format and columns of the statistics file are listed in Table 7. This file is useful in cases where the user wishes to know very simple information, such as which objects passed the linking filter and when, or the extent of the phase curves for the objects, without having to deal with a potentially large and unwieldy main output file.

9. Validation

To verify whether the results from `Sorcha` are meaningful and properly reproduce what is expected from a survey, we have independently computed every step of the simulation for two objects (2011 OB_{60}, a TNO, and 2010 TU_{149}, an MBA), starting with the ephemerides as seen from the Rubin Observatory over the span of a month, derived using JPL Horizons,[26] and compared to the `Sorcha` outputs for one of

[25] https://sorcha.readthedocs.io/

[26] https://ssd.jpl.nasa.gov/horizons/app.html#/

Table 6
Output Detection File Column Headings

Heading	Description
Default output columns	
ObjID	Unique object identifier of the simulated object (string)
fieldMJD_TAI[a]	Observation Mean Julian Date (MJD) in TAI ((International Atomic Time)
fieldRA_deg	R.A. of the center of the observation pointing (degrees)
fieldDec_deg	Decl. of the center of the observation pointing (degrees)
RA_deg	Object R.A. (degrees)
Dec_deg	Object decl. (degrees)
astrometricSigma_deg	Astrometric uncertainty in object R.A. and decl. position (degrees)
optFilter	Filter (band) for this observation (*ugrizy*)
trailedSourceMag	Trailed Source Magnitude
trailedSourceMagSigma	1σ uncertainty on trailed source magnitude
fiveSigmaDepth_mag	5σ limiting magnitude at the object's location on the camera focal plane
phase_deg	Sun-object-observer angle (phase angle) (degrees)
Range_LTC_km	Light-time-corrected object-observer distance (km)
RangeRate_LTC_km_s	Light-time-corrected rate of change of the object-observer distance (km s^{-1})
Obj_Sun_LTC_km	Object-Sun light-time-corrected distance (km)
object_linked[b]	Flag to identify where the synthetic object was linked by the linking filter
All other output columns	
FieldID	Observation pointing field identifier
RA_true_deg	Object R.A. unadjusted for astrometric uncertainty (degrees)
Dec_true_deg	Object decl. unadjusted for astrometric uncertainty (degrees)
RARateCosDec_deg_day	Object R.A. rate of motion (deg day^{-1})(degrees)
DecRate_deg_day	Object R.A. rate of motion (deg day^{-1})(degrees)
visitTime	Total length of time for a visit (seconds)
visitExposureTime	Total exposure time for a visit (seconds)
fieldRotSkyPos_deg	Angle of the field *y*-axis and celestial north
seeingFwhmGeom_arcsec	Geometric FWHM for the field (arcseconds)
seeingFwhmEff_arcsec	Effective FWHM for the field (arcseconds)
fieldFiveSigmaDepth_mag	5σ limiting magnitude at the center of the field of view (FOV)
H_filter	Absolute magnitude in the survey exposure's observing filter
PSFMag	Calculated PSF magnitude
PSFMagSigma	1σ uncertainty on PSF magnitude
trailedSourceMagTrue	Calculated trailed apparent magnitude unadjusted for photometric uncertainty
PSFMagTrue	Calculated PSF magnitude unadjusted for photometric uncertainty
SNR	Predicted signal-to-noise ratio (SNR) of detection
detectorID	Identifier of the camera detector covering the observation
date_linked_MJD[c]	Date (MJD in TAI) on which the object was discovered by linking
Columns from orbits input	See Table 1
Columns from physical parameters input	See Table 2
Columns from ephemeris generation	See Table 3

Notes. All positions and velocities are in respect to J2000.
[a] Reported as midpoint of the simulated survey observation.
[b] This column only appears if Sorcha has been instructed not to drop unlinked objects in the configuration file if the linking filter is on.
[c] This column only appear if the linking filter is on.

the reference LSST baselines. Due to limitations in the Horizons system, the ephemerides can only be generated at a one point per minute cadence and, since the LSST is approximately twice as fast as that (one image every 30 s, plus a few seconds of overheads), we interpolated these positions to the exposure time midpoint. We verified (again using an independent implementation) whether the object was inside the simplified circular footprint for every one of the simulated ≈19,000 exposures in this period, and computed the trailed source apparent magnitude given similarly interpolated distances (topocentric and heliocentric), as well as including color terms. The Sorcha-derived and independently computed list of simulated observations agreed with each other (that is, the two sets of pointings that would have seen the object were identical), with the mean difference μ and standard deviation σ between the Sorcha-derived R.A.s being $\mu = 8 \times 10^{-4}$ mas, $\sigma = 0.02$ mas for 2011 OB_{60}, and $\mu = 0.10$ mas, $\sigma = 1.02$ mas for 2010 TU_{149}. For decl.s, we have $\mu = 9 \times 10^{-5}$ mas, $\sigma = 0.007$ mas for 2011 OB_{60} and $\mu = 0.02$ mas, $\sigma = 0.42$ mas for 2010 TU_{149}. Both of these values show the high quality of the predictions expected from ASSIST (M. J. Holman et al. 2023), and they are below the LSST astrometric requirement of 10 mas precision (Ž. Ivezić et al. 2019). The Sorcha-derived and independently computed trailed source magnitudes had a systematic offset of -10^{-7} mag for 2011 OB_{60} and -1.9×10^{-8} mag for 2010

Table 7
Output Statistics File Column Headings

Heading	Description
ObjID	Unique object identifier for each input object (string)
optFilter	Relevant observing filter (band) (*ugrizy*)
number_obs	Number of observations for this object in the given optFilter
min_apparent_mag	Minimum calculated apparent magnitude for this object in the given optFilter
max_apparent_mag	Maximum calculated apparent magnitude for this object in the given optFilter
median_apparent_mag	Median calculated apparent magnitude for this object in given optFilter
min_phase	Minimum calculated phase angle for this object in the given optFilter (degrees)
max_phase	Maximum calculated phase angle for this object for the given optFilter (degrees)
object_linked[a]	True/False whether the object passed the linking filter
date_linked_MJD[b]	Date the object was linked (if it was linked) in Mean Julian Date (MJD) in TAI

Notes.
[a] This column only appears if Sorcha has been instructed not to drop unlinked objects in the configuration file and the linking filter is on.
[b] This column only appears if the linking filter is on.

TU_{149}, several orders of magnitude below the LSST design goals of 10 mmag. We illustrate these results in Figure 14. Such a stringent test, then, demonstrates that Sorcha produces correct results.

10. Benchmarking and Performance

A benchmarking script is included in the Sorcha repository (see footnote 17), which runs Sorcha on a test-set of 1000 MBAs for 1 yr of the survey, using default Sorcha settings with all filters turned on. On a 2020 M1 MacBook Pro, this benchmark script takes around 115 s to run. Users may run this benchmarking script themselves if they wish to test their own machine or access more granular details of the benchmark, as described in the additional documentation (see footnote 25). Sorcha has been tested extensively on HPC facilities. On the University of Rijeka's Bura supercomputer, 1 million model Centaurs with a chunk size of 5000, split across 100 cores, takes roughly 2 hr to run, or 200 core-hours. We note that the benchmark script may take longer to run on HPC facilities: HPC nodes typically have slower clocks than consumer-grade CPUs but many times the number of cores. These estimates are for Sorcha simulations that send all of the input population to ephemeris generator. Depending on the use case, it may be possible to further speed up Sorcha simulations by applying the faint object culling filter (as described in Section 7.11.2) to estimate the maximum brightness of the input objects and remove those that will never get bright enough to be observed before the ephemeris generation stage.

11. Utility Scripts

Sorcha has some useful utilities developed to aid the user, all of which are accessible via the CLI. For example, when Sorcha is installed, a demo can be run to check that the installation has proceeded correctly. The files used for the demo can be copied into a directory of the user's choice by running:

```
sorcha demo prepare
```

The demo command to run Sorcha using these files can be obtained through:

```
sorcha demo howto
```

The example configuration files given in Appendix C can be copied into a directory of the user's choice with:

```
sorcha init
```

The list of papers and software to cite for Sorcha can be found using:

```
sorcha cite
```

A utility also exists to collate the results of several Sorcha runs—for example, those on HPC facilities—into a single SQLite database, including the input files as separate tables on the database. This can be accessed via:

```
sorcha outputs create-sqlite
```

Finally, another utility for HPC users serves to check all of the log files from many Sorcha runs and determine whether any of them failed, producing an output file that will list failed runs and give the reason for failure.

```
sorcha outputs check-logs
```

More detailed help and explanation is available for all of the command line utilities via the standard -help flag.

12. Important Limitations and Caveats

We built Sorcha to handle a wide variety of use cases, but there are some scenarios that the software is not designed to handle. In addition, there are also some assumptions we have made. We list below a few noteworthy things that the user should be aware of and consider before using Sorcha for science.

If using the built-in ephemeris generator (Section 6), the orbits of the massive dwarf planet and asteroid perturbers within the main asteroid belt (listed in Table 5) cannot not be directly simulated, as they are already included as masses within the ASSIST+REBOUND integrations. Adding in a synthetic massless particle on the same orbit, would find that

orbit is significantly modified by the interactions with the perturber. Additionally, nongravitational forces such as cometary outgassing, Yarkovsky–O'Keefe–Radzievskii–Paddack (YORP: V. V. Radzievskii 1952; S. J. Paddack 1969; J. A. O'Keefe 1976) effect (see W. F. Bottke et al. 2006 and D. Vokrouhlický et al. 2015 for a detailed review), Yarkovsky forces (W. F. Bottke et al. 2006), or the impulse from a rocket engine (in the case of a moving artificial spacecraft) are not accounted for in the ASSIST +REBOUND *N*-body integrations used by `Sorcha`. At the time of writing, ASSIST is also unable to handle modeling small body collisions, breakup events, or accounting for planet impacts. If required, a user can provide separate ephemerides accounting for any of these situations as input into `Sorcha` rather than using the built-in ephemeris generator and its ASSIST+REBOUND integrations.

By default, `Sorcha` assumes that a synthetic planetoid's phase curve parameters are the same for each apparition. Over time as the object moves in its orbit, the viewing angle changes and more or less of a planetoid's surface may become visible or new regions with differing albedo may now be illuminated, changing the amount of sunlight reflected to the Earth and the observed phase curve. Distant objects like TNOs do not move very far in their orbits, so this effect is mostly negligible, but for some MBAs, Jupiter Trojans, and NEOs, this effect can be significant (e.g., T. Kwiatkowski & A. Kryszczynska 1992; J. M. Carvano & J. A. G. Davalos 2015; M. Mahlke et al. 2021; M. Schemel & M. E. Brown 2021; S. L. Jackson et al. 2022; J. E. Robinson et al. 2024). Bulk analysis of multiyear MBA phase curves from the Asteroid Terrestrial-impact Last Alert System survey explored the impact of this effect on the MBAs. M. Mahlke et al. (2021) measured a median value of 0.07 mag for the amplitude of the magnitude dispersion due to aspect angle changes between different apparitions. Future enhancements to `Sorcha` might include adding the recently proposed sHG_1G_2 phase curve model (B. Carry et al. 2024), which accounts for the variable viewing geometry.

Currently, we apply the same source detection and linking efficiencies across the sky regardless of the background stellar density. This may not necessarily be true in regions within or very near to the galactic plane, where the stellar crowding is significant. The LSST is using difference imaging to identify transient sources and moving solar system objects, which can help mitigate some of the impact from high stellar crowding in the survey observations (Ž. Ivezić et al. 2019; M. Jurić et al. 2021). Our assumption that the efficiency is the same as in regions with no crowding is likely the best choice at this stage before LSSTCam commissioning and the start of Rubin Operations. It would be straightforward to modify `Sorcha` to include a variable detection efficiency as a function of galactic latitude if there is a significant difference in the galactic plane fields discovery rates compared to the rest of the LSST footprint.

We also remind the reader that for the LSST, `Sorcha` computes two apparent magnitudes: the trailed source magnitude and the PSF magnitude. As introduced in Section 7.1, the PSF magnitude is used to assess whether a solar system object will be detected by the Rubin source detection algorithms. The trailed source magnitude is the resulting apparent magnitude from summing up all of the photons hitting the detector, taking into account that the source may be trailed/streaked. For most science cases, the user will need the trailed source magnitudes, and these values are included in the default output for `Sorcha`. If not using the default output option, we urge the user to take great care and ensure that they are using the correct apparent magnitude column in `Sorcha` output files for their analysis.

13. Summary and Conclusions

In this work, we have introduced the methodology and software design behind `Sorcha`, an open-source `Python` survey simulator that will take any defined input small body population and bias it to what a survey would have detected utilizing the survey's pointing history, observation metadata, camera footprint, search algorithm detection efficiency, and other information about the telescope and observatory. `Sorcha` is `pip` and `conda` installable. The source code is hosted in an open repository (see footnote 17). Tutorials and additional documentation (see footnote 25) are available as well. We welcome the community to add enhancements to the simulator package through pull requests on the repository. Additional light-curve or cometary activity classes developed by the community can be shared with a pull request to the `sorcha add-ons` repository (see footnote 23).

The significant challenge in developing `Sorcha` has been to optimize for the scale of the LSST discoveries (∼5 million+ detections/∼1 billion individual photometric and astrometric detections at the end of 10 yr). We have prioritized making `Sorcha` fast and easy to use, whether the user intends to run on hundreds to thousands of cores on an HPC system or executing several processes on a laptop or desktop computer. Our aim is for `Sorcha` to be a community resource that can accurately handle orbits from across the solar system, including hyperbolic and near-Earth orbits. One such example is how `Sorcha` will be used in the nightly Rubin SSP pipelines as part of the software to associate previously known solar system objects on secure orbits with real-time LSST transient detections within 60 s of the LSST Camera (LSSTCam) reading out an exposure.

Currently, `Sorcha` is configured to use the outputs from the Rubin Observatory LSST survey strategy simulations. Once the Rubin Observatory is operational, we will add in the capability to ingest the LSST's observation metadata. Although `Sorcha` has been developed with LSST in mind, much of the code base is applicable to other wide-field and pencil-beam surveys. With some effort, it would be possible for `Sorcha` to be modified to simulate what these surveys should have detected for an input model small body population. Future versions of the software may incorporate recent well-characterized wide-field discovery surveys such as OSSOS (M. T. Bannister et al. 2016, 2018), and the Dark Energy Survey (DES; P. H. Bernardinelli et al. 2020, 2022) or the Formation of the Outer Solar System: An Icy Legacy (FOSSIL; C.-K. Chang et al. 2021; E. Ashton et al. 2023) into `Sorcha` to enable joint constraints with the LSST Solar System discoveries.

Acknowledgments

This work was supported by an LSST Discovery Alliance LINCC Frameworks Incubator grant [2023-SFF-LFI-01-Schwamb]. Support was provided by Schmidt Sciences. S. R.M. and M.E.S. acknowledge support in part from UK Science and Technology Facilities Council (STFC) grants ST/V000691/1 and ST/X001253/1. G.F. acknowledges

support in part from STFC grant ST/P000304/1. This project has received funding from the European Union's Horizon 2020 research and innovation program under the Marie Skłodowska-Curie grant agreement No. 101032479. M.J. and P.H.B. acknowledge the support from the University of Washington College of Arts and Sciences, Department of Astronomy, and the DiRAC (Data-intensive Research in Astrophysics and Cosmology) Institute. The DiRAC Institute is supported through generous gifts from the Charles and Lisa Simonyi Fund for Arts and Sciences and the Washington Research Foundation. M.J. wishes to acknowledge the support of the Washington Research Foundation Data Science Term Chair fund, and the University of Washington Provost's Initiative in Data-Intensive Discovery. J.M. acknowledges support from the Department for the Economy (DfE) Northern Ireland postgraduate studentship scheme and travel support from the STFC for UK participation in LSST through grant ST/S006206/1. J.A.K. and J.M. thank the LSST-DA Data Science Fellowship Program, which is funded by LSST-DA, the Brinson Foundation, and the Moore Foundation; their participation in the program has benefited this work. S.E. and S.C. acknowledge support from the National Science Foundation through the following awards: Collaborative Research: SWIFT-SAT: Minimizing Science Impact on LSST and Observatories Worldwide through Accurate Predictions of Satellite Position and Optical Brightness NSF award No. 2332736 and Collaborative Research: Rubin Rocks: Enabling near-Earth asteroid science with LSST NSF award No. 2307570. R.R.L. was supported by the UK STFC grant ST/V506990/1. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation.

This work was also supported via the Preparing for Astrophysics with LSST Program, funded by the Heising Simons Foundation through grant 2021–2975, and administered by Las Cumbres Observatory. This work was supported in part by the LSST Discovery Alliance Enabling Science grants program, the B612 Foundation, the University of Washington's DiRAC Institute, the Planetary Society, Karman +, Breakthrough Listen, and Adler Planetarium through generous support of the LSST Solar System Readiness Sprints. Breakthrough Listen is managed by the Breakthrough Initiatives, sponsored by the Breakthrough Prize Foundation (http://www.breakthroughinitiatives.org).

This research has made use of NASA's Astrophysics Data System Bibliographic Services. This research has made use of data and/or services provided by the International Astronomical Union's Minor Planet Center. The SPICE Resource files used in this work are described in C. H. Acton (1996) and C. Acton et al. (2018). Simulations in this paper made use of the REBOUND *N*-body code (H. Rein & S. F. Liu 2012). The simulations were integrated using IAS15, a 15th-order Gauss-Radau integrator (H. Rein & D. S. Spiegel 2015). Some of the results in this paper have been derived using the `healpy` and HEALPix packages. This work made use of Astropy:[27] a community-developed core `Python` package and an ecosystem of tools and resources for astronomy (Astropy Collaboration et al. 2013, 2018; Astropy Collaboration et al. 2022). We thank the Vera C. Rubin Observatory Data Management Team and Scheduler Team for making their software open-source. We thank Dave Young and Conor MacBride for initial help setting up the `Python` project and repository. The authors also thank Michele Bannister and Rosemary Dorsey for conversations that helped improve the software's handling of interstellar objects. We also thank Aidan Berres, Ricardo Bánffy, Richard Cannon, Konstantin Malanchev, Brian Rogers, and Richard Cannon, for their contributions to documentation, discussions about feature implementations, and/or beta testing. We thank Max Mahlke for useful feedback on the manuscript. We also thank Jean-Marc Petit for his detailed and thoughtful review of this manuscript. We are additionally grateful to the members of the Rubin Observatory LSST Solar System Science Collaboration for useful feedback at the LSST Solar System Readiness Sprints. We also thank the contributors to Stack Overflow for their examples and advice on common `Python` challenges that provided guidance on solving some of the programming challenges we have encountered.

This material or work is supported in part by the National Science Foundation through Cooperative Agreement AST-1258333 and Cooperative Support Agreement AST1836783 managed by the Association of Universities for Research in Astronomy (AURA), and the Department of Energy under contract No. DE-AC02-76SF00515 with the SLAC National Accelerator Laboratory managed by Stanford University.

We are grateful for the use of the computing resources from the Northern Ireland High Performance Computing (NI-HPC) service funded by EPSRC (EP/T022175). We gratefully acknowledge the support of the Center for Advanced Computing and Modelling, University of Rijeka (Croatia), for providing supercomputing resources at HPC (High Performance Computing) Bura.

The authors wish to acknowledge the researchers who worked tirelessly to rapidly develop COVID-19 vaccines and subsequent boosters. Without all of their efforts, we would not have been able to pursue this work.

We acknowledge the contribution of pets Isha Bernardinelli; Freddie and Millie Merritt, Stella Schwamb; Richard, Calcifer, and Buttons West; that, by keeping us awake at night, yowling during our meetings, or providing general emotional support, led to improvements in this software and manuscript.

Data Access: The software presented here is available open-source at https://github.com/dirac-institute/sorcha.

Facility: Rubin:Simonyi

Software: sorcha, ASSIST (M. J. Holman et al. 2023; H. Rein et al. 2023), Astropy (Astropy Collaboration et al. 2013, 2018; Astropy Collaboration et al. 2022), Healpy (K. M. Górski et al. 2005; A. Zonca et al. 2019), Matplotlib (J. D. Hunter 2007), Numba (S. K. Lam et al. 2015), Numpy (C. R. Harris et al. 2020), pandas (W. McKinney 2010; The pandas development team 2020), Pooch (L. Uieda et al. 2020), PyTables (PyTables Developers Team 2002), REBOUND (H. Rein & S. F. Liu 2012; H. Rein & D. S. Spiegel 2015), rubin_sim (A. J. Connolly et al. 2014; P. Yoachim et al. 2023), rubin_scheduler (E. Naghib et al. 2019; P. Yoachim et al. 2024), sbpy (M. Mommert et al. 2019), SciPy (P. Virtanen et al. 2020), Spiceypy (A. Annex et al. 2020), sqlite (https://www.sqlite.org/index.html), sqlite3 (https://docs.python.org/3/library/sqlite3.html), tqdm (C. da Costa-Luis et al. 2023), Black (https://black.readthedocs.io/en/stable/faq.html), Jupyter Notebooks (T. Kluyver et al. 2016).

[27] http://www.astropy.org

Author Contribution

S.R.M. served as the lead developer of Sorcha from 2021 June onward, contributing a substantial portion of the current code base and the majority of unit tests. S.R.M. also led new feature implementations, oversaw ongoing code maintenance and repository management, resolved software issues, conducted extensive testing on both HPC clusters and local systems, and contributed to the documentation. For this paper, S.R.M. wrote the abstract and Sections 5, 8, and 10; S.R.M. wrote Sections 1 and 2 with reference to an earlier draft by G. F.; S.R.M. wrote Appendix B and the preambles to Section 3 and 7; and S.R.M. co-wrote Sections 7.4–7.9. S.R.M. created Figure 12 and generated Figures 8, 9, and 11 based on Jupyter notebooks created by S.C. and G.F. Additionally, S.R.M. made Tables 6 and 7 and created Tables 1–4 based on tables made by J.A.K. Also, S.R.M. is responsible for the overall layout and organization of the paper and for edits to the draft.

G.F. was the initial developer of Sorcha and contributed significantly to early drafts of the manuscript. Substrates of his text can be found in the article (in particular in Sections 1, 2, 5, and 7), and in the code. He also produced Figure 3 and contributed to the design of Figure 12. G.F. participated in the discussions and decisions about the overall design and implementation of Sorcha. He also provided feedback on the overall paper draft.

M.E.S. served as principal investigator and project manager of the Sorcha team. She also served as PI of the Sorcha LINCC Frameworks Incubator Proposal. She contributed to the discussions and decisions about the overall design and implementation of Sorcha and the documentation: this included leading the team in-person meetings and calls. She helped beta test the software, contributed to the documentation, formatted docstrings, and also reviewed pull requests submitted to the GitHub repository. She also led the development of the built-in citation function. She wrote Sections 4, 6, 7.3.1, 7.10, 12, and 13, and Appendix A.2. She also contributed to Sections 7.2, 7.4,7.7, and 7.10. She made Figures 1, 2, 4, 6, and 10. She also provided significant feedback on the overall paper draft, assisted with incorporating co-author feedback into the manuscript, and made significant contributions to the Sorcha documentation.

S.C. contributed to Sorcha software development by implementing several of the fundamental models of the telescope footprint and postprocessing functions. S.C. also assisted in validation of intermediate calculations and final data sets, and created visualizations for the intermediate and final data sets. S.C. also contributed to this publication by authoring Section 7.5.

P.H.B. contributed to the development of several of the Sorcha routines for ephemerides generation and coordinate conversion described in Section 6, as well as the design of the optional and user-defined cometary activity and light-curve models of Section 7.3. He implemented the independent validation test described in Section 9. He also helped with the design of Figure 8. In addition to these sections, he also wrote Appendix D, and provided feedback on the overall paper draft.

M.J. contributed the initial architecture design, the algorithm for the repeatable parallel random-number generation, the vectorized algorithm and implementation for the miniDifi linking emulation package, and the design and implementation of UX (user experience) elements such as the fast command line activation and options. He also produced Figure 13. M.J. also served as the liaison to the Rubin project and coordinated its application in Rubin Data Preview 0.3 (which was created using an early version of Sorcha).

M.J.H. led the development and refinement of Sorcha's ephemeris generator algorithm with ASSIST+REBOUND. He also helped verify the output from the ephemeris generator components and improved the algorithm to better handle fast-moving solar system objects. He also provided feedback on the paper manuscript.

J.A.K. provided testing, user feedback, and bug-fixing throughout Sorcha's development. He also contributed to the Sorcha documentation. He also provided feedback on the paper manuscript.

S.E. contributed to the conceptualization of Sorcha as well as providing feedback on the LINCC Frameworks Incubator proposal. He was involved in the discussions about the overall design and functionality of Sorcha. Moreover, S.E. contributed to the choice and implementation of the astrometric and photometric uncertainty models presented in Section 7.2 as well as the editing of this article. He also provided feedback on the paper manuscript.

D.O. applied the LF-PPT to the Sorcha code base, introduced several performance improvements, and implemented the plugin system allowing externally defined activity and light-curve models to be used by Sorcha. D.O. also contributed text to Appendix A.

M.W. contributed to the Sorcha code base, including efficiency improvements in array handling and integrating the first draft of the ASSIST+REBOUND ephemeris generation.

J.K. contributed to the Sorcha code base, including improvements to input file reading and random-number generation. J.K. also contributed text to Appendices A and A.3.

J.M. contributed to the overall code base, in particular providing testing of the linking filter, cleaning up logging functionality, developing example Jupyter notebooks for estimating input colors, contributing to various sections of the documentation, and writing details the operation of the faint object culling filter. He also provided feedback on the overall manuscript and figures, in particular Figures 1 and 2.

R.L.J. developed rubin_sim and rubin_scheduler. Some of the functions within Sorcha were adapted from rubin_sim. R.L.J. also contributed to useful discussion and provided feedback and guidance on how to use the simulated rubin_sim survey cadence pointing databases and how to apply trailing losses and uncertainty calculations. She also produced a Jupyter notebook that was used as an example template to develop Figure 8. She also provided feedback on the paper manuscript.

P.Y. developed rubin_sim and rubin_scheduler. Some of functions within Sorcha were adapted from rubin_sim. P.Y. also provided feedback and guidance on rubin_sim cadence simulation pointing databases.

R.R.L. implemented the data class configuration file reader. He also implemented the upgrades to enable the user to optionally specify the ephemeris generation auxiliary files and other additional variables in the configuration file. He also contributed updates to improve the overall user experience with the Sorcha CLI and other changes to improve the maintainability of the Sorcha code base.

M.S.P.K. developed the initial version of the LSSTCometActivity class that was modified for inclusion in Sorcha add-ons and that led to the development of the abstract comet class implementation.

J.M. provided feedback and contributed to discussions during coding sprints at the University of Washington. He developed the simulated linking algorithm (difi) that was used in early iterations of Sorcha that led to the development of miniDifi.

K.K. contributed to beta testing Sorcha for the edge case of simulated Earth impactors and very fast-moving potentially hazardous asteroids (PHAs).

S.P.N. developed the objectsInField (OIF) ephemeris generator/simulator that enabled the development of Sorcha. The output from OIF was the input into early iterations of Sorcha as the source of ephemeris calculations.

C.S. provided feedback and advice on incorporating cometary activity in to Sorcha's apparent magnitude calculations.

S.M.M. made contributions to early drafts of Sorcha documentation and setting up early versions of Sorcha to be pip installable.

C.O.C. contributed in the discussions about the development and enhancement of Sorcha during the LINCC Frameworks Incubator.

Appendix A
Repository Setup, Software Testing, and Package Architecture and Deployment

Sorcha uses software engineering best practices, processes, and tools to help ensure the accuracy, usability, and long-term maintainability of the code. Many of these were configured through the use of the LINCC (LSST Interdisciplinary Collaboration for Computing)-Frameworks Python Project Template[28] (LF-PPT; D. Oldag et al. 2024), which provides a suite of tools and conventions that are readily applied to new or existing projects. The LF-PPT automatically generates a familiar directory structure to organize code and provide the foundation for continuous integration (CI) with GitHub Actions[29] to exercise the code via unit tests and ensure adherence to user-defined coding style guides. These tools and practices ensure the accuracy of, and prevent regressions in, the software. To aid with long-term maintenance of the software, the LF-PPT provides a nightly CI smoke test that executes Sorcha's comprehensive suite of tests. This smoke test confirms that the package continues to build and run even as lower-level dependencies might change. This low-effort approach makes the code easier to maintain over many years. Additionally, to make documentation easy to produce, readily accessible, and maintainable, the LF-PPT provides a set of basic Sphinx documentation generator (see footnote 19) configuration files needed to automatically publish documentation to ReadTheDocs.[30] By default, the LF-PPT documentation configuration automatically includes user generated pages, API (Application Programming Interface) docstrings, and rendered Jupyter notebooks.

Sorcha uses multiple well-adopted package repositories to provide a mechanism for users to install and update the package. Sorcha is both conda/mamba and pip installable. The LF-PPT provides a workflow for automatically publishing a new release to PyPI[31] when a release is tagged in the Sorcha Github repository (see footnote 17). We also created additional recipes to automatically publish to conda-forge.[32] Minimal effort is required to deploy new versions of the code due to the automation. PyPI and conda-forge were selected as the release channels due to their popularity in the scientific community.

A.1. Command Line Interface Implementation

Importing the entire Sorcha Python package can take 5 s or more, and making the user wait that long just to print out an error message would be a poor user experience. To provide the user with a fast experience on the command line, Sorcha's command line interface (CLI) and parsing is separated from the core code. The large imports of Python packages, including the Sorcha package, are performed only in the execute function that runs the main Sorcha simulation. Importing Sorcha from the CLI execute function and not at the top-level of the module allows us to exit quickly and print the help information or print an error message (in case there was a mistake on the command line).

A.2. Random-number Generation

Sorcha implements a per-module randomization approach that provides the ability to force deterministic behavior during testing regardless of the order in which modules are executed, as described in M. E. Schwamb et al. (2024). The PerModuleRNG class creates random-number generators for each model using a global seed and a hash based on the module's name. By default, Sorcha uses the 4 bytes returned from os.urandom(), which returns a string of random bytes from the operating system's randomness source, as the global seed to ensure random behavior for scientific runs. This ensures that the random seed is initialized uniquely when launching nearly simultaneous parallel Sorcha runs across many nodes in an HPC cluster. For reproducibility (mainly for testing, debugging purposes), a user can set a hidden environmental configuration parameter to override this setting and force deterministic behavior.

A.3. Reader Classes

Sorcha reads data files via a series of reader classes that provide wrappers around common libraries for reading in data from such formats as CSV (comma-separated values), white-space-separated text, HDF5, and SQLite databases. By using a common wrapper, derived from an abstract ObjectDataReader class, the individual reader classes present the user with a common interface for accessing data files with functions such as read_rows(). Individual reader classes inherit from the ObjectDataReader class and are specialized for both the input format and category of data. For example, the OrbitAuxReader reads in data from comma or whitespace-separated files and contains logic for processing and standardizing different orbital specifications (e.g., Keplerian versus Cartesian parameters). The reader classes also incorporate additional error checking, such as required columns, and use Sorcha's logging framework.

A.4. Ephemeris Generation Auxiliary Files

Sorcha's ephemeris generator requires various SPICE (Spacecraft, Planet, Instrument, C-matrix, Events) kernels and

[28] https://github.com/lincc-frameworks/python-project-template
[29] https://docs.github.com/en/actions
[30] https://readthedocs.org/
[31] https://pypi.org/project/sorcha/
[32] https://github.com/conda-forge/sorcha-feedstock/

resource files[33] (C. H. Acton 1996; C. Acton et al. 2018) from the Jet Propulsion Laboratory's Navigation and Ancillary Information Facility (NAIF) and the Minor Planet Center (MPC's) observatory file (obscode_extended.json.gz[34]) to obtain the precise starting locations for the Moon, planets, and 16 asteroid perturbers, as well as the observatory's location on the Earth to initialize the *N*-body simulations and then calculate the R.A., decl., heliocentric state vectors, and light-time-corrected values. These auxiliary files present a challenge due to their size and frequency of updates. Large files are impractical to include with packaged code, and files that change frequently require additional effort to keep the distribution up to date. One alternative requires the end user to manually gather all of the required files and then provide the paths at runtime. `Sorcha` addresses all of these issues by using the data retrieval tool `Pooch` (L. Uieda et al. 2020), which can be run as part of a stand-alone bootstrap utility when `Sorcha` is first installed or automatically run within a `Sorcha` simulation if the auxiliary files are not found (see Section 11). `Sorcha` defines a registry file with the URL (Uniform Resource Locator) locations of all required auxiliary files. At runtime, `Pooch` uses the registry to download and cache the files in the default temporary directory of the operating system or on a directory defined by the user. During subsequent runs, the files are retrieved from the local cache and only downloaded as needed. The user is always free to move or delete the files from the cache to trigger `Pooch` to download the latest copy of the default auxiliary files during the next `Sorcha` run. If there is a need to run `Sorcha` with a new or older version of the NAIF SPICE kernels or the MPC observatory file, the `Sorcha` configuration file can be used to override the default file names and URL addresses for any of the auxiliary files `Sorcha` uses.

Appendix B
Quick-start Guide

This brief quick-start guide shows the user how to set up and run a set of demonstration input files through `Sorcha` using the command line. For detailed installation instructions, we refer the user to our documentation.[35]

`Sorcha` can be installed via `pip` or `conda` /`mamba` from the `conda-forge` channel:

```
conda install -c conda-forge sorcha
```

or:

```
mamba install -c conda-forge sorcha
```

or:

```
pip install sorcha
```

Once `Sorcha` is installed, the user must also download the auxiliary files used for ephemeris generation. This needs to be done only once. To install these files into the local system cache (there is an option to specify a directory for these files if running on an HPC setup), use:

```
sorcha bootstrap
```

`Sorcha` comes packaged with a set of demo files for 10 random solar system objects. This includes a small 1 yr version of the pointing database, a configuration file similar to those presented in Appendix C, an orbits file, and a physical parameters file.

```
ObjID a e inc node argPeri ma epochMJD_TDB FORMAT
2010_TU149 2.2103867 0.8245336 1.9679 58.91911 92.60419 324.21145 60200 KEP
2010_TC209 2.6411033 0.2378397 17.51668 25.01678 18.48118 347.09673 60200 KEP
2011_OB60 103.711231 0.6461554 19.42877 142.66859 226.19366 358.36581 60200 KEP
2011_WJ157 99.2799891 0.6226317 27.06784 76.41243 58.43528 355.56379 60200 KEP
2011_YA42 3.1910723 0.1754333 17.90056 291.08053 99.36952 70.55556 60200 KEP
2012_AW12 1.6899997 0.6063531 51.51998 151.77178 246.62641 150.38051 60200 KEP
2012_BB14 1.0627063 0.098754 2.64121 316.78101 255.55235 135.8483 60200 KEP
2012_HW1 1.0607726 0.6615131 49.57978 207.81039 163.70136 63.97462 60200 KEP
2013_CC 263 3.0000262 0.1126188 8.97766 336.5147 98.64025 69.02504 60200 KEP
2013_GD138 43.6879964 0.1086886 5.43381 57.00856 146.16056 24.30927 60200 KEP
```

The contents of the input physical parameters file for the same 10 synthetic objects with orbits described in the orbits file:

```
ObjID H_r GS u-r g-r i-r z-r y-r
2010_TU149 20.7 0.15 2.13 0.65 -0.19  0.14 -0.14
2010_TC209 18.45 0.15 1.9 0.58 -0.21 -0.3 -0.39
2011_OB60 6.9 0.15 1.72 0.48 -0.11 -0.12 -0.12
2011_WJ157 4.92 0.15 2.55 0.92 -0.38 -0.59 -0.70
2011_YA42 16.36 0.15 2.13 0.65 -0.19  0.14 -0.14
2012_AW12 19.79 0.15 1.9 0.58 -0.21 -0.3 -0.39
2012_BB14 24.99 0.15 1.72 0.48 -0.11 -0.12 -0.12
2012_HW1 19.99 0.15 2.13 0.65 -0.19 0.14 -0.14
2013_CC 263 18.46 0.15 1.72 0.48 -0.11 -0.12 -0.12
2013_GD138 8.14 0.15 2.55 0.92 -0.38 -0.59 -0.70
```

[33] https://naif.jpl.nasa.gov/naif/data_generic.html
[34] https://minorplanetcenter.net/data
[35] https://sorcha.readthedocs.io/en/latest/installation.html

For more details on the contents of all of the demonstration input files, we refer the user to Section 5. A single `Sorcha` command will copy these demo files into the local directory and print the command needed to run the `Sorcha` demo to the terminal:

```
sorcha demo prepare
```

We do not supply the main command to run `Sorcha` here as this demo command may change in future versions of the code. If the user requires the current working demo command to be reprinted to the terminal, they can run:

```
sorcha demo howto
```

The output from the `sorcha run` demo command can be simply copy-pasted into the terminal and run. Once `Sorcha` has finished running, it will have produced two log files and two output files: the main `Sorcha` output file, `testrun_e2e.csv`, and a statistics file, `testrun_stats.csv`. The contents of these files are described in Section 8. The top few lines of the main `Sorcha` output file will be a CSV file, the first few lines of which will appear similar to the following:

```
ObjID,fieldMJD_TAI,fieldRA_deg,fieldDec_deg,RA_deg,Dec_deg,astrometricSigma_deg,optFilter,...
2011_OB60,60225.247167832895,2.8340797698367206,-12.19406486,1.9825549664962523,
  -11.89549965,7.4867112566597e-06,i,...
2011_OB60,60225.27094733885,2.8340797698367206,-12.19406486,1.9821061017355532,
  -11.89565259,1.970228013144157e-05,z,...
2011_OB60,60225.28270233429,2.8340797698367206,-12.19406486,1.981888824400869,
  -11.89577997,1.368449919373884e-05,z,...
2011_OB60,60227.24471872862,1.830365601304555,-10.65341974,1.9444094822390525,
  -11.91153435,1.2466492192790201e-05,r,...
2011_OB60,60227.26843110208,1.830365601304555,-10.65341974,1.9439523895215416,
  -11.91172213,1.0506087754733871e-05,i,...
```

Note that this output has been truncated in both directions to fit this paper and that the user's precise numbers will differ due to Sorcha's randomization of an object's astrometry and photometry about its uncertainties. For the full list of columns that should be available in this output, we refer the user to the basic output section of Table 6.

The statistics file will look similar to this:

```
ObjID,optFilter,number_obs,min_apparent_mag,max_apparent_mag,median_apparent_mag,min_phase,max_phase,
  date_linked_MJD
2010_TC209,
  g,1,21.042222892374717,21.042222892374717,21.042222892374717,4.822674862328868,4.822674862328868,60259.0
2010_TC209,i,2,20.263741481736623,20.39869118796203,
  20.331216334849326,4.902659658311991,7.356710425955291,60259.0
2010_TC209,
  r,2,20.444480018298275,20.617951470547577,20.531215744422926,4.821141532804455,7.345792889497799,60259.0
2010_TC209,u,0,,,,,,,
2010_TC209,y,0,,,,,,,
2010_TC209,
  z,1,20.19324699340425,20.19324699340425,20.19324699340425,4.899566415676175,4.899566415676175,60259.0
2011_OB60,
  g,7,22.946130649671932,23.19931119066643,23.037745264926205,0.378554335727471,1.2668113261408172,60228.0
2011_OB60,
  i,20,22.307405543931022,22.734689782707683,22.565578703812037,0.3324387390211166,1.5654399545866802,60228.0
2011_OB60,
  r,11,22.503812312947016,22.752436749239006,22.610941659961966,0.3325287235981864,1.50119506319347,60228.0
2011_OB60,
  u,1,24.5667047017521,24.5667047017521,24.5667047017521,1.3083690945964197,1.3083690945964197,60228.0
2011_OB60,
  y,2,22.51622985410243,22.578367849124156,22.54729885161329,0.4251751188145726,0.744770018195348,60228.0
2011_OB60,
  z,11,22.262555361397094,22.7722417002819,22.38455767193579,0.3323812401128109,1.0086113081302872,60228.0
2011_YA42,
  g,55,21.988268774434818,23.17042737356163,22.645034273517435,5.779172614054609,16.858086742640804,60294.0
2011_YA42,
  i,114,21.099983644512545,22.41865729383654,21.80604194913399,4.27818434969482,16.858826689815402,60294.0
2011_YA42,
  r,119,21.27632010408789,22.60535423791973,21.992096786094883,4.27905901648886,16.858790856474048,60294.0
2011_YA42,
  u,8,23.287772926582647,24.681129730878084,23.965335308765795,4.786855583707764,16.28483860182808,60294.0
```

```
2011_YA42,
  y,103,21.039217193403374,22.324927626389975,21.830816671100585,4.289386839279333,16.626087100847716,60294.0
2011_YA42,
  z,82,21.466672869498748,22.642891414440403,22.050717444544205,4.288267265701924,16.42954705183178,60294.0
2012_HW1,
  g,4,21.28135059249862,21.921530934981835,21.306082738885898,18.24670580343961,23.799694527700037,60401.0
2012_HW1,
  i,12,19.902649799186086,22.351360183436128,20.769568042204128,15.429492575574567,32.98342142925829,60401.0
2012_HW1,
  r,7,20.516025653001538,21.380105641386635,20.748282653915954,16.941927704030164,25.057711920957264,60401.0
2012_HW1,
  u,1,22.702522463962406,22.702522463962406,22.702522463962406,18.286703173487435,18.286703173487435,60401.0
2012_HW1,y,5,20.195605736059495,21.8182694730977,20.3557245332764,15.835262453124184,28.7883450541416,60401.0
2012_HW1,
  z,9,20.484442207810996,22.260849939372516,21.961964257261524,15.419928152211138,30.44395714693862,60401.0
2013_CC 263,
  g,6,23.432040941615597,24.281615590762463,23.80201802687187,2.638120000168959,11.83521583002445,60375.0
2013_CC 263,
  i,17,22.625645785273722,24.14087814493943,23.239324069683995,2.914130006135205,18.413844880602955,60375.0
2013_CC 263,
  r,17,22.808332102472118,23.8611819839395,23.309735969610564,2.5782284903062447,15.665773562139645,60375.0
2013_CC 263,u,0,,,,
2013_CC 263,y,0,,,,
2013_CC 263,
  z,3,22.86153172084036,23.016242989892422,22.92988785440862,3.0658304290927627,11.506994018448005,60375.0
```

Once again, the user's precise numbers will differ due to `Sorcha`'s randomization of the objects' photometry and astrometry. Further explanation of the output files can be found in Section 8.

Appendix C
Example Configuration Files

We provide three example configuration files that will initialize `Sorcha` for simulating what the LSST should discover and detect using a `rubin_sim` cadence simulation as input for the survey pointing history. These example configuration files come installed with the `Sorcha Python` package and can be copied to any directory via the `sorcha demo prepare` command, as outlined in Section 11.

C.1. LSST Full Camera Footprint

This configuration file will force `Sorcha` to use the default model of the full LSST camera focal plane, accounting for detector and chip gaps. See Section 7.7.2 for further details. For most LSST science needs, we recommend using this setup for `Sorcha` unless there is a strong need to use the circular FOV approximation, such as speeding up the runtime.

```
# Sorcha Configuration File
# This configuration file will set up Sorcha to simulate what the LSST would discover.
# This version uses the full LSSTCam detector footprint to detect whether objects are lost to chip gaps.
# This is slightly slower than using a circular approximation, but more accurate.


[INPUT]

# The simulation used for the ephemeris input.
# ar=ASSIST+REBOUND interal ephemeris generation
# external=providing an external input file from the command line
# Options: "ar", "external"
ephemerides_type = ar

# Format for ephemeris simulation input file if a file is specified at the command line.
# This is also the format to which ephemeris files will be written out, if specified.
# Options: csv, whitespace, hdf5
eph_format = csv

# Sorcha chunk size: how many objects should be processed at once?
size_serial_chunk = 20000
```

```
# Format for the orbit, physical parameters, and complex physical parameters input files.
# Options: csv or whitespace
aux_format = csv

# SQL query for extracting data from the pointing database.
pointing_sql_query = SELECT observationId, observationStartMJD as observationStartMJD_TAI, visitTime, visitExposure
  Time,
 filter, seeingFwhmGeom as seeingFwhmGeom_arcsec, seeingFwhmEff as seeingFwhmEff_arcsec, fiveSigmaDepth as
 fieldFiveSigmaDepth_mag , fieldRA as fieldRA_deg, fieldDec as fieldDec_deg, rotSkyPos as fieldRotSkyPos_deg FROM
  observations order by observationId


[SIMULATION]
# Configuration for running the ASSIST+REBOUND ephemerides generator.

# the field of view of our search field, in degrees
ar_ang_fov = 2.06

# the buffer zone around the field of view we want to include, in degrees
ar_fov_buffer = 0.2

# the "picket" is our imprecise discretization of time that allows us to move progress
# our simulations forward without getting too granular when we don't have to.
# the unit is number of days.
ar_picket = 1

# the obscode is the MPC observatory code for the provided telescope.
ar_obs_code = X05

# the order of healpix which we will use for the healpy portions of the code.
# the nside is equivalent to 2**ar_healpix_order
ar_healpix_order = 6


[FILTERS]

# Filters of the observations you are interested in, comma-separated.
# Your physical parameters file must have H calculated in one of these filters
# and colour offset columns defined relative to that filter.
observing_filters = r,g,i,z,u,y


[SATURATION]

# Upper magnitude limit on sources that will overfill the detector pixels/have
# counts above the non-linearity regime of the pixels where one can't do
# photometry. Objects brighter than this limit (in magnitude) will be cut.
# Comment out for no saturation limit.
# Two formats are accepted:
# Single float: applies same saturation limit to observations in all filters.
# Comma-separated list of floats: applies saturation limit per filter, in order as
# given in observing_filters keyword.
bright_limit = 16.0


[PHASECURVES]

# The phase function used to calculate apparent magnitude. The physical parameters input
# file must contain the columns needed to calculate the phase function.
# Options: HG, HG1G2, HG12, linear, none.
phase_function = HG


[FOV]

# Choose between circular or actual camera footprint, including chip gaps.
# Options: circle, footprint.
camera_model = footprint
```

```
# The distance from the edge of a detector (in arcseconds on the focal plane)
# at which we will not correctly extract an object. By default this is 10px or 2 arcseconds.
# Comment out or do not include if not using footprint camera model.
footprint_edge_threshold = 2.

# Path to camera footprint file. Uncomment to provide a path to the desired camera
# detector configuration file if not using the default built-in LSSTCam detector
# configuration for the actual camera footprint.
# footprint_path= ./data/detectors_corners.csv


[FADINGFUNCTION]

# Uses the fading function as outlined in
# Chesley and Vereš (2017) to remove observations.

# Width parameter for fading function. Should be greater than zero and less than 0.5.
# Suggested value is 0.1 after Chesley and Vereš (2017).
fading_function_width = 0.1

# Peak efficiency for the fading function, called the 'fill factor' in Chesley and Veres (2017).
# Suggested value is 1. Do not change this unless you are sure of what you are doing.
fading_function_peak_efficiency = 1.


[LINKINGFILTER]

# SSP detection efficiency. Which fraction of the objects will
# the automated Rubin Solar System Processing (SSP) pipeline successfully link? Float.
SSP_detection_efficiency = 0.95

# Length of tracklets. How many observations of an object during one night are
# required to produce a valid tracklet?
SSP_number_observations = 2

# Minimum separation (in arcsec) between two observations of an object required
# for the linking software to distinguish them as separate and therefore as a valid tracklet.
SSP_separation_threshold = 0.5

# Maximum time separation (in days) between subsequent observations in a tracklet.
# Default is 0.0625 days (90mins).
SSP_maximum_time = 0.0625

# Number of tracklets for detection. How many tracklets are required to classify
# an object as detected?
SSP_number_tracklets = 3

# The number of tracklets defined above must occur in <= this number of days to
# constitute a complete track/detection.
SSP_track_window = 15

# The time in UTC at which it is noon at the observatory location (in standard time).
# For the LSST, 12pm Chile Standard Time is 4pm UTC.
SSP_night_start_utc = 16.0


[OUTPUT]

# Output format of the output file[s]
# Options: csv, sqlite3, hdf5
output_format = sqlite3

# Controls which columns are in the output files.
# Options are "basic" and "all", which returns all columns.
output_columns = basic
```

```
[LIGHTCURVE]

# The unique name of the lightcurve model to use. Defined in the "name_id" method
# of the subclasses of AbstractLightCurve. If not none, the complex physical parameters
# file must be specified at the command line.lc_model = none
lc_model = none


[ACTIVITY]

# The unique name of the actvity model to use. Defined in the "name_id" method
# of the subclasses of AbstractCometaryActivity. If not none, a complex physical parameters
# file must be specified at the command line.
comet_activity = none
```

C.2. LSST Circular Footprint Approximation

This configuration file uses the circular footprint filter discussed in Section 7.7.1. This setup is marginally faster than using the full LSSTCam footprint filter; for most science cases, however, we recommend the user use the configurations laid out in the previous file.

```
# Sorcha Configuration File
# This configuration file will set up Sorcha to simulate what the LSST would discover.
# This version simulates objects lost to chip gaps between LSSTCam's detectors by
# simulating a circular detector and approximating the chip gaps, removing all objects
# outside of a 1.75deg radius and 10 # This is slightly faster than using the full footprint.


[INPUT]

# The simulation used for the ephemeris input.
# ar=ASSIST+REBOUND interal ephemeris generation
# external=providing an external input file from the command line
# Options: "ar", "external"
ephemerides_type = ar

# Format for ephemeris simulation input file if a file is specified at the command line.
# This is also the format to which ephemeris files will be written out, if specified.
# Options: csv, whitespace, hdf5
eph_format = csv

# Sorcha chunk size: how many objects should be processed at once?
size_serial_chunk = 20000

# Format for the orbit, physical parameters, and complex physical parameters input files.
# Options: csv or whitespace
aux_format = csv

# SQL query for extracting data from the pointing database.
pointing_sql_query = SELECT observationId, observationStartMJD as observationStartMJD_TAI, visitTime,
  visitExposureTime,
 filter, seeingFwhmGeom as seeingFwhmGeom_arcsec, seeingFwhmEff as seeingFwhmEff_arcsec, fiveSigmaDepth as
 fieldFiveSigmaDepth_mag , fieldRA as fieldRA_deg, fieldDec as fieldDec_deg, rotSkyPos as fieldRotSkyPos_deg FROM
  observations order by observationId


[SIMULATION]
# Configuration for running the ASSIST+REBOUND ephemerides generator.

# the field of view of our search field, in degrees
ar_ang_fov = 2.06

# the buffer zone around the field of view we want to include, in degrees
ar_fov_buffer = 0.2

# the "picket" is our imprecise discretization of time that allows us to move progress
# our simulations forward without getting too granular when we don't have to.
```

```
# the unit is number of days.
ar_picket = 1

# the obscode is the MPC observatory code for the provided telescope.
ar_obs_code = X05

# the order of healpix which we will use for the healpy portions of the code.
# the nside is equivalent to 2**ar_healpix_order
ar_healpix_order = 6


[FILTERS]

# Filters of the observations you are interested in, comma-separated.
# Your physical parameters file must have H calculated in one of these filters
# and colour offset columns defined relative to that filter.
observing_filters = r,g,i,z,u,y


[SATURATION]

# Upper magnitude limit on sources that will overfill the detector pixels/have
# counts above the non-linearity regime of the pixels where one can't do
# photometry. Objects brighter than this limit (in magnitude) will be cut.
# Comment out for no saturation limit.
# Two formats are accepted:
# Single float: applies same saturation limit to observations in all filters.
# Comma-separated list of floats: applies saturation limit per filter, in order as
# given in observing_filters keyword.
bright_limit = 16.0


[PHASECURVES]

# The phase function used to calculate apparent magnitude. The physical parameters input
# file must contain the columns needed to calculate the phase function.
# Options: HG, HG1G2, HG12, linear, none.
phase_function = HG


[FOV]

# Choose between circular or actual camera footprint, including chip gaps.
# Options: circle, footprint.
camera_model = circle

# Fraction of detector surface area which contains CCD—simulates chip gaps
# for OIF output. Comment out if using camera footprint.
fill_factor = 0.9

# Radius of the circle for a circular footprint (in degrees). Float.
# Comment out or do not include if using footprint camera model.
circle_radius = 1.75


[FADINGFUNCTION]

# Uses the fading function as outlined in
# Chesley and Vereš (2017) to remove observations.

# Width parameter for fading function. Should be greater than zero and less than 0.5.
# Suggested value is 0.1 after Chesley and Vereš (2017).
fading_function_width = 0.1

# Peak efficiency for the fading function, called the 'fill factor' in Chesley and Veres (2017).
# Suggested value is 1. Do not change this unless you are sure of what you are doing.
fading_function_peak_efficiency = 1.
```

```
[LINKINGFILTER]

# SSP detection efficiency. Which fraction of the objects will
# the automated Rubin Solar System Processing (SSP) pipeline successfully link? Float.
SSP_detection_efficiency = 0.95


# Length of tracklets. How many observations of an object during one night are
# required to produce a valid tracklet?
SSP_number_observations = 2

# Minimum separation (in arcsec) between two observations of an object required
# for the linking software to distinguish them as separate and therefore as a valid tracklet.
SSP_separation_threshold = 0.5

# Maximum time separation (in days) between subsequent observations in a tracklet.
# Default is 0.0625 days (90mins).
SSP_maximum_time = 0.0625

# Number of tracklets for detection. How many tracklets are required to classify
# an object as detected?
SSP_number_tracklets = 3

# The number of tracklets defined above must occur in <= this number of days to
# constitute a complete track/detection.
SSP_track_window = 15

# The time in UTC at which it is noon at the observatory location (in standard time).
# For the LSST, 12pm Chile Standard Time is 4pm UTC.
SSP_night_start_utc = 16.0


[OUTPUT]

# Output format of the output file[s]
# Options: csv, sqlite3, hdf5
output_format = sqlite3

# Controls which columns are in the output files.
# Options are "basic" and "all", which returns all columns.
output_columns = basic


[LIGHTCURVE]

# The unique name of the lightcurve model to use. Defined in the "name_id" method
# of the subclasses of AbstractLightCurve. If not none, the complex physical parameters
# file must be specified at the command line.
lc_model = none


[ACTIVITY]

# The unique name of the actvity model to use. Defined in the "name_id" method
# of the subclasses of AbstractCometaryActivity. If not none, a complex physical parameters
# file must be specified at the command line.
comet_activity = none
```

C.3. LSST Known Object Predictions

This configuration file will run `Sorcha` with the default model of the full LSST camera focal plane, accounting for detector and chip gaps, but with randomization, fading function, vignetting, SSP linking, saturation limit, and trailing losses turned off. This will result in output listing all detections that lie on the CCDs with unadulterated apparent magnitudes. This configuration file is suitable for instances where the user only cares about the locations of the population of interest; for example, if one wishes to predict where

and when known objects appear in Rubin observations. As this configuration turns off almost all of Sorcha's postprocessing steps, we do not recommend its use for science.

```
# Sorcha Configuration File
# This configuration file will set up Sorcha to simulate what the LSST would discover.
# This version uses the full LSSTCam detector footprint to detect whether objects are lost to chip gaps.
# This is slightly slower than using a circular approximation, but more accurate.


[INPUT]

# The simulation used for the ephemeris input.
# ar=ASSIST+REBOUND interal ephemeris generation
# external=providing an external input file from the command line
# Options: "ar", "external"
ephemerides_type = ar

# Format for ephemeris simulation input file if a file is specified at the command line.
# This is also the format to which ephemeris files will be written out, if specified.
# Options: csv, whitespace, hdf5
eph_format = csv

# Sorcha chunk size: how many objects should be processed at once?
size_serial_chunk = 20000

# Format for the orbit, physical parameters, and complex physical parameters input files.
# Options: csv or whitespace
aux_format = csv

# SQL query for extracting data from the pointing database.
pointing_sql_query = SELECT observationId, observationStartMJD as observationStartMJD_TAI, visitTime,
  visitExposureTime,
 filter, seeingFwhmGeom as seeingFwhmGeom_arcsec, seeingFwhmEff as seeingFwhmEff_arcsec, fiveSigmaDepth as
 fieldFiveSigmaDepth_mag , fieldRA as fieldRA_deg, fieldDec as fieldDec_deg, rotSkyPos as fieldRotSkyPos_deg FROM
  observations order by observationId


[SIMULATION]
# Configuration for running the ASSIST+REBOUND ephemerides generator.

# the field of view of our search field, in degrees
ar_ang_fov = 2.06

# the buffer zone around the field of view we want to include, in degrees
ar_fov_buffer = 0.2

# the "picket" is our imprecise discretization of time that allows us to move progress
# our simulations forward without getting too granular when we don't have to.
# the unit is number of days.
ar_picket = 1

# the obscode is the MPC observatory code for the provided telescope.
ar_obs_code = X05

# the order of healpix which we will use for the healpy portions of the code.
# the nside is equivalent to 2**ar_healpix_order
ar_healpix_order = 6


[FILTERS]

# Filters of the observations you are interested in, comma-separated.
# Your physical parameters file must have H calculated in one of these filters
# and colour offset columns defined relative to that filter.
observing_filters = r,g,i,z,u,y
```

```
[SATURATION]

# Upper magnitude limit on sources that will overfill the detector pixels/have
# counts above the non-linearity regime of the pixels where one can't do
# photometry. Objects brighter than this limit (in magnitude) will be cut.
# Comment out for no saturation limit.
# Two formats are accepted:
# Single float: applies same saturation limit to observations in all filters.
# Comma-separated list of floats: applies saturation limit per filter, in order as
# given in observing_filters keyword.
bright_limit = 16.0


[PHASECURVES]

# The phase function used to calculate apparent magnitude. The physical parameters input
# file must contain the columns needed to calculate the phase function.
# Options: HG, HG1G2, HG12, linear, none.
phase_function = HG


[FOV]

# Choose between circular or actual camera footprint, including chip gaps.
# Options: circle, footprint.
camera_model = footprint

# The distance from the edge of a detector (in arcseconds on the focal plane)
# at which we will not correctly extract an object. By default this is 10px or 2 arcseconds.
# Comment out or do not include if not using footprint camera model.
footprint_edge_threshold = 2.

# Path to camera footprint file. Uncomment to provide a path to the desired camera
# detector configuration file if not using the default built-in LSSTCam detector
# configuration for the actual camera footprint.
# footprint_path= ./data/detectors_corners.csv


[FADINGFUNCTION]

# Uses the fading function as outlined in
# Chesley and Vereš (2017) to remove observations.

# Width parameter for fading function. Should be greater than zero and less than 0.5.
# Suggested value is 0.1 after Chesley and Vereš (2017).
fading_function_width = 0.1

# Peak efficiency for the fading function, called the 'fill factor' in Chesley and Veres (2017).
# Suggested value is 1. Do not change this unless you are sure of what you are doing.
fading_function_peak_efficiency = 1.


[LINKINGFILTER]

# SSP detection efficiency. Which fraction of the objects will
# the automated Rubin Solar System Processing (SSP) pipeline successfully link? Float.
SSP_detection_efficiency = 0.95

# Length of tracklets. How many observations of an object during one night are
# required to produce a valid tracklet?
SSP_number_observations = 2

# Minimum separation (in arcsec) between two observations of an object required
# for the linking software to distinguish them as separate and therefore as a valid tracklet.
SSP_separation_threshold = 0.5
```

```
# Maximum time separation (in days) between subsequent observations in a tracklet.
# Default is 0.0625 days (90mins).
SSP_maximum_time = 0.0625

# Number of tracklets for detection. How many tracklets are required to classify
# an object as detected?
SSP_number_tracklets = 3

# The number of tracklets defined above must occur in <= this number of days to
# constitute a complete track/detection.
SSP_track_window = 15

# The time in UTC at which it is noon at the observatory location (in standard time).
# For the LSST, 12pm Chile Standard Time is 4pm UTC.
SSP_night_start_utc = 16.0


[OUTPUT]

# Output format of the output file[s]
# Options: csv, sqlite3, hdf5
output_format = sqlite3

# Controls which columns are in the output files.
# Options are "basic" and "all", which returns all columns.
output_columns = basic


[LIGHTCURVE]

# The unique name of the lightcurve model to use. Defined in the "name_id" method
  # of the subclasses of AbstractLightCurve. If not none, the complex physical parameters
  # file must be specified at the command line.lc_model = none lc_model = none

[ACTIVITY]

# The unique name of the actvity model to use. Defined in the "name_id" method
# of the subclasses of AbstractCometaryActivity. If not none, a complex physical parameters
# file must be specified at the command line.
comet_activity = none

[EXPERT]
# Turning off all randomization, vignetting and trailing losses.
randomization_on = False
vignetting_on = False
trailing_losses_on = False
```

Appendix D
Magnitude Uncertainty Calculations

Throughout Sorcha, we use a few different approximations to the expected magnitude uncertainty at a given signal-to-noise. Here, we show that all of these formulations are equivalent, and any individual choice is purely a matter of convenience. By definition for a flux measurement f with uncertainty σ_f, the signal-to-noise is SNR $\equiv f/\sigma_f$,

For a nontrailed source with flux f and magnitude $m \equiv -2.5\log_{10}(f) + c$ (for some constant term c), we begin by directly deriving the magnitude m' after applying a small perturbation $\pm\delta f$ where $|\delta f < |f|$:

$$\begin{aligned} m' &= -2.5\log_{10}(f \pm \delta f) + c \\ &= -2.5\log_{10} f + c - 2.5\log_{10}(1 \pm \delta f/f) \\ &= m \mp 2.5\log_{10}(1 + \delta f/f). \end{aligned} \tag{D1}$$

If we take $\delta f = \sigma_f$, then, we have that $m' = m \pm \sigma_m$, where

$$\sigma_m = 2.5\log_{10}(1 + (\mathrm{SNR})^{-1}). \tag{D2}$$

We can also derive this uncertainty with the usual uncertainty propagation by the Jacobian of the transformation. In our case, we have that:

$$\sigma_m^2 = \left(\frac{\partial f}{\partial m}\right)^2 \sigma_f^2 = \left(\frac{2.5}{\ln(10)} \times \frac{1}{f}\right)^2 \sigma_f^2 \tag{D3}$$

$$\Rightarrow \sigma_m = \left| \frac{2.5}{\ln(10)} \frac{\sigma_f}{f} \right| \approx 1/(\mathrm{SNR}). \tag{D4}$$

To see that these two results are equivalent, we take the Taylor series of Equation (D2), leading to

$$-2.5\log_{10}(1 \pm 1/x) \approx \frac{2.5}{x\ln(10)}, \tag{D5}$$

where $x = \mathrm{SNR}$.

Finally, we derive the relationship between $m_{5\sigma}$ and SNR of Ž. Ivezić et al. (2019). From Poisson statistics, we have that

$$N^2 = N_0^2 + \alpha S, \tag{D6}$$

where N_0 is a constant noise floor and α is an instrumental scaling factor dependent on the shot noise. Then,

$$\sigma_m^2 \approx \frac{1}{(\mathrm{SNR})^2} = \frac{N_0^2}{S^2} + \frac{\alpha}{S}. \tag{D7}$$

By definition, $m_{5\sigma}$ is the magnitude where SNR = 5, where we expect that $\sigma_m = 0.2$. Under this condition, and assuming that $S \propto 10^{-0.4m}$, we define $y \equiv 10^{0.4(m-m_{5\sigma})}$

$$\sigma_m^2 = (0.2^2 - \gamma)y + \gamma y^2. \tag{D8}$$

γ, similarly to α, depends on image quality and instrumental properties (Ž. Ivezić et al. 2019). The term linear in y accounts for the uncertainty floor for bright sources (note that y is small for $m \ll m_{5\sigma}$), whereas the quadratic term dominates in the regime $m \approx m_{5\sigma}$, where we expect the shot noise will dominate. We note that, because of the typical steepness of solar system absolute magnitude distributions, the majority of sources will be near $m_{5\sigma}$.

ORCID iDs

Stephanie R. Merritt https://orcid.org/0000-0001-5930-2829
Grigori Fedorets https://orcid.org/0000-0002-8418-4809
Megan E. Schwamb https://orcid.org/0000-0003-4365-1455
Samuel Cornwall https://orcid.org/0000-0002-0672-5104
Pedro H. Bernardinelli https://orcid.org/0000-0003-0743-9422
Mario Jurić https://orcid.org/0000-0003-1996-9252
Matthew J. Holman https://orcid.org/0000-0002-1139-4880
Jacob A. Kurlander https://orcid.org/0009-0005-5452-0671
Siegfried Eggl https://orcid.org/0000-0002-1398-6302
Drew Oldag https://orcid.org/0000-0001-6984-8411
Maxine West https://orcid.org/0009-0003-3171-3118
Jeremy Kubica https://orcid.org/0009-0009-2281-7031
Joseph Murtagh https://orcid.org/0000-0001-9505-1131
R. Lynne Jones https://orcid.org/0000-0001-5916-0031
Peter Yoachim https://orcid.org/0000-0003-2874-6464
Ryan R. Lyttle https://orcid.org/0009-0007-8602-2954
Michael S. P. Kelley https://orcid.org/0000-0002-6702-7676
Joachim Moeyens https://orcid.org/0000-0001-5820-3925
Shantanu P. Naidu https://orcid.org/0000-0003-4439-7014
Colin Snodgrass https://orcid.org/0000-0001-9328-2905
Shannon M. Matthews https://orcid.org/0000-0001-8633-9141
Colin Orion Chandler https://orcid.org/0000-0001-7335-1715

References

Abbott, T. M. C., Adamów, M., Aguena, M., et al. 2021, ApJS, 255, 20
Acton, C., Bachman, N., Semenov, B., & Wright, E. 2018, P&SS, 150, 9
Acton, C. H. 1996, P&SS, 44, 65
Agarwal, J., Kim, Y., Kelley, M. S. P., & Marschall, R. 2024, in Comets III, ed. K. J. Meech et al. (Tucson, AZ: Univ. Arizona Press)
A'Hearn, M. F., Schleicher, D. G., Millis, R. L., Feldman, P. D., & Thompson, D. T. 1984, AJ, 89, 579
Alvarez-Candal, A., Jimenez Corral, S., & Colazo, M. 2022, A&A, 667, A81
Annex, A., Pearson, B., Seignovert, B., et al. 2020, JOSS, 5, 2050
Annis, J., Soares-Santos, M., Strauss, M. A., et al. 2014, ApJ, 794, 120
Araujo-Hauck, C., Sebag, J., Liang, M., et al. 2016, Proc. SPIE, 9906, 202
Ashton, E., Chang, C.-K., Chen, Y.-T., et al. 2023, ApJS, 267, 33
Astropy Collaboration, Price-Whelan, A. M., Sipőcz, B. M., et al. 2018, AJ, 156, 123
Astropy Collaboration, Price-Whelan, A. M., Lim, P. L., et al. 2022, ApJ, 935, 167
Astropy Collaboration, Robitaille, T. P., Tollerud, E. J., et al. 2013, A&A, 558, A33
Bannister, M. T., Kavelaars, J. J., Petit, J.-M., et al. 2016, AJ, 152, 70
Bannister, M. T., Gladman, B. J., Kavelaars, J. J., et al. 2018, ApJS, 236, 18
Bannister, M. T., Shankman, C., Volk, K., et al. 2017, AJ, 153, 262
Batygin, K., Adams, F. C., Brown, M. E., & Becker, J. C. 2019, PhR, 805, 1
Batygin, K., & Brown, M. E. 2016, AJ, 151, 22
Belyakov, M., Bernardinelli, P. H., & Brown, M. E. 2022, AJ, 163, 216
Bernardinelli, P. H., Bernstein, G. M., Sako, M., et al. 2020, PSJ, 1, 28
Bernardinelli, P. H., Bernstein, G. M., Sako, M., et al. 2022, ApJS, 258, 41
Bernardinelli, P. H., Smotherman, H., Langford, Z., et al. 2024, AJ, 167, 134
Bianco, F. & the SCOC 2022, Survey Cadence Optimization Committee's Phase 2 Recommendations, https://pstn-055.lsst.io/
Bianco, F. & the SCOC 2024, Survey Cadence Optimization Committee's Phase 3 Recommendations, https://pstn-056.lsst.io/
Bianco, F. B., Ivezić, Ž., Jones, R. L., et al. 2022, ApJS, 258, 1
Bottke, W. F., Morbidelli, A., Jedicke, R., et al. 2002, Icar, 156, 399
Bottke, W. F., Jr., Vokrouhlický, D., Rubincam, D. P., & Nesvorný, D. 2006, AREPS, 34, 157
Bowell, E., Hapke, B., Domingue, D., et al. 1989, in Asteroids II, ed. R. P. Binzel, T. Gehrels, & M. S. Matthews (Tucson, AZ: Univ. Arizona Press), 524
Brown, M. E., & Batygin, K. 2019, AJ, 157, 62
Brown, M. E., & Batygin, K. 2021, AJ, 162, 219
Brown, M. E., Holman, M. J., & Batygin, K. 2024, AJ, 167, 146
Buchanan, L. E., Schwamb, M. E., Fraser, W. C., et al. 2022, PSJ, 3, 9
Carry, B., Peloton, J., Le Montagner, R., Mahlke, M., & Berthier, J. 2024, A&A, 687, A38
Carvano, J. M., & Davalos, J. A. G. 2015, A&A, 580, A98
Chang, C.-K., Chen, Y.-T., Fraser, W. C., et al. 2021, PSJ, 2, 191
Cochran, A. L., Green, J. R., & Barker, E. S. 1989, Icar, 79, 125
Connolly, A. J., Angeli, G. Z., Chandrasekharan, S., et al. 2014, Proc. SPIE, 9150, 14
Cook, N. V., Ragozzine, D., Granvik, M., & Stephens, D. C. 2016, ApJ, 825, 51
da Costa-Luis, C., Larroque, S. K., Altendorf, K., et al. 2023, tqdm: A fast, Extensible Progress Bar for Python and CLI, v4.66.1, Zenodo, doi:10.5281/zenodo.8233425
Delgado, F., Saha, A., Chandrasekharan, S., et al. 2014, Proc. SPIE, 9150, 915015
Denneau, L., Jedicke, R., Fitzsimmons, A., et al. 2015, Icar, 245, 1
Dobson, M. M., Schwamb, M. E., Benecchi, S. D., et al. 2023, PSJ, 4, 75
Dobson, M. M., Schwamb, M. E., Fitzsimmons, A., et al. 2021, RNAAS, 5, 211
Ďurech, J., Sidorin, V., Kaasalainen, M., et al. 2010, A&A, 513, A46
Ďurech, J., Vávra, M., Vančo, R., & Erasmus, N. 2022, FrASS, 9, 809771
Engelhardt, T., Jedicke, R., Vereš, P., et al. 2017, AJ, 153, 133
Fedorets, G., Granvik, M., Jones, R. L., Jurić, M., & Jedicke, R. 2020, Icar, 338, 113517
Flaugher, B., Diehl, H. T., Honscheid, K., et al. 2015, AJ, 150, 150
Fraser, W. C., Benecchi, S. D., Kavelaars, J. J., et al. 2021, PSJ, 2, 90
Fraser, W. C., Pike, R. E., Marsset, M., et al. 2023, PSJ, 4, 80
Gladman, B., Kavelaars, J. J., Nicholson, P. D., Loredo, T. J., & Burns, J. A. 1998, AJ, 116, 2042
Gladman, B., Lawler, S. M., Petit, J.-M., et al. 2012, AJ, 144, 23
Gladman, B. J., Davis, D. R., Neese, C., et al. 2009, Icar, 202, 104
Górski, K. M., Hivon, E., Banday, A. J., et al. 2005, ApJ, 622, 759
Grav, T., Mainzer, A. K., & Spahr, T. 2016, AJ, 151, 172
Harris, C. R., Millman, K. J., van der Walt, S. J., et al. 2020, Natur, 585, 357
Hasegawa, S., Marsset, M., DeMeo, F. E., et al. 2024, AJ, 167, 224
Holman, M. J., Akmal, A., Farnocchia, D., et al. 2023, PSJ, 4, 69
Holman, M. J., Bernardinelli, P. H., Schwamb, M. E., et al. 2025, AJ, 170, 97
Holman, M. J., Payne, M. J., Blankley, P., Janssen, R., & Kuindersma, S. 2018, AJ, 156, 135
Hoover, D. J., Seligman, D. Z., & Payne, M. J. 2022, PSJ, 3, 71
Hunter, J. D. 2007, CSE, 9, 90

Ivezić, Ž. & the LSST Science Collaboration 2013, LSST Science Requirements Document, http://ls.st/LPM-17
Ivezić, Ž. & the SCOC 2021, Survey Cadence Optimization Committee's Phase 1 Recommendation, https://pstn-053.lsst.io/
Ivezić, Ž., Kahn, S. M., Tyson, J. A., et al. 2019, ApJ, 873, 111
Jackson, S. L., Rozitis, B., Dover, L. R., et al. 2022, MNRAS, 513, 3076
Jewitt, D., & Hsieh, H. H. 2024, in Comets III, ed. K. J. Meech et al. (Tucson, AZ: Univ. Arizona Press)
Jones, R. L., Gladman, B., Petit, J. M., et al. 2006, Icar, 185, 508
Jones, R. L., Chesley, S. R., Connolly, A. J., et al. 2009, EM&P, 105, 101
Jones, R. L., Slater, C. T., Moeyens, J., et al. 2018, Icar, 303, 181
Jurić, M., Axelrod, T., Becker, A., et al. 2021, Data Products Definition Document, https://lse-163.lsst.io/
Jurić, M., Eggl, S., Moeyens, J., & Jones, R. L. 2020, Proposed Modifications to Solar System Processing and Data Products, https://dmtn-087.lsst.io
Jurić, M., Kantor, J., Lim, K. T., 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
Kavelaars, J. J., Jones, R. L., Gladman, B. J., et al. 2009, AJ, 137, 4917
Kavelaars, J. J., Petit, J.-M., Gladman, B., et al. 2021, ApJL, 920, L28
Kluyver, T., Ragan-Kelley, B., Pérez, F., et al. 2016, Positioning and Power in Academic Publishing: Players, Agents and Agendas (Amsterdam: IOS Press), 87, https://eprints.soton.ac.uk/403913/
Kubica, J., Denneau, L., Grav, T., et al. 2007, Icar, 189, 151
Kurlander, J. A., Holman, M. J., Bernardinelli, P. H., et al. 2025, AJ, 169, 73
Kwiatkowski, T., & Kryszczynska, A. 1992, in Liege Int. Astrophysical Coll. 30, ed. A. Brahic, J. C. Gerard, & J. Surdej (Liege: Universite de Liege), 353
Lam, S. K., Pitrou, A., & Seibert, S. 2015, in The LLVM Compiler Infrastructure in HPC (New York: ACM), 1
Lawler, S. M., Kavelaars, J. J., Alexandersen, M., et al. 2018a, FrASS, 5, 14
Lawler, S. M., Shankman, C., Kavelaars, J. J., et al. 2018b, AJ, 155, 197
Lay, O., Masiero, J., Grav, T., et al. 2024, PSJ, 5, 149
Lazzarin, M., Magrin, S., Marchi, S., et al. 2010, MNRAS, 408, 1433
Levine, W. G., Cabot, S. H. C., Seligman, D., & Laughlin, G. 2021, ApJ, 922, 39
LSST Science Collaboration, Abell, P. A., Allison, J., et al. 2009, arXiv:0912.0201
Luu, J., & Jewitt, D. 1989, AJ, 98, 1905
Luu, J. X., & Jewitt, D. 1988, AJ, 95, 1256
Lykawka, P. S., & Ito, T. 2023, AJ, 166, 118
Mahlke, M., Carry, B., & Denneau, L. 2021, Icar, 354, 114094
Masiero, J. R., Dahlen, D. W., Mainzer, A. K., et al. 2023, PSJ, 4, 225
Masiero, J. R., Linder, T., Mainzer, A., Dahlen, D. W., & Kwon, Y. G. 2024, PSJ, 5, 222
McKinney, W. 2010, in Proc. of the 9th Python in Science Conf., ed. S. van der Walt & J. Millman (Austin, TX: SciPy), 56
McLoughlin, E., Fitzsimmons, A., & McLoughlin, A. 2015, Icar, 256, 37
Merritt, S., Fedorets, G., Schwamb, M. E., et al. 2025, JOSS, 10, 1
Mommert, M., Kelley, M., de Val-Borro, M., et al. 2019, JOSS, 4, 1426
Moro-Martín, A., Turner, E. L., & Loeb, A. 2009, ApJ, 704, 733
Muinonen, K., Belskaya, I. N., Cellino, A., et al. 2010, Icar, 209, 542
Murtagh, J., Schwamb, M. E., Merritt, S. R., et al. 2025, AJ, 170, 98
Myers, J., Jones, R. L., & Axelrod, T. 2013, Moving Object Pipeline System Design, https://docushare.lsst.org/docushare/dsweb/Get/Version-24308/LDM-156.pdf
Naghib, E., Yoachim, P., Vanderbei, R. J., Connolly, A. J., & Jones, R. L. 2019, AJ, 157, 151
Napier, K. J., Gerdes, D. W., Lin, H. W., et al. 2021, PSJ, 2, 59
O'Keefe, J. A. 1976, STIA, 77, 14534
Oldag, D., DeLucchi, M., Beebe, W., et al. 2024, RNAAS, 8, 141
Oldroyd, W. J., & Trujillo, C. A. 2021, AJ, 162, 39
Paddack, S. J. 1969, JGR, 74, 4379
Penttilä, A., Shevchenko, V. G., Wilkman, O., & Muinonen, K. 2016, P&SS, 123, 117
Petit, J. M., Kavelaars, J. J., Gladman, B. J., et al. 2011, AJ, 142, 131
Pike, R. E., Fraser, W. C., Volk, K., et al. 2023, PSJ, 4, 200
Pinilla-Alonso, N., Licandro, J., Brunetto, R., et al. 2024, A&A, 692, L11
Pokorný, P., Kuchner, M. J., & Sheppard, S. S. 2020, PSJ, 1, 47
PyTables Developers Team 2002, PyTables: Hierarchical Datasets in Python, https://www.pytables.org/
Radzievskii, V. V. 1952, AZh, 29, 162
Rein, H., Holman, M., & Akmal, A. 2023, matthewholman/assist: v1.1.1 Zenodo, doi:10.5281/zenodo.7778017
Rein, H., & Liu, S. F. 2012, A&A, 537, A128
Rein, H., & Spiegel, D. S. 2015, MNRAS, 446, 1424
Robinson, J. E., Fitzsimmons, A., Young, D. R., et al. 2024, MNRAS, 531, 304
Schemel, M., & Brown, M. E. 2021, PSJ, 2, 40
Schwamb, M. E., Brown, M. E., Rabinowitz, D. L., & Ragozzine, D. 2010, ApJ, 720, 1691
Schwamb, M. E., Fraser, W. C., Bannister, M. T., et al. 2019, ApJS, 243, 12
Schwamb, M. E., Jurić, M., Bolin, B. T., et al. 2021, RNAAS, 5, 143
Schwamb, M. E., Jones, R. L., Yoachim, P., et al. 2023, ApJS, 266, 22
Schwamb, M. E., Kubica, J., Juric, M., et al. 2024, RAAS, 8, 25
Schwamb, M. E., Levison, H. F., & Buie, M. W. 2018, RAAS, 2, 159
Schwamb, M. E., Volk, K., Wen, H., et al. 2018, arXiv:1812.01149
Seligman, D., & Laughlin, G. 2018, AJ, 155, 217
Shankman, C., Kavelaars, J. J., Bannister, M. T., et al. 2017, AJ, 154, 50
Shannon, A., Jackson, A. P., Veras, D., & Wyatt, M. 2015, MNRAS, 446, 2059
Sheppard, S. S., Tholen, D. J., Pokorný, P., et al. 2022, AJ, 164, 168
Sheppard, S. S., & Trujillo, C. 2016, AJ, 152, 221
Showalter, M. R., Benecchi, S. D., Buie, M. W., et al. 2021, Icar, 356, 114098
Silsbee, K., & Tremaine, S. 2016, AJ, 152, 103
Snodgrass, C., & Jones, G. H. 2019, NatCo, 10, 5418
Solontoi, M., Ivezić, Ž., West, A. A., et al. 2010, Icar, 205, 605
Stuart, J. S., & Binzel, R. P. 2004, Icar, 170, 295
The pandas development team 2020, pandas-dev/pandas: Pandas, Zenodo, doi:10.5281/zenodo.3509134
Trilling, D. E., Bellm, E. C., & Malhotra, R. 2018, AJ, 155, 243
Trilling, D. E., Gerdes, D. W., Jurić, M., et al. 2024, AJ, 167, 132
Trilling, D. E., Valdes, F., Allen, L., et al. 2017, AJ, 154, 170
Trujillo, C. A., & Sheppard, S. S. 2014, Natur, 507, 471
Uieda, L., Soler, S., Rampin, R., et al. 2020, JOSS, 5, 1943
Vereš, P., & Chesley, S. R. 2017a, AJ, 154, 12
Vereš, P., & Chesley, S. R. 2017b, AJ, 154, 13
Virtanen, P., Gommers, R., Oliphant, T. E., et al. 2020, NatMe, 17, 261
Vokrouhlický, D., Bottke, W. F., Chesley, S. R., Scheeres, D. J., & Statler, T. S. 2015, in Asteroids IV, ed. P. Michel, F. E. DeMeo, & W. F. Bottke (Tucson, AZ: Univ. Arizona Press), 509
Willmer, C. N. A. 2018, ApJS, 236, 47
Yoachim, P., Jones, L., Eric, H., Neilsen, J., et al. 2024, lsst/rubin_scheduler: v3.4.0, Zenodo, doi:10.5281/zenodo.14232232
Yoachim, P., Jones, L., Eric, H., Neilsen, J., et al. 2023, lsst/rubin_sim: v2.0.0, Zenodo, doi:10.5281/zenodo.10215451
Zavodny, M., Jedicke, R., Beshore, E. C., Bernardi, F., & Larson, S. 2008, Icar, 198, 284
Zonca, A., Singer, L., Lenz, D., et al. 2019, JOSS, 4, 1298