Accelerating and enabling discovery in the decade of astronomical surveys

Dino Bektešević

A dissertation
submitted in partial fulfillment of the
requirements for the degree of

Doctor of Philosophy

University of Washington

2025

Reading Committee:

Andrew Connolly, Chair

Željko Ivezić

Rory Barnes

Program Authorized to Offer Degree:
Astronomy

University of Washington

Abstract

Accelerating and enabling discovery in the decade of astronomical surveys

Dino Bektešević

Chair of the Supervisory Committee:
Andrew Connolly
Department of Astronomy

With the advent of a new generation of astronomical surveys such as the Legacy Survey of Space and Time (LSST) from the Rubin Observatory astronomers will have access to a wealth of data. If we are to fully exploit the data these surveys will generate, we will need to develop novel algorithmic approaches for analyzing astronomical images and tools to scale these algorithms to petabytes of data. In this thesis I focus on these two aspects scaling novel algorithms to extract more science from Rubin data than was previously possible, and developing a novel approach to the analysis and classification of lightcurves.

The challenges in scaling to petabytes of data are multi-faceted. The Vera C. Rubin Science Pipelines are a collection of algorithms and a workflow management functionality intended to be used to process data taken by the Rubin Observatory's Legacy Survey of Space (LSST). In the first chapter of this thesis I describe how we implemented an Amazon Web Services and Google Computing Services compliant cloud service backend for the Rubin Middleware components that enable executing the Rubin Science Pipelines on cloud resources. I demonstrate how for short-term projects with a large in-going dataset and a small out-going data volume of results the cloud is almost always cost effective. Analysis results may be retrieved much sooner by allocating more resources, than they would when allocating less compute resources, at the same cost.

In chapter 3 I demonstrate how new algorithms can be used to improve on the amount of science delivered by processing Rubin-like data on Rubin-like scales. Kernel Based Moving

Object Detection package (KBMOD) is a tool developed to perform searches for moving objects on collections of images using a shift-and-stack method along linear trajectories. Smotherman et al. (2023) demonstrated it can detect objects below the SNR of a single exposure. In this chapter I discuss work required to improve the performance of KBMOD and execute it on all of DEEP's data ($\approx$ 200TB, equivalent to 10 nights of Rubin data), while simultaneously being able to increase the range of searched angles by a 100% and the range of searched velocities by an additional 38% compared to the previous search. Compared to Smotherman et al. (2023), we achieve a 10% higher peak detection efficiency but a 0.3 magnitude lower limiting magnitude at which 50% of the objects were recovered. We identify the higher filtering threshold values, chosen due to a large number of estimated returned results, as the key culprit for the loss of limiting magnitude.

In the final chapter of the thesis I develop a new approach to time series classification. By applying ideas inherited from differential geometry on Riemannian manifolds I demonstrate how it's possible to construct a measure of distance between two curves based on nothing more than their shape. I consider two distance measures Square Root Velocity and varifold fidelity measures. The latter is robust different light curve parameterizations. Multiple classification schemes are constructed based on these distances including agglomerative (hierachical clustering), fitting a sum of Gaussians, and a K-Means like algorithm to find the generalized means (Frechet means). Classification accuracy for a high SNR dataset was 96.58%, 95.9% and 98.93% for the sums of Gaussians, agglomerative clustering and K-Means approach respectively. Validation of the approach on the PLAsTICC lightcurves dataset was less successful, achieving a top classification accuracy of 77.03%. Two key reasons for the drop in classification accuracy are identified: heteroskedastic uncertainties in the data, and reparameterizations of curves.

DEDICATION

To my wife, Agi, without whom the completion of this thesis would not be possible.

ACKNOWLEDGMENTS

The author wishes to express sincere appreciation to his advisor, supervisory committee and the University of Washington. The author wishes to thank the graduate students, postdoctoral researchers, team members of the AWS PoC team, team members of the Rubin Middleware team, and LINCC team for their collaboration, patience, and help in achieving the goals of this thesis. The author wishes to thank Yashil Sukhurdeep (Applied Mathematics and Statistics, Johns Hopkins University) for the helpful discussions and comments regarding elastic shape analysis.

The computations presented here were conducted at the Hyak, UW's high performance computing cluster, the STAMPEDE cluster at Texas Advanced Computing Center (TACC) at The University of Texas at Austin, and Cori computing cluster of the National Energy Research Scientific Computing Center (NERSC), a Department of Energy Office of Science User Facility.

The author acknowledges support from NSF SI2-SSE Grant: An Ecosystem of Reusable Image Analytics Pipelines, Grant No. 1739419 and AWS Cloud Services for extended funding for the purposes of performance testing.

The author acknowledges support from the NSF AST Grant: Probing the Outer Solar System: Searching Below the Noise, Grant No. 2107800.

CONTENTS

CHAPTER 1

INTRODUCTION

The first written records of names of stars date to Sumerians, approximately 2-5,000 BC and the first written records of what can be described as modern-day star catalogs were left by Babylonians and date to 2000 BC. Such catalogs usually contained a dozen asterisms, on-the-order-of or less than a hundred stars, and their primary purpose was time-telling and navigation. For a very long time, this sufficed. By ancient Greece (cca. 0-100AD), and similarly in time in Islamic and in ancient China, individual star catalogs numbering on the order of a thousand stars were created. Through the efforts of Galilei, Brahe, Bayer and Flamsteed by 1600s catalogs containing a few thousand stars were made. It was not until 1801, when Jérôme Lalande published a catalog containing 50,000 observable stars, that catalogs started becoming significantly larger. By the 1850s, Argelander, Krüger and Schönfeld, compiled Bonner Durchmusterung containing 320,000 stars. Carte du Ciel, a multi-decadal international effort to catalog observable stars up to 14th magnitude, followed soon after. By 1910 several million objects down to magnitude 11 were charted, and by its end in 1950s, 4.5 million objects were charted. In a span of 150 years, catalogs were expanded from ≈5000 stars to a 5 million - nearly 3 orders of magnitude; a rate of discovery unlikely to be repeated.

Primarily, these surveys were driven by technological innovations such as the invention of the telescope and photographic plates. Modern day sky surveys (post 1990s) follow the

same pattern. Technological innovation such as charged-coupled devices (CCDs), robotics, software automation, and access and availability to large computing resources continued to increase the number of objects in astronomical catalogs. Sloan Digital Sky Survey (York & SDSS Collaboration, 2000), Gaia (Gaia Collaboration et al., 2023) and PanSTARSS (Chambers et al., 2019) charted half a billion, a billion and more than 3 billion individual objects respectively. A modern day catalogs of objects are also much more than just catalogs of positions and are often large enough to make it impractical to work with. Brightness measurements are often provided in multiple different filters, multiple measurements of the same object are provided at different times, different parameters of multiple best-fit models are attached to each object, as well as additional information and measurements from supporting instrumentation (Ivezić et al., 2019).

1 A new generation of surveys

Today, we find ourselves, once again, faced with the prospect of increasing the number of observed objects in the catalogs by an order of magnitude within a decade. The Legacy Survey of Space and Time (LSST) conducted by the Vera C. Rubin Observatory (Rubin) using an 8 meter Simonyi Survey Telescope (SST) located at Cerro Pachon in Chile (Ivezić et al., 2019), is a survey of the southern hemisphere of the sky in a few different survey cadences (Bianco et al., 2022). The two major programs are the wide-deep-fast (WDF) survey mode and the deep drilling fields (DDF).

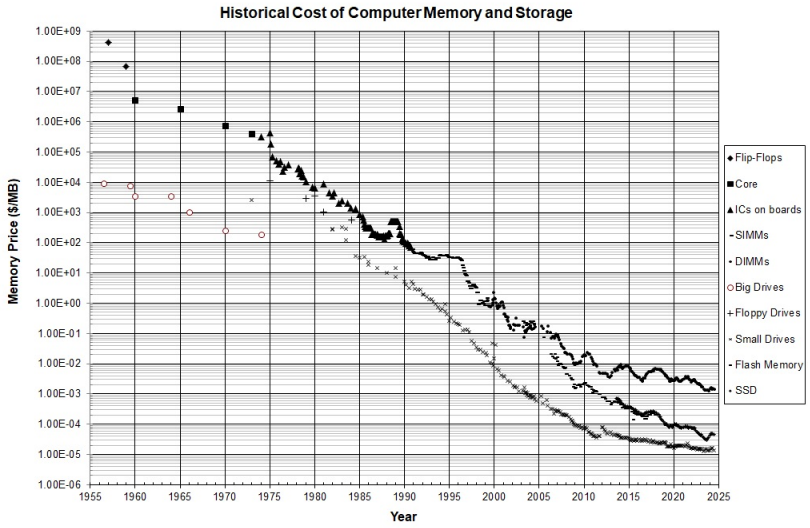
Approximately 90% of the total survey time is devoted to WFD, uniform observations of 18 000 deg^2 region in six different bands every 3 nights. During the anticipated 10 years of operations, each field is expected to be visited approximately 800 times, cumulatively across bands, and reach the co-added limiting magnitude of $m_r \approx 27.5$ mag. Majority of the remaining time is allocated to the deep drilling fields. Each DDF receives on the order of 23,000 visits, except COSMOS which receives approximately double that. The

COSMOS field has been selected to receive additional coverage in order to reach 10-year DDF depth within the first 3 years of the survey. Over the 10 years of observations the expected limiting magnitude for DDF fields is $m_r \approx 28.9$ mag. The WFD survey promises to deliver observations of 40 billion objects, split approximately equally between stars and galaxies. The DDF survey promises to deliver an observing cadence on the order of a minute, for objects within the target field, in all filters, approximately every 1.5 nights.

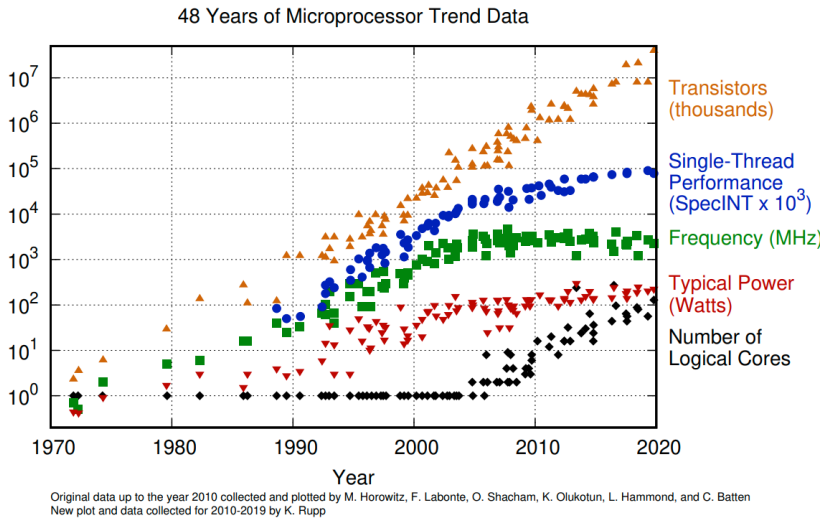
Qualitatively the data have been described as frames of a 10 year long movie of the night sky, and the data volume produced by Rubin is described as a fire hose. It is estimated approximately 20 terabytes (TB) of data a night will be produced. Over the 11 yearly data releases, the final catalog size is projected to be 15 PB in size, and the total size of the entire dataset is expected to be 500PB. Comparatively, the largest dataset released to date, PanSTARSS, is ≈60 time smaller. Purchasing and managing a cluster with sufficient data storage for a few petabytes remains within the realm of possibility for larger research institutions and universities, and often only a statistically significant subset of the data is required, for example, to test a new processing algorithm or perform measurements not available in the catalogs. At the University of Washington, the Hyak Klone Computing Cluster, a shared computing resource, accommodates approximately 3PB of data storage. Within the Department of Astronomy, a computing resource called Epyc has nearly 1PB of total storage. At the data volumes Rubin predicts it will produce, the Department of Astronomy would require the entire Klone cluster to store just 5% of Rubin data, and an entire Epyc to store just 1% of the data. Compared to 150TB of data, 10% of PanSTARSS data, it seems a relatively manageable task even though processing it successfully remains a challenge in and of itself.

While each large modern survey pushed the limits of available technology, the fast paced nature of improvement of existing, and development of new and improved, technology has been able to outpace the relatively slow nature of conducting the actual surveys themselves. What was once expensive and required a costly compute cluster can, today, be performed

on a consumer-grade workstation. Technological advancement continues to decrease the cost and increase the number of transistors, but the frequency and single-threaded performance of Central Processing Units (CPUs) have plateaued in the last decade and the cost per gigabyte of storage has not continued the same precipitous decline observable pre-2010s.



(a) Cost of storage per megabyte since 1955. Plot and data compiled by John C. McCallum (Oklobdzija, 2001). The copy with data extending to July 2024 taken from jcmit.net/disk2015.html



(b) Number of transistors in integrated circuits (ICs). Plot and data compiled by Karl Upp. The copy with data extending to 2021 taken from karlupp/microprocessor-trend-data GitHub.

Figure 1.1: The cost of storage has decreased significantly over the decades, driven by new technology and the ability to increase the density by which the data is being recorded. The increased memory density of modern, post-2000s, storage hardware often comes at a tradeoff of reliability and slower reading and writing speeds. The cost of storage continued to decline in the last decade, although not at the rate it was in early 2000s. The cost per gigabyte of storage is not as direct of a proxy for storage performance as transistor density is to performance of integrated circuits, because desired performance and reliability is often achieved as part of a larger storage solution (object storage, General Parallel File System, Redundant Array of Independent Disks Disks etc.) which incurs an overhead penalty and sometimes may require storing multiple copies of the same data. Gordon E. Moore observed in 1965 that the number of transistors in integrated circuits roughly doubles every year and this trend has continued to this day. A higher transistor count means higher core frequency can be achieved, which corresponds to the number of operations per second. However, the power consumption scales as f^3 so the the clock frequency, and subsequently the single-threaded performance, have not followed the same trend since 2000s. Performance continues to increase due to better code execution paths, branch predictions, SIMD operations, better caching strategies and purpose-specific performance accelerators (parallel computing, GPUs, FPGAs, ASIC etc).

2 Data processing at scale

In astronomy, high performance computing clusters are the de-facto standard environment for large scale data processing, whether simulations or observations. Most often, they provide the user with a shared POSIX-compliant filesystem (Standard, 2024) and a workload distribution system. Selecting a subset of the total dataset, downloading it to a local compute resource and performing novel or additional processing of the data is a common paradigm of data processing within astronomy research. However, it is becoming difficult to reconcile this paradigm with the cost of the resources required to perform such processing. As discussed in Section 1, the number and availability of clusters that can accommodate the upcoming new generation of sky surveys are low, and deploying such clusters is a high-effort and costly endeavor.

The technology (IT) industry has had to address similar issues. IT companies have sometimes found themselves outpacing their ability to procure resources to provide their offering to a fast-growing number of users. Once procured, the desire to cost-optimize the maintenance and cost of procurement of these resources led to new frameworks for highly parallelized processing (Posey et al., 2018), such as map-reduce (Dean & Ghemawat, 2008), new computer architecture paradigms, such as shared-nothing environments (Krish et al., 2013), near-data processing (Tang et al., 2019) and cloud computing (Sfiligoi et al., 2020). Today, these are key features of the approach to computing in the IT sector.

Compared to shared-everything environments, where each computing resource has the access to all of the data, in the shared nothing environment each computing node may not even be aware that it is part of a larger cluster, nor that data that is not locally available to it exists. Map-reduce is a framework for highly parallelized processing across multi-node clusters in both such environments. In the first step, data is mapped to a key by which the data is shuffled onto compute nodes. A reduce operation can then produce an aggregate result for each key, which are then "collected" and returned to the user. Near-data processing

is a technique that moves computing power closer to data storage to reduce data movement, such that it minimizes or completely removes the amount of data that has to be transferred between two nodes, removing the often large overhead incurred by moving large amounts of data. Cloud computing is an architecture paradigm in which a scalable and elastic pool of physical or virtual resources are made available to the user in a self-service rental-like basis. These resources can then be effectively "rented" from providers, configured for the use-case in question, for the duration and amount required by the user at a cost often at a fraction of that required to purchase the required hardware.

Adopting these new frameworks, instead of the current subset-download-process paradigm, could enable and accelerate scientific discovery by: a) allowing elastic procurement of both storage and compute resources, b) reducing barriers to entry for scientists, and c) lowering the price of large scale processing of both catalogs and images. In Chapter 2 we discuss how support for cloud computing services was added to Rubin data processing pipelines (Bosch et al., 2017), how data could be processed in the cloud and provide an estimate of the total cost based on a series of small scale tests and publicly available pricing.

The Rubin Science Pipelines are an interesting target because they were developed specifically for processing of Rubin datasets, but support multiple different instruments, and they implement cutting-edge algorithms for processing of both images and catalogs. However, there are shortfalls to this choice. Catalogs are carefully curated datasets, the exact content of which is determined by soliciting input from a wide array of astronomers, to meet science requirements as broadly as possible. It is possible and often desirable to compare, or supplement, the catalogs matched on the same object entries. It can be shown (Caplar et al., 2025; Zečević et al., 2019) that systems designed specifically for this purpose can perform better (easier to scale, faster, cheaper, lower barrier of entry etc.) than the less flexible more general purpose data analysis pipelines like Rubin Science Pipelines but they can not enable processing of images. Catalog analysis will be the source of most of the science produced by Rubin, but images remain the primary source of measurable information.

3 Image data processing at scale

Novel algorithms demonstrate how more information can be extracted from image data, given its particular observing cadence and quality, particularly in the task of detection of moving objects(Smotherman et al., 2021; Whidden et al., 2019). Rubin is particularly suited to the detection of moving objects because the described (see Section 1) observational cadence is primarily driven by the requirement to detect the majority of Near Earth Objects (NEOs) larger than 140 meters in diameter.

The exposure time is short to minimize the effects of trailing on the fast moving objects (apparent motion of NEOs is as high as a few degrees per day) and closely spaced with followup observations in order to enable unambiguous linking of detections. High quality photometry ensures the object's color are measured with high precision and observations in different filters occur in a short time-span to reduce color variations due to observing cadence. This approach allows accurate asteroid type classification based on their spectra, and because observations are repeated over many lunations, an object's phase curve can be precisely measured and its shape modeled.

The predominant source of NEOs are the Main Belt Asteroids (MBAs) (Binzel et al., 2015). MBAs are objects with semi-major axis between Mars and Jupiter of varying composition, but generally volatile-poor objects. A continuum has been shown to exist between the state of non-volatile and volatile-rich compositions and that surfaces of objects can form a volatility-poor crust, while the object may still contain a volatile rich core (Jewitt et al., 2024). The distinction between "asteroid" and "comet" is often not clear, but on the order of 1% of NEOs are associated Jupiter family comets (JFCs), a group of relatively short-lived, low inclination ($\leq 30^o$) comets with short orbits around the Sun caused by the gravitational pull of Jupiter (Shober et al., 2024).

The dynamical and physical lifetimes of most observed comets, and especially JFCs, are short compared to the age of the solar system (Duncan et al., 2004), so it has long been

presumed that a large and stable enough reservoir of volatile-rich objects must exist, from which objects can be scattered by some mechanism into observed cometary orbits. Objects existing past the orbit of Neptune are called trans-Neptunian Objects (TNOs). There are 2 major structures TNOs can be associated with: a) the Kuiper Belt and b) the Oort cloud.

Similarly to the main belt, the Kuiper Belt is a disk-like region in the outer Solar System, extending from 30AU to approximately 50AU. It is larger in volume and expected to be more massive than the main belt (Morbidelli et al., 2004). Some TNOs, predominantly from the Kuiper Belt since objects in the Kuiper belt are more gravitationally influenced by Neptune, evolve onto Neptune-crossing orbits. Such objects are called Centaurs and due to orbital resonances some of them reach Jupiter crossing orbits and become JFCs (Fraser et al., 2024). Classically, the Kuiper belt objects are also categorized into two populations: the scattered disk and the belt objects. The main distinction being the closeness and the time-scales of orbital interactions with Neptune, where those that experience macroscopic migration in semimajor axis in close encounters with Neptune are considered to belong to the scattered disc. Generally KBOs are objects that are far enough from the center of the Solar System that they have retained much of their volatile compounds and have not undergone as much physical processing as MBAs. The objects in the disk were, however, altered by a number of processes that have affected both the distribution of objects in the disk and the objects themselves. Analogous to the MBOs, the currently observable properties of the belt (DeMeo et al., 2015; Reddy et al., 2015; Rivkin et al., 2015) and KBOs can inform us about the formation of the giant planets, their subsequent migration (Brasser et al., 2009; Gomes et al., 2005; Morbidelli et al., 2007; Tsiganis et al., 2005) and vice-versa; the constraints on the migration of the outer planets can inform us about the migration of the inner gas giants and subsequently the formation of terrestrial planets. The study of KBOs is therefore of great interest.

Further still than the Kuiper Belt are the objects of the Oort cloud. Considered as the primary source of nearly-isotropic long period comets (LPCs), the Oort cloud is in the

semimajor axis range of 10,000-1000,000 au. The expected median inclination of objects belonging to the inner Oort cloud is 10-50 degrees (Levison & Dones, 2007), and the median eccentricity is 0.7. LPCs are thought to originate from early Solar System planetesimals that did not remain on stable orbits. The planetary perturbations increased the semimajor axis of such planetesimals, while not significantly altering their perihelion (Dones et al., 2004). If long-term smooth perturbations from local galactic matter ("galactic tide"), less smooth perturbations from passing molecular clouds (GMC), and nearby stars can lift the perihelion of the planetesimal out of the planetary zone before the planetesimals become unbound, the planetesimal will obtain an orbit that fits the above definition of the Oort cloud. The trajectories of the stars are randomly oriented, thus yielding an isotropic distribution of objects, and act to reduce the perihelion distance, thus some small percentage of Oort cloud objects enter the planetary zone again and become observable as LPCs.

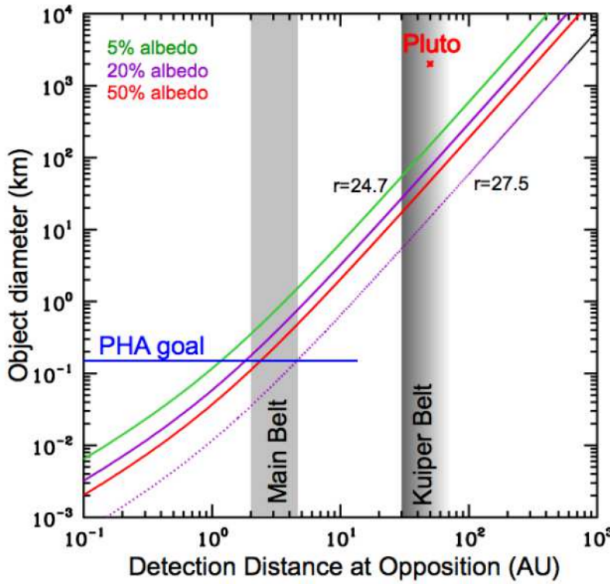


Figure 1.2: Detection limits for small Solar System objects as a function of distance. Main Belt Objects (MBOs) with diameters of 100 meters, and Kuiper Belt Objects (KBOs) with diameter of 100km, can be detected in a single exposure, depending on their albedos. Taken from Ivezić et al. (2019).

It is important to stress that the current understanding of TNOs is affected by observational biases towards large, high albedo, highly eccentric objects near their perihelion

passage. In order to answer the currently open questions on the composition of the early solar system, its formation, and dynamical evolution, it is crucially important to discover a large number of new TNOs. Rubin project it will detect 40, 000 Trans-Neptunian objects (TNOs) (Ivezić et al. (2019), see Figure 1.2), an order of magnitude increase from the ≈5,000 currently known TNOs. The limiting factor to detection of more of new TNOs is the 5 sigma confidence single exposure limiting magnitude set by the requirement for detection of large numbers of NEOs, as discussed earlier.

Rubin's solution to enabling science that requires lower limiting magnitudes is to align, resample and integrate the signal in images overlapping the same area of the sky in a process called "coaddition". As discussed in Section 1, the coaddition of individual images yields a lower single-band limiting magnitude ($m_r = 27.5$ for WFD and $m_r = 28.9$ for DDF). Coaddition, however, aligns the pixels onto the same coordinate grid on the sky. Thus, only static objects benefit from coaddition. Were an object to move between two exposures, it's signal would be added to different set of pixels in the final coadd, thus not yielding in an increase in signal-to-noise (SNR) of the object. If the motion of the object was known, the image pixel coordinates could be shifted appropriately before coaddition in such a way that the signal of the object would fall again on the same pixels. This is the core idea of shift-and-stack approaches to detecting moving objects in images.

By applying the shift-and-stack technique, whereby images containing the same moving object are offset from one another such that the moving object appears stationary in the coadded image, it is possible to detect objects below the limiting magnitude of a single image. The limiting magnitude of the coadd produced scales as $\Delta m = (5/2) \log \sqrt{N}$ (Smotherman et al., 2021), therefore, assuming the nominal Rubin observing cadence[1], three months of data would increase the limiting magnitude in the red band from $m_r = 24.7$ mag to $m_r = 26.1$ mag. Assuming a single power-law distribution of TNOs (Fraser et al., 2008), this increase in the limiting magnitude correlates to an 8-fold increase in the number of discoverable TNOs

[1] https://survey-strategy.lsst.io/

from 40,000 to 320,000 TNOs. In Chapter 3 we describe the work done in order to scale a shift-and-stack algorithm named Kernel Based Moving Object Detection (KBMOD) for the whole of the DECam Ecliptic Exploration Project (Trilling et al. (2024), DEEP) dataset. To demonstrate the ability to integrate with the Rubin Science Pipelines the entire DEEP dataset was reprocessed, producing a 200TB large data repository, from which approximately 20TB, a full Rubin night, of data was searched.

The proposed Rubin cadence allows for several KBMOD search strategies. Search strategy for moving objects on a monthly basis for the pointings with the most number of visits, and search strategy for moving objects within the DDF fields that are expected to have many inter-night pointings. An extension to the original KBMOD algorithm is described that makes the first search strategy possible. By correcting the reflex motion of the Earth, we show how new groups of images that overlap a region at a given distance within the Solar System can be identified and how the apparent motion of the object on the sky can be linearized over longer periods of time. However, this search strategy comes at a cost of extending the dimensionality of the search. Both search strategies involve searching data volumes approximately equal to or greater than the searched DEEP dataset.

4 Transient object classification

A key algorithm within the Rubin Science Pipelines is the image differencing analysis. Once overlapping images are coadded, a template image can be produced from the coadd that represents the observed area at some shorter exposure time. The template can then be subtracted from each individual exposure and source detection and measurement performed. The image difference analysis aims to capture objects that have changed their flux or position on the images, and issue their measurements in an alert stream sent to astronomers within 60 seconds of an observation (Bosch et al., 2018). The purpose of the alert stream is to enable fast optical and spectroscopic followup of such events in order to facilitate their

classification. Many different physical processes shape the apparent brightness of an object and how it changes over time. The measured brightness of object over time can vary at very different timescales. Wide area, dense temporal sampling is therefore the best strategy for discovery and analysis of both rare (neutron stars, black hole binaries, novae, flares, gamma-ray bursts, AGNs, gravitational lensing etc.) and variable sources (Cepheids, RR Lyrae, Delta Scuti, RV Tauri, recurrent novae, eclipsing binaries, exoplanet transits etc.).

Cosmological analyses (Sánchez et al., 2022), for example, focus on using Type Ia supernovae (SNia) events as standard candles to measure the expansion history of the universe, and in particular the dark energy equation of state ω. But such analyses are sensitive to contamination by misclassified events within the dataset. While the combination of optical and spectroscopic measurements enable the most accurate classification of events, only a small fraction of objects can be spectroscopically observed (Boone, 2019; Hložek et al., 2020), therefore accurate photometric classifications of lightcurves are needed. Supernovae events reach peak luminosity within a couple of days and decay on the order of weeks. With respect to SN lightcurves, Rubin's WFD cadence can provide only a sparsely sampled lightcurve. The samples occur in irregular intervals, in different bands on any given night and with heteroskedastic uncertainties. All these factors make correct classification of objects difficult.

The problem is made more difficult by the fact that, due to Rubin's limiting magnitude and area of coverage, on the order of 10^6 transient detections per night are expected so performance may have to be considered and the dataset is an unevenly weighted set of lightcurve measurements of a wide range of heterogeneous types of objects. Any classifier applied to the Rubin data has to be cognizant of this heterogeneity and not misclassify such lightcurves into one of the predominant groups.

The Supernova Photometric Classification Challenge (SNPCC, Kessler et al. (2010)) and Photometric LSST Astronomical Time-Series Classification Challenge (PLAsTiCC, Hložek et al. (2020); Kessler et al. (2019)) are two major community efforts to simulate survey-

realistic lightcurves with the purpose of testing classifier performance. Typically the best performing classifiers in both challenges focused on training a classifier on a labeled set of lightcurves that had spectroscopic confirmation of their type. However, this biases the classifier towards closer and brighter objects, for which high quality spectra could be obtained. As a consequence, if the labeled training data was not representative of the larger, lower SNR, test dataset, the classifiers performed worse. Data augmentation techniques proved critical for performance of classifiers in the PLAsTiCC challenge, as did the correct identification of classification features and training on a larger set of features but performing classification on a smaller subset of features.

In the last chapter 4, we investigated a novel approach to classification of lightcurves. We focus on the subfield of shape and elastic shape analysis and attempt to produce an unsupervised lightcurve classification algorithm based on the measured geodesic distance between curves. The approach successfully compares the intrinsic "shapes" of curves by measuring the "energy" it would require to, under certain constraints, warp, i.e. elastically deform, one lightcurve to match another. By comparing the amount of energy required to deform a lightcurve, a naturally interpretable classification can be constructed based on the idea that lightcurves that require low mutual deformation energy belong to the same class of objects. We formulate unsupervised classification schemes based on this idea and perform simplistic tests using bespoke simulated dataset and a subset of PLasTiCC test and training datasets. While the approach did not yield the precision or accuracy of existing classification approaches on real data, and struggled to deal with irregularly sampled data with uncertainties, we show that in ideal conditions it can produce accurate classifications. We briefly discuss the shortfalls of our attempts to address these issues and review how, by borrowing data augmentation approaches from PLAsTiCC challenge solutions and expanding the classification scheme to include additional information available, we could improve on our current results.

CHAPTER 2

VERA C. RUBIN OBSERVATORY SCIENCE PIPELINES ON AWS

1 Introduction

Aside 1.1: Historical perspective

Significant portions of the chapter refer to the state of the system as of 2020 and employ nomenclature, or otherwise refer to issues which may no longer exist. Cost estimates are also based on at-the-time current costs of AWS resources, some of which have since been deprecated or changed. Much of the described work was executed on w_2019_38 (week 38, 2019) Rubin Science Pipelines build, Pegasus versions 4.9−4.9.2 and HTCondor version 19.X.Y. Described work used the development branches of the Generation 2 Data Butler, before it was considered ready for use in the main code library and indeed before a PostgreSQL database was even implemented within the Generation 2 Data Butler itself. It was a part of this work that the first implementation PostgreSQL-backed Registry and user authentication, now the default and main choice of database technology in the Rubin stack. Pegasus >= 5.0 marked the first release of a ground-up design and implementation of a Python API. HTCondor Annex transitioned to resource and deployment management via the AWS *CloudFormation* service. Significant efforts have been put forth, both by the Rubin SQUARE and Middleware teams, to improve on this initial work. The LSST Batch Production Service (BPS) execution framework was designed and implemented, a local-to-task-butler and local-to-task repository and "butler zipping" were implemented, making many of the core issues of the described approach functionally nonexistent. The Rubin Interim Data Facility (IDF) was deployed on Google Cloud services in c. 2020, but the efforts to deploy on AWS remain important as the same codepath within the Middleware component itself is utilized. With the exception of task distribution and execution which became a Middleware component with the implementation of CTRL BPS, used services are mostly external to the Science Pipelines and Middleware components.

The Legacy Survey of Space and Time (Ivezić et al., 2019), operated by the Vera C. Rubin Observatory, is a 10-year astronomical survey due to start operations in 2025 that will image half the sky every three nights. LSST will produce ∼20TB of raw data per night which will be calibrated and analyzed in almost real time. Given the volume of LSST data, the traditional subset-download-process paradigm of data reprocessing faces significant challenges. We describe here, the first steps towards a gateway for astronomical science that would enable astronomers to analyze images and catalogs at scale. In this first step we focus on executing the Rubin LSST Science Pipelines, a collection of image and catalog processing algorithms, on Amazon Web Services (AWS). In this chapter I describe our initial work on

the performance, scalability and cost of deploying such a system in the cloud.

The currently pervasive model of sub-selecting, transferring to local compute resources and then reprocessing data has been successful for users processing past astronomical sky-survey data because technological developments and pricing made acquiring sufficient local compute resource affordable. With a new generation of sky surveys such as those operated by the Rubin Observatory delivering an order of magnitude more data this subset-download-process paradigm is no longer viable for users of these datasets. We describe a different approach, utilizing the elastic capabilities of cloud compute resources to enable users to scale their analyses to 100TB+ data sets.

Our goal is to provide astronomers with an interface and tools that allow them to reprocess and reanalyze an entire night's worth of Rubin data in hours, and to do so at a reasonable cost. As a first step towards such a gateway, we focus on image data reduction pipelines. These pipelines consist of a series of steps that remove any instrumental signature from the data, detect sources above a specified threshold, potentially cross reference these detections to previously known sources, and measure their properties (or features). A typical input and output of such a pipeline is shown on Figure 2.1 where a raw image has been processed to remove instrumental effects.

By design the LSST data reduction pipelines can support many different instruments. Some of the instruments that are currently supported, either directly or via community contributed packages, are Dark Energy Camera (DECam), Hyper-Suprime Cam (HSC), SDSS and Canada-France-Hawaii Telescope (CFHT). Exposing the Rubin LSST Science Pipelines functionality through a common interface would enable the astronomy community to define and execute custom analysis pipelines based on state-of-the-art astronomical data processing algorithms.

The need for scalability and the ability to share the resulting data across a range of communities naturally leads to a cloud computing model, whether commercial or academic. Focusing on cloud computing allows us to scale not only the compute resources required but

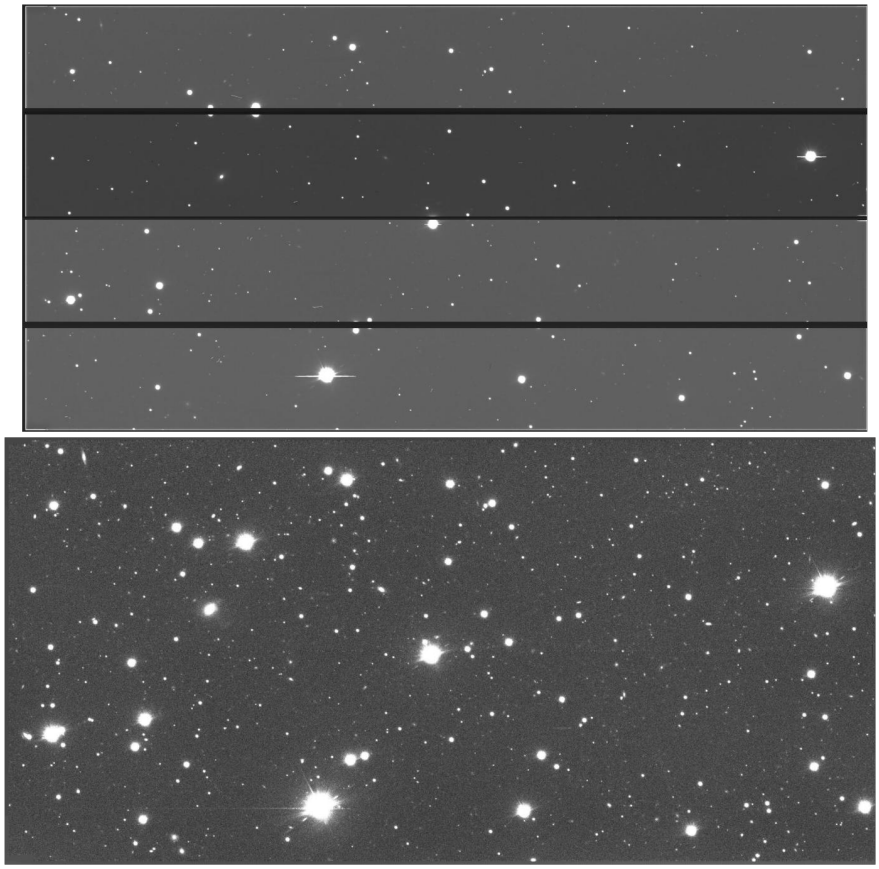


Figure 2.1: A raw, uncalibrated image from the Hyper Suprime Cam (HSC) (Aihara et al., 2018) (top) and the resulting calibrated exposure after processing with the Rubin LSST Science Pipelines (bottom). The images are not the same size because calibration adjusts for the optical distortion of the instrument.

their pay-as-you-go philosophy allows for cost optimization. The global nature of the cloud also facilitates faster and easier data sharing and data access between different scientific communities.

2 Technology Stack

While cloud technology has been embraced quickly by some scientific communities (see Posey et al. (2018) or Holzman et al. (2017)) its adoption by the astronomical community has been slower. Legacy code is typically designed with the assumptions that (a) data exists locally or there exists a globally accessible file system, (b) the computation is some form of batch

processing, and (c) the system is in general not state agnostic. While it is possible to create similar systems in the cloud, modern cloud approaches scale much better with shared-nothing filesystems (Stonebraker, 1986), containerization (Bernstein, 2014), and near-data processing architectures.

To achieve our goals we have to resort to new additions to the legacy codes and tools and to add functionality where it is missing. These tools and code changes are described in more details in sections 2.1 and 2.2, while section 2.3 describes the used cloud services. Here we will first provide a brief overview of Rubin LSST Science Pipelines and how they are used to process data and then we will briefly describe the modifications required to run the system in the cloud.

To facilitate running Rubin LSST Science Pipelines in the cloud we adopted a lift-and-shift strategy that is common in porting existing applications to the cloud. Initially, an exact copy of Rubin software was executed on cloud services configured to mimic the on-premise environment. Having verified functionality of the software we implemented Simple Storage Service (S3) back-end for the Datastore and a PostgreSQL backed Registry managed through the Relational Database Service (RDS) service. Because the Data Butler abstracts IO for the Middleware, the changes remove the need to have a shared local drive across the nodes.

Astronomical data reduction pipeline consists of many individual steps. Steps, generally speaking, need to be ordered as subsequent ones depend on data products of the previous one and each step can require significantly different resources than the previous steps. To describe this inter-dependency of steps, as well as their resource requirements, we use a workflow manager. The workflow manager of choice in this work was Pegasus (Deelman et al., 2015) due to the excellent user support and flexibility.

Finally, the last missing ingredient is actual resources. We need to be able to procure resources of desired specifications, then transfer and execute jobs on these resources and then deallocate the acquired resources. The resource manager of choice in this work was HTCondor (Thain et al., 2005) due to seamless integration with Pegasus, ability to procure

and manage resources in the cloud via HTCondor Annex. HTCondor Annex allows HTCondor deployment on cloud resources via acquisition of cloud compute resources external to an existing HTCondor pool by acquiring Elastic Compute 2 (EC2) instances. The schematic of the system is shown on Figure 2.2.

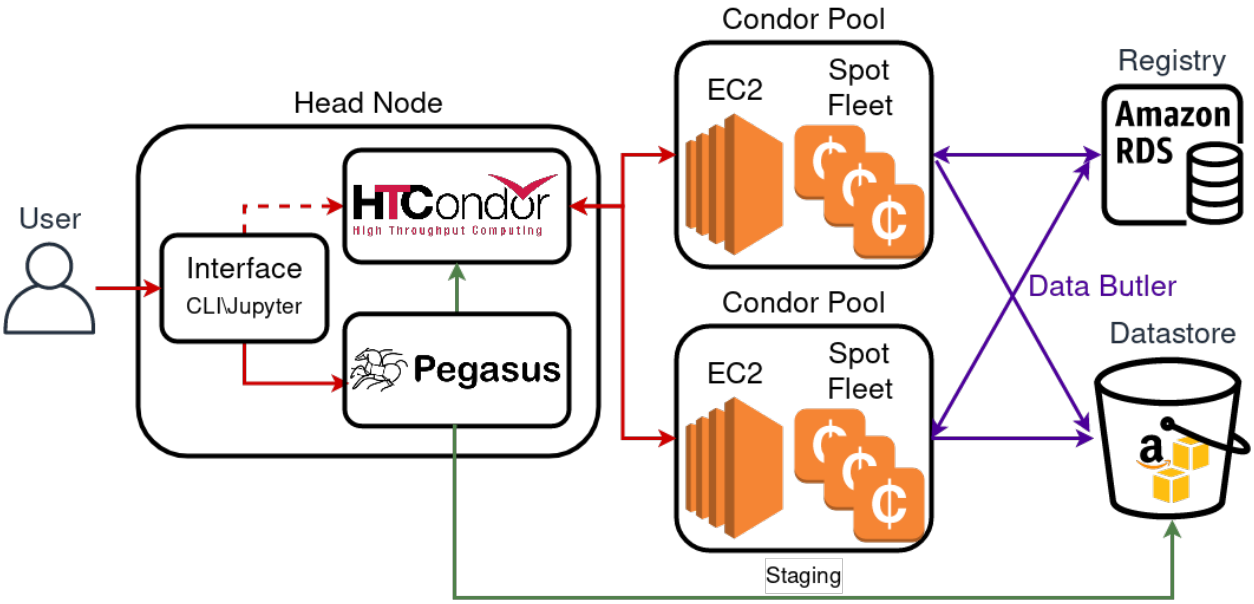


Figure 2.2: A user or users, leftmost, gains access to the head node which contains the Rubin LSST Science Pipelines, Pegasus and HTCondor. The user creates and submits a Quantum Graph to Pegasus. Pegasus clusters the jobs and begins submitting jobs to HTCondor. Users procure compute resources, workers, through HTCondor Annex. Compute resources are logically separated into units called Condor Pools, each of which is dynamically scalable and can consist of a mix of EC2 instances procured on-demand or via Spot Fleet requests. As Quanta are executed on the workers the persisted data is written to the Datastore, an S3 Bucket, and to the Registry, an RDS PostgreSQL database. Persistent data remains accessible from the head node.

Scaling is achieved by scheduling as many parallel jobs of the same type as resources allow. When the compute cluster is under-subscribed due to a low number of jobs to schedule simultaneously, the resource manager will deallocate the idle resources for cost optimization. New resources can also be allocated and dynamically added to the cluster when resource requirements of the jobs change. The system, and work required to create it, are described in more detail in Lim et al. (2019) and Butler et al. (2020). In the following chapters we focus on analyzing and identifying bottlenecks and mitigation strategies that lead to a decrease in

cost and an increase in performance.

These actions are undertaken from a persistent "master" node where we users could, eventually, interact with the components of our system through a Jupyter Notebook-like interface (see figure 2.3). While it is possible to do so currently, the integration of the various tools and packages with Jupyter is still in its infancy and most of the interaction with the cluster is performed through a terminal via ssh connection.

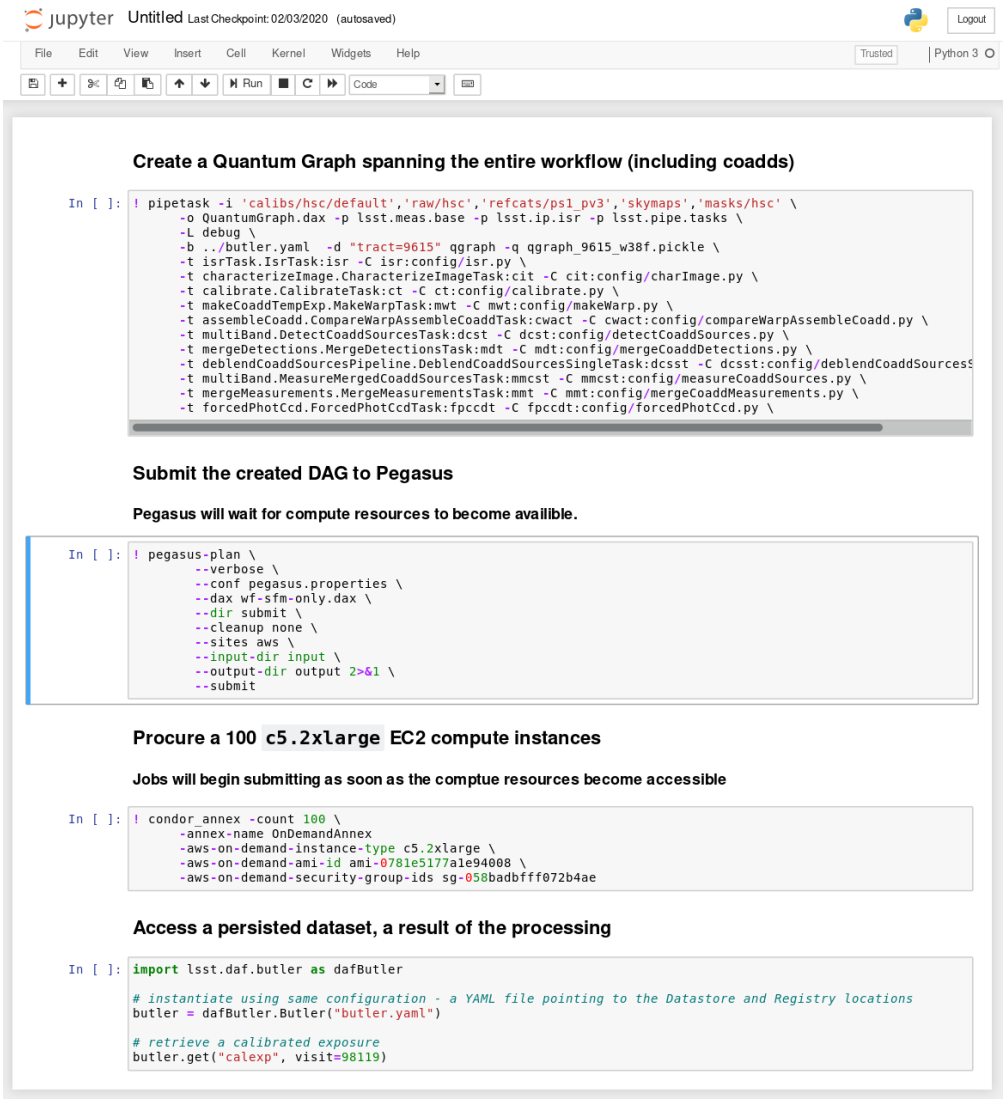


Figure 2.3: Executing a large scale processing job and fetching an image from a Jupyter Notebook.

2.1 Science Pipelines and Middleware

The Rubin LSST Science Pipelines (Bosch et al., 2018) represent the state-of-the-art in astronomical data reduction. They consist of configurable Tasks that can be chained into a pipeline. Such a pipeline is described by a directed acyclic graph (DAG) called a Quantum Graph (Dubois-Felsmann, 2017). A Quantum Graph consists of Quanta, where each Quantum is a Task applied to an individual dataset. The Rubin LSST Science Pipelines enable processing of astronomical data from a single exposure to overarching tasks such as joint calibration that constrains astrometric and photometric measurements across multiple exposures.

Tasks themselves are agnostic to file formats and locations of the data. The input-output (IO) and provenance is tracked through a Middleware component called the Data Butler (Jenness et al., 2018). The main purpose of the Data Butler is to isolate the end user from file organization, filetypes and related file access mechanisms by exposing datasets as, mostly, Python objects. Datasets are referred to by their unique IDs, or a set of identifying references. The Data Butler uses a Registry to resolve the dataset references and resolves the location, file format, appropriate IO operation and the Python object/type of the files stored in a Datastore.

The Registry is backed by an SQL database and the Datastore is usually backed by a shared filesystem. Significant amount of work in this paper was the implementation of a Registry and a Datastore capable of utilizing AWS resources and is described in detail in Lim et al. (2019) and Chiang et al. (2020).

2.2 Pegasus WMS and HTCondor

HTCondor (Thain et al., 2005) provides distributed job parallelization and is a powerful batch system for high throughput computing (HTC). Directed Acyclic Graph Manager (DAGMan) is HTCondor's metascheduler capable of managing workflows at a higher level than the

underlying HTCondor Scheduler. . The core functionality of DAGMan is to take a workflow DAG, unroll the job inter-dependencies, and match executable jobs to approrpaite resources. A HTCondor pool is a collection of compute resources. Jobs are matched to their resources based on the job resource requirements.

HTCondor Annex allows HTCondor deployment on cloud resources via acquisition of cloud compute resources external to an existing HTCondor pool. A pool can have multiple annexes, and each annex manages its own lifecycle. Unused compute resources are automatically deallocated after a user-set time spent idling.

Pegasus (Deelman et al., 2015) is a workflow management system built on top of HTCondor. It provides command line and API interfaces for scientists to write an abstract workflow independent of the underlying computing infrastructure. It provides command line and API interfaces that allow writing abstract workflows (DAXs) that are then translated to an executable workflow interpretable by DAGMan. The Rubin LSST Science Pipelines Quantum Graph is written in the form of a DAX, that is submitted to Pegasus, which translates it into an executable workflow and submits it to HTCondor. Additionally, Pegasus comes with various tools for data management, for example for log transfers, and supports different execution strategies by grouping jobs within or across nodes of a DAG.

2.3 Amazon Web Services

Datastore implements Simple Storage Service (S3) as a backend. S3 is an object storage that allows massive amounts of unstructured data, where each object typically is identified by a globally unique identifier, to be stored and accessed in a durable and highly scalable way. Boto3 is the Amazon Web Services (AWS) Software Development Kit (SDK) for Python. While the library itself supports interacting with many AWS services such as RDS, EC2 etc. in the Middleware component it's solely used to interact with the S3 service (e.g. by uploading/downloading objects in whole or in part). This ensures the Middleware component does not become tied solely into a single provider since the S3 has become the de-facto

standard for object storage API access. Most cloud providers offer storage solutions already compliant with S3 API, or offer interoperability services.

Relational Database Service (RDS) is the AWS cloud service that launches and configures databases with ease. PostgreSQL was chosen as the DBMS backend for the Registry. Primary drivers behind PostgreSQL were the ease of deploying in RDS, the common use of PostgreSQL as a Virtual Observatory (VO) Table Access Protocol (TAP) backend, no additional licensing fees usually associated with proprietary software and PostgreSQL's off-the-shelf support for spatial primitives needed for astronomy. The RDS databases can be backed up into snapshots as well as exported to downloadable files on S3.

Elastic Compute Cloud (EC2) is used to acquire compute resources. EC2 service uses virtualization technology to deliver a variety of different pre-configured instance types optimized for different use cases. Within each instance type there are several different instance sizes comprised of varying combinations of CPU, memory, storage, and networking capacity. Amazon Machine Image (AMI) provides the information required to launch an instance and we provide pre-built CentOS AMIs that contain and configure the required parts of the technology stack for both master and the workers. EC2 differentiates between "on-demand" and "spot" instances. On demand instances are allocated to users until they are de-allocated by the user, while spot instances are allocated to the user but can de-allocated at any time with a 2 minute warning. Spot instances prices change based on the demand but are on average much cheaper than on-demand instances.

3 Example Workflow

The example workflow is a subset of the well-understood and well-characterized HSC Release Candidate dataset[1]. This dataset is reprocessed using Rubin Obs. compute resources every two weeks in order to characterize the scientific validity and performance of the algorithms as they are being developed. The workflow consists of initialization, instrument signature

[1] https://jira.lsstcorp.org/browse/DM-11345

removal, characterization and calibration jobs (see Figure 2.4). Initialization job is responsible for registering metadata in the Registry. There are 6787 instrument signature removal (ISR) jobs. ISR removes image features that are the results of instrument flaws. Characterization consists of modeling the background, point spread function (PSF), repairing cosmic ray traces, detecting and measuring bright sources and measuring aperture correction. There are 6787 characterize tasks in the workflow. Finally, the calibration tasks detect all sources on the image and perform astrometry and photometry on the image.

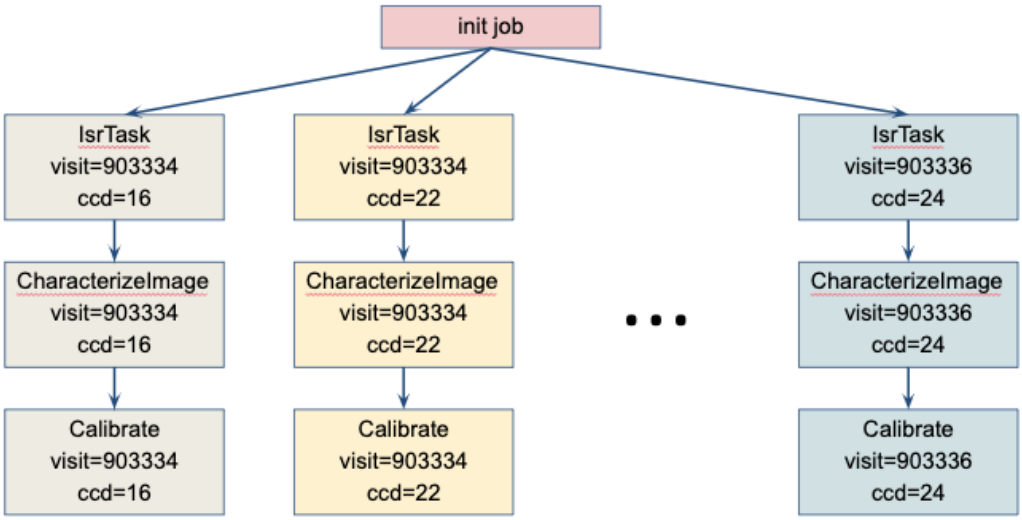


Figure 2.4: Example workflow on which performance and scalability was tested.

The total size of the input data is 0.2TB and the size of output data is approximately 2.7TB. In number of exposures processed, this represents ∼11% of the number of observations per night from Rubin Observatory. Because each LSST image will be approximately twice the size of a single HSC image, in terms of data volume the example workflow represents approximately 2% of a night. The scaling between processed data volume and number of visits is a function of observed part of the sky, regions with many objects will on average take longer to process, and the scheduling overheads, which themselves are a function of clustering and other execution optimization parameters. Figure 2.1 shows the comparison between the raw input image and the output data, a calibrated exposure.

4 Performance, scalability and cost

The example workflow was tested using different instance types and instance sizes. Whenever possible multiple tests for each instance type and size were executed. The tests were performed on on-demand and on spot instance types. On-Demand compute capacity is charged by the second, with no long-term commitments, and full control over the instance's life-cycle is retained by the user. Spot instance availability depends on the current availability of unused EC2 resources but no control over the life-cycle of the acquired instance are retained. They can, however, be purchased on the open market of instances at a significant discount compared to on-demand instances. The wall time is the cumulative value of execution times of all Tasks in the Quantum Graph, the scheduling and execution overhead. In total approximately 60 different runs were executed and a de-duplicated and cleaned-up sample of results can be seen in table 2.2.

Initial testing was performed with 25, 50 and 100 `c5.xlarge` instances. Each `c5.xlarge` instance has 4 vCPUs. Each vCPU is a hardware hyperthread on a 3.0 GHz Intel Xeon Platinum 8000-series processor, 8 GB of memory, up to 10 Gbps connection bandwidth and mounts an General Purpose (`gp2`) Elastic Block Storage (EBS) SSD volume. The two major bottlenecks in the processing were an EBS drive IO bottleneck and the staging of the scripts and configuration files.

A typical `gp2` EBS drive throughput is between 128 and 250 MBps depending on volume size and instance type. This is usually parameterized through IOPS (IO operations per second) with larger volumes having a higher base IOPS performance, f.e. a 100 GB drive's base performance is 300 IOPS. A volume can burst up to 3000 IOPS. The duration of the burst is governed by the amount of Burst Credits associated with the drive. Burst Credits recharge when IOPS usage is less than baseline performance. When the Data Butler can not instantiate objects from memory, it will download it to the local drive thus exhausting the Burst Credits on individual workers.

Staging is the process of setting up the required environment including the transfer of job scripts, their configuration files and other required files. Rubin Science Pipelines, Pegasus and HTCondor offer comprehensive and very detailed control over scheduling and execution of jobs. However, as discussed in Section 2, these tools were designed and optimized primarily with a more classical HPC computing architectures in mind. The Rubin Science pipeline export Quanta as pickle files, and the default behavior of Pegasus and HTCondor is to pickle any object that doesn't support serialization via pickling. The cumulative size of the quanta and processing configuration files, however, can be up to 5 GB and nearly 6 GB for output processing logs. This strains the IO performance on the head node as many of the configuration and output logs transfers occur in lock-step. During testing, Burst Credits often masked or delayed the onset of IO bottle-necking. For example, timings for 25 and 50 workers were the same (see Figure-2.5 unoptimized) because the Burst Credits were already exhausted before the first run, so both workflows were limited by the drive throughput, while the third one, launched the following day, executed against full Burst Credits ensuring it completes in a significantly shorter time frame than the same job a day earlier. The lack of direct immediate feedback made identifying the issue difficult.

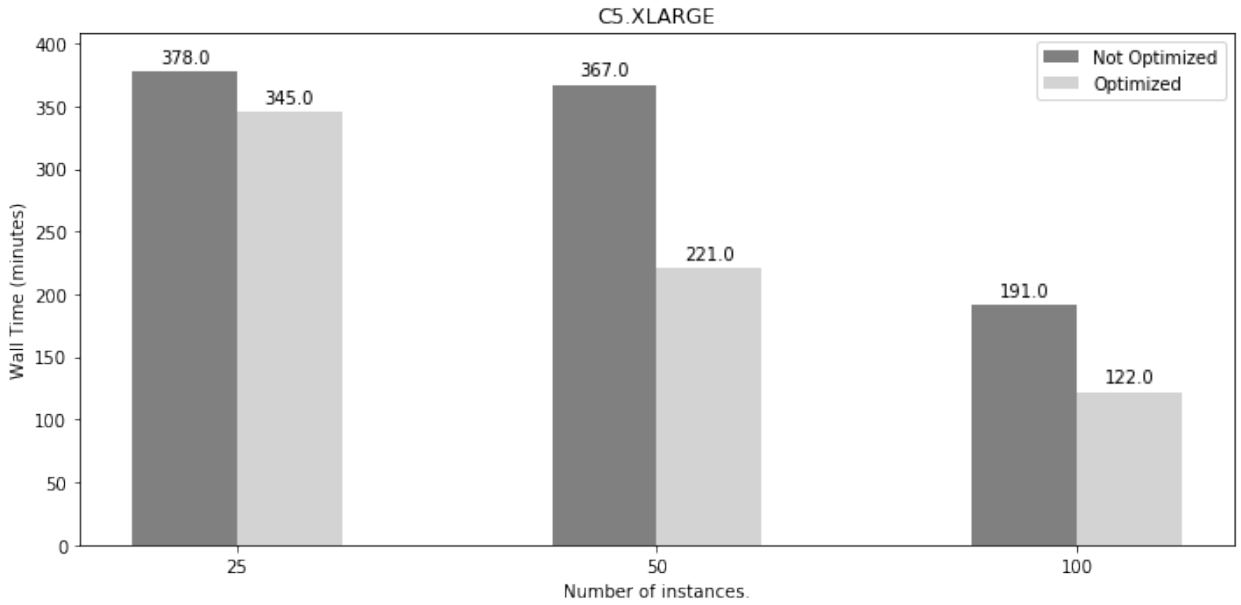


Figure 2.5: Non-optimized vs optimized workflow execution timings.

At the start of the execution of the pipelines a singular "initialization task" is executed in which various inter-dependencies and metadata about tasks, data and data products are retrieved and written to the Registry. The initialization job takes on average 22-30 minutes. This time should not be considered when scaling the execution times to larger datasets because the initialization job is required to run only once per each new output collection and thus would potentially not be run again for any subsequent processing steps, nor would it be run multiple times if the DAG itself contained more Tasks.

PostgreSQL registry itself was also poorly optimized for the kind of workflows that were executing. Partly some issues stemmed from the fact that the majority of use-cases for PostgreSQL database on RDS were to serve real-time web-sites and services. Thus adjusting PostgreSQL server parameters, such as maximal allowed number of connections, cache size per connection etc. and paying close attention IO limitations of EBS drives mounted on the instances hosting the database (often hidden from plain sight by burst credits), improved the workflow execution time. The largest optimization gains were, however, made when analysis of Registry query patterns revealed the access patterns and current Middleware's DB schema were not optimal for execution on PostgreSQL. Namely, the most common query pattern within Middleware is to create a large view of many smaller tables and then subselect from the view. A database view is a virtual table that presents a subset of a database in a structured format and their use generally assists with expressing complicated logic involving joins of many different larger or smaller tables. The Query Planner of the database in question attempts to resolve and optimize the lookups on virtual tables such that it can materialize as few of it into memory as it can. The process, however, can be sensitive to the exact query and definition of the view itself. Due to the complex schema of the Registry queries often resulted, or even required that, the entire view is materialized.

The code excerpt 2.1 shows one such problematic query, where total execution time to retrieve 4 results is 12 seconds. A temporary fix was put in place by materializing the views within the database, and indexing their columns. The measured execution time for the same

query was then just 0.021 ms; an improvement of 571 428 times. A trigger was set on the views to regenerate after new datasets were inserted into the tables they were created from.

Resolving many of these issues, moving Pegasus staging into S3, better job clustering, better indexing schemes, materialized views, larger cache values and more appropriately scaled EBS drives achieved performance increase and a cost decrease of 40 to 60 percent. Further improvements were achieved by clustering multiple single jobs into a single larger job and by relegating scheduling to DAGMan. This was because Pegasus would, by default, fork a new `condor_submit` process for every job. If the workflow was, for example, executed on 50 `c5.2xlarge` workers it would launch 400 jobs, which fork 400 processes and establish 400 head-worker staging transfer connections, simultaneously.

```
01 | Limit  (cost=1241332.85..1241332.96 rows=4 width=33) (actual time
   =12732.766..12732.774 rows=4 loops=1)
02 |   ->  Unique  (cost=1241332.85..1335844.05 rows=5400640 width=33) (
   actual time=12732.765..12732.770 rows=4 loops=1)
03 |         ->  Sort  (cost=1241332.85..1254834.45 rows=5400640 width=33) (
   actual time=12732.764..12732.767 rows=6 loops=1)
04 |               Sort Key: visit_detector_skypix_join.instrument,
   visit_detector_skypix_join.visit, visit_detector_skypix_join.detector,
    patch_skypix_join.skymap, patch_skypix_join.tract, patch_skypix_join.
   patch
05 |               Sort Method: external merge  Disk: 209008kB
06 |               ->  Hash Join  (cost=119574.91..342063.41 rows=5400640
   width=33) (actual time=763.846..1524.504 rows=4642107 loops=1)
07 |                     Hash Cond: (visit_detector_skypix_join.skypix =
   patch_skypix_join.skypix)
08 |                     ->  Seq Scan on visit_detector_skypix_join  (cost
   =0.00..3188.60 rows=206960 width=16) (actual time=0.008..16.289 rows
   =206960 loops=1)
09 |                     ->  Hash  (cost=56556.07..56556.07 rows=3259107
   width=25) (actual time=763.723..763.724 rows=3259107 loops=1)
10 |                           Buckets: 65536  Batches: 64  Memory Usage:
   3635kB
11 |                           ->  Seq Scan on patch_skypix_join  (cost
   =0.00..56556.07 rows=3259107 width=25) (actual time=0.006..240.128
   rows=3259107 loops=1)
12 | Planning time: 0.323 ms
13 | Execution time: 12759.391 ms
```

Listing 2.1: Output of `EXPLAIN ANALYSE SELECT * from visit_detector_patch_join LIMIT 4;` showing poor initial performance of the database. Executing the given query for just 4 results required 12 seconds, which was reduced to 0.012ms after performance improvements.

Run name	Instance count	Cost	Workflow walltime	Cumulative Walltime	Comments
			m5.xlarge		
20200116T213148	25	24.34	5h39m	18d20h	
20200115T172359	50	42.15	3h50m	18d23h	
20200116T175704	100	42.97	3h18m	18d5h	
			m5.2xlarge		
20200118T230757	25	29.44	3h18m	18d22h	
20200117T211929	50	70.86	22h27m	16d3h	Workers deallocated themselves, hung overnight.
20200118T202124	50	45.72	2h36m	18d6h	
20200117T173248	100	104.89	3h39m	16d7h	
			t3.large		
20200122T170649	25	23.21	10h23m	18d20h	
20200121T201329	50	21.09	6h10m	18d16h	Didn't refresh pool so timing is off. Pegasus got stuck on the last job. Pricing is only approximate.
20200121T163329	100	23.70	3h35m	18d18h	
			c5.xlarge		
20200123T165014	25	23.22	6h18m	14d17h	Walltime overestimate; jobs finished but annex was still alive. Accidentally restarted the main node during execution.
20200126T175516	50	25.81	6h7m	15d2h	
20200123T232152	100	44.86	3h11m	14d20h	Fastest init job (≈10 mintues) compared to ≈20 for others.
			c5.2xlarge		
20200127T010224	25	27.33	3h8m	16d9h	
20200127T162240	50	50.45	3h16m	16d4h	
20200128T192127	50	67.48	3h38m	15d20h	
20200128T234142	50	54.62	3h58m	15d16h	
20200129T034251	50	68.59	4h14m	15d0h	Cleared S3 before running.
20200129T184318	50	51.34	3h22m	16d1h	Turned off and hung whole night.
20200129T221952	50	58.47	3h24m	15d23m	Main node restart before starting.
20200130T021620	50	69.28	4h14m	15d11h	Didn't turn it off again

Table 2.1: Execution times and cost of unoptimized runs over different instance types and count. The variance in cumulative walltime expresses the varying time required to process the workflow, induced by various latencies and varying memory speeds and CPU processing power associated with different instance type. Optimal choice of number of instances and instance type is one that minimizes cost and the workflow execution time as measured at the main node; usually achieved by a choice of a instance count large enough to be completely saturated by jobs and the choice of an instance type powerful enough to minimize cumulative wall time.

Run name	Instance count	Cost	Workflow walltime	Cumulative Walltime	Comments
			m5.xlarge		
20200309T135739	25	24.72	5h35m	18d20h	Slightly overpriced.
20200916T152931	25	23.89	6h13m	18d4h	Slow execution (30 clusters with 100 staging clusters), AWS IAM service interruptions related. No issues otherwise.
20200917T163143	25	23.74	5h27m	18d12h	34 Clusters. Staging 10 clusters.
20200917T220539	25	23.30	5h20m	18d10h	Same setup as 20200916T152931
20200309T194101	50	25.76	3h8m	18d11h	-
20200918T160432	50	24.92	3h3m	18d3h	8 stage out jobs failed. Nothing concerning time estimates (17 clusters. 100 stageout)
20200918T192012	50	25.03	2h52m	18d6h	Longer idle timeout and 200 stageout jobs to prevent jobs failing. Still 4 transfer jobs failed.
20200309T225436	100	19.40	2h39m	18d16h	I forgot to allocate the compute till 32m43s after the jobs got scheduled.
20200911T165143	100	27.84	1h46m	18d12h	Cluster size 8. Stageout clusters 300. 1 failed job, not concerning.
20200911T184524	100	27.52	2h2m	18d7h	Forgot to add 99 instances until 31:13 after start
20200911T225652	100	27.86	1h53m	18d7h	-
			c5.xlarge		
20200915T161213	25	20.84	5h34m	14d0h	Cluster size 30. staging 100. Equivalent to 0127T010224, 0914T145829 and 0914T225022
20200915T220158	25	20.75	5h51m	14d7h	Same as 0915T161213, but failed to allocate resources for 46m49s by accident
20200305T182023	25	23.38	5h45m	14d15h	Potential overestimated pricing.
20200305T182023	50	27.43	3h41m	14d21h	Untrustworthy cost. I forgot about the run.
20200910T154845	50	22.99	3h3m	14d14h	Cluster 15. Stageout clusters 200
20200910T190023	50	22.62	3h10m	14d4h	Cluster 20. Stageout clusters 200
20200910T221726	50	23.63	3h7m	14d9h	Cluster 15. Stageout clusters 200
20200306T062555	100	27.36	2h2m	15d1h	-
20200909T222446	100	25.78	2h2m	14d11h	Cluster size 15. 1 fail - no real probs
20200910T012908	100	26.19	1h58m	14d10h	Cluster size 15. 1 fail - no real probs. 200 staging clusters
			c5n.xlarge		
20200922T164119	25	24.79	5h15m	16d20h	Cluster 34. Stageout 100. Forgot to allocate resources till 12 minutes in.
20200923T171259	50	26.87	2h46m	17d4h	Cluster 17. Stageout 200
20200923T201448	100	30.99	2h9m	17d8h	Cluster 17. Stageout 200
			c5.2xlarge		
20200302T222936	25	26.19	4h2m	16d4h	Cluster size 30, for 233 jobs. Nothing is optimal because I forgot about the job and wasn't paying attention.
20200914T145829	25	26.06	3h46m	15d19h	Cluster size 30
20200914T190252	25	24.52	3h40m	13d6h	Cluster 30. staging 200.
20200914T225022	25	25.18	3h53m	15d11h	Cluster 34. staging 100.
20200227T232007	50	26.72	1h54m	16d11h	Not price-optimal but should be comparative in timings.
20200228T014417	50	25.78	1h48m	16d16h	Should be close to optimal, not quote enough jobs. Pegasus cluster size 20 (test size 15 next)
20200909T202341	50	25.81	1h50m	15d20h	6 failed stageout jobs. Stuck at 96% complete, thus ergo the much shorter cumwall time. Cluster size 15.
20200302T180554	100	47.08	1h48m	14d19h	Not enough jobs in the cluster. Minor issues with killing workers.
20200302T210301	100	32.72	1h18m	16d15h	Cluster size 8 - not optimal because there's a small over-spill (too hard to guess the correct size).
20200909T171340	100	35.47	1h15m	18d7h	Cluster size 5 and staging jobs 500
20200909T184835	100	32.16	1h13m	16d18m	Cluster size 5 and staging jobs 300
			t3.2xlarge		
20200921T163404	25	22.92	3h56m	18d20h	Didn't allocate more resources till 13m12s after job started.
20200918T225225	50	27.15	1h53m	19d17h	cluster 17. stageout 200. 3 stageouts failed
20200921T203942	100	37.81	1h26m	21d14h	clsuter 8. stageout 400
			Spot		
20200310T191539	200	28.21	1h58m	-	Instance type and size: m5.large
20200310T183810	50	6.24	1h59m	-	Instance type and size: c5.2xlarge

Table 2.2: Execution times and cost of optimized runs over different instance types and count. Notably, except when more compute resources are allocated than there are jobs available, irregardless of the size of the cluster size and instance type cost remains approximately the same due to the pay-for-what-you-use pricing policy.

4.1 Resource Optimization

Choice of instance type, number of instances and workflow determines the cost (either in wall time or USD) for the execution of the Rubin LSST Science Pipelines. In Figure-2.6 we show the performance scaling tests after workflow optimization. The example workflow shows scaling that is approximately linear with the number of simultaneously allocated jobs, which is dependent on the number of vCPUs, memory, and job resource requirements. The scaling factor of 1.8 is only slightly less than what is expected from an idealized case. The 50 and 100 workers tests for `c5.2xlarge` family do not show linear scaling. The total duration was 1h18m when executing with 100 workers, meaning the intialization task became a significant portion of the total wall time. Additionally, the example workflow does not have sufficient parallelizable jobs to saturate the allocated resources, i.e. the compute cluster was undersubscribed. Because resource allocation is based on vCPU (which are hyperthreads for the selected instance families and sizes), it is still possible for the physical core to be occupied and therefore the worker will not deallocate automatically but its full resources will remain unused. To verify, we executed an additional test using 200 `m5.large` workers, counting 400 vCPUs with 2 GB per vCPU of memory. The test finished in approximately the same wall time, 1h58m, as its vCPU number counterpart, the 50 `c5.2xlarge` workers. The cost for the workflows was 28.21 USD and 26.72 USD respectively, which confirmed that the scaling is mainly dependent on the number of jobs that are able to run in parallel.

The cost of EC2 instances is shown in Figure 2.7. We find that the cost is approximately constant, within 7%, for tests using 25 and 50 workers. This result is to be expected as the cost of more powerful instances is offset by a reduction in wall time. Therefore, if the cluster is oversubscribed, increasing the number of workers decreases both the wall time and the cost of the workflow (if the instances suit the workflow resource requirements). The workflow executed on 100 `c5.2xlarge` workers costs significantly more because the cluster was under-subscribed but, as explained previously, it was not possible to deallocate workers.

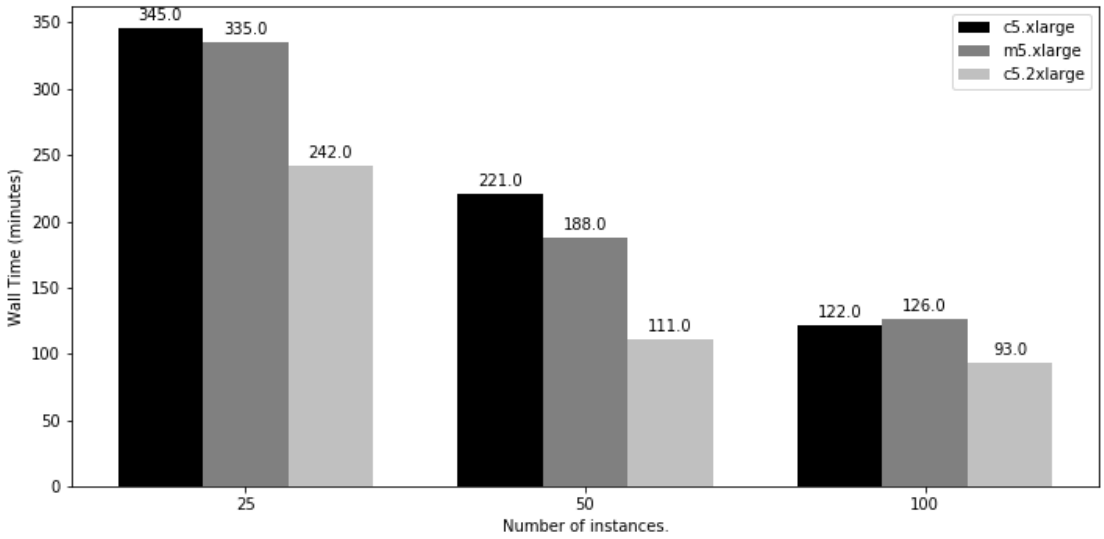


Figure 2.6: EC2 wall times for the example workflow.

The workflow executed on 100 `m5.xlarge` instances costs significantly less than its 25 and 50 worker counterparts because the reduction in wall time scaled as 1.8 meaning the instances were allocated for shorter periods of time. Significant savings are achieved by executing on Spot instances, with our example workflow, using 50 `c5.2xlarge` workers, reduced by a factor > 4 (see Figure-2.7). The total workflow cost is also offset by the constant cost of the RDS instance, the head node and amount of data stored in S3.

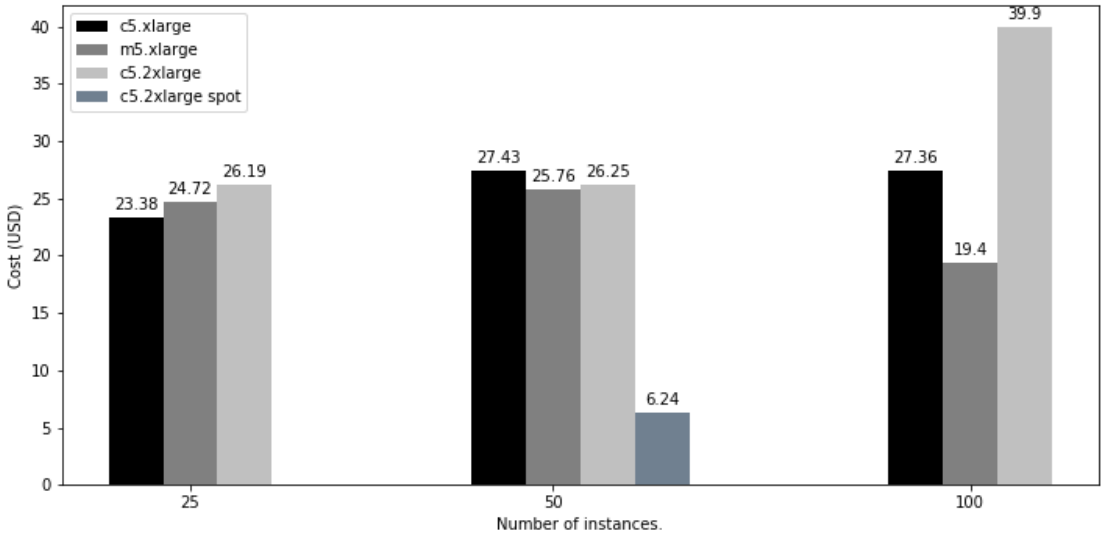


Figure 2.7: EC2 cost for the example workflow.

The scaling performance is not limited by IO or connection bandwidth on the workers mainly due to the distributive properties of S3. The upper limit is generally set by the size of the instance that hosts the Registry. Since all jobs need to contact the registry, there is an upper limit to the number of workers that can execute simultaneously set by the maximum number of simultaneous connections and maximum allowed number of locks per connection, which are in turn dependent on the available cache size of the RDS instance. The largest RDS instance available is the `db.m5.24xlarge` with 96 vCPUs, 384 GB of memory and 14000 Mbps of write speed to the attached EBS drives. The per-hour cost of such an instance is 8.45 USD/h of use, which is substantial in an on-demand provisioning scheme. Were this functionality offered as a service, however, cost can be significantly amortized by procuring reserved instances. For reserved instances a 40% discount is possible by reserving an instance for a year and 60% for three years.

5 Cost projections

The dataset used in this analysis represents approximately 11% of the number of nightly observations of Rubin or, given the difference in the size of the cameras, about 2% of Rubin's nightly data volume. In Chiang et al. (2020) we performed a cost estimate of a full Data Release Production (DRP) workflow, a workflow that in addition to the tasks described in Section 3 also includes combining images (coaddition) and source measurements on the combined images.

We base our on-premise resources estimates on Butler et al. (2020) where we multiply the per node cost with the walltime for a workflow executed using on-premise resources. The Verification Cluster at the National Center for Supercomputing Applications (NCSA) is used as a prototypical Rubin compute center. The Verification Cluster comprises 48 Dell C6320 nodes. Each node has 24 physical cores and 128 GB RAM, which due to hyperthreading results in 48 virtual cores per node. We estimate the cost of the on-premise compute resources

to be 0.408 USD per node-hour without including labor. We note that until the location of the Rubin Data Facility has been decided and concrete commitments are made this costing estimate will remain uncertain.

To provide a bound on the expected requirements and costs, in the following analysis we evaluate the DRP workflow using two different data abstraction layers known as the Generation 2 and 3 Butlers. Generation 2 Butler is the current and more mature implementation, and Generation 3 Butler, on which our work is based, is the less optimized next generation Data Butler. Processing took 17.8 node hours Chiang & Thrush (2020) and 88 node hours on the Verification Cluster for Data Butler Generation 2 and 3 respectively. This corresponds to run time costs of 7.3 USD and 35.9 USD. Given the large difference in the run times for the different generations, to estimate the total cost of processing Rubin data with the DRP pipelines, we use the *relative* performance of running these workflows in the cloud and on-premises and use that to scale the costs given in Chiang et al. (2020).

Rubin DRP estimates that approximately 2PB of raw science data will be produced annually. Based on an input size of 122 GB and a cost of 95 USD for the DRP workflow Chiang et al. (2020), assuming that the scaling remains linear, Rubin processing would cost approximately 0.7 million USD for the first year of DRP operations if we adopted no cloud optimizations. This compares to 0.9 million USD for the current on-premises implementation. If the optimizations we achieved in this paper can be extended across the entire DRP workflow, not just the example workflow we have evaulated, we estimate that Rubin yearly DRP processing costs would be between 600,000 to 900,000 USD. Note that this includes the compute, database access, and storage necessary to run the pipelines. It does not include the long-term archival costs of storing or querying the raw and processed images or the resulting catalogs. From these numbers, it is clear that optimization can yield substantial cost savings (between a factor of 1.7 and 2.5). If the optimizations shown in this paper can be extended across the full DRP workflow, we believe that the compute resources required for processing Rubin data could be cost neutral on the cloud.

6 Conclusions

We show that a lift-and-shift strategy is an appropriate and relatively easy way to adapt Rubin software to run in the cloud. While an unoptimized lift-and-shift approach can scale to multi-petabyte datasets, such as those that will be generated by Rubin Observatory, it is not a cost effective solution. While the current work does not address storage or networking costs, we identify several compute performance bottlenecks that drive processing costs and propose solutions to mitigate them. The most significant compute performance gains are achieved by careful management of workflow execution IO. Trimming significant amounts of log transfers, staging in an S3 bucket, and clustering reduced the example workflow walltime from 210–255 minutes to 128–140 minutes when executed on 50 `c5.xlarge` instances. The large dispersion in the unoptimized lift-and-shift example workflow walltimes is due to the nature of some of the bottlenecks.

We investigate the impact of EC2 instance size and type on performance and cost. We show that compute resources must be tailored to the requirements of individual Tasks within a complex workflow. For the example workflow, for small numbers of instances, walltime is a function of instance type and instance size and walltime scales linearly with the number of jobs that can run in parallel. Above 50 instances, this scaling no longer holds because the example dataset is not large enough to saturate the compute resources of the workers. Costs remain flat relative to instance type and number of instances for < 50 instances but for larger instance counts the under utilization of the compute resources increases the total cost. While the cost estimates for workflows run on inappropriately selected compute resources might not differ significantly, due to different pricing of instances, the total wall time of such a workflow will. Selecting a cost and performance optimal instance type and size requires careful benchmarking of each individual component of a workflow. For the case of the Rubin pipelines, adding more fine grained control that would allow associating, launching, or at least targeting Quanta to specific resources would enable better resource allocation, improve

total cost, and reduce run times.

Lastly, while the current results might not match the cost estimates for purchasing large scale on-premise compute resources for a long term 10-year sky survey such as the LSST, we show that it is possible to approach those estimates within 30% to 40%. We believe that for such a difference in relative pricing between on-premise and on-cloud resources, the added benefits of elasticity could outweigh the absolute cost of the workflow. Based on the work presented, we believe it is possible to achieve this relative price difference even when not including different provisioning schemes other than on-demand and spot. For longer lasting projects it would also be possible to reserve the EC2 and RDS resources required with 1 to 3 year reserved instance contracts. The pricing of EC2 and RDS instances is then further reduced by up to 72% and 65% respectively. With further work on the optimization of the workflow and system performance we believe, for the compute resources, cost parity with on-premises solutions can be achieved.

We note that the elasticity provided by moving into the cloud could allow the full Rubin data set to be reprocessed in under a week rather than the 6 months expected from an on-premises solution. Supplementing on-premise resources with on-demand or spot compute resources in the cloud could improve the speed with which research is conducted yielding a total net saving in delivering science results. This approach would transform how quickly science discoveries could be made using Rubin data.

CHAPTER 3

KBMOD

1 Introduction

The chemical composition and orbital dynamics of trans-Neptunian objects (TNOs) carry a record of dynamical and chemical history of the Solar System (Duncan et al., 1995; Fernández & Ip, 1984; Hahn & Malhotra, 2005; Levison et al., 2008; Malhotra, 1993; Nesvorný & Vokrouhlický, 2016; Tsiganis et al., 2005). Dominated by giant planets, their current orbital dynamics provide insight into historical orbital instabilities of Neptune (Dawson & Murray-Clay, 2012), the presence of, current and historical, planetary mass perturbers (Batygin & Brown, 2016), perturbations from sources external to the Solar System, such as close stellar encounters (Jílková et al., 2015) and perturbations from Galactic tides (Bannister et al., 2017). However, because of their distance and relatively low albedos, TNOs are challenging to observe. As late as the mid 2000's there were less than 500 known TNO; none of which were discovered beyond the 50 au distance from the Sun (Allen et al., 2002).

Multiple dedicated surveys using large-format CCD cameras and multi-meter telescopes (Jones et al., 2006; Schwamb et al., 2012; Trujillo et al., 2001) were used to improve on the number of known objects since. For example, the Deep Ecliptic Survey (Trilling et al., 2024) (DEEP) is a three-year NOAO/NOIRLab survey that was assigned 46.5 nights on the 4m Blanco Telescope on Cerro Tololo using the Dark Energy Camera (DECam; Flaugher et al. (2015)) that was designed to detect TNOs. The survey collected a series of exposures totaling a few hours on sky for each of several 2.7 square-degree DECam fields of view achieving approximate limiting magnitude of 26.2 mag in the VR filter.

The existence of DEEP and other such surveys motivates attempts to use the data for discovery of TNOs (Becker et al. (2018); Bernardinelli et al. (2020a); Gerdes et al. (2016, 2017) and more). The fundamental methodology of detection is shared across all such searches: identify objects that systematically change their position compared to stationary background stars across multiple images. Often the search is performed in catalog space, irreversibly relating detectability of an object to the limiting magnitude of a single exposure and the

cadence of a survey. Novel approaches for coadding of images while accounting for motion of the object, have shown promise, although often at the cost of compute resources (Fraser et al., 2024; Heinze et al., 2015; Napier et al., 2023; Nguyen et al., 2024; Whidden et al., 2019). It has been shown to be possible to mitigate these computational limitations, whether by hardware accelerators, better algorithmic-based search patterns, or by selective choices of search parameter space. Kernel Based Moving Object Detection (KBMOD) (Smotherman et al., 2021, 2023; Whidden et al., 2019) is one of these solutions, capable of testing billions of test trajectories per minute, by accelerating likelihood based digital tracking using Graphical Processing Units (GPUs).

In this chapter we will describe the results of performing a search for TNOs, using KBMOD on DEEP survey data. In section 2 we will describe the DEEP dataset on which we performed the search and how the data was processed by the Vera C. Rubin Science Pipelines. In 3 we will describe the search strategy and choice of methodology used to determine completion rates of the survey, describe the population of simulated objects injected into the data, the shortfalls of the DEEP survey cadence with respect to the generation of image differencing templates and how we resample images so that the pixels between images are all registered to the same coordinate grid on the sky. In Section 4 we begin by describing the workflow performance and used resources. We continue by describing the false positive filtering process and how positional uncertainties were derived directly from the measured likelihoods by executing an additional narrow pencil-like search for each result that passed filtering. Validation of the results by comparisons to inserted simulated objects, is performed and processing completeness and peak detection efficiency are estimated. In Section 5 we show the 44 preliminary results of the search that are statistically consistent with the detection of a new object in the outer solar system. In Section 6 we compare the performance and the completeness of the results of subset of this processing that matches the previous processing by as described by Smotherman et al. (2023). We identify the time-span within which orbits of objects remain approximately linear as the key constraint of KBMOD search.

We measure the threshold at which a trajectory becomes sufficiently non-linear for detection with KBMOD and how we implemented a correction for the reflex motion of the Earth can enable long time-base searches with KBMOD, for processing of Rubin data. Finally, in Section 7, we summarize our results and discuss future work, with particular focus on the upcoming LSST survey.

2 Data

2.1 DEEP Survey Strategy

Here, we provide a summary of the DEEP survey cadence, for the complete DEEP survey strategy see Trujillo et al. (2024). DEEP is evenly divided between the A and B semesters, roughly April-June and August-September. In each semester two patches of the sky are observed. The patches are selected such that they are located near the ecliptic, and that they could be observed for at least four hours each night. The patches are labeled A0, A1 for the first semester and B0, B1 for the second. Each patch has an origin field labeled with a small case 'a' (A0a, B1a...) from which it fans out in a triangular fashion consistent with the dispersion of differential apparent rates of motion of the objects themselves (Trilling et al., 2024; Trujillo et al., 2024). In each year of observing the number of patches increases. In Year I, three patches are observed. In Year II three new patches are observed and the Year I patches are re-visited. In Year III four new patches are observed while patches Year II and Year I are revisited. The slowest moving objects are thus expected to remain in their original field while faster moving objects are expected to be found in fields observed in subsequent years. The survey reported loss of A and B semesters in 2020 due to the COVID pandemic, extending the survey by a year, and forcing an update to sampling the region of sky in 2022 and 2023.

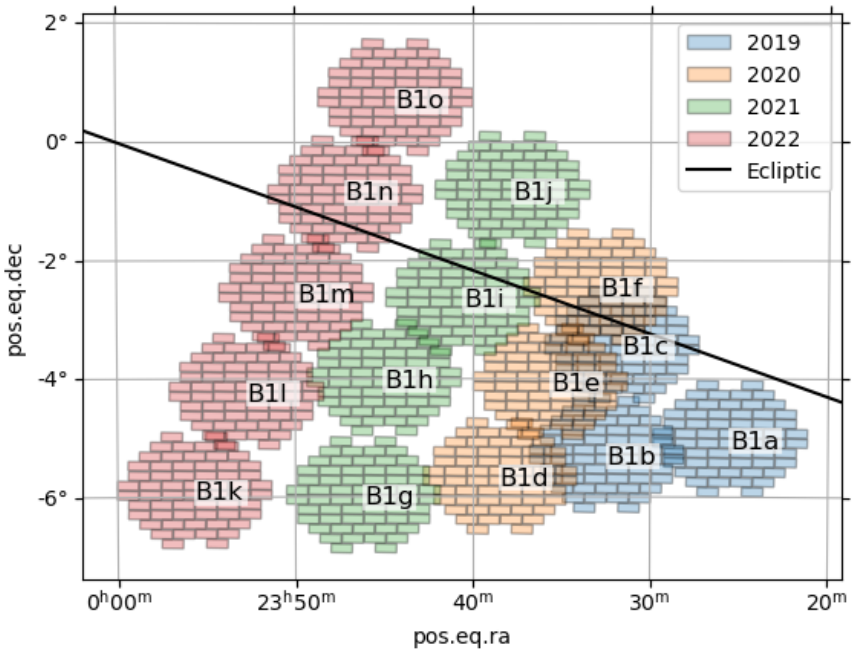


Figure 3.1: Pointing footprints for the B1 patch as derived from the WCS data colored by year and labeled with origin field.

2.2 Vera C. Rubin Science Pipelines

In the next decade, the LSST will produce a collection of datasets for which the data volume is measured on the petabyte scale. Such a large survey necessitates an extensive data management effort. To facilitate that effort Rubin has developed a set of interacting software components that facilitate access, processing, and analysis of the data and data products while meticulously tracking their provenance (Bosch et al., 2017). In order to develop and assess the integration and scalability of KBMOD, and related processing tooling, the DEEP data set was reprocessed using Rubin software.

There are two major components of the Rubin software that facilitate processing: Rubin Middleware and the Rubin Science Pipelines. The primary component of the Rubin Middleware is the Rubin Data Butler (Jenness et al., 2022). The Butler abstracts data file location and file formats from the user and developers of the Science Pipelines. The Rubin Science Pipelines are a collection of state-of-the-art image and catalog analysis algorithms (Bosch et al., 2018, 2019). There are two major components to the Rubin Science Pipelines: the

Alert Processing pipelines (AP) and the Data Release pipelines (DRP). The DRP generates calibrated images, coadds, image differences, and catalogs of detections and measurements derived from these datasets.

Target	Data product	Expected	Exist	Completion	Survey fraction
A0				**95.83**	22.5
	Raw	116932	116932	100.0	
	Calexp	114422	114322	99.9	
	Coadd	323567	311857	96.4	
	DifferenceExp	114025	112059	98.3	
A1				**76.85**	18.9
	Raw	98084	98084	100.0	
	Calexp	91064	76970	84.5	
	Coadd	136541	121812	89.2	
	DifferenceExp	76875	75378	98.1	
B0				**93.51**	14.3
	Raw	74090	74090	100	
	Calexp	72296	70398	97.4	
	Coadd	157807	154131	97.7	
	DifferenceExp	70386	69286	98.4	
B1				**93.41**	44.4
	Raw	230516	230516	100.0	
	Calexp	224686	219079	97.5	
	Coadd	343018	325502	94.9	
	DifferenceExp	218491	215323	98.6	

Table 3.1: Target completion (bold), the ratio of the number of successfully created difference exposures over the number of ingested raw images, per target stare A0, A1, B0 and B1. A breakdown of the number of expected and successfully processed key dataset types for each stare is shown under each target stare. The completion per dataset type is the ratio of expected and produced number of that particular dataset type. Finally, the survey fraction describes the number of raw images within the target stare versus total number of raw images in the survey. The number of produced coadds is different than the number of calibrated exposures because coadds are realized against the partitioning of the sky given by a skymap (Bosch et al., 2018), which does not have to map onto individual CCD footprints. Lower completion in A1 stare is associated with worse weather conditions and poorer data quality causing failures during processing with Rubin Science Pupelines.

The total cumulative data volume for the DEEP processing is approximately 200 terabytes (TB). This includes the raw image data, downloaded from NOIRLab, the instrument corrected, background subtracted, astrometry and photometry calibrated frames ("calexp"),

the calibrated images resampled onto a common pixel grid ("warps"), the stacked warps ("coadds") and difference images ("differenceExposure"). Two 5-sigma source catalogs were produced for calibrated exposures and difference images respectively. The data volume of difference exposures only is approximately 20TB.

The Rubin Science Pipelines image processing was executed on University of Washington Klone cluster and the Texas Advanced Computing Center STAMPEDE cluster. Due to the different memory footprints of each step in the processing, different numbers of differently scoped resources are allocated for each step. The total processing time for calibrated exposures was approximately 150 CPU-core days (1 wallclock day) 250 CPU-core (7 wallclock) days for coadds and 250 CPU-core (7 wallclock) days for image differences. Further overheads were incurred in the transfer of data, approximately 2 wallclock days.

3 Search and validation strategy

The DEEP dataset field labels, found in the image file headers, are not fully reliable. The pointings from Trujillo et al. (2024) were used to seed centroids for a K-Means clustering algorithm implemented in SciPy and new labels, based on the position of the center pixel on the image, were generated. These clusters were then additionally grouped by their CCD designation and night of observation. Successfully processed difference images were segmented into 13 132 image collections that form the basic unit on which the KBMOD search is executed. Due to observational restrictions, weather or failure to process the image collections span a range in the numbers of overlapping images.

KBMOD performs its search by testing a range of linear trajectories for every pixel in the image. Each trajectory is evaluated in pixel coordinates and varies its v_x and v_y, i.e. the angle and the rate of motion as shown on Figure 3.2. A new pixel coordinate is then calculated based on that angle, timestamps of exposures in the collection and the trajectory's velocity. In this scheme, all pixels must be aligned to the same world coordinate system, as

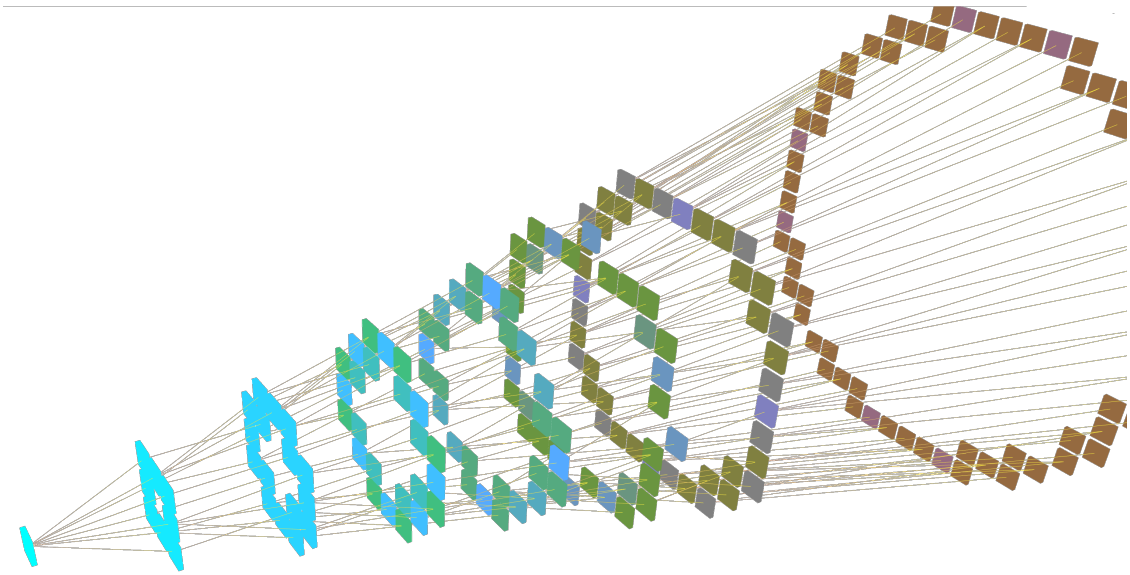


Figure 3.2: Diagram of a search performed by KBMOD for a single guess velocity and an uniform distribution of angles in the range $[0, 360]$ degrees. Small colored squares represent the searched pixels of an image and the lines the searched trajectories. Each plane in which the squares are found are images taken at a different time. The radius of the base of the search cone is determined by the search velocity and the section of the cone searched by the range of search angles.

otherwise the searched trajectory would not truly be locally linear on the sky.

In previous processing of the data (Smotherman et al., 2023) this step was performed as part of the creation of the image differences by the Rubin Science Pipelines. The calibrated exposures were resampled to the common sky coordinate grid of the overlapping coadd, from which the subtraction template would be derived, thus ensuring that all difference images were sampled on the common pixel grid. However, the image-subtraction algorithm has changed since then. When producing coadds, Rubin Science Pipelines resamples the calexps to a common skymap, but when deriving the template for the image difference, the coadd cutout is resampled to the coordinate grid of the calexp again. Consequently, no two image differences are sampled on the same on-sky coordinate grid. Before the search can be performed, the difference images in the collection must be resampled to the same coordinate grid. In order to support this functionality *Reproject*, an Astropy affiliated package for image reprojection, is used to fit a new, shared, WCS to all of the images within a collection and

resample them to a common coordinate grid using the DeForest (2004) adaptive anti-aliased resampling algorithm.

KBMOD offers a choice of several trajectory generators that produce the next guess trajectory for each pixel. The chosen search is centered on the ecliptic and spans a $-90°$ to $90°$ range relative to it. This limitation stems purely from the estimated time of a single search and the estimated number of results returned, where storage and hardware resources have to be taken into account. During the time-span of single night observations, 4 to 7 hours, the motion of real objects is dominated by the reflex motion of the Earth scaled by the object's topocentric distance. The rate of motion was constrained to $80 < r < 500$ pix/day in order to fully cover the space of bound orbits from approximately 20 to 100 au.

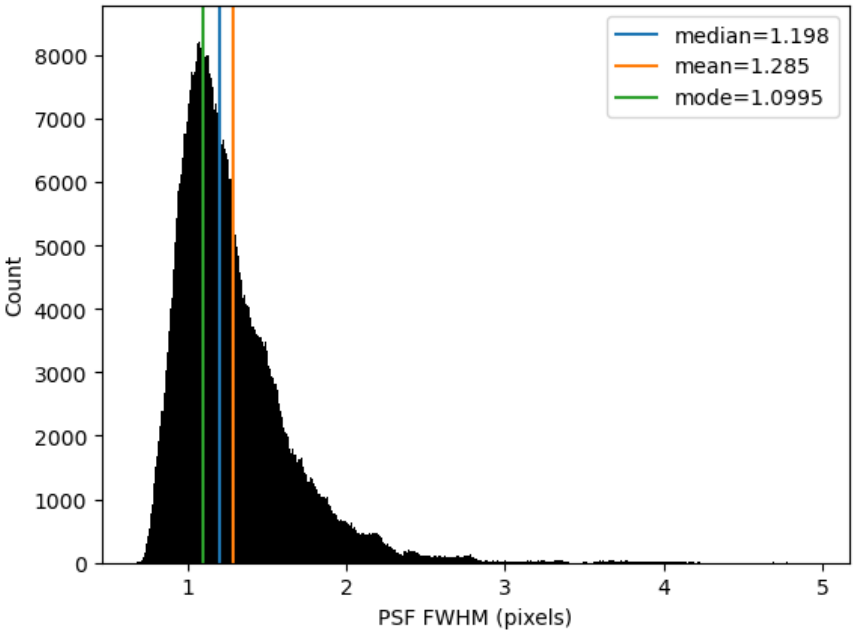


Figure 3.3: Distribution of PSF FWHM calculated as $2.355 \cdot 0.2637 \cdot \sigma_{psf}$ across all calexps that were included in the search where the PSF σ values were measured during image calibration and characterization steps in the Rubin Science Pipelines processing.

The choice of how many samples within the searched angles and rates of motion are somewhat limited by the availability of the compute resources, but are primarily driven by the detectability criteria of the KBMOD algorithm. KBMOD does not perform the search by directly accumulating pixel values. As described by Whidden et al. (2019), from

the set of overlapping images resampled such that their World Coordinate Systems (WCS, Greisen & Calabretta (2002)) are mutually rectilinear, the likelihood images are computed by convolving the images with their respective Point Spread Functions (PSF). The PSF is a two-dimensional brightness distribution produced in the detector by the image of an unresolved source such as a star, and, for a star located at the position y, we write the PSF as $T(x_i - y)$. For each pixel, $n(\mathbf{x}_i)$, two likelihood images are created. Likelihood image Ψ represents the likelihood the measured value of the pixel is due to an object located at y, while the variance image Φ represents the likelihood the measured value of the pixel is due to noise. We can write:

$$\Psi(\mathbf{y}) = \sum_i \frac{1}{\sigma_i^2} n(\mathbf{x_i}) T(x_i - y) \tag{3.1}$$

$$\Phi(\mathbf{y}) = \sum_i \frac{1}{\sigma_i^2} T(x_i - y) \tag{3.2}$$

so that the total likelihood of a trajectory through the images can be expressed as $\Psi_{\text{coadd}} = \sum_i \Psi_i(\mathbf{y}_i)$ and $\Phi_{\text{coadd}} = \sum_i \Phi_i(\mathbf{y})$ and a signal to noise ratio of a trajectory can be defined as:

$$\nu = \frac{\Psi_{\text{coadd}}}{\sqrt{\Phi_{\text{coadd}}}} \tag{3.3}$$

Accumulating values of likelihood will increase at a faster rate, compared to the variance, only when the searched pixel lies within an objects footprint on the image. As a rule of thumb, and given the dim nature of these objects, it is safe to assume that is on average this area equal to the size of PSF FWHM. The measured mode PSF FWHM of the entire DEEP survey is 1.1 arcseconds (see Figure 3.3). At the nominal pixel-scale of DECam of 0.2637 arcseconds/pixel[1] the footprint of a PSF has a diameter of 4.17 pixels. Considering that the distance between endpoints is 5 pixels for the largest image collection, containing 121 images, at least half of the trajectory length should lie within the PSF footprint assuming the starting point is the center pixel in the first image. Notably, since the maximum distance

[1]DECam Data Handbook Chapter 2 https://noirlab.edu/science/documents/scidoc0436

between neighboring trajectories is set for the largest image collection, and considering the footprint increases as the PSF FWHM increases, most image collections will have a smaller endpoint distances between consecutive trajectories.

A more complete discussion of the detectability of a trajectory, in a broader context of non-linearity of the apparent motion of an object on the sky, can be found in Section 6.1. The validation of the maximal distance between two test pixels that still allow for the detection of the object, estimated here a priori, can be found in the section 6.1 (in particular see Figure 3.22).

Because KBMOD runs a brute-force search of linear trajectories in a cone centered on each pixel (Whidden et al., 2019), running the search over difference images minimizes the number of false-positive detections caused by unfortuitous alignment of stars and galaxies that would otherwise be visible in a calexp. However, a number of image differencing artifacts may remain within the difference images themselves. A bitmask for each searched image is constructed from the masks provided by the Rubin Science Pipelines. Any pixel masked by Rubin Science Pipelines as 'bad', 'clipped', 'crosstalk', cosmic ray, edge, no data, saturated, sensor edge or 'suspect' are ignored in the search.

A series of performance tests were conducted with KBMOD on a machine containing NVIDIA GeForce RTX 2080Ti, AMD Ryzen Threadripper 3970X 32-Core processor and a 500GB NVME drive in order to provide an best-case estimate of the processing time. A linear relationship of the processing time and the size of the image collection is established as $t_{\text{processing}} = 26.4 N_{\text{images}} + 36.8$ seconds. A estimated total of ≈3500 wallclock hours would be required to process the whole dataset, sequentially, on a single machine using 32 parallel processes to resample the images. Approximately 93% of the cumulative processing time is required to process all collections containing 40 or more images. The tests can be used to estimate the cumulative processing time, as shown on Figure 3.4. This is a best-case estimate because it does not take into account additional IO overheads that may be incurred by accessing data on network drives, scheduling and batch execution overheads etc.

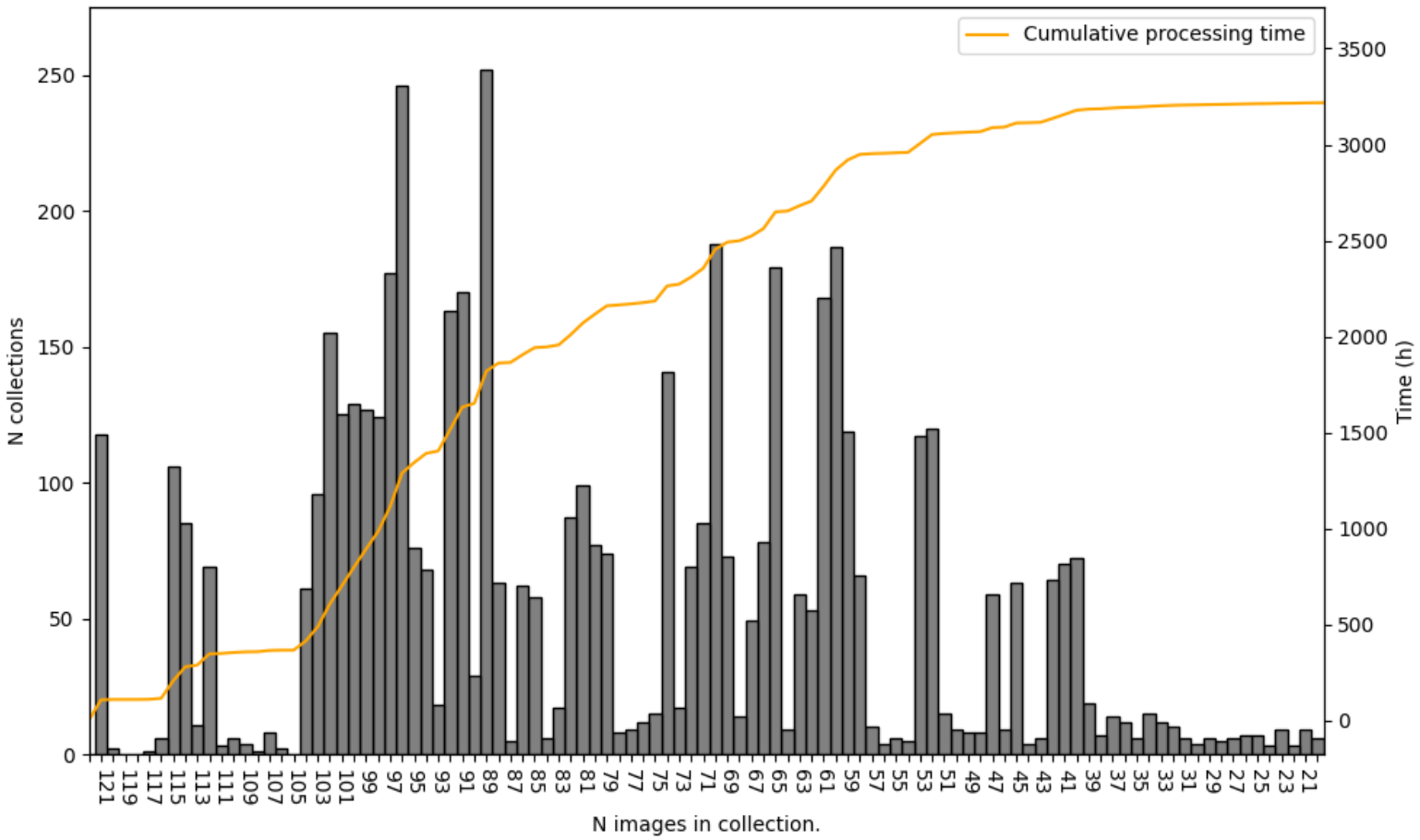


Figure 3.4: Distribution of number of images per image collection. The majority of image collections contain less than 10 images and are not displayed in this plot. Timings were executed on a machine containing NVIDIA GeForce RTX 2080Ti and AMD Ryzen Threadripper 3970X 32-Core processor. Timings are thus not necessarily representative of timings for large scale distributed clusters where data IO and the underlying GPU may be different, but may vary by a factor of few. From these timings a linear relationship to the number of images within a collection was derived as $t \approx 26.4 N_{\text{images}}[s]$ that was used to estimate the cumulative processing time in this plot.

3.1 Simulated moving sources

To estimate of the completeness, biases, and recovery rates in the survey simulated moving objects were injected into the raw images before processing. Detection efficiency of the search is then performed by fitting a likelihood of recovery of an simulated object given object parameters such as magnitude m and rate of motion r as described in Bernardinelli et al. (2020b). Briefly reviewed, we maximize the likelihood of detection:

$$\mathcal{L} = \prod_i p_{\text{detected}}(m_i|\theta) \prod_j (1 - p_{\text{undetected}}(m_j|\theta)) \tag{3.4}$$

assuming that the completeness follows a logit[2] function as a parameter of magnitude motion (Bernardinelli et al., 2020b; Bernardinelli et al., 2024; Smotherman et al., 2023) given as:

$$p_m = \frac{c}{1 + \exp\{\kappa_1(m - m_{50}\})} \tag{3.5}$$

or magnitude and rate of motion given as:

$$p(r|r_{50,1}, r_{50,2}, \kappa_1, \kappa_2, r_0) = \begin{cases} \frac{1}{1+\exp(\kappa_1(r-r_{50,1}))}, & r < r_0 \\ \frac{1}{1+\exp(\kappa_2(r-r_{50,2}))}, & r > r_0 \end{cases}$$

$$p(m, r|m_{50}, \kappa_1, \kappa_2, r_{50,1}, r_{50,2}, r_0, \kappa_1, \kappa_2, c) = c \cdot p(m|m_{50}, \kappa_1, \kappa_2, 1) \cdot p(r|r_{50,1}, r_{50,2}, \kappa_1, \kappa_2, r_0).$$

The m_{50} and m_{25} are the limiting magnitudes at which 50% and 25% completeness has been achieved. The κ_1 and κ_2 are the parameters of sharpness of the transition between "detected" and "not detected" regions for magnitude and rate logit functions respectively, where the limit $\kappa \to \infty$ represents a step function. The parameter $c \in [0, 1]$ is the peak recovery efficiency.

[2] quantile function associated with the standard logistic distribution

A sample of 54,177 simulated orbits were generated, 23,849 of which were TNOs and 5,716 of which were binaries. The location of the magnitude at which 50% recovery efficiency was achieved is a function of the number of images in an image collection and the limiting magnitude of a single exposure. In order to better constrain this point, simulated objects were densely sampled in the range of 24 to 26 magnitude; a range roughly corresponding to the limiting magnitude of a single exposure to a few magnitudes deeper. The goal of the simulated population is to fully characterize detectability and associated biases present in the algorithm and subsequent filtering. Therefore, the generated population of simulated objects occupies a very broad range of orbital parameter space, with significantly increased density of simulated objects within the range of orbital parameters we believe are targeted by the search. There are three distinct distributions that were used to derive the orbits of the simulated objects that follow the prescription given in Bernardinelli et al. (2024) (see also Figure 3.5):

- The classical Kuiper Belt objects that densely cover low eccentricity ($e < 0.4$) with semi-major axis a at 25 to 100 au range. Sampling the inclination distribution $i \sin \mu$ in the 0 - 60° range leads to an approximately uniform distribution of i within the survey footprint with the longitude of ascending node Ω and argument of perihelion ω being uniformly distributed in the 0-360° range.

- A logarithmically distributed excited population in the $20 < a < 2000$ au range, with perihelions uniformly distributed in the 10 - 100 au range. The inclination distribution spans 0 - 90° with the same uniform distribution of Ω and ω

- A uniformly sampled population of orbits half of which are logarithmically distributed between $25 < a < 1000$ au and half uniformly distributed between 25au and 100au.

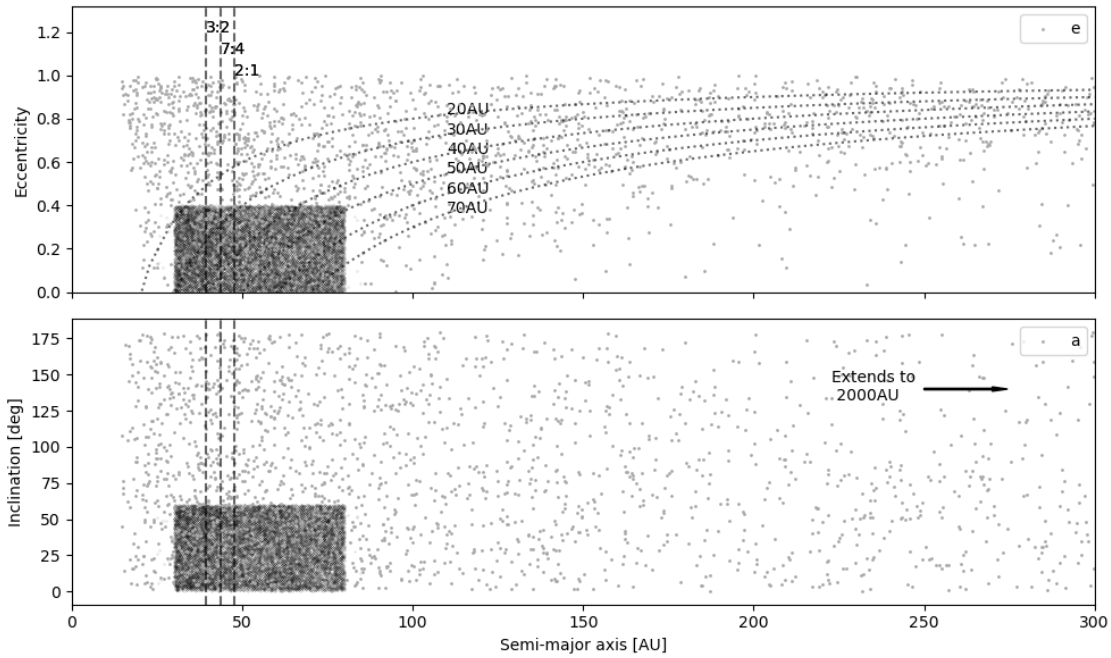


Figure 3.5: The distribution of orbital parameters of all (23,849) simulated orbits. The classical Kuiper Belt object appears as the very densely sampled square in the 25 - 100 au range. The Kuper Belt objects are superimposed on a uniform background of scattered population that has a uniform sampling in the range 0 - 100 au, and trails off logarithmically to 2000 au. Vertical dashed lines correspond to the approximate location of select Neptunian p:q mean-motion resonances. The curved dotted lines in the top panel correspond to lines of constant pericenter.

3.2 Template generation

The difference images are created by subtracting a photometrically matched template, derived from a coadd, from the calibrated exposure. The choice of statistic when creating the coadd, as well as the selection of calibrated exposures considered in creation of a coadd from which further image difference templates are derived, is arbitrary.

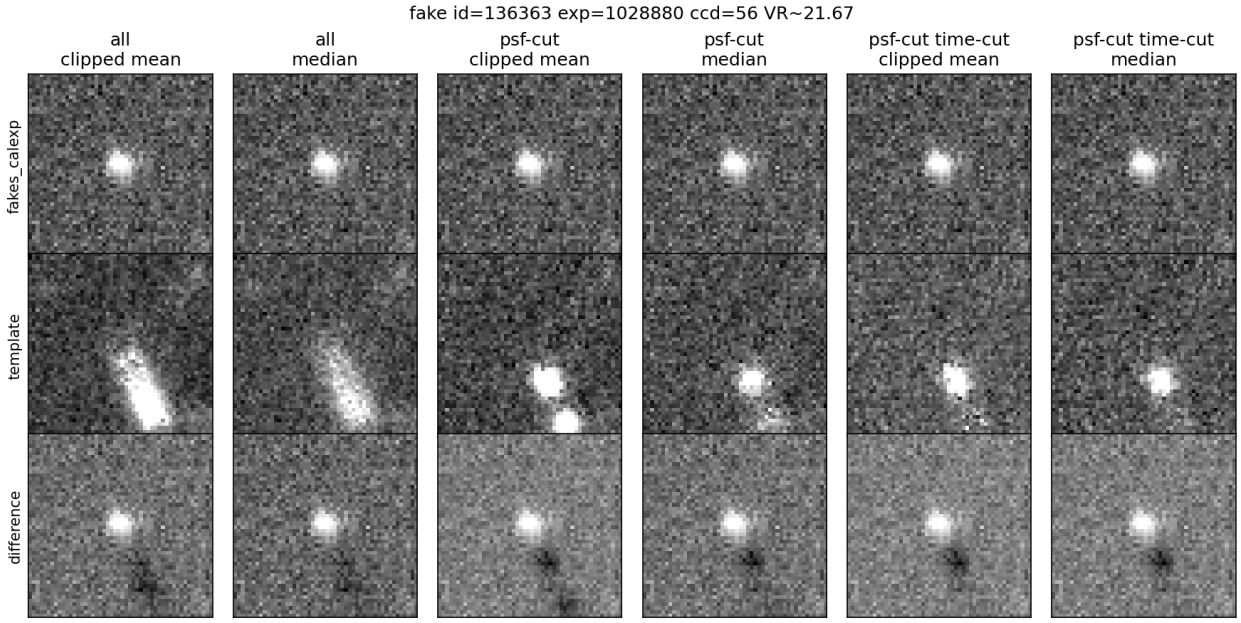


Figure 3.6: Postage stamps from the calibrated exposure (top row), the derived coadd template (middle row) and the difference exposure (bottom row) against the various combination of coadd statistic and coadd data selection choices, for one simulated object of magnitude 21.7. In order from left to right, all images with a sigma-clipped mean statistics, all images with a median statistic, top 33% images with the best PSF with a sigma-clipped median statistic, followed by a median statistic. Finally, only images from a different night than the exposure ID, with a best-PSF selection using both the sigma-clipped mean and median are displayed. In this case, for a rather bright slow-moving object, it is easy to see how its signature propagates through the template into the image difference.

DEEP cadence repeats exposures of the same area of the sky many times in a single night and multiple nights in a row. Approximately a year later, the sequence is continued, but offset for the dispersion of differential rate of motion of TNOs (Trujillo et al., 2001). Thus, the selection of images of the same region in the sky always occupy the neighboring few nights. Any object traversing across the CCD will therefore inevitably have its signal included in the

produced coadd. This additional signal above the sky background will propagate itself into the difference exposure. Different choices of the coaddition statistic and included calexps yield different image differences, containing artifacts that subsequently can effect the search for moving objects.

A series of overlapping images containing simulated moving objects were produced. These were then re-processed multiple times to create different coadds in order to investigate the effects of coaddition statistic and data selection function on the detectability of the object itself. There were 4 different coaddition statistics that were investigated: mean, sigma-clipped median, median, sigma-clipped mean, and mean. Sigma clipping is an iterative process of outlier rejection in which data outliers that are less or more than a specified number of standard deviations are rejected and a new statistic (i.e. mean or median) is estimated in each iteration. Three different data selection functions were considered: all data, just the top 33% of images with the best PSF FWHM (psf-cut) and a set of images selected such that, when possible, no image considered in the coadd was taken less than 10 minutes apart from the image collection in question.

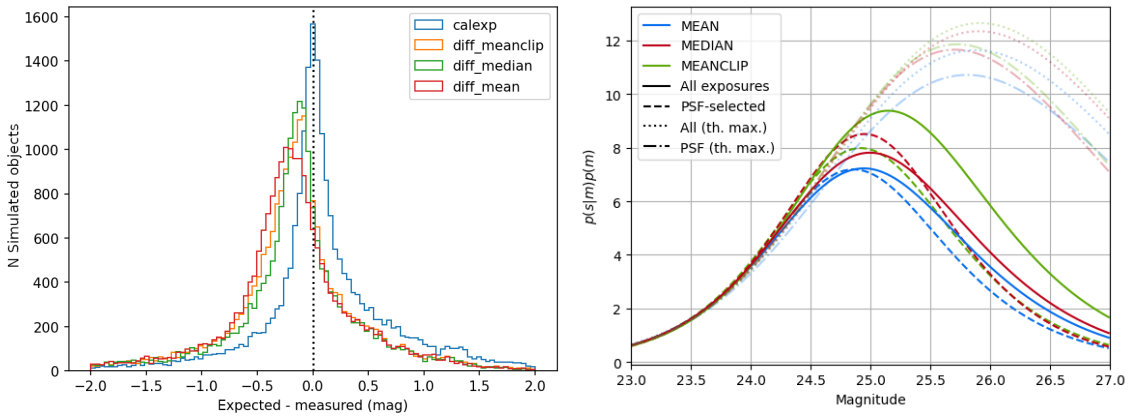


Figure 3.7: Forced photometry on simulated moving objects performed on the difference images (left) showing the systematic reduction in measured flux in the difference images, when coadds are created using data containing the object itself. Probability of detecting an object at a given magnitude (right) is estimated by positionally associating the search results to known simulated objects. When creating coadded templates, using all overlapping images with a sigma-clipped median yields the largest probability of recovery at lower magnitudes compared to other data selection and coaddition statistic functions.

Forced photometry is a process in which photometry is performed at a specified position in an image assuming that there exists an object at that location. Forced photometry measurements were then performed on the difference images, at the simulated object's location, and the difference between the expected measured flux and the measured flux were compared. Multiple runs of KBMOD were executed against the resulting difference images. The results were associated with the simulated objects when the trajectory of the starting pixel for the found detected object was within 2 arcseconds of the object. The probability of finding the object given the objects magnitude were evaluated as described in Bernardinelli et al. (2024). The choice of using all of the images with a sigma-clipped mean coaddition statistic showed itself to be the best template candidate as it removed the least amount of signal.

The effect of the chosen image selection on coadd creation is the formation of a bright region of increased flux along the length of the entire streak of the moving object within the coadd (see Figure 3.6 middle row). These brighter streaks persist through the creation of an image difference template and create regions in which the flux is over-subtracted, dubbed "negative wells", as shown in the last row of the Figure 3.6. This leads to a loss of flux of the object and to asymmetrical PSF-like appearance of the object within the image.

4 Processing validation

The largest 5 262 image collections were searched with KBMOD. This encompasses all image collections that contain more than 38 images in the collection, roughly corresponding to 90% of the estimated cumulative processing time required (see Section 3).

Using Parsl (Chard et al., 2021), the workflow was described in Python as a DAG, which could then be executed with the UW's Klone cluster utilizing the Slurm (Yoo et al., 2003) scheduler. While scaling to a few thousand jobs was not an issue with Parsl, the limited available storage space, Klone's early termination execution environment, and IO limits made

workflow execution more challenging.

In order to manage memory footprints, processing was executed in batches depending on the size of the image collections within. The images are scaled to the common zero-point, resampled to the same coordinate grid and the likelihood images are derived from the processed variance and image planes by convolving them with their PSF. The results of the pre-processing, a workunit, are materialized to disk and then a KBMOD search is performed, after which the workunits were removed. Jobs were assigned a retry policy in order to manage early termination failures. Because it is not easily discernible whether the job failed without the possibility of recovery, or was terminated early, multi-hour failing jobs were often retried several times. Including these overheads, total wallclock processing time is 8.1 days and an estimated 4 486 CPU-core hours were utilized, a 27.5% increase in used CPU hours compared to the best case estimate. Approximately 375 thousand potential results were recovered.

4.1 Filtering

During execution, potential results were filtered to keep only the results whose cumulative likelihood exceeded 7σ. Within each image collection, results were clustered in starting pixel position (x, y) and velocity (v_x, v_y), space using DBSCAN algorithm (Chaudhuri et al., 2014). Only the most likely result within each cluster is kept.

A 21 by 21 pixel postage stamp were created centered on the positions of the inserted simulated sources. A set of additional postage stamps, using search parameters that do not cover the parameter space of physically meaningful orbits, were created. Only orbits nearly perpendicular to the ecliptic were searched for this set of postage stamps. A machine learning network based on the ResNet-50 (He et al., 2015) was trained on mean and median coadds using the first and second set of postage stamps as true and false positive samples respectively. The training set was augmented by creating mean and median coadds after applying arbitrary rotation, mirroring, and translations of the postage stamps used to create

the coadd. Lower signal-to-noise ratio coadded postage stamps were created for training purposes by selecting a subset of 25 or more individual postage stamps from the two postage stamp datasets and creating new co-adds from this subset of postage stamps. Example stamps are shown on Figure 3.8. A 95% recovery rate was achieved, while rejecting 75% of the false positives was achieved in the training set. The set of potential results was then further filtered using the trained network to recover a set of approximately 87 thousand likely results.

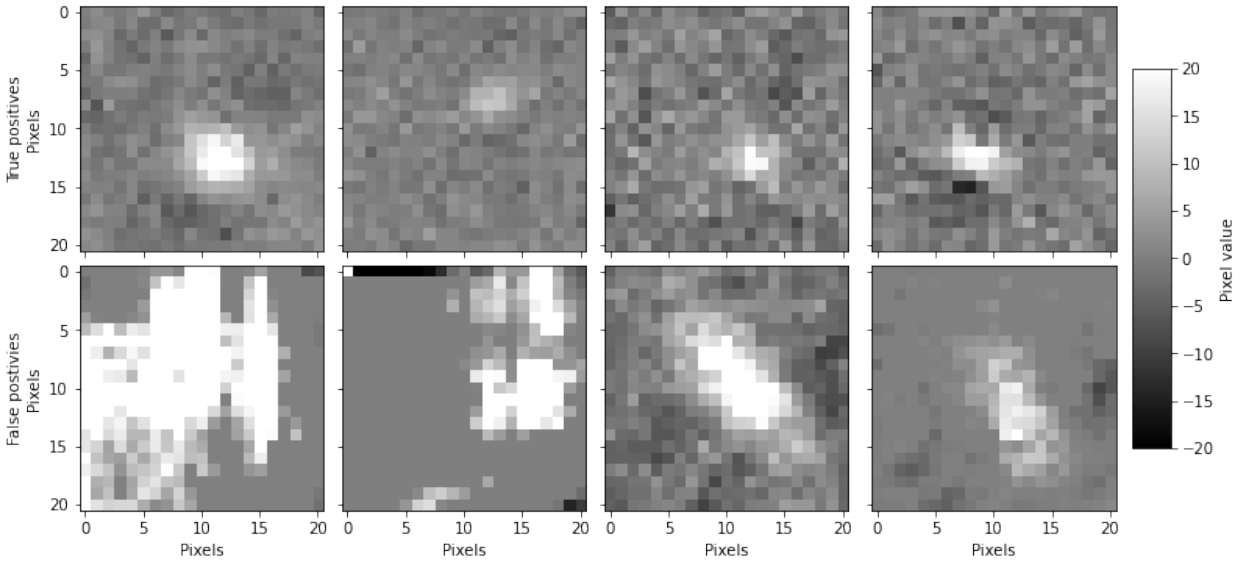


Figure 3.8: Example median coadded postage stamps. True positive stamps are centered on the simulated moving sources (see Section 3.1). The false positive stamps (second row) were centered on the trajectories returned from KBMOD searches perpendicular to the ecliptic thus minimizing the changes of detecting a real object, while the true positive stamps (first row) were created from postage stamps centered on the positions of the injected simulated sources.

4.2 Uncertainty propagation

An additional KBMOD search was executed for each likely result. A narrow pencil-like trajectories centered on the likely result itself were searched within a 10 pixel neighborhood. The range of angles $[-45°, 45°]$ were searched around the direction of motion $\boldsymbol{x}$ of the likely result with a 1 degree resolution. With respect to velocity $\boldsymbol{v}$, a range $[-45, +45]$ around the likely result's velocity, were searched with a 0.55 pixel/day resolution. The search improves the accuracy of the position and velocity values of the guess trajectory and improves the likelihood of detection (see Figure 3.9).

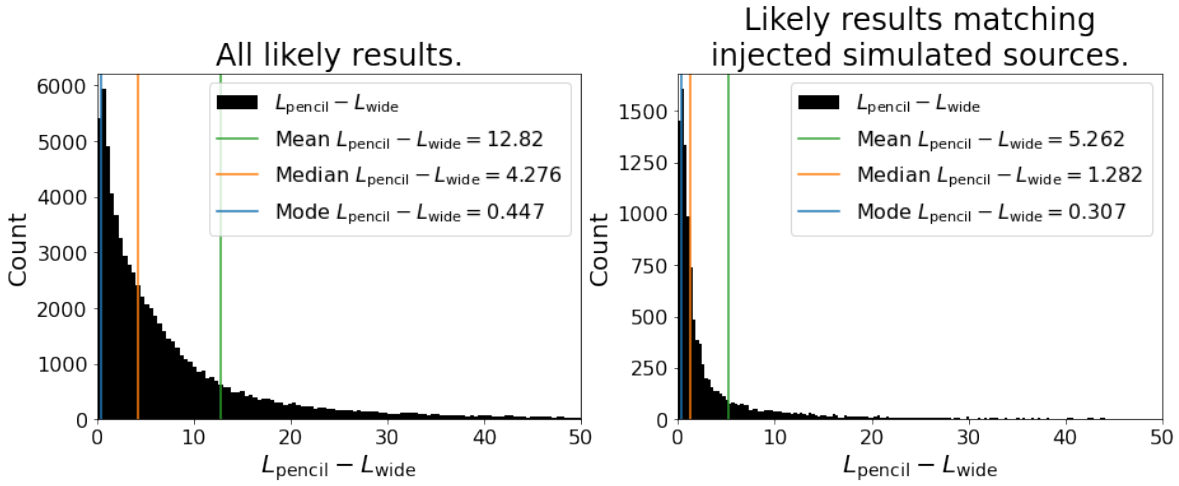


Figure 3.9: Difference between the KBMOD SNRs of the same likely results after the additional pencil-wide search. On average we find an increase of the likelihood of the likely result, showing improved trajectory was found in the narrower pencil-like search. On the left are all searched likely results, majority of which can still be expected to be false detections, while on the right are those likely results that match the positions of an injected simulated object. For details on how the likely results were matched to injected simulated sources see Section 4.

The primary purpose, however, for the set of trajectories $(\boldsymbol{x}, \boldsymbol{v})$ and their associated log-likelihoods $\nu \equiv \nu(\boldsymbol{x}, \boldsymbol{v})$, recovered in narrower pencil-like searches, was the calculation of the expectation value of on-sky position, velocity and their associated uncertainties. The expectation value of position and velocity, in the space of pixels searched by KBMOD (see Equations 3.2) and their covariance is given by:

$$\hat{\boldsymbol{x}} \equiv \boldsymbol{E}[\boldsymbol{x}] \quad = \frac{\sum_i \boldsymbol{x_i} e^{\nu_i}}{\sum_i e^{\nu_i}} \tag{3.6}$$

$$\hat{\boldsymbol{v}} \equiv \boldsymbol{E}[\boldsymbol{v}] \quad = \frac{\sum_i \boldsymbol{v_i} e^{\nu_i}}{\sum_i e^{\nu_i}} \tag{3.7}$$

$$\Sigma_{\boldsymbol{xv}} = \mathrm{Cov}[X_i, Y_j] = \quad E[X_i Y_j] - \hat{X}_i \hat{Y}_j, \tag{3.8}$$

where X_i and Y_j are the components of position (x, y, v_x, v_y) and velocity. The corresponding start and end pixel coordinates of the trajectory, $\boldsymbol{p}_1$ and $\boldsymbol{p}_2$ respectively, can be calculated relative to an arbitrary time origin t_0. For simplicity it is useful to define $dt_1 = t_{\mathrm{initial}} - t_0$ and $dt_2 = t_{\mathrm{final}} - t_0$) and take $t_0 = t_{\mathrm{initial}}$ such that $dt_1 = 0$. The transform, it's Jacobian and covariance matrix are then given by

$$\hat{\boldsymbol{p}}_1 \quad = \hat{\boldsymbol{x}} + \hat{v}(t_{\mathrm{initial}} - t_0) \tag{3.9}$$

$$\hat{\boldsymbol{p}}_2 \quad = \hat{\boldsymbol{x}} + \hat{v}(t_{\mathrm{final}} - t_0) \tag{3.10}$$

$$J_p \quad = \begin{pmatrix} 1 & 0 & dt_1 & 0 \\ 0 & 1 & 0 & dt1 \\ 1 & 0 & dt_2 & 0 \\ 1 & 0 & 0 & dt_2 \end{pmatrix} \tag{3.11}$$

$$\Sigma_{\boldsymbol{pp}} \quad = J_p \Sigma_{\boldsymbol{xv}} J_p^T. \tag{3.12}$$

Of interest are the on-sky coordinates of the line segment that best fits the observed object's orbit and associated positional uncertainties. The transformation between the pixel coordinates and the world coordinates is given by the World Coordinate Systems (WCS) transformations. WCS are not, in the general case, linear maps but they can be linearized within a small enough neighborhood of a point. Similarly to the previous transformation, we can say:

$$\hat{\boldsymbol{s}}_1 \equiv \{\alpha_{\text{initial}}, \delta_{\text{initial}}\} = WCS(\boldsymbol{p}_1) \tag{3.13}$$

$$\hat{\boldsymbol{s}}_2 \equiv \{\alpha_{\text{final}}, \delta_{\text{final}}\} = WCS(\boldsymbol{p}_2) \tag{3.14}$$

$$J_{\alpha\delta} = \begin{pmatrix} J_{\text{WCS}} & 0 \\ 0 & J_{\text{WCS}} \end{pmatrix} \tag{3.15}$$

$$\Sigma_{\boldsymbol{ss}} = J_{\alpha\delta}\Sigma_{\boldsymbol{pp}}J_{\alpha\delta}^T. \tag{3.16}$$

The Jacobian is, in this case, a block-diagonal matrix of the Jacobian associated with the transformation from pixel coordinates to world coordinates denoted J_{WCS}. There are two transformations associated with the block element J_{WCS}: the transformation in the intermediate pixel coordinates (u, v) and the projection effects of the transformation of the intermediate pixel coordinates (u, v) from a projection[3]. Generally, the effects of a projection can locally be approximated as a linear map given by the expression:

$$J_{\text{deprojection}} = \begin{bmatrix} \cos(\delta)\frac{d\alpha}{du} & \cos(\delta)\frac{d\alpha}{dv} \\ \frac{d\delta}{du} & \frac{d\delta}{dv} \end{bmatrix}. \tag{3.17}$$

For example, gonomonic projection, whose inverse is defined as[4]:

$$\sin(\delta) = \cos(k)\sin(\delta_0) + \frac{v\sin(k)\cos(\delta_0)}{\rho}$$
$$\tan(\alpha - \alpha_0) = \frac{u\sin(k)}{(\rho\cos(\delta_0)\cos(k) - v\sin(\delta_0)\sin(k))},$$

where $\rho = \sqrt{u^2 + v^2}$ and $k = \arctan\rho$. The distortions are linearized around a new ref-

[3]Note the inverse direction of this transform. We are given u and v and are looking for corresponding α and δ, whereas the more common direction of transformations are from a coordinate to its image.

[4]https://mathworld.wolfram.com/GnomonicProjection.html

erence coordinate $(\alpha_0, \delta_0) \equiv (u_0, v_0) \equiv (x_0, y_0)$ and not with respect to the originating WCS's reference coordinate (neither the world origin (CRVAL1, CRVAL2) nor the pixel (CPRIX1, CRPIX2)). Conveniently, the new on-sky reference position is selected to be the average of the start and end positions of the trajectory $\overline{s}$ to minimize the errors resulting from the linearization of the distortion at either end of the trajectory. The new reference pixel is related to the new reference pixel position $\overline{p} = (p_1 + p_2)/2$ via the WCS. As the search strategy is focused on relatively short single-night observations, the slow-moving distance TNO objects limit the distance between p_1 and p_2, on the order of magnitude, to a few arcseconds (<20 pixels) - well within the validity of small angle approximation. The analytic derivations of the transform can be evaluated with symbolic math packages, such as Wolfram Alpha of SymPy for each projection, but it is more convenient to adopt a parameterization used by LSST *Coord*[5] class as it unifies several projections via clever substitution:

$$\sin(\delta) = \frac{\cos(k)\sin(\delta_0) + v\sin(k)\cos(\delta_0)}{\rho} \tag{3.18}$$

$$tan(\alpha - \alpha_0) = \frac{u\sin(k)}{\rho\cos(\delta_0)\cos(k) - v\sin(\delta_0)\sin(k)}, \tag{3.19}$$

where

$$\begin{aligned}
\rho &= \sqrt{u^2 + v^2} & \\
k &= \arctan\rho & \text{Gnomonic} \\
k &= 2\arctan\tfrac{\rho}{2} & \text{Stereographic} \\
k &= 2\arcsin\tfrac{\rho}{2} & \text{Lambert} \\
k &= \rho & \text{Postel,}
\end{aligned}$$

[5] https://github.com/LSSTDESC/Coord/blob/releases/1.3/coord/celestial.py#L845

The derivatives are given by:

$$\begin{aligned} d\sin(\delta) &= \cos(\delta)d\delta = \sin\delta_0 dc + \cos\delta_0(vds + sdv) \\ d\tan(\alpha-\alpha_0) &= \sec^2(\alpha-\alpha_0)d\alpha \\ &= \frac{(uds+sdu)A - us\left(\cos\delta_0 dc - \sin\delta_0(vds+sdv)\right)}{A^2}, \end{aligned}$$

where

$$s = \tfrac{\sin k}{\rho}$$

$$c = \cos k$$

$$A = c\cos\delta_0 - vs\sin\delta_0.$$

Writing the full differential in terms of its components yields:

$$m = \sqrt{1-(c\sin\delta_0 + vs\cos\delta_0)^2}$$

$$\tfrac{d\delta}{du}m = \sin\delta_0\tfrac{dc}{du} + v\cos\delta_0\tfrac{ds}{du}$$

$$\tfrac{d\delta}{dv}m = \sin\delta_0\tfrac{dc}{dv} + \cos\delta_0\left(v\tfrac{ds}{du} + s\right)$$

$$n = A^2 + (us)^2 = (c\cos\delta_0 - vs\sin\delta_0)^2 + (us)^2$$

$$\tfrac{d\alpha}{du}n = \left(u\tfrac{ds}{du} + s\right)A - us\left(\cos\delta_0\tfrac{dc}{du} - v\sin\delta_0\tfrac{ds}{du}\right)$$

$$\tfrac{d\alpha}{dv}n = u\tfrac{ds}{dv}A - us\left(\cos\delta_0\tfrac{dc}{dv} - \sin\delta_0\left(v\tfrac{ds}{dv} + s\right)\right).$$

Substituting the definition for gnomonic projection $k = \arctan\rho$ yields:

$$\begin{aligned} s &= \frac{\sin(k)}{\rho} = \frac{1}{\sqrt{1+\rho^2}} \\ \frac{ds}{du} &= -\frac{u}{(1+\rho^2)^{\frac{3}{2}}} = -us^3 \\ \frac{ds}{dv} &= -\frac{v}{(1+\rho^2)^{\frac{3}{2}}} = -vs^3 \end{aligned} \qquad \begin{aligned} c &= \cos k = \frac{1}{\sqrt{1+\rho^2}} \\ \frac{dc}{du} &= -\frac{u}{(1+\rho^2)^{\frac{3}{2}}} = -us^3 \\ \frac{dc}{dv} &= -\frac{v}{(1+\rho^2)^{\frac{3}{2}}} = -vs^3, \end{aligned}$$

for stereographic projection $k = 2\arctan\rho/2$:

$$\begin{aligned} s &= \frac{\sin(k)}{\rho} = \frac{4}{4+\rho^2} \\ \frac{ds}{du} &= \frac{-8u}{(\rho^2+4)^2} = \frac{dc}{2du} \\ \frac{ds}{dv} &= \frac{-8v}{(\rho^2+4)^2} = \frac{dc}{2dv} \end{aligned} \qquad \begin{aligned} c &= \cos k = \frac{4-\rho^2}{4+\rho^2} \\ \frac{dc}{du} &= \frac{-16u}{(\rho^2+4)^2} = -us^2 \\ \frac{dc}{dv} &= \frac{-16u}{(\rho^2+4)^2} = -vs^2, \end{aligned}$$

for Lamberts projection $k = 2\arcsin\rho/2$:

$$\begin{aligned} \frac{\sin(k)}{\rho} &= \frac{\sqrt{4-\rho^2}}{2} \\ \frac{ds}{du} &= \frac{-u}{2\sqrt{-\rho^2+4)}} = \frac{-u}{4s} \\ \frac{ds}{dv} &= \frac{-v}{2\sqrt{-\rho^2+4)}} = \frac{-v}{4s} \end{aligned} \qquad \begin{aligned} \cos k &= \frac{2-\rho^2}{2} \\ \frac{dc}{du} &= -u \\ \frac{ds}{dv} &= v, \end{aligned}$$

and finally for the Postel, $k = \rho$, projection:

$$\begin{aligned} s &= \frac{\sin(k)}{\rho} = \frac{\sin(\rho)}{\rho} \\ \frac{ds}{du} &= \frac{u}{\rho^2}\left(\cos\rho - \frac{\sin\rho}{\rho}\right) = (c-s)\frac{u}{\rho^2} \\ \frac{ds}{dv} &= \frac{v}{\rho^2}\left(\cos\rho - \frac{\sin\rho}{\rho}\right) = (c-s)\frac{v}{\rho^2} \end{aligned} \qquad \begin{aligned} c &= \cos k = \cos\rho \\ \frac{dc}{du} &= \frac{-u\sin\rho}{\rho} = -su \\ \frac{dc}{dv} &= \frac{-v\sin\rho}{\rho} = -sv. \end{aligned}$$

The WCS intermediate pixel transform is by design an affine map. In other words the "cd" matrix in the CD formalism, or the PC matrix and scaling in the PC formalism (Calabretta &

Greisen, 2002; Greisen & Calabretta, 2002). These are provided in the headers and equivalent to each other:

$$J_{\text{affine}} = \begin{bmatrix} CD11 & CD12 \\ CD21 & CD22 \end{bmatrix} \tag{3.20}$$

$$= \begin{bmatrix} CDELT1 \cdot PC11 & CDELT1 \cdot PC12 \\ CDELT2 \cdot PC21 & CDELT2 \cdot PC22 \end{bmatrix} \tag{3.21}$$

Additional non-linear corrections, such as Simple Imaging Polynomial (SIP), TPV, TNX or ZPZ World Coordinate Systems can also be considered. Such nonlinear corrections to (u, v) can be expressed as a matrix multiplication. For example, in the SIP convention defined as:

$$\begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} CD11 & CD12 \\ CD21 & CD22 \end{bmatrix} \begin{bmatrix} u + f(u,v) \\ y + g(u,v) \end{bmatrix}, \tag{3.22}$$

each polynomial $f(u,v) = \sum_{pq} A_{pq} u^p v^q$ and $g(u,v) = \sum_{pq} B_{pq} u^p v^q$ can be expressed as an matrix multiplication $f = \boldsymbol{u}A\boldsymbol{v}$ and $g = \boldsymbol{u}B\boldsymbol{v}$, where $\boldsymbol{u}$ and $\boldsymbol{v}$ are the vectors in terms of polynomial exponents of the intermediate coordinates; or their offsets from the center coordinates of the distortion - depending on the convention. In this sense it is easy to evaluate the Jacobian of the SIP correction at the new reference intermediate pixel position $\overline{u} = (u_0, v_0) = \overline{p} - \text{CRPIX}$,

$$J_{\text{SIP}} = \begin{bmatrix} (\boldsymbol{v}Ad\boldsymbol{u})|_{\overline{uv}} & (d\boldsymbol{v}A\boldsymbol{u})|_{\overline{uv}} \\ (\boldsymbol{v}Bd\boldsymbol{u})|_{\overline{uv}} & (d\boldsymbol{v}B\boldsymbol{u})|_{\overline{uv}} \end{bmatrix}, \tag{3.23}$$

since the derivation of a polynomial $d\boldsymbol{u} = d(\sum_{i=0}^{N} u^i) = \sum_{i=1}^{N-1} iu^i$ is trivial to implement. To put it simply, the new reference position in the intermediate pixel position is calculated as $\overline{u} = (u_0, v_0) = \overline{p} - \text{CRPIX}$ and the vectors $\boldsymbol{u} = [u_0^0, \cdots, u_0^{\text{A\_ORDER+1}}]$ and $\boldsymbol{v} = [v_0^0, \cdots, c_0^{\text{B\_ORDER+1}}]$ are created. For simplicity both polynomials are expanded to

the same order, with zero coefficients assigned to non-existing orders, such that ORDER = A_ORDER = B_ORDER. To derive the $\boldsymbol{u}$ and $\boldsymbol{v}$, all that is required is to multiply the vectors by a vector containing exponents shifted to the left by 1 place, i.e. $du = [0, u_0^1 \cdots, u_0^{\mathrm{ORDER}}]$. AstroPy's WCS object, for example, contains the A and B matrix of elements correctly interpreted from the FITS Header and already padded to the same order, so that calculating the elements of the SIP Jacobian (e.g. $\boldsymbol{v}Ad\boldsymbol{u}$) are trivial.

Considering non-linear polynomial corrections to the WCS is generally not needed due to image resampling. During image resampling the WCS of each exposure is considered, and a new footprint on the sky is calculated. A new WCS is then created, not including nonlinearities, which is used to create the new coordinate-pixel grid onto which new pixel values are calculated. The non-nonlinearities of each individual image present themselves as non-uniform grid of samples on the new coordinate grid from which the new values of pixels are calculated by interpolation. This naturally incurs penalties such as correlated noise, aliasing, moire effects etc, but are not considered major issues in the DEEP dataset as exposures within the night are very well aligned, with at worst a few pixel translation between two images.

4.3 Orbit linking and orbit fitting

After the fine mesh grid search, for each likely result (see Section 4.1) we have improved measurements of a) expected values of starting positions, velocities and their covariance matrix, b) on-sky positions in each observed time-stamp and the associated covariance, and c) the tracklet-like start and end observed positions with associated covariance matrix. Because the searched image collections were created by grouping the same CCDs in the same nights, the returned results represent observed inter-night tracklets. The goal is to find which of these single-night tracklets belong to the same orbit, "linking" them toghether, and then perform an orbit fit on the linked tracklets. From all fitted orbits we can sub-select (based on the quality of fit parameters such as reduced chi-square, astrometric residuals, number of

linked tracklets etc.) to find the set of orbits we believe to be real.

To link the detected tracklets into multi-night orbit observations, we use the procedure described in Bernardinelli et al. (2022) which was shown to be 99% effective for DES data. The linking proceeds by first finding a multi-night tracklet. Single night (a start or end of a single-night tracklet) detections from one night are linked to detections in other nights by clustering all pairs of exposures within some time $\Delta t = t_2 - t_1$, for which a bound TNO orbit is possible for a given distance. The distances are searched in bins of inverse distance, $\gamma = 1/d$ (Bernstein & Khushalani, 2000), for $30 < d < 300$ and the full timespan of the survey. Triplets are then found by decomposing the motion of the pair into the parallactic and binding axes as defined by Bernardinelli et al. (2022).

An orbit is fit to the found triplet of linked detections as described by Bernstein & Khushalani (2000) with a strong Gaussian prior on the inverse distance γ. Because the start and end point of each tracklet detected by KBMOD, are highly correlated, we adopt the same approach as Smotherman et al. (2023) and modify the Bernstein & Khushalani (2000) orbit fitting to include the full uncertainty covariance as determined by the refined fit (see 4.2). In order to maintain numerical stability and robustness, the initial fitting proceeds in 2 steps: first an orbit is fit with homoskedastic uncertainties on all positions, after which the heteroskedastic uncertainties are used. After the triplet has been fit with an orbit, a search for individual detections consistent with the fitted orbit is performed over all exposures. If the detection lies within 4σ of the predicted error ellipse, it is added to the list of detections used to fit the orbit and in an iterative procedure, the orbit fitting is attempted again until no new individual detections can be found.

Each orbit fit result is then evaluated by its quality of fit metrics $\chi^2/\nu < 5$, the number of unique nights (NUNIQUE$>$ 4) on which detections were found and the time between the first and last detections in the orbit (ARC$>$ 1 year). Orbit candidates not satisfying these criteria will not be considered real. We recover 9,967,411 linkages in this process ($\approx 200GB$), which we are currently validating.

A similar but different linking approach, HelioLinC, is described by Holman et al. (2018). Similarly to the approach described above, the orbits of objects are propagated along Keplerian orbits under the assumption of a monopole gravitational acceleration from the Sun. The observed topocentric coordinates are then transformed to heliocentric frame, removing the reflex motion of the Earth. The heliocentric coordinates are then propagated to the same epoch by assuming the inverse scaled distance γ and the rate of change of scaled distance $\dot{\gamma}$. When the assumed distance and its rate of change is close to the true value the components of the angular position of the object at the reference time, α and β and the corresponding angular rates of motion of the inertial coordinate system, $\dot{\alpha}$ and $\dot{\beta}$, (referred to as "arrows" in Holman et al. (2018)) cluster together. A clustering algorithm then finds these clusters returning them as linked tracklets. The approach was shown to be able to correctly recover linked tracklets for main belt (MBO) and near-Earth objects (NEOs) over a period of approximately two weeks to 20 days. Compared to the Bernardinelli et al. (2022) linking approach, the approach described by Holman et al. (2018) requires an a priori choice of test orbit hypothesis and clustering parameters, potentially leading to lower completion rates. The advantage of the approach is that it simplifies the computational complexity of the problem from $O(N^3)$ to $O(N \log N)$. Two clustering runs were executed using HelioLinC yielding 278 potential objects with orbit-fits with less than 1 arcsecond astrometric residuals, and second yielding 536 potential objects. The results are discussed in greater detail in Section 5.

4.4 Completeness and peak detection efficiency estimation

Due to the extensive software development that was performed to accelerate and scale up KBMOD as was described in Smotherman et al. (2023), validation of correctness of execution is warranted. To validate the search results, the catalog of simulated objects is positionally crossmatched to the catalog of improved tracklet measurements. To allow for tracing of the results across filtering, two crossmatch catalogs are made. In the first catalog, we position-

ally crossmatch all retrieved results to the simulated object ephemeris we injected into the dataset. Because this set of results does not go through improved KBMOD fitting, there are no measured uncertainties. The crossmatch is performed by calculating the ephemeris of each result at the timestamps of the image collection it was detected in. If more than a set threshold, 50%, of these ephemeris are closer to the injected simulated object ephemeris than 1 arcsecond it is considered a match. This is considered a lossy crossmatch catalog, since for very low searched velocities it's possible to inadvertently match to the ephemeris of an injected simulated object, even when not accurately recovering the true direction of motion. The second crossmatch catalog looks at the improved KBMOD fit of tracklets to filtered results, for which we have uncertainties. To account for the uncertainty on the position of our improved likely results, the distance is calculated using Mahalanobis' distance (Mahalanobis, 1936) function. Given a probability distribution Q on $\mathbb{R}^N$, with a mean $\vec{\mu} = (\mu_1, \cdots \mu_N)$ and the point $\vec{x} = (x_1, \cdots x_N)$, Mahalanobis distance of the point $\vec{x}$ from the distribution Q is given as:

$$d_M(\vec{x}, Q) = \sqrt{(\vec{x} - \vec{\mu})^\mathsf{T} \mathbf{\Sigma}^{-1} (\vec{x} - \vec{\mu})} \tag{3.24}$$

where Σ is the covariance matrix. The probability distribution is taken to be a skew normal distribution centered on the position of a result at the earliest timestamp of the image collection it belongs to, $\mu = (\alpha_0, \delta_0)$, and with a $Cov(\alpha, \delta)$ calculated as described in Section 4.2. For numerical stability, before the covariance matrix is inverted, a 100 mili arcsecond (mas) covariance is added to the diagonal elements.

The retrieved cross-matched positions to inserted simulated objects are unbiased and no noticeable correlation between right ascension and declination exists (see Figure 3.10). The largest systematic offset is 30 mas with covariances in the range of 170-280 mas. The positions of likely results are, therefore, determined with an accuracy better than the size of a DECam pixel within a single night. By cross-matching the set of likely results using both crossmatch catalogs, out of 23,849 unique simulated orbits, we recover 7,678 and 7,278

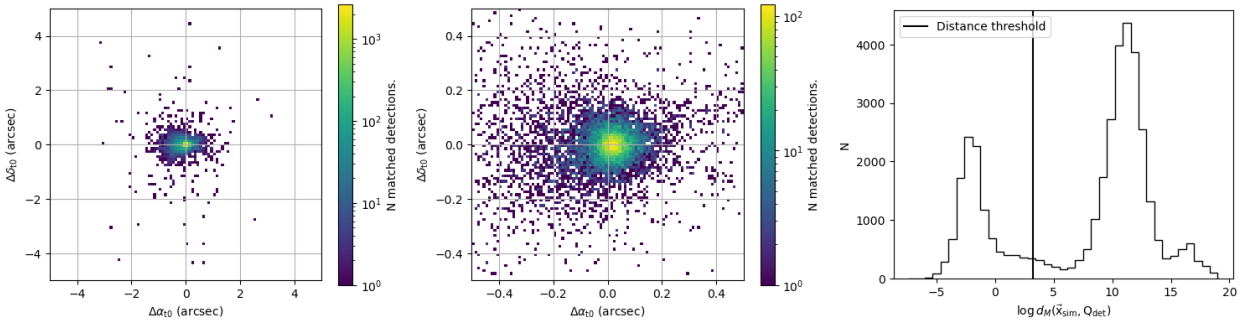


Figure 3.10: A 2D log-histogram of differences between recovered positions of results and positions of matching simulated objects, as measured at the first time stamp of each search collection. A result was considered to match a simulated object if the Mahalanobis distance value was smaller than 25. The right-hand side figure is a zoom into the center of the left figure showing how majority of the matches have the same initial coordinate positions as the matching simulated objects with only a minor systematic offset (30 mas) with respect to the right ascension.

unique simulated orbits for lossy and Mahalanobis cross-match catalogs respectively, thus determining the contamination rate of the lossy cross-match catalog to $\approx 5.2\%$.

Out of approximate 87 thousand likely results, a total of 12,428 likely results (i.e. single night tracklets, $N_{\text{tracklets}}$) are matched to a simulated orbit. Of that number some orbits are only matched by a single tracklet, while some orbits are matched several times. The number of unique orbits that are matched by at least n tracklets, $N_{\text{orbit}}(N_{\text{tracklets}} > n)$ are shown in the Figure 3.11. The majority of the orbits are matched by a single tracklet, and the number of matched tracklets per orbit drops precipitously for larger n. The miss-matched tracklets in the first, i.e. "lossy", crossmatch catalog originate predominantly from the set of only singly-matched tracklets.

Care must be taken when comparing the simulated orbits and matched results for several reasons. Firstly some properties of the injected simulated sources change between different observing nights. Magnitude of the object depends on its phase and current geocentric distance - which changes over time, especially for objects on orbits with large eccentricity. Secondly, not all simulated moving objects are detectable with KBMOD because not all data was searched, and a portion of the data with inserted objects may have failed to successfully

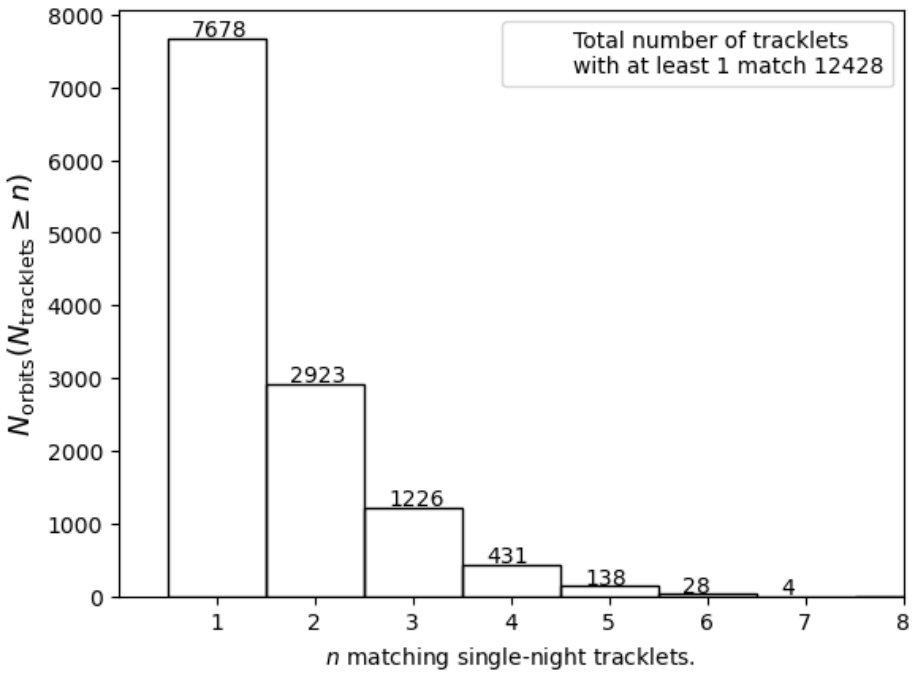


Figure 3.11: Histogram of the number of orbits matched to at least n tracklets. The number of tracklets within a linkage quickly falls off with the number of tracklets with only a few tracklets being linked more than 6 times, i.e. across the entire duration of the survey.

pass the Rubin Science Pipeline processing steps required to create the difference image to search through. Therefore a number of simulated objects exists outside of the footprint searched with KBMOD, skewing the population statistic. Thirdly, due to the choice of orbital parameters of the TNO populations used to create the the ephemeris and object properties that were injected into the images (see Section 3.1), not all objects would be visible even in the largest image collections. Due to their distance, size, phase etc, the magnitude of the object may be too faint to be detectable. For example some objects that were injected at orbits with semi-major axis exceeding 2000 au. Finally, the object's magnitude, rate of motion, topocentric and heliocentric distance vary over the almost 4 years time span covered by the survey as the relative position of the Earth, Sun and the object change. Therefore, comparing the properties of the results directly against the properties of all injected simulated objects as relative to each other is not always possible. In certain periods, the simulated object may not be detectable within an image collection, while it may be detectable in a

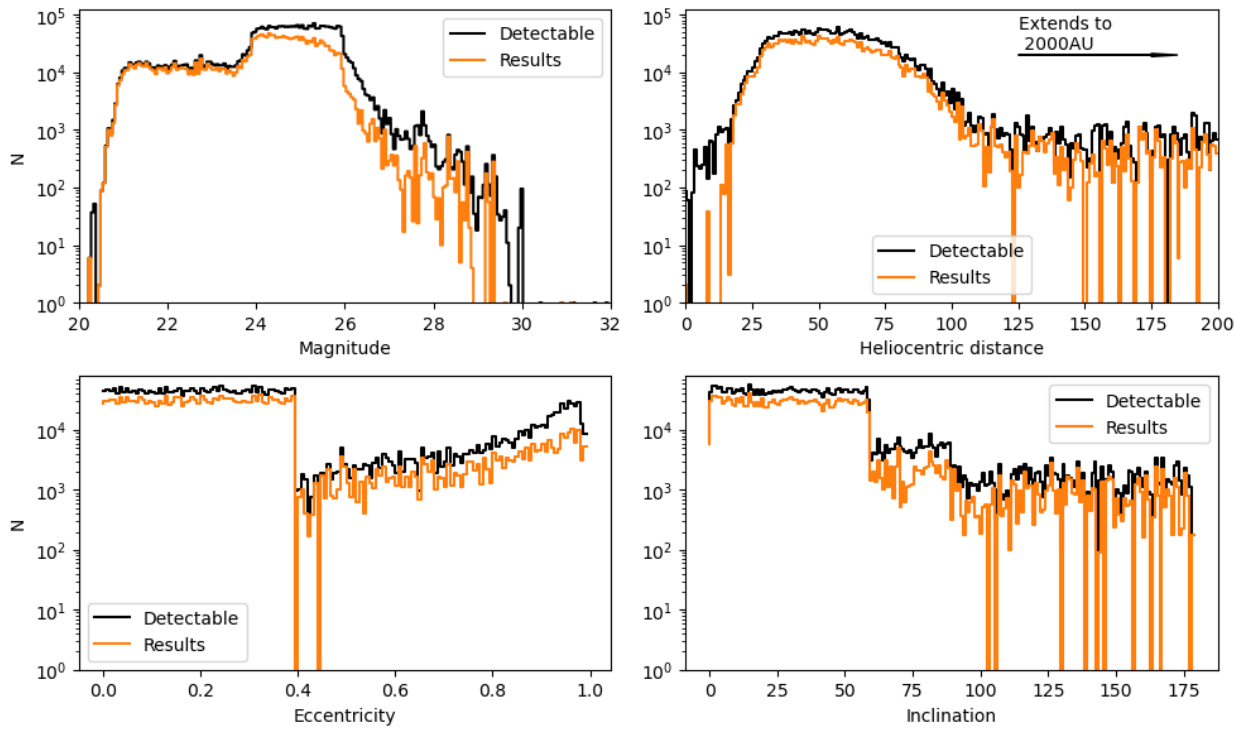


Figure 3.12: Histogram of the distribution of the mean values of per-night magnitude, heliocentric distance eccentricity and inclination of simulated objects matched to a result, in orange. In black are all simulated objects brighter than magnitude 26, binned and averaged in per-night sections as described in the text. Fewer objects are recovered at fainter magnitudes, as expected, and at very large inclinations. The apparent lower detection rates of simulated objects during a period of their orbit in which they were closer than $\approx$ 20 au from the Sun may be a consequence of a smaller number of such simulated objects or their non-linear motion. Overall, only small bias is shown across both orbital elements and object's properties for this relatively bright cut on magnitude where we expect the completion rates to be hight.

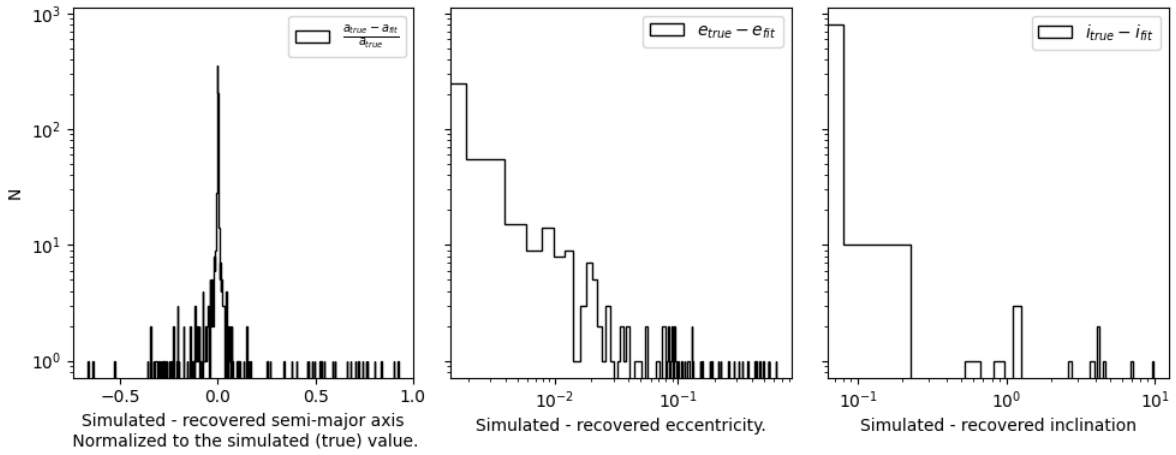
different period covered by a different image collection. These biases are taken into account by the survey completion statistic (see Figure 3.14 and Equation 3.4), but not in direct comparison.

For comparison purposes, the inserted simulated objects partitioned into per-night ephemeris and only those portions of the orbit that are present within the searched image collections are kept. For each visible portion of an orbit the mean values of the osculating elements and the object's properties, such as magnitude, phase etc. are calculated. The partitioned objects catalog is sub-selected to contain only those objects that were present within the searched image collections and are brighter than some threshold magnitude. In Figure 3.12 the properties of all detectable objects vs recovered objects are shown for simulated objects above magnitude 26. Strong bias is expected primarily from magnitude and heliocentric distance, as the distance sets the range of velocities an object's apparent motion is expected to exhibit. Simulated objects with very high inclinations ($> 60°$) also appear to be recovered less frequently than simulated objects with $i < 60°$. Given the large number of simulated objects with large inclinations this appears to be a real trend (see Figure 3.5). Qualitatively, visible simulated objects were not detected in great numbers either, at least during a search period in their orbit in which they were closer to the Sun than $\approx$ 20au. This could be due to increased non-linearity of their apparent motion, the fact their apparent velocity lays outside of the searched boundaries, but also because there are progressively fewer and fewer of such objects to make a determination with certainty.

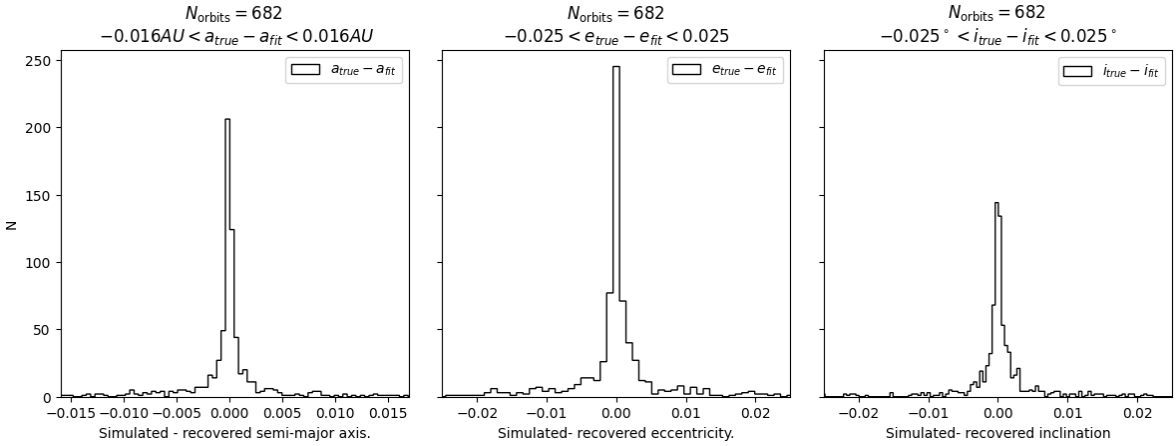
The top 1000 most often matched orbits are selected from the set of likely results in order to validate that orbit fitting recovers the expected inserted simulated orbit parameters. This corresponds to 527,047 simulated ephemeris, or 3,396 single-night tracklets, i.e. we expect to match an orbit at least 3 times (see Figure 3.11). The selection of likely results is then further sub-selected for only those results that have no nan values within their covariance matrix. Orbit fitting was performed using Bernstein & Khushalani (2000) code, treating each tracklet's start and end points as observations and validating the fit against the orbit

fit to the simulated ephemeris. The resulting orbit fits were filtered based on quality of fit and ad-hoc metrics ($\chi^2/\nu < 2$, $0 < e < 1$, $a > 0$ and $N_{\text{tracklets}} > 2$) and only those orbits that matched the conditions were kept. From the 1000 used orbits, 854 remain after goodness of fit cuts. There are 12 orbits that do not exhibit good fit accuracy, as shown in Figure 3.13a, defined as a recovered orbit with an inclination that is different from injected inclination by more than 1 degree or with a recovered semi-major axis that is different than the injected semi-major axis by more than 10%. For 682 orbits, the orbit fit is very accurate as shown on Figure 3.13b. Half of the selected simulated objects were matched by 3 tracklets, and approximately half were matched primarily by 4 and 5 tracklets. The longest matched orbits had 7 tracklets, spanning the full time-span of the DEEP survey (> 3 years). The mean and median length of arc that was observed, in years, are 1.896 and 2.032 years respectively.

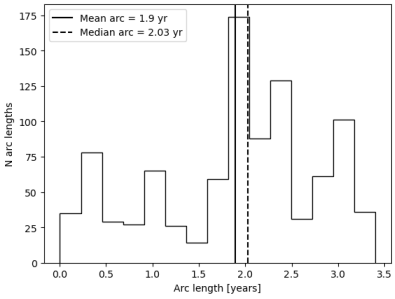
Finally, the last validation test that simulated objects can be used for is estimation of the peak efficiency and completeness rates. The completion of the survey was modeled as described in Section 3. Results are collated from per-night-per-patch into patches (A0a, A1a, B0a etc.) to minimize the effects data quality has on the search results and completeness was fitted per patch. These are shown on figure 3.14. All of the patches that have a 50% completeness magnitude $m_{50} < 25$ come from the A1 field, with the exception of a single patch A0g. Data quality metrics, F.e. sky noise, indicate that the first 12 highest measured sky noise levels belong to the A1 field. The measured magnitude at which 50% of the simulated objects are recovered throughout the survey is $m_{50} = 25.51$. The lowest limiting magnitudes achieved were measured for the B1 fields, where B1m had the lowest limiting magnitude achieved of $m_{50} = 26.1$.



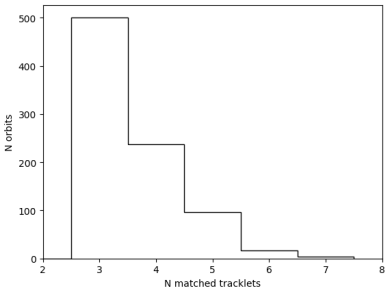
(a) Log plot of the 854 orbits remaining after goodness of fit cuts. There are 12 orbits for which the difference between the simulated inclination and recovered inclination is larger than 1 degree or for which the difference between the simulated and recovered semi-major axis exceeds 10% of the simulated semi-major axis.



(b) Zoomed in view to the difference between true and recovered orbital parameters for the best fit orbits ($a = a_{\text{true}} \pm 0.016au$, $e = e_{\text{true}} \pm 0.025$ and $i = i_{\text{true}} \pm 0.025°$.).



(c) Length of observed arc, in years.



(d) Most orbits are matched by 3 tracklets.

Figure 3.13: From left to right, differences between simulated object's orbital parameters and orbital parameters fitted to tracklets matched to the object, for semi-major axis, eccentricity, and inclination. On the bottom row, are the observed arclength, in years, and the number of tracklets matched to the top 1000 orbits. Predominantly, each orbit has 3 matched tracklets over a span of 2 years, resulting in a good orbit fit across 870 orbits within the sample.

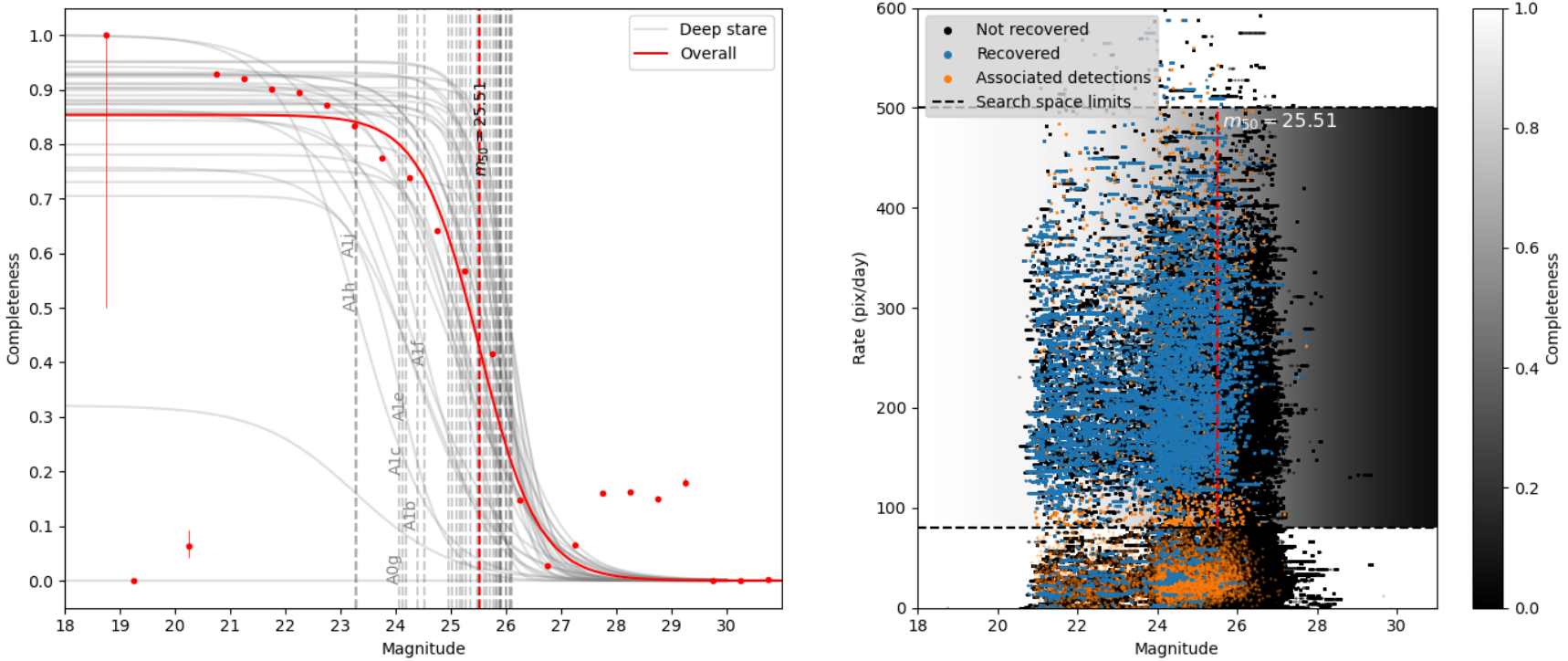


Figure 3.14: Survey completeness as a function of magnitude (top). Overall completeness across the entire survey is plotted in red, while completeness per sky-patch is shown in dashed blue. Survey completeness as a function of rate and magnitude (bottom). The rate of motion and magnitude of each simulated object was individually calculated for each searched night and plotted in black dots on the right-hand-side figure. Blue dots are the recovered crossmatched search results. The orange dots represent a portion of a orbit of an simulated object that was not detected directly, but could be associated with an orbit of a detected simulated object. The color gradient in the plot represents the survey completeness curve (top) within the searched parameter space. Some object can be detected outside of the searched velocity range, if the object is brighter and has a larger PSF FWHM on the image and if its apparent motion on the sky is close to linear. Whenever the length of the searched orbit arc is sufficiently long to pass the limiting number of observations for which there is sufficient signal to pass the likelihood threshold, the observation will be recorded.

Notable feature of the completeness is the excess 10-25% completeness rate for objects dimmer than 27th magnitude, associated with companion member of compact simulate binary moving objects. Thus these detections are not necessarily direct detection, but a product of cross-matching due to the inability to discern whether the search was matching on the primary or the secondary object of the binary. Another notable feature of the completeness is the sudden dip of the completeness at magnitude 20. This is an effect of low number of simulated objects that are this bright in the dataset, as can be seen on Figures 3.5 and 3.12.

Few image collections, which have very low recovery rate due to stellar density, large failure rate during processing of images, or due to low number of inserted simulated objects usually associated with low number of images within the collection making them unlikely candidates to receive a simulated moving object. The nights and stares with designation of **20190601_A1b**, **20210506_A1j**, **20210507_A1f**, **20210513_A1e**, **20210518_A1d**, **20210506_A0i** and **20210515_A1h** have less than 40%, and in the case of A1j, A01, A1f and A1d less than 20% recovery rate. On average 500 simulated moving objects are inserted per focal plane. The last three, **20210518_A1d**, **20210506_A0i** and **20210515_A1h** have less than 50 inserted simulated objects per the entire focal plane due to the very small number of images in that night, presumably due to weather. The first four nights (A1b, A1j, A1f and A1e) have a low completion rates, 20 to 40%, but no strong correlation of the recovery rate could be found with the number of images, PSF FWHM, sky background counts or sky noise across the survey. The strongest correlation was found between measured sky noise and recovery rates, but a linear fit to the trend shown only a $R^2 = 0.454$ (see Figure 3.15).

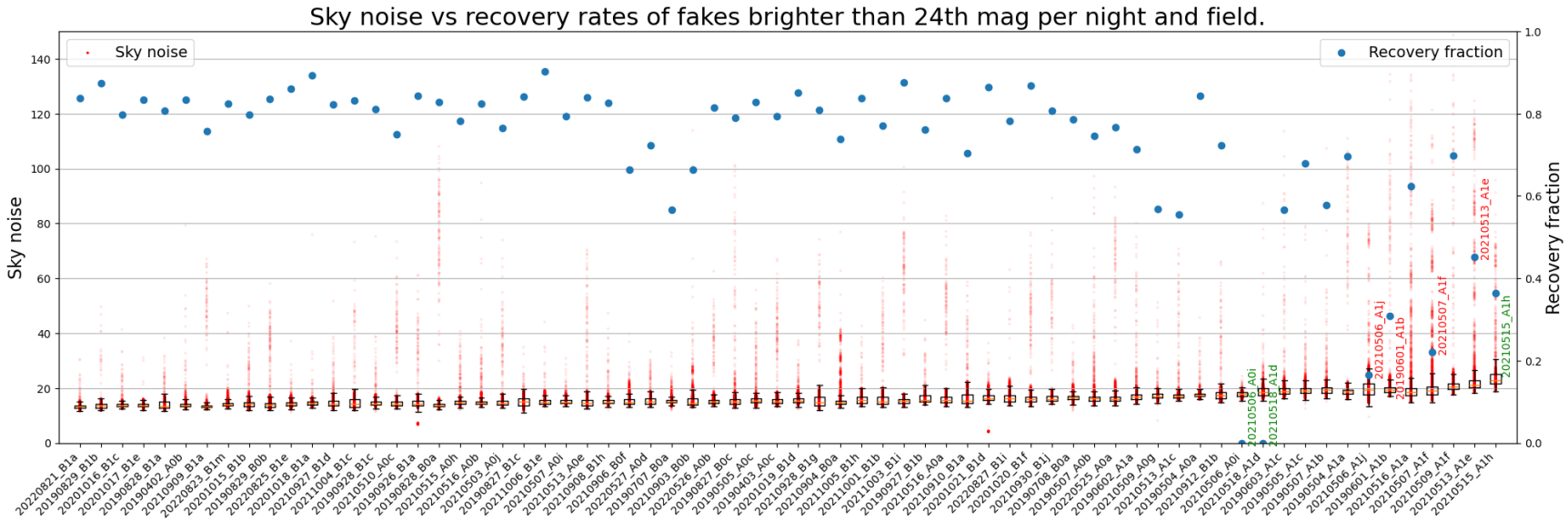
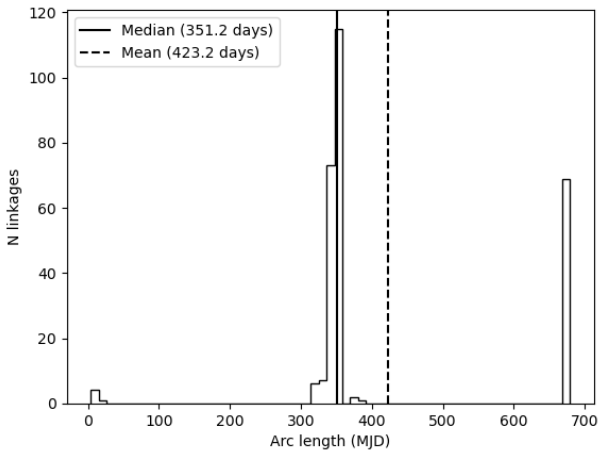


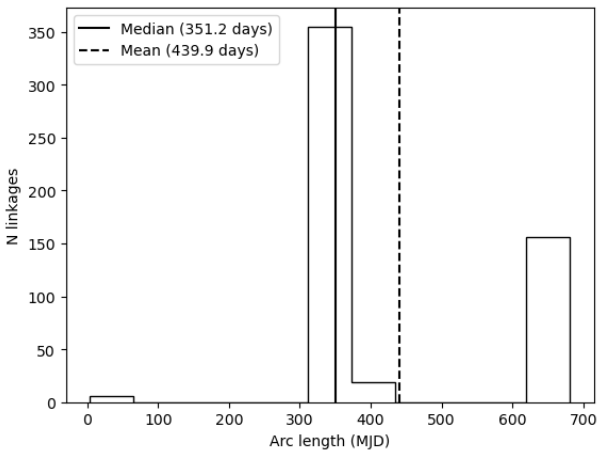
Figure 3.15: Sky noise values, as reported by Rubin Science Pipelines, plotted on the left-hand side axis against recovery rates per image collection on the right-hand-side axis. In green are the three image collections identified to contain very low numbers of inserted simulated objects, and in red are the names of four image collections that have unusually low completion rates. All sky noise values over 500 (out of the upper limits of this plot) occur for the last 4 image collections. The values of measured sky noise for the image collection are plotted with red points with the boxplot indicating mean, upper and lower quantiles of their distribution. Blue dots are the recovery fraction of detectable simulated objects, defined to be brighter than 24th magnitude in this plot to guarantee good recovery efficiency and minimize noise when calculating recovery fraction. A similar, albeit weak, trend can be noticed with PSF FWHM, and sky background counts.

5 Preliminary results

Preliminary results consist of linkages obtained from two HelioLinC runs. The first run tested 29,567 hypotheses from 20 to 100 au and found 278 potential objects that had orbit fits with astrometric residuals less than 1 arcsecond. The chosen clustering radius was $2 \cdot 10^5 km$ ($1.34 \cdot 10^{-3} au$). The clustering radius is scaled by geocentric distance relative to 1 au, so at 40 au the nominal clustering radius is $1.6 \cdot 10^7 km$ ($0.1 au$). Most of found linkages were seen on three nights, 18 were seen on four nights and 2 were seen on five nights. Nearly all (273 out of 278) had 1-year or longer arcs, and 65 had 2-year arcs (see figures 3.16a and 3.16b). The second HelioLinC run tested a much finer grid of 343,792 hypotheses, from 20 to 100 au and found 536 potential objects that had orbit fits with less than 1 arcsecond astrometric residuals. The clustering radius is the same and the arcs of recovered objects remained nearly identical.



(a) Length of observed arcs, in days, for the 278 objects in the first run of HelioLinC.



(b) Length of observed arcs, in days, for the 536 objects in the second run of HelioLinC.

Figure 3.16: Arc lengths, in days, of recovered linkages in both HelioLinC runs.

The unique ids of the first tracklet of the linkage was propagated through the linking process in order to cross-match the linkages to injected simulated objects. For details on the quality and validation of the positional cross-matching see Section 4. In the first run of HelioLinC, 228 potential objects (80%) were were matched to an injected simulated object and, in the second run, 425 (79.3%) potential objects were matched to a known injected simulated object. The remaining 50 and 111 potential objects from both runs were positionally matched, at every linked ephemeris, to known TNO objects. To retrieve the list of known objects at the linked ephemeris, the Sky Body Tracker (SkyBOT) service (Berthier et al., 2006) was used. A circle with 5 arc second radius was queried for known objects, for each ephemeris of each linkage at the epoch of observation. A threshold of 70% of matched ephemeris was used to filter out any potential objects that could match a known object. No potential objects, remaining after the filtering, had a single detected ephemeris within 5 arcseconds of a known object.

To the remaining objects that do not match any injected simulated objects, or known real objects found with SkyBOT, an orbit was fitted using Bernstein & Khushalani (2000) orbit fitting code, as described in section 4.3, assuming a homoskedastic errors of 0.1 arcseconds on each ephemeris. The linking astrometric root mean square (RMS) error, and the χ^2 per degree of freedom (χ^2/ν) quality of fit metric were used to sub-select those objects that are statistically consistent with an the fitted orbit and across the HelioLinC linkage. There are 2 objects from the first HelioLinC run, and 13 objects from the second, which $\chi^2/\nu < 1$ and $\mathrm{RMS}_{\mathrm{astrometric}} < 0.1^{"}$ and are rejected as false linkages. The remaining 44 objects are the detections statistically consistent with a detection of new objects (see Figure 3.17). All potentially new objects consistent with the orbit of a newly detected object that were recovered in the first HelioLinC run were recovered in the second run. The parameters of their orbits are shown on Figure 3.18.

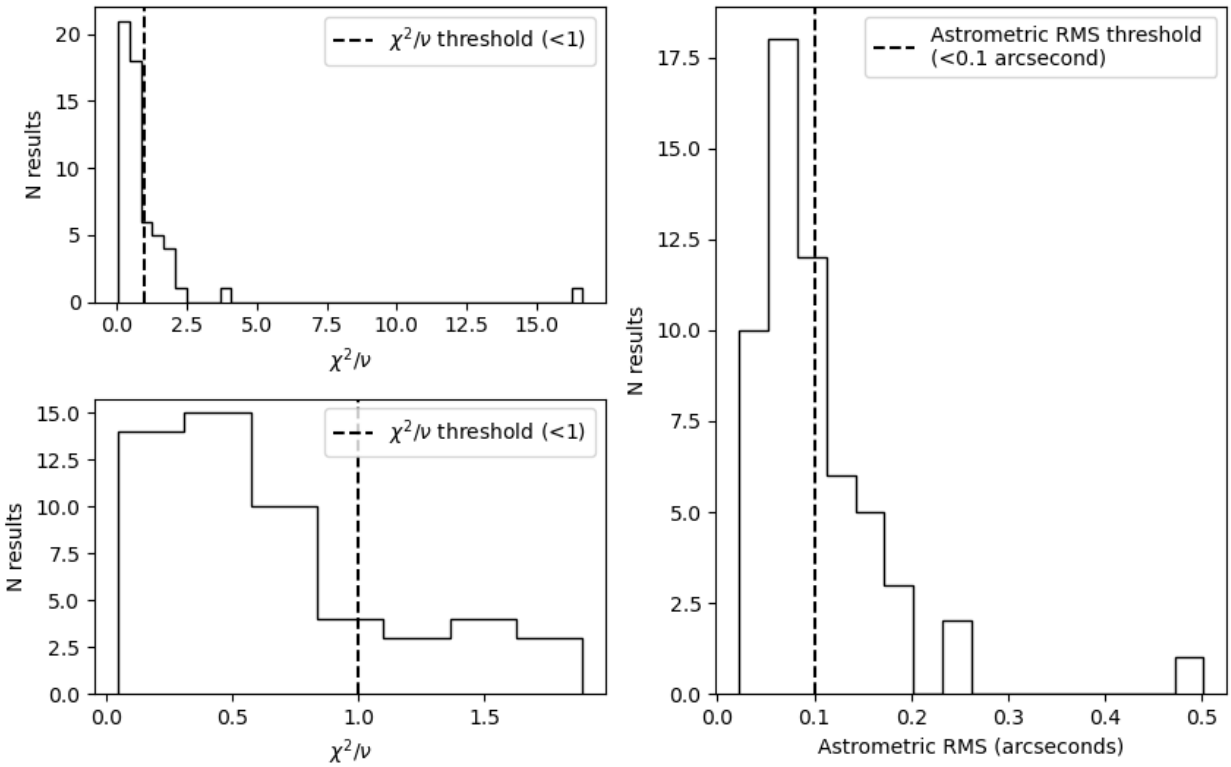


Figure 3.17: The reduced χ^2/ν (left) and astrometric RMS (right) of the potential objects that do not positionally match an injected simulated or the ephemeris of known real objects. The set of objects for which the $\chi^2/\nu < 1$, shown as dashed black line on left figures, and $\text{RMS}_{\text{astrometric}} < 0.1^{"}$, shown as dashed black line on the right hand side figure, are statistically consistent with the orbit of a newly detected object.

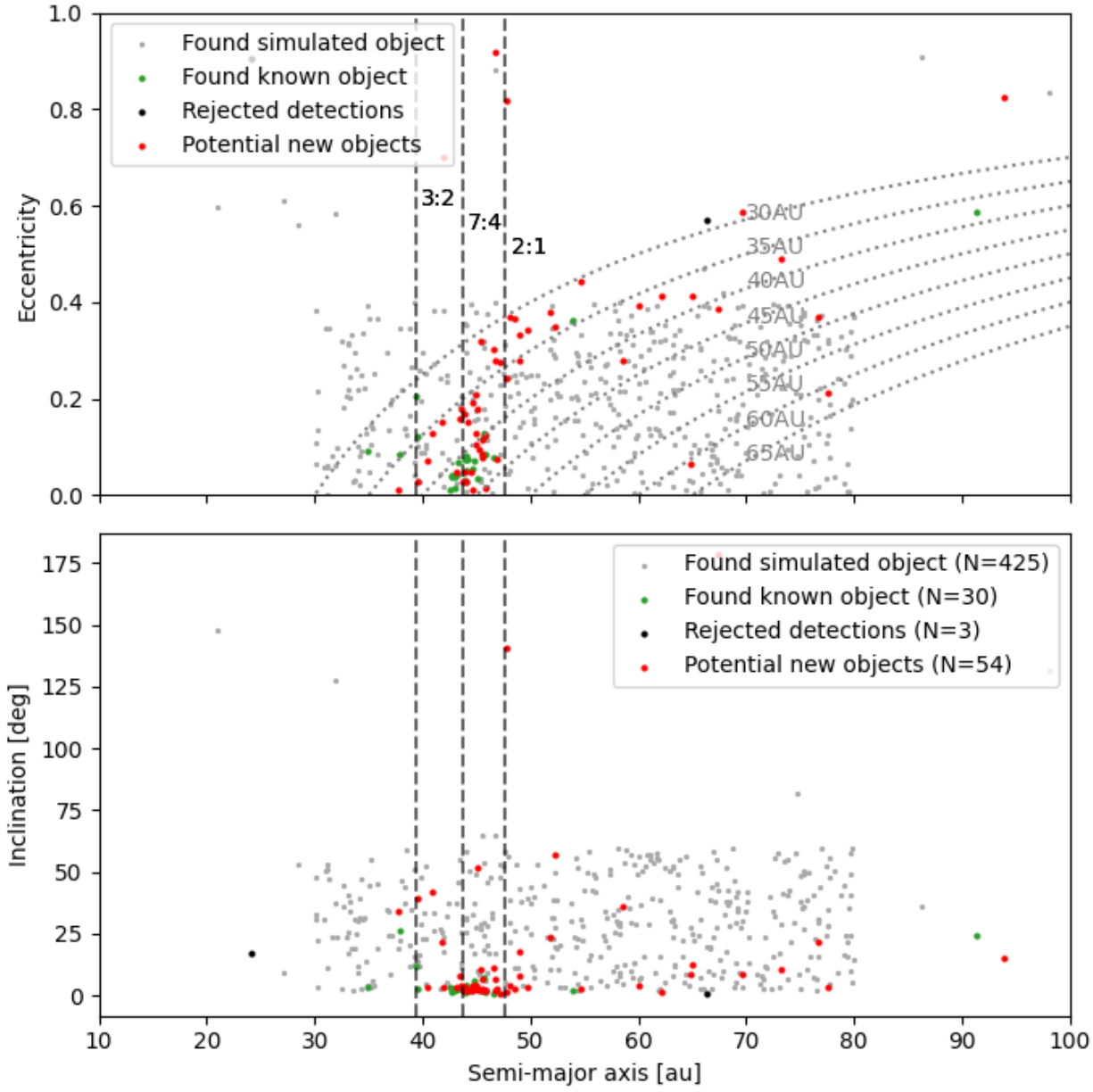


Figure 3.18: The orbital parameters of objects recovered from the linking procedure. The semi-major axis a versus eccentricity e (top) and the inclination i, in degrees on bottom figure, are shown for all recovered linkages matched to a injected simulated objet in light-gray, found known objects in orange, potential objects consistent with a newly detected orbit in red and objects rejected in the quality of fit filtering in black dots. The recovered orbital parameters are consistent with the distribution of injected simulated objects and the distribution of objects most statistically consistent with a new detected objects cluster around the expected distribution of Kuiper Belt objects.

6 Discussion

In Chapter 3 we discuss the search for new TNOs within the DEEP dataset using KBMOD. To achieve this goal we re-processed all of DEEP data using the Rubin science Pipelines weekly build w_2024_09 and made numerous changes to the KBMOD package. Key new additions to KBMOD development operations are the addition of a build, documentation, and a continuous integration systems. Key new software architecture changes to KBMOD are the addition of image collections and standardizers, refactoring of the core KBMOD search algorithm to allow pipelining, i.e., the addition of the work unit abstraction, and the development of an external package for workflow management based on Parsl called kbmod-workflows[6]. Numerous performance improvements and bug-fixes were implemented. Notably, the transition of the core C++ code to Eigen, a C++ template library for linear algebra: matrices, vectors, numerical solvers, and related algorithms, which better integrates with Pybind11, a library that allows interoperability between C++ and Python. The move to Eigen for core algorithm implementation allows us to pass NumPy arrays to C++ code (and vice-versa) as pointers, avoiding a memory-expensive copies. The addition of the work unit and workflow management now allows for 2 new search algorithm changes: the resampling of the images and the repeating of the search for each result. Resampling allows KBMOD to search through upcoming Rubin observations which will not be rotated and resampled to a common sky map, and to correct for the reflex motion of the Earth, which is discussed in more details in two following Sections 6.1 and 6.1. Repeated searches allow us to revisit the found results and perform repeated, significantly denser, pencil-shape searches around the initial result and derive improved positional and velocity measurements and uncertainties.

The processing of all of the DEEP data, both with Rubin Science Pipelines and KBMOD, is made difficult by the lack of dedicated resources. On Klone the primary source of large compute and storage resources is achieved by accessing the *checkpoint* queue, which is a queue

[6] https://github.com/dirac-institute/kbmod-wf

that allows access to all currently unused resources, but that resource will be re-allocated to users who own them upon their request, requiring fault tolerance and resulting in higher rates of failures. Logging and log analysis tools for both KBMOD and Rubin Science Pipelines are critical to achieving a high processing and analysis success rate. Changes to KBMOD code described above seem to have had a marked improvement on performance, compared to Smotherman et al. (2023). With the current Rubin observing cadence, excluding the DDF, we expect to have to process, with KBMOD only, a comparable volume of data processed for this work on an monthly basis. The time to results is currently dominated by the initial search by KBMOD and filtering of the linkages into orbits deemed likely to be real because it cannot be easily automated.

The completion can be measured for datasets like DEEP by as described in Bernardinelli et al. (2024), Smotherman et al. (2023) or Bernardinelli et al. (2022) and briefly reviewed in Section 4.3. However, data processed by Rubin will not insert any fake objects in their images. Inserting fake objects directly into difference exposures is only possible under the assumption that the coadd itself contains no signal from the object. Otherwise the behaviour observed in Section 3.2 is not reproduced. An object's signal will not remain within the coadd only in scenarios in which there are a sufficient number of images, with a long enough time span between individual images, and over sufficiently long total time span of the entire dataset. The required period of time is also a function of the object's apparent rate of motion, ultimately, the data must enable the coadd statistic to converge to an unbiased value of the background. Creating a bespoke coadd for each considered set of image collections spanning the same time may be prevented by the computing and storage requirements to process and store all these additional coadds. The recovery completion may also be tracked in real time by comparing to known detections and detections of moving objects made by other approaches or via dedicated follow-up of potential detections.

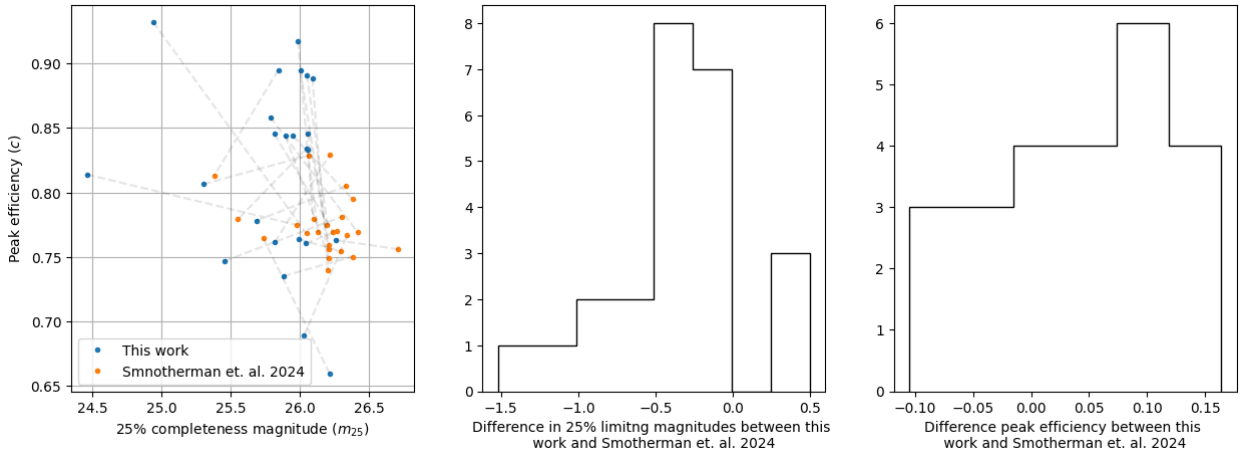


Figure 3.19: Comparison between peak efficiency c and 25% limiting magnitudes m_{25} (see equation 3.4) between this work and Smotherman et al. (2023) for the subset of fields searched in both processing. The overall number of newly discovered object is lower, due to higher filtering likelihood threshold resulting in lower limiting magnitude, although the peak achieved efficiency of detection is higher.

Our results show no strong unexpected biases and have acceptable completion rates. The m_{50} completion magnitude is ≈ 0.5 magnitude less than the previous processing by Smotherman et al. (2023). A 10% higher peak efficiency is achieved for the fields previously searched, see Figure 3.19. Total number of recovered objects will be smaller due to the significantly lower limiting magnitude of the search. The current search should return $n/n_{\text{previous}} \approx 1.1 \cdot 10^{-0.3} \approx 77\%$ (Equation 3.4). The likelihood threshold and n_{obs} is too high in this processing as is evident by the low number of results, to no results, being recorded for the smallest ≈ 2000 searched image collections, resulting in the higher m_{50} and m_{25} completion magnitude. The number of images required to be above the threshold was set to half the number of images in the searched collection, or 15, whichever was larger, and the likelihood threshold was set to 7 for the whole search, which is higher than the previous search. The searched linear trajectories in the image stretched $(-90°, 90°)$ around the ecliptic, and the spanned range of velocities was $(80, 500)$pixels/day. The performance improvements and the pipelining of the execution enabled a doubling in the searched angle parameter space (from $(-45°, 45°)$) and $\approx 38\%$ increase in the span of the searched velocity range, over roughly 10 times as much data.

Due to concerns about the potential number of returned results, as extrapolated from previous processing, the filtering thresholds were set higher than before. It is worthwhile to reprocess the same dataset with lower thresholds in order to retrieve a larger number of potentially new objects within the fields that were not previously searched. The DEEP data is also being reprocessed again in such a way as to include all overlapping images across all 4 years of observations in order to create coadds that do not contain the residual signal of the object itself within them. This was not done in the first processing of DEEP because such jobs require a large memory over an extended period of time, and increase the preemption rates when processed on Klone. The new reprocessing is being performed on the STAMPEDE cluster and the coadds are partly being created on Epyc, thus working around the issue at a cost of a significantly larger total processing time.

The preliminary results are derived from linkages produced by HelioLinC. Approximately 80% of recovered linkages are injected simulated objects, which further validates the correctness of processing and analysis. Of the remaining 20% of objects, approximately 50% and 30%, for the less and more dense hypothesis sampling runs of HelioLinC respectively, are known objects. The total number of recovered known objects between the two HelioLinC runs are similar (22 and 30) which is consistent with the fact that the underlying dataset is constant, therefore doubling of the number of real results is not expected. The number of results that do not match any injected simulated object or known real object (26 and 58 respectively) and the number of results that do match an injected simulated object (228 and 425 respectively) roughly doubles between the two runs of linking indicating the importance of a well sampled grid of scaled distance γ and its rate of change $\dot{\gamma}$ to successful recovery of objects. The choice of HelioLiNC hypothesis parameters overlaps the densest portion of orbital space covered by injected simulated objects (see figure 3.5), but they should certainly be extended to cover a broader range of test distances. The chosen clustering parameters, linking astrometric RMS and χ^2/ν are, we believe, very strict to minimize the number of potential spurious detections within the results.

There are 44 detected potential objects that are statistically consistent with a detection of a new object. The number increases to 54 detected objects consistent with the detection of a new real object if the selected threshold on astrometric RMS is set to $\text{RMS}_{\text{astrometric}} < 0.2"$ instead. This change results in the majority of the rejected detections, black dots on figure 3.18, including the two highly eccentric objects ($e > 0.6$), to be considered real detections in our analysis. More analysis should be performed in order to conclusively include or exclude these results as statistically significant. This can be achieved by performing linking as described by Bernardinelli et al. (2022) and cross referencing the retrieved linkages, performing force photometry measurements on the positions within the images containing the object and by visual inspection of the moving coadds and cutouts and is part of the ongoing efforts. The recovered injected simulated objects are consistent with the population of injected simulated orbits (see figure 3.5) and the recovered known and potentially new objects are consistent with the results from Smotherman et al. (2023). The orbits fitted to the recovered results assumed a 0.1" homoskedastic errors on each ephemeris which seems to slightly overestimate the errors as evident by the skewed χ^2/ν in Figure 3.17. The length of observed arcs of the recovered objects are predominantly 1 year long, with approximately 50% of objects having 2 years long observed arc (see figures 3.16a and 3.16b, which is shorter than, but remains consistent with, the average length of observed arcs as recovered by positional crossmatching of all likely results to known injected fakes (see figure 3.13c). The apparent bias towards 1 year long observation arcs is the consequence of the accumulation of error during orbit propagation of "arrows" before the clustering step in HelioLinC as described in the section 4.3.

Further detailed analysis of the data is required in order to construct the final set of objects consistent with new object discovery, primarily by completing the analysis of the linkages retrieved via Bernardinelli et al. (2022) and by propagating derived positional uncertainties through the orbit fitting code as described in the section 4.3 in order to better constrain the quality of fit metrics.

6.1 Longer time-base searches.

There are two practical advantages to enabling longer time-base searches. Firstly, it is advantageous to search for longer tracks using KBMOD. All search trajectories described so far represent a relatively short trajectory on the image itself. This is a consequence of the relatively short time-base of the observing cadence, where majority of the images are collected in a period of 4 to 7 hours in a single night. The negative side-effect of this fact is the amount of effort that has to be invested in filtering of the results. While the image differences produced by the Rubin Science Pipelines are of very high quality, artifacts of poor PSF matching for very dim and very bright objects remain in the images. Such artifacts are very localized phenomena on the image. Therefore, when searching for longer trajectories over the image, the core KBMOD algorithm would filter these spurious artifacts as false positive detections by design.

The second benefit is the relaxation of the requirements of the observing cadence. A good dataset candidate for KBMOD has a large number of images overlapping the same, or similar, area of the sky in a relatively short period of time. However, a majority of the datasets are created by observing cadences that prefer a larger sky area coverage instead. LSST will be one such survey, where an area observed on a particular night, will be re-observed on average 3 days later. The key limitation of KBMOD searches described so far is the time-period in which the orbit remains sufficiently linear. By the time an appreciable number of visits occur for an area on the sky, the objects apparent motion on the sky would already be non-linear.

Finding solutions to the problem of linearization of the apparent motion of the object on the image is a key issue for KBMOD. There are two key questions that present themselves: a) how non-linear can an orbit be before KBMOD is unable to detect it and b) how can the apparent motion of an object be linearized over a longer time-period.

Non-linearity threshold

To test how far away from the center coordinates of the object on the a series of tests executed. The tests consisted of creating a noise-less image in which a single object was added as a Gaussian with an amplitude 100 and standard deviation of 1.3. The object was rendered onto an image following a circular path and spaced linearly equally apart on the path. Note that this implies the paths with smaller radii have greater separation between the points, compared to those with larger radii, and a slightly irregularly spaced points on the image but equal separation of points on the length of the arc of the path. The change in the spacing, however, is sufficiently small not to cause concern. A KBMOD search was executed on the mock collection of images and all of the results were kept.

A Euclidean and a Mahalonobis distance are measured for every point along every trajectory. For each point along the trajectory we can also assign a "detected" and "not detected" grade based on an arbitrary cutoff on either the likelihood or the lightcurve. The distance at which half of the trajectory points are not detected and half of them are, we call the 50% detection efficiency limit η_{50}.

For a threshold cut-off of 10% of the inserted amplitude, the measured distance at which the number of marked detected and not-detected match is $\eta_{50}^{\text{Euclidean}} = 3.5$ pixels, corresponding to $\eta_{50}^{\text{Mahalonobis}} = 4.5$. In Section 3 the limiting condition for the grid search parameters required that the end points of two neighboring search trajectories are no more than 5 pixels apart in the largest image collection, i.e. containing the largest number of images. The values were calculated by using an idealized mean values of the full focal plane pixelscale and PSF diameter. The results described in this chapter further motivates this limit because they implies that limit of detection has a diameter of ≈6 pixels around the true position of the object.

However, the test cases are also idealized by: a) the choice of a very low cutoff of 10% of the inserted amplitude and b) noiseless images. Repeating the tests with a larger cutoff of 50% of the inserted amplitude, we find the $\eta_{50}^{\text{Mahalonobis}} = 2.5$ and $\eta_{50}^{\text{Euclidean}} = 2$ pixels.

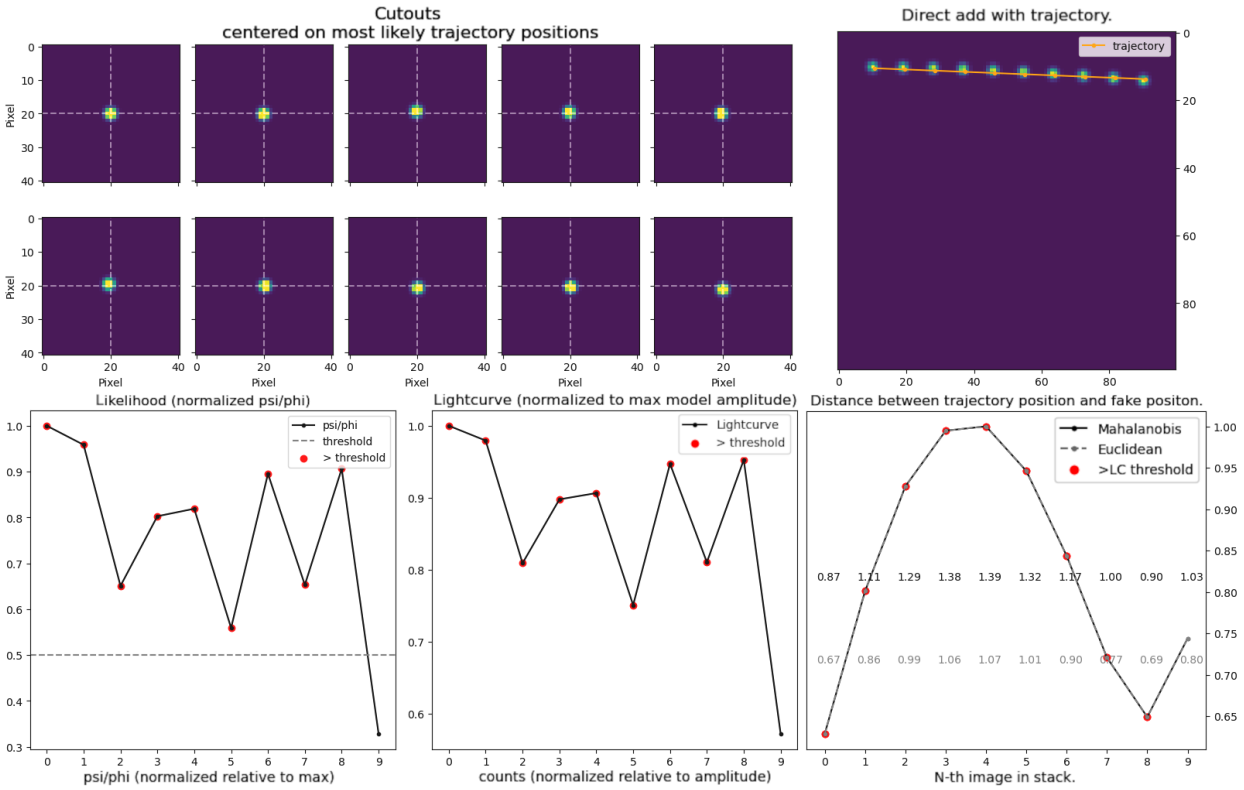


Figure 3.20: A single test from a set of tests described above. Top-left are individual cutouts in each of the simulated images centered on the best-fitting trajectory shown on the top-right in orange. Top-right is a direct sum of all simulated images position of the object as a single image and the best matching trajectory (orange) returned by KBMOD. On the bottom are, from left to right, the likelihood tracked by KBMOD, the lightcurve (i.e. the pixel value at the searched pixel), and finally the distance to the correct known position. The likelihood is a weighted lightcurve in this scenario due to the lack of noise in the simulated data.

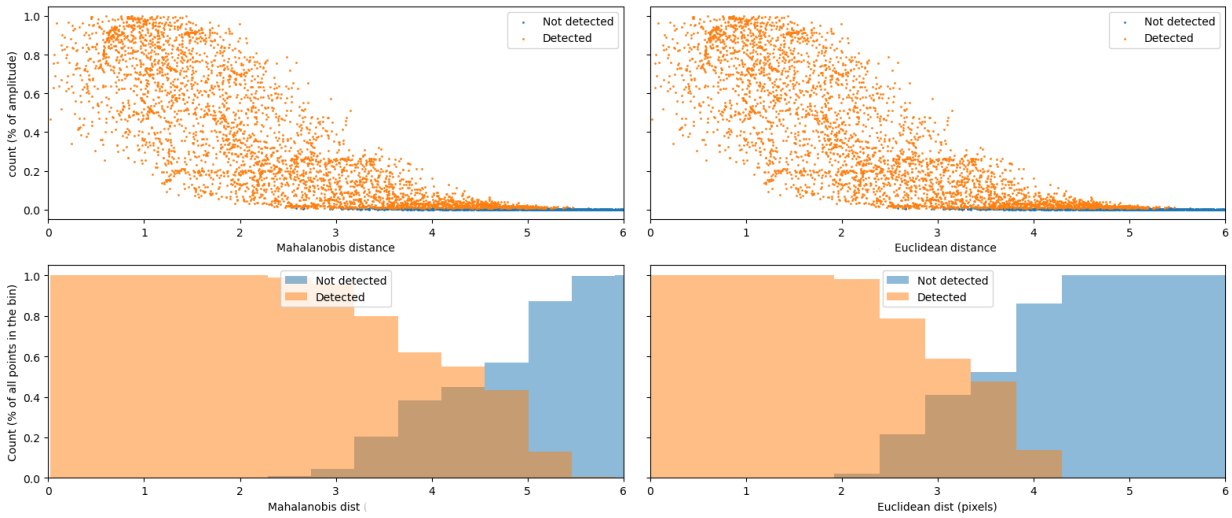


Figure 3.21: Mahalanobis and Euclidean distances for each trajectory point returned by the tests. In orange color are those test points that pass the likelihood test

The larger cutoff serve as a proxy for more noisy data, or lower signal objects due to the time requirement of executing this many searches on realistic data. The limiting condition of 5 pixels falls under this limit, however, the condition is applied to the largest image collection (121 images). Significant portion of the collections contain less than 100 images, and generally the trajectories are sampled at a greater density through most of the images within a collection.

Observing the detection efficiency η as a function of the radius of the circle on which the object was moving on yields the expected results. Smaller circles "bend more sharply" and therefore are less well-approximated by a tangent through any given point. Large circles "bend" less so a tangent remains a reasonable approximation for an longer length of arc of that circle. Therefore, the larger the radii, the larger the detection efficiency η we expect.

The measure of how much a circle "bends" is given by curvature. Intuitively, curvature measures the amount by which a curve deviates from being a straight line or by which a surface deviates from being a plane. A circle is the canonical example of a curve of constant curvature, the curvature of a straight line is zero, and the curvature at an arbitrary point of a differentiable (smooth) function is a scalar and it varies depending on the chosen point. Because of this, to make the curvatures between two different functions comparable, it is

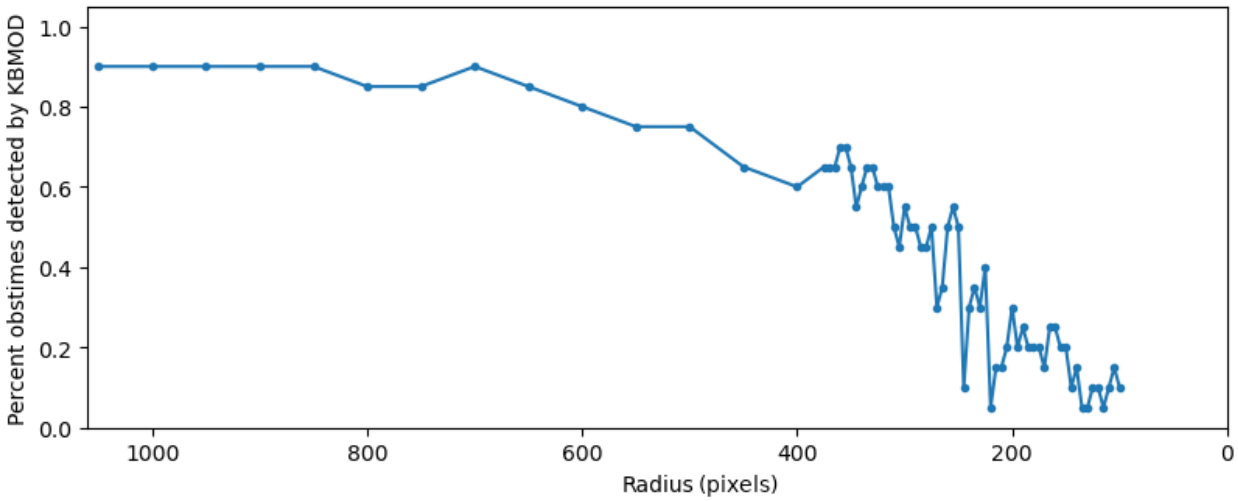


Figure 3.22: The detection efficiency η as a function of the radius of the circle on which the object was moving, for threshold cutoff set to 50%. The η_{50} is achieved for circles of radius $\approx$ 350-375 pixels.

often useful to parameterize two curves in the same way.

A useful way to compare the curvatures of different curves is to parameterize them in terms of their arc-length. Arc length is the distance between two points along a segment of a curve, i.e. from an arbitrary starting point following the graph of the function, to some final point also on the graph of the function. Put explicitly, we are performing the following substitution:

$$y = f(x) \quad \rightarrow \quad x = f(s) \tag{3.25}$$

$$g(s) = (x(s), y(s)), \qquad s \in [a, b], \tag{3.26}$$

where s is some real number within the arbitrary range $[a, b]$. We can then define curvature $\mathcal{K}$

$$\mathcal{K} = \left\| \frac{d\boldsymbol{T}}{ds} \right\|, \tag{3.27}$$

where $\boldsymbol{T} = (x'(s), y'(s))$ the tangent vector at some distance along the segment s of the curve.

Curvature, defined in this way, can be interpreted as the rate of change of the tangent as a function of the arc length of that function. Often it is convenient to require that $[a, b] = [0, 1]$ such that $\|g'(s)\| = \sqrt{x'(s)^2 + y'(s)^2}$ since then $\boldsymbol{T}$ will be a unit tangent vector. All graphs of a function are form of a parametrized curve given by $x = t$ and $y = f(t)$ so we can perform the differentiation of $\boldsymbol{T}$ in a general sense:

$$\mathcal{K} = \frac{|\ddot{y}|}{(1 + \dot{y}^2)^{\frac{3}{2}}}. \tag{3.28}$$

The curvature of a circle $y^2 + x^2 = r^2 \equiv r\cos(s), r\sin(s)$ is a constant and is given as the inverse of its radius $\mathcal{K}_{\text{circ}} = 1/r$. The motion of objects in the sky, however, are not curves of constant curvature. As mentioned before, the velocity of the object itself changes, depending on its eccentricity and barycentric distance, and its apparent motion on the sky changes depending on the positions of the Sun and the Earth relative to the object. Ephemeris of a well known object on an low eccentricity orbit can be generated to a high accuracy using JPL Ephemeris service. We select a, relatively, nearby object at 10 au distance and generate ephemeris in total duration of ≈2 years, with a 7h time-step, so that at least 1 full of the object's apparent retrograde motion is visible.

Using the tests described above, and ensuring the orbit of the object and the generated circles are equally parameterized as otherwise the absolute values are not directly comparable, we can calculate that 50% detection efficiency occurs for circles of curvature 3.44. Performing the same derivations as above, except numerically over the ephemeris of the moving object, we measure the curvature of an real orbit. Selecting sections of the curve whose curvature is less than the curvature at which 50% detection efficiency occurs, i.e. < 3.44, allows us to see the segments of the object's orbit in which the object would theoretically be detectable. Since the ephemeris were generated with a timestep of ≈ 7h across the observed period, and due to the numerical differentiation the curvature is measured within a 14h window centered on the point. In other words, the portion of the orbit that is sufficiently linear to be detectable by KBMOD is the local segment around the ephemeris in question, not with respect to

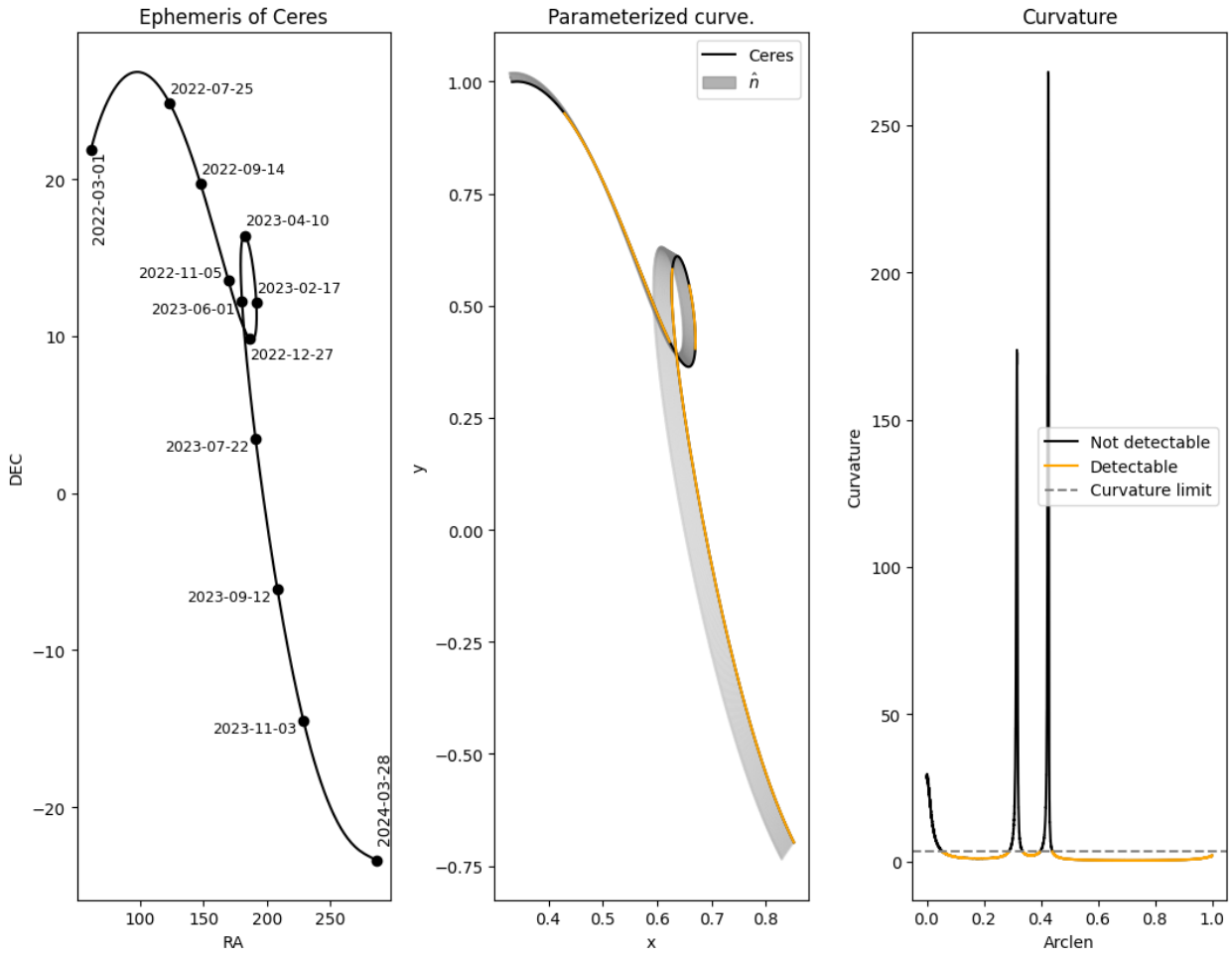


Figure 3.23: The apparent positions of object Ceres from March 2022 to March 2024 as returned by JPL Ephemeris service. The portion of the curve deemed detectable, i.e. which local curvature is less than the curvature of the circle at which 50% detection efficiency was achieved, are marked in orange. The calculated curvature is on the right.

the entire segment of the curve marked as "detectable". This is the core issue with this approach. Re-parameterizations, i.e. a different cadence of sampling of the positions are not directly comparable. Thus non-equally sampled ephemeris need to be fitted by interpolating polynomials, in practice the interpolated B spline polynomials with a generous smoothing factor proved the most successful, and re-sampled at the same times. The success rates of this process vary depending on the density and smoothness of the ephemeris.

Interestingly, as the object's apparent velocity on the sky varies, depending on the object's orbit, its current true anomaly, and as its relative position to the observer changes, the time spent in each "detectable" segment varies. If we calculate the total time we've observed this object, ≈757.75 days, and add up the time the object spent within the segments marked as "detectable", ≈387 days, we find that the object was detectable approximately 51% of the observed time.

Correction of the reflex motion of the Earth

Parallax is the apparent change in the position of the object, at a distance that is closer to the background against which its position is measured. The closer the object and the larger the change in position of the observer the larger the parallax. In the case of two orbits of the same eccentricity, the object on the orbit with smaller semi-major orbit will orbit faster, i.e. with a higher frequency. Due to the parallax effect, the apparent position and motion of the slower orbiting body changes as the faster body approaches the slower orbiting body. The apparent motion of the slower body becomes retrograde and then, once the faster body has overtaken the slower one, prograde again. This effect is referred to as the reflex motion of the Earth and is the cause of loops in ephemeris of celestial bodies within the Solar System (see Figure 3.23). The existence of this parallax allows the measurements of distance to sufficiently close objects (Adolphson et al., 2015; Bailer-Jones et al., 2021; Singal, 2015). For KBMOD, however, the existence of parallax presents a problem, as it makes the apparent motion of these objects less linear over time.

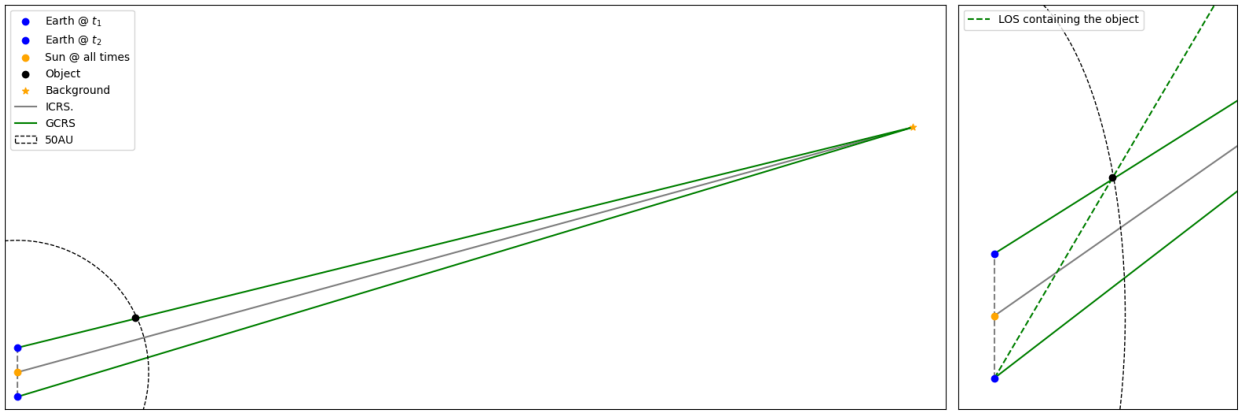


Figure 3.24: A sketch of how the apparent change of position, as measured against stationary background objects, comes about with the change of observers position. At t_1 the line of sight (LOS) containing the object, in green, is resolved to a position (α, δ) relative to the background objects. At time t_2 that same LOS does not contain the object anymore due to the parallax effect. Without a priori knowledge of the distance to the object, it is not possible to correct for this parallax. The background objects are sufficiently far away, such that the LOS to them from all three positions (Earth at t_1, the Sun and finally Earth at t_2), are approximately parallel as demonstrated in the right-hand-side panel. For a sufficiently distance object the three lines of sight would be so close to parallel that the error in the sky coordinate (α, δ) they are pointing to could not be measured.

An object on a circular orbit with a radius of 100 au would have a period of approximately 1000 years and an angular velocity of ≈3.5 arcseconds per day. Such an object would move approximately an arcminute a month. To estimate the order of magnitude of the reflex motion of the Earth we can estimate the magnitude of the parallax under which we would see an object at a distance of 100 au. By definition, the Earth crosses the fundamental plane of the Solar System during equinoxes. A stationary point lying perpendicular to the line connecting the position of the Earth during Vernal and Autumnal equinoxes, would, again - by definition, have International Celestial Reference System (ICRS) coordinates of $(90, 23.45)$. The point then forms an isosceles triangle with a base equal to the diameter of Earth's orbit and the height equal to the distance of the object from the barycenter. The parallax is then given as $2\tan(1/d)$. At a distance of $100au$, the parallax is $\approx 1.1°$; demonstrating how reflex motion of the Earth dominates the apparent motion of distant Solar System objects.

The key idea here is that the apparent motion of the object on the sky can be linearized for a greater period of time if the reflex motion of the Earth could be compensated for by, for example, appropriately shifting the images in the collection. This is not possible without a priori guessing the distance to the object. Given we are searching for unknown objects of unknown distances, the dimensions of KBMOD's search parameter space has to be extended by an additional parameter - the guess distance.

The challenge in removing the parallax is the fact that the reported coordinates of the footprint of the image are expressed in ICRS coordinates. Thus the parallax is implicit within the WCS of the image, since the origin of ICRS is the barycenter of the Solar System. The given WCS has to be translated to the origin at the time of observation, fundamentally a translation by the position of the observatory from the geocentric coordinate system origin. The guess distance to the object is expressed in terms of distance from the barycenter of the Solar System, as doing otherwise would be impractical. Therefore, a new, topocentric, distance of the object to the new origin has to be calculated. The new topocentric coordinates and distance can then be transformed back to the ICRS system in a manner that accounts for the parallax. The WCS can then be corrected for the effect of the parallax.

Assuming a single barycentric guess distance and performing this calculation for every exposure in the dataset would then yield coordinates of the footprint of each exposure due to the fact that the orbit and rotation of the Earth changes with time. Two key challenges to the problem are acquiring the precise position of the observatory at every time-stamp at which an observation was made in the dataset, and a fast calculation of the new topocentric guess distance. Thankfully, due to ephemeris generators such as JPL Ephemeris service and SPICE Toolkit the generation of precise position of the Earth are relatively fast, and, due to the static nature of the dataset, can all be pre-generated once for all guess distances. The calculation of the topocentric distance of the object at a guess barycentric distance R, is the problem of finding an intersection of a ray and a sphere.

This is a well known, solved, problem. Given a ray, parametrically, defined as:

$$x = x_1 + (x_2 - x_1)s = x_1 + is \tag{3.29}$$

$$y = y_1 + (y_2 - y_1)s = y_1 + js \tag{3.30}$$

$$z = z_1 + (z_2 - z_1)s = z_1 + ks, \tag{3.31}$$

$$\tag{3.32}$$

and a sphere defined as

$$(x - l)^2 + (y - m)^2 + (z - n)^2 = r^2. \tag{3.33}$$

then substituting the coordinates yields the quadratic equation:

$$at^2 + bt + c = 0, \tag{3.34}$$

where

$$a = i^2 + j^2 + k^2 \tag{3.35}$$

$$b = 2i(x_1 - l) + 2j(y_1 - m) + 2k(z_1 - n) \tag{3.36}$$

$$c = (x_1 - l)^2 + (y_1 - m)^2 + (z_1 - n)^2 - r^2. \tag{3.37}$$

The system of equations has a solution only when the discriminant is larger than 0. In this particular case we can always assume that the point on the ray representing our observatory will lie within the radius of the circle, as otherwise implies the guess distance $< 1au$. Therefore the quadratic equation will always have 1 positive and 1 negative solution, corresponding to the point of intersection in the positive and negative direction wrt. the point representing the observing location. Thus the positive solution is the correct one and the negative solution can be discarded.

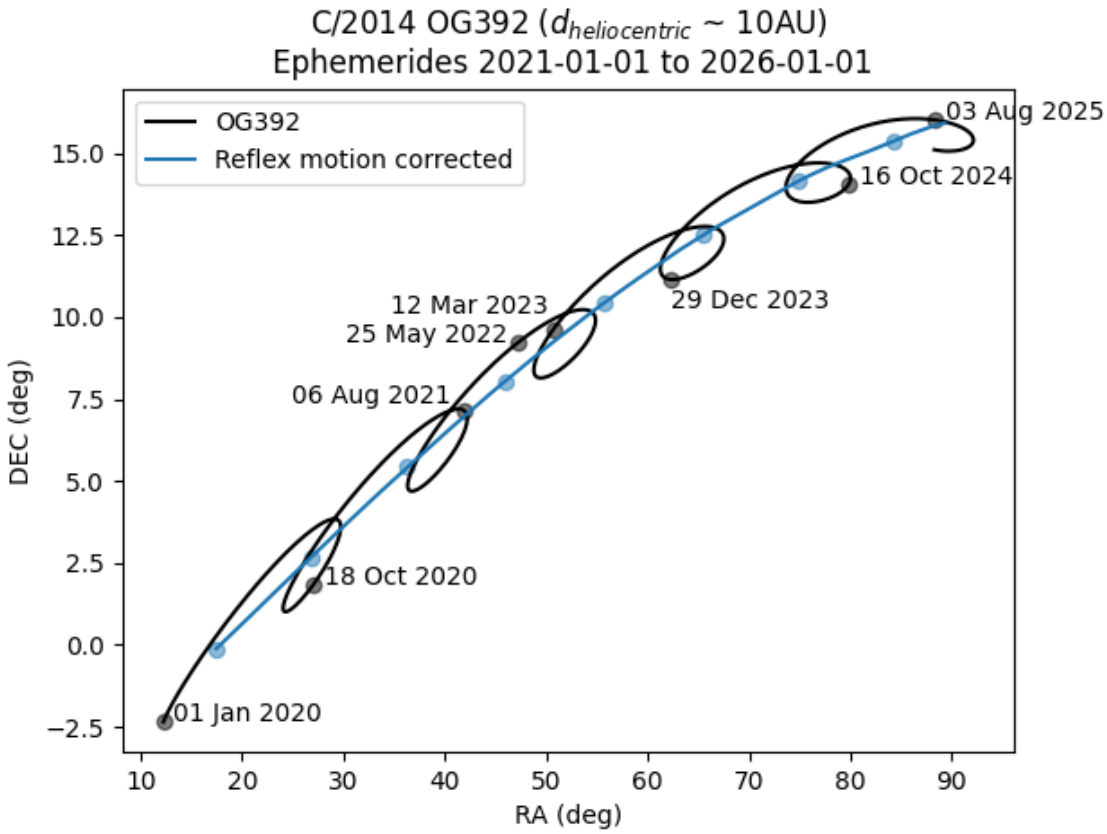


Figure 3.25: The ephemeris of the comet PanSTARRS as it would be appear from Cerro Tololo from 2021 to 2026. Many instances of retrograde motion due to reflex motion of the Earth are present, making it more difficult to detect with KBMOD. After correcting for the reflex motion of the Earth, blue line, the motion is visible more linear over longer periods of time. Due to the object's orbit itself the overall motion is still not completely linear, but at larger distances that effect is less pronounced even at larger time periods.

The solution allows for an optimized implementation. All 472 thousand difference exposures in the DEEP dataset were corrected for parallax for guess distances from 40 au to 106.2 au in steps of 0.1 au in less than an hour accounting for the generation of the ephemeris. The approach can be tested and shown that it linearizes the apparent motion of the object by creating ephemerides of a relatively nearby object, such that there are many instances of retrograde motion visible, and then correcting them for parallax as described above.

Parallax correction of the reported footprints of the considered data allows us to find and search images based on the exposure footprints as they would appear projected at the guess distance. Due to the specific cadence of the DEEP survey, where the survey fans out in a triangular fashion consistent with the dispersion of differential apparent rates of motion of TNOs, the effect of the parallax correction is the collapsing of the fan back unto itself[7].

Within the DEEP dataset, even exposures from subsequent nights are shifted relative to the previous nights to account for the expected motion of the objects. If we were to place the images into a collection for processing, the produced resampled images would look more alike to a mosaic than an fully overlapped image stack. By applying the correction for the reflex motion of the Earth to the WCS before resampling, the end result is a reprojection of the subsequent exposures to more closely overlap the previous ones, creating deeper searchable image stacks over a larger area of the sky. The largest single night single CCD collections in the searched dataset were 121 images deep. After reprojecting all of the DEEP image footprints, in the range of 40 to 102 au, and preforming clustering at each guess distance, 273 new potential search pointings with more than 300 images in them were identified. The largest collection of images was identified at $(353.0856, -4.4447)$ for a distance of 63.3 au containing 720 overlapping images. These collections were not searched in this work, as the size of the collection exceeded the required GPU memory on available hardware.

[7] https://epyc.astro.washington.edu/~dinob/kbmod_runs/estimates/reflex_correction_animation/

7 Conclusion

In this chapter we set out to achieve two core goals: a) complete the missing stares of DEEP, not analyzed in the first processing (Smotherman et al., 2023) due to resource requirements of KBMOD at the time, and b) demonstrating scalability and integration with Rubin Science Pipelines. We cover all of the steps required to process a dataset, from its raw data format, through to KBMOD at the scale of one-night of Rubin data. We discussed all aspects of this process, from processing of raw data into coadds and difference images by Rubin Science Pipelines, processing of created image differences with KBMOD, to validation of results by comparing the inserted simulated object properties with the retrieved orbits fitted to sets of tracklets.

The total volume of data processed with Rubin Science Pipelines is approximately 200TB, 20TB of which are the created image difference, in approximately 652 CPU-core days. Logging and log analysis proved to be key functionality for a high processing completion and good science validation. Lack of consistent compute and sufficient storage resources are the primary issues driving delays and failures.

To validate the results of processing and estimate the completeness and efficiency of the search, simulated objects on approximately 23 thousand unique orbits were inserted into the dataset. Approximately 5 million epehemerides and magnitudes, as a function of its albedo, phase and observational geometry, were evaluated and rendered onto the images. Forced photometry measurements at positions of inserted simulated objects (see Figure 3.6) in both the calibrated exposures and difference images show that we are over-subtracting flux in the difference images as a consequence of the object remaining in the coadd. This is a consequence of DEEP survey cadence and is not expected in Rubin data.

Significant software development time was invested into productization, performance and KBMOD functionality. While difficult to compare directly, as the processing steps do not retain a 1-to-1 mapping with the previous KBMOD processing, a 40% reduction in wall-

clock processing time per single exposure is achieved as compared to what is reported in Smotherman et al. (2021)[8]. Pipelining and workflow management allow us to more easily scale, even in fault-tolerant environments, removing significant overheads associated with management of data and results in the previous KBMOD processing. The cumulative effects of this work is the ability to search 10 times the volume of data searched previously, while simultaneously increasing the range of searched angles by 100% and the range of searched velocities by 38% at a comparable trajectory sampling density, in approximately the same amount of wall-clock time. An estimated 4 486 CPU-core hours were utilized, which includes time periods between two batch submissions in which no processing occurred. Logging and log analysis was once again crucial to achieving good processing completeness. Insufficient storage to materialize, even temporarily, all of the work units simultaneously ($\approx$ 20TB in total) is the primary driver of processing overheads as the processing had to be executed in batches.

Approximately 400 thousand potential results were recovered, 87 thousand of which passed filtering and had their positional uncertainties measured (see Section 4.2). Approximately 8 thousand simulated orbits were recovered from the dataset by positional crossmatching to the catalog of injected ephemeris. Detected simulated objects did not exhibit strong biases with orbital elements (see Figure 3.12). Detection fraction exhibits the strongest correlation with magnitude of the object, as expected, and a small bias towards lower eccentricity orbits which warrants further testing.

Precision, and correctness of execution, results and the new approach to uncertainty estimation were verified by fitting an orbit to the top 1000 orbits with the largest number of associated tracklets. There were 854 recovered orbit fits that were considered successful, i.e. $i_{fit} < i_{true} \pm 1°$ and $a_{fit} =< a_{true} \pm 10\%$; the majority of which ($\approx 80\%$) had recovered the injected orbital parameters within a narrow range around their true values: $a = a_{\text{true}} \pm 0.016au$, $e = e_{\text{true}} \pm 0.025$ and $i = i_{\text{true}} \pm 0.025°$. The high precision of the fits is attributed

[8]Previous processing time was reported to be 1300 seconds for an stack containing 20 images (excluding filtering), currently estimated processing time for an equivalent collection is 524 seconds (including filtering).

to the long length of arc, 2 years on average, in which the simulated objects were detected.

The simulated objects were used to estimate the total completeness, peak recovery efficiency of the processing and the limiting magnitude at which a 50% and 25% recovery rates were achieved. The achieved limiting magnitude at which 50% of simulated objects were recovered $m_{50} = 25.51$ mag, with a peak efficiency $c = 85.37\%$, and the limiting magnitude at which 25% simulated objects were recovered is $m_{25} = 25.98$ mag, with peak efficiency $c = 87.7\%$, across the whole survey. The mean single exposure limiting magnitude is $m_{25} \approx 23.8$ mag, i.e. we are able to recover objects 2.18 magnitudes dimmer compared to a single exposure detection limit.

Compared to previous processing by Smotherman et al. (2023), we achieve a 10% higher peak efficiency and 0.3 mag reduction of the limiting magnitude m_{25}, across the whole survey. We estimate that this change of limiting magnitude and efficiency will result in this search recovering approximately 77% of objects recovered by the previous search within the B1 field. The primary driver of the loss of the estimated number of total recovered objects is the selection of filtering parameters. Due to concerns of too large a number of results being returned, filtering thresholds were set higher than in the previous processing. The larger number of returned results has not materialized, however, indicating the search should be repeated with lower filtering thresholds in order to recover more objects within each field. From the remaining, approximately, 74 thousand results, that did not match to real or simulated objects, following the tracklet linking procedure described by Bernardinelli et al. (2022), we recover approximately 10 million ($\approx$ 200GB) multi-night linkages. These are currently undergoing orbit fitting and filtering process very similar to the one performed to validate the results (see Section 4).

Preliminary analysis of results retrieved by linking of tracklets with HelioLinC in the range of 20 to 100 au returns 44 detections consistent with the orbit of a new, not previously discovered, object. The orbital parameters of these objects are consistent with objects found by Smotherman et al. (2023). Approximately 80% (425 our of 536) of the linked tracklets

match injected simulated objects. The recovered linkages have orbital parameters consistent with the injected distribution of simulated KBO objects and an average observed length of arc of approximately 1 year. Approximately 50% of recovered linkages have an observed length of arc of 2 years. Thirty previously known objects are recovered in the search. Further analysis of the potentially new detections recovered in this process is required by: identifying the recovered linkages within the set of linkages recovered by linking procedure described by Bernardinelli et al. (2022), performing orbit fitting with the propagated measured positional covariance matrix as described in Section 4.3, performing force photometry measurements at the ephemeris of the potential results and by a visual inspection of cutouts and the moving coadds of the objects to rule out spurious detection. The number of returned results is not consistent with the expected number of results as estimated from the survey completeness metric. The number of results is expected to increase as the validity range of the hypotheses tested with HelioLinC is increased, and through analysis of linkages produced by following the Bernardinelli et al. (2022) linking process.

We are currently reprocessing the entire DEEP dataset on STAMPEDE where we were able to create coadds using all of available overlapping images in the DEEP data. This was difficult to achieve before, due to memory and time requirements of such jobs leading to large job failure rates on Klone. The reprocessing effectively solves the problem of the object's signal remaining present within the constructed coadd, leading to loss of flux and non-symmetrical PSFs in the image difference (see Section 3.2). We aim to run a KBMOD search on this dataset with the goal of characterizing the effects of the "negative wells" and the asymmetric PSF-like footprints on the effectiveness of the RESNet50 filtering step and measured positional uncertainties. In combination with reprocessing the dataset, as described in this chapter, we should be able to provide the best characterization of the effects of filtering parameter and data quality on the performance of KBMOD to date.

In Section 3 we discussed the image resampling as a novel functionality added to KBMOD. Resampling of images to a common topocentric coordinate grid was always a requirement

for KBMOD, but it used to be performed by the Rubin Science Pipelines during creation of image differences. Between the current and previous processing of DEEP, the Rubin Science Pipelines algorithms changed and no longer resample the image differences to a common coordinate grid. This functionality was added into the KBMOD as a part of this work, and has proven crucial to enabling KBMOD searches of Rubin data.

The apparent on-sky motion of TNOs is dominated by the reflex motion of the Earth. It is the reflex motion of the Earth that sets the maximum time-span in which an orbit remains approximately linear. In Section 6.1 we measured the idealized tolerance of KBMOD to deviation from linearity and provide a framework within which it is possible to evaluate the timespan in which a particular orbit remains sufficiently linear to be detected. In Section 6.1 we described how, with an additional guess distance, we can efficiently calculate the reflex motion of the Earth. We discuss how it's possible to adjust the WCS of each image to correct for the reflex motion of the Earth at a guess distance and then resample the image correcting for the parallax, a process referred to as reprojecting, resulting in images sampled to the same coordinate grid, but at a given distance instead of in infinity.

Rubin's cadence differs significantly from DEEP. Compared to the hundreds of overlapping images a night Rubin will deliver, on average, 2 to 4 images of a given area, usually in a different filter, every 2 to 4 days as a part of its Wide Fast Deep observing program. This is a sufficient time-span for the objects apparent motion on the sky to become nonlinear. Leveraging this new functionality and Rubin's commissioning data we have been able to recovere known objects using both reprojection and data from different filters.

In a period of 30 days, from Rubin, we can expect image collections containing 15 to 30 images. The change in limiting magnitude given a number of images follows the formula $\Delta m = (5/2) \log \sqrt{N}$ equating to an increase of the limiting magnitude by 1.2 to 1.8 mag. While not explicitly discussed in this Chapter we have developed tools to identify the regions with the largest number of new overlapping images, and on a scale of approximately 20TB, process the incoming Rubin data on a rolling monthly basis.

CHAPTER 4

SHAPE ANALYSIS

1 Introduction

In the last 15 years, significant effort has been made in the area of differential geometry and applications to computer vision (Abbena et al., 2006; Gorodski, 2016; Turaga & Srivastava, 2015). The motivation to include differential geometry in the field of computer vision is natural. Consider, for example, images of two similar but distinctly different irregular objects. Images of these objects could be taken from any number of angles such that some or all features may or may not be visible, or at different distances from the objects such that their apparent size changes, and the same type of object may appear in multitude of colors such that the values of the pixels themselves can change for the same type of object. This makes correct classification of these images from raw pixel data a very difficult problem.

If, for example, the object shared a common feature - a distance between two particular points, orientation between 3 particular points and so on - it would be possible to define a single or multiple vectors describing that feature. Forcing such feature vectors, or some product of them, to a unit norm effectively makes the problem scale-invariant. Transforming them, by minimizing some kind of distance between set of features, can make comparisons between them rotationally or translationally invariant. Sometimes, these transformations are not arbitrary, but are constrained in some ways. For example, applying inferences made on pixel values of a 2D image of a smooth surface with a known 3D point cloud should not yield results that lie outside of the boundaries of the projection of that shape on the image.

In this chapter we will look at some recent developments in the field of Riemmanian Computing in Computer Vision (RCCV) to evaluate how could it be used in the field of Astronomy. But before I attempt to describe the efforts made I want to motivate why in a clear, easy to understand and visual way, I believe the application of differential geometry to astronomical data holds promise. The challenge with engaging with this material is that its theoretical background lies in rather abstract fields of measure theory and differential geometry. Measures are a generalization of everyday common ideas, such as length, area,

volume, magnitude, mass, probabilities and so on and differential geometry looks for ways to apply these measures on smooth, differentiable, sets of numbers. But measures may be defined for sets for which we do not have an intuition of what a concept of a distance, area, or volume would be. In a very simplified example, a set of subsets $T = \{\{\dots\}, \{\dots\}, \quad \dots\}$ of a set is called a Borel set. One might not immediately be able to see how a concept of "length" could be defined on that set, but not only that it is possible, by developing the concept of σ-algebra and Borel measures, one can show that continuous probability is precisely an equivalent of it (**?**). In order to clearly and explicitly articulate the implications of the theory one would have to engage with it ab ovo, and begin many derivations from the elementary definitions of such sets and the concept of a distances between them. This is a task that is wholly out of scope for this work, but some understanding of the concepts is still required in order to understand it.

The claim in this chapter is that there is one such measure that can be defined for curves, and in particular for a differentiable topological curve, and that that measure can be used to define a distance between two different curves. Furthermore, that we are able to use this distance between curves in order to separate a set of curves into classes of equivalencies. In order to show this one has to first define what kind of set do curves come from, define a measure on it, and then derive the distance between two elements, or subsets of elements, of such a set.

To facilitate introducing the concepts and terminology in a, hopefully, easier, manner we will take one of the simplest of visualizable examples - spheres. Instead of distances between curves, we will observe distances between points on a sphere. To astronomers spheres are interesting because points on the surface of the Earth lie (approximately) on a sphere, positions of objects in the sky are measured on a sphere, the reader and all of their respected friends live on the surface on a sphere and so on. To lie on a sphere means to have a position that belongs to a set of allowable values M. So how would one calculate, let's say, an average position of the reader and all their friends, so that none would have to unfairly

travel a longer distance just to meet each other?

Because the friends are distributed over the entire globe, this becomes a much harder problem than, let's say finding an at least approximately average position of all the friends within the city. The key difference is that the underlying space on which they are now operating is non-linear. Being non-linear implies that for arbitrary positions p_1 and p_2, and some scalars a and b, their linear combination, $ap_1 + bp_2 \notin M$, does not have to end up in the set of allowable values. If the astronomer measures the positions of their friends using an right angle ruler, irrespective of the origin, they quickly discover that just by averaging the measured coordinates of all of their friends, they do not end up on the surface of the Earth and in fact, often significantly, deep under it.

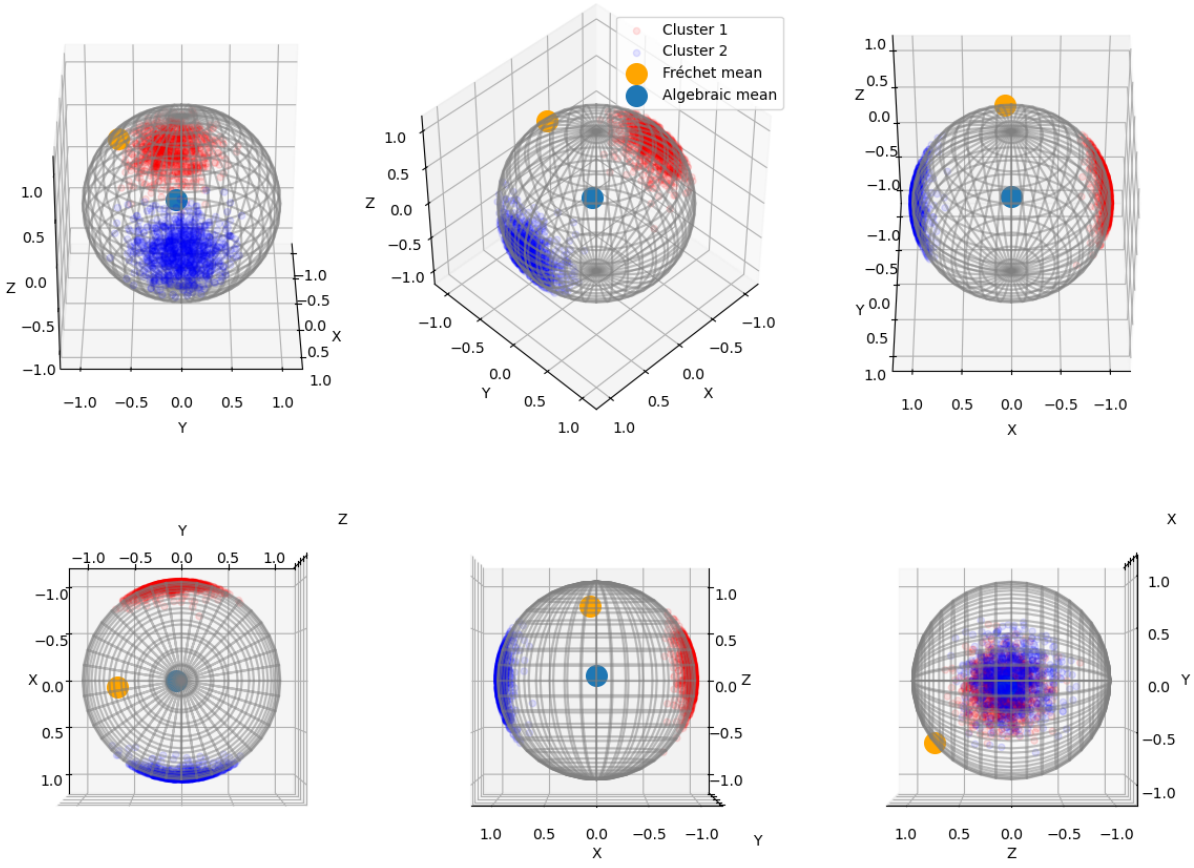


Figure 4.1: Points on the surface of a sphere, their Frechet and Arithmetic mean. Note how the arithmetic does not lie on the surface of the sphere.

But the astronomer could equip themselves with a globe and a map, marking on both, all

of the positions of their friends. On the globe, being a good astronomer, they mark down the great circles segments that lead to each of their friends. For each great circle segment, they read off the coordinates on the globe and transfer them to the map. These curves on a flat map represent the shortest way to reach each of their friends. For each curve they mark down a tangent vector to the surface of the Earth, with an origin in their position. They average these vectors, and mark a new position on the map. Transferring this position back to the globe, they can repeat this procedure until they find the best average position of the meeting. To their surprise, the found position not only lies on the surface of the Earth, but, unlike attempting to average their map coordinates or the coordinates measured by the right angle ruler, is truly equally distant from each friend.

In the language of differential geometry, the globe is a **differentiable manifold** embedded in $\mathbb{R}^3$ space, the map of the Earth is a **chart** of the manifold, tangents to the shortest curves to friends were constructed in the **tangent space** of the readers position, the function of the great circle is a **geodesic** and the described procedure is effectively equivalent to that of gradient descent to a Frechet's p-mean (Eichfelder et al., 2018). These are useful tools and concepts that we will use to define a generalized concept of a distance on any smooth manifold, which we will now attempt to do in an abridged fashion. For motivation on why specifically smooth manifolds see Sections 1.1 and 2.

There are many definitions of a manifold, attributed to different subfields of mathematics, and different ways of constructing them, but loosely speaking it is a non-linear space which is locally Euclidean. An atlas is a collection of charts that cover the manifold and each manifold is has at least 1 atlas. A chart for a topological space M is a homeomorphism φ (a bijective, continuous map with an inverse) from an open subset U of M to an open subset of a Euclidean space, traditionally given as a pair (U, φ).

Put simply, a projection that is bijective (one-to-one) between the manifold and Euclidean space, paired together with its domain, is a chart. For example, a Mercator projection is a chart, but Azimuthal Equidistant Projection isn't unless we explicitly exclude the poles. For

a sphere defined as $\mathbb{S}^2 = \{(x, y, z) \in R^3; \quad x^2 + y^2 + z^2 = 1\}$ the function $\varphi(x, y, z) = (x, y)$ is the classic orthogonal projection that maps the sphere to a unit circle. Two such charts are needed to cover the north ($z > 0$) and the south ($z < 0$) hemispheres, leaving out the equator $z = 0$, two more covering the "east" and the "west" hemispheres complete the equator, except for the two points where the two equators meet. Two more charts are needed, for 6 charts in total, to complete the atlas. On the example of the astronomer these are the functions and the maps what they would have to use when they were transferring their positions to the globe and back.

For each point p on the manifold, a set of all tangent vectors through that point is denoted as $T_p(M)$. Tangent vector $T_{p,i}(M)$ is the derivative of a differentiable curve passing through the point p, also called a directional derivative. The directional derivative of a scalar function f with respect to a vector v at some point p can be given as $T_{p,i}(M) = \nabla_v f = \boldsymbol{v}\frac{\partial f}{\partial p}$. In order for the directional derivative to exist the manifold must be differentiable in all points. $T_p(M)$ is a vector space of the same dimension as M and is called a tangent space. A collection of all $T_p(M)$ is called a tangent bundle of M, denoted as $TM = \cup_{p \in M} T_p(M)$.

Generally, a tangent bundle and tangent spaces are neither atlases nor charts. A chart provides the map from a part of the manifold to Euclidean space, such that any curve on the subset of the manifold can be expressed in the chart. The tangents are defined as derivatives of these curves, but these derivatives exist because the chart is differentiable and invertible. The tangents we get depend on the chart we select. But once we carry out whatever calculus in the chart, we *lift* the result back to the manifold M. The end result is completely independent of the choice of the chart we used and is a fundamental property of M itself. This is a very important point.

The idea of lifting or pushing forward certain operations into dual and tangent spaces, in order to circumvent practical limitations of computation without changing the result, is already very useful, but there is one more additional property, which will enrich the structure of manifold even further, the inner product. Let a Riemannian manifold (M, g) be a pair of

differentiable manifold M and the Riemannian metric g. If $(U, \varphi = (x^1 \cdots x^n); \varphi : U \to \mathbb{R}^n)$ is a chart of M, the coordinate vector fields are given as $\left\{\frac{\partial}{\partial x^1} \cdots \frac{\partial}{\partial x^n}\right\}$. For $p \in U$ and $u, v \in T_p(M)$ we can write:

$$u = \sum_i u^i \frac{\partial}{\partial x^i}\bigg|_p \tag{4.1}$$

$$v = \sum_i v^i \frac{\partial}{\partial x^i}\bigg|_p, \tag{4.2}$$

$$\tag{4.3}$$

then

$$g_p(u, v) = \sum_{i,j} u^i v^j g_p\left(\frac{\partial}{\partial x^i}, \frac{\partial}{\partial x^j}\right) = \sum_{i,j} u^i v^j g_{ij}(p). \tag{4.4}$$

and we can say that the Riemann metric g associates to each $p \in M$ a positive-definite inner product $g_p : T_p(M) \times T_p(M) \to \mathbb{R}$. Note how in the final expression $p \in U$ and $u, v \in T_p(M)$. Note also that, if we placed the elements g_{ij} into an $n \times n$ matrix it would be symmetric ($g_{ij} = g_{ji}$) and positive definite ($x^T M x > 0; \forall x \in \mathbb{R}; x \neq 0$). If we had another chart U' of M, so that it overlaps the chart U ($U \cap U' \neq \emptyset$) then by the chain rule we also have:

$$g_{i'j'} = \sum_{k,l} \frac{\partial x^k}{\partial x^{i'}} \frac{\partial x^l}{\partial x^{j'}} g_{kl}. \tag{4.5}$$

Since a tangent space is a convenient domain for computation, it's useful to define two mappings: a) the mapping from $T_p(M) \rightarrow M$, such that the origin of the tangential space is the point p - $\exp_p(0) = p$, called the exponential map $Exp_p(\boldsymbol{v})$ and b) its inverse, the logarithm map $Log_p(q)$. Note how, in the exponential map, $p \in M$, $\boldsymbol{v} \in T_p(M)$ and $Exp_p(\boldsymbol{v}) \in M$ and in the Logarithm map $p, q \in M$ and output is a tangent vector $\boldsymbol{v} \in T_p(M)$. The existence of the metric also provides us with a norm of a tangent vector $\|u\| = \sqrt{< u, u >_p}$. This allows us to define a distance for a, at least once, differentiable curve $\alpha \in C^1$

$$L[\alpha] = \int_0^1 < \alpha'(t), \alpha'(t) >_{\alpha(t)} dt. \tag{4.6}$$

Note again the similarities to Section 6.1 because this is precisely the definition of an arc-length used there. The $\alpha'(t) \in T_{\alpha(t)}(M)$ is defined using the Riemann metric at that point and again parameterized such that $\alpha(0) = p_1$ and $\alpha(1) = p_2$. Then the path with the smallest length is called a geodesic:

$$d^* = \arg\min L[\alpha]. \tag{4.7}$$

The length of the shortest geodesic between any two points defines the distance between those two points $d(p_1, p_2) = L[d^*]$. Now, finally, we have the concept of distance. So we can define an intrinsic mean:

$$\mu = \arg\min_{p \in M} \sum_i^N d(p, p_i)^2 \tag{4.8}$$

The points p_K which minimize the intrinsic mean are called Karcher means, and if there is a unique point $p_F \in M$ that is strictly the minimum, then it's called the Frechet mean (Eichfelder et al., 2018). For the sphere and a few other simple manifolds, the Riemannian metric g is rather simple and can be evaluated analytically.

Aside 1.1: Induced metric and distance

Evaluating the metric of a sphere is a classic exercise in all relativity textbooks. The parametrization of the sphere in spherical coordinates is substituted into the expression for the cannonical Riemann metric in Euclidean space $ds^2 = dx^2 + dy^2 + dz^2$ and the metric $ds^2 = R^2(d\theta^2 + \sin^2\theta d\phi)$ is derived. In the language of differential geometry, the composition of the functions (the substitution) is a smooth map between two manifolds $\mathbb{S}^2 \rightarrow \mathbb{R}^3$, since $\mathbb{R}^n$ is a manifold itself by the identity function called a pullback. This map is the pullback of the Euclidean metric, and we can say that it induces the metric on the manifold $\mathbb{S}^2$. Since a metric defines the inner product for every point on the manifold, we say it induces a distance. More precisely the distance is a map $X \times X =\rightarrow X$ that is non-negative, symmetric and satisfies the inequality of the triangles, a Riemannian metric is a covariant 2-tensor field on the manifold M, i.e. it assigns to each point p the inner product $g_p : T_p(M) \times T_p(M) \rightarrow \mathbb{R}$.

This is well known by the astronomer and is the reason why they immediately began to follow great circles when performing their iterative procedure to find the average position of all their friends. Effectively the astronomer in the example cheated. Intrinsically, they understood what the metric of a sphere is, whether or not they knew its mathematical expression explicitly. They knew the geodesic, the solution to Equation 4.7, are curves called "great circles" and how to construct them on the manifold, and they also knew the exponential and logarithm map of the manifold as the projections used to create the maps (charts) they used to measure the actual distance to their friends. If they did not have that information a priori, they would have had to attempt to find the shortest path between themselves and their friends by effectively testing all possible paths on a chart, integrating the distance in the chart as they moved along towards their friend, looking for a set of paths for which the sum of lengths integrates to the shortest total distance. For each step they had to use the exponential and logarithmic maps in order to move their position between the chart and the globe and to construct the charts themselves they would have had to construct a tangent space in each point they arrived in. This would be difficult and tedious work for an astronomer to perform but, luckily, today we can leverage auto differentiation and implementations of numerical calculation schemes that mimic this same process. Thus

we can perform the same calculations for manifolds for which the metric is very difficult, or even impossible, to express analytically. For example, manifolds such as Klein bottles (Jakobson et al., 2005) shown on Figure 4.2.

In Section 1.1 we identify the manifold of all smooth absolutely continuous real functions as our Riemannian manifold of curves. Applying the same principles outlined in the introduction so far, namely defining a map in the tangent space and then, by its pullback, defining the distance on the manifold itself, in Section 2 we define a distance function on the manifold of curves. We demonstrate that the defined distance is able to discern between classes of curves of different "shape" and that it significantly outperforms more naive approaches, such treating curves as elements of $\mathbb{R}^N$ space and applying the more intuitive measures such as Euclidean distance between vectors in that space; an analogy for the astronomers attempts to measure the distances between them and their friends by treating their coordinates as elements of $\mathbb{R}^2$ or $\mathbb{R}^3$ space.

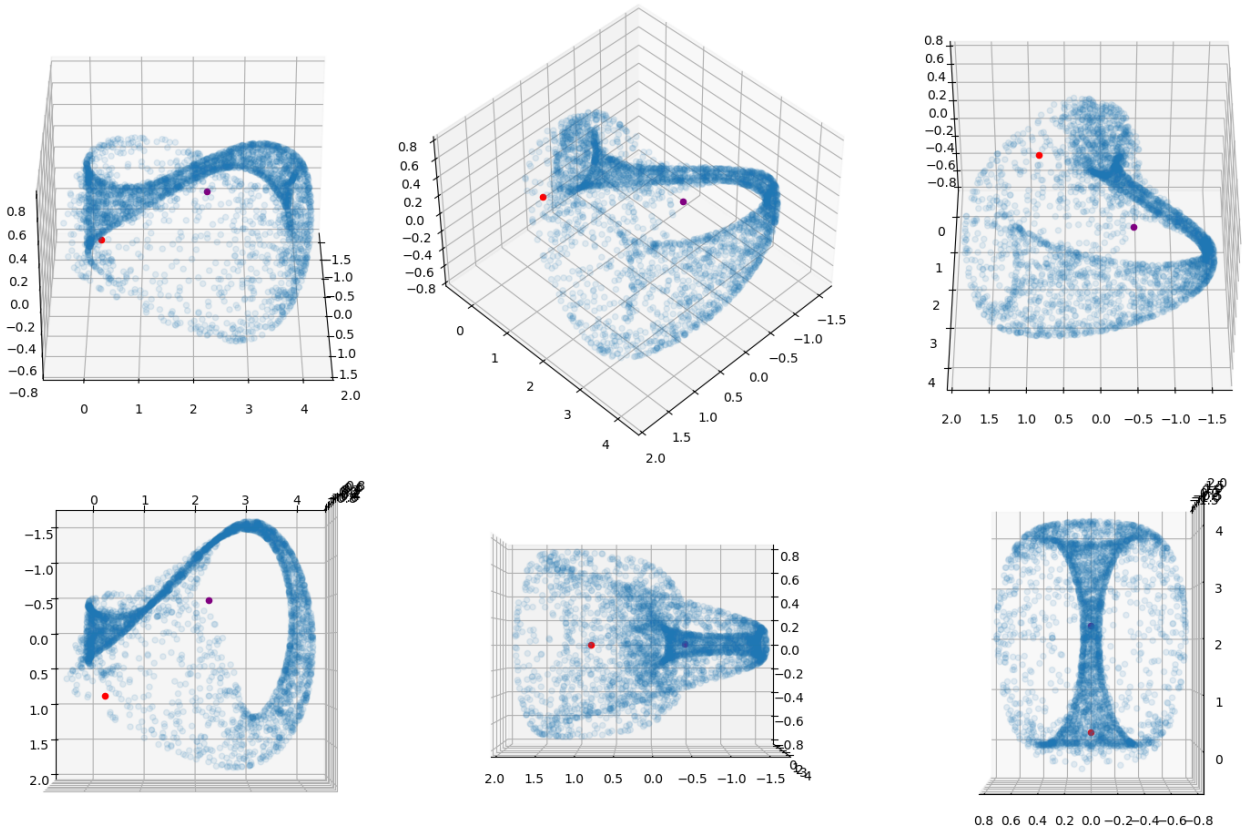


Figure 4.2: Non-uniformly sampled points on the surface of a Klein bottle and their Frechet and arithmetic mean. Note how the arithmetic mean does not lie on the surface of the Klein bottle.

Importantly, once the astronomer began treating the sphere from the perspective of differential geometry, they noticed that the tangent space $T_p(M)$ itself is a vector space, and that it naturally allows application of statistical procedures they were already familiar with. For example principal component analysis (PCA) or support vector machines (SVM), thus enabling different approaches to linear regression (and, 1982) and clustering (Ben-Hur et al., 2002). Because the geometry of the problem, even when non-linear, is inherently accounted for in this approach, we yield better results than performing the analysis on the manifold embedding itself. On Figure 4.3 we show how performing a PCA analysis on a set of randomly drawn points on a sphere. We compare two approaches: a) treating the points as if they were Euclidean coordinates and b) treating the points as if they belong to a manifold. The classical PCA analysis (second row on Figure 4.3), which treats the points as if they were elements of a 3D Euclidean space, explains less variance in the dataset than the tangent PCA (first row on the Figure 4.3). The tangent PCA allows us to reduce the dimensionality of the problem while, simultaneously, explain greater amount of variance with the same number of principal components. Notably, the principal components in the classical approach are vectors with origin in the arithmetic mean of the points, while in the tangent PCA the principal components are great circles on the sphere, much like in the example with the astronomer.

A group G is said to act on manifold M if there is a mapping $\phi : G \times M \to M$ for which a sequence of operations can be replaced by a cumulative transformation ($\phi(g_2, \phi(g_1, p)) = \phi(g_2 \cdot g_1, p)$) and it has an identity element. Every such transformation has an inverse. The consequence of the existence of a group action is that the manifold is naturally partitioned into classes of equivalencies, based on which action (or sequence of actions) are required to transform one point into another $\exists g \in G : p_2 = \phi(g, p_1)$, called orbits. The set of all orbits of the general linear group GL, for example, include transformations like translations, rotations, reflections and warping of the space (shearing, contraction, dilation etc.). The set of all orbits (equivalence classes) is called a quotient space of M under G. Applying classical statistical tools such as PCA, support vector machines, K-means clustering etc. to quotient

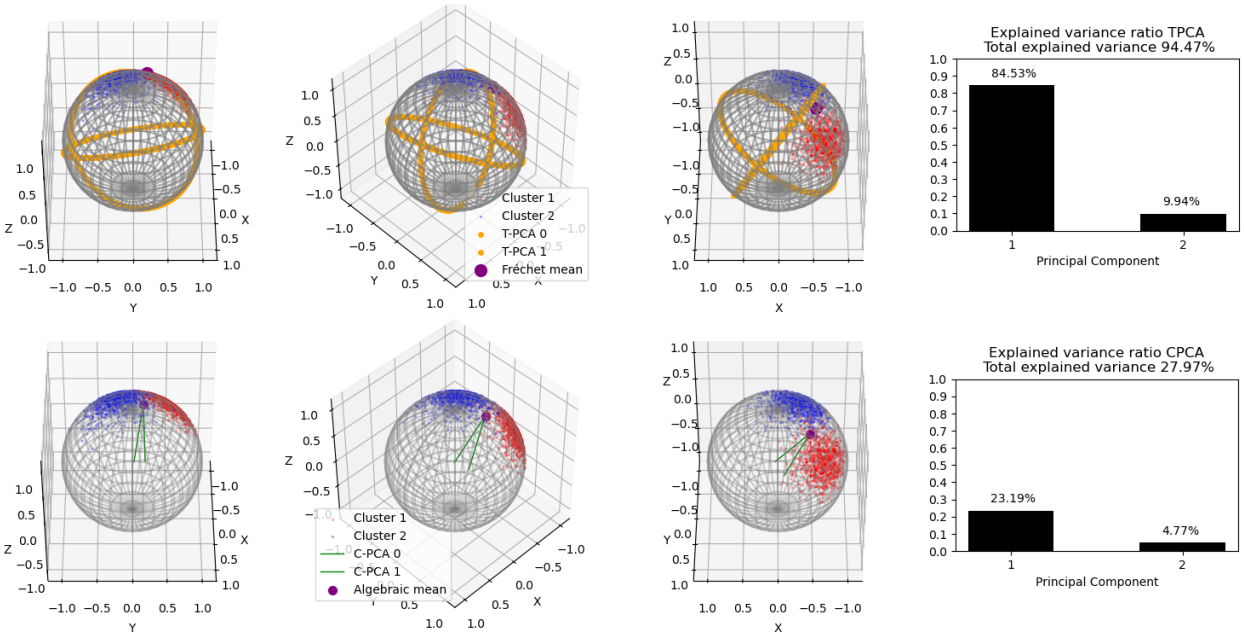


Figure 4.3: Tangent PCA performed (top row) versus PCA performed (bottom row). Fundamentally the process is the same, a mean point is selected from which an SVD is performed. For tPCA, the mean is the Frechet mean, and the Log Map is applied to the data to calculate their tangents at p ($\boldsymbol{v} \in T_p(M)$) before subtracting the mean and performing SVD. The classical PCA was applied directly to the coordinates themselves yields inferior results.

spaces enables us to classify, or perform regression on the data in ways that are invariant to certain transformations (Turaga & Srivastava, 2015).

In Section 3 we discuss issues with the distance between curves, as will be defined in Section 2. In the same chapter, we propose two different workarounds to this problem. One approach involves introducing yet another distance. This new distance is defined on the quotient space of our manifold of curves, analogous to group actions just discussed, and is thus rendered invariant to certain unwanted transformations. To define this distance, see Section 4.1, we borrowed from the idea of SVMs and their reproducible kernel property, to define a vector product on our quotient manifold. In Section 5 we describe how to solve this new problem we called "relaxed matching" and demonstrate that it produces expected results. In Section 6 we describe what happens when the approach is applied to astronomical datasets and the lessons learned, before summarizing and concluding in Section 7.

Finally, I would be remiss if I did not mention that the motivating examples were all focused on simple embedded surfaces that can be expressed in analytic fashion. This was done in part for practical reasons, as the metric and the geodesic can be analytically derived, and in part in hope that the visualizations on simpler examples would assist in understanding more abstract manifolds that can not be easily visualized. But it is important to stress that analytic manifolds are a subgroup of smooth (infinitely differentiable) manifolds, which themselves are a subgroup of differentiable manifolds. These can then even further be generalized by removing the requirement of positive definiteness, to make Pseudo-Riemannian manifolds. Lie groups, for example,[1] are all differentiable manifolds. There are many problems encountered daily by astronomers that already lie on one such manifold, or can be shown to lie on a manifold via a relatively simple transformation of data. The generality of the approach, alongside its (partly) demonstrated advantages, motivates this investigation into its applicability even in a broader sense.

[1] To avoid listing them all see table of Lie groups.

1.1 Application in Astronomy

Light curves are a ubiquitous tool in an astronomers tool box. Used to classify objects into specific sub-categories, as is the case with variable stars, infer the physical properties of the system, as with supernovae, or to detect new objects as is the case with exoplanet transits. Significant effort has been put forth within, and in collaboration between, each of these communities, to create, measure and compare the performance of light-curve classification tools (Hložek et al., 2020; Kessler et al., 2019). Consequently light-curve analysis tools cover a wide-range of approaches - from specialized targeted and deterministic algorithms, to large, most often supervised, machine learning algorithms. Some of the applied approaches include integrating external knowledge about the observed object, their population distributions, and even integrating multiple different kinds of measurement of the objects in order to produce the final classification. One of the challenges in these applications is the heterogeneous and often sparse sampling of the light curve. But the key features of well-sampled lightcurves is that any two light-curves from different class of objects tend to look rather distinct from one another. For example, while the variability within the class can be very large, it is difficult to mistake an AGN lightcurve with that of a supernova or a periodic variable star. These differences motivate the attempt to apply elastic shape analysis to light curves.

Shape analysis is a subfield of computer vision that attempts to detect, segment, annotate, measure or otherwise describe shapes in images and is of the earliest problems encountered in computer vision. Elastic shape analysis (Hartman et al., 2021, 2023; Joshi et al., 2007; Srivastava et al., 2011) focuses on the analysis of shapes by considering geometrical information in a manner that is invariant to certain transformations, including elastic deformations of the object's shape itself. To achieve this, the approach leverages the ideas borrowed from differential geometry, briefly introduced in Section 1. Younes (1998) showed that it is possible to define a distance between two geometric curves by defining an Riemannian distance on an infinite dimensional group acting on the curves, which can be explicitly computed

by numerically solving a variational problem. Over the last two decades many different approaches to solving this problem were attempted. Providing a comprehensive review of this research here would not be possible, but the research has slowly converged towards similar solutions. For a more complete overview of the historical context see Srivastava et al. (2011), whose prescription is followed here.

We consider light curves as elements of $\mathrm{AC}_0([0,1],\mathbb{R})$, i.e., the space of all absolutely continuous real-valued functions f defined on the unit interval such that $f(0) = 0$. This space, called a pre-shape space, is an infinite dimensional manifold on which we can, analogous to the sphere example in Section 1, define a distance between its elements. The elements of this manifold are curves, thus measuring the distance between two elements on this manifold measures the distance between two curves. Because, as we will discuss, the distance on that manifold, as defined by either Younes (1998) or Srivastava et al. (2011), is not invariant to certain transformations we have to identify elements of the preshape space that belong to the same orbits of shape-preserving transformations. We will discuss two attempts at providing a solution to this problem. In both cases we are, effectively, constructing a space called the shape space. Because the preshape space is a Riemannian manifold, the shape space inherits this Riemannian structure and is also a manifold. We will show that it is possible to induce a metric on the shape space manifold, which can be used to compute geodesics between two curves. Leveraging the distance as a measure of similarity of two lightcurves we aim to classify them into different categories in an unsupervised manner. The goal is to answer the following questions:

1. Can elastic shape analysis be applied onto light-curves?

2. How well does it perform?

3. What are the limitations of the approach?

2 The Square Root Velocity Distance

The square root velocity (SRV) transform is a mapping that takes an absolutely continuous function f and returns a scaled version of its gradient ∇f, where the scaling factor is the square root of the magnitude of the gradient. More formally, we define the SRV transform as the mapping $Q : \mathrm{AC}_0([0,1],\mathbb{R}) \to L^2([0,1],\mathbb{R})$, which is defined for all $f \in \mathrm{AC}_0([0,1],\mathbb{R})$ and $t \in [0,1]$ as:

$$Q(f)(t) := \begin{cases} \frac{\nabla f(t)}{\sqrt{|\nabla f(t)|}} & \text{if } |\nabla f(t)| > 0, \\ 0 & \text{else.} \end{cases} \tag{4.9}$$

Note that $L^2([0,1],\mathbb{R})$ denotes the space of all square integrable real-valued functions defined on the unit interval, i.e., all functions $f : [0,1] \to \mathbb{R}$ with for which

$$\int_0^1 |f(t)|^2 dt < +\infty.$$

The SRV transform induces the so-called SRV distance on $\mathrm{AC}_0([0,1],\mathbb{R})$ via the pullback of the L^2 norm on $L^2([0,1],\mathbb{R})$. More specifically, for two given light curves $f_1, f_2 \in \mathrm{AC}_0([0,1],\mathbb{R})$, this distance is defined as:

$$d_Q(f_1,f_2)^2 := \|Q(f_1) - Q(f_2)\|^2_{L^2([0,1],\mathbb{R})} = \int_0^1 \left| \frac{\nabla f_1(t)}{\sqrt{|\nabla f_1(t)|}} - \frac{\nabla f_2(t)}{\sqrt{|\nabla f_2(t)|}} \right|^2 dt \tag{4.10}$$

As described in Srivastava et al. (2011), the SRV metric is not invariant to translation, rotation, nor reparameterizations. In Figure 4.4 three different toy models are presented, broadly corresponding to the types of lightcurves observed by astronomers. Starting from the top left the curves constructed from individual Gaussians and representing a class of recurrent, often non-periodic, sources. On the top right, are three different curves that exhibit no periodicity, but a sharp spike at some observing time t_0, followed by an exponential decay. These curves are toy examples of a broad range of lightcurves associated with the

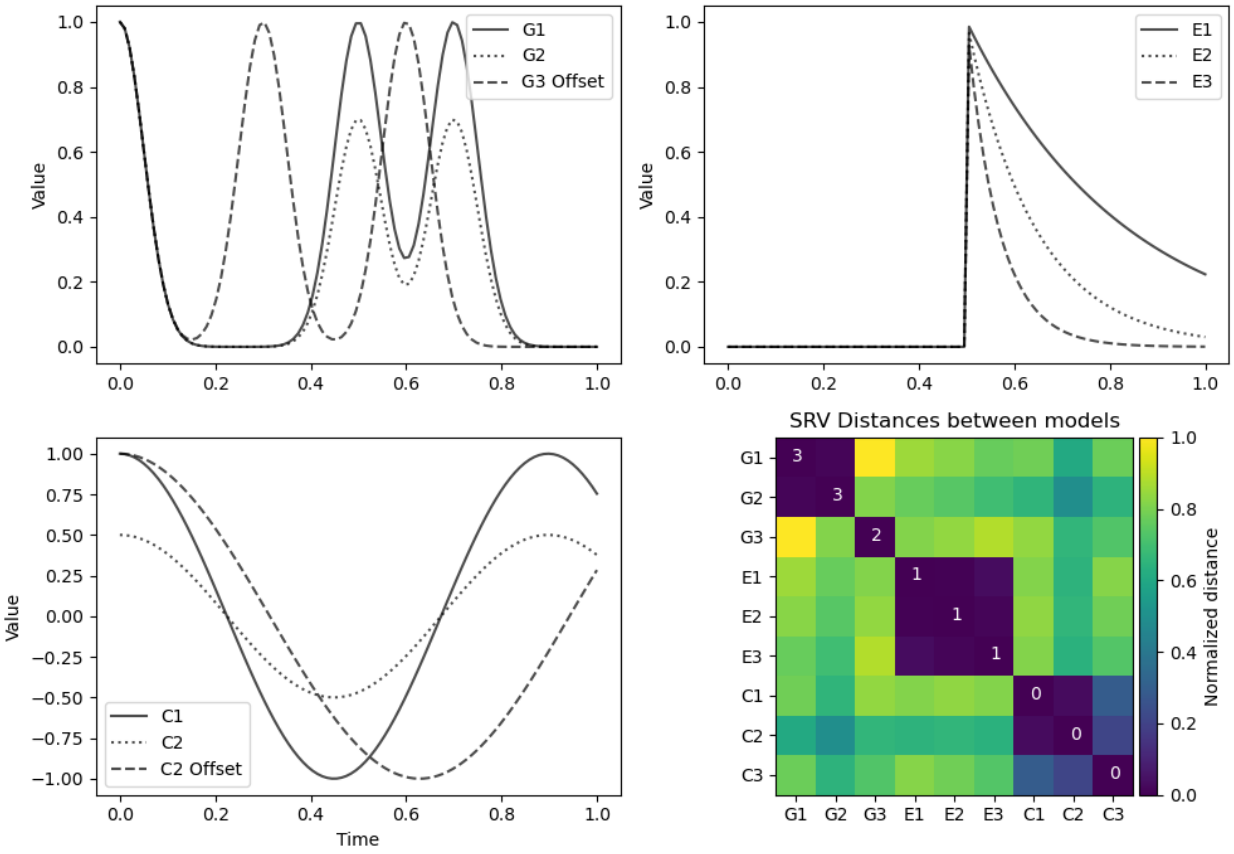


Figure 4.4: SRV distances between simple toy models of light curves. Top left is the sum of 3 Gaussians that have at least 1 Gaussian centered at $t = 0$. Two of the curves have two additional Gaussians at t_1 and t_2 with different amplitudes, while one of the curves has the two remaining Gaussians offset by $dt = 0.1$. Exponential functions vary only the decay time τ (top right) while the cosine functions offset the phase of one of the curves. In the bottom right is the distance matrix between all pairwise matches of all toy samples of curves with the numbers marking their final clustering label after applying agglomerative clustering using the precomputed distance matrix and the the number of clusters set to 4. Note the sensitivity to different peak times of $G_{1,2}$ and G_3, and phase offsets $C_{1,2}$ and C_3, stemming from the fact that the SRV distance is not invariant to reparametrizations or translations of the curves.

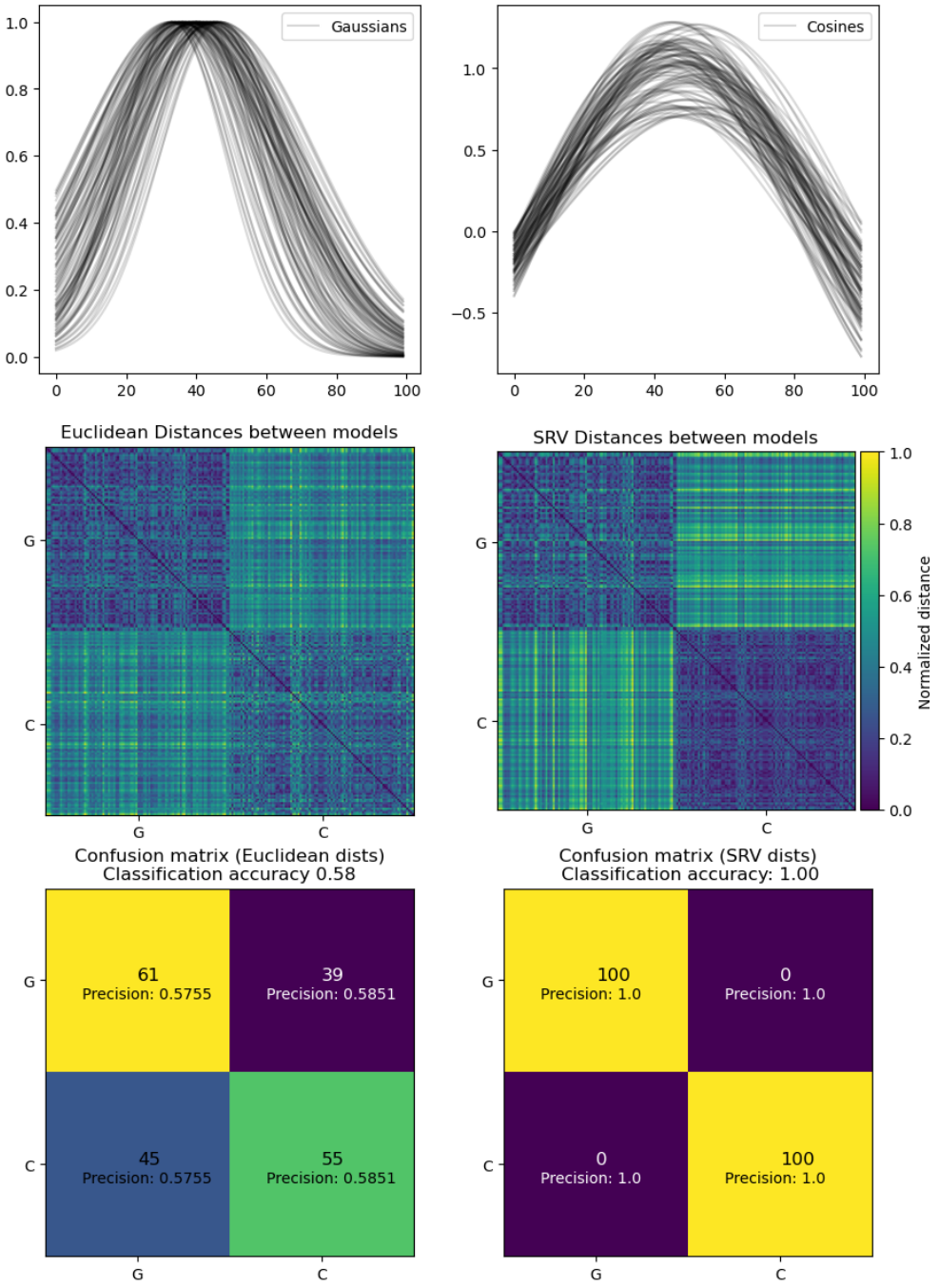


Figure 4.5: Comparison of SRV distance and Euclidean distance and the results of Agglomerative clustering on pre-calculated distance matrices. Clustering parameters were kept the same such that the two results are directly comparable. While the SRV distance is not invariant to translations or reparametrizations the intra-cluster variance is often sufficiently smaller compared to inter-cluster variance of calculated distances. In this example we are comparing two qualitatively similar sets of curves (first row): single-peak Gaussians (top left) and cosine curves (top right). The agglomerative clustering is performed on a pre-calculated distance matrix (second row). Compared are the Euclidean (see Equation 4.11) and SRV distances (see Equation 4.10). Agglomerative clustering results, bottom row, clearly show that SRV distance maximizes the variance between the two classes of curves in this example.

supernovae events. Finally, on the bottom left a series of cosine curves, representing a broad class of periodic lightcurves. To an astronomer these appear identical with the difference in phase attributed to the period and, or, time of observation of the object. However, similarly to the Gaussians in the top left panel, as can be seen in the distance matrix of the curves in the bottom right panel, that the two offset curves G3 and C3 yield large SRV distances to other G and C curves respectively. This indicates that if the toy exponential decay curve models were not as precisely aligned at the same peak brightness time t_0 that the resulting distances in the matrix would not classify them as same curves. This is confirmed, again, on the example of the G3 curve which is, as we can see in the 3rd row or column of the distance matrix, a large outlier.

In a practical sense, offsets between the peaks and different phases of the toy model curves represent the inherent stochastic variability of the underlying physical processes, periods and phases of recurrent transient events, different relative observing times of two different sources on the sky, the ages of those sources and so on. In a qualitative way we may classify these curves into the three classes of curves, as presented on the Figure 4.4, and this is representative of how an astronomer might consider this particular classification problem. This stands in contrast to how other fields of research may tackle classification of these curves, where a case can be made that Figure 4.4 is in fact displaying 5 different classes of curves.

While SRV distance is sensitive to both translation and reparameterization of curves, the variability of curves within the same class is often notably smaller than the distance between two different classes of curves, even when they are relatively similar. Shown on Figure 4.5 are sets of 100 single-peaked Gaussian and 100 cosine curves. The values of the amplitude, mean and standard deviation for the Gaussians and the phase, period and amplitude for cosine curves were chosen such that the two curves qualitatively resemble each other. SRV and Euclidean distance were calculated between pairs of curves with no mechanism of addressing the miss-alignment or parametrization issues between the curves in order to make the two

distances directly comparable. The Euclidean distance used in Figure 4.5 was defined as:

$$d_E(f_1, f_2) = \sqrt{\sum_i |x_{1,i} - x_{2,i}|^2}. \tag{4.11}$$

SRV distance clearly distinguishes between the two classes of curves in this example, whereas Euclidean distance does not. Despite that, as will be discussed and is shown in Figure 4.6, the effects reparameterization can be large. Therefore, before we can proceed forward with the classification of curves based on their shapes, we must provide a solution to this problem of reparametrizations.

3 Factoring out reparametrizations

The common ways to deal with this problem is to factor out the transformations either by a) performing the inference on the quotient space, or b) transforming the data to factor these transformations out before applying the SRV metric. In the most simplistic cases of well sampled high signal-to-noise ratio measurements it might be trivial address these issues. For such curves, features can be consistently detected and curves can be aligned based on them. Curves with time-sampling dense enough such that a smoothed interpolation polynomial can be used as an appropriate tool for resampling can then be resampled to appear to be observed at the same times relative to each other. However, this is not a trivial problem to solve and becomes especially difficult at low SNR.

In this Section we propose a different approach to dealing with parametrization invariance of the SRV metric. For every curve $q \in L^2([0,1], \mathbb{R})$ there exists a curve β such that the given curve q is the SRV function of that β, unique up to a translation. This curve can be obtained from the equation

$$\beta = \int_0^t q(s)ds$$

To address the issue of scaling we rescale all of the curves to be of unit length:

$$\int_D \|q(t)\| = \int_D \dot{\beta}(t)dt = 1 \tag{4.12}$$

Thus, the SRV functions associated with these curves are elements of a unit hypersphere in the Riemann manifold $L^2([0,1],\mathbb{R}^n)$, since all curves will have the same norm (length). Invariance to rotations is not a concern we need to address in this work. In practice, SRV not being invariant to reparameterizations means that our observed (discrete) light curves have to be sampled in the exact same way for us to be able to compare them via the SRV distance from the previous section. This is a severe limitation for practical applications, as the brightness of light curves is usually measured at different times and a different number of times.

Ideally, one would like to compare light curves regardless of how they are sampled. To do this, we model the light curves as unparameterized absolutely continuous functions. The light curves are taken to be elements of the quotient space $\mathcal{S} := \mathrm{AC}_0([0,1],\mathbb{R})/\mathrm{Diff}([0,1])$, that is, the space of parametrized light curves modulo reparametrizations, which are modeled as diffeomorphisms of the parameter space $[0,1]$. By a diffeomorphism on $[0,1]$, we mean an absolutely continuous mapping $\phi : [0,1] \to [0,1]$, which has an absolutely continuous inverse.

Consequently, to compare unparameterized light curves, we require a distance defined on $\mathcal{S}$. Thankfully, the SRV distance on the space of parametrized light curves from (4.10) has the key property of being invariant under the action of the group of diffeomorphisms of the parameter space $\mathrm{Diff}([0,1])$, i.e., it satisfies

$$d_Q(f_1 \circ \phi, f_2 \circ \phi) = d_Q(f_1, f_2)$$

for any given light curves $f_1, f_2 \in \mathrm{AC}_0([0,1],\mathbb{R})$ and $\phi \in \mathrm{Diff}([0,1])$. This can be seen by applying a change of variables in the expression for the SRV distance from Equation (4.10). Thus, the SRV distance descends to a distance on the quotient space $\mathbb{S}$, given by:

$$d_{\mathbb{S}}(f_1, f_2) = \inf_{\phi \in \text{Diff}([0,1])} d_Q(f_1, f_2 \circ \phi) \tag{4.13}$$

With a slight abuse of terminology, we henceforth refer to the quotient space distance, namely $d_{\mathcal{S}}(\cdot, \cdot)$, as the SRV distance.

It follows that computing the SRV distance between unparameterized light curves involves solving an optimization problem over the infinite dimensional reparameterization group $\text{Diff}([0, 1])$, which is a challenging endeavor that is usually accomplished by discretizing the group into a finite dimensional approximation space, and solving the discretized problem via, for example, a dynamic programming approach such as dynamic time warping (DTW, Olsen et al. (2018)).

4 Relaxed matching

We now address the computation of the SRV distance for unparameterized curves, whose expression is given in Equation (4.13). Our goal is to compute this distance without actually performing the computationally expensive minimization over the infinite dimensional reparameterization group. To achieve this, let $f_1, f_2 \in \mathcal{S}$ be two light curves, and we reformulate Equation (4.13) as follows:

$$\inf_{f \in \text{AC}_0([0,1],\mathbb{R})} d_Q(f_1, f)^2 \text{ such that } f = f_2 \circ \phi. \tag{4.14}$$

We emphasize that the minimization here occurs over the space of absolutely continuous curves $\text{AC}_0([0, 1], \mathbb{R})$ rather than over the reparameterization group $\text{Diff}([0, 1])$ - which is a much easier problem to discretize and solve numerically. Intuitively, the problem involves finding an absolutely continuous function f that is close to f_1 in terms of the SRV distance, while being equal to f_2 up to a reparameterization.

If we now have access to a similarity term Δ that has the property that $\Delta(f, f_2) \approx 0$,

then $f \approx f_2 \circ \phi$. We could therefore use this relation to relax problem Equation (4.14) by using a Lagrange multiplier $\lambda > 0$ as follows:

$$\inf_{f \in \mathrm{AC}_0([0,1],\mathbb{R})} d_Q(f_1, f)^2 + \lambda \Delta(f, f_2). \tag{4.15}$$

Solving the problem above thus involves finding an absolutely continuous function f that is close to f_1 in terms of the SRV distance, while being equal up to f_2 up to a reparametrization ϕ - a property of f that is enforced indirectly through the similarity term Δ.

To identify a good choice for Δ we turn to varifold fidelity metrics, which are distances between the varifold representations of absolutely continuous curves. Given an absolutely continuous curve $f \in \mathrm{AC}_0([0,1],\mathbb{R})$, the varifold μ_f associated with f is a positive Radon measure on the product space $\mathbb{R} \times [0,1]$ which is defined as the image measure:

$$(f, n_f) * \mathrm{vol}_f \tag{4.16}$$

where

$$n_f = \frac{\nabla f}{|\nabla f|} \tag{4.17}$$

is the unit gradient field of f, and

$$\mathrm{vol}_f(t) = |\nabla f(t)| dt \tag{4.18}$$

for $t \in [0,1]$ is the volume form of f, i.e., the arclength measure induced by f. In other words, for any Borel set $B \subset \mathbb{R} \times [0,1]$, $\mu_f(B)$ measures the total length with respect to vol_f of all $t \in [0,1]$ such that $(f(t), n_f(t)) \in B$. Then, any norm $\|\cdot\|$ on the space of varifolds should induce a distance on $\mathcal{S}$, given for any $[f_1], [f_2] \in \mathcal{S}$ by $\|\mu_{f_1} - \mu_{f_2}\|$, where we again emphasize that this expression does not depend on the choice of parametrization for f_1 and f_2 in the respective equivalence classes $[f_1]$ and $[f_2]$.

4.1 Reproducible Kernel Hilbert Spaces

Embedding lower dimensional data into a higher-dimensional space is often successfully employed in the field of machine learning with data that lie in Euclidean space. The procedure yields a richer high-dimensional representation of the original data distribution, sometimes making tasks such as classification easier. Most notably used in support-vector machines (SVD) a similarity function $\mathbf{k} : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ is trained on labeled samples (x_i, y) from which some set of weights is learned. Labeling of unlabeled data is then treated as the application of the similarity function $\mathbf{k}$ called the kernel. Most famously, the approach does not require the computation of the coordinates in implicit space of features $\mathcal{X} \times \mathcal{X}$ but instead requires computing only all pairwise inner products of the images of pairs of data points. Emphasis on "only" as this still remains computationally expensive. This is known as the "kernel trick". The same concepts extend themselves to manifolds. Points on manifold M are mapped to elements in higher dimensional Hilbert space $\mathcal{H}$. A kernel function $k : M \times M \to \mathbb{R}$ is used to define the inner product on $\mathcal{H}$, making it a Reproducible Kernel Hilbert Space (RKHS).

As described above on a practical, familiar example, a Hilbert space is an inner-product space that is complete with respect to the norm induced by the inner product. An RKHS is a Hilbert space of functions acting on a non-empty set $\mathcal{X}$ in which all evaluation functionals are bound and continuous. The inner product of RKHS can be defined by a bivariate function $\mathcal{X} \times \mathcal{X}$, known as the reproducing kernel of RKHS (Aronszajn, 1950), hence the name.

The challenge is identifying a kernel that is provably positive definite in order to define a valid RKHS. While many such kernels are known for $\mathbb{R}^n$ generalizing them to manifolds is challenging. Nevertheless, this is possible as originally shown by Aronszajn (1950) and outlined in Turaga & Srivastava (2015) Ch. 3 and in Smola & Vishwanathan (2004). The full proof and conditions required for positive definiteness will not be reproduced here, but a brief overview is provided for the case of Gaussian kernels for the sake of context.

Theorem 1. *Let (M, d) be a metric space and define $k : M \times M \to \mathbb{R}$ by:*

$$k = \exp\left(-\gamma d^2(x, y)\right). \tag{4.19}$$

Then k is a a positive definite kernel for all $\gamma > 0$ if and only if there exists an inner product space ν and a function $\psi : M \to \nu$ such that

$$d(x, y) = \|\psi(x) - \psi(y)\|_\nu. \tag{4.20}$$

A kernel $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ is positive definite if it's symmetric ($f(x, y) = f(y, x) \forall x, y \in \mathcal{X}$ and f is also negative definite, since $\exp(-\gamma f(x, y))$ can then be shown to be positive definite for all $\gamma > 0$. Therefore the d^2 in the theorem must be found such that it's negative definite.

It can be shown (Turaga & Srivastava, 2015) that a value of -2 can be factored out of a generic definition of distance by expanding the square in the expression of the vector product:

$$\sum_{i,j} c_i c_j \|x_i - x_j\|_\nu^2 = \sum_{ij} c_i c_j < x_i - x_j, x_i - x_j >= -2 \sum_{i,j} c_i c_j < x_i, x_j >_\nu$$

as all the other terms vanish. Thus d^2 is negative definite, and k is positive definite for all γ, which proves the forward direction. In the other direction, we assume that k is positive definite, therefore d^2 must be negative definite. Cowling (1983) demonstrates a general case of how for a negative definite kernel there exists a mapping $f(x, y) = \|\psi(x) - \psi(y)\|^2 + h(x) + h(y)$ for which $h = 0$ if $f(x, x) = 0 \forall x$. Since d is a metric, $d(x, x) = 0 \forall x \in M$, therefore there exists a $d(x, y) = \|\psi(x) - \psi(y)\|_\nu$ as required by the theorem.

Although we can prove their existence, the construction of the kernel itself remains a challenge. It can be shown that it is possible to obtain a Gaussian kernel from the geodesic distance only when the Riemann metric that defines the geodesic distance essentially renders the manifold equivalent to $\mathbb{R}^n$. This is possible for some Riemannian manifolds, for others it can be shown to be theoretically impossible (Turaga & Srivastava, 2015).

We consider two different kernels, a Gaussian (Turaga & Srivastava, 2015) and a Cauchy-Binet kernel (Smola & Vishwanathan, 2004) that we believe are applicable in our scenario. Norms defined from positive definite kernels on $\mathbb{R} \times [0,1]$ have been shown to lead to particularly advantageous expressions for numerical computations. Specifically, we consider the class of norms $\|\cdot\|_{V^*}$, where V is a reproducing kernel Hilbert space of functions on $\mathbb{R}\times[0,1]$, whose kernel is of the form:

$$k_V(x_1, n_1, x_2, n_2) = \Psi(|x_1 - x_2|)\Phi(n_1.n_2), \tag{4.21}$$

where, Ψ and Φ are two functions defining a radial kernel on $\mathbb{R}$ and a zonal kernel on $[0,1]$ respectively. For instance, one can choose Ψ to be a Gaussian kernel with width $\sigma > 0$, i.e.,

$$\Psi(|x_1 - x_2|) = \exp\Big(-\frac{|x_1 - x_2|^2}{\sigma^2}\Big), \tag{4.22}$$

while Φ can be chosen as a Cauchy-Binet kernel of the form:

$$\Phi(n_1.n_2) = (n_1.n_2)^2. \tag{4.23}$$

Following from the particular form of μ_{f_1} and μ_{f_2} as well as the reproducing kernel property in V, the inner product of the two varifolds in V^* can be explicitly derived as follows:

$$\langle \mu_{f_1}, \mu_{f_2} \rangle_{V^*} = \int_0^1 \int_0^1 \Psi(|f_1(t_1) - f_2(t_2)|)\Psi(n_1(t_1).n_2(t_2))\mathrm{vol}_{f_1}(t_1)\mathrm{vol}_{f_2}(t_2). \tag{4.24}$$

Accordingly, the squared varifold kernel distance between μ_{f_1} and μ_{f_2} can be obtained by a quadratic expansion of the norm:

$$\|\mu_{f_1} - \mu_{f_2}\|^2_{V^*} = \|\mu_{f_1}\|^2_{V^*} - 2\langle \mu_{f_1}, \mu_{f_2} \rangle_{V^*} + \|\mu_{f_2}\|^2_{V^*}. \tag{4.25}$$

The squared varifold distance is ideally suited for use as the discrepancy term Δ in the

relaxed matching problem outlined in (4.15) due to its reparametrization invariance, which finally allows us to formulate the varifold-based relaxed matching problem for lightcurves:

$$\inf_{f\in \mathrm{AC}_0([0,1],\mathbb{R})} d_Q(f_1, f)^2 + \lambda \|\mu_f - \mu_{f_2}\|^2_{V^*}. \tag{4.26}$$

5 Numerical Optimization

We now describe how to numerically solve the relaxed matching problem (4.26) in order to estimate the SRV distance between two light curves $f_1, f_2 \in \mathrm{AC}_0([0,1],\mathbb{R})$. To do so, we will explain how to discretize the problem and solve it as a finite-dimensional optimization problem.

First, we explain how light curves are discretized. We assume, as is the case in practice, that there are N timepoints $0 = t_1 < t_2 < \ldots < t_N = 1$ at which we measure the brightness of a given supernova to produce its associated light curve $f \in \mathrm{AC}_0([0,1],\mathbb{R})$. We denote the brightness measured at time t_i by $f^i := f(t_i)$ for $i = 1, \ldots, N$. Thus, a discretized light curve is simply a vector $f = [f_1, \ldots, f_N] \in \mathbb{R}^N$.

We will write and $\nabla f^i := \nabla f(t_i)$ for the value of the gradient of f at these timepoints, and note that the gradient field

$$\nabla f = [\nabla f^1, \ldots, \nabla f^N] \in \mathbb{R}^N, \tag{4.27}$$

can be computed via finite differences. The discretized unit gradient field of f can also be computed accordingly as $n_f = [\frac{\nabla f^1}{|\nabla f|}, \ldots, \frac{\nabla f^N}{|\nabla f|}] \in \mathbb{R}^N$, where $|\nabla f| = \sqrt{\sum_{i=1}^N (\nabla f^i)^2}$.

The discrete volume form of f at time t_i, which we write as $\mathrm{vol}_f^i = \mathrm{vol}_f(t_i)$, can be approximated as $\mathrm{vol}_f^i = |\nabla f^i|$.

These discretizations let us approximate the SRV distance between discretized lightcurves $f_1 = [f_1^1, \ldots, f_1^N] \in \mathbb{R}^N$ and $f_2 = [f_2^1, \ldots, f_2^N] \in \mathbb{R}^N$ as follows:

$$d_Q(f_1, f_2)^2 \approx \sum_{i=1}^{N} \left| \frac{\nabla f_1^i}{\sqrt{|\nabla f_1^i|}} - \frac{\nabla f_2^i}{\sqrt{|\nabla f_2^i|}} \right|^2. \tag{4.28}$$

Note that here, we require *both* f_1 and f_2 to be in $\mathbb{R}^N$, i.e., sampled in the same way using the same number of timepoints. Moreover, the discretizations introduced above also let us approximate the varifold inner product between the varifolds associated to the discretized lightcurves $f_1 = [f_1^1, \ldots, f_1^N] \in \mathbb{R}^N$ and $f_2 = [f_2^1, \ldots, f_2^M] \in \mathbb{R}^M$ as follows:

$$\langle \mu_{f_1}, \mu_{f_2} \rangle_{V^*} \approx \sum_{i=1}^{N} \sum_{j=1}^{M} \Psi(|f_1^i - f_2^j|) \Phi(n_{f_1}^i . n_{f_2}^j) \mathrm{vol}_{f_1}^i \mathrm{vol}_{f_2}^j. \tag{4.29}$$

We can then recover the discrete version of the squared varifold distance by a quadratic expansion of the norm above as in Equation (4.25). Note that here, we do not require f_1 and f_2 sampled in the same way - they can be sampled inconsistently and arbitrarily.

If we want to write Equation (4.29) even more explicitly using the expressions of the Gaussian kernel for Φ and the Cauchy-Binet kernel for Ψ, as well as the more explicit expressions for the unit gradient fields n_{f_1} and n_{f_2} and for the volume forms vol_{f_1} and vol_{f_2} in terms of the function values and gradient field only, we obtain:

$$\langle \mu_{f_1}, \mu_{f_2} \rangle_{V^*} \approx \sum_{i=1}^{N} \sum_{j=1}^{M} \exp\left(- \frac{|f_1^i - f_2^j|^2}{2\sigma^2} \right) \left(\frac{\nabla f_1^i}{|\nabla f_1|} \cdot \frac{\nabla f_2^j}{|\nabla f_2|} \right)^2 |\nabla f_1^i| |\nabla f_2^j| \tag{4.30}$$

As a result, equipped with the discretizations for the SRV distance from Equation (4.28) and for the squared varifold distance from the quadratic expansion of the norm in Equation (4.30), we can solve for the discrete version of the relaxed matching problem in Equation (4.26) between two discrete light curves $f_1 = [f_1^1, \ldots, f_1^N] \in \mathbb{R}^N$ and $f_2 = [f_2^1, \ldots, f_2^M] \in \mathbb{R}^M$ as follows:

$$\inf_{f \in \mathbb{R}^N} d_Q(f_1, f)^2 + \lambda \|\mu_f - \mu_{f_2}\|_{V^*}^2, \tag{4.31}$$

using the L-BFGS algorithm for example. To reiterate, the distances between two unparam-

eterized curves f_1 and f_2 are measured by finding an absolutely continuous test curve f that we can freely parameterize in any fashion. We minimize the SRV distance between the test curve and one of the curves $d_Q(f, f_1)$, penalized by the varifold distance to the second curve $\|f - f_2\|^2_{V^*}$. The parameter λ is a user-set weight that scales the varifold similarity metric, i.e. sets how strongly we penalize reparametrizations. The SRV distance has the effect of measuring the amount of energy required to elastically deform the test curve f, starting from curve f_1, until it minimizes the distance to the test curve f_2. The varifold distance is expected to produce a class of equivalencies for all differently parameterized curves, i.e. when two differently parameterized curves of the same shape are evaluated, the varifold distance between the test curve f and target curve f_2 is expected to be near zero. When two curves are both of different shape and parameterization, both SRV and the varifold distance are expected to be large.

We validate both approaches on the same toy dataset that was used in Figure 4.4. In that example, curves C1 and C2 were phase-aligned cosine curves with different amplitudes and curve C3 was a cosine curve of equal amplitude to C1 but with a different phase. Results of the validation are shown of Figure 4.7. Moving from left to right in Figure 4.6 we first compare the equally sampled and peak aligned cosine curves C1 in blue and C2 in red, and find the SRV distance of 0.66. On the second plot from the left, we have resampled the points on the curve C2 (in red) such that the C1 and C2 curves are no longer equally parametrized. As expected from two unequally parametrized curves the measured distance is substantially larger than the expected value. By applying an resampling procedure on the C2 curve we may resample it in the same manner as the source curve C1. This procedure produces an SRV distance effectively equal to the expected value (Figure 4.6, right panel). This process of resampling does not affect already equally parameterized curves as shown in Figure 4.7a. Finally, in Figure 4.7b, we demonstrate the relaxed matching minimization produces the same value for distance for the same curves regardless of their parametrization.

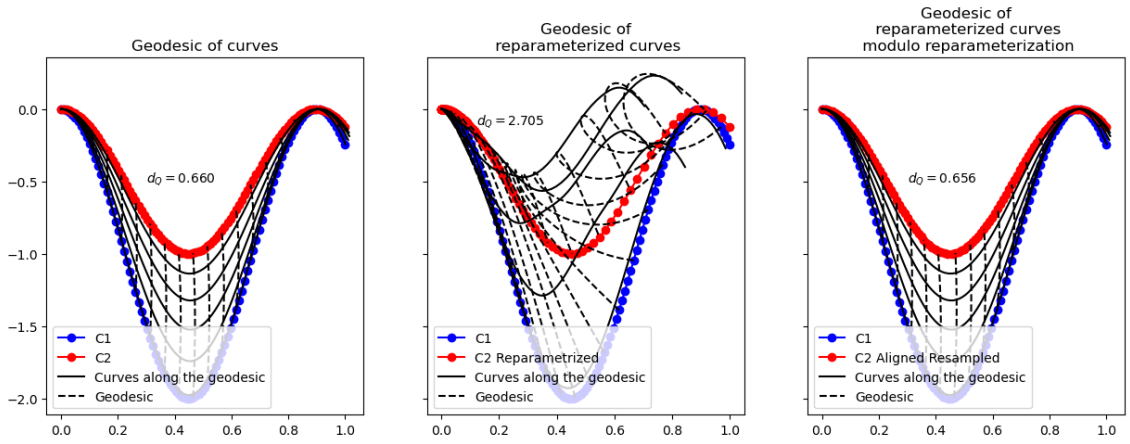
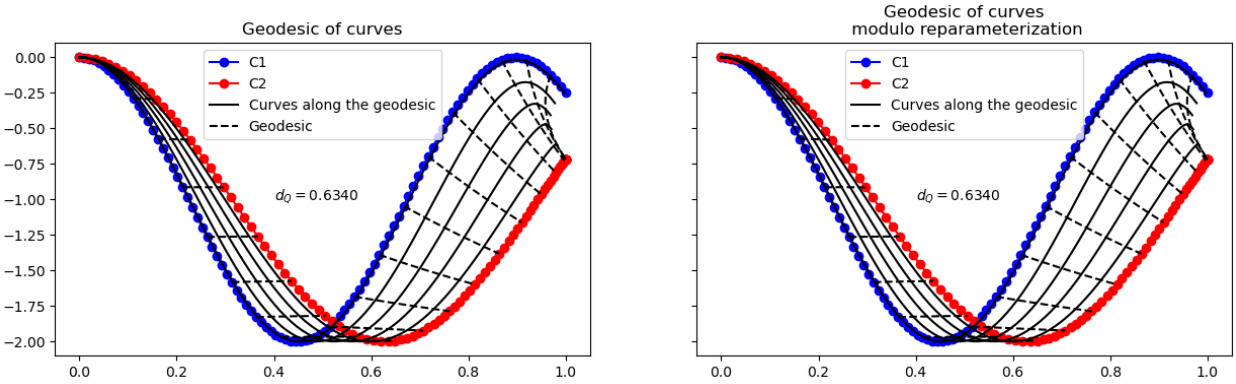


Figure 4.6: Distances between curves C1 (blue) and C2 (red) from Figure 4.4. The C2 curve is sampled more sparsely along the curve in the middle panel, simulating different observing times and therefore different parametrization between curves C1 and C2. The distance is calculated on the feature aligned and resampled approximation of the middle C2 curve in the right panel. The distance is effectively equal to the true distance (left), while the calculated distance of the differently parametrized curves (middle) is significantly larger.

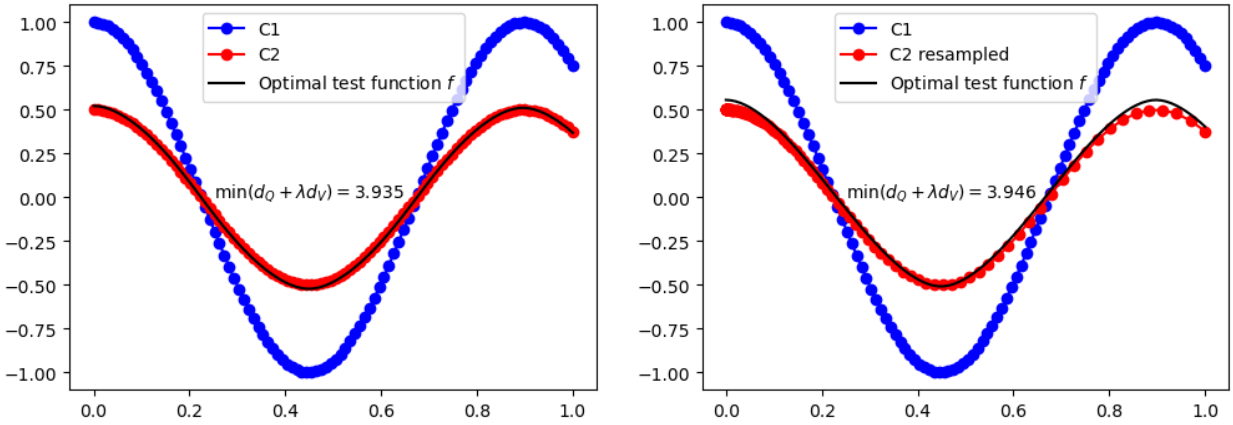
6 Results

We approach the testing of the problem using two different datasets. The first dataset is a set of models simulated using the SNANA (Kessler et al., 2009) package. Developed to facilitate comparison of different supernova (SN) lightcurve analysis methods the SNANA package is capable of simulating multiple SN light curve models as well as light curves of other transients often confused for SN objects during analysis such as micro-lensing events, stellar flares, Cepheids etc. SNANA is additionally capable of simulating light-curves based on survey conditions accounting for non-photometric conditions and varying time intervals between observations due to bad weather. At each survey epoch and sky location, the simulation uses measured observing conditions to generate realistic light curves.

Surveys such as LSST will generate multi-band measurements of objects visible from the southern hemisphere on approximately 3 day cadence. To prepare for the orders of magnitude more lightcurve data than what is available today, 3 years of observations following LSST cadence were simulated for 19 different models. More than 100 million transients and



(a) Distances between the C1 (blue) and C3 (red) curves. Feature alignment and resampling process does not affect the calculated SRV distance when applied to the same two curves. Notably, the translation of the curves induces a much smaller change in measured distance compared to reparameterization (see left and middle panels of Figure 4.6). As discussed in Figure 4.5 the intra-cluster variance between similar curves is often sufficiently smaller compared to inter-cluster variance of calculated distances between curves making clustering based on the SRV distance possible even when the alignment and resampling or relaxed-matching procedure do not perfectly address the issues of translational or reparameterization invariance.



(b) Distances between the C1, C2 and C2 sampled more sparsely along the length of its curve calculated by the relaxed matching minimization procedure. The distance value of the relaxed matching problem, displayed in the middle each panel, is the minimized sum of the SRV and varifold distances and isn't directly comparable to distance in Figure 4.6. The SRV distance of the optimal test function f and the source curve C1 is 0.632, very close to the true value of $d_Q = 0.66$ (see left panel of Figure 4.6)

Figure 4.7: Validation and comparison of the SRV distance calculated by feature alignment and resampling procedure (Figure 4.6) and the relaxed matching minimization approach (Figure 4.7b). Both approaches are successful in addressing the issue of differently parametrized curves without affecting the distance of equally parametrized curves.

variable sources were generated and presented as a challenge on the data science competition platform Kaggle. The Photometric LSST Astronomical Time-series Classification Challenge (PLAsTICC) (Kessler et al., 2019) compared classification solutions submitted by over 1000 different teams between September 28th 2018 and December 17th 2018 (Hložek et al., 2020).

Ultimate goal is to show that the described approach is able to resolve different classes within the PLAsTICC challenge data. However, as we will discuss, the challenges posed by numerical optimization of the problem of reparameterized curves are significant. Therefore we use SNANA to simulate a simpler dataset, corresponding closer to the same curve parameterization conditions required by the approach in order to evaluate its viability. We simulate 1000 light-curves at half-day cadence (12h) in a single passband, r, sampled over a period of 60 days at very high signal to noise ratio. For non-recurring transients, such as SNs, there is a well defined time of bolometric peak brightness that is forced to a small MJD range. However, the time of the peak varies by band. Recurring transients, such as RR Lyra, have no well-defined phase and therefore can not be forced to the same parameterization by the simulation itself. Simulations were executed on the Cori cluster, a supercomputer at the National Energy Research Scientific Computing Center (NERSC), retired in 2023. This is our second dataset used to validate the feasibility of the approach. Because SNANA accounts for the distribution and likelihoods of observing an object in an observation, the numbers of simulated lightcurves for a particular class can be different than the target count of 1000. The final counts of simulated lightcurves are shown in Table 4.1.

6.1 SNANA Simulations

To demonstrate viability and correctness of classification we use a random subset of all simulated curves for three models: Active Galactic Nuclei (AGN), SALT2 template based SNIIa and Cepheid lightcurves. Note that the Cepheid model lightcurves are not equally parameterized in this test. To assign labels to the lightcurves we calculate the full distance matrix between all pairs of lightcurves in the sample. In the simplest approach to classification, the

Model name	Count
CART	1000
SNIIn+HostXT-V19	1000
dwarf-nova	300
AGN	145
SNII-Templates	1000
uLens-Binary	378
SNIa-SALT2	1000
SNIax	5
d-Sct	462
KN-K17	1000
Mdwarf-flare	304
RRL	6
PISN-STELLA-HECORE	10
uLens-Single-PyLIMA	291
uLens-Single-GenLens	349
SNIIb+HostXT-V19	1000
SNIb-Templates	1000
SNIc-Templates	1000
TDE	1000
PISN-STELLA-HYDROGENIC	1000
SNIc+HostXT-V19	1000
EB	81
SNIa-91bg	1000
SNIcBL+HostXT-V19	1000
Cepheid	44
SNIIn-MOSFIT	132
SLSN-I+host	1000
SNIb+HostXT-V19	1000
SLSN-I-no-host	56
SNII-NMF	1000
ILOT	192
PISN-MOSFIT	1000
SNII+HostXT-V19	2
KN-B19	1000

Table 4.1: Number of simulated light curves per included lightcurve model.

histogram of distances can then be partitioned into classes by fitting a probability distribution model such as Gaussian Mixture Models (see Figure 4.8). GMMs are a probabilistic model that assumes all the data points are generated from a mixture of a finite number of Gaussian distributions with unknown parameters.

GMM clustering achieved a 96.6% classification accuracy and relatively high per-group classification precision. The key reason for the success of this method is the selection of lightcurve classes being compared. Cepheids are variable stars that, despite their misalignment in phase and unequal parameterization due to differing periods, are periodic smooth curves. AGN lightcurves most resemble a random walk, thus lacking periodicity, while SN lightcurves are smooth, aperiodic, and have low variance of shapes within the class itself. Thus the distances between any two SN lightcurves will be smaller than between SN or Cepheids; Cepheids will have larger distances to both SN and AGNs, but due to their periodicity the distance between two Cepheid lightcurves will be smaller than the distance between different classes of objects; thus yielding neat clusters of distances between curves belonging to the same class of objects. The approach is sensitive to the normalization of the distance matrix, due to the nature of GMMs, as well as the normalization of the curves we are comparing. Normalizing the area under the lightcurve, compared to normalizing its arc-length as is often successful in computer vision problems, compared to normalizing the range of its maxima and minima yield different results. Finding a normalization scheme that maximizes the separation of clusters of distances in the distance histogram could yield better results. Note additionally that in the simplistic simulations we also rely on a single observing band. Cepheids are expected to peak in all bands at approximately the same time, while the bolometric peak times for SN events vary across individual bands. Multi-dimensional histogram segmentation is a common approach to image segmentation problems in which the three histograms are usually color values of the pixels (Jyothirmayi et al., 2016; Reddy et al., 2007; Srinivas & Rao, 2007). The same approach could yield clearer results applied to this problem as well.

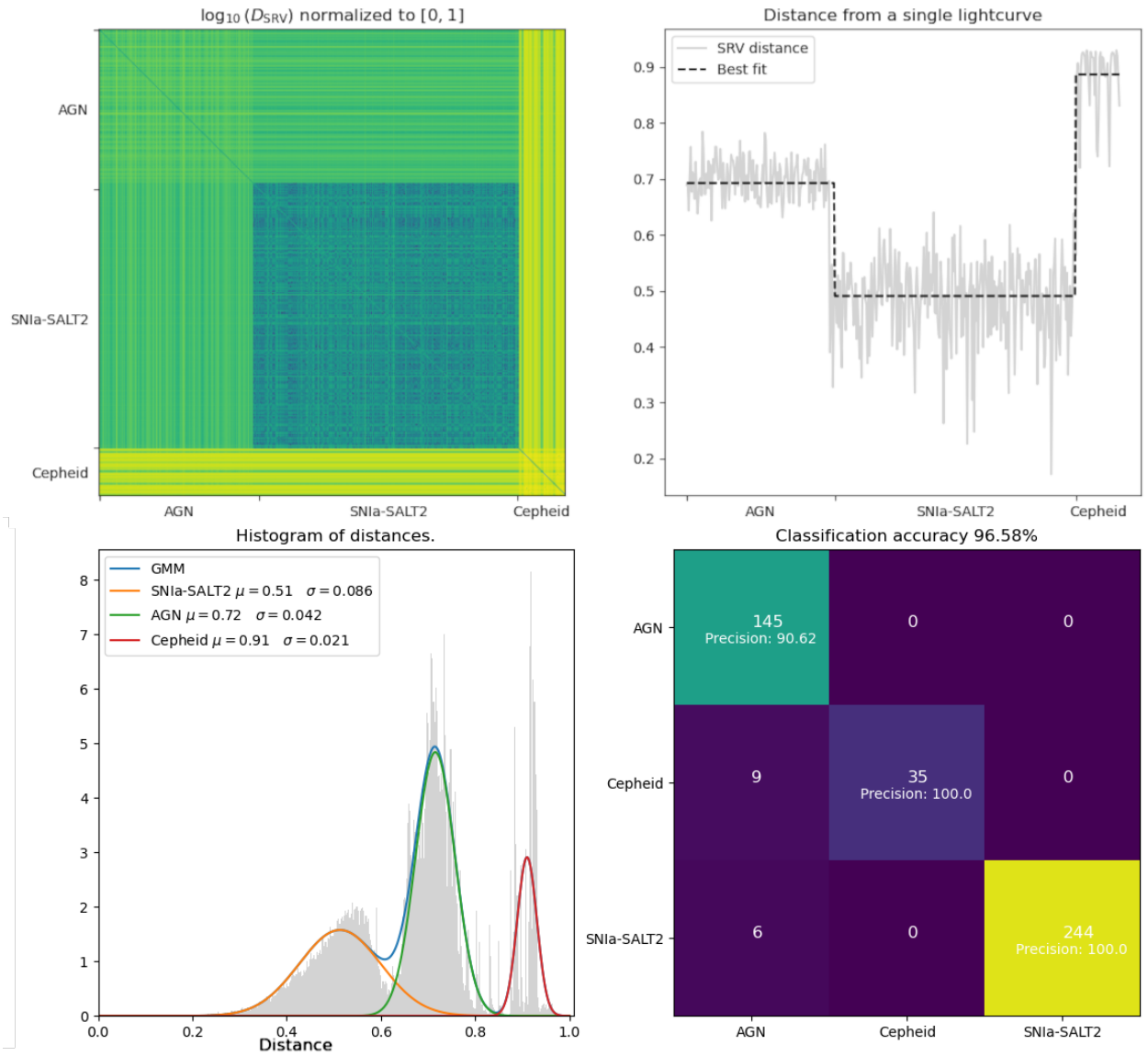


Figure 4.8: Classification of unequally distributed subsets of AGN, SN1a, and Cepheid lightcurves. The subset contains 145 AGN lightcurves, 250 SN1a lightcurves derived from SALT 2 templates, and 44 Cepheid lightcurves. The selection bias between different classes of lightcurves has little effect on the effectiveness of the approach because the distances between each class are well resolved. The full log-distance matrix between all pairs of curves normalized to the range $[0-1]$ is shown on the left-most plot. The second plot from the left shows the distances from a single selected curve (a row or a column of the distance matrix). A best-fit step-function, based on the Heaviside function, plotted in black. Classification was performed on the histogram of all distances, shown in third plot from the left, by fitting a 3 component Gaussian mixture model. Each element of the distance matrix can then be assigned a probability of belonging to a particular class of objects, and the highest likelihood is selected as the associated classification for that cell. The median association along a row or column is used as the final class of the object.

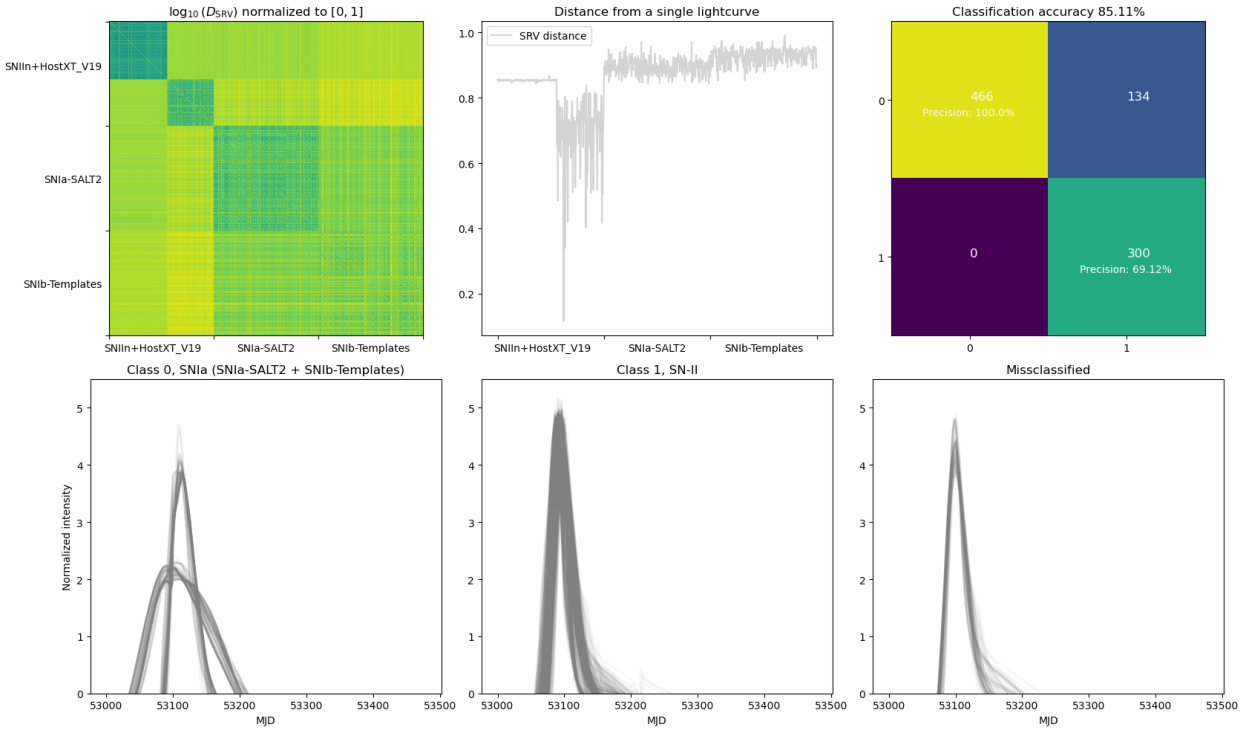


Figure 4.9: Classification of Type I and Type II SN lightcurves, with 300 samples for each class with the Type I class sampling from 2 different Type I models. The top row shows the full, normalized, log distance matrix. Top middle selects a single lightcurve (row or column) of the distance matrix and plots the values of distances as a line. The top right is the confusion matrix of classification that yields relatively good classification accuracy, albeit with a relatively low precision for Type II SN lightcurves. The bottom row plots the samples from each of the classes demonstrating how Type I and II SN lightcurves are sufficiently different from each other to be discernible using SRV distances. The bottom right shows the 124 misclassified lightcurves.

The high classification accuracy between the AGN, SN and Cepheid curves is expected, especially when they are well sampled as is the case in the simulated set of lightcurves, due to their distinct shapes. Supernovae classifications are generally split by two factors: a) their spectra and b) the physical processes undergone by their progenitor. Type I supernovae feature a hydrogen-less spectra while Type II feature a hydrogen line in their spectra. Type Ia supernovae are associated with thermal runaway processes occurring in a binary systems of a donor and a white dwarf stars, while all other sub-types (Ib, Ic, IIP, IIL, IIn, IIb subtypes) are associated with core collapse. Qualitatively, to first order, all lightcurves of SN events are similar, a rapid growth in bolometric luminosity over a period of 2-3 days, with a subsequent decrease in the luminosity over a few days. Key difference between Type I and II lightcurves is the decay rate post bolometric peak, where Type II exhibits a longer linear falloff compared to the Type I lightcurves that obey a luminosity-decline rate relationship that correlates the width and the peak brightness of the SN with its decline rate. Type Ia oftentimes exhibit an bump in their lightcurve, an increase of luminosity during the decay of intensity, associated with the changing optical depth of the expanding shell of gas. Because of the variability in their peak brightness, width, duration and stretch of the lightcurve, however, most SN lightcurves are rather similar. Thus we would expect the classification to perform worse than the previous example. Because we are only comparing between two classes of curves, with 3 distinct shapes, we use agglomerative clustering again and label the two Type I classes with the same label.

One of the challenges of agglomerative clustering is determining the optimal distance cut-off on the decision tree, used to assign labels to the clusters. Generally two approaches are employed: a) the acceleration cutoff measures how often the decision tree splits into branches and attempts to determine at what distance cutoff does additional branching stop producing statistically meaningful classes between neighboring branches and b) the inconsistency criterion that is determined as the difference of a cluster merge's height h to the average distance and normalizing it by the standard deviation of the distance formed over the depth's previ-

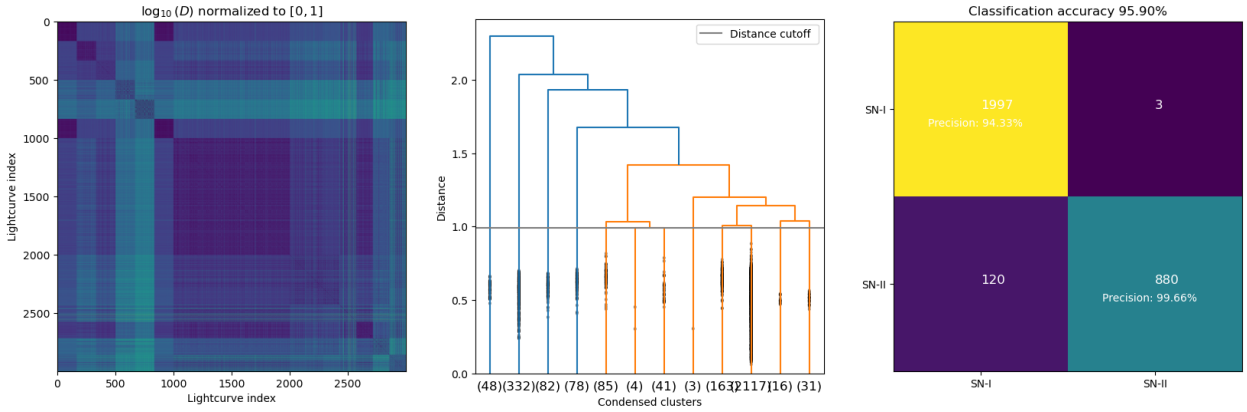


Figure 4.10: A truncated dendogram ($p = 11$) of the agglomerative clustering of a sample of 1500 lightcurves, with the distance threshold of 1.0 is shown on the left. The distance threshold assigns 12 distinct labels to classified data. The third branch from the right contains nearly all of the Type I SN lightcurves, while the remaining branches are associated with Type II. Shown on the right is the confusion matrix of all 3000 classified lightcurve. Many self-similar subclasses of lightcurves are visible as squares of different intensity along the primary diagonal of the matrix.

ous levels (Zahn, 1971). The default implementation of the SciPy's agglomerative clustering algorithm, used for this classification, is the inconsistency criterion which is very sensitive to the depth of the tree, thus yielding different results across the different numbers of samples. It is easy to discern, by observing the distance matrix in Figure 4.8, that there is significant structure within the lightcurves. There are at least 5 readily apparent classes of self-similar lightcurves visible to the naked eye.

Using a dendogram (the decision tree) of the classification of 900 total lightcurves and determining a new optimal distance threshold at the value of 0.99 was determined. The optimal distance yields 3 distinct classes of lightcurves, which are re-labeled to the two target classes of lightcurves, yielding a 99.89% classification accuracy. Further bifurcation yielded no improvements to the classification until nearly the depth of the tree, suggesting that over-fitting was not occurring. To demonstrate that the determination of the distance is not an artifact of lightcurve selection effects and over-fitting, but that the approach has predictive powers, the sample of lightcurves was increased to include all available lightcurves per class (1000) and without the change to any of the classification parameters and cutoff

distance the classification was repeated. Classification accuracy of 91.73% was achieved. There were 12 recovered distinct classes of curves. Nearly all of Type I SN lightcurves were located on a single branch, indicating they share a very self-similar lightcurve shape, distinct from that of Type II SNs, which exhibited a larger variation in the shape of the curves as measured by the SRV distance modulo reparametrizations and translation.

Although it would likely be possible to find at least an empirical solution to the problem of classification, for example, a mix of SNI, SNII, AGN and Cepheid lightcurves simultaneously, unsupervised classification of labeled data will always suffer from degeneracies forcing the problem-specific optimization, or making uniform global optimization of the problem difficult. Thus, one more approach was tested on the SNANA simulations. The approach is, effectively, K-means clustering of the data. First, a random subsets of the lightcurves are labeled. The Frechet mean is used to calculate the centroid of each class. The distance matrix of all curves was calculated and the lightcurves were re-labeled. In an iterative procedure, the new Frechet means were calculated for the new labels, and the relabeling was repeated. The iterations were terminated when the distance between new and old centroids were smaller than the desired user-set tolerance or when the inertia, defined in our implementation as the change of the total area under the curve, was smaller than an user-set allowed threshold. Centroids derived in this approach are then used to label the dataset. The interesting aspect of this approach is the lack of any labels associated with the dataset, with the clustering being driven purely by the geometric information inferred from the shape of the curve itself. The same two classifications described above were performed with the K-Means clustering and achieved 87.95% accuracy for the 250 instances of three distinct classes of lightcurves each (AGN, Cepheids and SNIa-SALT2 models) and 98.9% for the comparisons of 250 selected SN Type I and Type II lightcurves. The K-Means approach is notably more sensitive to large outliers in the dataset. In the presence of a large outlier, it biases the Frechet mean estimate and often is designated as a class unto itself, thus yielding a lower classification accuracy.

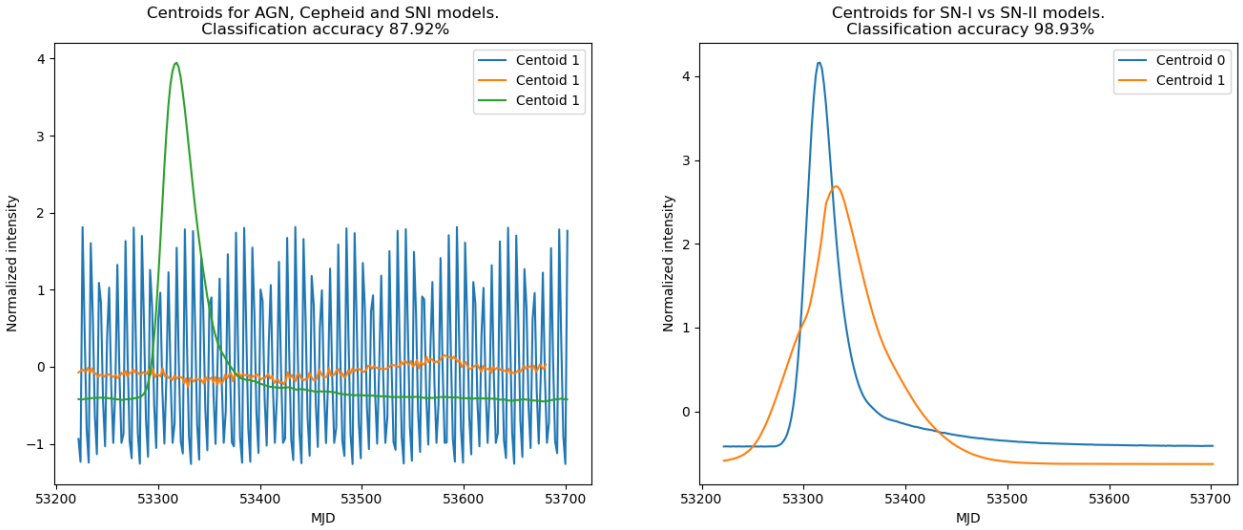


Figure 4.11: The K-Means centroids that best-fit the two different sets of curves: a mixture of AGN, Cepheids and SN-Ia models and a mixture of Type I and Type II SN models. The different model shapes are clearly reflected in the shape of each centroid.

6.2 PLAsTICC

The key difference between the SNANA simulations produced for the purposes of testing in this work and PLASTICC is the survey area and survey cadence. Thus PLAsTICC data will not contain lightcurves sampled every 12h, but with a cadencce that matches that of Rubin survey strategy. Therefore lightcurves are sample on the order of days, in different filters, and have an heteroskedastic measurement errors. Unfortunately this significantly increases the difficulty of correct classification with this approach.

There are two key issues that have to be resolved. Firstly, smoothness of the derivative greatly affects the end result. The smoothness of the derivative is affected by the SNR and cadence of the measurement of the lightcurve and the choice of the algorithm by which they are computed. Using finite difference methods tends to produce "spiky" derivatives that can take large positive or negative values. This will produce very noisy distance matrices where the variance in distances between similar curves can not be easily separated from other classes. Secondly, successful minimization of the given problem suffers as a consequence of unstable derivatives.

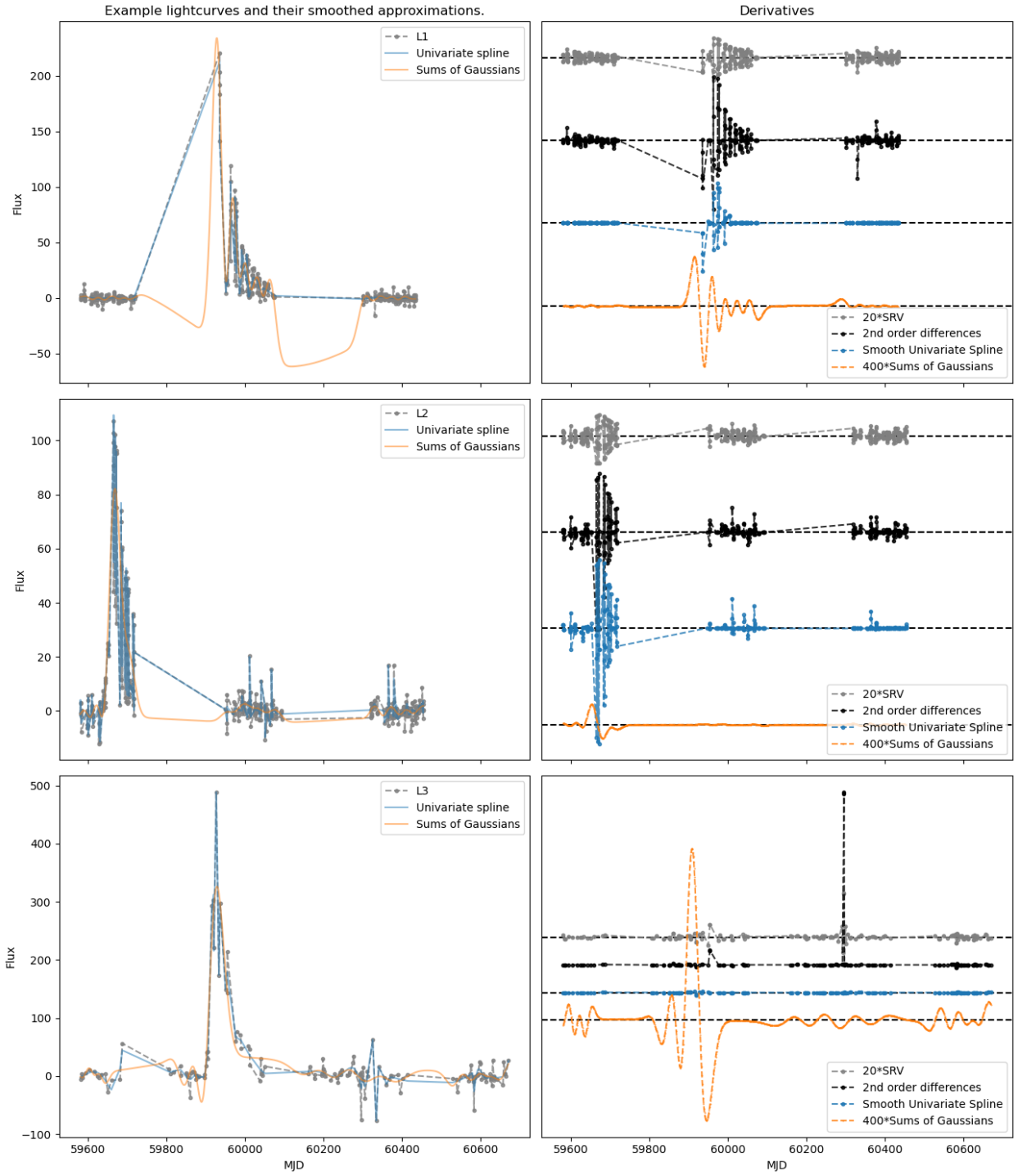


Figure 4.12: Three random lightcurves drawn from the PLAsTICC test dataset on the left, and their derivatives on the right. Depending on the selected algorithm used to calculate the differentials (second order finite difference in blue, univariate interpolating spline in orange and the SRV metric using finite differences in gray) different classifications may be retrieved. Finite difference methods of calculating derivatives are often very noisy and obtain very large values for points with large associated uncertainties. Smoothed approximations to the curves smooth out the large variability in the differential values, but may impose a shape on the given curve that doesn't correspond to the original lightcurve, as shown by the orange line in the left hand side panels.

A number of different minimizers and their parameter values were tested, including varying the value of the penalty weight λ (see Equation 4.15). Of the 15 different SciPy minimizers the *L-BFGS-B* and *SLSQP* appeared to be the most successful. The MINUIT package, a C++ library for numerical optimization and statistical analysis developed and maintained by the European Organization for Nuclear Research (CERN), performs more stable than the SciPy optimizers. In practice MINUIT seemed to successfully minimize the problem more often than SciPy. The expressions for distance were also implemented in Autograd and Jax, libraries for automatic differentiation. However, Jax supported only the *BFGS* minimizer which often seemed to struggle to successfully converge to the solution.

Different attempts at smoothing and resampling the original data were attempted as shown on Figure 4.12. The goal was to resample curves to a derived pre-shape space that is more uniformly and more smoothly sampled than the raw data, in hopes the smoother functional approximation may yield more stable behavior of the derivatives being calculated, resulting in a more robust minimization procedure. Simplistic alignment of the curves was also performed in this process. The global maxima of the curves would be assigned the same coordinate in time and a smoothed function would be evaluated in the observing time window around that point. Of such functions, univariate, smoothing and non-smoothing, interpolating splines of varying order as well as approximations via sums of Gaussians were attempted (shown in left column on Figure 4.12). In the case of sums of Gaussians, the resulting curve depends strongly on the sampling cadence, with potentially rich structure often appearing at time-stamps where no real observations existed. Time-spans in which the object received no observations can be filled with zeros (not shown on the Figure 4.12), which would, for the appropriate smoothing methods such as sums of Gaussians, prevent the smoothing function from obtaining negative values. In practice, however, this most often resulted in classifications to be most sensitive to the cadence of the survey, not necessarily the inherent shape of the curve itself. Interestingly, when two objects were observed in a similar cadence, this procedure would not yield significantly different classification results. Although

the shape of the curve was changed, all curves that were missing data in that timestamp would result in equal change in distance from one-another effectively adding a constant to the distance matrix. Unfortunately, none of the attempts yielded satisfying results and the bulk of the dataset fails minimization. As is shown in the right column of Figure 4.12, the varying choices of smoothing functions as well as the choice of the implementation of the numerical derivatives used to calculate the tangents at points in which we have measured the brightness of the object, yield different results. Particularly in the case of low SNR measurements, flux measurement outliers may result in large derivative values (see bottom right of Figure 4.12). Such outliers result in sub-optimal classification accuracy. Interpolating splines and sums of Gaussians smoothing approaches are able to address these issues, but, as discussed and as shown in top left of Figure 4.12, at a cost of potentially altering the shape of the curve and biasing the classification towards cadence.

The best results were achieved by subselecting lightcurves that belong to SN Type I and Type II classes only. Using SNCosmo (Barbary et al., 2025), a library for supernova cosmology analysis, SALT2 template based lightcurve models were fitted to the LSST r band flux data. The lightcurves were pre-aligned to match the time of their their peak luminosity in the band. Lightcurves were then resampled to the same times, thus approximately aligning them before classification. The alignment is not perfect since the fitting procedure can fail to converge, or to correctly identify the peak luminosity time, but it does decrease the variance between the alignment of the curves. The pre-aligning and resampling was performed in order to minimize the effects of reparametrizations on the curves. No cuts on the quality of the fit were made, except to discard fits that fail with an error. A sample of PLAsTICC data containing 274 SNIa, 245 SNIbc and 267 SNII was selected. Relabeling was performed as previously, the cluster with the largest count was assigned SNI label and all the other branches were assigned SNII label. The classification accuracy achieved was 73.03%. This is statistically significantly better than for expected random chance accuracy of 52%, for a the same number of labels, but falls short of majority of PLAsTICC solutions.

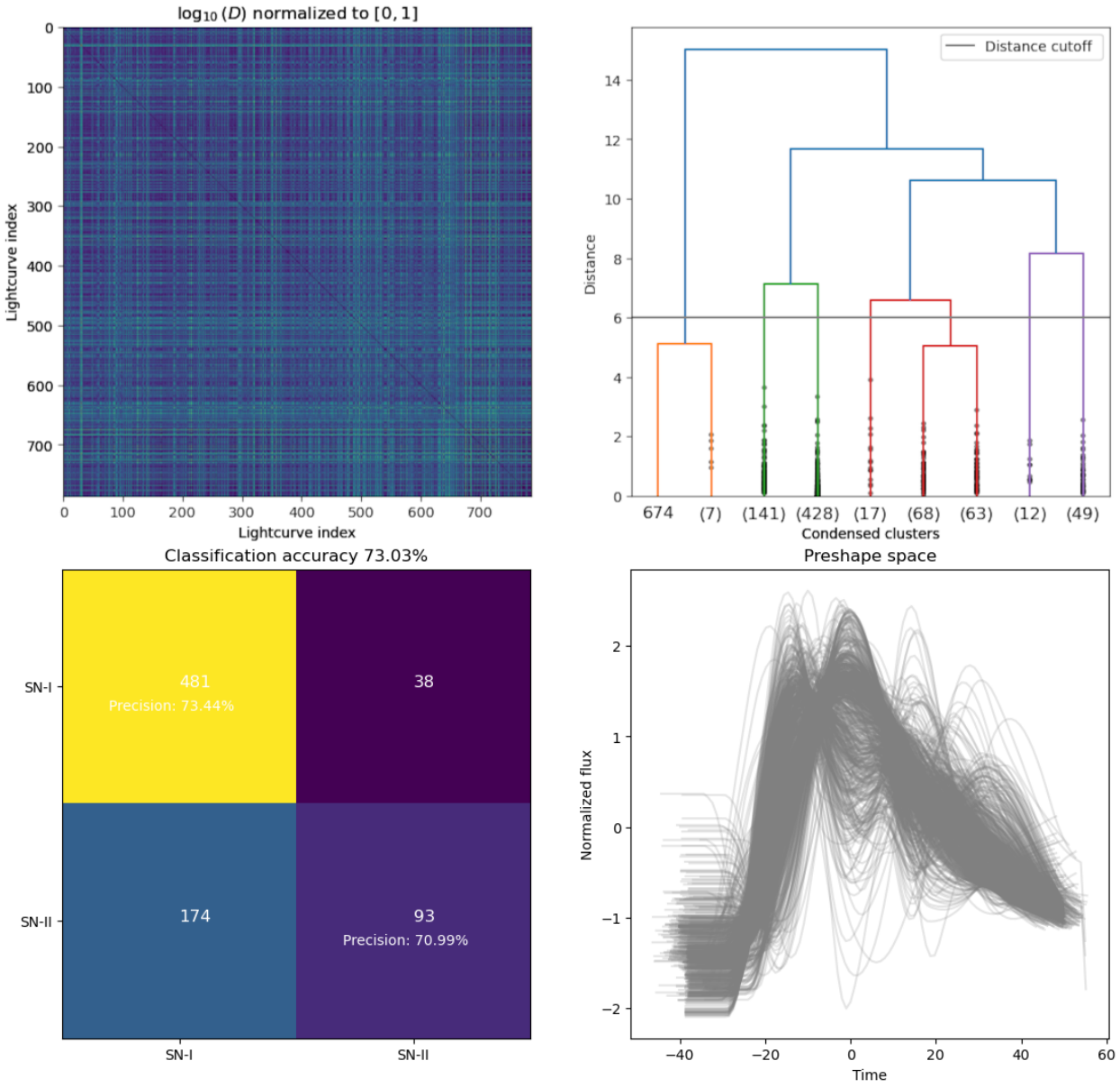


Figure 4.13: Classification of a set of SN Type I and Type II curves. From left to right displayed are the distance matrix, agglomerative clustering dendogram, confusion matrix and the aligned curves in the pre-shape space. Compared to SNANA simulations the much greater variance of the shape of the curves yields a lower classification accuracy, even in the best-case scenario when we focus on classification of SN lightcurves only.

7 Conclusion

A novel approach to describing and comparing lightcurves was investigated. By applying ideas inherited from differential geometry on Riemannian manifolds it is possible to demonstrate how inherent geometrical constraints of the problem, inherited by measured data, can be leveraged to improve on common statistical methods such as K means clustering or PCA. By knowing, or assuming, the manifold on which the data lies, one can show how to derive the exponential and logarithm maps of a data point on the manifold to its tangential space and, in some cases, derive the metric on the manifold itself. The metric induces a distance measure on the manifold that is then used to calculate commonly used statistics such as the mean. The benefit of this approach is that the metric is a property of the manifold itself and will thus reflect the constraints on the possible measurable values imposed by the manifold.

We applied this approach on lightcurves of objects. Lightcurves are treated as elements of the space absolutely continuous real-valued $\mathrm{AC}_0[0,1],\mathbb{R})$ functions, an infinitely dimensional manifold. The square velocity function (SRV) transform is chosen to induce a distance on $(\mathrm{AC}_0[0,1],\mathbb{R})$, motivated by the successful application of the SRV transform in the field of computer vision. The SRV distance is demonstrated to successfully classify a high SNR, well-sampled set of lightcurves produced for this work using SNANA simulations modeled after the PLAsTICC dataset. Three different classification schemes are investigated (Gaussian Mixture Models applied on the histogram of distances, Agglomerative (hierarchical) clustering and K means clustering) on subsets of the simulated lightcurves. The test subsets of lightcurves contained different classes of objects (AGN, Cepheids and SN1a models in one test case, and SNIa-SALT2, SN1b-Templates and SNII-Templates models), with a different total number of included lightcurves and in different proportions of the number of included lightcurves within each test dataset. All approaches yielded relatively high classification accuracy rates in all cases, especially on the high SNR validation dataset simulated with SNANA. The SRV distance produces excellent characterizations of different shapes

of lightcurves, often resulting in the ability to detect smaller subsets within each individual class. Approaches such as agglomerative clustering, or K-means, must produce a fine enough segmentation to capture the full variability of the shapes of a particular class of objects with sufficiently tight precision. Thus, derived labels must then be re-labeled to the target classification. We demonstrate the effectiveness of this approach by classifying a small dataset of lightcurves to derive the initial decision tree, which is then expanded without repartitioning by a significantly larger dataset, and yielding a comparatively high classification accuracy.

While interesting, in the sense of fine-grained comparisons and fine-grained selection of very particular shapes of lightcurves from a set of well sampled effectively noiseless set of curves, we demonstrate the negative aspects of this approach. Classification of more realistic lightcurves from the PLAsTICC challenge do not produce accurate classifications. The reparametrizations of lightcurves, as a result of differing observing cadence, is identified as a key issue of classification of realistic lightcurves. The first common approach to rendering the SRV metric invariant to reparametrizations is to construct a space of pre-shapes (normalized, translated, elastically deformed and otherwise reparametrized sets of curves). Such an approach is usually possible when curves contain easily identifiable features on which they might be aligned on. A second common approach is to identify a group action on the manifold that partitions the manifold into classes of equivalencies and perform the minimization of the SRV distance the constructed quotient space. The lightcurves are thus treated as elements of the quotient space $\mathcal{S} := \mathrm{AC}_0([0,1],\mathbb{R})/\mathrm{Diff}([0,1])$, that is, the space of parametrized light curves modulo reparametrizations that are modeled as diffeomorphisms of the parameter space $[0,1]$. We identify the varifold similarity metric as an appropriate metric to induce the distance on the quotient space due to its invariance to reparametrizations. A minimization approach is developed that involves finding a test curve f that has the same parametrization and minimizes the SRV distance to a source lightcurve f_1, while being equal to the target lightcurve f_2. To find the geodesic distance between two differently parameterized lightcurves we minimize Equation 4.31 in which the SRV distance $d_Q(f_1, f)^2$

is given by Equation 4.10 and the varifold similarity metric $\|\mu_f - \mu_{f_2}\|^2_{V^*}$ by Equation 4.30. A number of different approaches to this implementation were attempted, but no numerically stable, sufficiently accurate solution was found. The best results were achieved only for a subset of all available models in the PLAsTICC dataset, after creating a preshape space of at least approximately aligned curves. Only SN curves were considered and only after fitting a SALT-2 template based model and approximately aligning them based on the time of maximum luminosity. The process of detecting and aligning features of two curves is challenging in the case of low SNR lightcurves and often yields poorly approximated smoothing functions and numerically unstable derivatives that prevent successful optimization solution. When applied on a subset of just the supernovae lightcurves from the PLAsTICC data we are able to achieve a 73.03% classification accuracy.

The key issues of classification of real lightcurves are related to the noise associated with individual measurements as well as, usually, insufficient sampling cadence resulting in a poorly defined shape of the curve. Although the application on realistic lighcurves was not very successful, the approach shows promise and the field continues to rapidly improve. Since the attempts described in this chapter, new ways of quotienting out reparametrizations were found (Brigant, 2018) that are significantly simpler than the attempted approach. In combination with a more appropriate differentiation implementation (Knowles & Renka, 2014) of data with uncertainties and the application to a different, or perhaps a more specific problem, remains possible. Ultimately, in order to compete with current methods of lightcurve classifications, it is likely that the proposed approach will likely have to be used in addition to or as a part of a larger classification scheme such that more of the available data may be utilized. Performing the clustering including distances of a single object across different filters is an example of such an approach as was discussed earlier in this chapter, since some objects exhibit distinct and predictable behavior across the different observing bands.

CHAPTER 5

CONCLUSIONS

Figure 5.1: A coadd, covering approximately a 4"x4" area, constructed from 101 overlapping images in the DEEP dataset centered on the $(\alpha, \delta) = (352.8°, -4.5°)$ in the constellation of Aquarius. There are no known large scale structures in the area indicating that the low surface brightness structure streaking the image is likely the galactic cirrus even at a galactic latitude of $b = -60.4$. The approximate limiting magnitude of the image is 26.5.

Rubin will deliver data of unprecedented quality (see Figure 5.1) covering the southern hemisphere of the sky, for 10 years. The Rubin Science Pipelines are state of the art image analysis pipelines capable of processing this data. The quality and the quantity of the data that will drive scientific discovery for the next decade and possibly longer. But this data comes at a cost. The large data volumes raises barriers of entry, decelerate "time to science", and raise costs. The unprecedented depth of the coadds and the fast observing cadence, enable (sometimes require) novel data analysis approaches.

In Chapter 2 we examined if cloud computing could alleviate some of these issues. Adopting a lift-and-shift strategy, we piece-wise added support for cloud services to the Rubin Science Pipelines and Middleware components. We demonstrated that costs of processing a dataset remain approximately equal for close-to optimal deployments, at varying sizes of deployed resources due to the cost-scaling model adopted by cloud providers. Put simply, the cost to process the same dataset remains the same whether it's processed in an hour on many resources or in a week on a few. This does not apply when resource deployment is not optimal, such when more resources are allocated than there are jobs in the workflow. Adding support for cloud services to Rubin Middleware component allowed the Interim Data Facility to be deployed on Google Cloud Services for the duration of 3 years, at an undisclosed cost, and near the peak of the global pandemic when acquiring and deploying resources on premise faced delays. We show how the cloud does not remain cost competitive with on-premise resources over long periods of time. The limiting factor that determines the duration of the time cloud computing remains cost effective is set by: a) the total volume of data, b) the volume of data stored in high-availability storage types and c) user data access patterns. The cost of long-term high-availability large-scale data storage will quickly out-pace other costs and the costs of data access are levied against the data-owner and not the user. Cloud computing remains cost-competitive in two scenarios. First, when the output of processing is significantly smaller than the input dataset, because uploading data to the cloud service providers remains free of charge. Second, when the time-to-results offsets total cost of the

workforce and cost of purchasing, deploying and maintaining an on-premise solution. Then many factors such as total data volumes, user data access patterns, ability to deploy tiered-storage solutions, ability to deploy on-premise solution, cost of long-term maintenance of those resources etc. have to be considered. In this scenario, the period of time that cloud services can remain competitive is often measured in a few (1-5) years.

In Chapter 3 we focused on extracting more science from the same dataset, by applying novel moving object techniques. We apply KBMOD, a shift and stack moving object detection algorithm, to the DEEP dataset. We discuss the performance improvements and functionality added to KBMOD required to be able to process Rubin data. Several key novel functionalities were developed to facilitate the integration of KBMOD and the Rubin Science Pipelines: the pipelining of KBMOD search execution, via abstractions of image collections and work units, the image resampling capability, where an image is resampled onto a new, overlapping, coordinate grid, and a fast analytic solution for the correction of the reflex motion of the Earth, which in tandem with resampling enables KBMOD to linearize the apparent motion of object over longer time periods than a couple of days. The cumulative effect of the performance improvements is the ability to execute KBMOD on $\approx$ 20 times larger dataset, while simultaneously being able to increase the range of searched angles by 100% and the range of searched velocities by an additional 38% compared to earlier work by Smotherman et al. (2023). We achieve a 10% higher peak detection efficiency, but a 0.3 magnitude lower limiting magnitude m_{25} at which 25% of the objects were recovered. We identify the higher filtering threshold values, chosen due to a large number of estimated returned results, as the key culprit for the loss of m_{25} limiting magnitude. Due to the loss of m_{25} limiting magnitude we expect to recover 77% of the objects compared to the previous processing within the overlapping data. Analysis of our results retrieved by linking of tracklets with HelioLinC in the range of 20 to 100 AU returns 44 detections consistent with the orbit of a new, not previously discovered, object. The orbital parameters of these objects are consistent with objects found by Smotherman et al. (2023). The recovered linkages have

orbital parameters consistent with the injected distribution of simulated KBO objects. The average observed length of arc of approximately 1 year and, approximately, 50% of recovered linkages have an observed length of arc of 2 years. Thirty previously known objects are recovered in the search. Further analysis of the potentially new detections is currently being performed.

In the final Chapter 4, we investigate a novel approach to describing and comparing lightcurves. In contrast to current best-performing lightcurve classifiers, we attempt to construct a way to perform an unsupervised classification of lightcurves. We identify lightcurves as elements of the space of absolutely continuous real-valued functions on the unit interval ($\mathrm{AC}_0([0,1],\mathbb{R})$). We identified that space as a manifold and chose the SRV distance as an appropriate metric to compare its elements. The constructed manifold and its metric are not invariant to reparameterizations, so we define a relaxed matching approach that allows for an additional measure of similarity to be used. Constructing a quotient preshape space $\mathcal{S} := \mathrm{AC}_0([0,1],\mathbb{R})/\mathrm{Diff}([0,1])$ of all curves modulo reparameterization we compare a test function, parameterized equally to the first lightcurve, and to the second, not equally parameterized, lightcurve. We solve this problem by simultaneously minimizing the SRV distance to the first lightcurve and the varifold similarity measure to the second lightcurve. By simulating equally parameterized, high SNR, lightcurves using SNANA we demonstrate the effectiveness of this approach by classifying the light curves with a high degree of accuracy and precision. We implement 3 different unsupervised approaches to classification of lightcurves: a) Gaussian mixture models, b) agglomerative (hierarchical) clustering and c) k-means clustering based on the Frechet mean. We attempt to apply the same approach to more realistic, non-equally parameterized, low SNR, lightcurve data used in the PLAsTiCC challenge. We discuss the difficult problem of successfully minimizing the cost function due to numerical instabilities, heteroskedastic uncertainties and uneven sampling of the lightcurves. We do not achieve classification accuracy comparable to current approaches between two co-mixed types of SN1 and SN2 supernovae lightcurves. With an achieved classification accuracy of

73%, however, there seems to be a non-negligible predictive power even when applied to realistic data. We discuss how the next steps should be to integrate this approach as an additional step within the current best-performing supervised classification algorithms and search for potential improvements to their performance.

BIBLIOGRAPHY

Abbena, E., Salamon, S., & Gray, A. 2006, Modern Differential Geometry of Curves and Surfaces with Mathematica, Third Edition, Textbooks in Mathematics (Taylor & Francis). https://books.google.com/books?id=owEj9TMYo7IC

Adolphson, M., Cantu, S., Montgomery, K., & Renshaw, T. 2015, Minor Planet Bulletin, 42, 25

Aihara, H., Arimoto, N., Armstrong, R., & et al. 2018, Publications of the Astronomical Society of Japan, 70, S4, doi: 10.1093/pasj/psx066

Allen, R. L., Bernstein, G. M., & Malhotra, R. 2002, AJ, 124, 2949, doi: 10.1086/343773

and, J. M. 1982, The American Statistician, 36, 15, doi: 10.1080/00031305.1982.10482771

Aronszajn, N. 1950, Transactions of the American Mathematical Society, 68, 337. http://www.jstor.org/stable/1990404

Bailer-Jones, C. A. L., Rybizki, J., Fouesneau, M., Demleitner, M., & Andrae, R. 2021, The Astronomical Journal, 161, 147, doi: 10.3847/1538-3881/abd806

Bannister, M. T., Shankman, C., Volk, K., et al. 2017, The Astronomical Journal, 153, 262, doi: 10.3847/1538-3881/aa6db5

Barbary, K., Bailey, S., Barentsen, G., et al. 2025, SNCosmo, v2.12.0, Zenodo, doi: 10.5281/zenodo.14714968

Batygin, K., & Brown, M. E. 2016, The Astronomical Journal, 151, 22, doi: 10.3847/0004-6256/151/2/22

Becker, J. C., Khain, T., Hamilton, S. J., et al. 2018, The Astronomical Journal, 156, 81, doi: 10.3847/1538-3881/aad042

Ben-Hur, A., Horn, D., Siegelmann, H. T., & Vapnik, V. 2002, J. Mach. Learn. Res., 2, 125–137

Bernardinelli, P. H., Bernstein, G. M., Sako, M., et al. 2020a, The Astrophysical Journal Supplement Series, 247, 32, doi: 10.3847/1538-4365/ab6bd8

—. 2020b, The Astrophysical Journal Supplement Series, 247, 32, doi: 10.3847/1538-4365/ab6bd8

—. 2022, The Astrophysical Journal Supplement Series, 258, 41, doi: 10.3847/1538-4365/ac3914

Bernardinelli, P. H., Smotherman, H., Langford, Z., et al. 2024, AJ, 167, 134, doi: 10.3847/1538-3881/ad1527

Bernstein, D. 2014, IEEE Cloud Computing, 1, 81

Bernstein, G., & Khushalani, B. 2000, The Astronomical Journal, 120, 3323, doi: 10.1086/316868

Berthier, J., Vachier, F., Thuillot, W., et al. 2006, in Astronomical Society of the Pacific Conference Series, Vol. 351, Astronomical Data Analysis Software and Systems XV, ed. C. Gabriel, C. Arviset, D. Ponz, & S. Enrique, 367

Bianco, F. B., Ivezić, Ž., Jones, R. L., et al. 2022, ApJS, 258, 1, doi: 10.3847/1538-4365/ac3e72

Binzel, R., Reddy, V., & Dunn, T. 2015, The Near-Earth Object Population: Connections to Comets, Main-Belt Asteroids, and Meteorites (University of Arizona), 243–257, doi: 10.2458/azu_uapress_9780816532131-ch013

Boone, K. 2019, The Astronomical Journal, 158, 257, doi: 10.3847/1538-3881/ab5182

Bosch, J., AlSayyad, Y., Armstrong, R., & et al. 2018, An Overview of the LSST Image Processing Pipelines. https://arxiv.org/abs/1812.03248

Bosch, J., Armstrong, R., Bickerton, S., & et al. 2017, Publications of the Astronomical Society of Japan, 70, doi: 10.1093/pasj/psx080

Bosch, J., Armstrong, R., Bickerton, S., et al. 2018, PASJ, 70, S5, doi: 10.1093/pasj/psx080

Bosch, J., AlSayyad, Y., Armstrong, R., et al. 2019, in Astronomical Society of the Pacific Conference Series, Vol. 523, Astronomical Data Analysis Software and Systems XXVII, ed. P. J. Teuben, M. W. Pound, B. A. Thomas, & E. M. Warner, 521, doi: 10.48550/arXiv.1812.03248

Brasser, R., Morbidelli, A., Gomes, R., Tsiganis, K., & Levison, H. F. 2009, Astronomy and Astrophysics, 507, 1053–1065, doi: 10.1051/0004-6361/200912878

Brigant, A. L. 2018, A discrete framework to find the optimal matching between manifold-valued curves. https://arxiv.org/abs/1703.05107

Butler, M., Lim, K. T., & O'Mullane, W. 2020, DM sizing model and purchase plan for the remainder of construction, https://dmtn-135.lsst.io

Calabretta, M. R., & Greisen, E. W. 2002, A&A, 395, 1077, doi: 10.1051/0004-6361:20021327

Caplar, N., Beebe, W., Branton, D., et al. 2025, arXiv e-prints, arXiv:2501.02103, doi: 10.48550/arXiv.2501.02103

Chambers, K. C., Magnier, E. A., Metcalfe, N., et al. 2019, The Pan-STARRS1 Surveys. https://arxiv.org/abs/1612.05560

Chard, K., Babuji, Y., Woodard, A., et al. 2021, ACM SIGAda Ada Letters, 40, 73–75, doi: 10.1145/3463478.3463486

Chaudhuri, K., Dasgupta, S., Kpotufe, S., & von Luxburg, U. 2014, Consistent procedures for cluster tree estimation and pruning. https://arxiv.org/abs/1406.1546

Chiang, H.-F., Bektesevic, D., & the AWS-PoC team. 2020, AWS Proof of Concept Project Report, https://dmtn-137.lsst.io

Chiang, H.-F., & Thrush, S. 2020, S18 HSC PDR1 reprocessing, https://dmtn-160.lsst.io

Cowling, M. G. 1983, Annals of Mathematics, 117, 267. http://www.jstor.org/stable/2007077

Dawson, R. I., & Murray-Clay, R. 2012, The Astrophysical Journal, 750, 43, doi: 10.1088/0004-637x/750/1/43

Dean, J., & Ghemawat, S. 2008, Communications of the ACM, 51, 107, doi: 10.1145/1327452.1327492

Deelman, E., Vahi, K., Juve, G., & et al. 2015, Future Generation Computer Systems, 46, 17, doi: 10.1016/j.future.2014.10.008

DeForest, C. E. 2004, Sol. Phys., 219, 3, doi: 10.1023/B:SOLA.0000021743.24248.b0

DeMeo, F. E., Alexander, C. M. O., Walsh, K. J., Chapman, C. R., & Binzel, R. P. 2015, The Compositional Structure of the Asteroid Belt (University of Arizona Press), 13–42. http://www.jstor.org/stable/j.ctt18gzdvc.8

Dones, L., Weissman, P. R., Levison, H. F., Duncan, M. J., & Binzel, R. P. 2004, Oort Cloud Formation and Dynamics (University of Arizona Press), 153–174. http://www.jstor.org/stable/j.ctv1v7zdq5.17

Dubois-Felsmann, G. 2017, SuperTask Architecture and Design, https://dmtn-055.lsst.io

Duncan, M., Levison, H., Dones, L., & Binzel, R. P. 2004, Dynamical Evolution of Ecliptic Comets (University of Arizona Press), 193–204. http://www.jstor.org/stable/j.ctv1v7zdq5.19

Duncan, M. J., Levison, H. F., & Budd, S. M. 1995, The Astronomical Journal, 110, 3073, doi: 10.1086/117748

Eichfelder, G., Hotz, T., & Wieditz, J. 2018, An algorithm for computing Fréchet means on the sphere. https://arxiv.org/abs/1802.09243

Fernández, J., & Ip, W.-H. 1984, Icarus, 58, 109, doi: 10.1016/0019-1035(84)90101-5

Flaugher, B., Diehl, H. T., Honscheid, K., et al. 2015, The Astronomical Journal, 150, 150, doi: 10.1088/0004-6256/150/5/150

Fraser, W. C., Dones, L., Volk, K., et al. 2024, The Transition from the Kuiper Belt to the Jupiter-Family Comets (University of Arizona Press), 121–152. http://www.jstor.org/stable/jj.21819446.11

Fraser, W. C., Kavelaars, J., Holman, M., et al. 2008, Icarus, 195, 827–843, doi: 10.1016/j.icarus.2008.01.014

Fraser, W. C., Porter, S. B., Peltier, L., et al. 2024, Planetary Science Journal, 5, 227, doi: 10.3847/PSJ/ad6f9e

Gaia Collaboration, Vallenari, A., Brown, A. G. A., et al. 2023, A&A, 674, A1, doi: 10.1051/0004-6361/202243940

Gerdes, D. W., Jennings, R. J., Bernstein, G. M., et al. 2016, The Astronomical Journal, 151, 39, doi: 10.3847/0004-6256/151/2/39

Gerdes, D. W., Sako, M., Hamilton, S., et al. 2017, The Astrophysical Journal, 839, L15, doi: 10.3847/2041-8213/aa64d8

Gomes, R., Levison, H. F., Tsiganis, K., & Morbidelli, A. 2005, Nature, 435, 466, doi: 10.1038/nature03676

Gorodski, C. 2016, An introduction to Riemannian geometry (Preliminary version), https://www.ime.usp.br/~gorodski/teaching/mat5771-2021/riem-geom-gorodski-2016.pdf

Greisen, E. W., & Calabretta, M. R. 2002, A&A, 395, 1061, doi: 10.1051/0004-6361:20021326

Hahn, J. M., & Malhotra, R. 2005, The Astronomical Journal, 130, 2392, doi: 10.1086/452638

Hartman, E., Sukurdeep, Y., Charon, N., Klassen, E., & Bauer, M. 2021, in 2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW) (IEEE), 4420–4428, doi: 10.1109/cvprw53098.2021.00499

Hartman, E., Sukurdeep, Y., Klassen, E., Charon, N., & Bauer, M. 2023, International Journal of Computer Vision, 131, 1183–1209, doi: 10.1007/s11263-022-01743-0

He, K., Zhang, X., Ren, S., & Sun, J. 2015, Deep Residual Learning for Image Recognition. https://arxiv.org/abs/1512.03385

Heinze, A. N., Metchev, S., & Trollo, J. 2015, The Astronomical Journal, 150, 125, doi: 10.1088/0004-6256/150/4/125

Hložek, R., Ponder, K. A., Malz, A. I., et al. 2020, Results of the Photometric LSST Astronomical Time-series Classification Challenge (PLAsTiCC). https://arxiv.org/abs/2012.12392

Holman, M. J., Payne, M. J., Blankley, P., Janssen, R., & Kuindersma, S. 2018, AJ, 156, 135, doi: 10.3847/1538-3881/aad69a

Holzman, B., Bauerdick, L. A. T., Bockelman, B., & et al. 2017, arXiv e-prints, arXiv:1710.00100. https://arxiv.org/abs/1710.00100

Ivezić, Ž., Kahn, S. M., Tyson, J. A., et al. 2019, ApJ, 873, 111, doi: 10.3847/1538-4357/ab042c

Jakobson, D., Nadirashvili, N., & Polterovich, I. 2005, Extremal metric for the first eigenvalue on a Klein bottle. https://arxiv.org/abs/math/0311484

Jenness, T., Bosch, J. F., Schellart, P., & et al. 2018, Abstracting the storage and retrieval of image data at the LSST. https://arxiv.org/abs/1812.08085

Jenness, T., Bosch, J. F., Salnikov, A., et al. 2022, in Society of Photo-Optical Instrumentation Engineers (SPIE) Conference Series, Vol. 12189, Software and Cyberinfrastructure for Astronomy VII, 1218911, doi: 10.1117/12.2629569

Jewitt, D., Hsieh, H. H., & Dotson, R. 2024, The Asteroid-Comet Continuum (University of Arizona Press), 767–798. http://www.jstor.org/stable/jj.21819446.29

Jílková, L., Portegies Zwart, S., Pijloo, T., & Hammer, M. 2015, Monthly Notices of the Royal Astronomical Society, 453, 3158, doi: 10.1093/mnras/stv1803

Jones, R. L., Gladman, B., Petit, J.-M., et al. 2006, Icarus, 185, 508, doi: 10.1016/j.icarus.2006.07.024

Joshi, S., Klassen, E., Srivastava, A., & Jermyn, I. 2007, in Proceedings / CVPR, IEEE Computer Society Conference on Computer Vision and Pattern Recognition. IEEE Computer Society Conference on Computer Vision and Pattern Recognition, Vol. 4679, 387–398, doi: 10.1007/978-3-540-74198-5_30

Jyothirmayi, T., Rao, K., Rao, P. S. R., & Satyanarayana, C. 2016, International Journal of Electrical and Computer Engineering (IJECE), 6, 2188, doi: 10.11591/ijece.v6i5.pp2188-2196

Kessler, R., Bernstein, J. P., Cinabro, D., et al. 2009, Publications of the Astronomical Society of the Pacific, 121, 1028–1035, doi: 10.1086/605984

Kessler, R., Bassett, B., Belov, P., et al. 2010, Publications of the Astronomical Society of the Pacific, 122, 1415–1431, doi: 10.1086/657607

Kessler, R., Narayan, G., Avelino, A., et al. 2019, Publications of the Astronomical Society of the Pacific, 131, 094501, doi: 10.1088/1538-3873/ab26f1

Knowles, I., & Renka, J. 2014, Electronic Journal of Differential Equations (EJDE) [electronic only], 2014, 235. http://ejde.math.txstate.edu

Krish, K. R., Khasymski, A., Wang, G., Butt, A. R., & Makkar, G. 2013, in 2013 IEEE International Conference on Big Data, 313–318, doi: 10.1109/BigData.2013.6691589

Levison, H. F., & Dones, L. 2007, in Encyclopedia of the Solar System, ed. L. A. A. McFadden, P. R. Weissman, & T. V. Johnson (Elsevier Inc.), 575–588, doi: 10.1016/B978-012088589-3/50035-9

Levison, H. F., Morbidelli, A., VanLaerhoven, C., Gomes, R., & Tsiganis, K. 2008, Icarus, 196, 258, doi: 10.1016/J.ICARUS.2007.11.035

Lim, K.-T., Guy, L., , & Chiang, H.-F. 2019, LSST + AWS Proof of Concept, https://dmtn-114.lsst.io

Mahalanobis, P. C. 1936, Proceedings of the National Institute of Sciences of India, 2, 49. https://api.semanticscholar.org/CorpusID:239595337

Malhotra, R. 1993, Nature, 365, 819, doi: 10.1038/365819a0

Morbidelli, A., Brown, M. E., & Binzel, R. P. 2004, The Kuiper Belt and the Primordial Evolution of the Solar System (University of Arizona Press), 175–192. http://www.jstor.org/stable/j.ctv1v7zdq5.18

Morbidelli, A., Tsiganis, K., Crida, A., Levison, H. F., & Gomes, R. 2007, The Astronomical Journal, 134, 1790–1798, doi: 10.1086/521705

Napier, K. J., Lin, H.-W., Gerdes, D. W., et al. 2023, The DECam Ecliptic Exploration Project (DEEP): V. The Absolute Magnitude Distribution of the Cold Classical Kuiper Belt, arXiv, doi: 10.48550/arXiv.2309.09478

Nesvorný, D., & Vokrouhlický, D. 2016, The Astrophysical Journal, 825, 94, doi: 10.3847/0004-637X/825/2/94

Nguyen, T., Woods, D. F., Ruprecht, J., & Birge, J. 2024, AJ, 167, 113, doi: 10.3847/1538-3881/ad20e0

Oklobdzija, V. 2001, The Computer Engineering Handbook, Computer Engineering Handbook 2e (Taylor & Francis). https://books.google.com/books?id=38Aj3CjHgc8C

Olsen, N. L., Markussen, B., & Raket, L. L. 2018, Journal of the Royal Statistical Society Series C: Applied Statistics, 67, 1147–1176, doi: 10.1111/rssc.12276

Posey, B., Gropp, C., Wilson, B., & et al. 2018, in 2018 18th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (CCGRID), 253–262

Reddy, P. P., Rao, S. K., & Yarramalle, S. 2007, International Journal of Computer Science and Network Security, 7, 317

Reddy, V., Dunn, T. L., Thomas, C. A., Moskovitz, N. A., & Burbine, T. H. 2015, Mineralogy and Surface Composition of Asteroids (University of Arizona Press), 43–64. http://www.jstor.org/stable/j.ctt18gzdvc.9

Rivkin, A. S., Campins, H., Emery, J. P., et al. 2015, Astronomical Observations of Volatiles on Asteroids (University of Arizona Press), 65–88. http://www.jstor.org/stable/j.ctt18gzdvc.10

Schwamb, M. E., Rabinowitz, D. L., Tourtellotte, S., Hadjiyska, E. I., & Brasser, R. 2012, in LPI Contributions, Vol. 1667, Asteroids, Comets, Meteors 2012, ed. LPI Editorial Board, 6288

Sfiligoi, I., Wuerthwein, F., Riedel, B., & Schultz, D. 2020, arXiv e-prints, arXiv:2002.06667. https://arxiv.org/abs/2002.06667

Shober, P. M., Tancredi, G., Vaubaillon, J., et al. 2024, Astronomy and Astrophysics, 687, A181, doi: 10.1051/0004-6361/202449635

Singal, A. K. 2015, Astrophysics and Space Science, 359, doi: 10.1007/s10509-015-2476-3

Smola, A., & Vishwanathan, S. 2004, in Advances in Neural Information Processing Systems, ed. L. Saul, Y. Weiss, & L. Bottou, Vol. 17 (MIT Press). https://proceedings.neurips.cc/paper_files/paper/2004/file/058d6f2fbe951a5a56d96b1f1a6bca1c-Paper.pdf

Smotherman, H., Connolly, A. J., Kalmbach, J. B., et al. 2021, The Astronomical Journal, 162, 245, doi: 10.3847/1538-3881/ac22ff

Smotherman, H., Bernardinelli, P. H., Portillo, S. K. N., et al. 2023, The DECam Ecliptic Exploration Project (DEEP) VI: first multi-year observations of trans-Neptunian objects. https://arxiv.org/abs/2310.03678

Srinivas, Y., & Rao, K. 2007, Current Science, 93

Srivastava, A., Klassen, E., Joshi, S. H., & Jermyn, I. H. 2011, IEEE Transactions on Pattern Analysis and Machine Intelligence, 33, 1415, doi: 10.1109/TPAMI.2010.184

Standard, I. O. G. 2024, IEEE/Open Group Std 1003.1-2024 (Revision of IEEE Std 1003.1-2017), 1, doi: 10.1109/IEEESTD.2024.10555529

Stonebraker, M. 1986, IEEE Database Eng. Bull., 9, 4. http://sites.computer.org/debull/86MAR-CD.pdf

Sánchez, B. O., Kessler, R., Scolnic, D., et al. 2022, The Astrophysical Journal, 934, 96, doi: 10.3847/1538-4357/ac7a37

Tang, X., Kandemir, M., Zhao, H., Jung, M., & Karakoy, M. 2019, Performance Evaluation Review, 47, 27, doi: 10.1145/3309697.3331487

Thain, D., Tannenbaum, T., & Livny, M. 2005, Concurrency and Computation: Practice and Experience, 17, 323, doi: 10.1002/cpe.938

Trilling, D. E., Gerdes, D. W., Jurić, M., et al. 2024, AJ, 167, 132, doi: 10.3847/1538-3881/ad1529

Trujillo, C. A., Jewitt, D. C., & Luu, J. X. 2001, The Astronomical Journal, Volume 122, Issue 1, pp. 457-473., 122, 457, doi: 10.1086/321117

Trujillo, C. A., Fuentes, C., Gerdes, D. W., et al. 2024, AJ, 167, 133, doi: 10.3847/1538-3881/ad1523

Tsiganis, K., Gomes, R., Morbidelli, A., & Levison, H. F. 2005, Nature, 435, 459, doi: 10.1038/nature03539

Turaga, P., & Srivastava, A. 2015, Riemannian Computing in Computer Vision (Springer International Publishing). https://books.google.com/books?id=rffpCgAAQBAJ

Whidden, P. J., Bryce Kalmbach, J., Connolly, A. J., et al. 2019, The Astronomical Journal, 157, 119, doi: 10.3847/1538-3881/aafd2d

Yoo, A. B., Jette, M. A., & Grondona, M. 2003, in Job Scheduling Strategies for Parallel Processing, ed. D. Feitelson, L. Rudolph, & U. Schwiegelshohn (Berlin, Heidelberg: Springer Berlin Heidelberg), 44–60

York, D. G., & SDSS Collaboration. 2000, The Astronomical Journal, 120, 1579, doi: 10.1086/301513

Younes, L. 1998, SIAM Journal on Applied Mathematics, 58, 565. http://www.jstor.org/stable/118345

Zahn, C. 1971, IEEE transactions on computers, C-20, 68

Zečević, P., Slater, C. T., Jurić, M., et al. 2019, The Astronomical Journal, 158, 37, doi: 10.3847/1538-3881/ab2384