Kepler’s Sieve

Learning Asteroid Orbits from Telescopic Observations

Rycroft Group Meeting: 30-Mar-2021

Michael S. Emanuel

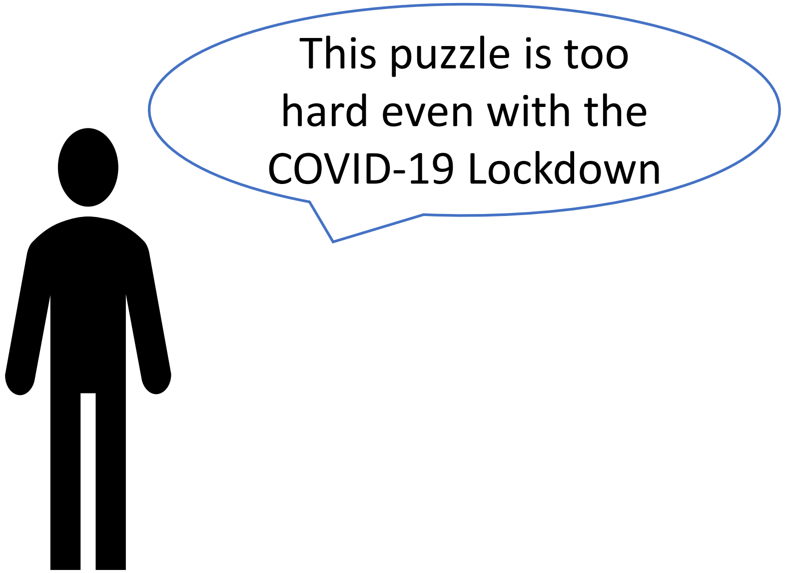
Acknowledgments

- Masters Thesis Advisor: Pavlos Protopapas
- PhD Advisor: Chris Rycroft

Introduction

The Asteroid Search Problem

- Many asteroids (about 958,000 known) in the Solar System
- We want to learn their orbits
- Biggest data source: telescope detections
- Easy once you know which detection matches which asteroid
- This is like a jigsaw puzzle with millions of pieces!



Combining Tracklets vs. Orbital Element Search

- “Tracklet”: two detections close to each other in time and direction
- Existing search methods: greedy search over tracklets
  - Try to extrapolate a tracklet to find additional detections
  - Attempt to fit an orbit when you have enough tracklets
- Drawbacks
  - Myopic – can only connect detections made close in time
  - Suffers from combinatorial explosion
- Proposed novel method: search Orbital Elements
  - 6D space; large, but scales well
  - Cost scales as $N_{\mathrm{ast}} \cdot N_{\mathrm{obs}}$ rather than N_{obs}^r
  - But can we make it work?

Search Overview

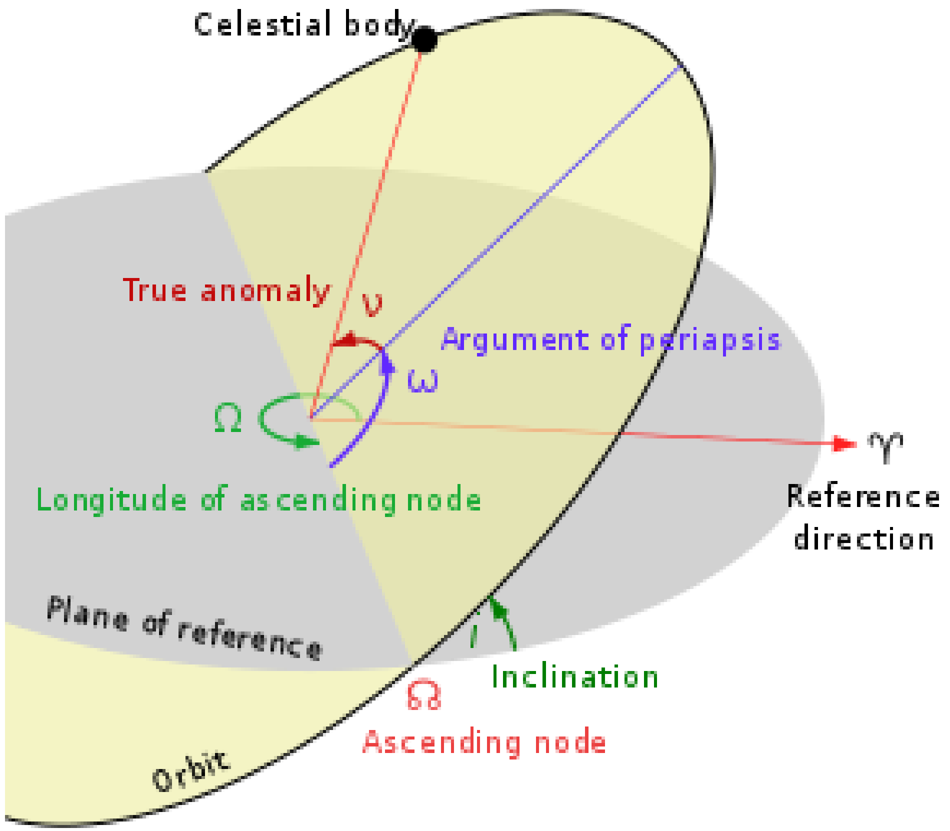
- Initialize candidate orbital elements a, e, i, Ω, ω, f
- Mixture parameters: N_h, R, τ
- Compute position $\mathbf{q}$ and velocity $\mathbf{v}$ from candidate elements
- Compute direction $\mathbf{u}_{pred}$ from $\mathbf{q}$, $\mathbf{v}$; include light time and topos
- Compute distance s from $\mathbf{u}_{pred}$ to $\mathbf{u}_{obs}$ for ZTF observations
- Compute log likelihood $\mathscr{L}_i$ for each candidate element
- Gradient descent…
- The rest is details! Which you will now hear all about…

Integrating the Solar System

REBOUND Integrator for N-Body Problem

- REBOUND is a modern, open source integrator
  - github.com/hannorein/rebound
- It numerically solves the gravitational N-body problem
- Considered the “gold standard” for orbits in this work
- IAS15 adaptive integrator uses Gauss-Radau quadrature and a “predictor-corrector” scheme
- Horizons: API provided by NASA JPL to obtain state vectors (position and velocity) of objects in the Solar System
- Considered “gold standard” for initial conditions of an integration

Keplerian Orbital Elements



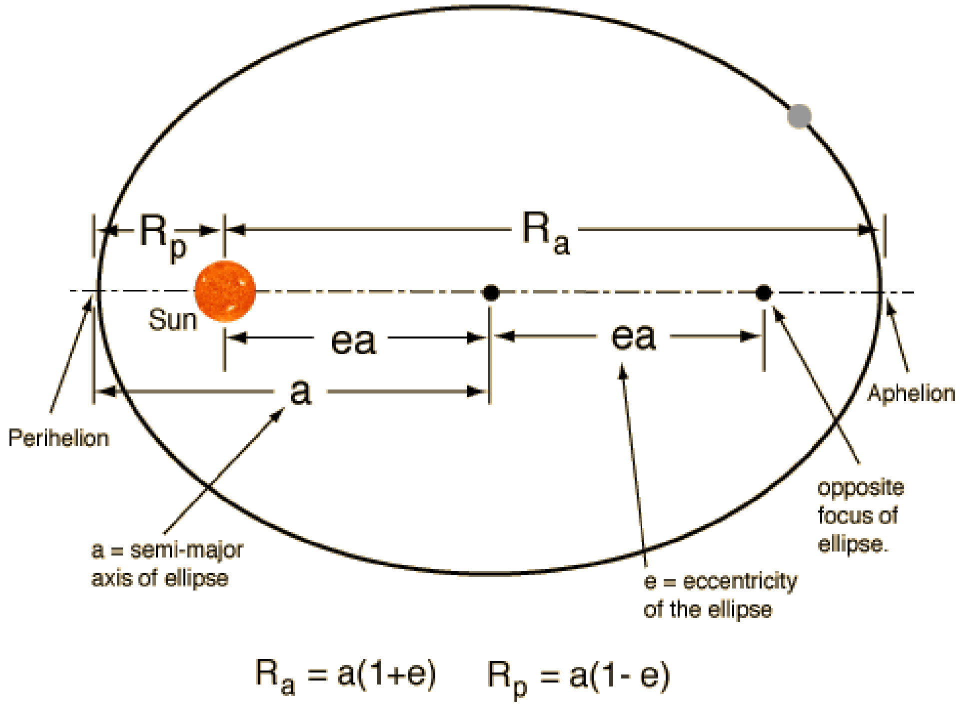
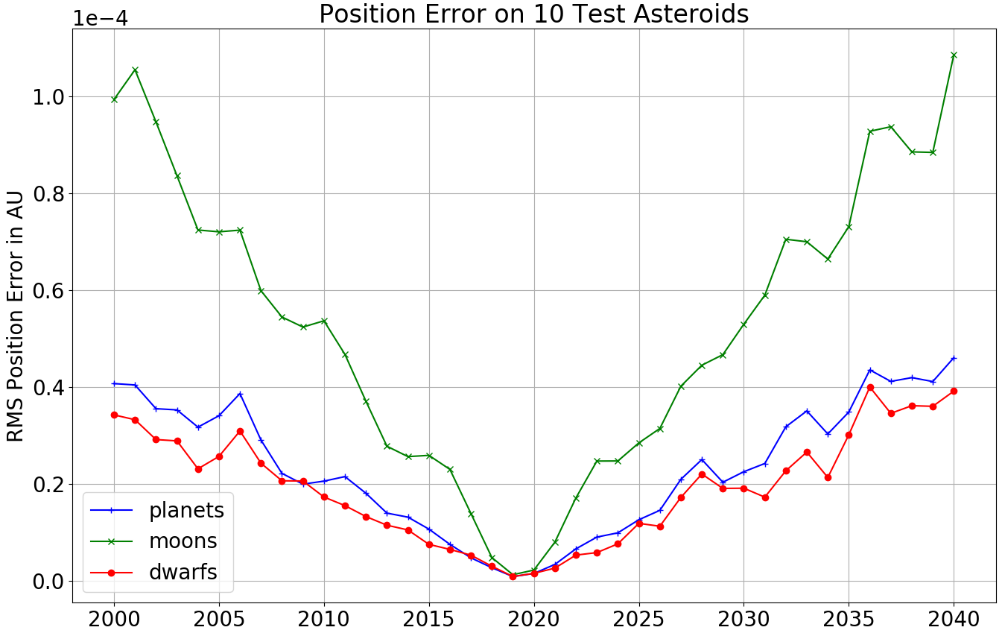


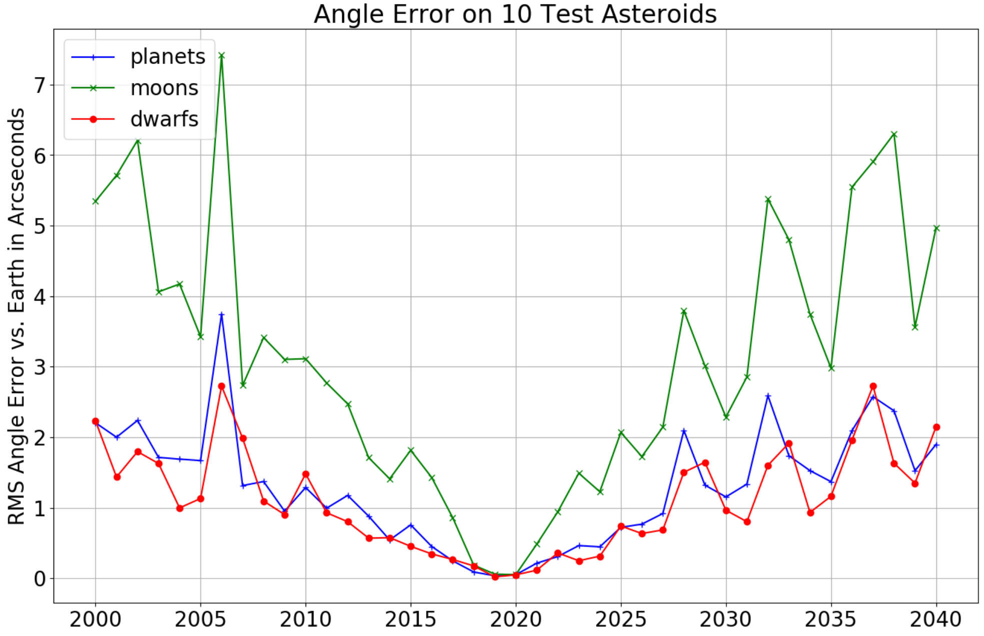
Image Credits: WikiPedia, Cool Cosmos

- Semi-major axis *a* and eccentricity *e* describe the size and shape of the orbital ellipse
- Inclination *i*, ascending node Ω, perihelion ω are angles orienting orbit in the ecliptic plane
- True anomaly *f* is location of the body on its orbital ellipse

Validating Integration vs. Horizons

- Integrate three collections of massive bodies for 40 years at daily interval
- Initial conditions from Horizons at MJD 58600 / 2019-04-27
- Test results on first 10 IAU asteroids; query their positions from Horizons
- Report error in position (AU) and instantaneous angle from asteroid to Earth (arc seconds)
- Accuracy is excellent!
    - RMS error on planets is 5.4E-6 AU
    - Angle error from asteroids to planets 0.8 arc seconds





Bulk Integration of 733,489 Asteroids

- Download asteroid orbital elements from JPL
- Data available for 733,489
- Integrate these daily for 40 years
- Save results to disk
    - REBOUND simulation archives
    - Numpy arrays
- Job takes 4:30 on 40 CPU cores
- Writes 1.37 TB output to disk

```
# Load all the asteroid elements
ast_elt = load_ast_elt()
```

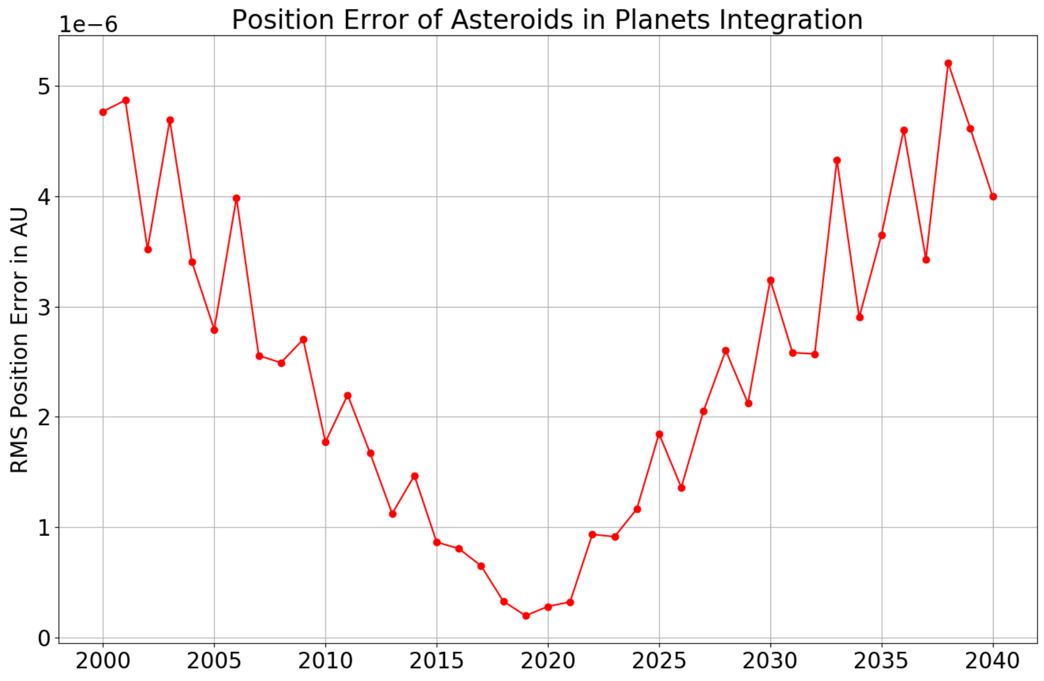
```
ast_elt
```

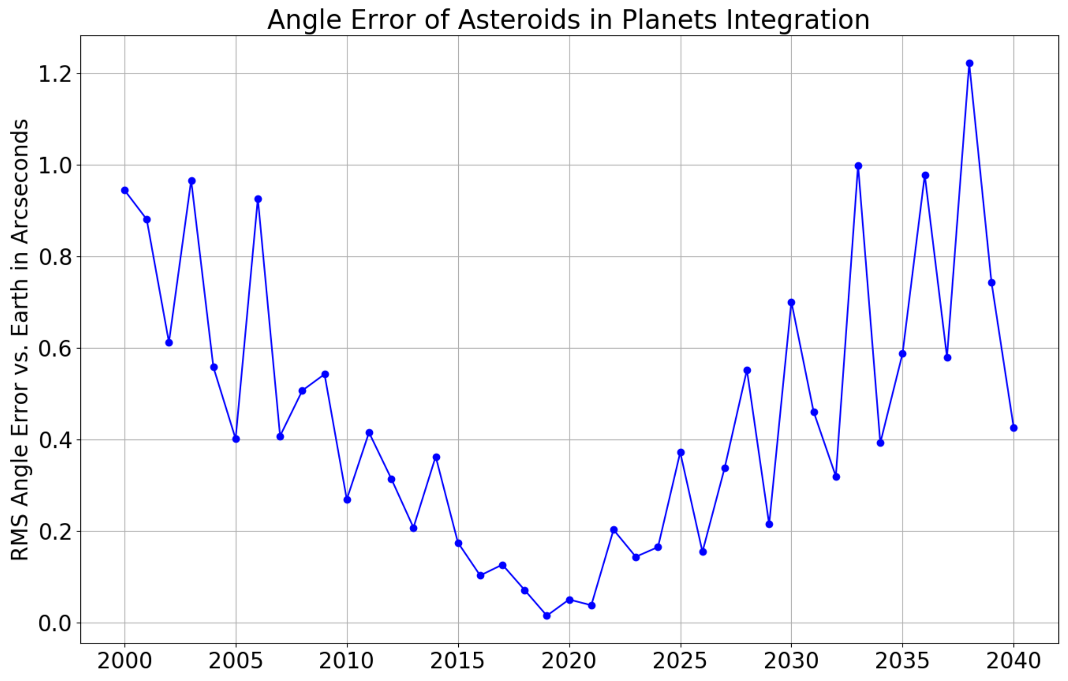
	Num	Name	epoch	a	e	inc	Omega	omega	M	H	G	Ref	f
Num													
1	1	Ceres	58600.0	2.769165	0.076009	0.184901	1.401596	1.284522	1.350398	3.34	0.12	JPL 46	1.501306
2	2	Pallas	58600.0	2.772466	0.230337	0.608007	3.020817	5.411373	1.041946	4.13	0.11	JPL 35	1.490912
3	3	Juno	58600.0	2.669150	0.256942	0.226699	2.964490	4.330836	0.609557	5.33	0.32	JPL 108	0.996719
4	4	Vesta	58600.0	2.361418	0.088721	0.124647	1.811840	2.630709	1.673106	3.20	0.32	JPL 34	-4.436417
5	5	Astraea	58600.0	2.574249	0.191095	0.093672	2.470978	6.260280	4.928221	6.85	0.15	JPL 108	-1.738676
...	...	...	...	...	...	...	...	...	...	...	...	...	...
1255499	1255499	2019 QG	58600.0	0.822197	0.237862	0.220677	5.066979	3.770460	0.503214	21.55	0.15	JPL 1	0.807024
1255501	1255501	2019 QL	58600.0	2.722045	0.530676	0.113833	4.741919	2.351059	5.297173	19.21	0.15	JPL 1	-2.082964
1255502	1255502	2019 QQ	58600.0	1.053137	0.389091	0.172121	5.648270	2.028352	3.266522	25.31	0.15	JPL 1	-3.081905
1255513	1255513	6331 P-L	58600.0	2.334803	0.282830	0.141058	6.200287	0.091869	2.609695	18.50	0.15	JPL 8	2.827595
1255514	1255514	6344 P-L	58600.0	2.812944	0.664688	0.081955	3.199363	4.094863	2.738525	20.40	0.15	JPL 17	3.032066

733489 rows × 19 columns

Validate Asteroid Integration vs. Horizons

- Test bulk asteroid integration on first 25 IAU asteroids
- Report position error in AU and angle error to Earth in arc seconds
- Excellent results! RMS 2.49E-6 AU and 0.45 arc seconds





Integrate Kepler Two Body Problem in TensorFlow

$$r(\theta) = \frac{a \cdot (1 - e^2)}{1 - e \cdot \cos(\theta - \theta_0)}$$

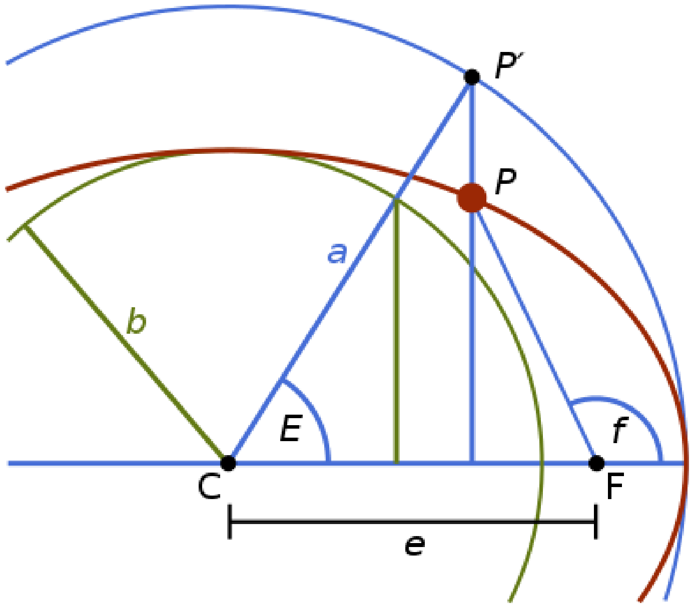
- Analytical solution to Kepler problem is an ellipse
- 5 of the 6 orbital elements *a*, *e*, *i*, Ω, ω constant
- The Mean Anomaly *M* is linear in time (2nd Law)

$$M(t) = M_0 + N \cdot (t - t_0)$$

- Kepler's Equation relates orbital anomalies:

$$M = E - e\sin(E) \qquad \text{Kepler's Equation}$$

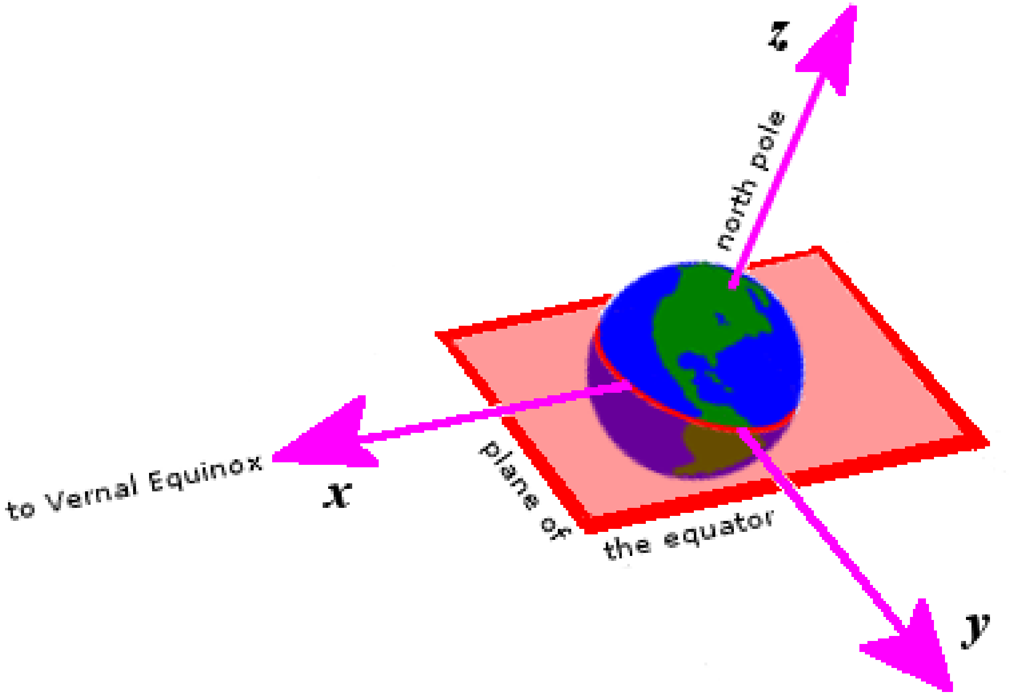
$$\tan\left(\frac{f}{2}\right) = \sqrt{\frac{1+e}{1-e}} \cdot \tan\left(\frac{E}{2}\right) \qquad \text{true to eccentric}$$

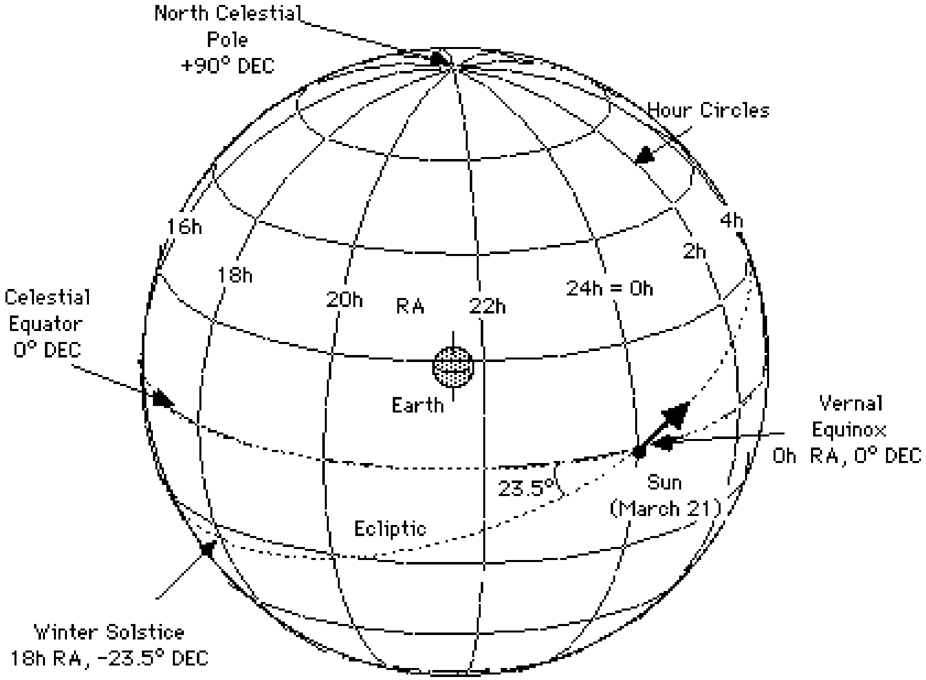


- Convert *M* to *E* to *f*, then to Cartesian coordinates
- TensorFlow is fast! 5000 time points in ~300 μ sec
- Apply calibration d**q**, d**v** to match REBOUND integration at input orbital elements

Predicting Directions from Positions

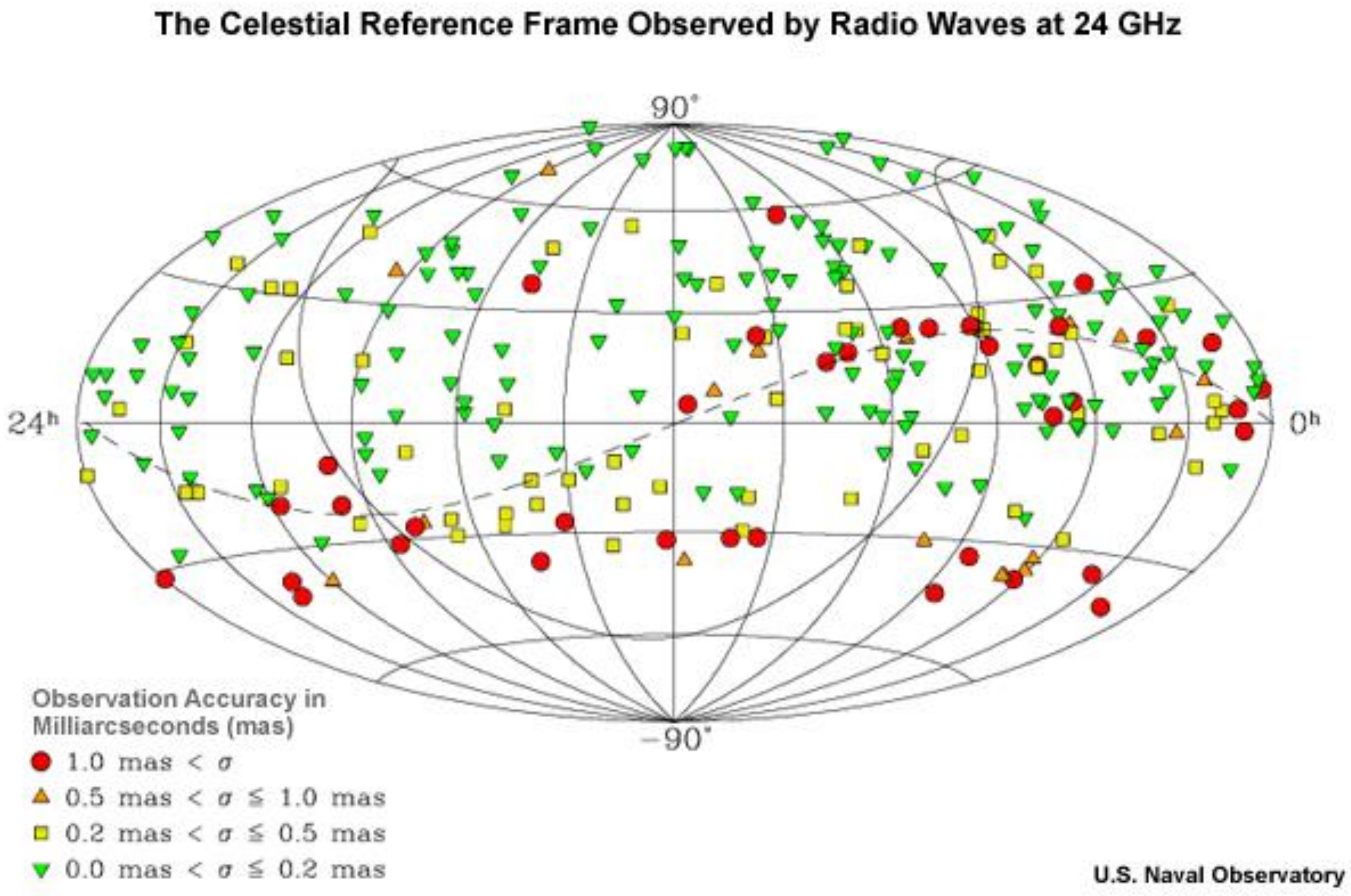
Right Ascension and Declination





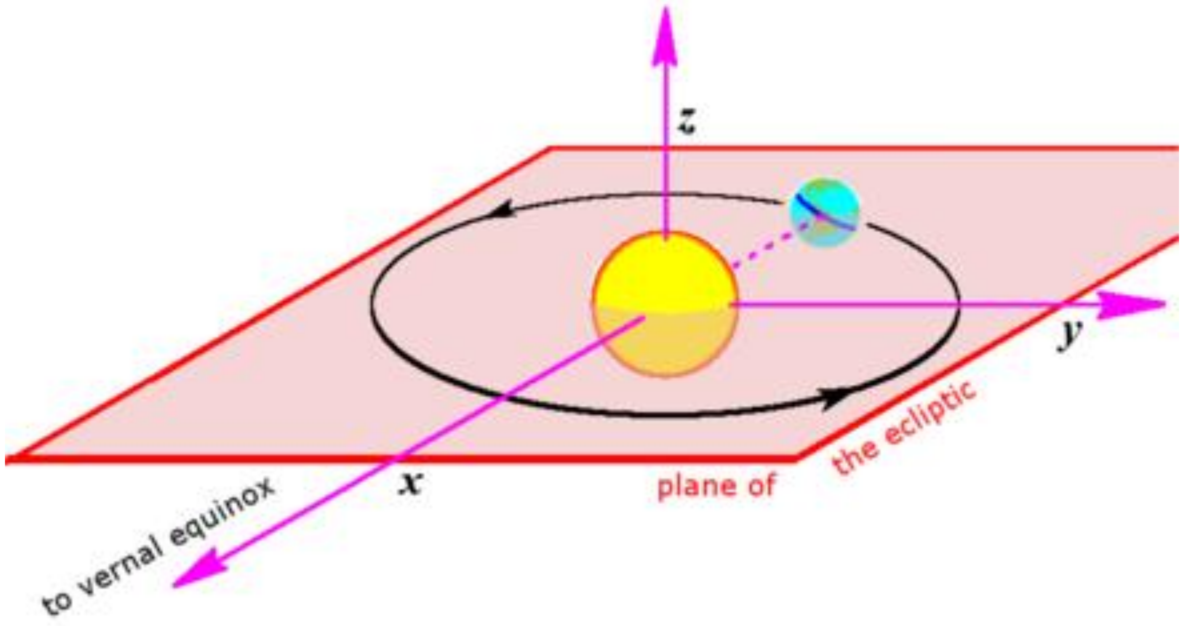
- Fundamental plane is aligned with Earth's equator
- Intuitive, dates to ancient astronomers
- Two problems: precession (drift) and nutation (wobbles) in direction of North Pole

International Celestial Reference Frame (ICRF)



- Modern system for RA/Dec
- Based on 232 extragalactic objects
- Addresses precession and nutation
    - Quasars don't move!
    - Not in **direction** anyway
- Amazingly accurate
    - ~2 milliarc-seconds
- Intuition: like using Polaris instead of Earth's axis for the North Pole
- Except you use 232 stars to get a highly accurate composite direction

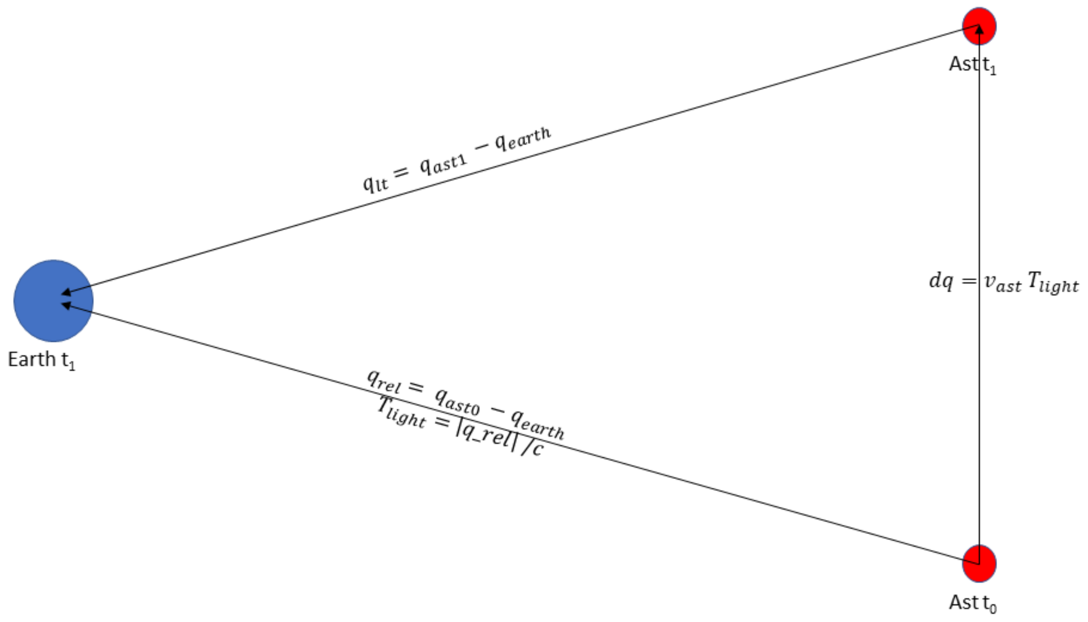
From RA/Dec to Barycentric Mean Ecliptic



- RA/Dec is ideal for observing the stars
- But not for calculations involving orbits in the Solar System
- Inside the Solar System we want an inertial frame aligned with the ecliptic: BME
- Convert between ICRF and BME using `astropy` library

```
obs_icrs = astropy.SkyCoord(ra=ra, dec=dec, obstime=obstime, frame=ICRS)
obs_ecl = obs_icrs.transform_to(BarycentricMeanEcliptic)
```

Calculate Direction from Position and Velocity



- Need to remember light speed *c* is finite!
- Otherwise wrong by ~285 arc seconds

$$\mathbf{q}_{\text{rel}} = \mathbf{q}_{\text{ast}} - \mathbf{q}_{\text{earth}}$$

$$T_{\text{light}} = \left\|\mathbf{q}_{\text{rel}}\right\|/c$$

$$\Delta\mathbf{q}_{\text{ast}} = \mathbf{v}_{\text{ast}} \cdot T_{\text{light}}$$

$$\mathbf{q}_{\text{lt}} = \mathbf{q}_{\text{rel}} - \Delta\mathbf{q}_{\text{ast}}$$

$$\mathbf{u} = \mathbf{q}_{\text{lt}}/\left\|\mathbf{q}_{\text{lt}}\right\|$$

- Earth velocity doesn't matter, only asteroid velocity
- BME is an inertial frame
- Stellar aberration inapplicable, would be double counting

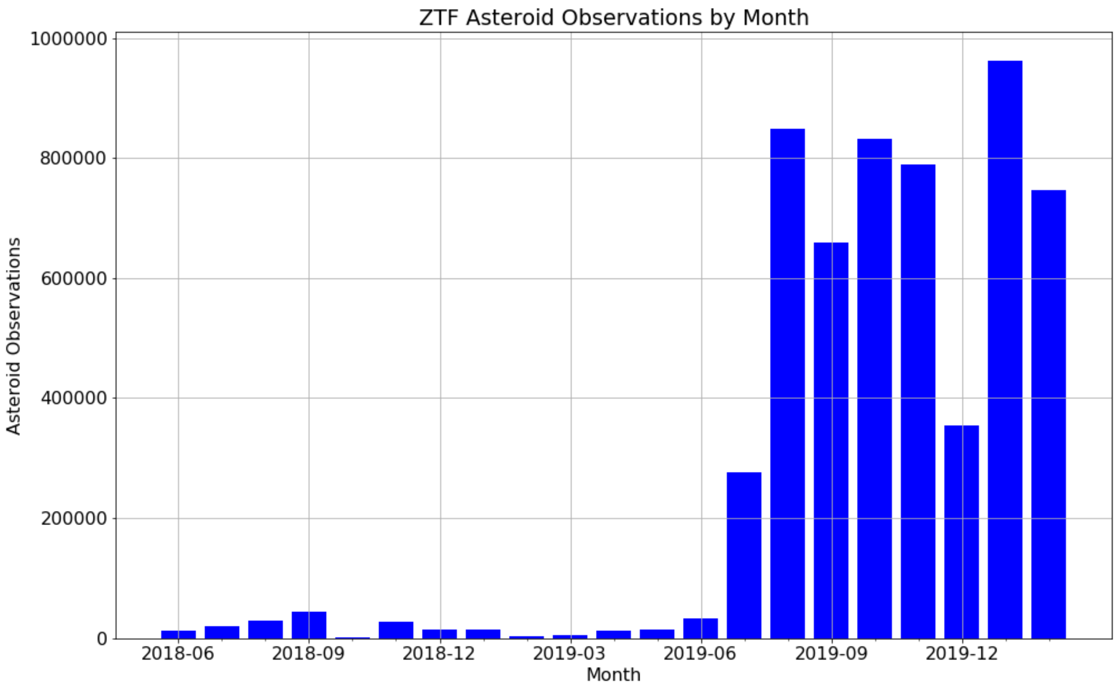
- Also need "topos adjustment" for observatory: Palomar Mountain, not geocenter!
- Topos adjustment worth 0-5 arc seconds on first 16 asteroids

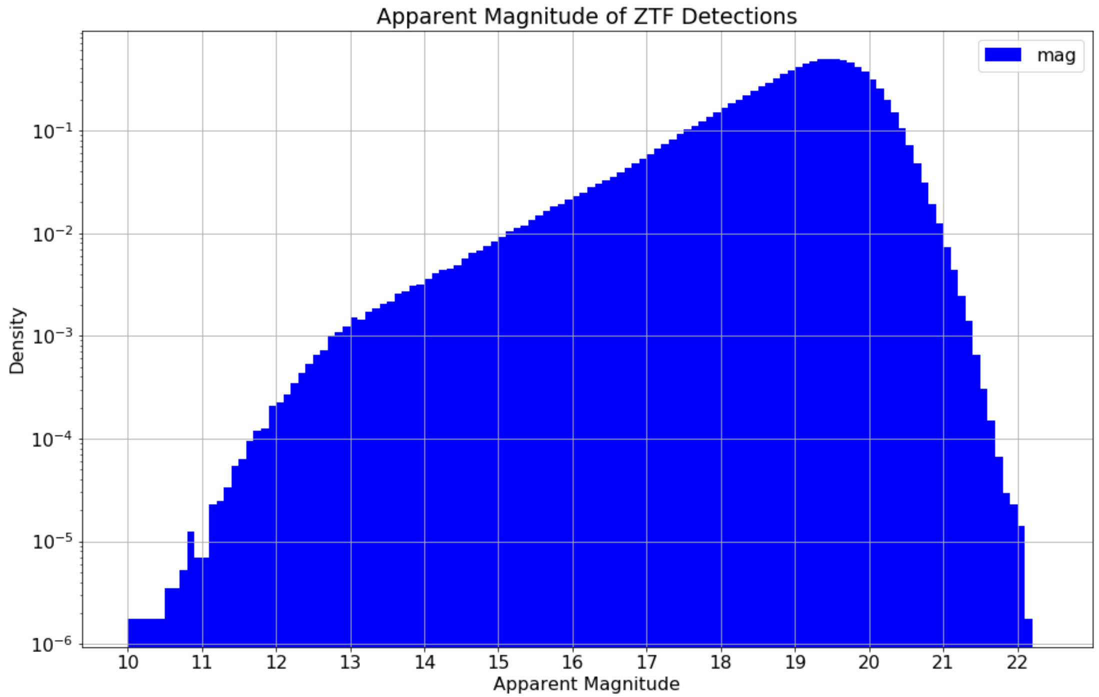
Validating Astrometric Direction

- Check these results by comparing vs. JPL, SkyField
- Downloaded Mars at 3 hour intervals over 10 years (~29000 rows)
  - Both state vectors ***q***, ***v*** and observer RA / Dec
- Computed astrometric directions from Earth to Mars
  - MSE and SkyField identical: 0.027 arc seconds
  - Both MSE and SkyField differ from JPL by 1.6 arc seconds
- Separately downloaded JPL RA/Dec on first 16 asteroids
- Compared to MSE direction calculated from integrated orbits
- RMS error: 0.873 arc seconds!

Analysis of ZTF Asteroid Detections

EDA of ZTF Detections



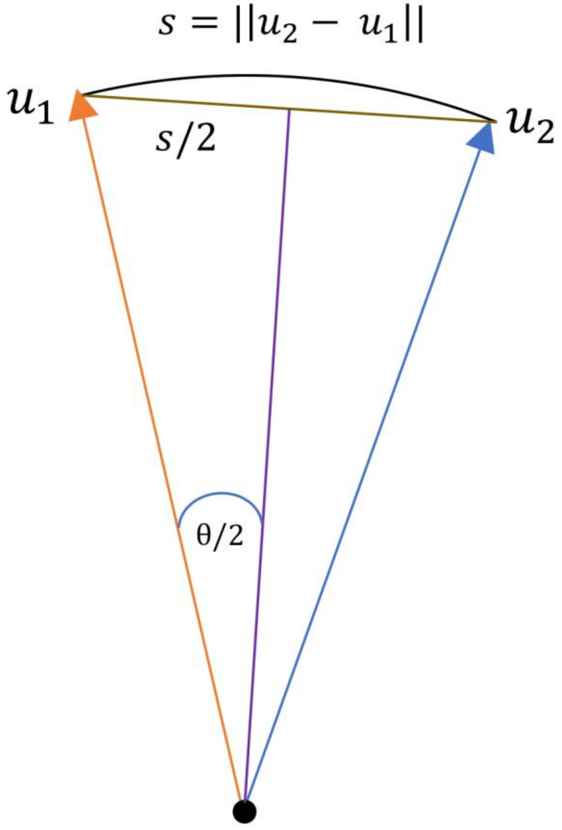


- ZTF: Zwicky Transient Facility; survey of northern sky at Palomar Mountain by Cal Tech
- Fast, deep survey: 3750 square degrees / hour to depth of 20.5 mag
- First light in 2017, but asteroid detections ramp up in July 2019; 7 months of data
- Enriched with machine learning pipeline that filters probable asteroid detections
- 5.69 million possible asteroid detections used for masters thesis in May 2020
- Data includes: MJD, RA, DEC, MAG
- Update: As of March 2021 now have ~158 million asteroid detections in a database!

Converting Cartesian to Angular Distance

- How far apart are two directions in the sky?
- Convert RA/Dec to directions u_1 and u_2 in the BME
- Compute Cartesian distance s between u_1 and u_2
- Angular distance θ is geodesic (great circle distance)

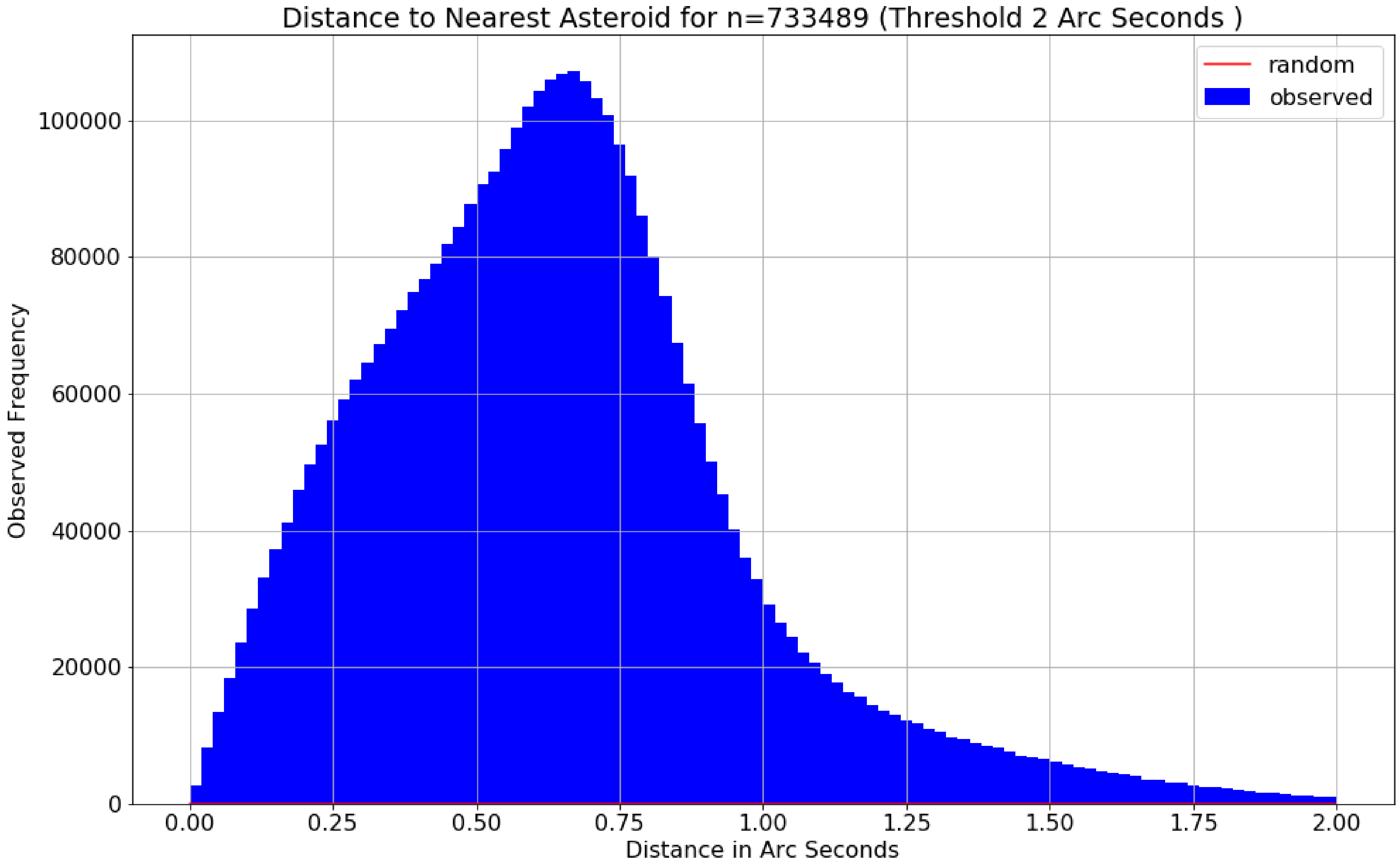
$$\sin(\theta/2) = s/2$$



Nearest Asteroid to Each ZTF Detection

- Compute direction $u_{obs} = (u_x, u_y, u_z)$ from RA/Dec for each detection
- Compute direction u_{ast} for every asteroid in the catalogue
- 5.7E6 detections x 7.3E5 asteroids = 4.2E12 (4.2 trillion) interactions!
  - Too big for naïve brute force attack
- “Only” 97,111 different MJDs with ZTF detections
- Work in chunks of 1000 asteroids at a time, find nearest to each ZTF
- Then perform reduction operation to find globally nearest asteroid
- Still large compute job: 25 hours on 40 CPUs, 256 GB RAM server

Nearest Asteroid: 65.7% Within 2.0 Arc Seconds!



Statistical Distribution of Distance on Sphere

- What is the statistical distribution of s if we guessed directions uniformly at random?

$$s^2 = 2 \cdot (1 - z) \qquad z = 1 - s^2/2$$

- This is useful parameterization because…
- "Orange Slicing Theorem" for solid angle measure:

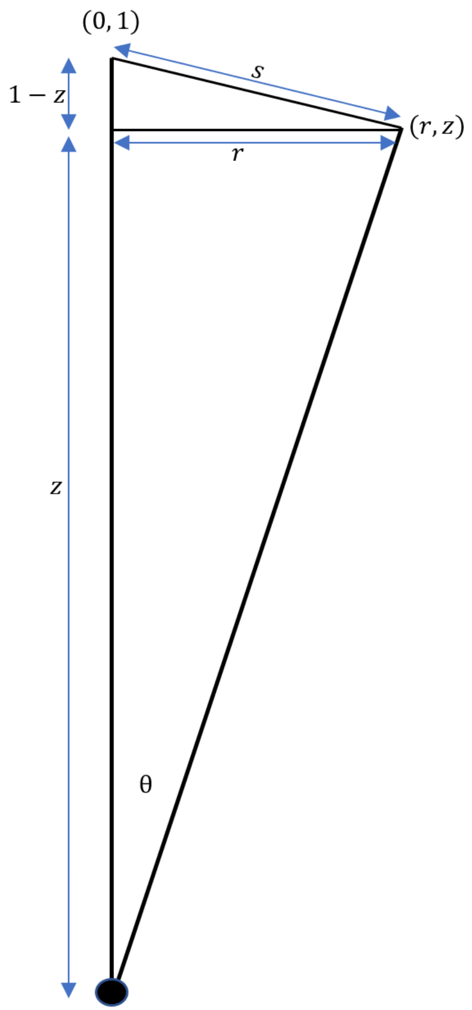
$$d\Omega = dz \cdot d\phi$$

- Think of Z and S as random variables:

$$Z \sim \mathrm{Unif}(-1, 1) \quad S^2 \sim \mathrm{Unif}(0, 4)$$

- Conditional on a max (threshold) distance τ

$$S^2 | S \leq \tau \sim \mathrm{Unif}(0, \tau^2)$$



Distribution of Nearest Asteroid Distance

- Set a threshold distance τ and define relative squared distance V

$$V = (S/\tau)^2 \qquad V \sim \mathrm{Unif}(0,1)$$

- We have n = 733,489 guesses and are picking closest

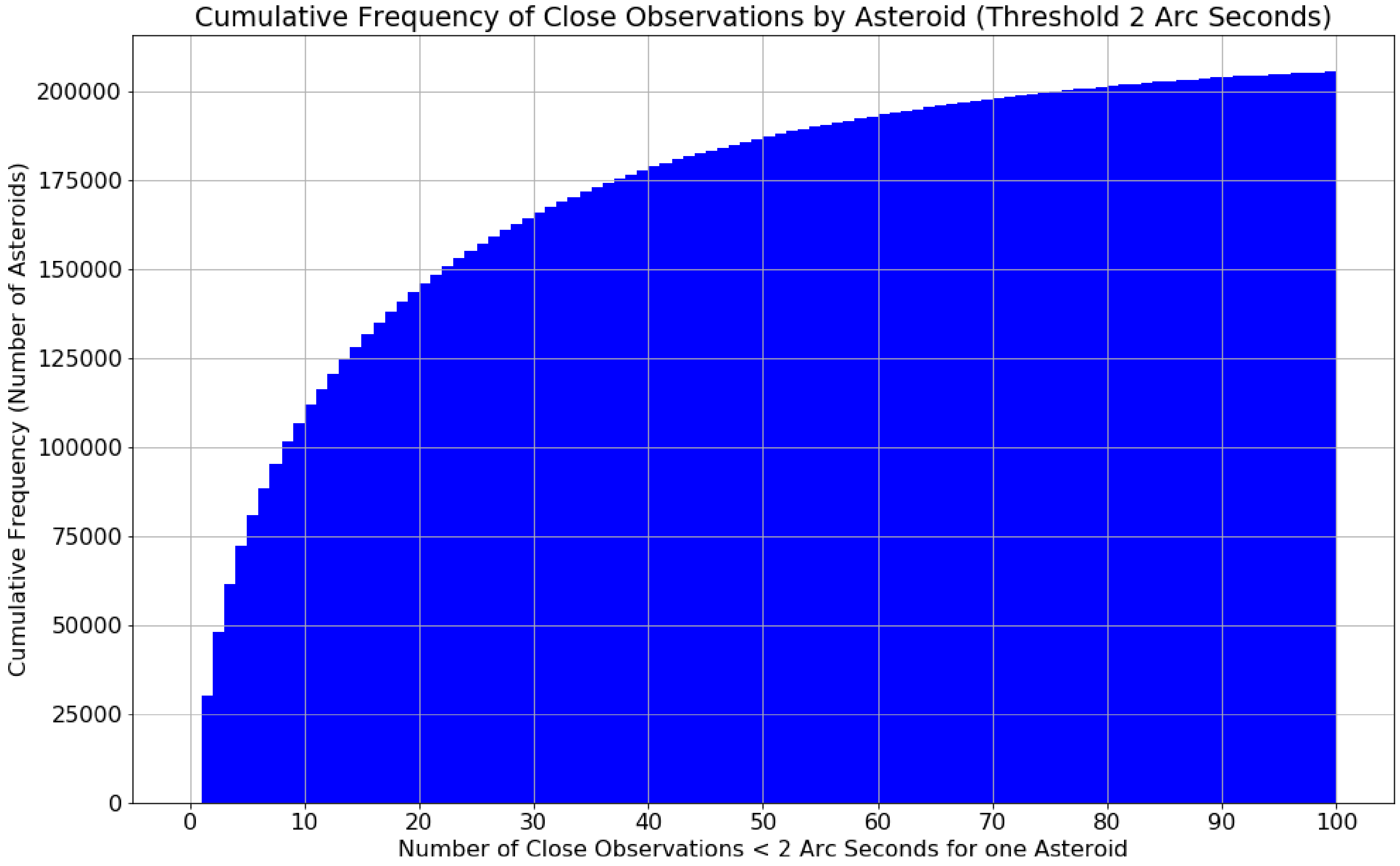
$$V_1, \ldots V_n \overset{i.i.d.}{\sim} \mathrm{Unif}(0,1)$$

- Stat 110: The minimum of n i.i.d. uniforms has a Beta distribution

$$U_{(1)} \sim \mathrm{Beta}(1,n)$$

- How many hits at 2.0 arc seconds would we get by luck?
- Only 98. But we got 3.75 million of them!
- Conclusion: This whole apparatus works to a tolerance of 2.0 arc seconds

Cumulative Distribution of Hits per Asteroid



- Suppose we want to rebuild the asteroid catalog…
- How many asteroids do we have a sporting chance of finding?
- Count hits at 2.0 arc seconds
- How many asteroids have at least
  - 20 hits? 63,746
  - 10 hits? 100,508
- We have a shot at 13.6% of the catalogue if we require 10+ hits

Asteroid Search Using Orbital Elements

Assemble ZTF Detections Near Elements

```
# Load unperturbed element batch
ztf_elt_ast = load_ztf_batch(elts=elts_ast, thresh_deg=1.0, near_ast=False)
```

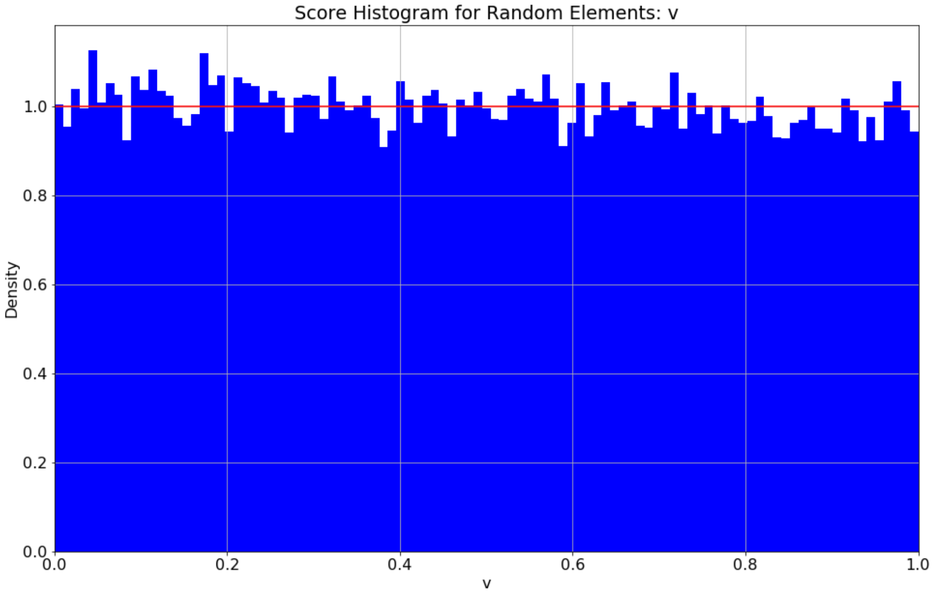
```
# Review
ztf_elt_ast[cols]
```

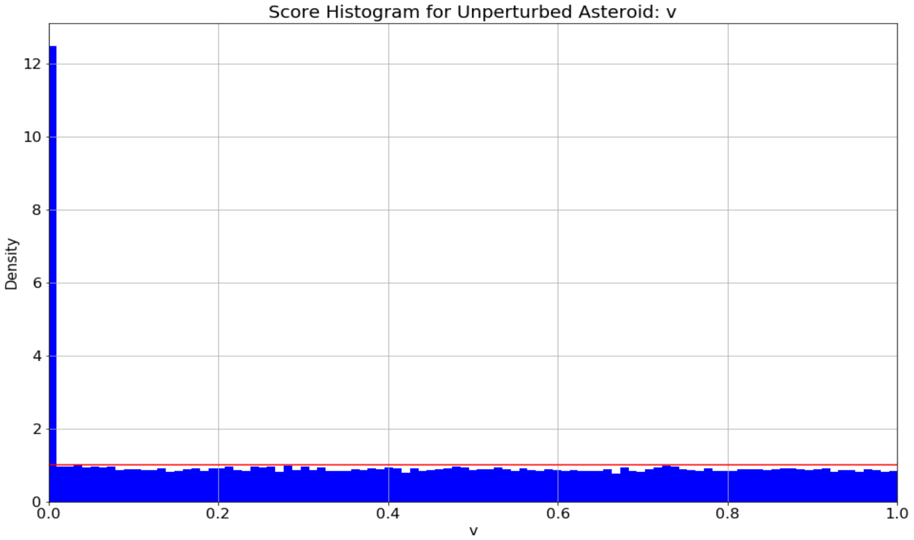
	element_id	ztf_id	mjd	ra	dec	ux	uy	uz	mag_app	elt_ux	elt_uy	elt_uz	s_sec	v	is_hit
0	733	53851	58348.197581	266.229165	-13.513802	-0.063945	-0.983101	0.171530	16.755600	-0.057300	-0.982042	0.179751	2191.408734	0.370552	False
1	733	73604	58348.197581	265.761024	-13.509148	-0.071871	-0.982578	0.171389	16.035999	-0.057300	-0.982042	0.179751	3467.151428	0.927559	False
2	733	82343	58389.193252	270.331454	-11.244934	0.005674	-0.977422	0.211222	17.196199	0.000919	-0.977996	0.208622	1124.103915	0.097503	False
3	733	257221	58685.471227	29.693832	42.180412	0.643725	0.603886	0.470042	19.289200	0.639004	0.610779	0.467571	1797.091521	0.249197	False
4	733	327000	58691.465972	33.104905	44.059131	0.601970	0.636719	0.481893	17.725201	0.606278	0.637608	0.475272	1639.539679	0.207419	False
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
90206	324582	5650588	58904.176701	44.164238	29.650540	0.623416	0.752309	0.213037	18.084700	0.627640	0.750696	0.206212	1688.638104	0.220027	False
90207	324582	5650589	58904.176250	44.164062	29.650536	0.623417	0.752307	0.213038	18.165199	0.627641	0.750695	0.206213	1688.601889	0.220018	False
90208	324582	5650665	58904.176250	44.368640	28.490480	0.628284	0.753618	0.193182	19.025200	0.627641	0.750695	0.206213	2757.856412	0.586871	False
90209	324582	5650697	58904.176250	43.296207	29.505908	0.633424	0.743491	0.214467	19.852800	0.627641	0.750695	0.206213	2555.278205	0.503822	False
90210	324582	5650705	58904.176250	44.621045	29.303550	0.620689	0.756675	0.205398	19.647400	0.627641	0.750695	0.206213	1898.912116	0.278236	False

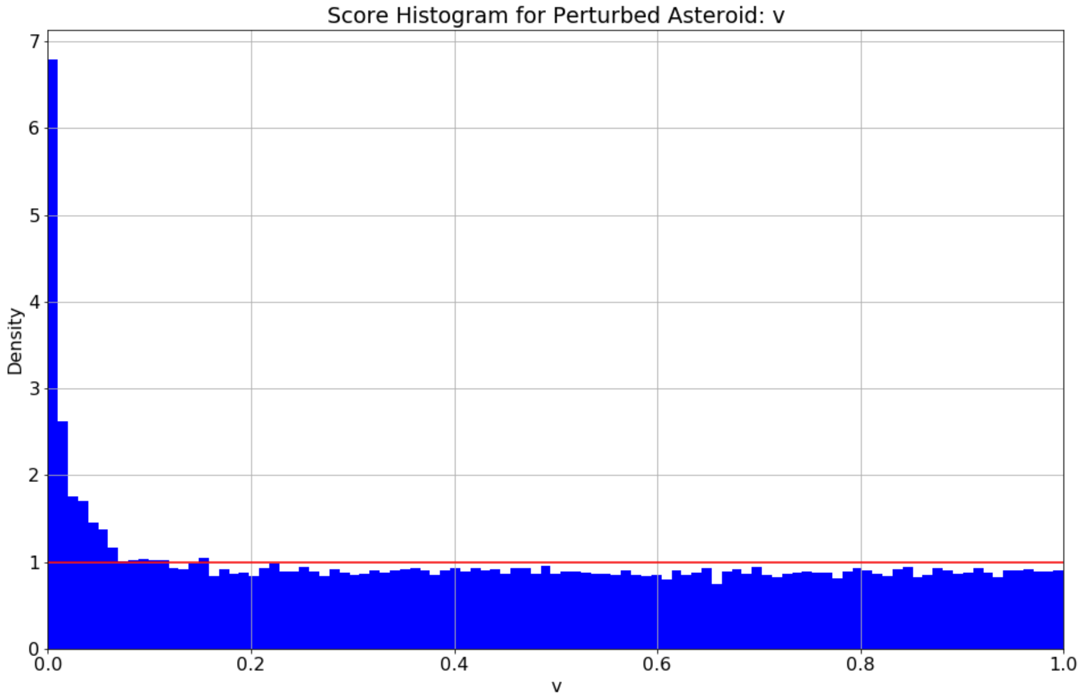
90211 rows × 15 columns

- Integrate the candidate elements on the fly in REBOUND and compute directions
- Filter the ZTF detections to those within threshold of the elements

Distribution of $V = (S/\tau)^2$ for 3 Element Batches







- Plot V for 3 batches: random, unperturbed, perturbed
- Results match theory perfectly!
- Random elements close to uniform distribution
- Unperturbed: uniform on misses with spike in first bucket
- Perturbed: in between; hits leak out to ~250 arc seconds

Log Likelihood Objective Function

- Mixture probability model: V mixture of h hits, (1-h) misses

$$V|\mathrm{Hit} \sim \mathrm{Expo}(\lambda) \qquad V|\mathrm{Miss} \sim \mathrm{Unif}(0,1)$$

- Relate decay rate to "resolution" parameter R

$$f(v) \propto e^{-\lambda v} = e^{-\lambda s^2/\tau^2} \qquad f(v) \propto e^{-s^2/2R^2} \qquad \lambda = \frac{\tau^2}{2R^2}$$

- The resolution R controls how tightly the model focuses
- Mixture PDF:

$$h \cdot \frac{\lambda \cdot e^{-\lambda v}}{1 - e^{-\lambda}} + (1-h)$$

- Log Likelihood:

$$\mathcal{L}(\mathbf{v}, h, \lambda) = \sum_{j=1}^{n} \log\left(h_j \cdot \frac{\lambda \cdot e^{-\lambda_j v_j}}{1 - e^{-\lambda_j}} + 1 - h_j\right)$$

Search Overview

- Six trainable orbital elements a, e, i, Ω, ω, f; epoch not trainable
- Three trainable mixture parameters: N_h, R, τ
- Compute position **q** and velocity **v** from candidate elements
- Compute direction $\mathbf{u}_{pred}$ from **q**, **v;** include light time and topos
- Compute distance s from $\mathbf{u}_{pred}$ to $\mathbf{u}_{obs}$ for ZTF observations
- Compute log likelihood $\mathscr{L}_i$ for each candidate element
- Gradient descent…
- The rest is details!

Search Techniques I: Uniform Scale, Gradient Clipping and Independent Weights

- Control variables on uniform scale in [0, 1]
  - e.g. a = a_min x exp(a_ x log(a_max / a_min)); a_ trainable in [0,1]
- Clip gradients by norm; max || Grad $\mathscr{L}$ || = 1
  - would be better to do this elementwise, but requires custom class
- Track log likelihood and hits for each candidate element before summing them in the objective function
- Revert changes only on elements that got worse during an episode
- Weight log likelihood for each element in batch independently

$$\mathcal{L} = \sum_{i=1}^{b} w_i \cdot \mathcal{L}_i$$

  - equivalent to controlling 64 learning rates independently
  - reduce learning rate on an element when it overshoots

Search Techniques II: Mixture vs. Joint Mode, Encouraging Convergence

- Joint mode: learn all parameters jointly
- Mixture mode: only learn N_h, R, τ
- Learning rate: 2^{-12} in mixture mode vs. 2^{-16} in joint mode
- Modified objective function in mixture mode

$$\mathcal{L} = \sum_{i=1}^{b} w_i \cdot \frac{\mathcal{L}_i}{R_i^{\alpha} \cdot \tau_i^{\beta}}$$

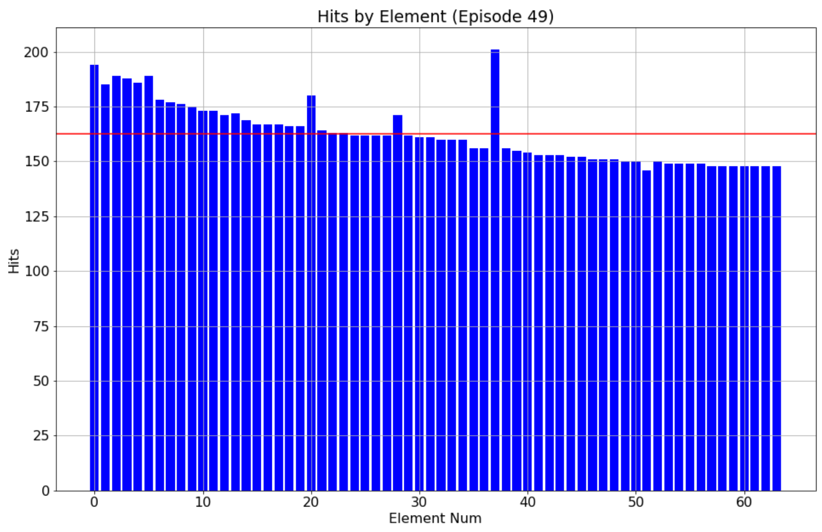
- Theoretical motivation: likelihood would always look better with a larger τ; this encourages the model to converge
- Like adjusting score for degree of difficulty in diving and gymnastics

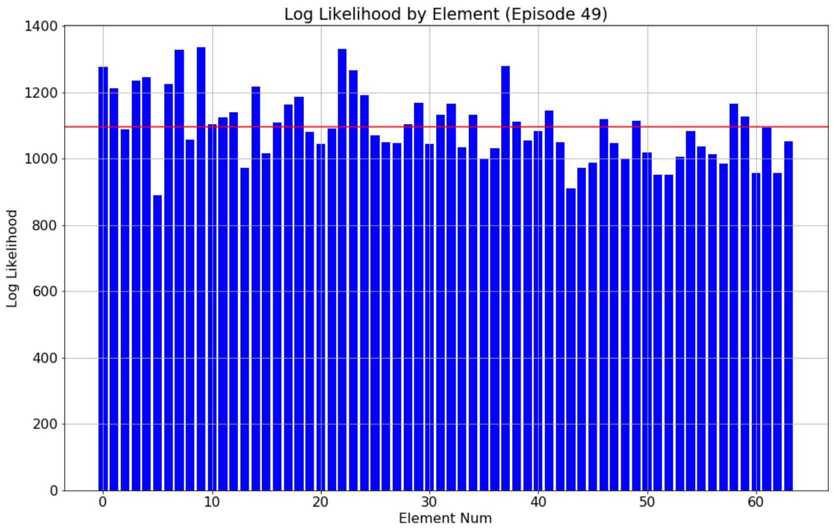
Asteroid Search Results

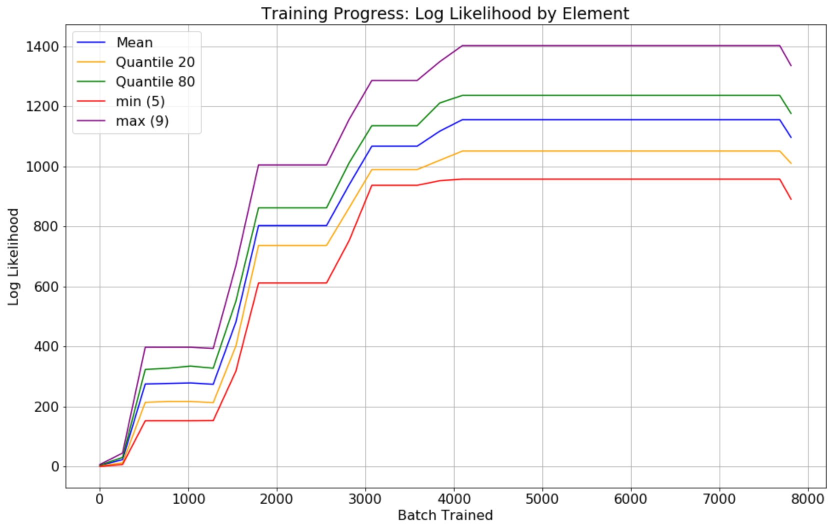
Comparing Two Orbital Elements

- How far apart are two 6D orbital elements ε_1 and ε_2?
- A naïve Euclidean norm makes no sense at all
- Idea 1: inject the elements into space at a set of times
  - The distance between two elements is the mean distance in AU between the orbits they describe
  - Set 240 sample time points at monthly intervals from 2010 to 2030
- Idea 2: transform elements into low dimensional Cartesian space
  - Try to make each component approximately normal
  - Try to make joint distribution approximately multivariate normal
  - Use the Mahalanobis distance on these transformed elements

Train Known Asteroid Elements: Unperturbed

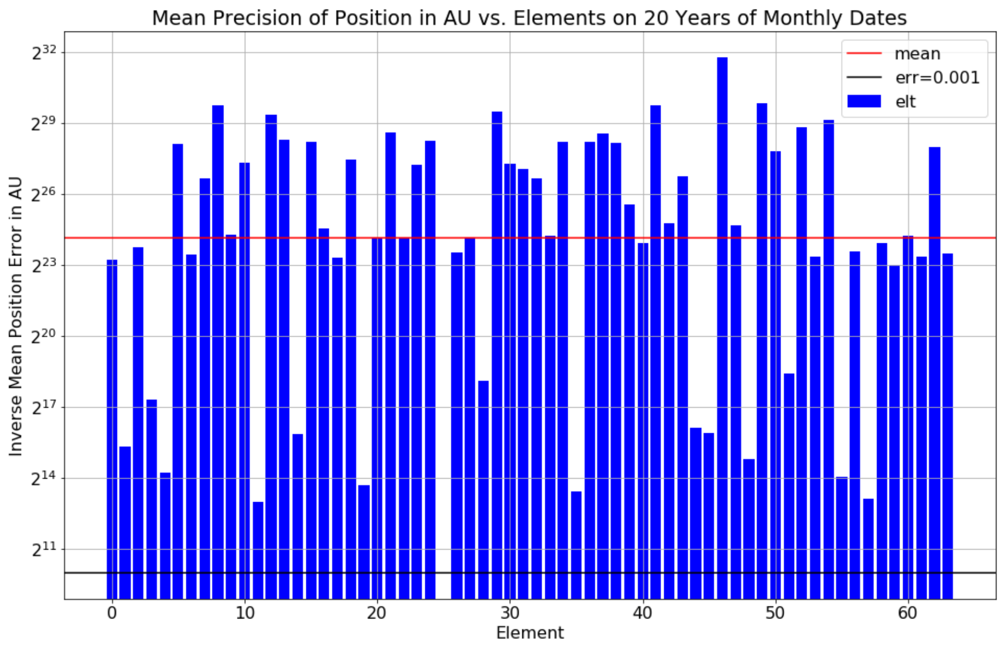


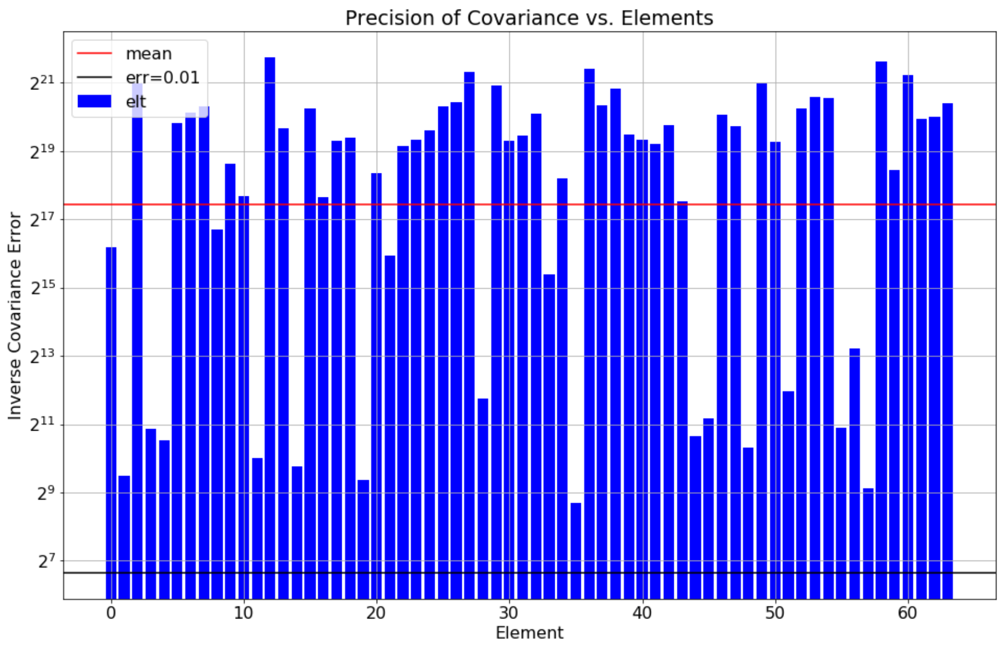




- Start with correct elements but uninformed mixture parameters
- Convergence is almost perfect
- Recovered Elements: 64 (100%)
- Hits: 162.6
- Resolution: 3.0 arc seconds
- Log Like: 1097

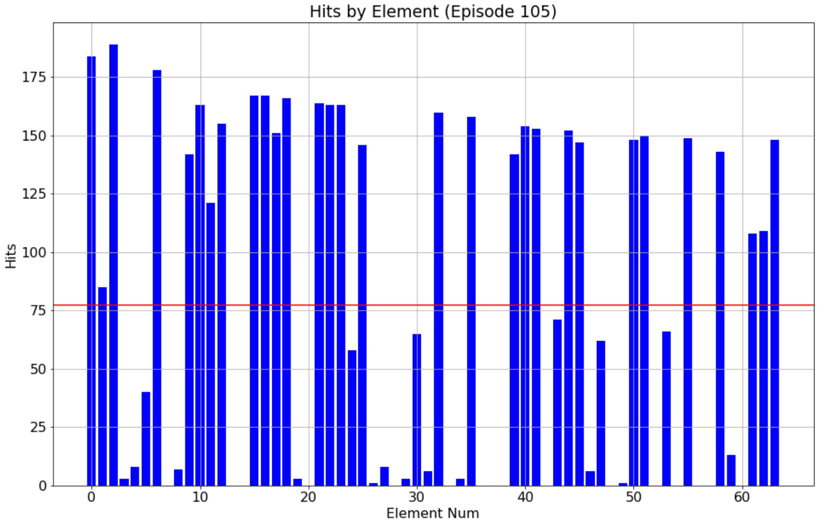
Fit Quality: Unperturbed Elements

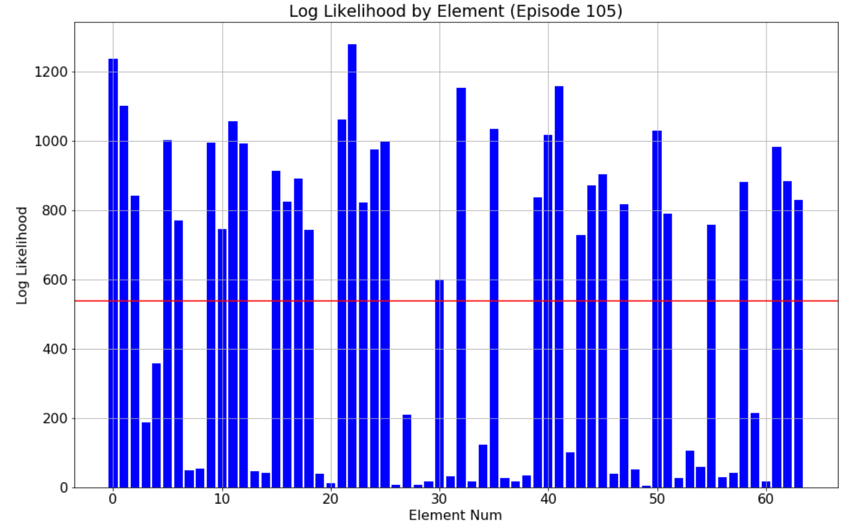


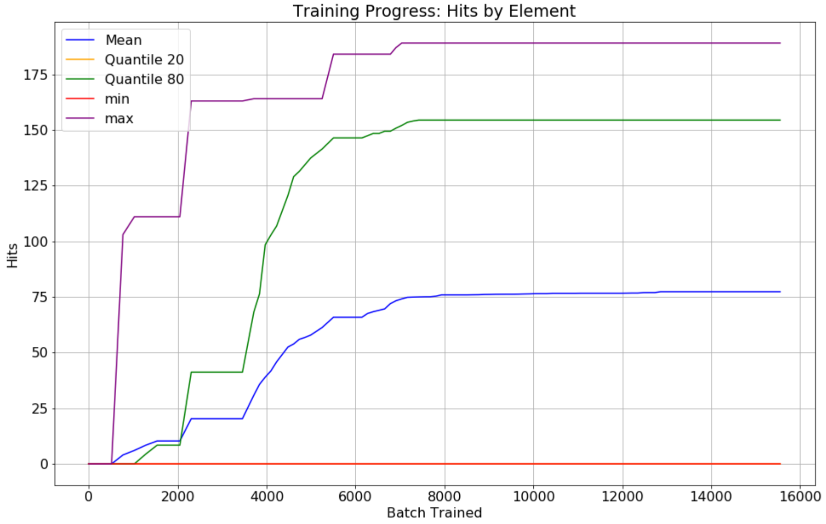


- Do recovered elements match the nearest asteroid?
- 4.6E-8 AU mean distance
- 5.7E-6 covariance norm
- The fit is almost perfect
- Big deal, this is about as hard as hitting a baseball off a tee…

Train Known Asteroid Elements: Small Perturbation

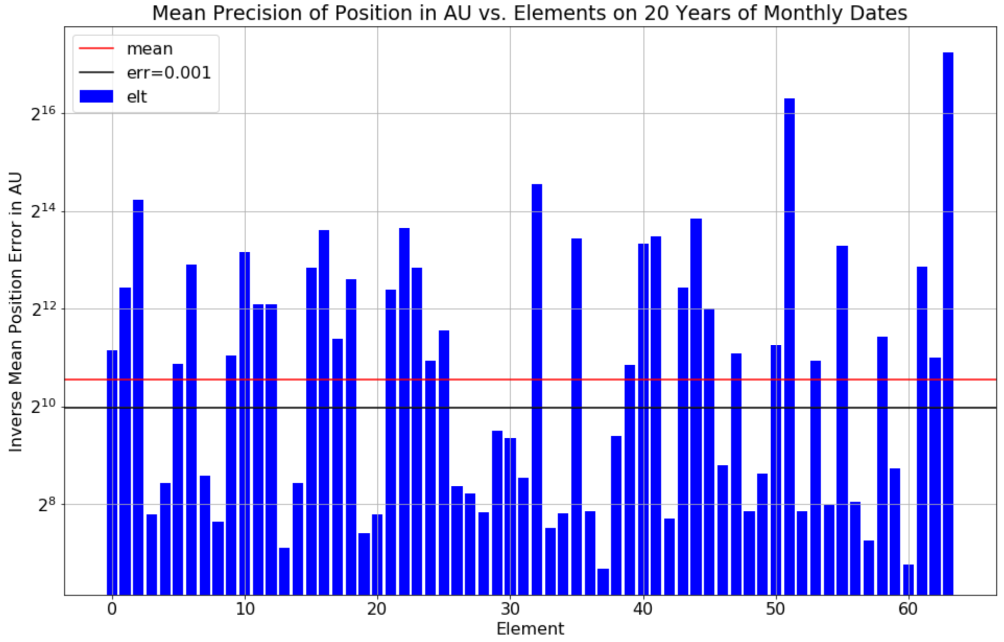


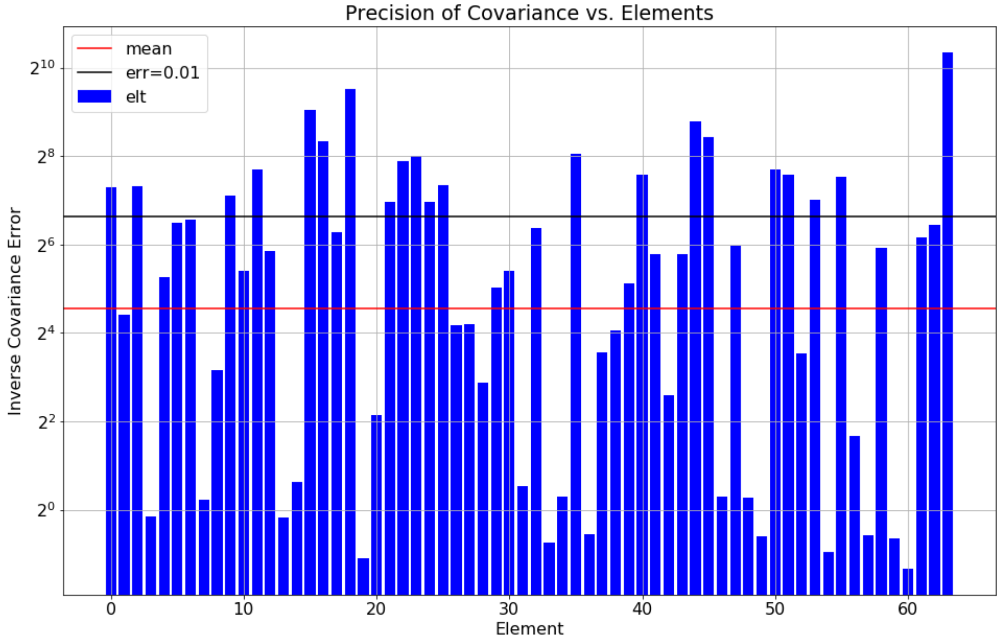




- Small perturbation:
  - 1.0% to *a*
  - 0.25% to *e*
  - 0.05 degrees to *i*
  - 0.25 degrees to Ω, ω, f
- Convergence is very good
- Recovered Elements: 42 (65.6%)
- Hits: 117.5
- Resolution: 18.2 arc seconds
- Log Like: 798
- **Challenge: get full convergence**

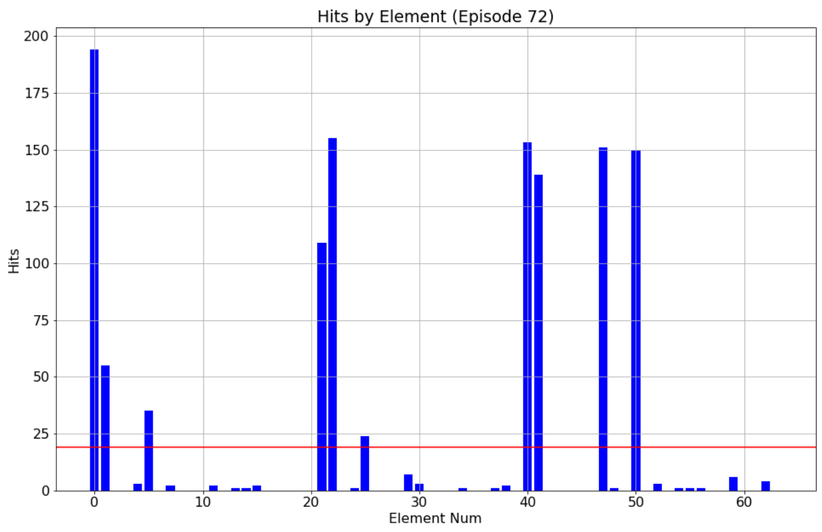
Fit Quality: Small Perturbation

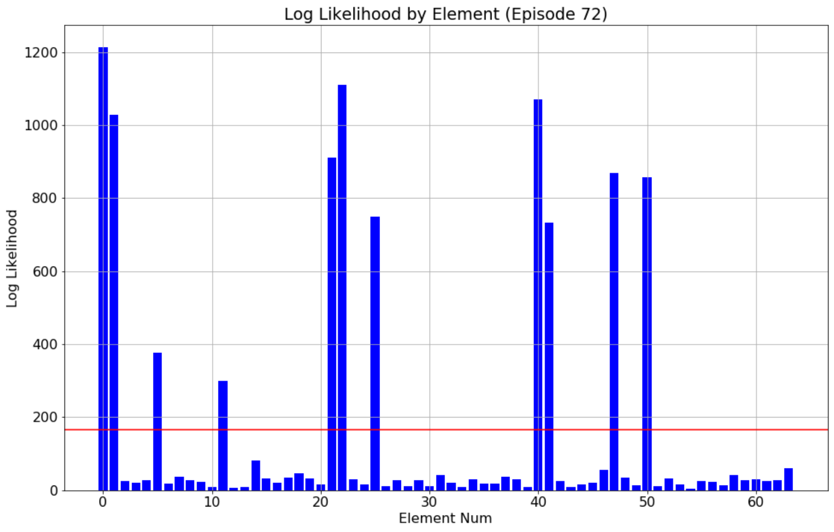


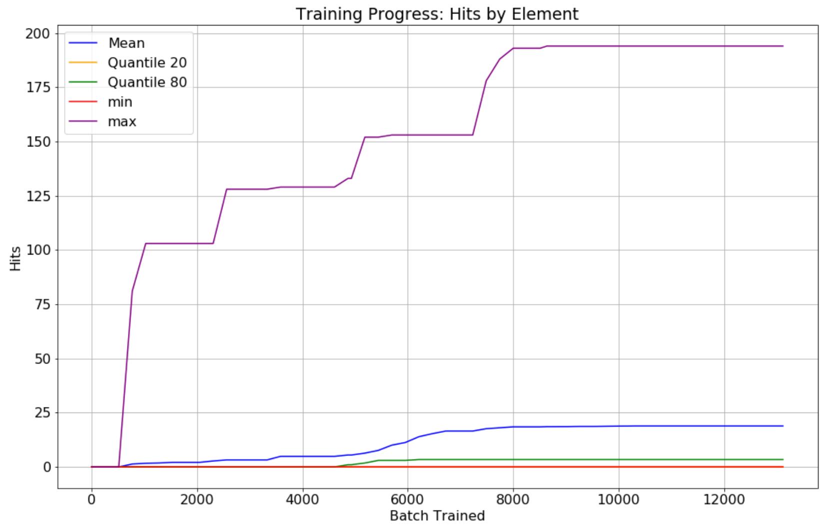


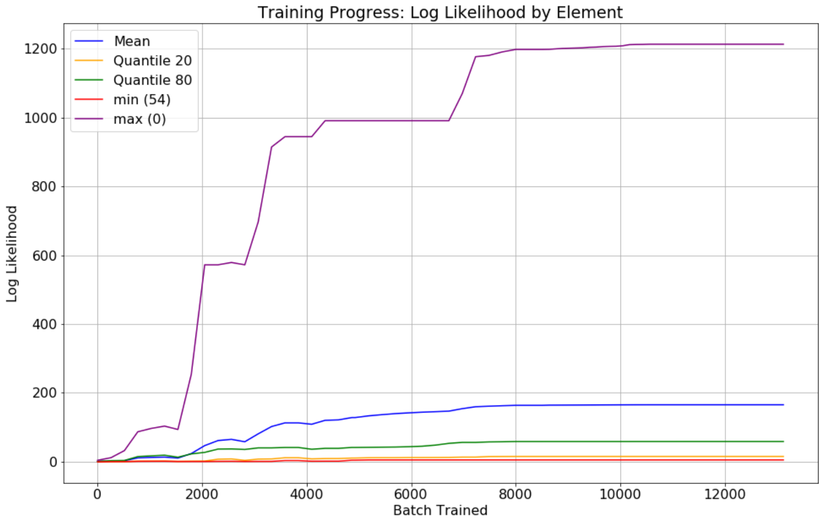
- Do recovered elements match the nearest asteroid?
- 2.6E-4 AU mean distance
- 0.012 covariance norm
- This is still a very good fit on the 42 elements that have been recovered
- This is like your little league coach lobbing the ball over the plate in batting practice…

Train Known Asteroid Elements: Large Perturbation



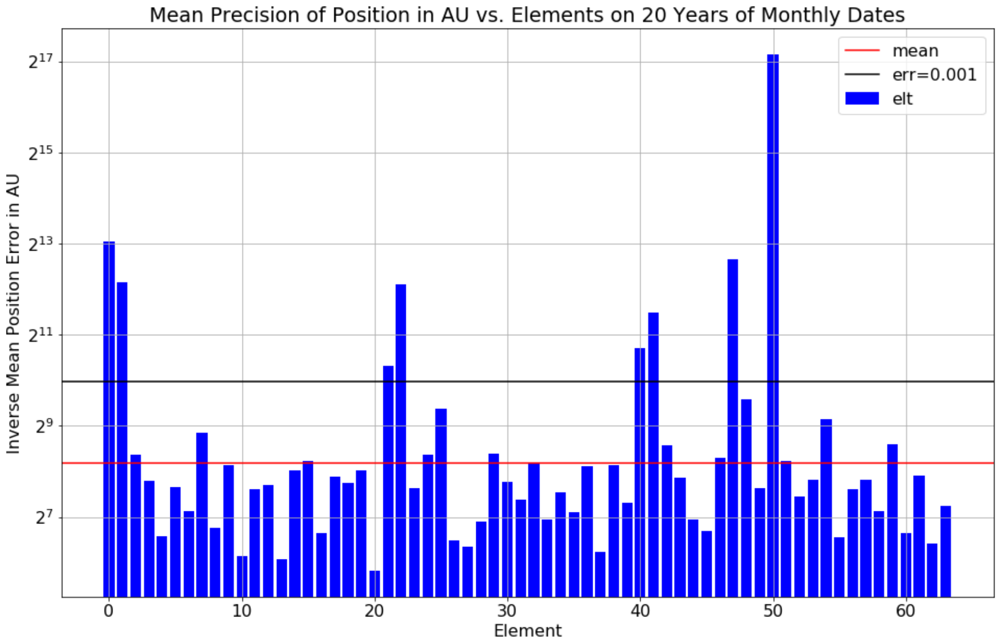


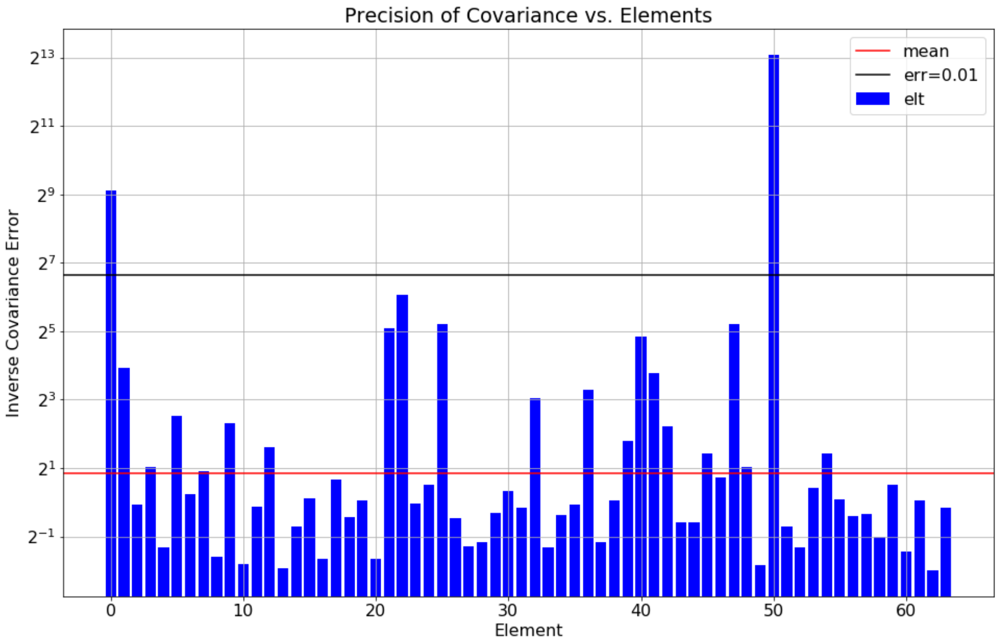




- Small perturbation:
  - 5.0% to *a*
  - 1.0% to *e*
  - 0.25 degrees to *i*
  - 1.0 degrees to Ω, ω, f
- Convergence is decent
- Recovered Elements: 12 (18.8%)
- Hits: 98.2
- Resolution: 32.4 arc seconds
- Log Like: 748
- Many of these elements were perturbed so far the original is no longer even the nearest asteroid!

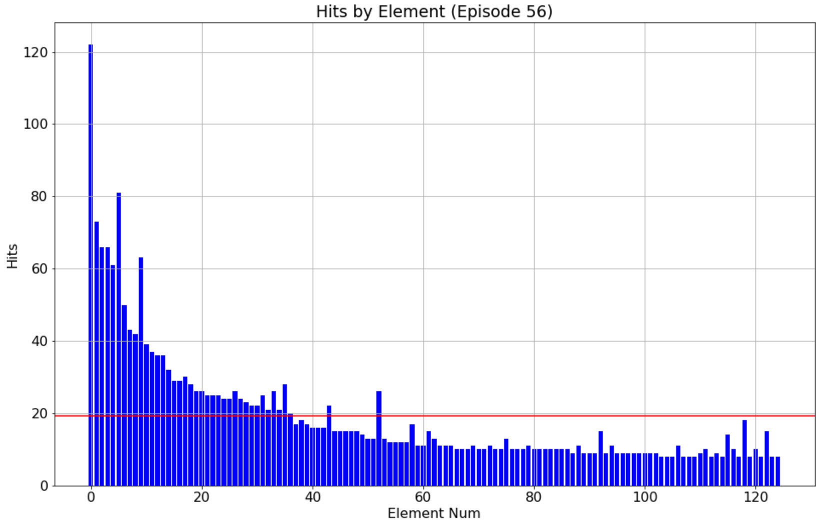
Fit Quality: Large Perturbation

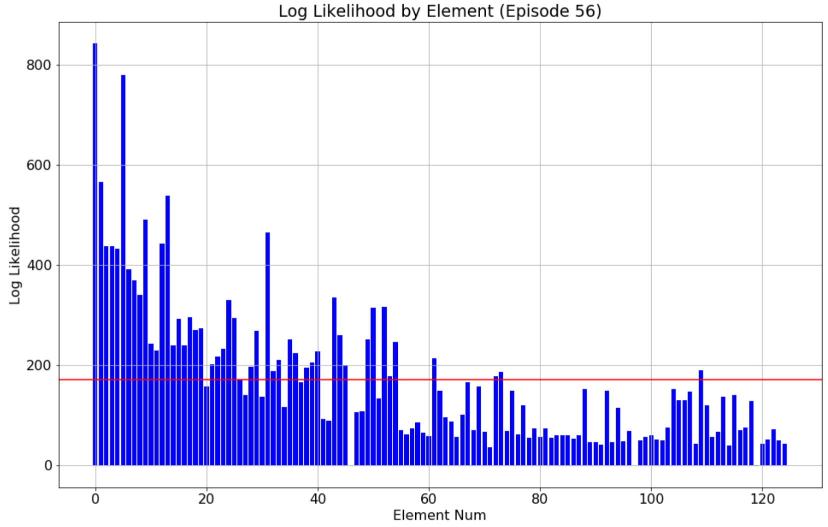


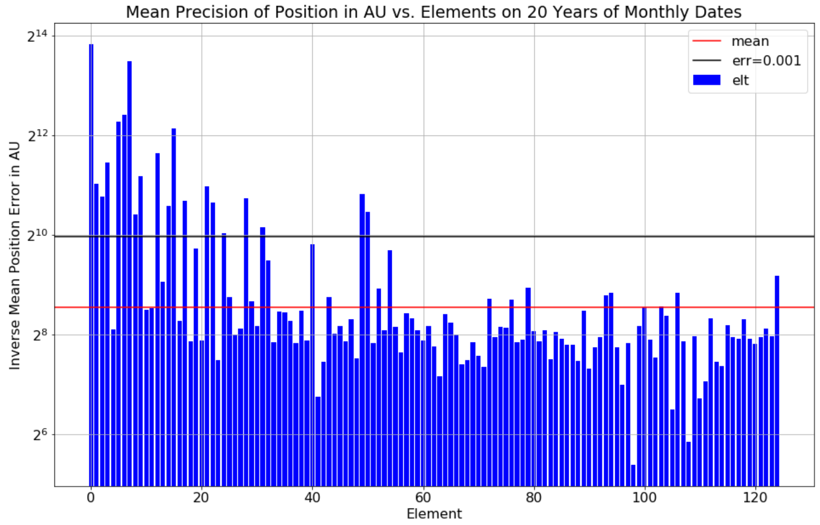


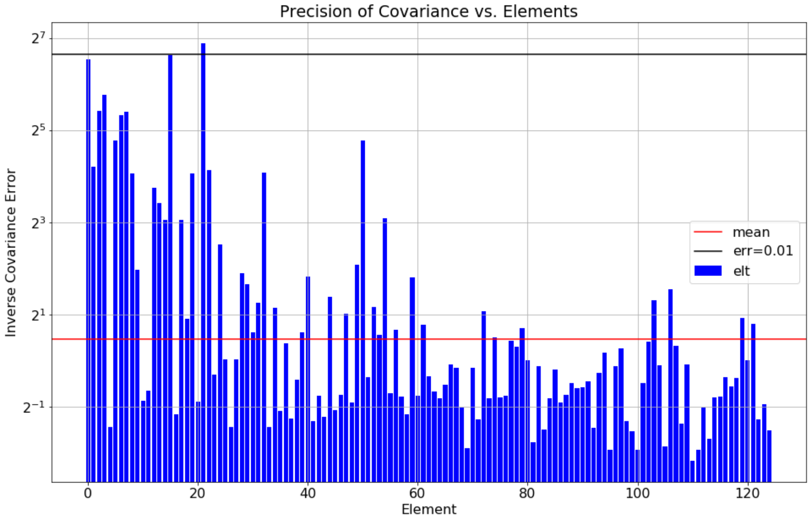
- Do recovered elements match the nearest asteroid? Quite well.
- 4.5E-4 AU mean distance
- 0.032 covariance norm
- This is a decent fit on the 12 elements that have been recovered
- This is a lot harder than the last task-like facing a high school pitcher…

Search Known Asteroids with Random Initializations









- ~4096 random seeds
- Each batch started with 1024 random elements
- Selected best 64 by mean log(v)
- Trained for ~5 days on 4 GPUs
- Reported fits with ≥ 8 hits and resolution < 20 arc seconds
- Recovered Elements: 125
- Hits: 19.2
- Resolution: 9.3 arc seconds
- Nearest Asteroid Distance: 2.66E-3 AU
- Nearest Asteroid Cov. Norm: 0.73
- Comments on fit quality:
  - Decent fit on some
  - Probably spurious on others
  - Overall shows that this can work
  - But not ready for production
- Baseball analogy continued: facing Roger Clemens, but trying to get hit by the pitch to get on base cheaply

Masters Thesis Conclusions

- Prove that asteroid search over orbital elements works
  - Need an adequate initialization and representation in data set
- Built a working prototype in TensorFlow
  - First demonstration of efficient astrometric computations on GPU?
- High quality integration of the Solar System and astrometric directions
  - Associated each of 5.7E6 ZTF observations to nearest asteroid
  - 4.2E12 interactions, possibly novel and useful data set
- Proof of concept for an automated asteroid search pipeline
  - Random initialization is not going to work, need to initialize intelligently

Progress Since June 2020

New Baby + Pandemic + House Move $\stackrel{?}{=}$ Research

- Between a new baby (Ruth, August 2020)
- and pandemic parenting
- and moving from Boston to Newton…
- Research output has been “suboptimal”
- But I have made a fair amount of progress on the unglamorous back end

From Numpy to Database Back End

- Masters Thesis used “default” strategy to manage data
    - Loose Numpy files in a data folder, organized carefully
- Efficient for prototyping, but does not scale for “big data” applications
- Started with 5.7 million detections in early snapshot
    - Painful process to upload new data
    - ZTF dataset is now up to 158 million candidate asteroid detections!
- Also, what if we want to merge data from multiple surveys, e.g. Pan-STARRS or the upcoming Vera Rubin telescope?
- Solution: put all the data into a powerful SQL database!
    - I chose a Maria-DB instance because it is fully open source

Setting up a SQL Database

- Running an instance of a SQL database is easier than you think
  - Popular variants include MySQL, MariaDB, PostgreSQL and MS SQL Server
- If you a have a Linux computer, it’s almost as easy as

  ```
  $ sudo apt install mariadb-server && sudo apt install mariadb-client
  ```
- If you want to do serious computation, it is probably more practical to use an instance hosted by a cloud provider e.g. AWS
  - I am both crazy and a dinosaur, so I have a rack mounted server in my basement
  - This is probably not a great strategy for non crazy people
- But installing your own instance is still a great way to learn and do low cost prototyping before you start paying for big rented instances

Python Database Connections

- Extracting data from a SQL database into Python is easy
- Pandas DataFrame is familiar to most of us, very similar abstraction to a SQL resultset
  - Utility functions sql2df, sp2df convert a SQL statement or stored procedure call into a Pandas DataFrame. Very nice!
- Writing to SQL databases from Python is harder than it should be
  - Pandas comes with a canned method to this,
  - Works fine on toy data sets, but it is execrably slow on real data…
  - converting every row to be insert to a line of SQL text
  - I ended up writing a function to convert a DataFrame into CSVs in parallel with with Dask; this isn't great, but it's usable

SQL Database 101 by Example

Asteroid — Enter a SQL expression to filter results (use Ctrl+Space)

	AsteroidID	AsteroidNumber	AsteroidName	BodyID	H	G
1	1	1	Ceres	1,000,001	3.4	0.12
2	2	2	Pallas	1,000,002	4.2	0.11
3	3	3	Juno	1,000,003	5.33	0.32
4	4	4	Vesta	1,000,004	3	0.32
5	5	5	Astraea	1,000,005	6.9	0.15
6	6	6	Hebe	1,000,006	5.8	0.24
7	7	7	Iris	1,000,007	5.6	0.15
8	8	8	Flora	1,000,008	6.5	0.28
9	9	9	Metis	1,000,009	6.3	0.17
10	10	10	Hygiea	1,000,010	5.5	0.15

Column Name	#	Data Type	Not Null	Auto Increment	Key	Default	Expression	Comment
AsteroidID	1	int(11)	[v]	[ ]	PRI			Internal integer ID for Asteroids; in practice this is the same as AsteroidNumber field
AsteroidNumber	2	int(11)	[v]	[ ]	UNI			The IAU asteroid number when it exists; otherwise a sequential counter starting at 1,000,000
AsteroidName	3	varchar(32)	[v]	[ ]	UNI			The IAU asteroid name where applicable or asteroid designation
BodyID	4	int(11)	[v]	[ ]	UNI			Foreign key to KS.Body table
IsNumberedAsteroid	5	tinyint(1)	[v]	[ ]				Flag indicating whether or not this is an IAU numbered asteroid
H	6	double	[v]	[ ]				The H brightness parameter
G	7	double	[v]	[ ]				The G brightness parameter

Horizons State Vectors as DB Tables

HorizonsVectors — Enter a SQL expression to filter results (use Ctrl+Space)

	TimeID	HorizonsBodyID	mjd	qx	qy	qz	vx	vy	vz
1	58,176,000	1	40,400	0.362	-0.117	-0.043	0.003	0.028	0.002
2	58,176,000	2	40,400	0.613	-0.397	-0.041	0.011	0.017	-0
3	58,176,000	3	40,400	0.121	-1.009	-0	0.017	0.002	0
4	58,176,000	4	40,400	-0.11	-1.459	-0.028	0.014	0	-0
5	58,176,000	5	40,400	-5.38	-0.851	0.124	0.001	-0.007	0
6	58,176,000	6	40,400	7.894	4.837	-0.398	-0.003	0.005	0
7	58,176,000	7	40,400	-18.265	-1.166	0.233	0	-0.004	-0
8	58,176,000	8	40,400	-16.055	-25.706	0.899	0.003	-0.002	-0
9	58,176,000	9	40,400	-30.483	2.744	8.523	0	-0.003	0
10	58,176,000	10	40,400	0.005	0.001	-0	-0	0	-0

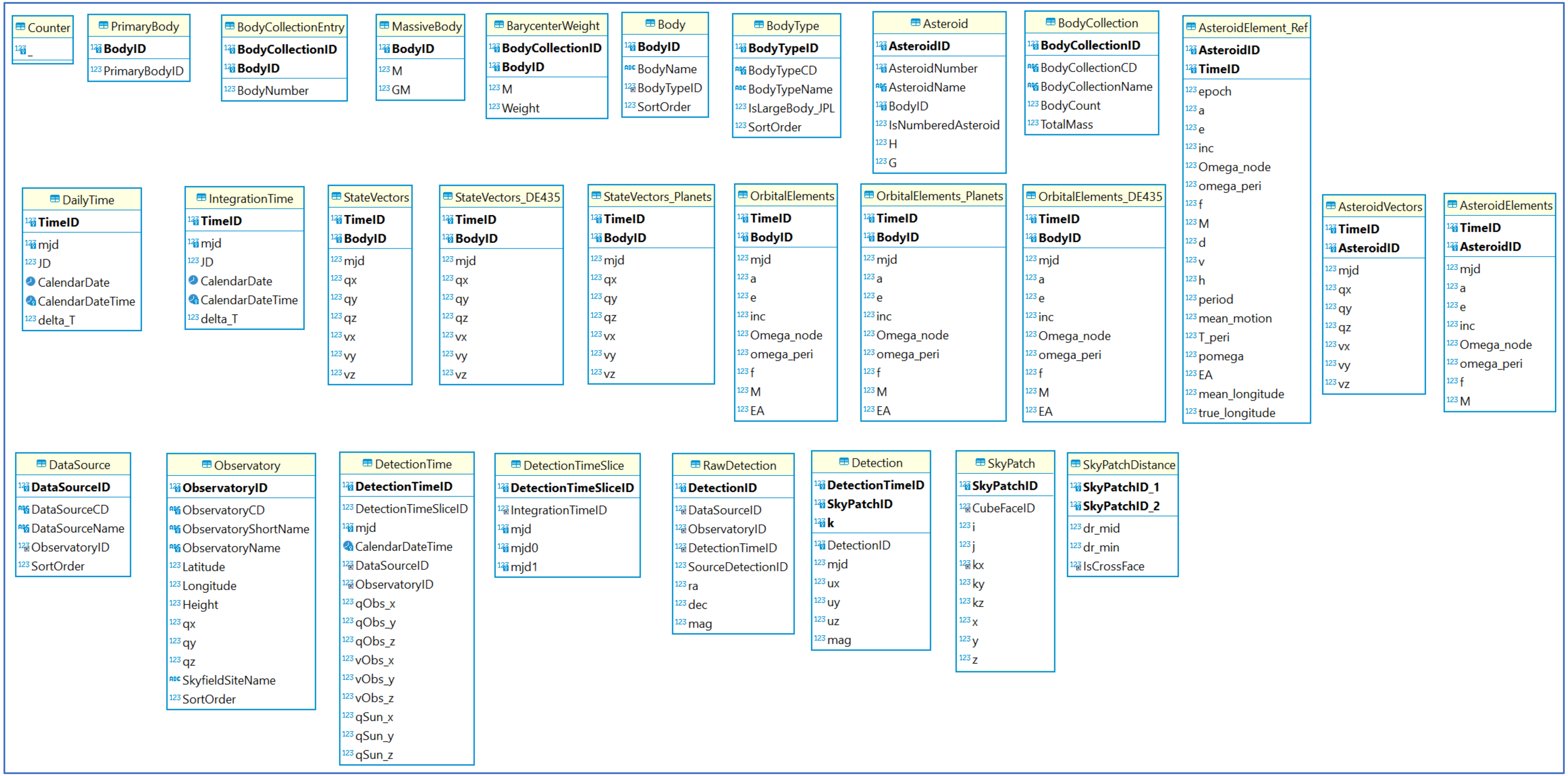
HorizonsBody — Enter a SQL expression to filter results (use Ctrl+Space)

	HorizonsBodyID	HorizonsBodyName	Body	BodyID
1	1	LB.Mercury Barycenter	2	1
2	2	LB.Venus Barycenter	2	2
3	3	LB.Earth-Moon Barycenter	2	3
4	4	LB.Mars Barycenter	2	4
5	5	LB.Jupiter Barycenter	2	5
6	6	LB.Saturn Barycenter	2	6
7	7	LB.Uranus Barycenter	2	7
8	8	LB.Neptune Barycenter	2	8
9	9	LB.Pluto Barycenter	2	9
10	10	LB.Sun	1	10
11	199	LB.Mercury	3	199
12	299	LB.Venus	3	299
13	301	LB.Moon	4	301
14	399	LB.Earth	3	399

```sql
SELECT
    b.BodyID,
    b.BodyName,
    COALESCE(mb.M, 0.0) AS m,
    hv.qx,
    hv.qy,
    hv.qz,
    hv.vx,
    hv.vy,
    hv.vz
FROM
    JPL.HorizonsVectors AS hv
    INNER JOIN JPL.HorizonsBody AS hb ON hb.HorizonsBodyID = hv.HorizonsBodyID
    INNER JOIN JPL.HorizonsTime AS ht ON ht.TimeID = hv.TimeID
    INNER JOIN KS.Body AS b ON b.BodyID = hb.BodyID
    LEFT JOIN JPL.MassiveBody AS mb ON mb.HorizonsBodyID = hv.HorizonsBodyID
WHERE
    hb.HorizonsBodyName = HorizonsBodyName AND ht.TimeID = @TimeID;
```

- HorizonsBodyID is called a **primary key**
- Use integer identifiers for performance
- HorizonsVectors includes HorizonsBodyID
- This is called a **foreign key** in DB design
- Follow a simple naming rule
    - TableNameID is the PK to TableName…
- SQL query at left selects the state vectors for the sun and planets at a given date

Entity Relationship Diagram: KS



ER Diagram: JPL and ZTF

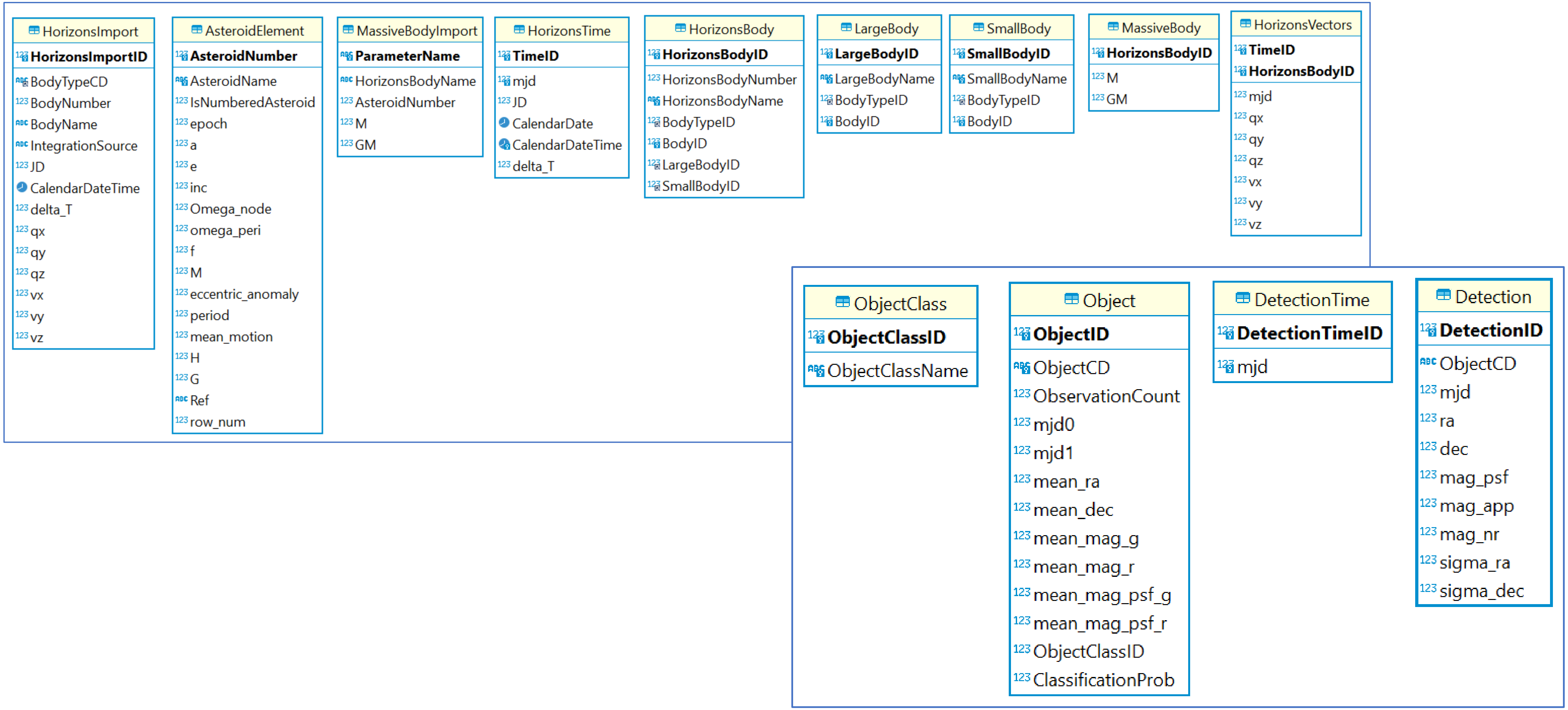


Table Descriptions and Sizes

Table Name	Engine	Row Count	Data Length	Index length	Description
AsteroidElements	Aria	3,595,918,656	291G	109G	Keplerian orbital elements (a, e, i, Omega, omega, f) for all known asteroids computed in rebound using the planets as massive bodies and initial conditions from DE435 at MJD 59000.
AsteroidVectors	Aria	3,595,918,656	263G	109G	State vectors (position and velocity) for asteroids computed in Rebound using the planets as massive bodies and initial conditions from DE435 at MJD 59000.Initial conditions for aster
SkyPatchDistance	Aria	1,861,084,200	57G	74G	Distance bewteen two SkyPatch cells; only cataloged for neighbors that are reasonably close.
OrbitalElements_DE435	Aria	253,440,352	20G	6.2G	State vectors (position and velocity) for Solar Systems bodies computed in rebound using the planets as massive bodies and initial conditions from DE435 at MJD 59000. Includes reco
StateVectors_DE435	Aria	254,160,353	18G	6.2G	State vectors (position and velocity) for Solar Systems bodies computed in rebound using all the massive bodies from the DE-435 integration with initial conditions at MJD 59000. Inclu
SkyPatchGridDistance	Aria	310,180,328	12G	12G	Distance bewteen two SkyPatchGrid cells; only cataloged for neighbors that are reasonably close.
RawDetection	Aria	158,843,412	9.7G	5.6G	Detections of possible asteroids across multiple data sources, as quoted in RA/DEC by original source.
SkyPatch	Aria	25,165,824	3.6G	1.1G	Collection of discrete patches of the sky corresponding to a cube in which the unit sphere is inscribed.
OrbitalElements_Planets	Aria	43,200,010	3.5G	1.1G	State vectors (position and velocity) for Solar Systems bodies computed in rebound using the planets as massive bodies and initial conditions from DE435 at MJD 59000.Includes recor
StateVectors_Planets	Aria	47,520,011	3.5G	1.2G	State vectors (position and velocity) for Solar Systems bodies computed in rebound using the planets as massive bodies and initial conditions from DE435 at MJD 59000. Includes reco
SkyPatchGrid	Aria	4,194,304	696M	43M	SkyPatchGrid describes the 4N^2 square grid cells on one major face
Counter	Aria	16,777,216	675M	144M	Utility table - enables a SQL query to emulate a for loop. Data type is signed integer; range is non-negative integers up to 2^24.
IntegrationTime	Aria	10,713,601	442M	359M	Distinct time stamps at which MSE integrated positions of solar system bodies are available.
AsteroidElement_Ref	Aria	958,695	124M	32M	Orbital elements of asteroids as of reference dates; used to initialize integrations. Primary for these elements is the Sun, NOT the Solar System Barycenter! See https://rebound.readthed
CounterSigned	Aria	2,097,151	84M	29M	"Utility table - enables a SQL query to emulate a for loop. Data type is signed integer; range is non-negative integers up to 2^24.";
Asteroid	Aria	958,724	46M	33M	Census of all known asteroids including reference to KS.Body table. Orbital Elements stored separately.
Body	Aria	958,809	38M	17M	Solar System bodies used in the Kepler Sieve application.
Detection	Aria	676,053	38M	18M	Detections of possible asteroids across multiple data sources; enriched with unit direction and SkyPatch.
DetectionTime	Aria	216,702	24M	9.8M	Time at which one or more detections were made an observatory. Cartesian position and velocity includes topos adjustment.
DetectionTimeSlice	Aria	165,284	6.8M	12M	Time at which one or more detections were made an observatory. Cartesian position and velocity includes topos adjustment.
DailyTime	Aria	37,201	1.5M	1.3M	Distinct time stamps at sampled once per day.
Minutes	Aria	1,440	72K	32K	Enumerate the 1440 minutes in a day to support linear interpolation in queries.
BarycenterWeight	Aria	438	32K	24K	Weighting factors for bodies in collections.
BodyCollectionEntry	Aria	438	32K	32K	Members of body collections.
MassiveBody	Aria	354	24K	16K	Mass of heavy objects included in DE 435 integration, sources from technical comments.
BodyCollection	Aria	16	16K	32K	Collections of bodies used in solar system integrations.
BodyType	Aria	6	16K	24K	Types of Solar System bodies.
CubeEdge	Aria	12	16K	24K	The 12 edges of a cube characterized by the pair of vertices sorted in ascending order.
CubeEdgeVertex	Aria	24	16K	16K	Relate a cube edge to a cube vertex if the vertex is one end of the edge.
CubeFace	Aria	6	16K	24K	The six faces of a cube described as the index of the constant axis and its value.
CubeFaceEdge	Aria	24	16K	24K	Relate a cube face to a cube edge if the edge is on the perimeter of the face.
CubeFaceNeighbor	Aria	6	16K	16K	Relate a cube face to its four neighbors based on the direction of approach.
CubeFacePair	Aria	24	16K	16K	Each pair of distinct cube faces that are connected by an edge
CubeVertex	Aria	8	16K	24K	The eight vertices of a cube where each coordinate is at +/- 1.
DataSource	Aria	1	16K	40K	Data sources for asteroid detections.
Observatory	Aria	2	16K	40K	Astronomical observatories and their position on earth.
OrbitalElements	Aria	0	8K	8K	State vectors (position and velocity) for Solar Systems bodies computed in rebound using the planets as massive bodies and initial conditions from DE435 at MJD 59000.
PrimaryBody	Aria	0	8K	8K	Default primary for each body when integrated in the Planets or DE435 collections. Almost always the Sun except that the primary of the Moon is Earth.
StateVectors	Aria	0	8K	8K	State vectors (position and velocity) for Solar Systems bodies computed in Rebound using the planets as massive bodies and initial conditions from DE435 at MJD 59000. Not limited t

What Next?

Enrich ZTF Detection Data

- Compute direction $u_{obs} = (u_x, u_y, u_z)$ for every detection
- Assign integer SkyPatchID to every detection
- Associate every ZTF detection with nearest known asteroid
  - Populate a table keyed by (DetectionID, TimeSliceID) with payload including predicted direction and SkyPatchID
  - Join this against ZTF detections with match on TimeSliceID and SkyPatchID matching only on nearby parts of the sky
- Compute the exact position for every candidate asteroid that might be the nearest to each detection, then take the closest one

Generate Candidate Elements from Tracklets

- A set of orbital elements has 6 degrees of freedom
- One detection has 2 D.O.F., RA and DEC; a tracklet has 4
  - In theory a set of three detections close together has 6, but in practice you only get 4, RA, DEC, and their time derivatives
  - If you have a third detection or tracklet that is separated in time, you should be able to fit orbital elements if it they are consistent with a realistic orbit
- To discover new asteroids, probably need some way to generate a 2D space of candidate orbital elements consistent with a single tracklet
- Current idea: provisionally assign r (distance from sun to object) and r'; these follow a well behaved distribution and are easy to work with

Mille Grazie: Thank you for Your Attention!

- Questions?
- Comments?
- Suggestions?