Towards Real-Time Asteroid Tracking with Fink-FAT and Outfit

Roman Le Montagner
Fink@World
10/06/2026

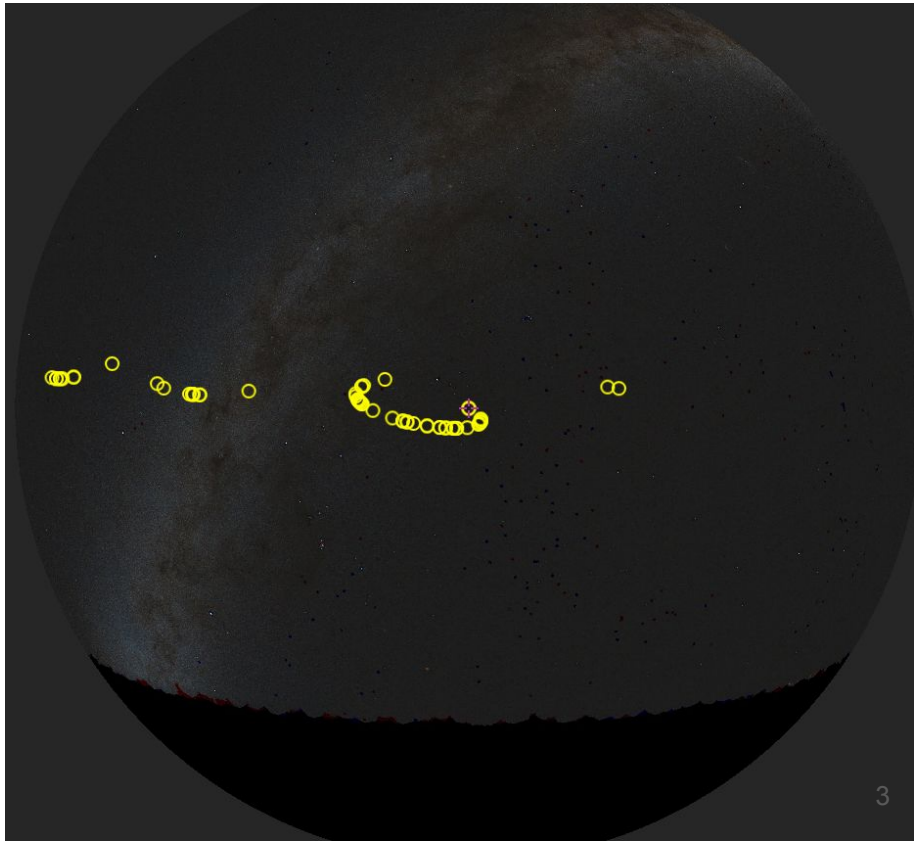
Goal : Finding new/unknown asteroides

- within the LSST alert stream through Fink
- Most of the object (Stars, Galaxies, …) are static objects

Goal : Finding new/unknown asteroides

- within the LSST alert stream through Fink
- ~~static objects~~ => moving object

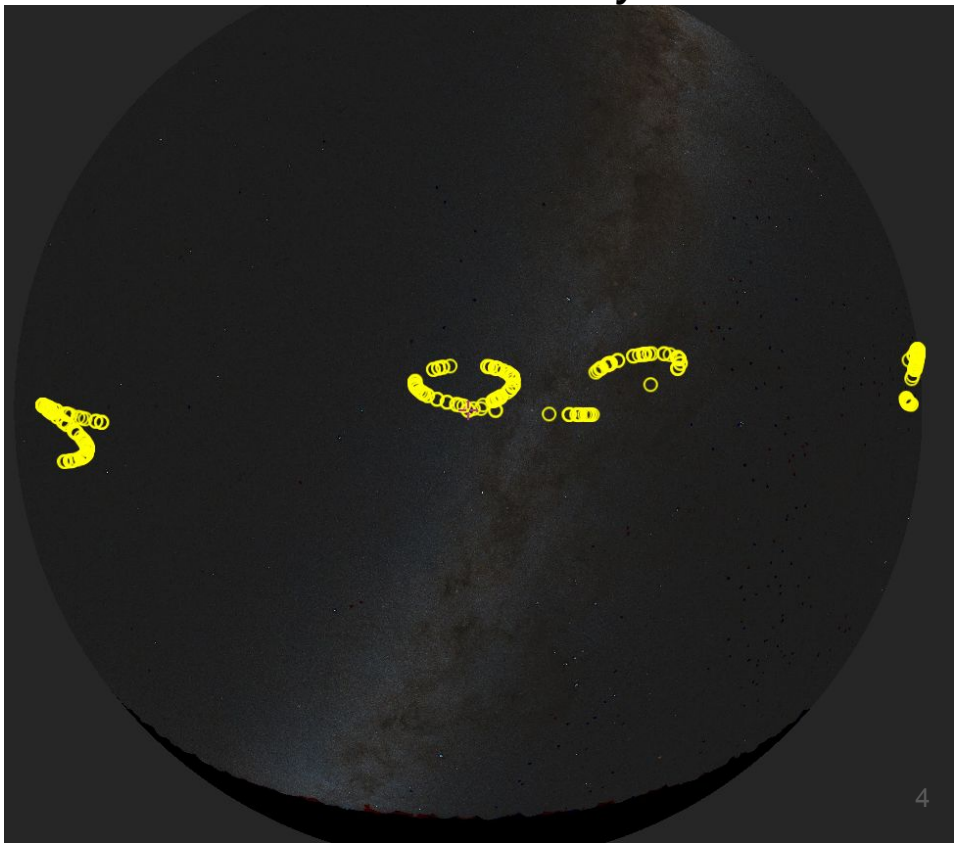
33803 JulienPeloton



Goal : Finding new/unknown asteroides

- within the LSST alert stream through Fink
- ~~static objects~~ => moving object

8467 Benoitcarry



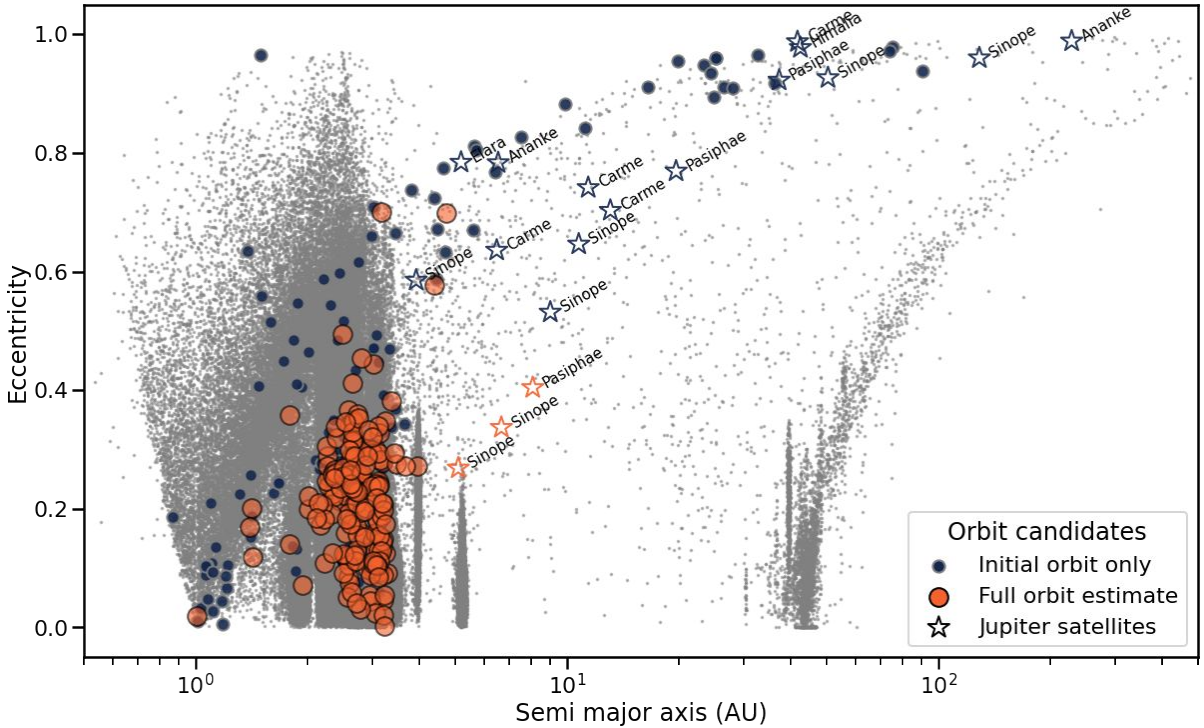
Goal : Finding new/unknown asteroides

- within the LSST alert stream through Fink
- ~~static objects~~ => moving object
- we need linking software
  - Moving Object Processing System (MOPS)(Kubica et al. 2007; Denneau et al. 2013)
  - HelioLinC (Holman et al. 2018)
  - ZTF’s Moving Object Discovery Engine (ZMODE) (Masci et al. 2019)
  - Tracklet-less Heliocentric Orbit Recovery (THOR) (Moeyens et al. 2021)

Goal : Finding new/unknown asteroides

- within the LSST alert stream through Fink
- ~~static objects~~ => moving object
- we need linking software
  - Moving Object Processing System (MOPS)(Kubica et al. 2007; Denneau et al. 2013)
  - HelioLinC (Holman et al. 2018)
  - ZTF's Moving Object Discovery Engine (ZMODE) (Masci et al. 2019)
  - Tracklet-less Heliocentric Orbit Recovery (THOR) (Moeyens et al. 2021)
  - Fink-FAT (LeMontagner et al. 2024)

- Linking algorithm in Python / Pandas
- Orbit fitting using Orbfit + Spark



- Linking algorithm in Python / Pandas
- Orbit fitting using Orbfit + Spark

Several issues with original Fink-FAT :

- Linking code maintenance very difficult
- Orbift + spark inefficient
- linking one month of ZTF alerts < 3 hours
  - orbit fitting = 2 hours @ 24 cores
  - < 1 hours for linking (single core)
  - Recall < 45%

FINK-FAT in Rust

- more expressive syntax than python + pandas
- Efficiency leveraging a compiled language
- memory safety without garbage collector
- Built-in concurrency safety, making parallel code easier and safer to write
- Like in python, rust have a package registry (crates.io)

Fink-FAT ecosystem (3 crates)

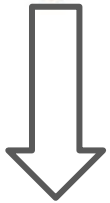
PHOTOM

https://crates.io/crates/photom

- Rust library for loading, structuring and querying astronomical observation datasets
  - Multi-observer support
  - Many IO supports
    - Parquet
    - MPC-80 columns
    - ADES
    - custom Serialization / Deserialization
  - Parallel Iteration
  - Coordinates Conversion
    - Equatorial - Ecliptic - Cartesian unit sphere
    - Full uncertainty propagation through jacobian

Fink-FAT ecosystem (3 crates)

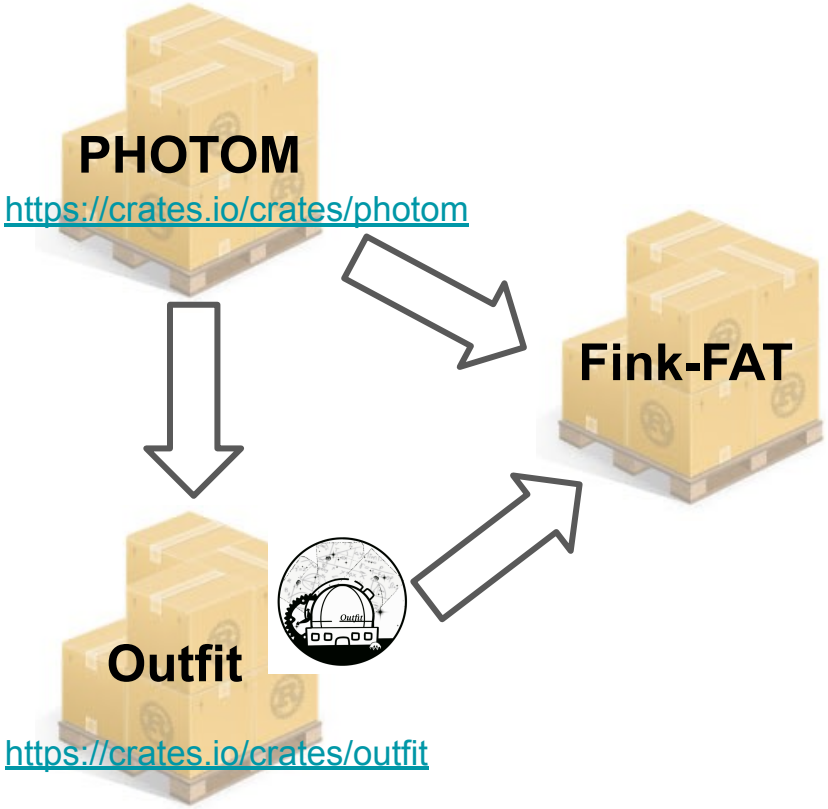
https://crates.io/crates/photom



https://crates.io/crates/outfit

- Orbit Determination toolkit
  - Initial Orbit Determination (Gauss method)
  - Differential correction (Least Square)
  - Sequential + Parallel API
  - Ephemerides (2-body and N-body)
  - Multiple orbits representation with respective conversion
    - Keplerian
    - Equinoctial
    - Cometary

Fink-FAT ecosystem (3 crates)



- Linking toolkit
  - Build tracklets candidates
  - Propagate tracklets to extend them
  - Fit orbits when enough points

Fink-FAT in a nutshell

Seeding **only intra-night** - How we starts the trajectory candidates ?

1. Build pairs of alerts (a,b)

Cuts :

- magnitude similarity : $|m_a - m_b| \leq \Delta m_{max}$
- angular speed limit : $\Delta\theta(a,b) / (t\_b - t\_a) \leq \omega\_max$

Algorithm efficiency :

- Spatio-temporal bucketing using healpix

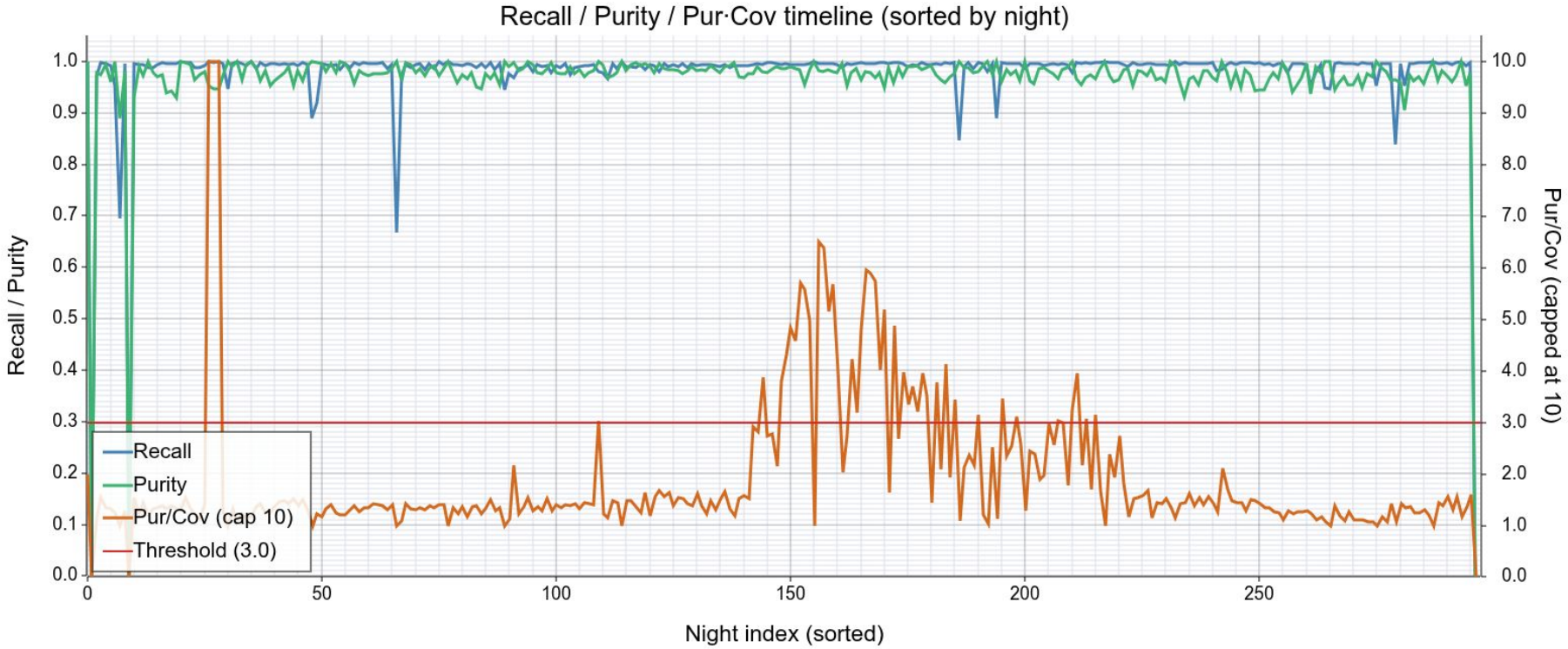
Fink-FAT in a nutshell

Seeding **only intra-night**

1. Build pairs of alerts (a,b)
2. Build triplets of alerts (a,b,c)

- extend generated pairs if any third alerts in the same night can match
- Same cuts than pairs generation but between b and c
    - + cuts on quadratic fit residuals on tangent plane around the observation a
- reusing the same spatio-temporal buckets

Fink-FAT in a nutshell

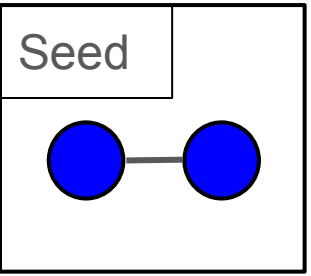


Fink-FAT in a nutshell

- From Seeds to Tracklets

Each pairs / triplets is transform into a **kalman filter** (Optimal recursive iterator)

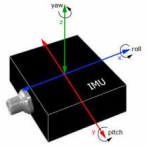
Swerling (1958), Kalman (1960), Kalman-Bucy (1961)



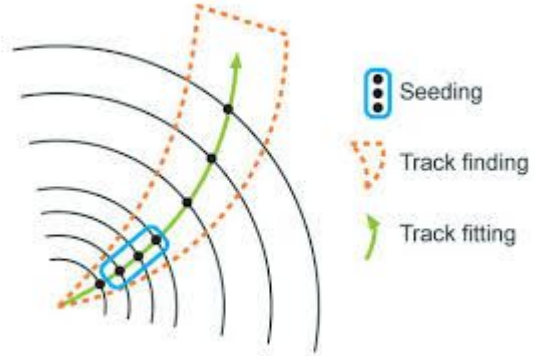
$\mathbf{x}_k$ - State matrix

$\mathbf{P}_{k|}$ - Covariance matrix

Applications



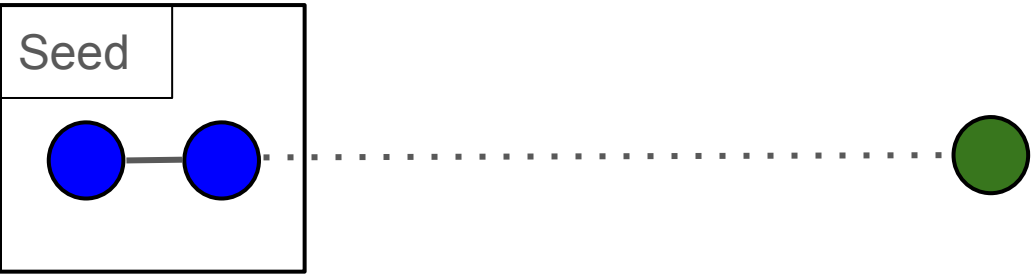
- Robotics
- IMU — Inertial Measurement Unit
- Particle physics => Particle Tracking



Fink-FAT in a nutshell

- From Seeds to Tracklets

Each pairs / triplets is transform into a **kalman filter** (Optimal recursive iterator)



- F_k : transition matrix, how the state evolve with time
- B_k,u_k : external command and forces, if following external object with no control => B_k,u_k = 0
- Q_k : process noise matrix, uncertainty model, how much my predictions can drift from reality at each steps

Predict

Predicted state estimate

$$\hat{\mathbf{x}}_{k|k-1} = \mathbf{F}_k\hat{\mathbf{x}}_{k-1|k-1} + \mathbf{B}_k\mathbf{u}_k$$

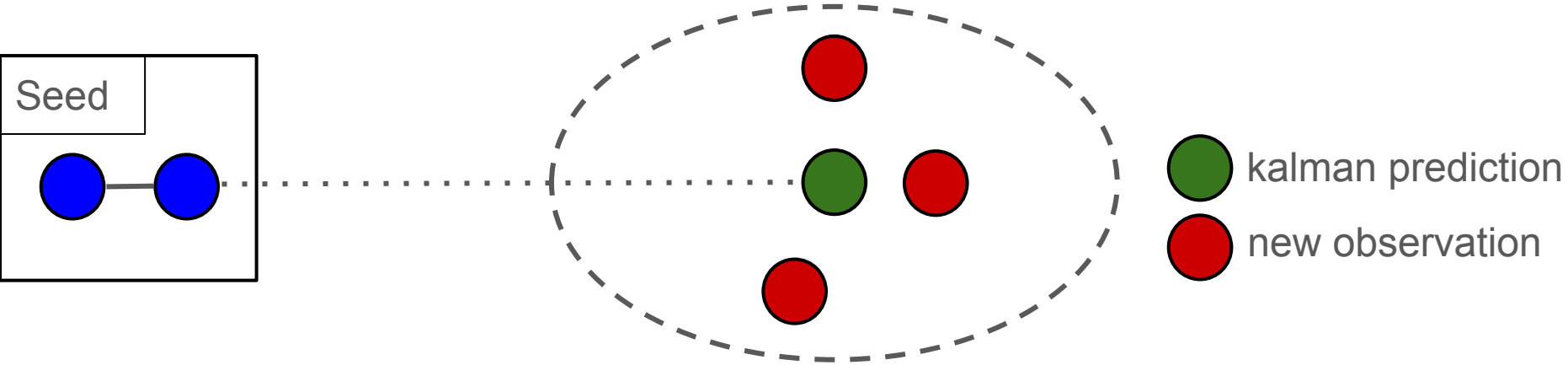
Predicted error covariance

$$\mathbf{P}_{k|k-1} = \mathbf{F}_k\mathbf{P}_{k-1|k-1}\mathbf{F}_k^T + \mathbf{Q}_k$$

Fink-FAT in a nutshell

- From Seeds to Tracklets

Each pairs / triplets is transform into a **kalman filter** (Optimal recursive iterator)



Predict

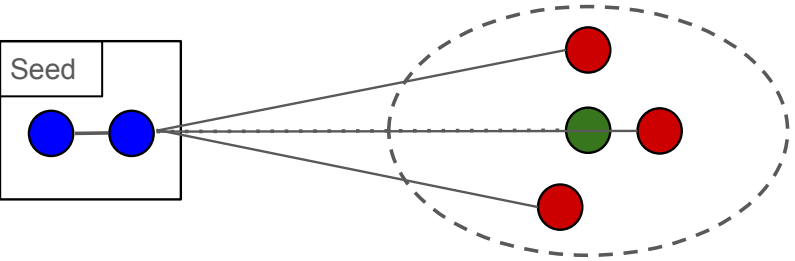
Predicted state estimate $\hat{\mathbf{x}}_{k|k-1} = \mathbf{F}_k\hat{\mathbf{x}}_{k-1|k-1} + \mathbf{B}_k\mathbf{u}_k$

Predicted error covariance $\mathbf{P}_{k|k-1} = \mathbf{F}_k\mathbf{P}_{k-1|k-1}\mathbf{F}_k^T + \mathbf{Q}_k$

Fink-FAT in a nutshell

- From Seeds to Tracklets

Each pairs / triplets is transform into a **kalman filter** (Optimal recursive iterator)



Correct

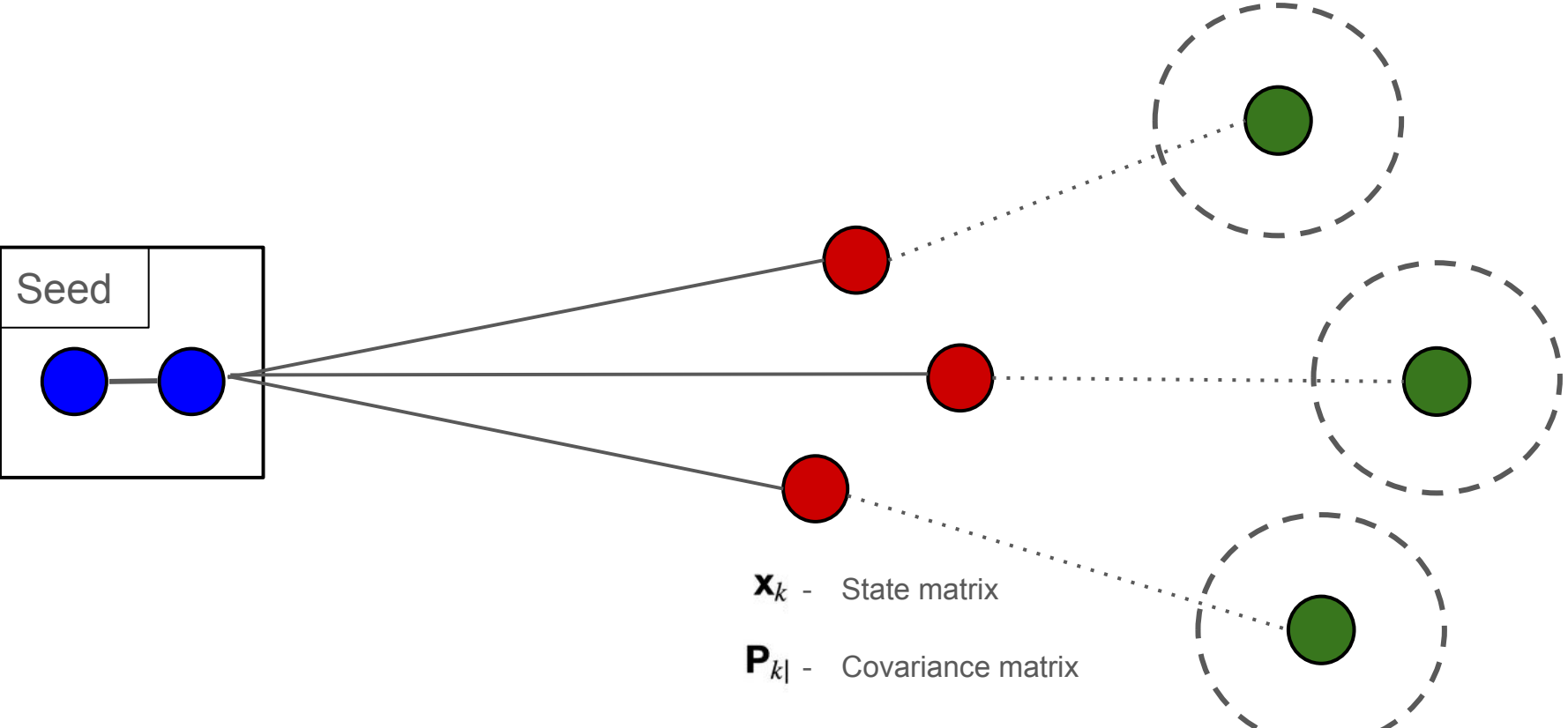
Innovation residual	$\tilde{\mathbf{y}}_k = \mathbf{z}_k - \mathbf{H}_k\hat{\mathbf{x}}_{k\|k-1}$
Innovation covariance	$\mathbf{S}_k = \mathbf{H}_k\mathbf{P}_{k\|k-1}\mathbf{H}_k^T + \mathbf{R}_k$
Optimal Kalman gain	$\mathbf{K}_k = \mathbf{P}_{k\|k-1}\mathbf{H}_k^T\mathbf{S}_k^{-1}$
Corrected state estimate	$\hat{\mathbf{x}}_{k\|k} = \hat{\mathbf{x}}_{k\|k-1} + \mathbf{K}_k\tilde{\mathbf{y}}_k$
Corrected estimate covariance	$\mathbf{P}_{k\|k} = (\mathbf{I} - \mathbf{K}_k\mathbf{H}_k)\mathbf{P}_{k\|k-1}$

- H_k : Observation Matrix, Projects the internal state into the sensor measurement space
- S_k : Total uncertainty in the measurement space (model + sensor)
- K_k : Optimal kalman gain, Balances trusts between model prediction and sensor measurements

Fink-FAT in a nutshell

- From Seeds to Tracklets

Each pairs / triplets is transform into a **kalman filter** (Optimal recursive iterator)



Fink-FAT in a nutshell

Kalman-based Asteroid tracking

x_k : State vector
(Ecliptic coordinates)

$$\mathbf{x} = (\lambda,\ \dot{\lambda},\ \ddot{\lambda},\ \beta,\ \dot{\beta},\ \ddot{\beta})^\top$$

F_k : Transition Matrix

$$F_3(\Delta t) = \begin{pmatrix} 1 & \Delta t & \frac{\alpha\Delta t - 1 + e^{-\alpha\Delta t}}{\alpha^2} \\ 0 & 1 & \frac{1 - e^{-\alpha\Delta t}}{\alpha} \\ 0 & 0 & e^{-\alpha\Delta t} \end{pmatrix}$$

Acceleration update with noise

$$\ddot{\lambda}_{k+1} = e^{-\alpha\Delta t}\,\ddot{\lambda}_k + \mathbf{w}_k, \qquad \mathbf{w}_k \sim \mathcal{N}(0,\ \sigma_a^2(1 - e^{-2\alpha\Delta t}))$$

Fink-FAT in a nutshell

x_k : State vector (Ecliptic coordinates)

$$\mathbf{x} = (\lambda, \dot{\lambda}, \ddot{\lambda}, \beta, \dot{\beta}, \ddot{\beta})^\top$$

Kalman-based Asteroid tracking

F_k : Transition Matrix

$$F_3(\Delta t) = \begin{pmatrix} 1 & \Delta t & \frac{\alpha\Delta t - 1 + e^{-\alpha\Delta t}}{\alpha^2} \\ 0 & 1 & \frac{1 - e^{-\alpha\Delta t}}{\alpha} \\ 0 & 0 & e^{-\alpha\Delta t} \end{pmatrix}$$

Acceleration update with noise

$$\ddot{\lambda}_{k+1} = e^{-\alpha\Delta t}\,\ddot{\lambda}_k + \mathbf{w}_k, \qquad \mathbf{w}_k \sim \mathcal{N}(0,\, \sigma_a^2(1 - e^{-2\alpha\Delta t}))$$

Based on Singer model (Singer, 1970)

- Models acceleration as a time-correlated random process (Ornstein-Uhlenbeck)
- One fixed parameters α

- If $\alpha\Delta t \ll 1$: $e^{-\alpha\Delta t} \approx 1$, acceleration is nearly constant => constant acceleration model
- If $\alpha\Delta t \gg 1$: $e^{-\alpha\Delta t} \approx 0$, acceleration is forgotten, white noise => constant velocity model

α = 0.05

Fink-FAT in a nutshell

Kalman-based Asteroid tracking

x_k : State vector (Ecliptic coordinates) $\mathbf{x} = (\lambda, \dot{\lambda}, \ddot{\lambda}, \beta, \dot{\beta}, \ddot{\beta})^\top$

F_k : Transition Matrix $F_3(\Delta t) = \begin{pmatrix} 1 & \Delta t & \frac{\alpha\Delta t - 1 + e^{-\alpha\Delta t}}{\alpha^2} \\ 0 & 1 & \frac{1 - e^{-\alpha\Delta t}}{\alpha} \\ 0 & 0 & e^{-\alpha\Delta t} \end{pmatrix}$

Estimating Optimal Tracking Filter Performance for Manned Maneuvering Targets

ROBERT A. SINGER, Member, IEEE
Hughes Aircraft Company
Ground Systems Group
Fullerton, Calif. 92634

https://fr.scribd.com/document/894422827/Singer-1970-Estimating

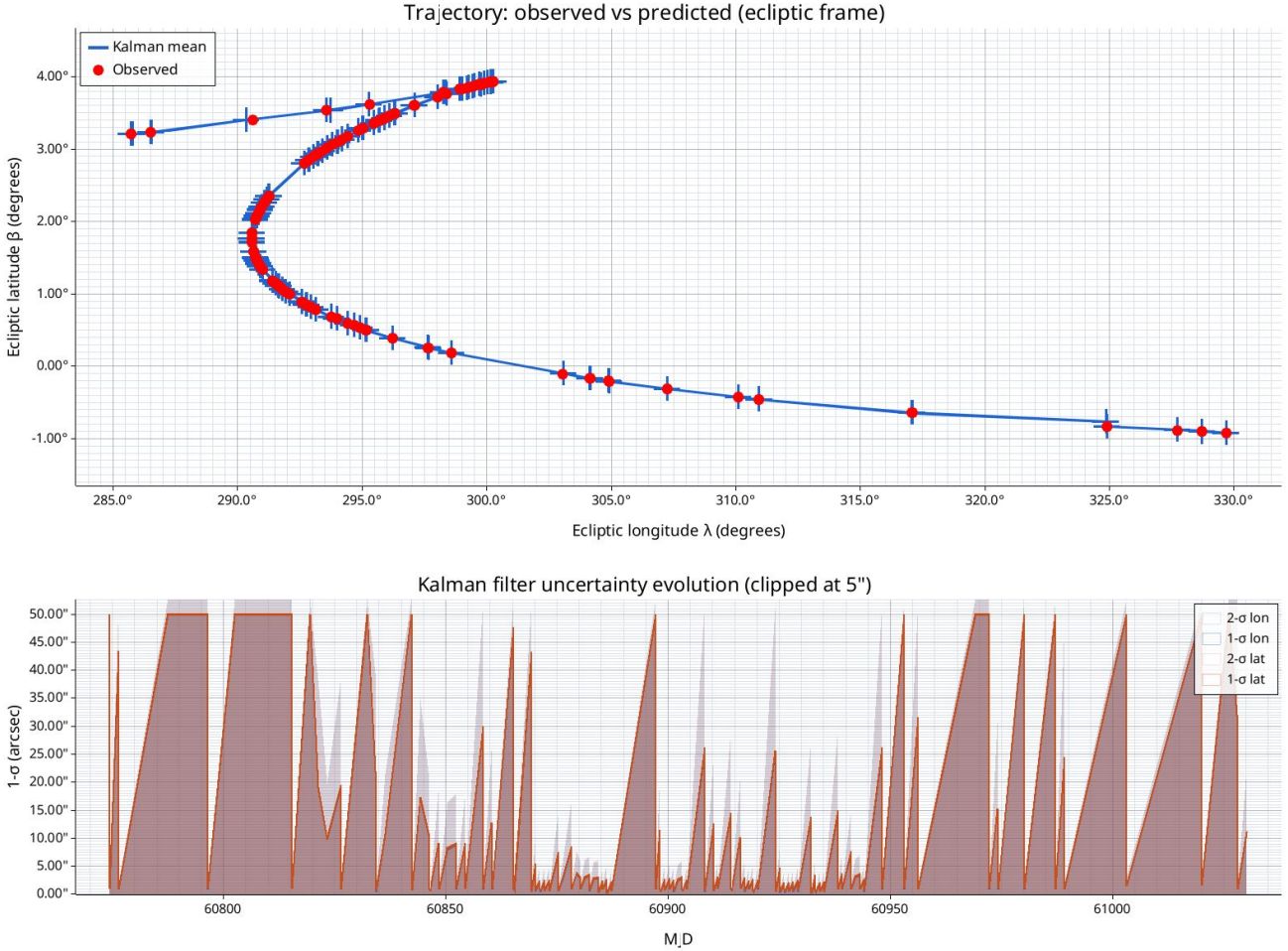
I. Introduction

Most tactical weapons systems require accurate tracking of manned maneuverable vehicles such as aircraft, ships, and submarines. Although much attention has been given in the literature to tracking orbital, suborbital, and reentering targets, little [1], [2] has been given to this important problem of tracking piloted threats. This paper treats this problem by first developing a simple target model that closely represents the ensemble behavior of different maneuvering vehicles and that when used in the appropriate Kalman filter yields a tracking algorithm that provides optimal performance for this class of targets and that is easily implemented.

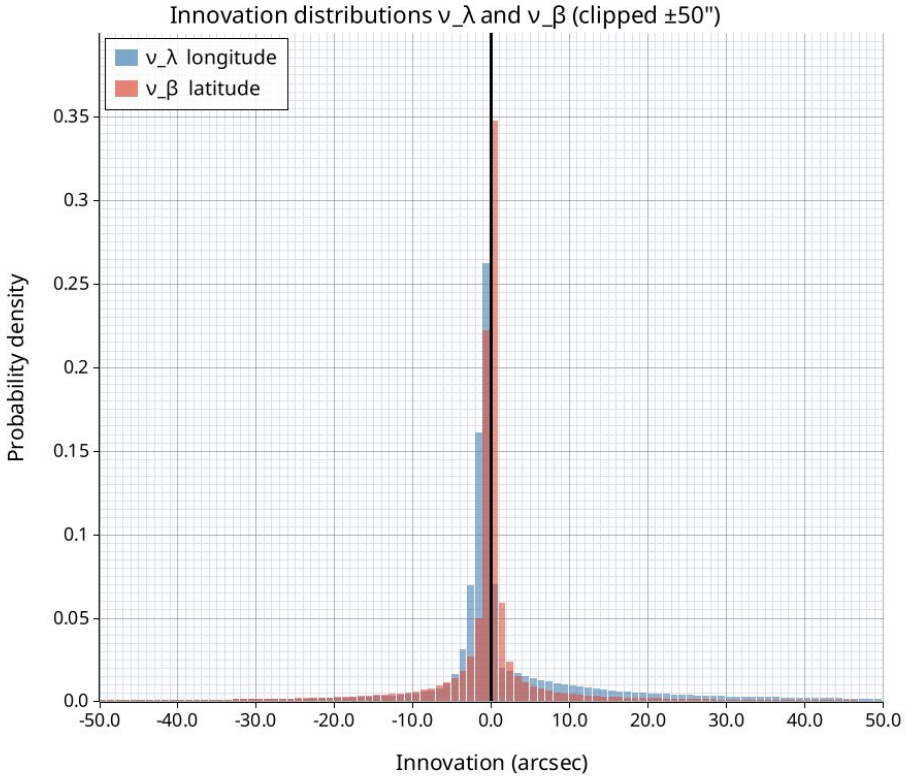
- If $\alpha\Delta t \gg 1$: $e^{-\alpha\Delta t} \approx 0$, acceleration is targetted, white noise [illegible]

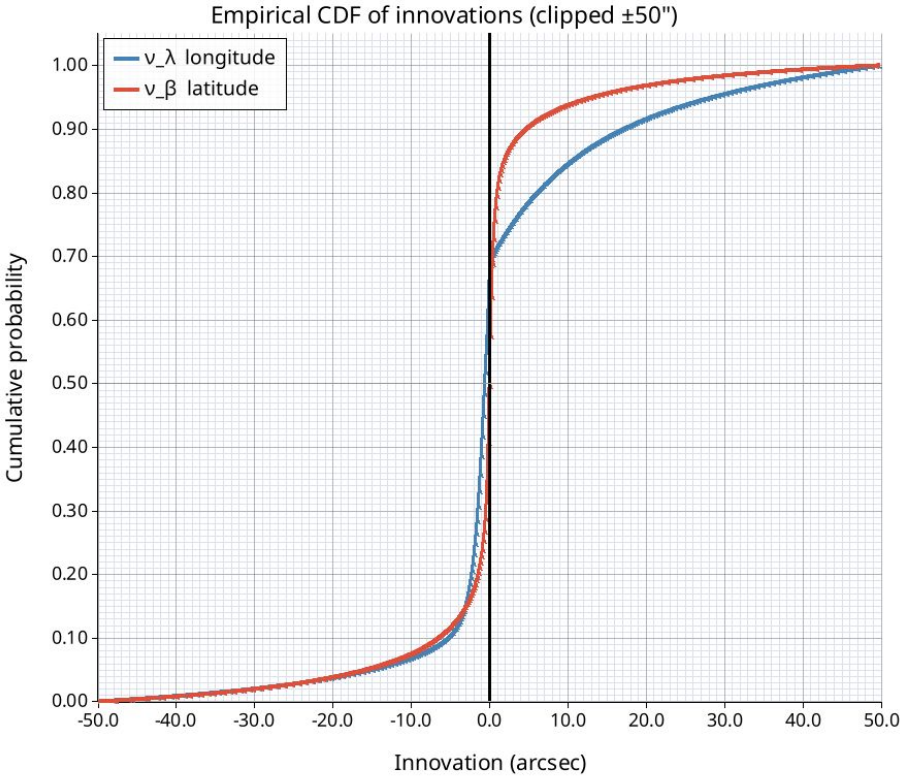
  constant velocity model

Fink-FAT in a nutshell

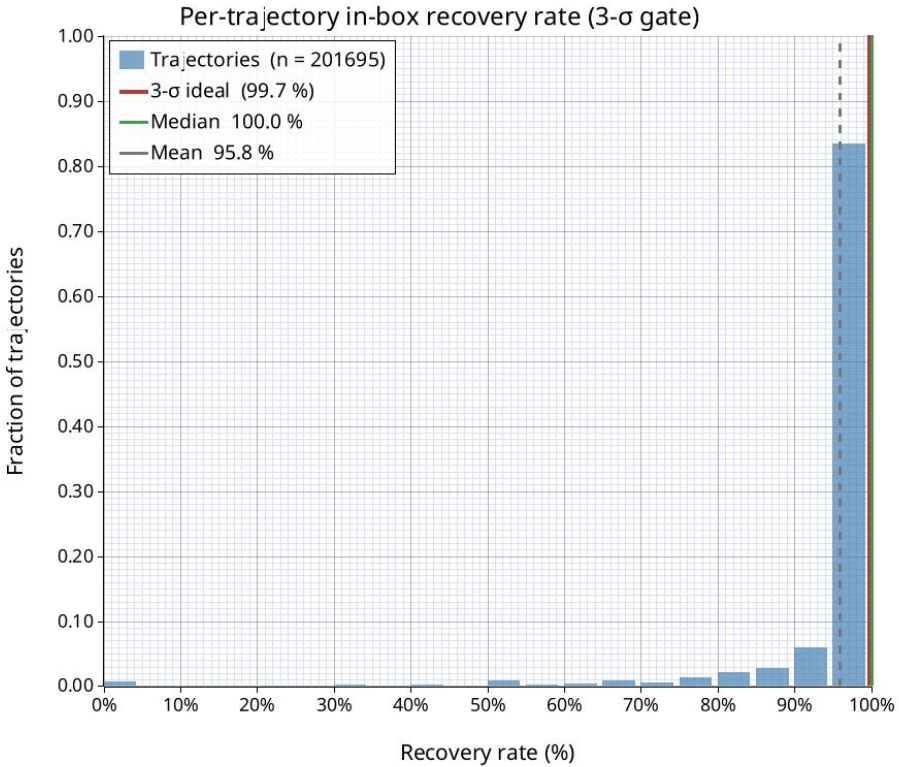


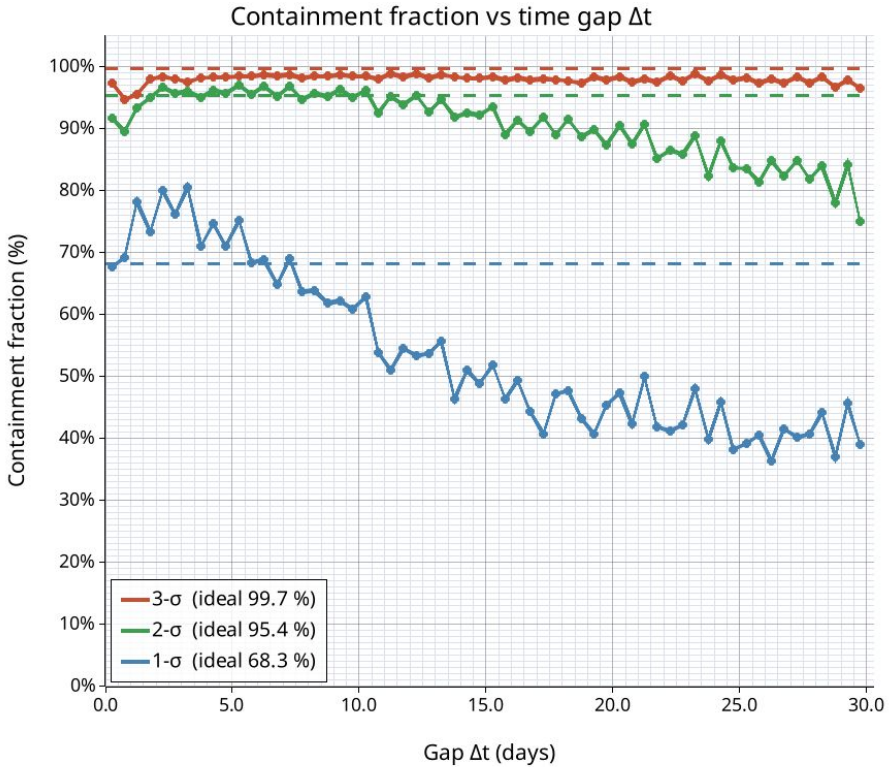
Fink-FAT in a nutshell



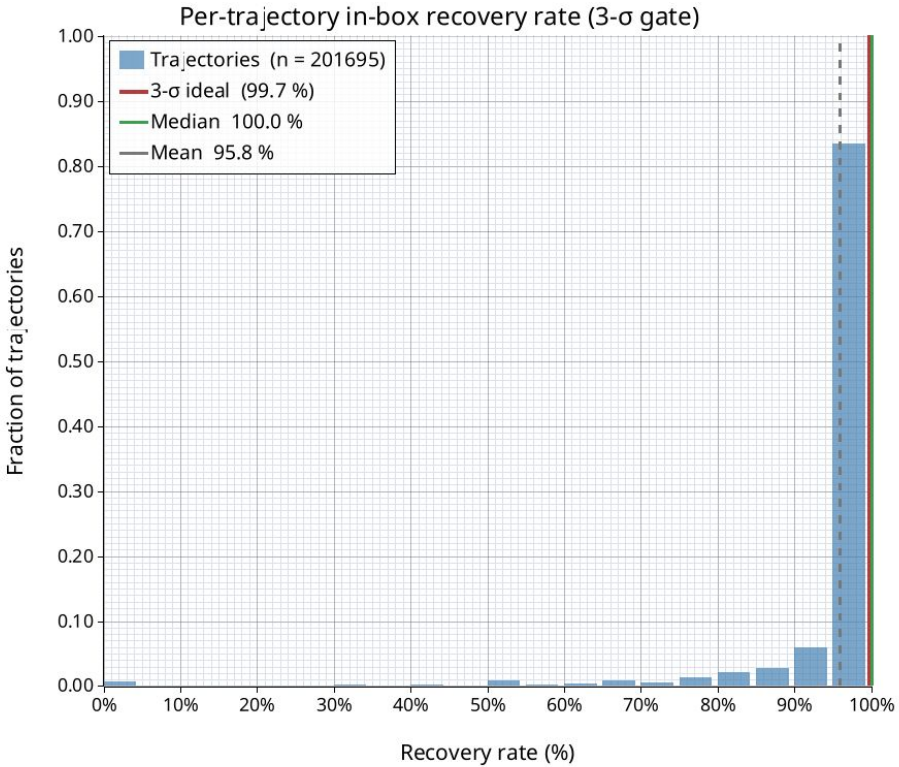


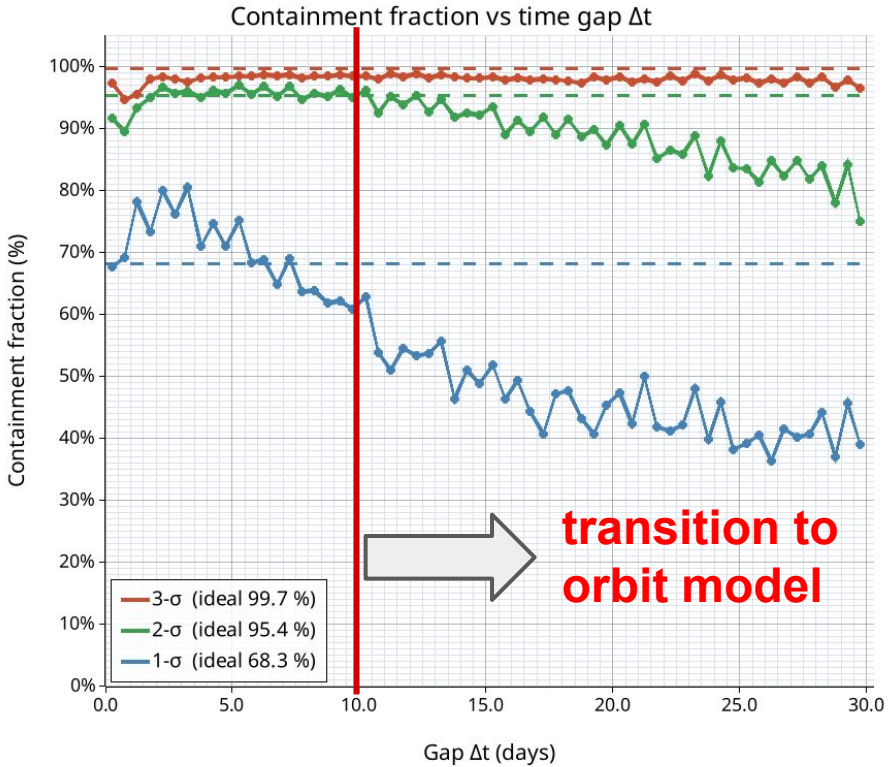
Fink-FAT in a nutshell





Fink-FAT in a nutshell





Fink-FAT - Reconstruction performances on ZTF

Full 2025 dataset -> 297 nights processed

Data volume

Metric	Total	Mean / night
Observations	6,346,308	21,368
Associated observations	2,904,292	9,779
New seeds created	1,494,765	5,033

Time performances

Step	Mean (ms)	Median (ms)	Max (ms)
Association	3,087	1,458	17,608
Orbit promotion	6,369	4,448	90,937
Kalman update	1,659	278	43,770
Seeding	231	89	2,286
Total	**11,359**	**6,558**	**105,087**

Fink-FAT - Reconstruction performances on ZTF

Full 2025 dataset -> 297 nights processed

Orbit candidate generated:

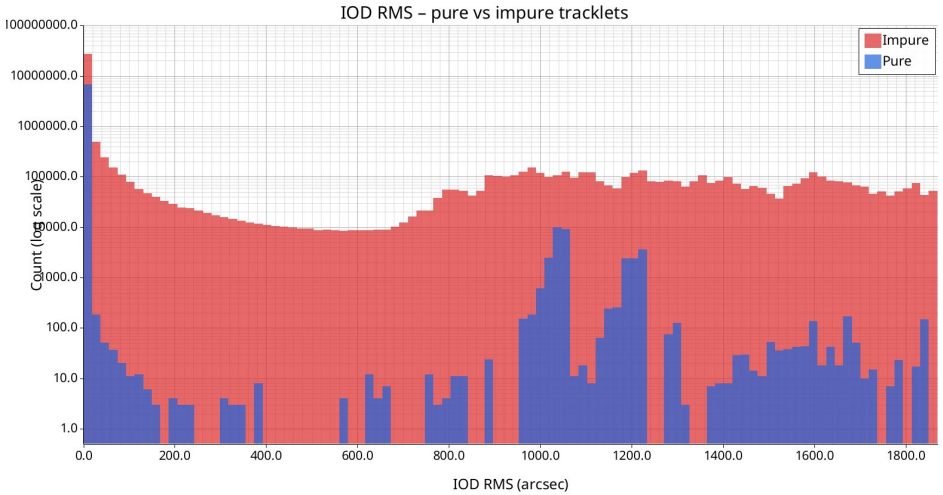
- 4_204_395 orbit
- Pure orbit :
  - 1_934_209 (46.00%)

Tracklet statistics final number of points

Stat	Value
Mean	4.3
Median	3.0
Std	4.8
Max	33.2

Fink-FAT - Reconstruction performances on ZTF

Full 2025 dataset -> 297 nights processed



Orbit candidate generated:

- 4_204_395 orbit
- Pure orbit :
  - 1_934_209 (46.00%)

Current issues :

- RMS/Chi^2 distribution doesn't allow to distinguish between pure and impure tracks
- Too short observation arc (just few days)
- Too few points
- Not yet ephemerides associations

Future Plans

Short terms

- Better understanding of the linking performances and orbit estimations
- Implementing the ephemerides associations
- Improve Outfit robustness (fit Gaia data at sub mas scale)
- Improve code quality for production

and beyond

- Submit our linking to MPC
- Put the prediction/associations steps within a science module (livestream)
  - requires better understanding of the prediction/associations time complexity
  - build benchmarks
- Produce hyperbolic candidates (interstellar objects)
- Ephemerides services for candidates orbits (Follow-up)