Application of Neural Ordinary Differential Equations to Meteoroid Atmospheric Entry Dynamics for the Retrieval of Physical Parameters

Matteo Magliano

University of Torino – Dept. of Mathematics "Giuseppe Peano"

Supervisor: Prof. M. Beccuti; Co-supervisors: D. Bonino, D. Barghini

Torino, 15 July 2026



UNIVERSITÀ DI TORINO

Thesis Objective

Objective

Investigate the use of deep learning, to model meteoroid atmospheric entry dynamics, and retrieve the underlying, non-observable physical parameters that govern it.

Ambition

Move toward a fully **data-driven** representation of the entry dynamics that could learn even beyond the approximations built into the equations that describe it.

Meteorites are fragments of a meteoroid that survive atmospheric ablation and reach the ground.

Why recover meteorites?

- Fresh-fallen samples reveal information on the **composition** and **thermal history** of asteroids in the early solar system;
- To increase and refine the meteorite classification scheme, allowing more detailed links between celestial bodies.

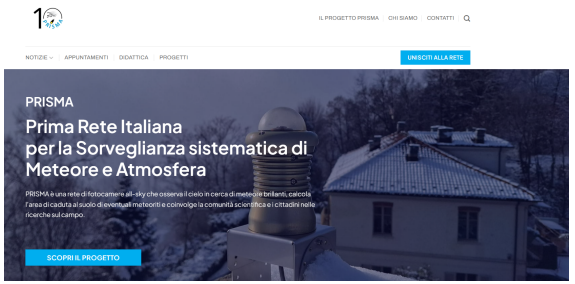
How?

$\Rightarrow$ Recovery requires an **accurate** and **precise approximation** of the *strewn field*.

The PRISMA Project

Overview & Objective

- **PRISMA**: *Prima Rete Italiana per la Sorveglianza sistematica di Meteore e Atmosfera*;
- Established in 2016, coordinated by **INAF** (Italian National Institute for Astrophysics);
- **Primary objective:** systematic meteoroids observation and the subsequent recovery of meteorite fragments;





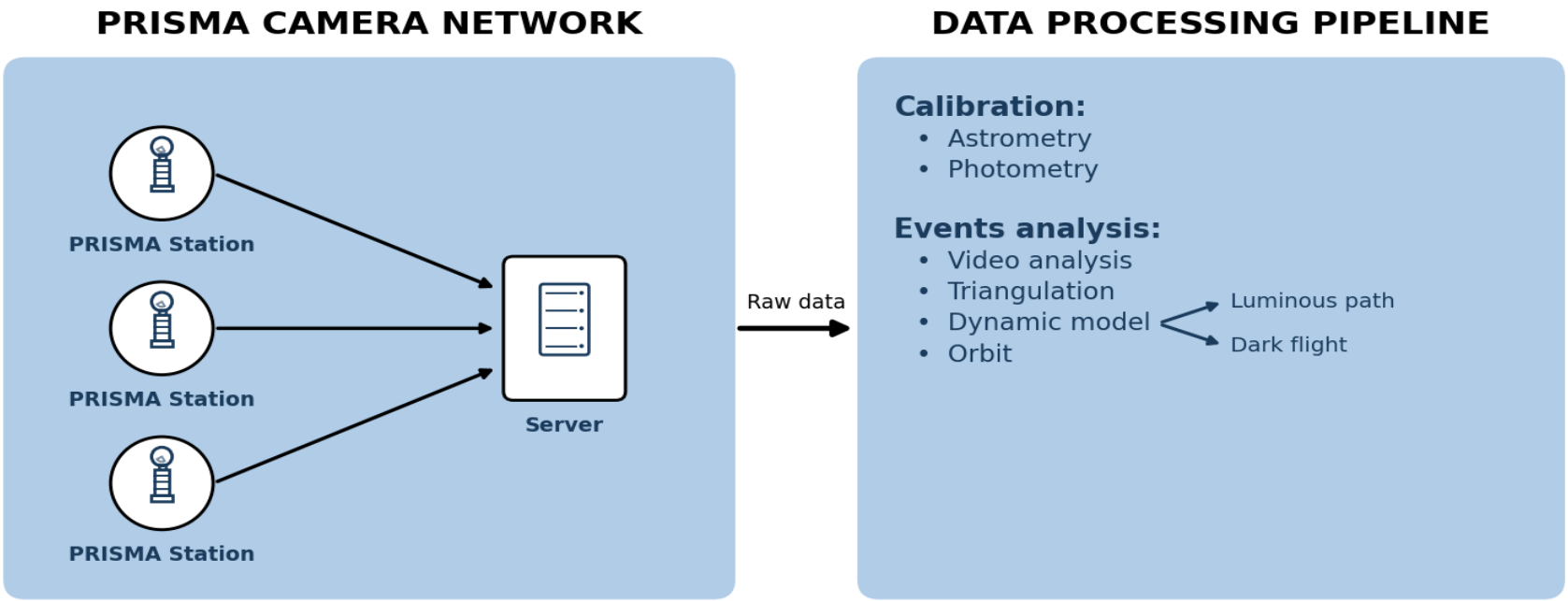
INAF
ISTITUTO NAZIONALE
DI ASTROFISICA

Prima Rete Italiana Sorveglianza Sistematica Meteore & Atmosfera
PRISMA

The PRISMA Project

From observation to data

- ≈90 cameras deployed across Italian territory;
- Standard detection frames: exposure of $1/30$ s;
- Dedicated 5-second exposure every 10 minutes for calibration purposes;



From Data to Trajectories

The traditional approach

Luminous path: Governing dynamic system of differential equations (height h, velocity v, magnitude $\mathcal{M}$)

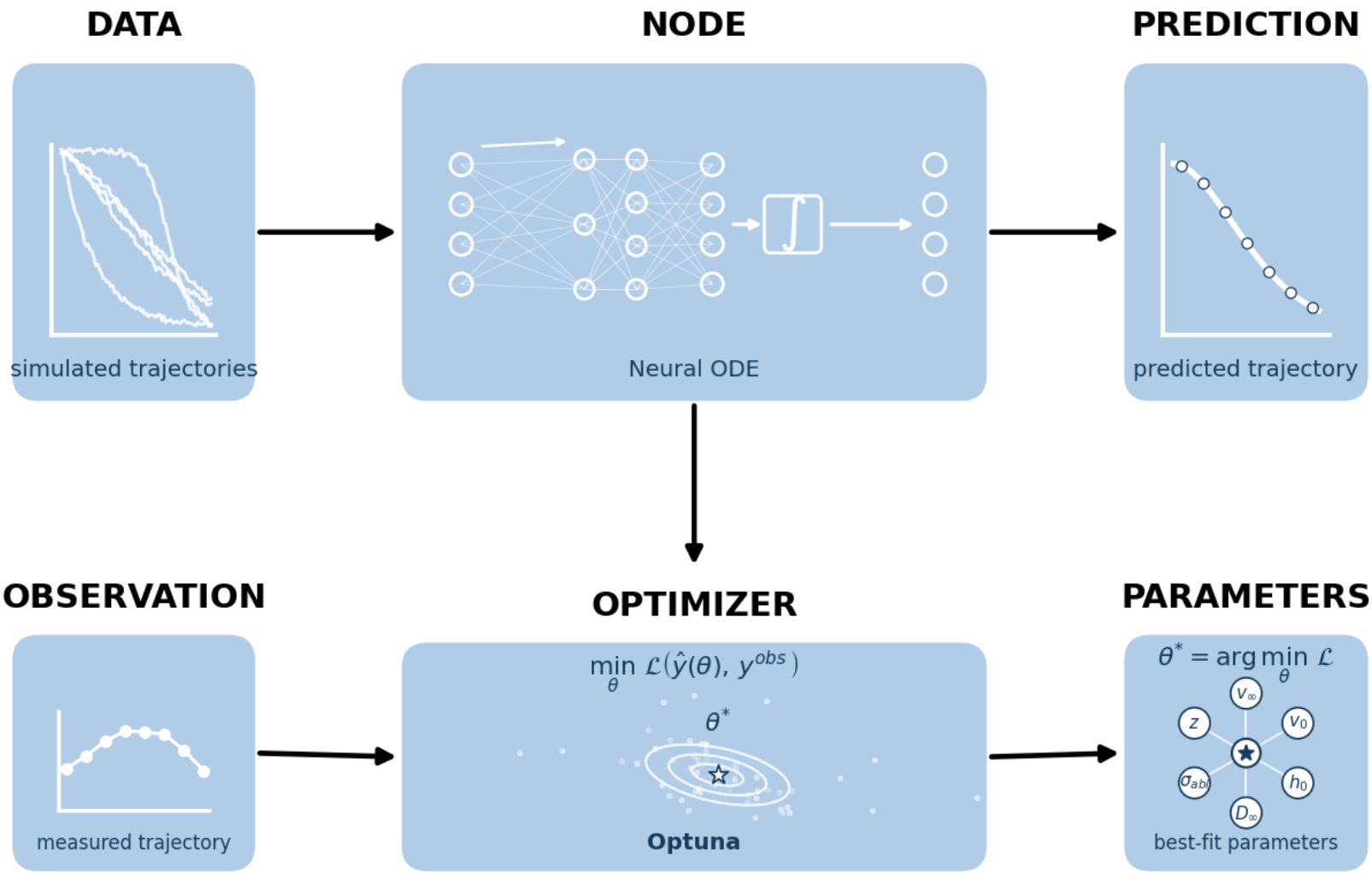
$$\begin{cases} \dfrac{dh}{dt} = -v \sin z \\ \dfrac{dv}{dt} = -\dfrac{\Gamma \rho_a v^2}{D} \\ D = D_\infty \exp\left\{ \dfrac{1}{2}\sigma(1-\mu)(v^2 - v_\infty^2) \right\} \\ \mathcal{M} = -2.5\left\{ \log_{10}\left[\left(\dfrac{\tau M_\infty \Gamma}{D_\infty}\right) \rho_a v^3 \left(1 + \dfrac{\sigma v^2}{2}\right) e^{\frac{\sigma\mu}{2}(v^2 - v_\infty^2)} \right] - 3.185 \right\} \end{cases}$$

Traditional solution pipeline

1. **Initial Analytical Estimation**: simplified version of the model provides guess values;
2. **Numerical Integration**: numerical solver integrates the system;
3. **Parameter Optimization**: Optimization algorithm minimizes the residuals against the observed data of (h, v).

The NODE Framework

Diagram Overview



Building the NODE

Synthetic Dataset: from scarce data to simulated data

Why a synthetic dataset?

- Atmospheric entries are **rare events**; PRISMA data are sparse, incomplete, and noisy;

$\Rightarrow$ Not enough real observations to train the NODE model directly

- A **synthetic dataset simulator**, fed with randomly sampled physical parameters, generates a large set of realistic synthetic trajectories;

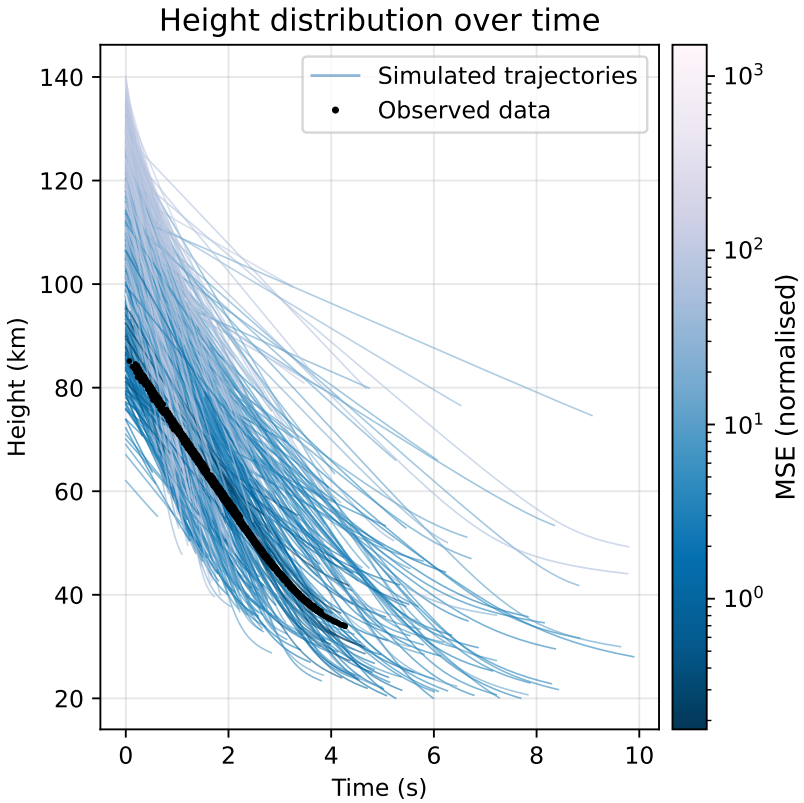
$\Rightarrow$ This same scarcity of real data is also what currently limits **fine-tuning** on real observations.

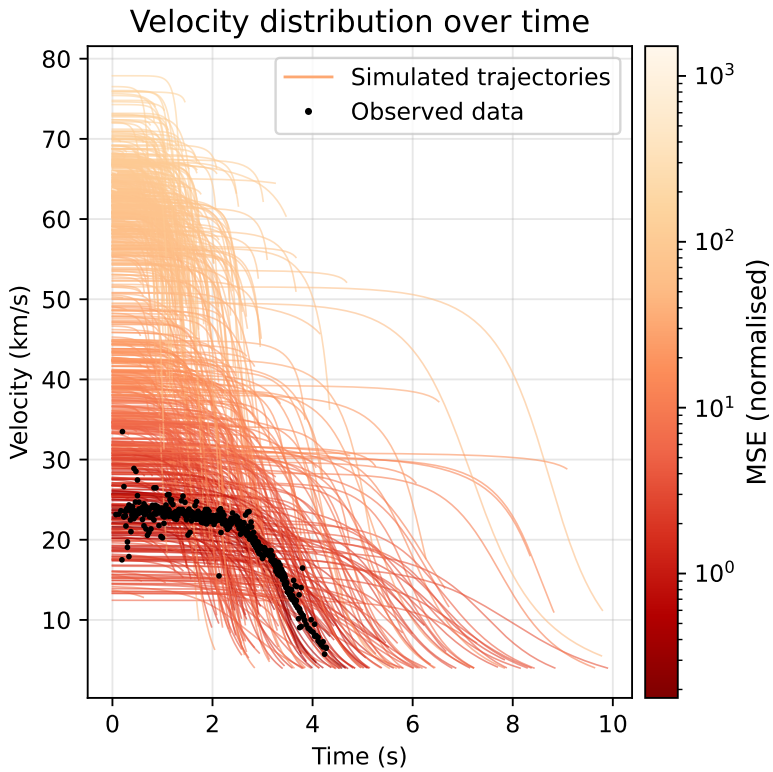
Result

- Only the **luminous segments** of the flight are retained
  $\Rightarrow$ clean, dense, consistent training data, representative of the full variability of atmospheric entry.

Building the NODE

Synthetic Dataset: Simulated trajectories

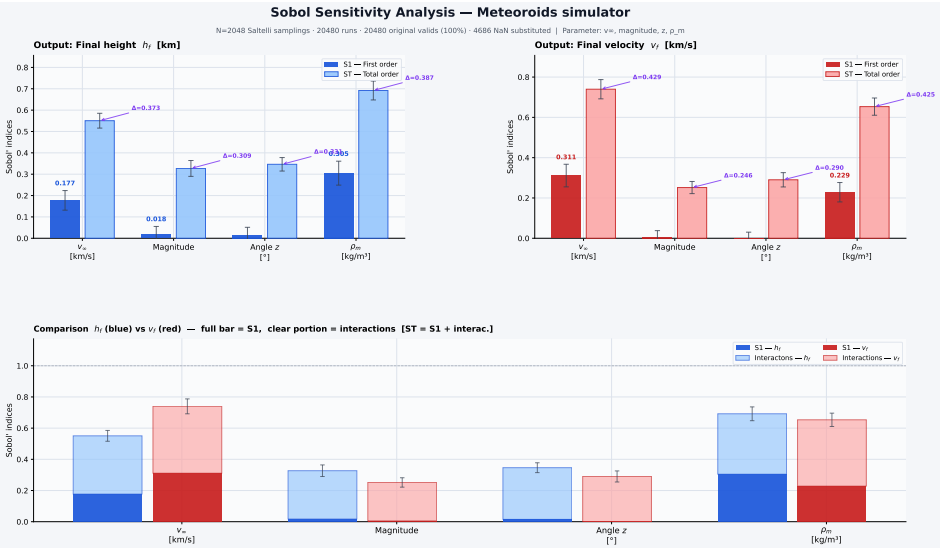


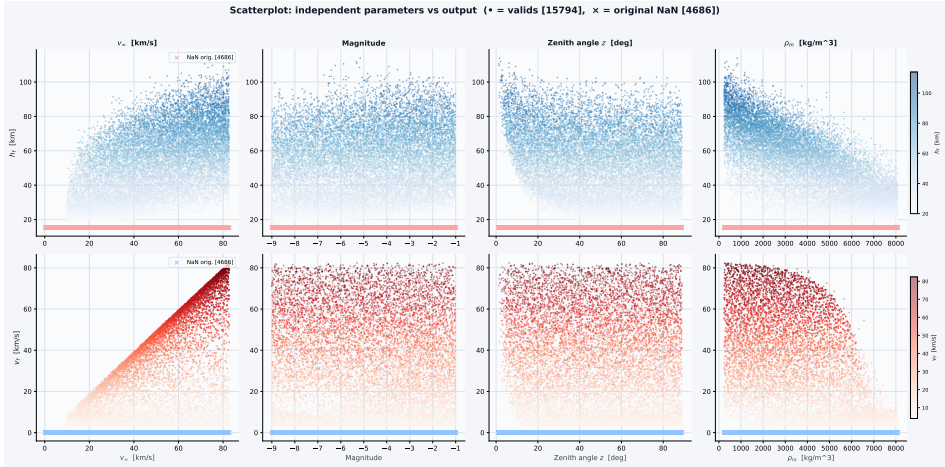


1,000 synthetic trajectories of height (left) and velocity (right) generated by the stochastic sampling pipeline colored by normalized MSE with respect to one observed bolide profile (black dots). Dark curves indicate trajectories that best reproduce the observed data; light curves indicate poor agreement.

Building the NODE

Sensitivity Analysis (Sobol' indices)[1]





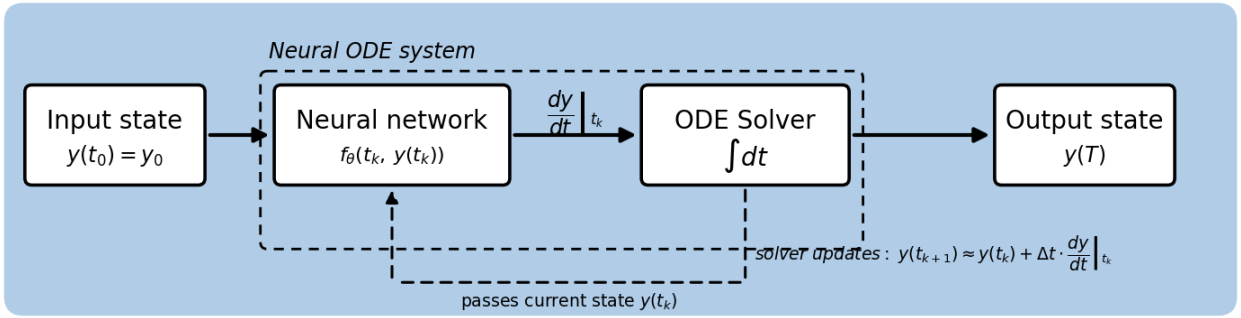
Key results

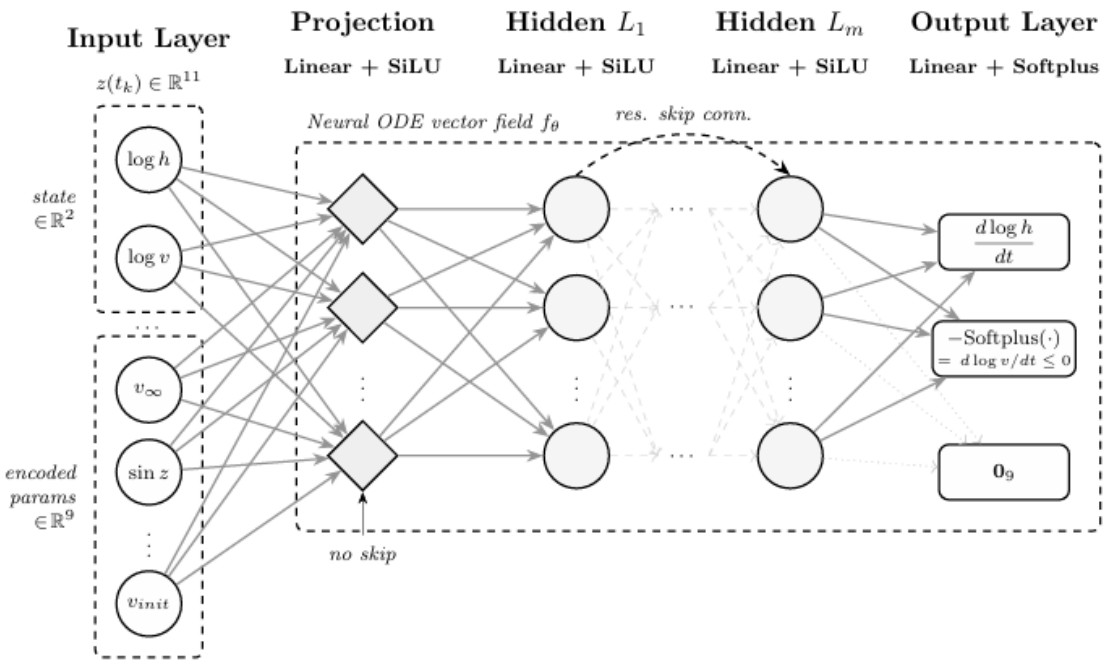
- **Final height** (h_f): dominated by meteoroid density ρ_m $(S_1 = 0.305)$, then v_∞ $(S_1 = 0.177)$
- **Final velocity** (v_f): dominated by v_∞, then ρ_m;
- Entry angle z and magnitude: negligible first-order effect;
- Strong non-linear parameter interactions.

[1] Sobol', I.M. (2001). *Global sensitivity indices for nonlinear mathematical models*. Math. Comput. Simul. 55(1-3), 271–280.

Building the NODE

Architecture [2]





- Residual network (ResNet) with `SiLU` activations;
- physics-informed **Softplus** constraint enforcing monotonic deceleration.

[2] R. T. Q. Chen et al., 2018. *Neural Ordinary Differential Equations*, Advances in Neural Information Processing Systems

Building the NODE

Training & physical constraints

Training strategy

- The NODE model is trained on **synthetic trajectories**: given the initial state and the physical parameters it learns to predict the evolution of h and v;
- Predicted trajectories are compared with the ground-truth ones, and the neural network weights are updated to minimize the discrepancy, guided by a **physics-informed loss**;

Loss terms

- **Reconstruction & Tail penalty:** log-space MSE + relative errors on h, v with extra focus on the final flight segment;
- **Kinematic constraint:** penalizes violations of $dh/dt = -v \sin z$
- **Dynamic consistency**: predicted dv/dt must match the physical deceleration law;

Building the NODE

Experimental Setup and Optuna[3]

Experimental Setup

- Trained on $N = 8,000$ synthetic trajectories (70% train / 15% val / 15% eval);

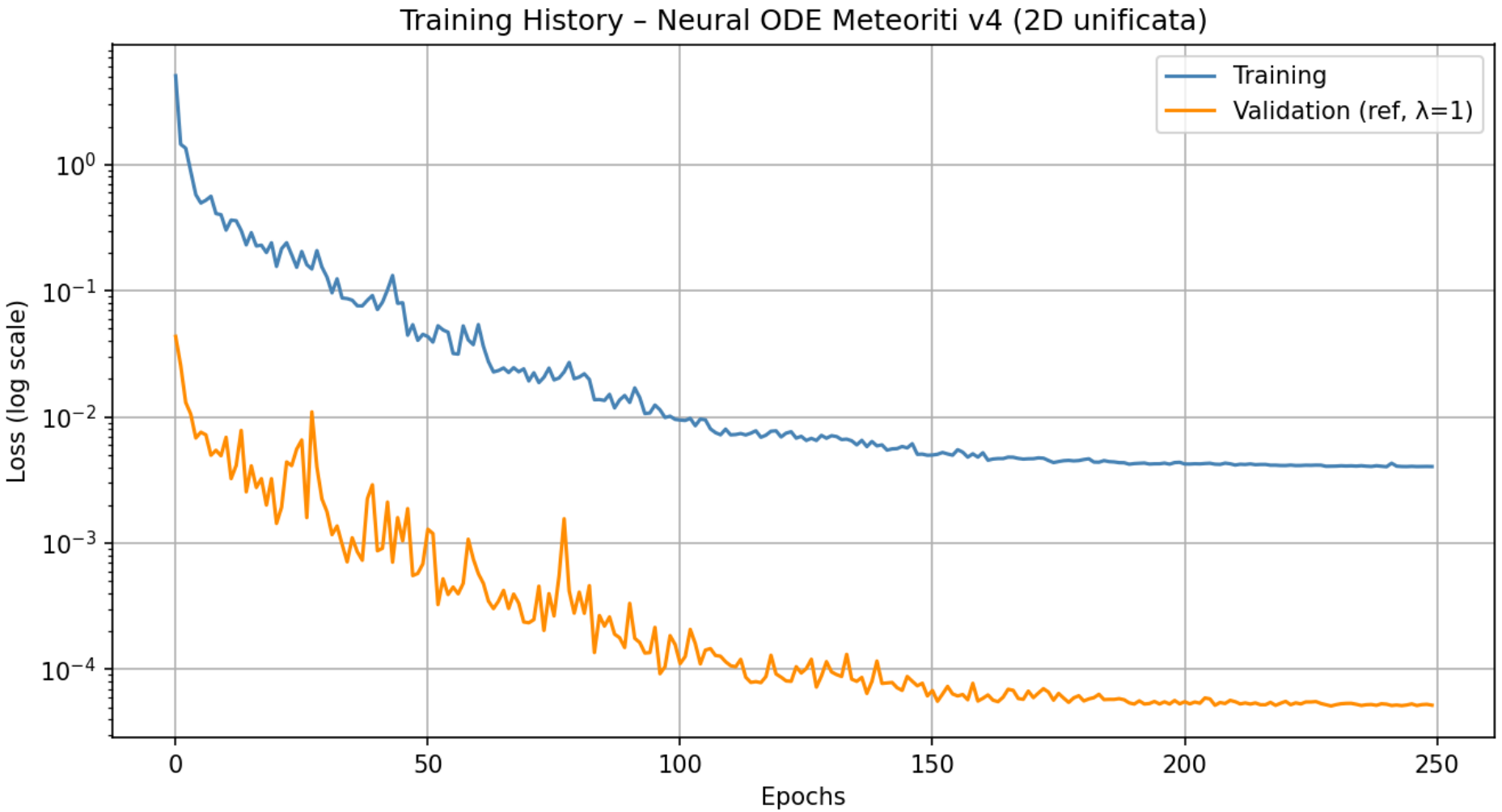
Optuna

- Both the **loss weights** and the **neural network's hyperparameters** are tuned via **Optuna**, an automate hyperparameter search: for each candidate configuration, the model is trained and evaluated on held-out data, through the so called validation loss;
- This value is then used to guide the search toward better configurations, before selecting the best one.

[3] Akiba et al., 2019. *Optuna: A Next-Generation Hyperparameter Optimization Framework*, International Conference on Knowledge Discovery & Data Mining (KDD).

Building the NODE

Loss Convergence

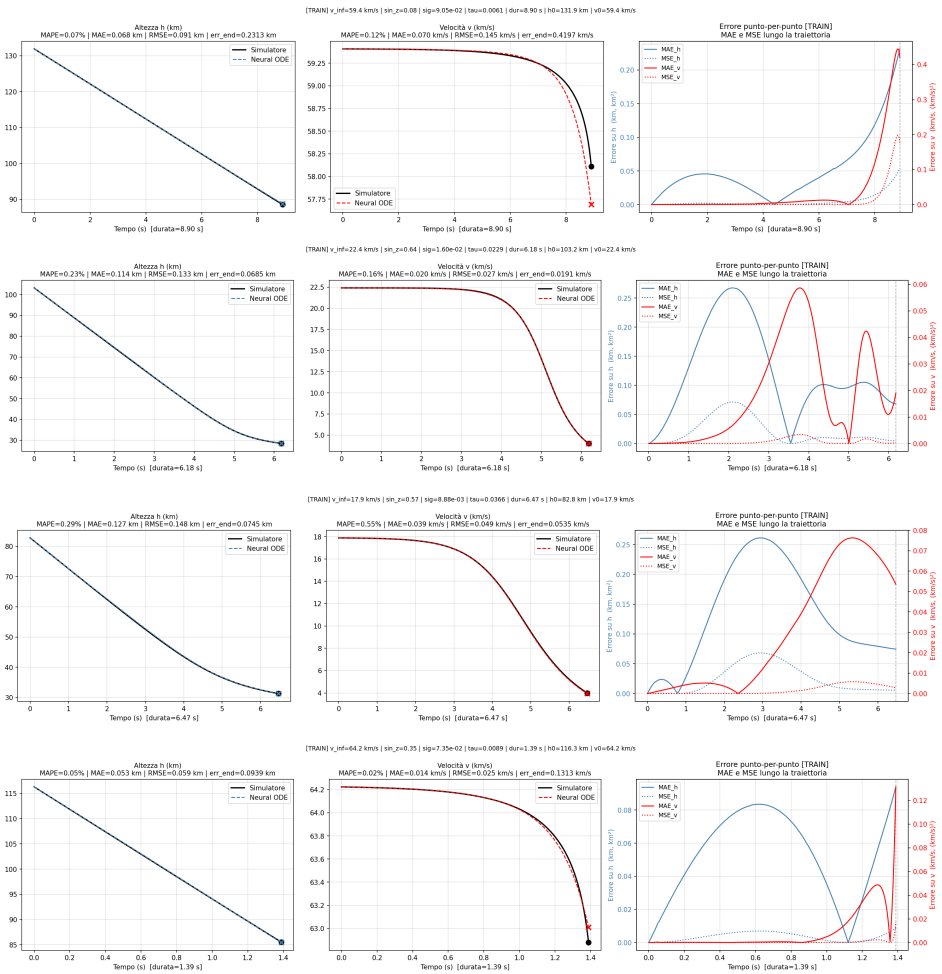


- Steady decrease, **no divergence** in both training and validation loss.

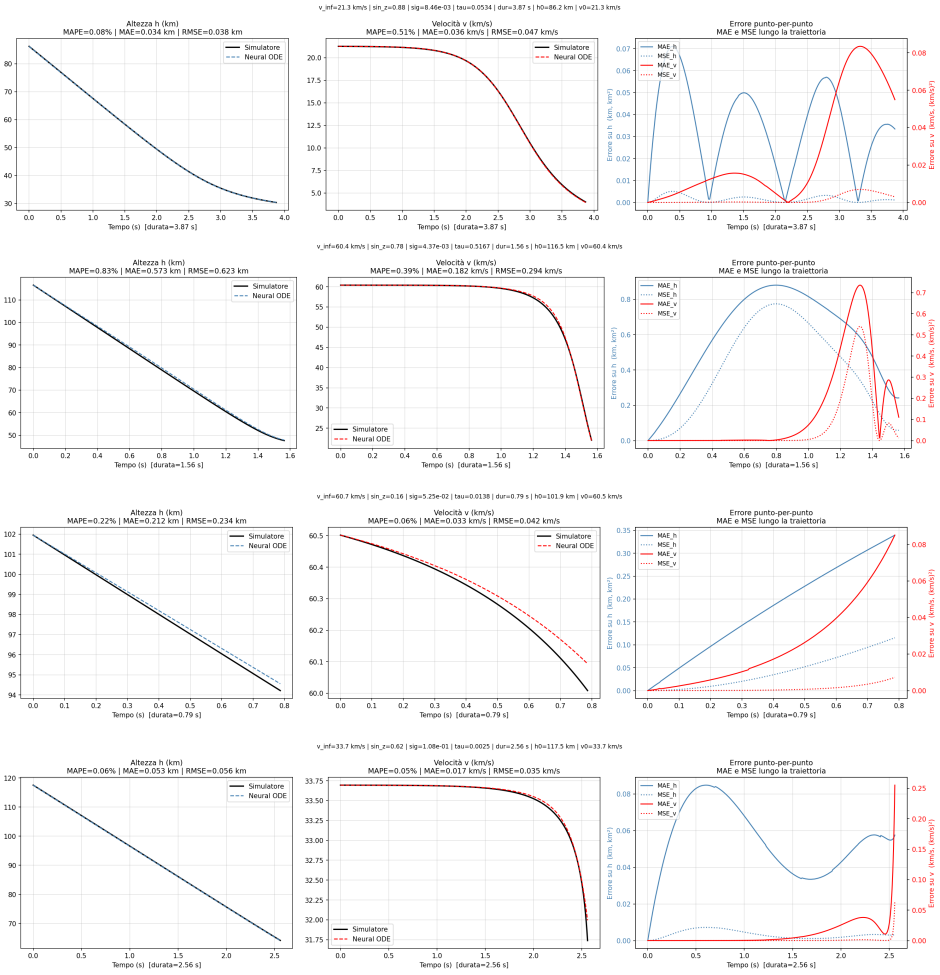
Building the NODE

Training & Evaluation performance

Training

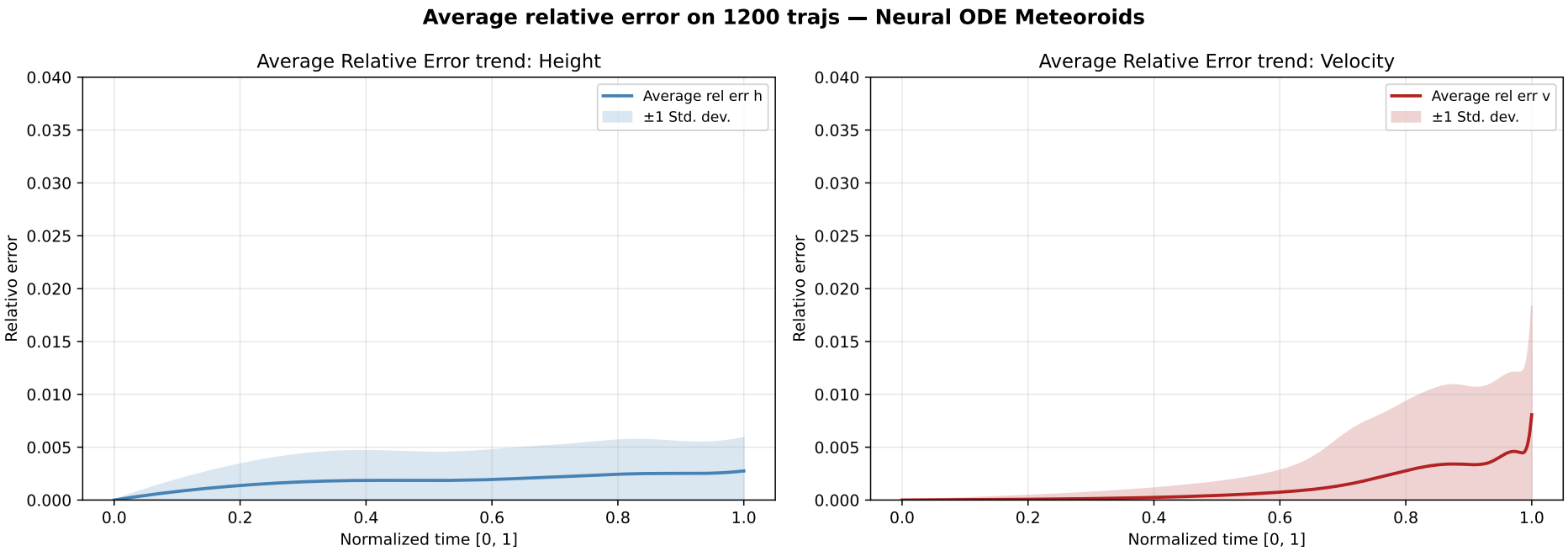


Evaluation



Building the NODE

Evaluation set: overall assessment



- Curve classification: **94.25% OK**, 5.35% warning, 0.50% bad;
  (OK: error on h and v below 2%, and error on the final segment of the trajectory below 25%)
- Residual errors concentrated in the **final deceleration phase**, hardest region to learn;

Inverse Problem

From trajectory to parameters

Task

- Given an **observed trajectory** $y^{obs} = \{(h_t, v_t)\}$ recover the latent parameters that generated it

$$\theta^* = \arg\min_{\theta \in \Omega} \mathcal{L}\left(\hat{y}(\cdot\,;\theta),\, y^{obs}\right)$$

Solution strategy

- Solved via **Optuna**, using the **CMA-ES** algorithm: candidate parameters are proposed, fed through the NODE, and scored against y^{obs}; the best candidates then guide the search toward progressively better ones;
- First validated on **synthetic observations** (known ground-truth parameters), before any application to real data;

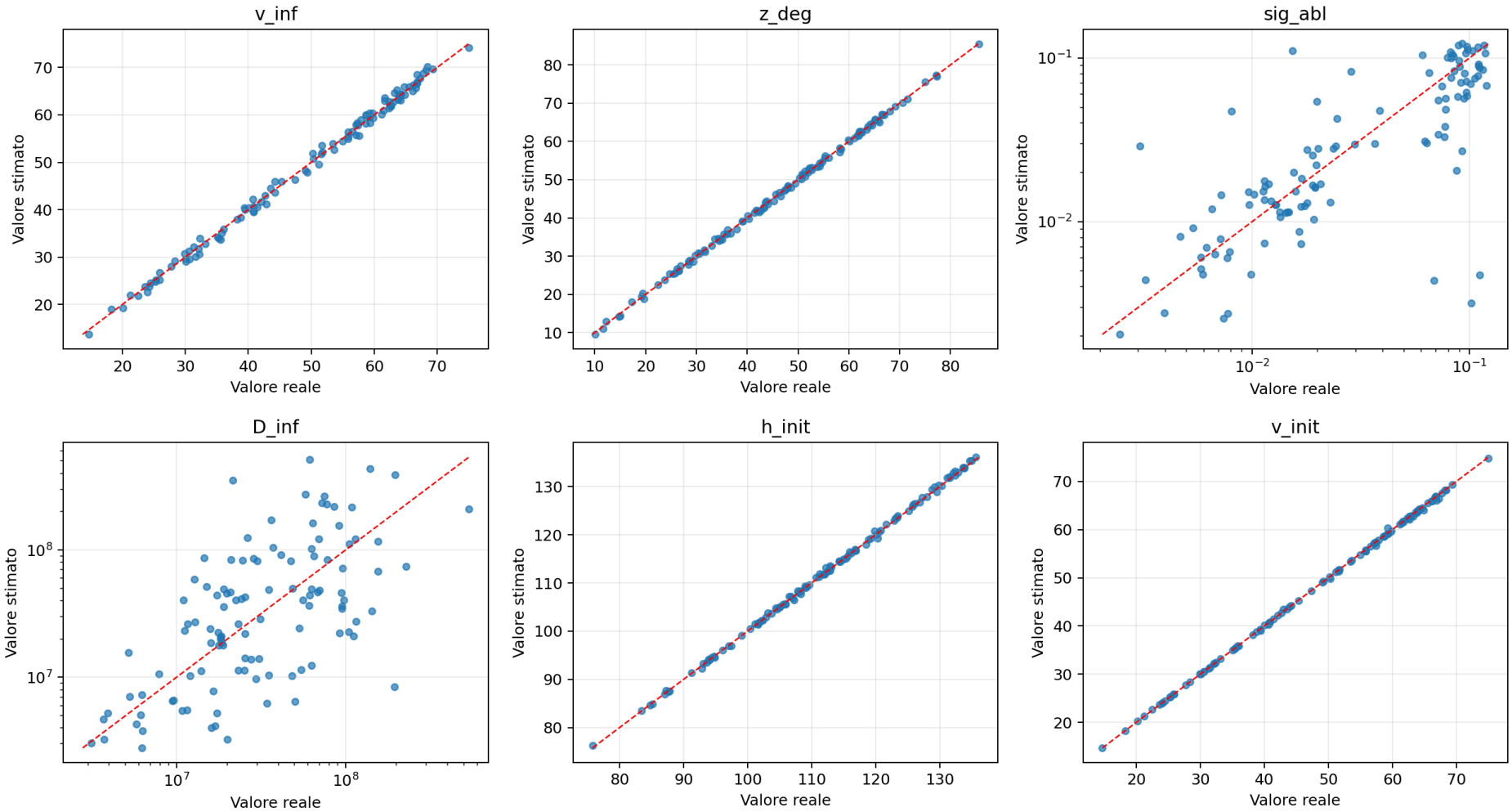
Summary statistics of the relative retrieval error (%)

Parameter	Median	Mean	p95	Min	Max
v_∞	1.65%	1.90%	4.52%	0.01%	6.88%
z	0.83%	1.18%	4.20%	0.01%	5.77%
σ	27.32%	51.15%	99.01%	0.25%	847.70%
D_∞	63.35%	107.82%	365.04%	0.25%	1539.90%
h_0	0.17%	0.24%	0.64%	0.01%	0.83%
v_0	0.14%	0.26%	1.11%	0.00%	1.66%

- **Kinematic parameters** (v_∞, z, h_0, v_0): good accuracy (median error $< 2\%$)
- **Ablation parameters** (σ, D_∞): large errors $\rightarrow$ structural **degeneracy**

Inverse Problem

Retrieval accuracy: qualitative results



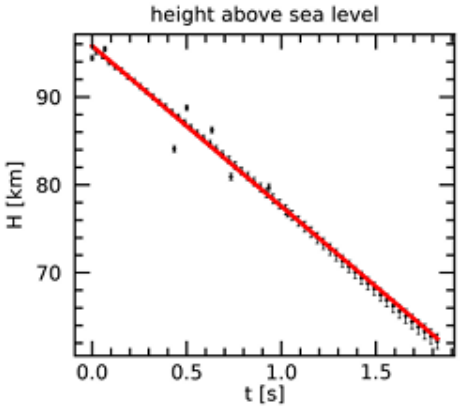
Application to Real Data

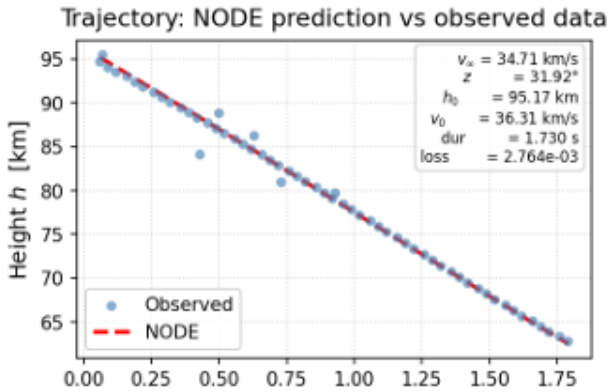
First test on real PRISMA events

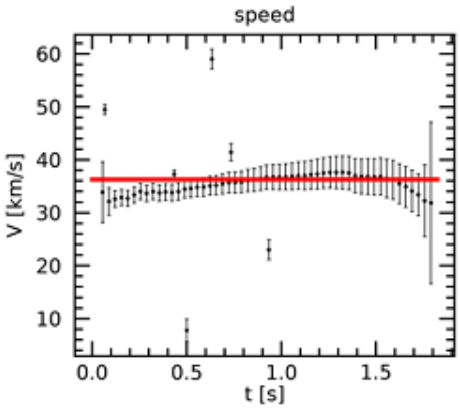
- Conducted on **16 real meteoroid events** recorded by PRISMA, using the NODE trained model (no fine-tuning on real data);
- No ground-truth parameters available $\Rightarrow$ reconstruction quality assessed via trajectory fidelity to observed points;
- Comparison with the traditional physics-based fit to assess the NODE-based inversion quality.

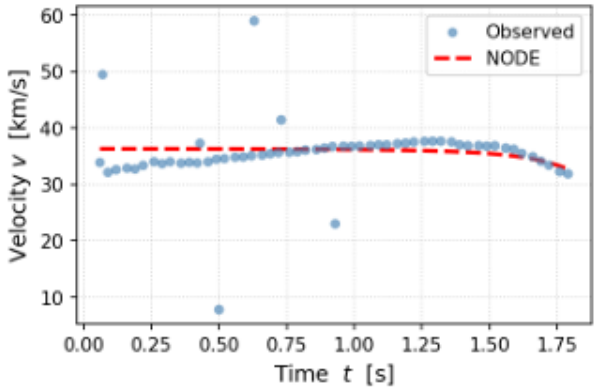
Application to Real Data

Traditional VS NODE reconstruction





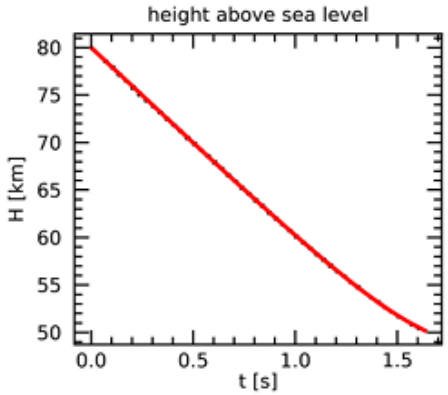


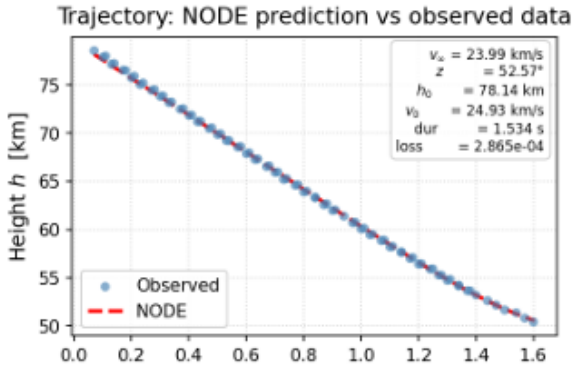


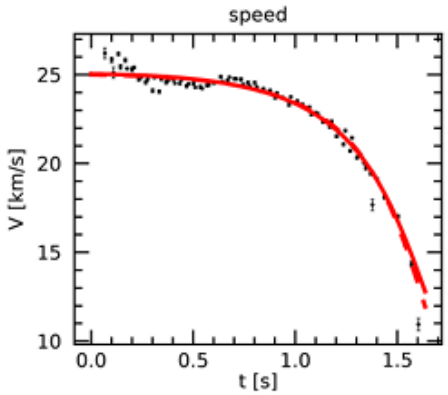
Physics-based model fit (left) compared with NODE-based prediction (right) for either height (top) and velocity (bottom) on a real observed event.

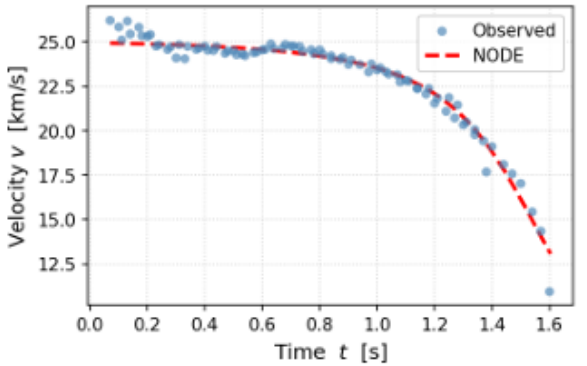
Application to Real Data

Traditional VS NODE reconstruction





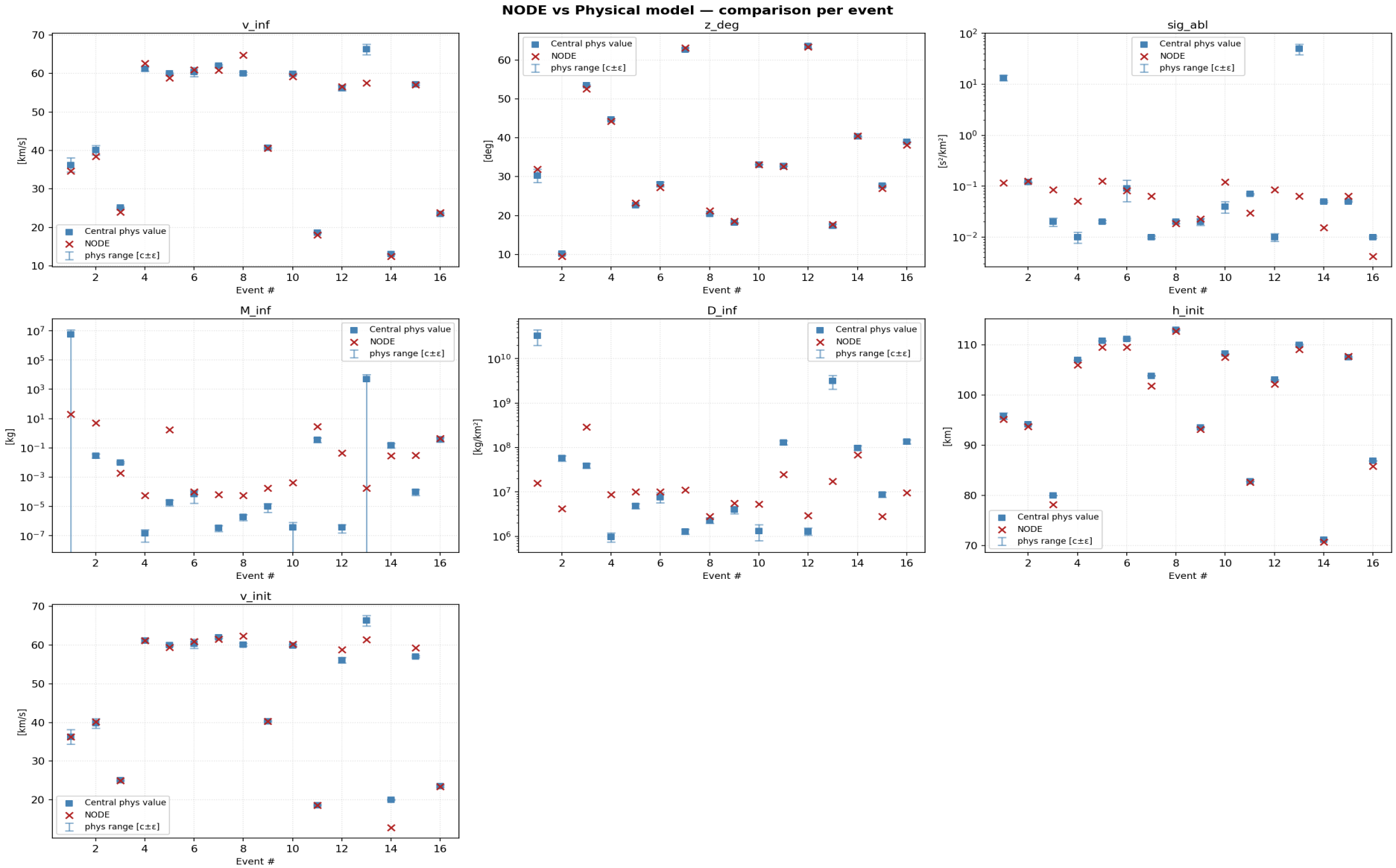




Physics-based model fit (left) compared with NODE-based prediction (right) for either height (top) and velocity (bottom) on a real observed event.

Application to Real Data

Inverse problem: retrieved parameters model comparison



Application to Real Data

Limitations

Structural limitation

- The parameter degeneracy of σ and D_∞ persists on real events.

Observational limitation

- No ground truth on real events $\Rightarrow$ results are a **consistency check**, not a validation;
- Real PRISMA data contains astrometric noise, timing uncertainties, and coverage gaps $\rightarrow$ effects on retrieval accuracy not yet systematically characterized

Conclusions and Future Research

Future Research

- Systematic **noise-injection study** on synthetic trajectories, to characterize retrieval robustness under realistic observational conditions;
- Extension toward a rigorously validated, real-data framework;
- **Fine-tuning** the NODE on real meteoroid recordings, to learn residual dynamics beyond current physical approximations;

Conclusions and Future Research

Conclusions

- On synthetic data, the NODE-based framework is able to learn the atmospheric entry dynamics with an overall good capacity, **retrieving non-observable physical parameters** in a satisfying way;
- Particularly accurate on **kinematic parameters** (v_∞, z); ablation parameters (σ, D_∞) remain affected by **structural degeneracy**;
- First test on **real PRISMA events** shows results mostly consistent with the traditional pipeline.

Takeaway

The combination of classical physical models and modern deep learning architectures represents a promising direction for the study of meteor events.

Thank you

for your attention!

Appendix

How the training was refined

- Diagnosis: the main bottleneck was not the average error, but the model's ability to reproduce the **final, fastest part** of the velocity profile;
- **Optuna/architecture alignment**: loss weights and learning rate chosen for a small network may not be optimal for a more expressive network;
- **Non-uniform temporal grid**: quadratic warping places $\sim 4\times$ more time steps in the tail than in the head, giving denser supervision exactly where dynamics are hardest $\rightarrow$ single largest improvement in curve quality.

Appendix

CMA-ES: optimization mechanism

- **Derivative-free, population-based** global optimizer well suited to high-dimensional, purely continuous search spaces like ours (9 parameters, no categorical variables);
- At each iteration, CMA-ES maintains a multivariate Gaussian $\mathcal{N}(\mathbf{m}, \gamma^2\mathbf{C})$ over the parameter space:
  - $\mathbf{m}$: current best estimate (mean)
  - $\mathbf{C}$: covariance matrix – learns the **shape** and correlations of the search space
  - γ: global step size
- A population of candidates is sampled from this Gaussian, evaluated, and the best ones update $\mathbf{m}, \gamma, \mathbf{C}$ for the next iteration, then the search progressively adapts its shape to the loss landscape;
- **IPOP restart**: if the search stagnates, it restarts with a doubled population size, reducing the risk of getting stuck in local minima;

Appendix

Sobol' indices: theoretical background

The idea

- A **variance-based** method: decomposes the total variance of the model output, $Var(Y)$, into contributions from each input variable and their interactions.

First-order index S_i

- Measures the **direct** contribution of input X_i alone to the output variance, given $Y = f(X_1, \ldots, X_p)$;
- Computed by comparing variance across base and cross-sampling matrices (Saltelli sampling scheme).

Total-order index S_{T_i}

- Captures the **full effect** of X_i, including all its interactions with every other parameter;
- The gap $\Delta = S_{T_i} - S_i$ quantifies how much of the parameter's influence comes from **coupling** with others, rather than acting alone.

Appendix

An alternative to CMA-ES: SNPE-NSF

The idea

- Instead of returning single point estimate, learn the full **posterior** $p(\theta \mid x)$ of the physical parameters given an observed trajectory.

Method

- Posterior trained with **Sequential Neural Posterior Estimation (SNPE)**, using a **Neural Spline Flow (NSF)** as the conditional density estimator
- Uniform prior over the normalized parameter space;
- Hyperparameters (e.g. network size, spline transforms, learning rate) tuned via **Optuna**.

Key advantage

- Returns a full **uncertainty interval** for each parameter, not just a single value, directly exposing which parameters are well identified and which remain degenerate.