Inferring Launch and Environmental Parameters from Simulated Trajectories

Machine Learning for Quantum Scientists (physics7516)

Leon Galbas

Rheinische Friedrich-Wilhelms-Universität Bonn

31/07/2026

Outline

Outline

Can launch parameters be inferred from object trajectories?

$$\text{trajectory } \left\{\mathbf{r}(t_1), \ldots, \mathbf{r}(t_T)\right\} \longrightarrow \text{ML model} \longrightarrow \text{launch parameters } \widehat{\lambda}$$

$$\lambda = \Big(\underbrace{v_x, v_y, v_z,}_{\substack{\text{initial}\\ \text{velocity}}} \underbrace{u_x, u_y, u_z,}_{\substack{\text{wind}\\ \text{velocity}}} \underbrace{\omega_x, \omega_y, \omega_z,}_{\substack{\text{rotation}\\ \text{axis}}} \underbrace{\beta_D, \beta_M}_{\substack{\text{drag and}\\ \text{Magnus}\\ \text{parameters}}} \Big)$$

Simulated trajectories

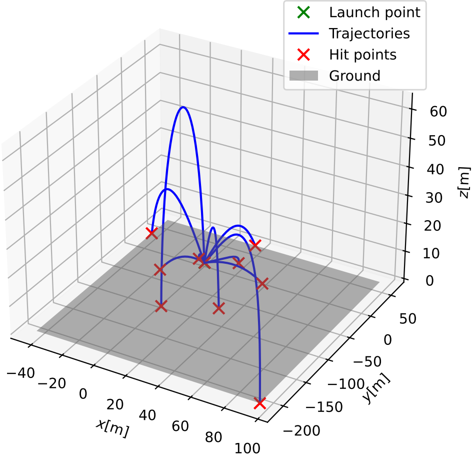


Figure: Ten exemplary simulated trajectories.

Equations of motion

$$\frac{\mathrm{d}\mathbf{r}}{\mathrm{d}t} = \mathbf{v}, \qquad \frac{\mathrm{d}\mathbf{v}}{\mathrm{d}t} = \underbrace{\mathbf{g}}_{\text{gravity}} - \underbrace{\beta_D |\mathbf{q}| \mathbf{q}}_{\text{drag}} + \underbrace{\beta_M \left(\widehat{\omega} \times \mathbf{q}\right)}_{\text{Magnus}}, \qquad \mathbf{q} = \mathbf{v} - \mathbf{u}.$$

- $\mathbf{q}$: velocity relative to the wind
- drag: deceleration opposite the relative motion
- Magnus term: spin-induced curvature

Simulation parameters

Quantity	Symbol	Sampled range
Initial speed	v_0	$0 - 75\,\mathrm{m/s}$
Wind speed	u	$0 - 25\,\mathrm{m/s}$
Drag strength	β_D	$0{,}003 - 0{,}04/\mathrm{m}$
Magnus strength	β_M	$0 - 0{,}3/\mathrm{s}$

200 000 trajectories

10 000 raw points $\longrightarrow$ 100 time points

Integrate from launch until the projectile reaches $z = 0$.

Outline

Two model approaches

	Engineered features + MLP	Time series + GRU
Input	24 trajectory summaries	100 ordered observations
Examples	time, range, apex, derivatives	t, x, y, z, optionally $\mathbf{v}, \mathbf{a}$
Strength	compact and interpretable	retains temporal structure

160,000 training + 40,000 test

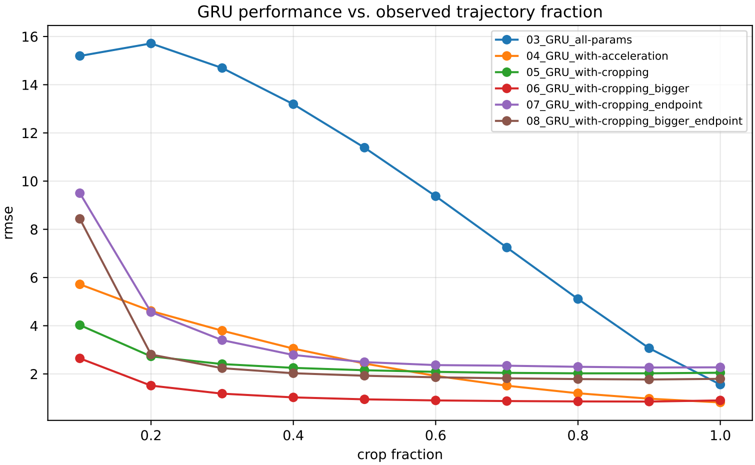
Outline

Model performance on full trajectories

Target RMSE	MLP	GRU, position	GRU, derivatives
v_x [$\mathrm{m\,s^{-1}}$]	1.016	1.793	1.027
u_x [$\mathrm{m\,s^{-1}}$]	3.857	2.667	1.359
ω_x	0.413	0.247	0.129
β_D [$\mathrm{m^{-1}}$]	0.00445	0.00300	0.00127
β_M [$\mathrm{s^{-1}}$]	0.0664	0.0378	0.0190

Table: Comparison of the prediction performance for selected test set parameters and model configurations.

Model performance on partial trajectories



Model performance on partial trajectories

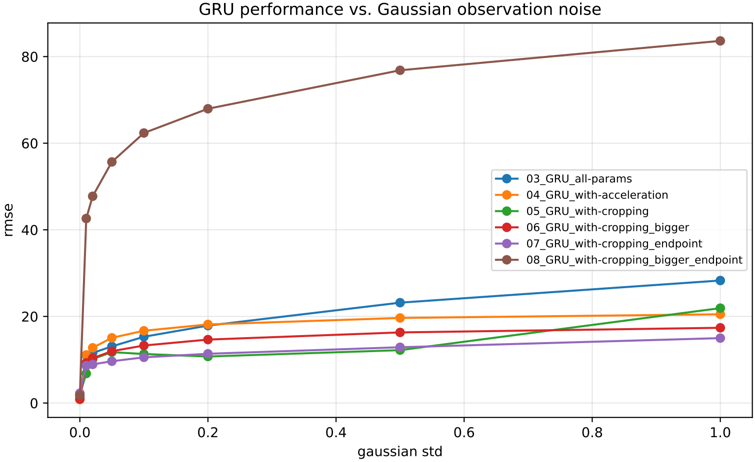
Observed fraction	Target	RMSE
10%	v_x, full-flight training	$6.62\,\mathrm{m\,s^{-1}}$
10%	v_x, crop training	$2.16\,\mathrm{m\,s^{-1}}$
50%	u_x, crop training	$1.64\,\mathrm{m\,s^{-1}}$
50%	β_M, crop training	$0.0223\,\mathrm{s^{-1}}$

Table: RMSE for selected parameters for Model 04 (derivative-augmented GRU trained on complete trajectories) and Model 06 (larger derivative-augmented GRU trained with random cropping).

At 10% observation, crop training reduces the v_x error by about two thirds.

The largest cropping gain occurs early in the flight

Model performance on noisy trajectories



Model performance on noisy trajectoriese

Target RMSE	Clean	$\sigma = 0.01$m	Increase
v_x [m s^{-1}]	1.03	21.0	20.4×
u_x [m s^{-1}]	1.36	9.50	7.0×
β_D [m^{-1}]	0.00127	0.0120	9.4×

Table: Noise robustness of Model 04, the derivative-augmented GRU, evaluated on clean positions and with $\sigma = 0.01$m positional noise.

noisy positions $\longrightarrow$ numerical derivatives $\longrightarrow$ amplified noise

Outline

Conclusion

- Engineered features are strongest when they directly encode the target.
- GRU derivative inputs best recover the full physics from clean data.
- Random-prefix training enables prediction from incomplete flights.
- Real measurements require noise-aware training and robust derivatives.

The trajectory contains enough information to recover much of the hidden physics, but reliability depends on the chosen representation.

Source Code

The full source code for the project is available on GitHub under `https://github.com/leon-galbas/physics7516-ml-final-project`.