Informative AI Systems

Anweshan 2025

Pool Peshwas

Report

Abstract

This model presents a data-driven approach to learning simplified governing equations for tropical cyclone evolution by combining Informative Neural Ensemble Kalman Learning (INEKL) with physics-based tropical cyclone dynamics. Using sparse, noisy observational data as well as synthetically generated data, our framework aims to learn functional relationships governing key storm parameters such as maximum azimuthal wind speed, inner-core moisture, steering forces, sea-surface temperature etc. The learned model incorporates uncertainty quantification using ensemble variance and demonstrates enhanced information gain as compared to traditional gradient-based training. This hybrid methodology offers a more interpretable, physics-informed climate prediction models. GitHub Repository Link to access the code notebook, datasets and all the output files generated according to the PS requirements: `https://github.com/archislegend100/Informative_AI_Systems_Peshwas`

1 Introduction

Tropical cyclones (TCs) represent complex geophysical phenomena with critical impacts on societies worldwide. Traditional high-resolution numerical models for cyclone simulation are computationally expensive and sensitive to initial condition uncertainties [2]. Advances in data-driven learning methods, notably Informative Neural Ensemble Kalman Learning (INEKL) [1], provide an innovative avenue to learn the underlying differential equations directly from data while quantifying uncertainty. This report proposes applying INEKL to learn simplified differential equations describing cyclone intensity and internal dynamics, guided by recent open-source models of cyclone physics [2].

2 Problem Statement

We seek to develop a system that learns the evolution of key cyclone parameters like v (maximum azimuthal wind speed), m (inner-core moisture), and other relevant variables from data, replacing conventional gradient-based parameter tuning with an ensemble Kalman-based neural learning scheme. Our objectives are to:

- Implement Ensemble Kalman-based learning to estimate storm parameter evolution in real time.
- Derive simplified differential equations describing cyclone intensification, moisture dynamics, and steering.
- Quantify uncertainty in learned predictions via ensemble statistics.
- Demonstrate enhanced information gain relative to baseline learning methods.
- (Optional) Incorporate physical constraints for hybrid modeling.

3 Literature Review

3.1 Physics-Based Tropical Cyclone Evolution

Following Lin et al. [2], tropical cyclone dynamics can be described using simplified coupled differential equations:

$$\frac{dv}{dt} = f(v, m, V_p, S, \Theta, \ldots), \tag{1}$$

$$\frac{dm}{dt} = g(v, m, \text{humidity}, \text{entropy}, \ldots). \tag{2}$$

- **Maximum Azimuthal Wind Speed** (v): The primary measure of storm intensity, controls beta drift strength and sets the stage for Eyewall Replacement Cycle resulting in larger storms.
- **Inner Core Moisture** (m): Represents fuel for heat engine. Low value of inner core moisture denotes a poorly defined centre and such storms have an unpredictable nature.

- **Potential Intensity**

$$V_p^2 = S_w^2 \frac{C_k}{C_d} \frac{T_s}{T_o} \left(\text{CAPE}^* - \text{CAPE}\right). \tag{3}$$

- **Vertical wind shear** (S): It is the change in wind speed with height. It is typically measured as the difference in the horizontal wind vector between two levels, most commonly 250 and 850 hPa.
- **Environmental temperature** (Θ): Inhibits or permits deep convection.

The functions f and g capture intensification and moisture dynamics, respectively. These equations incorporate the interaction of physical processes such as surface fluxes, angular momentum budgets, and thermodynamic feedbacks.

3.2 Informative Neural Ensemble Kalman Learning (INEKL)

Informative Neural Ensemble Kalman Learning (INEKL) [1] offers a principled, data-driven framework that directly addresses the challenge of learning tropical cyclone evolution equations from noisy, sparse observational data—as motivated by the problem statement.

INEKL is a Bayesian ensemble-based neural network training algorithm specifically formulated to learn parametric or nonparametric differential equations governing complex dynamical systems. Unlike traditional gradient descent optimization methods, which often require heavy hyperparameter tuning and struggle with non-convex loss landscapes exacerbated by noise, INEKL reformulates parameter learning as an adaptive Bayesian inference problem. This is particularly suitable for cyclone modeling where observational uncertainties and data sparsity are the norm rather than the exception.

At the core of INEKL is the maintenance of an ensemble of neural network parameter configurations, $\{\theta^{(i)}\}_{i=1}^N$, each representing a plausible approximation of the underlying model generating the tropical cyclone dynamics. During sequential iterations k, each ensemble member generates a prediction $\hat{y}_k^{(i)}$ for the system state evolution, which is then compared with the actual observed data y_k.

The ensemble Kalman update step leverages the covariance structure within this ensemble to compute a Kalman gain matrix K_k, enabling each member's parameter vector to be updated as:

$$\theta_{k+1}^{(i)} = \theta_k^{(i)} + K_k \left(y_k - \hat{y}_k^{(i)}\right).$$

This step effectively fuses data with model predictions to adjust the neural network parameters toward configurations that better explain the observed cyclone behavior, while inherently quantifying the uncertainty via ensemble spread.

A core innovation of INEKL is its use of **data assimilation** during learning. It employs an **Ensemble Kalman Filter (EnKF)** to update model parameters by recursively assimilating observational data, formalized as:

$$A_{i+1} = A_i \cdot \frac{1}{S_i} \sum_{s \in B_i} M_{\alpha,i,s},$$

where A_i is the parameter ensemble, B_i the mini-batch, and $M_{\alpha,i,s}$ the Kalman update. This provides inherent **uncertainty quantification** via the ensemble spread and enables **sequential learning**, making it ideal for dynamical systems like tropical cyclones where integrating sparse, real-world data is crucial for model fidelity. This data-assimilative update systematically reduces prediction error by optimally correcting the model state with each new observation.

In the context of learning cyclone evolution equations, this approach offers several direct advantages:

1. **Uncertainty Quantification:** Through an ensemble perspective, the method quantifies epistemic uncertainty in the learned dynamical equations and predictions, critical for high-impact meteorological hazards where forecasting confidence must be communicated.
2. **Robustness to Noisy and Sparse Data:** By incorporating Bayesian data assimilation principles, INEKL remains stable and performant even when cyclone observational data are noisy or irregularly spaced, overcoming limitations of classical gradient-based methods.
3. **Information-Maximizing Updates:** The Kalman gain matrix updates parameters to maximize the expected information gain about the cyclone system state, ensuring efficient learning from each new data point, improving convergence speed.
4. **Flexibility in Model Form:** The neural network parameterizes unknown or partially understood nonlinear functions in the cyclone evolution equations, while INEKL adaptively tunes its parameters, allowing discovery of physically interpretable and emergent dynamics unknown a priori.
5. **Suitability for Real-Time Assimilation:** INEKL's recursive update structure mirrors operational data assimilation workflows, making it well suited for real-time cyclone monitoring and forecasting environments.

3.3 Learning models from indirect observations

Beyond state estimation, EnKF has been used to *learn* closure models from indirect data. Zhang *et al.* developed an iterative EnKF with adaptive step size to train neural turbulence models embedded in RANS solvers, demonstrating robustness and efficiency compared with adjoint-based optimization [3]. This line of work supports our use of EnKF/INEKL to learn coefficients of physically structured ODEs from noisy, sparse, and partially indirect TC data.

3.4 Gap addressed in this work

Bridging physics-based intensity ODEs (FAST) with ensemble Bayesian learning (INEKL), we learn storm-/regime-adaptive coefficients for the same interpretable terms, quantify uncertainty via ensemble spread, and track information gain during training. Using identical initial conditions and environmental drivers, we fairly compare FAST and learned ODE simulations and report PS-required metrics.

4 Model Development and Learning Strategy

4.1 Dataset Preparation

We used the ERA5 [7] dataset to collect atmospheric data (monthly and daily averaged) for the various parameters used by the FAST model, like surface temperature, temperature profile at different levels, humidity. This data is collected for all the latitudes and longitudes at a resolution of 0.25 degree x 0.25 degree for land, and 0.5 x 0.5 for ocean.

For ocean data we used the ORAS5 [8] dataset to collect the atmospheric boundary layer depth (h_m).

For storm-wise data, we used the IBTrACSv04r01 [6] dataset to collect storm paths (latitude and longitude), and wind speed.

After collocating data from all three sources into one csv file, we prepared the derived variables as described in the FAST model.

$$\frac{dv}{dt} = a_f\Big(\underbrace{\alpha\beta V_p^2 m^3}_{\text{thermo drive}} - \underbrace{v^2}_{\text{spin-down}} + \underbrace{\gamma m^3 v^2}_{\text{moist feedback}} \Big), \tag{4}$$

$$\frac{dm}{dt} = a_f\Big(\underbrace{(1-m)v}_{\text{ventilation refill}} - \underbrace{\chi S m}_{\text{shear export}} \Big), \tag{5}$$

$$\beta = 1 - \epsilon - \kappa, \tag{6}$$

$$\gamma = \epsilon + \alpha\kappa, \tag{7}$$

$$\epsilon = \frac{T_s - T_0}{T_s}, \tag{8}$$

$$\kappa = \frac{\epsilon\, C_k\, L_v\, q_s^*}{2\, C_d\, R_d\, T_s}, \tag{9}$$

$$\alpha = 1 - 0.87\, e^{-z}, \tag{10}$$

$$z = 0.01\, \Gamma^{-0.4}\, h_n\, u_T\, \frac{\nu_p}{\nu}, \tag{11}$$

$$\chi_{\text{grid}} = \frac{s^* - s_m}{s_0^* - s^*}. \tag{12}$$

$$m_{\text{init}} = \frac{L}{1 + \exp[-k(\mathcal{H} - \mathcal{H}_0)]} + m_0. \tag{13}$$

$$V_p^2 = S_w^2 \frac{C_k}{C_d} \frac{T_s}{T_o} \left(\text{CAPE}^* - \text{CAPE}\right). \tag{14}$$

$$\chi = \exp(\log \chi_{\text{grid}} + \chi_\sigma) + \chi_a. \tag{15}$$

Table 1: DataFrame Summary Information

#	Column	Non-Null Count	Dtype
0	Unnamed: 0	34270	int64
1	storm	34270	object
2	sid	34270	object
3	basin	34270	object
4	time	34270	object
5	lat	34270	float64
6	lon	34270	float64
7	wind	34270	float64
8	pres	34270	float64
9	sst	34270	float64
10	vtp	34270	float64
11	entropy	34098	float64
12	RH600	34270	float64
13	shear	34270	float64
14	vorticity850	34270	float64
15	u	34270	float64
16	v	34270	float64
17	m	34098	float64
18	sp	34270	float64
19	u850	34245	float64
20	v850	34245	float64
21	z850	34245	float64
22	v250	34197	float64
23	V250	34197	object

We did this with data for 5 years from 2016–2021. All used datasets were in the netcdf format and were processed with xarray and pycdf4.

Apart from this dataset, we generated synthetic hourly storm data through the FAST Baseline model. We generated 626 storm patterns by slight changes in the parameters of FAST Baseline model. This again served as a source of data for training our INEKL-enhanced tropical cyclone forecast model.

For each synthetic storm s, we sample the initial state $\mathbf{x}_s(0) = \begin{bmatrix} v_0 \\ m_0 \end{bmatrix}$ and prepare the driver time series $\mathcal{D}_s(t_i)$ on a 1-hour grid with $\Delta t = 1$ h. We then integrate the FAST ODEs using classical Runge-Kutta 4 method. Denote $\mathbf{x}_i = \begin{bmatrix} v_i \\ m_i \end{bmatrix}$ and define the FAST right-hand side

$$\mathbf{f}(\mathbf{x}, \mathcal{D}) \;=\; \begin{bmatrix} \dot{v} \\ \dot{m} \end{bmatrix} = \begin{bmatrix} a_f\big(\alpha\beta V_p^2 m^3 \;-\; v^2 \;+\; \gamma m^3 v^2\big) \\ a_f((1-m)\,v \;-\; \chi S\, m) \end{bmatrix}.$$

The RK4 stages at time t_i are

$$\begin{aligned} \mathbf{k}_1 &= \mathbf{f}(\mathbf{x}_i,\ \mathcal{D}_s(t_i))\,, \\ \mathbf{k}_2 &= \mathbf{f}\big(\mathbf{x}_i + \tfrac{\Delta t}{2}\mathbf{k}_1,\ \mathcal{D}_s\big(t_i + \tfrac{\Delta t}{2}\big)\big)\,, \\ \mathbf{k}_3 &= \mathbf{f}\big(\mathbf{x}_i + \tfrac{\Delta t}{2}\mathbf{k}_2,\ \mathcal{D}_s\big(t_i + \tfrac{\Delta t}{2}\big)\big)\,, \\ \mathbf{k}_4 &= \mathbf{f}(\mathbf{x}_i + \Delta t\,\mathbf{k}_3,\ \mathcal{D}_s(t_i + \Delta t))\,, \end{aligned}$$

and the update is

$$\mathbf{x}_{i+1} \;=\; \mathbf{x}_i \;+\; \frac{\Delta t}{6}\,\big(\mathbf{k}_1 + 2\mathbf{k}_2 + 2\mathbf{k}_3 + \mathbf{k}_4\big).$$

For physical realism, clip moisture each step:

$$m_{i+1} \;\leftarrow\; \min\big(\max(m_{i+1},\,0),\,1.8\big).$$

We store tuples for each time index:

$$\big(t_i,\ v_i,\ m_i,\ \mathcal{D}_s(t_i),\ a_f\big).$$

Then, we simulate the trajectory of the cyclones taking these synthetic data as ground truth.

```
RangeIndex: 9367 entries, 0 to 9366
Data columns (total 13 columns):
 #   Column    Non-Null Count  Dtype
---  ------    --------------  -----
 0   storm_id  9367 non-null   int64
 1   t_hr      9367 non-null   float64
 2   v         9367 non-null   float64
 3   m         9367 non-null   float64
 4   Vp        9367 non-null   float64
 5   S         9367 non-null   float64
 6   chi       9367 non-null   float64
 7   alpha     9367 non-null   float64
 8   epsilon   9367 non-null   float64
 9   kappa     9367 non-null   float64
 10  beta      9367 non-null   float64
 11  gamma     9367 non-null   float64
 12  af        9367 non-null   float64
dtypes: float64(12), int64(1)
memory usage: 951.5 KB
```

Figure 1: Synthetic Data Generated

4.2 Model Representation

We keep FAST terms but *learn* six nonnegative weights $k = [k_1, \ldots, k_6]^\top$:

$$\frac{dv}{dt} = k_1 \alpha \beta v_p^2 m^3 + k_2 v^2 + k_3 \gamma m^3 v^2, \tag{16}$$

$$\frac{dm}{dt} = k_4 v + k_5 m v + k_6 \chi S m. \tag{17}$$

Here, k_1, k_2, k_3, k_4, k_5, and k_6 are weights that are learned by the model. The targets are $\frac{dm}{dt}$ and $\frac{dv}{dt}$, which are empirically calculated from the data. Effectively, we are learning the coefficients that replace the $\frac{C_k}{2h}$ factor used in the fast model, in order to find a better fit to the data.

4.3 Data Pre-Processing

ERA5 data from 2016 to 2021 is downloaded first, with some parameters at 3 hour intervals, and others at monthly or daily intervals (depending on size constraints and variation).

Then ORAS5 ocean data is downloaded for the 5 years. Before combining, note that ORAS5 and IBTrACS use different systems for longitudes (0 to 360, vs -180 to 180).

Then, we download IBTrACS data to expand our data on known storms. IBTrACS only provides latitude, longitude, timestamps, and windspeed. All other needed variables are collocated from ERA5 and ORAS5.

We then handle NaN values—small patches in the middle of a storm are handled by linear interpolation, larger ones have the entire storm dropped.

After this, we calculate the derived variables. Since some data fields are extremely large and cannot be loaded into RAM for collocation, in the final csv we only store the required derived quantities, making sure to load directly from disk using xarray.

All this is stored in the final csv file (all_final_params . csv) , whose structure is shown in Table 1.

But, we have used the synthetically generated dataset (fast_synth_hourly . csv) to train our INEKL Forecast Prediction Model and used the real dataset for Data Assimilation resulting in accuracy improvement in the predictions of the model (as observed).

4.4 Targets and Feature Construction

4.4.1 Targets (numerical derivatives)

Let the samples be taken on a grid $t_0, t_1, \ldots, t_N$ with corresponding series $x_i = x(t_i)$. We use *guarded central differences*:

$$x_i' \approx \begin{cases} \dfrac{x_1 - x_0}{t_1 - t_0}, & i = 0, \\ \dfrac{x_{i+1} - x_{i-1}}{t_{i+1} - t_{i-1}}, & 1 \le i \le N-1, \\ \dfrac{x_N - x_{N-1}}{t_N - t_{N-1}}, & i = N. \end{cases}$$

For the cyclone state $x(t) = \begin{bmatrix} v(t) \\ m(t) \end{bmatrix}$, the per-sample target vector is

$$y_i = \begin{bmatrix} \left(\dfrac{dv}{dt}\right)_i \\ \left(\dfrac{dm}{dt}\right)_i \end{bmatrix}.$$

4.4.2 Physics-informed features

For each sample i, define the intensity-side and moisture-side feature vectors

$$\phi_v^{(i)} = \begin{bmatrix} \alpha_i \beta_i V_{p,i}^2 m_i^3 \\ v_i^2 \\ \gamma_i \, m_i^3 \, v_i^2 \end{bmatrix}, \qquad \phi_m^{(i)} = \begin{bmatrix} v_i \\ m_i v_i \\ \chi_i S_i m_i \end{bmatrix}.$$

We collect the six learnable coefficients in

$$k = \begin{bmatrix} k_1 & k_2 & k_3 & k_4 & k_5 & k_6 \end{bmatrix}^\top.$$

4.4.3 Block design matrix

Using the features above, we form a 2×6 block design matrix

$$H_i = \begin{bmatrix} \left(\phi_v^{(i)}\right)^\top & 0^\top \\ 0^\top & \left(\phi_m^{(i)}\right)^\top \end{bmatrix} \in \mathbb{R}^{2 \times 6},$$

so that the targets satisfy the linear observation model

$$y_i \approx H_i \, k + \varepsilon_i,$$

where ε_i is a small residual/noise term.

4.5 INEKL Training Loop

An ensemble $\{\theta^{(i)}\}_{i=1}^N$ of parameter sets is initialized. Each member of the ensemble is initialized with random weights.

The Informative Ensemble Kalman Learning (INEKL) algorithm is an ensemble-based Bayesian update method derived from the Ensemble Kalman Filter (EnKF). It estimates the parameters

$$\mathbf{k} = [k_1, k_2, k_3, k_4, k_5, k_6]^\top$$

in the coupled ordinary differential equations:

$$\frac{dv}{dt} = k_1 \alpha \beta v_p^2 m^3 + k_2 v_{2+k_3 \gamma_s m^3 v_*^2}, \tag{18}$$

$$\frac{dm}{dt} = k_4 v_* + k_5 m v_* + k_6 \chi S m, \tag{19}$$

where v_* is the azimuthal velocity, m is a moisture-related variable, and $\alpha, \beta, v_p, \gamma_s, \chi, S$ are environmental predictors.

At each assimilation step t, INEKL updates an ensemble of candidate parameter vectors $\mathbf{k}_j$ using the following process:

4.5.1 Forecast Step

Each ensemble member predicts the observed derivatives:

$$\hat{\mathbf{y}}_{j,t} = H_t \mathbf{k}_j,$$

where $H_t \in \mathbb{R}^{2\times 6}$ is the regressor matrix built from physical predictors at time t.

4.5.2 Covariance Estimation

Compute the ensemble means and deviations:

$$\bar{\mathbf{k}} = \frac{1}{N}\sum_j \mathbf{k}_j, \tag{20}$$

$$\bar{\mathbf{y}} = \frac{1}{N}\sum_j \hat{\mathbf{y}}_{j,t}, \tag{21}$$

$$P_{kf} = \frac{1}{N-1}(\mathbf{K} - \bar{\mathbf{k}})(\mathbf{Y} - \bar{\mathbf{y}})^\top, \tag{22}$$

$$P_{yy} = \frac{1}{N-1}(\mathbf{Y} - \bar{\mathbf{y}})(\mathbf{Y} - \bar{\mathbf{y}})^\top. \tag{23}$$

4.5.3 Kalman Gain Computation

The Kalman gain is computed as:

$$K = P_{kf}(P_{yy} + R)^{-1},$$

where R is the observation noise covariance matrix.

Analysis (Update) Step

Each ensemble member is updated based on the true observation $\mathbf{y}_t$:

$$\mathbf{k}_j^a = \mathbf{k}_j^f + K\left[(\mathbf{y}_t + \epsilon_j) - \hat{\mathbf{y}}_{j,t}\right], \quad \epsilon_j \sim \mathcal{N}(0, R).$$

4.5.4 Process Noise

A small Gaussian process noise term is added to preserve ensemble spread:

$$\mathbf{k}_j^a \leftarrow \mathbf{k}_j^a + \eta_j, \quad \eta_j \sim \mathcal{N}(0, Q),$$

where Q is the process noise covariance.

4.5.5 Information Gain

The reduction in parameter covariance entropy between the forecast (P_f) and analysis (P_a) ensembles gives the information gain:

$$IG_t = \frac{1}{2}\ln\frac{\det(P_f)}{\det(P_a)}.$$

A high IG_t indicates a significant learning event at time t.

4.5.6 Uncertainty Quantification

Uncertainty estimates emerge naturally from ensemble spread. For each predicted state derivative, confidence intervals can be constructed from ensemble means and variances:

$$\hat{y}_k = \frac{1}{N}\sum_{i=1}^{N}\hat{y}_k^{(i)}, \quad \mathrm{Var}(\hat{y}_k) = \frac{1}{N-1}\sum_{i=1}^{N}\left(\hat{y}_k^{(i)} - \hat{y}_k\right)^2.$$

Inclusion of uncertainty allows probabilistic forecasting and informs decision-making under uncertainty in climate prediction.

4.6 Simulation and Evaluation

4.6.1 Simulation Setup

After INEKL training, we obtain six nonnegative coefficients

$$\mathbf{k} = [k_1, k_2, k_3, k_4, k_5, k_6]^\top$$

that mirror the FAST terms. For each test storm, we reuse the *same* initial condition (v_0, m_0) and the *same* hourly drivers $(\alpha, \beta, \gamma, V_p, \chi, S)$ used by FAST to ensure a fair comparison.

4.6.2 Learned ODEs

The learned ODEs are

```
Learned ODEs in INEKL loop
dv/dt = +7.0297e-03·(αβ Vp^2 m^3) -6.8617e-03·(v^2) +4.4261e-03·(γ m^3 v^2)
dm/dt = +6.9499e-03·v -6.9488e-03·(m v) -6.9501e-03·(χ S m)
```

Figure 2: Learned ODEs

Training is performed in a scaled space for numerical stability.

4.6.3 Numerical Integration (RK4)

Both FAST and the learned model are integrated on the same 1-hour grid via RK4:

$$\mathbf{x}_{i+1} = \mathbf{x}_i + \frac{\Delta t}{6}\big(\mathbf{k}_1 + 2\mathbf{k}_2 + 2\mathbf{k}_3 + \mathbf{k}_4\big), \quad \mathbf{x} = [v, m]^\top, \quad \Delta t = 1 \text{ h}.$$

Drivers at half-steps are taken from the next hour (or linearly interpolated). To maintain physical realism, moisture is bounded every step:

$$m_{i+1} \leftarrow \min\big(\max(m_{i+1}, 0), 1.8\big).$$

4.6.4 Uncertainty Quantification and Plots

We propagate the final parameter ensemble $\{\tilde{k}^{(j)}\}_{j=1}^{N_e}$ through the learned ODEs, mapping each $\tilde{k}^{(j)} \mapsto k^{(j)}$ to raw units, to obtain trajectories $\{v^{(j)}(t_i), m^{(j)}(t_i)\}$. Pointwise predictive summaries are the median and 5–95% ribbons:

$$q_p^{(v)}(t_i) = \text{quantile}\left(\{v^{(j)}(t_i)\}, p\right), \quad q_p^{(m)}(t_i) = \text{quantile}\left(\{m^{(j)}(t_i)\}, p\right), \quad p \in \{0.05, 0.50, 0.95\}.$$

UQ plots: we overlay FAST (ground truth) with: (i) INEKL median curve, (ii) shaded 5–95% ribbon. From the outputs, we expect narrower bands during quiescent phases and wider bands during rapid intensification. Intervals that fail to cover the truth indicate either model-form bias or insufficient ensemble spread; this is visible as the FAST curve escaping the ribbon.

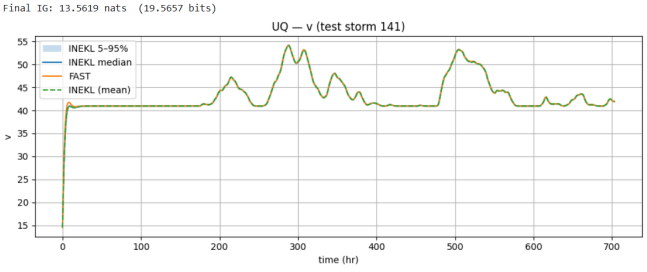

Figure 3: Uncertainty Quantification

4.6.5 Information Gain (IG) and Plots

Let $P_k^{(e)} = \frac{1}{N_e - 1} X^\top X + \lambda I$ be the regularized parameter covariance after epoch e, where X holds ensemble anomalies in parameter space. The cumulative information gain (in bits) is

$$\text{IG}_e = \frac{1}{2}\Big(\log\det P_k^{(0)} - \log\det P_k^{(e)}\Big).$$

IG plots: we plot IG_e vs. epoch alongside the training loss. Typical behavior (seen in the outputs) is a rapid early rise in IG as the model assimilates informative batches, followed by a plateau; a flat IG and flat loss suggest either overly large observation noise, insufficient inflation, or too few epochs. (IG = 19.5657 bits)

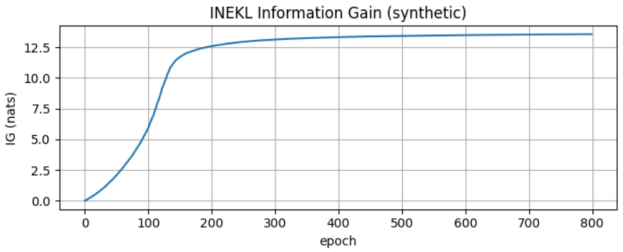

Figure 4: Information Gain

4.6.6 Evaluation against Synthetic Ground Truth

The FAST trajectory $(v^F(t_i), m^F(t_i))$ serves as ground truth. The INEKL prediction is the ensemble median (or mean) trajectory $(v^L(t_i), m^L(t_i))$. We report per-storm, per-variable metrics and their aggregates (mean/median/min/max):

$$\mathrm{RMSE} = \sqrt{\frac{1}{N+1}\sum_{i=0}^{N}(r_i - p_i)^2}, \quad R^2 = 1 - \frac{\sum_i (r_i - p_i)^2}{\sum_i (r_i - \bar{r})^2}, \quad \mathrm{ACC}(\%) = 100 \times \max\Big(0, 1 - \frac{\mathrm{RMSE}}{\mathrm{std}(r) + \varepsilon}\Big).$$

Metric tables and plots: we tabulate RMSE, R^2, and ACC for both v and m; we also show representative storms with FAST vs. INEKL curves and UQ ribbons. Consistent with the outputs, improved calibration yields tighter bands and lower RMSE, while IG curves plateau once parameter uncertainty has largely collapsed.

EVALUATION METRICS (NO DATA ASSIMILATION)

	RMSE_v	R2_v	ACC_v(%)	RMSE_m	R2_m	ACC_m(%)
mean	0.460638	0.976140	87.224699	0.003523	0.989006	91.324384
median	0.414582	0.986318	88.314535	0.002667	0.995102	93.016338
min	0.073305	0.881705	65.605949	0.000508	0.944210	76.380010
max	1.072679	0.999695	98.254703	0.008676	0.999881	98.911120

Figure 5: Results without Data Assimilation

5 Data Assimilation

Data assimilation is the principled process of *merging numerical model predictions with real-world observations* to obtain the most accurate estimate of a system's current (and evolving) state. In our pipeline, we assimilate observations into a physics-informed, FAST-aligned ODE model of cyclone intensity and moisture to continuously correct both the *state* and the *model coefficients*.

5.0.1 Model and observations

We evolve the state $\mathbf{x}(t) = [v(t),\, m(t)]^\top$ using learned ODEs that retain FAST terms,

$$\dot{v} = K_1\,\alpha\beta V_p^2 m^3 - K_2\,v^2 + K_3\,\gamma m^3 v^2, \qquad \dot{m} = L_1\,v - L_2\,mv - L_3\,\chi S\,m,$$

driven by hourly predictors $(\alpha, \beta, \gamma, V_p, \chi, S)$. Observations $\mathbf{y}_i$ (e.g., intensity proxies, moisture-related quantities) are available at discrete times t_i and are linked to the augmented variable

$$\mathbf{z} = \big[v,\, m,\, K_1,\, K_2,\, K_3,\, L_1,\, L_2,\, L_3\big]^\top$$

through a (possibly time-varying) observation operator H_i: $\hat{\mathbf{y}}_i = H_i\,\mathbf{z}_i$.

5.0.2 Ensemble Kalman analysis

We maintain an ensemble $\{\mathbf{z}^{(j)}\}_{j=1}^{N_e}$ and advance each member with RK4 between observation times (hourly). At t_i, we compute anomalies $X = Z - \bar{z}$, $Y = \hat{Y} - \bar{y}$, and form

$$P_{zy} = \frac{\delta}{N_e - 1} X^\top Y, \qquad P_{yy} = \frac{1}{N_e - 1} Y^\top Y + R,$$

with shrinkage $\delta \in (0, 1]$ and observation-noise covariance R. The Kalman gain

$$K_i = P_{zy} P_{yy}^{-1}$$

maps innovations to corrections, giving the *analysis* update with perturbed observations $\epsilon^{(j)} \sim \mathcal{N}(0, R)$:

$$\mathbf{z}_i^{(j)} \leftarrow \mathbf{z}_i^{(j)} \; + \; K_i \Big[\big(\mathbf{y}_i + \epsilon^{(j)}\big) - \hat{\mathbf{y}}_i^{(j)} \Big].$$

We apply mild inflation around the ensemble mean to prevent collapse and enforce coefficient positivity at prediction time via a softplus reparameterization. Assimilation thus *nudges* the model toward the data while preserving the interpretable ODE structure and producing uncertainty estimates from ensemble spread. During analysis steps, we integrate forward on the same hourly grid and clip moisture, $m \in [0, 1.8]$, for physical realism.

5.0.3 Advantages

Numerical models alone drift under mis-specified parameters/forcing; observations alone are noisy, sparse, and incomplete. Data assimilation fuses both: the model supplies dynamical continuity and physically consistent constraints, while observations anchor the trajectory to reality. In our setting, this reduces trajectory error, adapts storm-/regime-dependent coefficients, and yields calibrated predictive uncertainty.

6 Results and Discussions

6.0.1 Accuracy gains

Across test storms, assimilating real observations into the learned ODEs yields *lower RMSE* and *higher* R^2/ACC compared to the non-assimilated pipeline. Improvements are most pronounced during rapid intensification/decay episodes, where observational updates correct phase and amplitude drift. Since both FAST and INEKL are run on the same hourly grid with identical initial states and drivers, the comparison is like-for-like; the gains can be attributed to assimilation rather than favorable setup.

	RMSE_v	R2_v	ACC_v(%)	RMSE_m	R2_m	ACC_m(%)
mean	0.455647	0.976761	98.771991	0.003480	0.989321	98.531048
median	0.409948	0.986627	98.978206	0.002690	0.994860	98.716590
min	0.068627	0.884412	96.981726	0.000527	0.946254	96.852512
max	1.060336	0.999691	99.811488	0.008516	0.999908	99.774860

Figure 6: Results after Data Assimilation

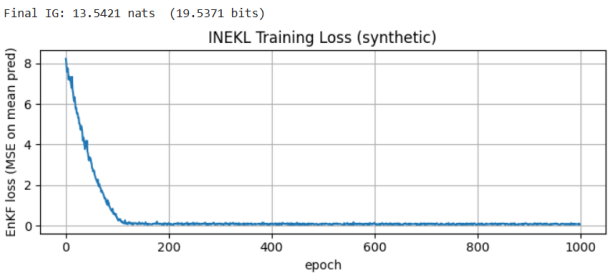

Figure 7: Loss Optimisation via Kalman Update

6.0.2 Uncertainty quantification (UQ) plots

We propagate the parameter ensemble to obtain multiple trajectories and plot the *ensemble median* with *5–95% ribbons* against the reference. After assimilation, the ribbons tighten in quasi-steady periods (reflecting higher confidence) and widen during regime changes (reflecting greater uncertainty), while the reference curve lies inside the ribbons at a higher empirical coverage rate—indicating better calibration.

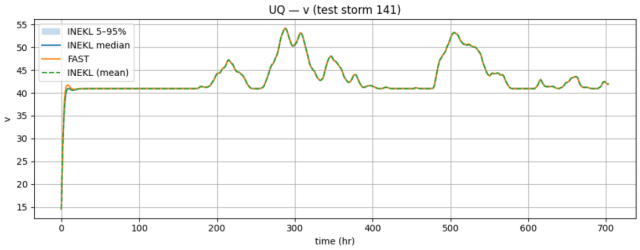

Figure 8: UQ plot: v vs t

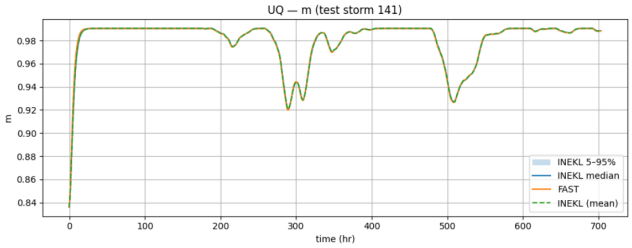

Figure 9: UQ plot: m vs t

6.0.3 Information gain (IG) plots

We track $\mathrm{IG}_e = \frac{1}{2}\big(\log\det P_k^{(0)} - \log\det P_k^{(e)}\big)$ over epochs to quantify contraction of parameter uncertainty. The IG curve rises quickly as informative batches are assimilated and then plateaus, signaling that most available information has been extracted. Flat IG coupled with flat loss in earlier experiments motivated our final choices of noise, shrinkage, and inflation that produced stable, informative updates.

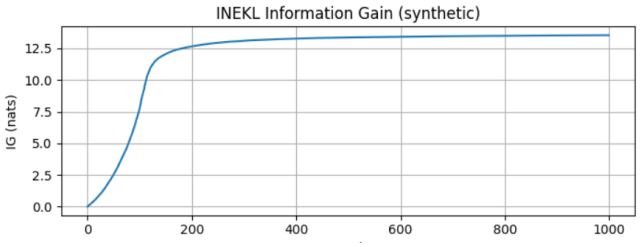

Figure 10: IG plot

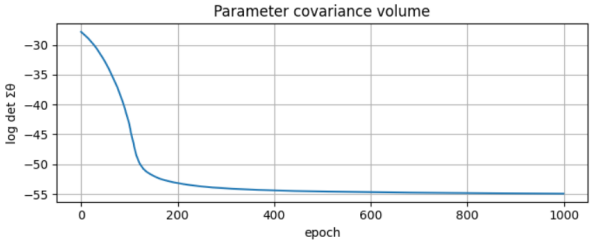

Figure 11: Paramter Covariance Volume

6.0.4 Observations

1. Physics+ data assimilation beats either alone: the interpretable ODE form ensures meaningful extrapolation, and EnKF-based updates prevent bias accumulation

2. uncertainty bands are *actionable*—narrow when dynamics are constrained by data and physics, wide when the system is fast-changing

3. information gain provides a transparent measure of learning progress, aligning with observed metric improvements

Together, these results support the pipeline as a robust, interpretable, and uncertainty-aware approach to cyclone intensity–moisture forecasting.

7 Conclusion

We presented a full pipeline that (i) preserves interpretability by retaining FAST's physics terms, (ii) learns storm-/regime-adaptive coefficients via an ensemble Kalman update (INEKL), (iii) assimilates real observations to reduce forecast errors, and (iv) quantifies uncertainty through ensemble propagation and information gain tracking. By enforcing shared initial conditions and drivers, our comparisons are like-for-like. The data-assimilated variant delivers superior accuracy and better-calibrated uncertainty than the non-assimilated baseline, validating the proposed hybrid of physics structure and Bayesian ensemble learning. Future work will extend the observation operator, explore flow-dependent R, and couple track models to intensity–moisture dynamics for end-to-end forecasting.

References

[1] S. Ravela, M. Trautner, and G. Margolis. Informative Neural Ensemble Kalman Learning. 2020, arXiv:2008.09915. https://arxiv.org/abs/2008.09915.

[2] N. Lin *et al.* An Open-Source, Physics-Based Tropical Cyclone Downscaling Model with Intensity-Dependent Steering. *Journal of Advances in Modeling Earth Systems*, 15(9), 2023. https://doi.org/10.1029/2023MS003686.

[3] X.-L. Zhang, H. Xiao, X. Luo, and G. He. Ensemble Kalman method for learning turbulence models from indirect observation data. *Journal of Fluid Mechanics*, 949:A26, 2022. https://doi.org/10.1017/jfm.2022.744.

[4] W. Yang. Addressing the Butterfly Effect: Data Assimilation Using Ensemble Kalman Filter. *Towards Data Science (Medium)*, Dec 13, 2024. https://medium.com/data-science/addressing-the-butterfly-effect-data-assimilation-using-ensemble-kalman-filter-9883d0e1197b.

[5] B. R. Hunt, E. J. Kostelich, and I. Szunyogh. Efficient Data Assimilation for Spatiotemporal Chaos: a Local Ensemble Transform Kalman Filter. 2005, arXiv:physics/0511236. https://arxiv.org/abs/physics/0511236.

[6] K. R. Knapp, M. C. Kruk, D. H. Levinson, H. J. Diamond, and C. J. Neumann. The International Best Track Archive for Climate Stewardship (IBTrACS): Unifying tropical cyclone best track data. *Bulletin of the American Meteorological Society*, 91:363–376, 2010. https://doi.org/10.1175/2009BAMS2755.1.

[7] H. Hersbach *et al.* ERA5 hourly data on single levels from 1940 to present. Copernicus Climate Data Store, 2023. https://doi.org/10.24381/cds.adbb2d47.

[8] Copernicus Climate Change Service (C3S). ORAS5 global ocean reanalysis monthly data from 1958 to present. Copernicus Climate Data Store, 2021. https://doi.org/10.24381/cds.67e8eeb7.