DYNABR: DYNAMICS-INTEGRATED BASE–RESIDUAL LEARNING FRAMEWORK FOR NON-COOPERATIVE SATELLITE ORBIT FORECASTING

Anonymous authors
Paper under double-blind review

ABSTRACT

Growing congestion in low Earth orbit (LEO) makes accurate orbit forecasting essential for spacecraft safety. For non-cooperative targets, forecasters rely solely on external observations. Uncertain target properties and environments limit physics-based perturbation modeling, while data-driven models fit complex patterns but struggle to incorporate known dynamics. We first construct a dataset covering 2,000 LEO targets, including satellites, rocket bodies, and space debris, by propagating discrete two-line element (TLE) records into regularly sampled orbital state sequences. We then propose DynaBR, a unified forecasting model. Global Dynamics Reference provides the dominant orbital structure; Local Dynamics Alignment calibrates future references using historical phase discrepancies; and Dynamics-Conditioned Residual Forecasting predicts corrections from historical residuals and aligned future references to compensate for perturbation effects that approximate dynamics cannot explain. Experiments show that DynaBR achieves the strongest overall forecasting performance on unseen targets, achieves the strongest overall performance across forecasting horizons while substantially outperforming both physics-based and data-driven methods at longer horizons and across all target types. Over a 48-hour horizon, DynaBR achieves an average relative reduction of 99.45% in mean position error across 12 SOTA baselines.

1 INTRODUCTION

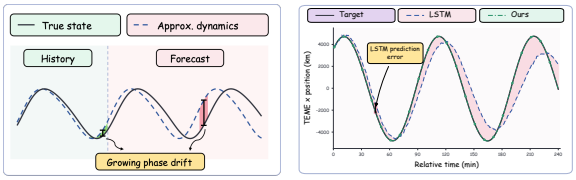
As of July 2026, space surveillance networks regularly tracked approximately 47,070 objects in Earth orbit (European Space Agency, 2026). Forecasting non-cooperative objects, including satellites that do not share navigation data, rocket bodies, and space debris, is increasingly important for conjunction screening and spacecraft safety. These targets provide neither navigation data nor precise ephemerides, so forecasters rely on maintained orbital records that are irregularly updated and imperfect. The task is to predict their motion over many orbital revolutions from this limited information.

Physics-based models provide a strong foundation for orbital forecasting by encoding established dynamical priors and directly capturing the dominant orbital motion. Their accuracy, however, is limited by effects that are difficult to observe or parameterize for non-cooperative targets, including uncertain shapes, attitudes, area-to-mass ratios, and environmental conditions (Mehta et al., 2023). Data-driven models offer greater flexibility in modeling such unresolved effects, but when used alone must learn both the dominant orbital dynamics and their deviations from data. These two paradigms are therefore naturally complementary: physics provides transferable predictive structure, while learning can model discrepancies beyond the available physical description (Pathak et al., 2018; Yin et al., 2021). Effectively realizing this complementarity, however, presents two key challenges.

First, a dynamics prior shared across targets may not match each target's local evolution: target-specific properties and time-varying perturbations can induce accumulating phase discrepancies, as figure 1(a) illustrates. Our preliminary experiment on 32 validation windows shows that calibrating the reference using only historical phase trends reduces mean endpoint position error by an average of 78.86% across five forecast horizons, supporting local adaptation of the shared prior using recent history. Second, local adaptation leaves unexplained discrepancies, while direct forecasting without dynamics guidance also faces difficulties. figure 1(b) illustrates LSTM's trajectory deviations;

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

over the full 48-hour test interval, its mean position error reaches 6921.521 km, compared with 17.662 km for the physics-based baseline. This gap highlights the value of explicit dynamical priors for long-horizon orbit forecasting. Effective integration therefore requires retaining the global dynamics reference while combining historical residuals with the locally aligned future reference to fit remaining variations that resist explicit modeling for non-cooperative targets.



(a) (b)

Figure 1: Challenges in non-cooperative orbit forecasting: (a) accumulating phase drift under approximate dynamics.; (b) non-negligible forecasting errors.

To address these challenges, we propose DynaBR, a unified model that integrates dynamical references with data-driven corrections. Global Dynamics Reference uses dynamical laws shared across targets to generate historical and future references that capture dominant orbital structure. Local Dynamics Alignment adjusts the future reference using phase discrepancies in the current historical window, mitigating phase drift during long-term extrapolation. Dynamics-Conditioned Residual Forecasting encodes historical residuals relative to the raw reference and predicts corrections using the aligned future reference and forecast lead time. The aligned reference conditions correction prediction and provides the trajectory reconstruction baseline, integrating local alignment into the shared network's predictions. This design fits target-specific perturbation effects from data and reduces reliance on complete perturbation models and known target properties.

Over a 48-hour forecast horizon on unseen targets, DynaBR achieves a mean position error of 0.975 km, compared with 17.662 km for the strongest physics-based baseline. Against SOTA data-driven baselines TimesNet and TimesFM-3, DynaBR improves prediction accuracy by more than 300-fold in both comparisons, measured by the ratio of baseline mean position error to DynaBR's. DynaBR also exhibits lower variability in position and velocity errors, indicating greater prediction stability. We further evaluate the effects of sequence construction and model design through controlled data comparisons and assessments across forecast horizons and target types.

Our contributions are as follows:

- We construct a TLE-derived orbital time-series dataset covering diverse non-cooperative objects, with unified states and object-disjoint evaluation.
- We propose DynaBR, which calibrates a dynamical reference using historical phase discrepancies and learns the remaining corrections conditioned on the aligned future reference.
- We show that phase alignment substantially improves the dynamical reference, while learned corrections further improve long-horizon forecasts on unseen objects.

We will release the dataset, model code, and data construction and evaluation tools.

2 Preliminaries

2.1 Problem Statement

For target i, we represent its orbital state at time step t as $\mathbf{x}_t^{(i)} = [(\mathbf{r}_t^{(i)})^\top, (\mathbf{v}_t^{(i)})^\top]^\top \in \mathbb{R}^6$, where $\mathbf{r}_t^{(i)}$ and $\mathbf{v}_t^{(i)}$ denote position and velocity. We sample states at a fixed interval Δt. Given L historical states, the model predicts the next H states:

$$\widehat{\mathbf{X}}_{L+1:L+H}^{(i)} = F_\theta\left(\mathbf{X}_{1:L}^{(i)}\right). \tag{1}$$

The model shares parameters θ across targets. We use disjoint target sets for training, validation, and testing to evaluate generalization to unseen non-cooperative targets. At inference, the model receives only historical positions and velocities.

2.2 BASIC CONCEPTS

Non-cooperative Targets. Non-cooperative targets include satellites, rocket bodies, and space debris that provide forecasters with neither navigation data nor precise ephemerides. Forecasters rely on external observations, such as radar range measurements and optical angular measurements, and estimate target states through orbit determination. Limited observation windows and measurement errors constrain the completeness and accuracy of historical states, while uncertain target properties further complicate orbital extrapolation. Collisions with these targets can threaten operational spacecraft and generate additional debris (Kazemi et al., 2024).

Orbital Records and States. The two-line element (TLE) format records a space object's orbital parameters in two fixed-width lines of text, describing its orbit at a reference epoch. Through orbital propagation, we compute its three position components (x, y, z) and three velocity components (v_x, v_y, v_z) at a specified time. Together, these six components form the orbital state $\mathbf{x} = [x, y, z, v_x, v_y, v_z]^\top \in \mathbb{R}^6$ (Vallado et al., 2006).

Orbital Dynamics. Orbital states and dynamical equations require an explicit reference frame. We use the True Equator Mean Equinox (TEME) frame for all input and output states. In an Earth-centered inertial frame, central gravity and perturbations govern orbital evolution:

$$\dot{\mathbf{r}}_I = \mathbf{v}_I, \qquad \dot{\mathbf{v}}_I = -\frac{\mu}{\|\mathbf{r}_I\|^3}\mathbf{r}_I + \mathbf{a}_p(\mathbf{r}_I, \mathbf{v}_I, t), \tag{2}$$

where the subscript I denotes inertial-frame quantities, μ is Earth's gravitational parameter, and $\mathbf{a}_p$ denotes perturbing acceleration. Setting $\mathbf{a}_p = \mathbf{0}$ yields the two-body model; additional terms account for effects such as Earth's nonspherical gravity and atmospheric drag. Practical models retain only selected perturbations and rely on limited knowledge of environmental and target parameters, so their predictions may deviate from target trajectories.

2.3 RELATED WORK

Orbit Prediction. Orbit prediction involves multivariate time-series forecasting with limited observations, perturbations, and dynamical differences across non-cooperative targets. Keplerian Dynamics(KepDyn) models central gravity, while Perturbed Orbital Dynamics(PertDyn) adds perturbations. Data-driven methods include linear DLinear (Zeng et al., 2023), recurrent LSTM and GRU (Hochreiter & Schmidhuber, 1997; Cho et al., 2014), convolutional TCN and TimesNet (Bai et al., 2018; Wu et al., 2023), and Transformer-based PatchTST, iTransformer, and Informer (Nie et al., 2023; Liu et al., 2024; Zhou et al., 2021). Pretrained models enable zero-shot forecasting (Das et al., 2024; Ansari et al., 2024); our baselines include TimesFM-3 and Chronos-2. Studies investigate propagation error correction, physics-informed prediction, and position–velocity coupling (Curzi et al., 2022; Zhang et al., 2026; Yang et al., 2026). Model and parameter errors limit physical approaches, while generic data-driven models lack explicit orbital structure and must fit dominant dynamics and their deviations.

Physics-Informed Machine Learning. Physics-informed machine learning (PIML) incorporates physical knowledge into models. Physics-informed neural networks (PINNs) compute equation residuals through automatic differentiation and combine them with data and initial or boundary condition errors to solve forward and inverse problems (Raissi et al., 2019). XPINNs introduce space–time domain decomposition (Jagtap & Karniadakis, 2020), while gradient statistics guide loss balancing (Wang et al., 2020). These methods still depend on the prescribed dynamics. Unknown shapes, attitudes, and area-to-mass ratios limit perturbation modeling for non-cooperative targets (Mehta et al., 2023). Small equation residuals therefore need not imply accurate trajectories, and excessive physical regularization may restrict corrections to dynamical model errors.

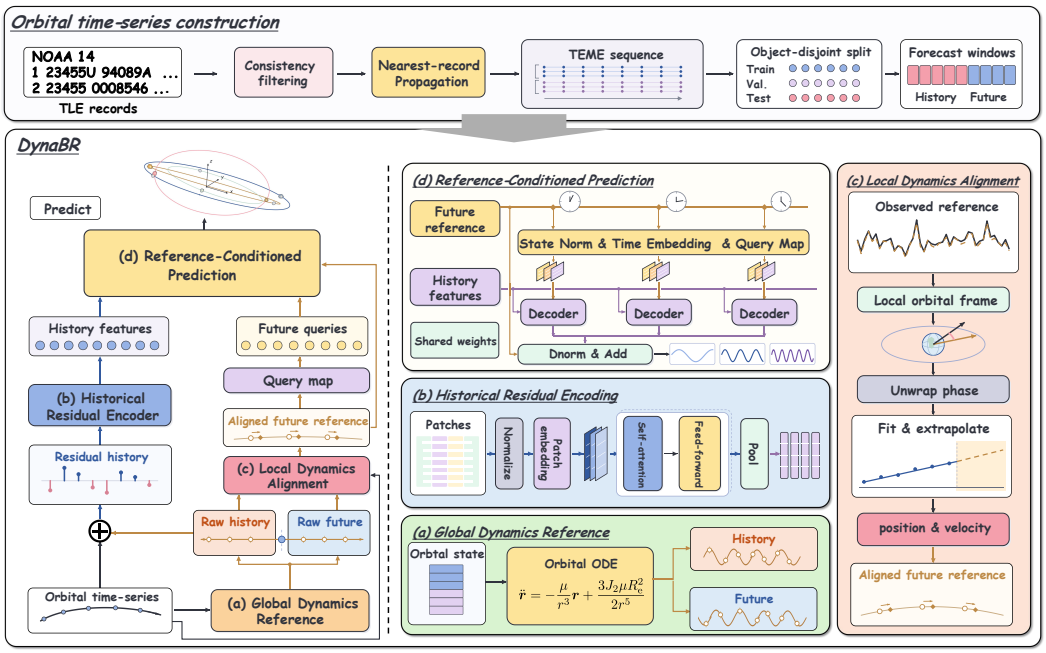


Figure 2: DynaBR Overview.

3 METHODOLOGY

3.1 METHOD OVERVIEW

We first construct unified orbital sequences from discrete TLE records (section 3.2). As figure 2 illustrates, DynaBR integrates global dynamics guidance, local phase calibration, and residual forecasting within a unified model to predict future orbits of non-cooperative targets.

Global Dynamics Reference. The model uses orbital dynamics common across targets to generate historical and future references, providing the dominant orbital structure so the network can focus on fitting residual discrepancies (section 3.3).

Local Dynamics Alignment. The model fits historical phase-discrepancy trends to calibrate the future reference and mitigate phase-error accumulation during long-term extrapolation (section 3.4).

Dynamics-Conditioned Residual Forecasting. A shared network encodes historical residuals against the raw reference, uses the calibrated reference and lead time to predict corrections, and reconstructs future states (section 3.5). The calibrated reference conditions correction prediction and provides the reconstruction baseline, integrating local phase calibration into the network's predictions.

3.2 ORBITAL TIME-SERIES CONSTRUCTION

We construct an orbital time-series dataset from more than 21 million valid TLE records, covering 2000 LEO non-cooperative targets: 283 satellites, 157 rocket bodies, 803 debris, and 757 unknown objects. This coverage supports unified modeling and cross-target generalization evaluation.

We remove records with implausible orbital parameters and identify anomalous records based on their consistency with the surrounding orbital history. Long temporal gaps are used to divide each target history into independent trajectory segments. Detailed screening criteria and thresholds are provided in Appendix A.1.

Let $\tau_{i,k}$ denote the epoch of TLE record $\mathcal{E}_{i,k}$ for target i. For each sampling time t, we select the most recent TLE available at or before t,

$$k^*(t) = \arg \max_{k:\, \tau_{i,k} \le t} \tau_{i,k}, \tag{3}$$

and propagate it forward under the TLE convention to obtain the TEME position–velocity state

$$\mathbf{x}_i(t) = \mathcal{P}\big(t; \mathcal{E}_{i,k^*(t)}\big) \in \mathbb{R}^6. \tag{4}$$

Thus, the state at each time is constructed exclusively from orbital information available no later than that time, without combining results from future TLE records.

We sample the propagated states every Δt and extract windows containing L historical and H future states. Targets are assigned to disjoint training, validation, and test sets, ensuring that evaluation measures generalization to previously unseen objects under a common forecasting setting.

3.3 Global Dynamics Reference

Forecasters often lack information about non-cooperative targets' shapes, attitudes, and area-to-mass ratios, limiting accurate modeling of target-specific perturbations. Known gravitational laws nevertheless describe the dominant orbital structure. We therefore construct an approximate dynamical model with central gravity and the dominant perturbation from Earth's nonspherical gravity. Starting from the last historical state $\mathbf{x}_L$, we propagate backward through the historical interval and forward through the forecast interval:

$$\mathbf{c}_j = \Phi_{\mathrm{dyn}}(t_j, t_L; \mathbf{x}_L), \qquad j = 1, \ldots, L+H, \qquad \mathbf{c}_L = \mathbf{x}_L. \tag{5}$$

Here, Φ_{dyn} denotes numerical orbital propagation, and $\mathbf{c}_j = [(\mathbf{r}_j^c)^\top, (\mathbf{v}_j^c)^\top]^\top$ contains reference position and velocity. The historical and future references thus follow the same dynamical trajectory. This design provides the dominant orbital structure through dynamics common across targets, allowing the shared network to focus on discrepancies from incomplete dynamics and reducing the burden of fitting entire trajectories.

3.4 Local Dynamics Alignment

Incomplete dynamics can produce phase drift: the reference progresses along its orbit at a different rate from the target, accumulating discrepancies during extrapolation. We estimate the historical phase-discrepancy trend and adjust the future reference, correcting this systematic error without directly estimating unknown physical parameters.

We construct radial, along-track, and normal unit vectors $(\mathbf{e}_{R,j}, \mathbf{e}_{T,j}, \mathbf{e}_{N,j})$ from the historical reference and compute the in-plane phase discrepancy:

$$\phi_j = \mathrm{unwrap}[\mathrm{atan2}(\langle \mathbf{r}_j, \mathbf{e}_{T,j} \rangle, \langle \mathbf{r}_j, \mathbf{e}_{R,j} \rangle)]\,. \tag{6}$$

The unwrap operation removes artificial 2π discontinuities. We estimate the least-squares phase slope $\widehat{\beta}$ over the last $W = 1280$ historical samples and apply $\Delta\phi_{L+h} = \phi_L + h\widehat{\beta}$, anchored to the measured terminal phase. Since $\mathbf{c}_L = \mathbf{x}_L$, $\phi_L = 0$ modulo 2π, so the rotation is equivalent to $h\widehat{\beta}$ (Appendix A.2). We rotate the future reference position and velocity around the local orbital normal:

$$\mathbf{b}_{L+h}^{+} = \begin{bmatrix} \mathcal{R}(\mathbf{e}_{N,L+h}, \Delta\phi_{L+h})\, \mathbf{r}_{L+h}^c \\ \mathcal{R}(\mathbf{e}_{N,L+h}, \Delta\phi_{L+h})\, \mathbf{v}_{L+h}^c \end{bmatrix}, \qquad h = 1, \ldots, H, \tag{7}$$

where $\mathcal{R}$ denotes an axis-angle rotation. This phase calibration aligns the future reference with the target's recent evolution, providing an adapted reference for residual forecasting.

3.5 Dynamics-Conditioned Residual Forecasting

Phase alignment cannot eliminate all forecast discrepancies from incomplete perturbation modeling. We therefore use a shared sequence network to fit temporal patterns reflecting target-specific perturbations in historical residuals and predict corrections using the aligned future reference and forecast lead time. This design uses data to capture perturbation effects that explicit dynamics cannot adequately describe, supporting more accurate orbit forecasts with less dependence on target properties and complete perturbation equations.

3.5.1 HISTORICAL RESIDUAL ENCODING

We compute historical residuals relative to the raw dynamics reference and extract their temporal representation through a shared encoder:

$$\mathbf{d}_{1:L}^{\text{hist}} = \mathbf{x}_{1:L} - \mathbf{c}_{1:L}, \qquad \overline{\mathbf{h}} = \text{Pool}\big[\mathcal{E}_\theta\big(\mathcal{N}_d(\mathbf{d}_{1:L}^{\text{hist}})\big)\big]. \tag{8}$$

Here, $\mathcal{N}_d$ normalizes residual channels using training-set statistics, while $\mathcal{E}_\theta$ includes joint temporal patching, embedding, and Transformer encoding. We partition all six synchronized channels into non-overlapping temporal patches, preserving simultaneous position–velocity discrepancies while shortening the encoding sequence.

Historical residuals retain discrepancy patterns relative to the raw reference, while future corrections use the aligned reference. This design allows the network to use historical discrepancy evolution to predict errors that remain after phase calibration.

3.5.2 REFERENCE-CONDITIONED PREDICTION

The historical representation alone does not explicitly specify the reference state at each forecast step. We therefore encode the normalized future reference $\widetilde{\mathbf{b}}_{L+h}^{+}$ and lead time $\tau_h = h/H$ into a query, then combine it with the historical representation to predict a correction:

$$\mathbf{q}_h = \mathcal{Q}_\theta\Big([\widetilde{\mathbf{b}}_{L+h}^{+}, \tau_h, \sin(2\pi\tau_h), \cos(2\pi\tau_h)]\Big), \quad \widehat{\widetilde{\mathbf{d}}}_{L+h} = \mathcal{G}_\theta([\overline{\mathbf{h}}, \mathbf{q}_h]). \tag{9}$$

The future reference supplies orbital-state information, while lead time distinguishes extrapolation stages, allowing the network to predict different corrections at similar orbital positions across successive cycles. We normalize reference states using statistics from the current historical window. The model encodes history once and predicts all future corrections in parallel.

3.5.3 TRAJECTORY RECONSTRUCTION AND TRAINING

We restore corrections to physical units and add them to the aligned future reference to obtain position and velocity forecasts:

$$\widehat{\mathbf{y}}_{L+h} = \mathbf{b}_{L+h}^{+} + \mathcal{N}_{\text{out}}^{-1}\left(\widehat{\widetilde{\mathbf{d}}}_{L+h}\right). \tag{10}$$

Here, $\mathcal{N}_{\text{out}}^{-1}$ restores corrections to physical units using the output scales defined in Appendix A.2. These differ from the input scales used by $\mathcal{N}_d$.

The aligned reference both conditions correction prediction and provides the reconstruction baseline, integrating local dynamics alignment into the shared network's predictions. We optimize the network through the error between reconstructed and target trajectories:

$$\mathcal{L}(\theta) = \mathbb{E}_{(\mathbf{x}_{1:L}, \mathbf{y}_{L+1:L+H}) \sim \mathcal{D}_{\text{train}}}\left[\ell_{\text{phys}}(\widehat{\mathbf{y}}_{L+1:L+H}, \mathbf{y}_{L+1:L+H})\right], \tag{11}$$

where ℓ_{phys} measures state prediction errors after reconstruction in physical units. The encoder, query map, and decoder share parameters across targets, while the dynamics reference and local alignment depend on each input. DynaBR therefore forecasts unseen targets using only their historical states.

4 EXPERIMENTS

4.1 EXPERIMENTAL SETUP

Dataset and Forecasting Horizon. We conduct experiments on the orbital time-series dataset constructed in section 3.2. Each object is represented by a position–velocity sequence in the TEME coordinate frame, sampled every $\Delta t = 60$ seconds. Unless otherwise specified, each forecasting window contains $L = 2048$ historical states and $H = 2880$ future states, corresponding to a 48-hour forecast horizon. The training, validation, and test objects are disjoint.

Evaluation Metrics. We compute Euclidean position and velocity errors in kilometers and kilometers per second, respectively. We report object-balanced mean errors, standard deviations, and the pooled 95th percentile (p95) over all forecast points. For horizon analysis, we evaluate prefixes of the same complete forecast trajectory.

Table 1: 48-hour cumulative-prefix errors on 200 test objects. Mean and Std summarize object-mean errors; p95 pools all forecast points. Bold and underlined values denote best and second-best results.

Category	Method	Position (km)			Velocity (km/s)		
		Mean	p95	Std	Mean	p95	Std
Physics-based	KepDyn	884.8571	2609.2773	298.4324	0.8864	2.6558	0.3210
	PertDyn	<u>17.6619</u>	<u>100.6779</u>	<u>28.0329</u>	<u>0.0176</u>	<u>0.0996</u>	<u>0.0278</u>
Data-driven	DLinear	6497.3262	11950.3272	2185.2808	6.4926	12.3062	2.4040
	TCN	4285.9565	8398.8446	1511.5763	4.2717	8.5473	1.6282
	LSTM	6921.5213	9119.5392	483.0518	6.8516	9.1044	0.5494
	GRU	6906.3986	9074.3035	459.6345	6.8339	9.0652	0.5370
	PatchTST	558.1302	1069.7751	564.8316	0.6180	1.3218	0.6383
	iTransformer	858.5259	1973.7706	841.8029	0.9026	2.0991	0.9438
	Informer	7074.2711	8339.0279	274.0605	7.0166	8.1985	0.3458
	TimesNet	331.8117	680.3908	610.0400	0.3850	0.8274	0.6849
Temporal Foundation	TimesFM-3	389.4962	1634.2439	168.9574	0.3632	1.4861	0.1794
	Chronos-2	5638.2227	13726.3778	2078.9286	5.6395	13.9679	2.1934
Ours	**DynaBR**	**0.9747**	**4.0436**	**1.2229**	**0.0009**	**0.0037**	**0.0011**

Implementation Details. We run the experiments on a server with an Intel i9-14900KF CPU, 60 GB of RAM, and an NVIDIA RTX 5090 GPU, using Python 3.12.13 and PyTorch 2.11.0+cu130.

4.2 Performance Comparison

Overall Performance. Table 1 compares all methods over the 48-hour forecast horizon for non-cooperative targets. DynaBR achieves a mean position error of 0.975 km and a mean velocity error of 0.0009 km/s. Compared with the strongest physics-based baseline, DynaBR reduces position and velocity errors by 94.5% and 94.9%, respectively. Compared with TimesNet, it reduces the mean position error by more than 99.7%. DynaBR also achieves lower p95 errors and error dispersion, indicating more stable performance across targets and forecast windows.

Performance across Forecast Horizons. We evaluate five forecast horizons: 1, 6, 12, 24, and 48 hours. Figure 3 shows that errors increase with the forecast horizon for most of methods. Perturbed Orbital Dynamics performs similarly to DynaBR at 1 hour, but its errors grow faster at longer horizons. From 6 to 48 hours, DynaBR maintains lower position and velocity errors with smaller error dispersion, indicating that it limits error accumulation during long-horizon forecasting.

Performance across Target Types. We evaluate each method on satellites, rocket bodies, and space debris under the same model configuration. Table 2 shows that DynaBR outperforms all baselines across the three non-cooperative target types. Its mean position errors range from 0.381 to 1.514 km, compared with 6.764–28.766 km for the strongest baseline, corresponding to reductions of approximately 94–95%. Space debris presents the greatest forecasting difficulty because its physical properties are more uncertain, yet DynaBR still achieves a mean position error of 1.514 km and reduces the strongest-baseline error by 94.7%. These results show that the method maintains its advantage across diverse non-cooperative targets.

Table 2: Test object-mean errors over 48 hours: position (km) and velocity (km/s).

Method	Satellites		Rocket		Debris	
	Pos.	Vel.	Pos.	Vel.	Pos.	Vel.
TimesNet	323.901	0.3956	319.469	0.3420	286.648	0.3630
TimesFM-3	381.680	0.3543	451.199	0.4329	363.650	0.3368
KepDyn	779.178	0.7626	904.592	0.9261	982.264	0.9841
PertDyn	21.564	0.0214	6.764	0.0068	28.766	0.0285
DynaBR	**1.125**	**0.0011**	**0.381**	**0.0003**	**1.514**	**0.0014**

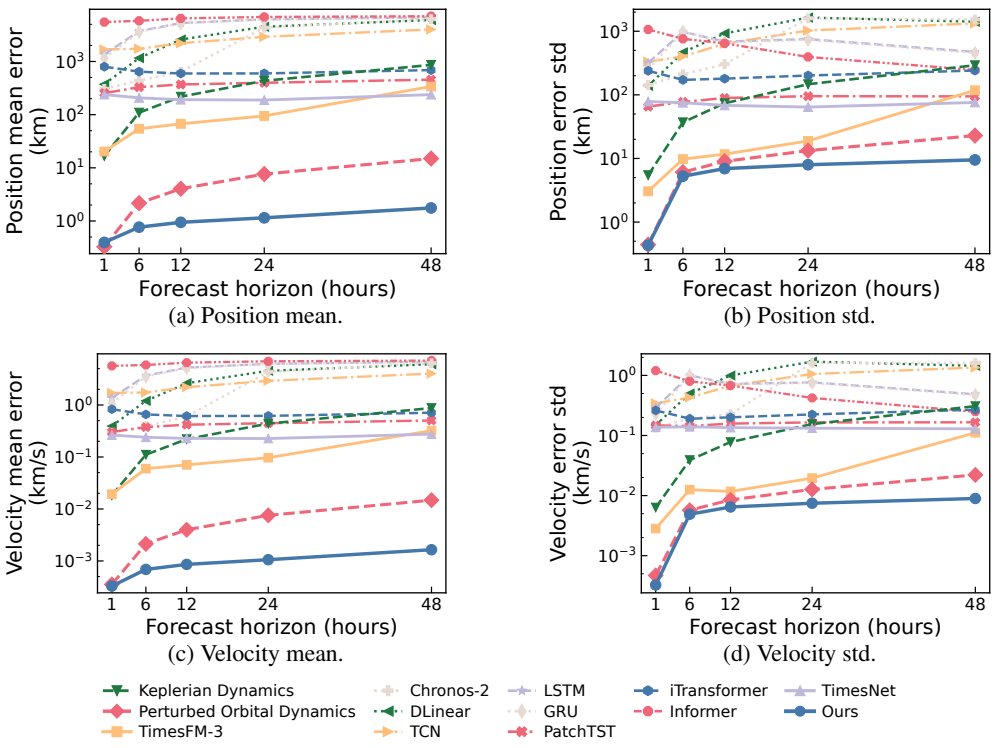


Figure 3: Forecasting performance across different forecast horizons. We report the mean and standard deviation of position and velocity errors over progressively longer forecast prefixes.

4.3 Ablation Study

Ablation on Dynamics Reference. We first evaluate how Local Dynamics Alignment improves the global dynamics reference without residual forecasting, isolating its ability to correct systematic discrepancies during long-horizon extrapolation. Table 3 shows that alignment reduces the mean position error from 14.937 km to 2.562 km, an 82.85% reduction. It also reduces the mean velocity error by 86.04% and the position p95 error by 90.08%. These results indicate that recent historical discrepancies contain phase trends useful for forecasting. Local alignment exploits these trends to limit error accumulation and provide a more accurate reference for residual forecasting. For non-cooperative targets with uncertain physical properties, this design calibrates reference trajectories directly from historical states, reducing reliance on additional target parameters.

Ablations on the Aligned Dynamics Reference. We fix the aligned dynamics reference across all variants to examine historical residual representation and future dynamics conditioning within Dynamics-Conditioned Residual Forecasting. Table 4 shows that replacing historical residuals with absolute states increases the mean position error by 30.36%. Residual representation therefore helps the network focus on variations that approximate dynamics leave unresolved, reducing the burden of fitting complete orbital evolution. Removing future dynamics conditioning increases the mean velocity error by 25.53%, indicating that the future reference provides both a reconstruction baseline and dynamical state information at each forecast time. Together, these designs enable the model to use historical discrepancies and future references to compensate for perturbation effects that resist explicit modeling for non-cooperative targets.

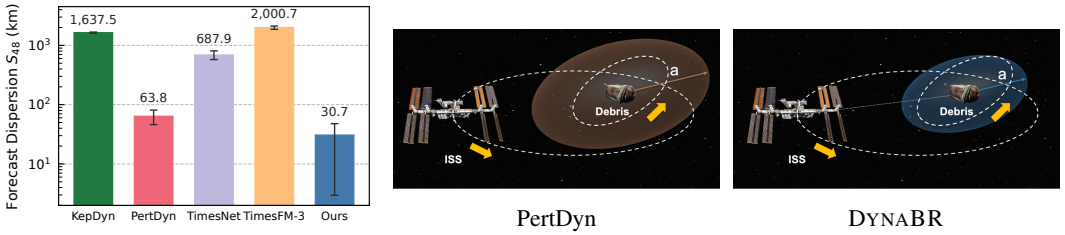
Both design choices contribute beyond the aligned dynamics reference. Replacing historical residuals with absolute state histories increases the mean position error from 1.759 km to 2.293 km. This indicates that representing history relative to the dynamics reference provides a more effective description of the remaining model discrepancy than encoding the full orbital trajectory. Removing future dynamics conditioning produces a smaller increase in mean position error, but noticeably degrades the mean velocity error to 0.002065 km/s. This suggests that the future reference is useful not only as a reconstruction baseline, but also as context for predicting horizon-dependent corrections.

Table 3: Dynamics reference ablation.

Reference	Position		Velocity	
	Mean	Decrease	Mean	Decrease
Raw	14.937	–	0.014777	–
Aligned	**2.562**	**82.85%**	**0.002063**	**86.04%**

Table 4: Residual forecasting ablations.

Variant	Position		Velocity	
	Mean	Increase	Mean	Increase
w/o Hist. Residuals	2.293	30.36%	0.001908	15.95%
w/o Future Conditioning	1.867	6.15%	0.002065	25.53%
DYNABR	**1.759**	–	**0.001645**	–



(a) Forecast uncertainty

(b) Uncertainty ellipsoids

Figure 4: Forecast uncertainty in an ISS conjunction scenario. (a) Position-error dispersion across forecast horizons. (b) Schematic comparison of position uncertainty ellipsoids for PertDyn and DYNABR at 48 hours. Arrows indicate motion directions, and a denotes a schematic major semiaxis. Ellipsoid dimensions, object sizes, and distances are illustrative; the equivalent radii reported in the text follow a Gaussian, isotropic approximation.

4.4 CASE STUDY: FORECAST UNCERTAINTY IN ISS CONJUNCTION ASSESSMENT

To examine the relevance of non-cooperative orbit forecasting to conjunction assessment, we consider a controlled encounter between the International Space Station (ISS) and a non-cooperative debris object. We use the 48-hour validation predictions to estimate object-balanced position-error covariances in the local RTN frame and hold the nominal encounter geometry fixed across methods.

figure 4(a) compares position-error dispersion at 48 hours. We quantify its overall scale as $S_h = \sqrt{\mathrm{tr}(\Sigma_h)}$, where Σ_h denotes the position-error covariance at forecast horizon h. At 48 hours, DYNABR achieves $S_{48} = 30.7$ km, compared with 63.8 km for PertDyn, while the other baselines exhibit substantially larger dispersion. These results show that DYNABR provides more concentrated position estimates for non-cooperative targets over long forecast horizons.

figure 4(b) illustrates the corresponding position uncertainty regions for conjunction assessment. Under a Gaussian, isotropic approximation and neglecting ISS position uncertainty, we compute the equivalent radius at the nominal 95% confidence level as $a = \sqrt{\chi^2_{3,0.95}/3}\, S_{48}$. This approximation yields an equivalent radius of approximately 103.0 km for PertDyn and 49.6 km for DYNABR. The ellipsoids illustrate this difference schematically; their drawn semiaxes do not represent measured covariance eigenvalues. The smaller equivalent radius demonstrates that DYNABR substantially reduces the spatial dispersion of forecast errors. By combining a global dynamics reference, local dynamics alignment, and dynamics-conditioned residual forecasting, DYNABR compensates for deviations that approximate dynamics leave unresolved and provides more concentrated position estimates for subsequent conjunction screening.

5 CONCLUSION

We proposed DynaBR, a unified framework for non-cooperative orbit forecasting that integrates Global Dynamics Reference, Local Dynamics Alignment, and Dynamics-Conditioned Residual Forecasting. Experiments on our orbital time-series dataset demonstrate higher prediction accuracy and lower error dispersion than competing baselines across satellites, rocket bodies, and space debris. In future work, we will extend these forecasts to intelligent spacecraft control to support collision avoidance and maneuver planning in congested orbital environments.

AI USE STATEMENT

Generative AI tools were used during this work to polish writing. We have reviewed all AI-assisted work. We take responsibility for the final content of this work.

REFERENCES

Abdul Fatir Ansari, Lorenzo Stella, Caner Turkmen, Xiyuan Zhang, Pedro Mercado, Huibin Shen, Oleksandr Shchur, Syama Sundar Rangapuram, Sebastian Pineda Arango, Shubham Kapoor, Jasper Zschiegner, Danielle C. Maddix, Hao Wang, Michael W. Mahoney, Kari Torkkola, Andrew Gordon Wilson, Michael Bohlke-Schneider, and Yuyang Wang. Chronos: Learning the language of time series. *Transactions on Machine Learning Research*, 2024. URL https://openreview.net/forum?id=gerNCVqqtR.

Shaojie Bai, J. Zico Kolter, and Vladlen Koltun. An empirical evaluation of generic convolutional and recurrent networks for sequence modeling, 2018. URL https://arxiv.org/abs/1803.01271.

Kyunghyun Cho, Bart van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using RNN encoder–decoder for statistical machine translation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing*, pp. 1724–1734. Association for Computational Linguistics, 2014. doi: 10.3115/v1/D14-1179. URL https://aclanthology.org/D14-1179/.

Giacomo Curzi, Dario Modenini, and Paolo Tortora. Two-line-element propagation improvement and uncertainty estimation using recurrent neural networks. *CEAS Space Journal*, 14:197–204, 2022. doi: 10.1007/s12567-021-00375-3. URL https://link.springer.com/article/10.1007/s12567-021-00375-3.

Abhimanyu Das, Weihao Kong, Rajat Sen, and Yichen Zhou. A decoder-only foundation model for time-series forecasting. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 10148–10167. PMLR, 2024. URL https://proceedings.mlr.press/v235/das24c.html.

European Space Agency. Space environment statistics. ESA Space Debris User Portal, 2026. URL https://sdup.esoc.esa.int/discosweb/statistics/. Last update: 31 July 2026.

Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural Computation*, 9(8): 1735–1780, 1997. doi: 10.1162/neco.1997.9.8.1735. URL https://doi.org/10.1162/neco.1997.9.8.1735.

Ameya D. Jagtap and George Em. Karniadakis. Extended physics-informed neural networks (XPINNs): A generalized space-time domain decomposition based deep learning framework for nonlinear partial differential equations. *Communications in Computational Physics*, 28(5): 2002–2041, 2020. doi: 10.4208/cicp.OA-2020-0164. URL https://doi.org/10.4208/cicp.OA-2020-0164.

Sajjad Kazemi, Nasser L. Azad, K. Andrea Scott, Haroon B. Oqab, and George B. Dietrich. Orbit determination for space situational awareness: A survey. *Acta Astronautica*, 222:272–295, 2024. doi: 10.1016/j.actaastro.2024.06.015.

Yong Liu, Tengge Hu, Haoran Zhang, Haixu Wu, Shiyu Wang, Lintao Ma, and Mingsheng Long. iTransformer: Inverted transformers are effective for time series forecasting. In *International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=JePfAI8fah.

Piyush M. Mehta, Smriti N. Paul, Nicholas H. Crisp, Philip L. Sheridan, Christian Siemes, Günther March, and Sean Bruinsma. Satellite drag coefficient modeling for thermosphere science and mission operations. *Advances in Space Research*, 72(12):5443–5459, 2023. doi: 10.1016/j.asr.2022.05.064. URL https://doi.org/10.1016/j.asr.2022.05.064.

Yuqi Nie, Nam H. Nguyen, Phanwadee Sinthong, and Jayant Kalagnanam. A time series is worth 64 words: Long-term forecasting with transformers. In *International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=Jbdc0vTOcol.

Jaideep Pathak, Alexander Wikner, Rebeckah Fussell, Sarthak Chandra, Brian R. Hunt, Michelle Girvan, and Edward Ott. Hybrid forecasting of chaotic processes: Using machine learning in conjunction with a knowledge-based model. *Chaos*, 28(4):041101, 2018. doi: 10.1063/1.5028373. URL https://arxiv.org/abs/1803.04779.

Maziar Raissi, Paris Perdikaris, and George Em Karniadakis. Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. *Journal of Computational Physics*, 378:686–707, 2019. doi: 10.1016/j.jcp.2018.10.045. URL https://maziarraissi.github.io/PINNs/.

David A. Vallado, Paul Crawford, Richard Hujsak, and T. S. Kelso. Revisiting Spacetrack Report #3. In *AIAA/AAS Astrodynamics Specialist Conference and Exhibit*, 2006. URL https://celestrak.org/publications/AIAA/2006-6753/. AIAA 2006-6753.

Sifan Wang, Yujun Teng, and Paris Perdikaris. Understanding and mitigating gradient pathologies in physics-informed neural networks, 2020. URL https://arxiv.org/abs/2001.04536.

Haixu Wu, Tengge Hu, Yong Liu, Hang Zhou, Jianmin Wang, and Mingsheng Long. TimesNet: Temporal 2D-variation modeling for general time series analysis. In *International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=ju_Uqw384Oq.

Yue Yang, Zhiqiang Wu, Saiyu Qi, and Fan Ma. Velocity-coupled representation refinement for satellite orbit prediction, 2026. URL https://arxiv.org/abs/2608.23728. Preprint.

Yuan Yin, Vincent Le Guen, Jérémie Donà, Emmanuel de Bézenac, Ibrahim Ayed, Nicolas Thome, and Patrick Gallinari. Augmenting physical models with deep networks for complex dynamics forecasting. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=kmG8vRXTFv.

Ailing Zeng, Muxi Chen, Lei Zhang, and Qiang Xu. Are transformers effective for time series forecasting? In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, pp. 11121–11128, 2023. doi: 10.1609/aaai.v37i9.26317. URL https://arxiv.org/abs/2205.13504.

Donghao Zhang, Siyuan Liang, Boyuan Yang, Tao Qi, Shangguang Wang, and Qing Li. AutoOrbit: Physics-informed satellite orbit prediction. In *Proceedings of the 32nd ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 2026. doi: 10.1145/3770855.3818960. URL https://dl.acm.org/doi/10.1145/3770855.3818960.

Haoyi Zhou, Shanghang Zhang, Jieqi Peng, Shuai Zhang, Jianxin Li, Hui Xiong, and Wancai Zhang. Informer: Beyond efficient transformer for long sequence time-series forecasting. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pp. 11106–11115, 2021. doi: 10.1609/aaai.v35i12.17325. URL https://arxiv.org/abs/2012.07436.

A Appendix

A.1 Dataset and Forecasting Settings

A.1.1 Trajectory Dataset and Forecast Windows

We evaluate history-to-future forecasting on 2,000 object trajectories, partitioned into 1,600 training, 200 validation, and 200 test objects. Each trajectory spans 180 days at a 60-second cadence; the selected intervals range from January 2006 to February 2025. Trajectories must have finite states at the specified cadence and satisfy $6500 \leq \|\mathbf{r}\|_2 \leq 8500$ km and $6 \leq \|\mathbf{v}\|_2 \leq 9$ km/s. These state-level criteria complement the TLE filters below. Table 5 gives the population by registry category, and

Table 6 specifies its numerical representation and sampling settings. Stored position and velocity are converted from meters and meters per second to km and km/s for modeling.

One forecasting example has $L = 2048$ history states and $H = 2880$ future states, corresponding to approximately 34.1 and 48 hours. Every object has 52 non-overlapping 4,928-state slots. Training draws one slot per object and epoch using a deterministic affine permutation, visiting 50 distinct slots across 50 epochs. Validation and test use slots $0, 13, 26, 38, 51$, giving five windows per object and 1,000 windows per split. Objects, rather than time intervals or launch groups, define the train/validation/test partition.

Table 5: Registry categories in the 2,000-object dataset. Unknown objects remain unclassified and are included in all overall metrics.

Category	Total	Train	Validation	Test
Satellite (PAY)	283	214	12	57
Rocket body (R/B)	157	126	9	22
Debris (DEB)	803	622	118	63
Unknown	757	638	61	58
Total	2,000	1,600	200	200

Table 6: Trajectory representation and forecasting-window settings.

Field	Value
State representation	TEME position–velocity six-state
Stored arrays	2,000 float64 arrays, each $[259{,}200, 6]$
Stored units / cadence	m, m/s / 60 s
Model units	km, km/s
Object roles	1,600 train / 200 validation / 200 test
History / forecast	2,048 / 2,880 samples
Training slots	One per object and epoch; 50 distinct slots
Evaluation slots	$0, 13, 26, 38, 51$; 1,000 windows per split

A.1.2 Source Records and Object Selection

The preprocessing pipeline takes public two-line element (TLE) records grouped by NORAD catalog number and a Satellite Catalog (SATCAT)-compatible object registry. We retain TLE epochs from 2006-01-01 UTC onward and construct regularly sampled trajectories through the filtering and reconstruction procedure below. Section A.1.5 describes reconstruction timing and source metadata.

The initial on-orbit screen excludes an object if its registry decay date is populated, its latest-TLE perigee altitude is below 120 km, or its latest TLE epoch precedes the construction reference time by more than 14 days. After record filtering and reconstruction, we select trajectories using the LEO selection criteria: orbital period 88–128 minutes, perigee altitude 200–2,000 km, and apogee altitude at most 2,000 km.

A.1.3 Filtering and Segmentation

We filter redundant records and isolate temporal discontinuities before reconstructing each trajectory. Records are sorted by TLE epoch, and the first thinning pass retains the newer record whenever two retained epochs are separated by less than half of the newer record's orbital period, $\Delta t < 0.5(86400/n)$, with mean motion n in revolutions per day. This rule also resolves equal-epoch duplicates in favor of the later record in the ordered input. Records before 2006-01-01 UTC are then discarded.

Long-gap thresholds are estimated independently for each object. Here MAD denotes the median absolute deviation. Let $g_1, \ldots, g_m$ be adjacent epoch gaps and let $G_{0.9}$ contain gaps no larger than their 90th percentile. The equal-width histogram uses

$$w_g = \mathrm{median}(G_{0.9}) + 3\,\mathrm{MAD}(G_{0.9}). \tag{12}$$

The first empty bin has left edge $T_g = kw_g$; a segment boundary is introduced when an adjacent gap is strictly greater than T_g. If no empty bin occurs, $T_g = \infty$ and the object remains one segment.

Within each segment, an N-record cubic fit predicts mean motion. For the candidate at epoch $i+1$, the screen uses the normalized absolute discrepancy $T_A = |n_{i+1} - n_{\mathrm{reg},i+1}|/|n_{\mathrm{reg},i}|$ and its ratio T_R to the fitted local mean-motion change. A candidate triggers when $T_A > A_{\mathrm{tol}}$ and $T_R > R_{\mathrm{tol}}$. The next record then distinguishes a persistent change from an isolated jump: continuation within A_{tol} classifies a persistent event and retains the records, whereas a second large change removes the isolated middle record. Table 7 lists the screening parameters. PAY, R/B, DEB, and UNK denote payload, rocket body, debris, and unknown registry categories, respectively.

Table 7: Mean-motion screening parameters $(N, A_{\mathrm{tol}}, R_{\mathrm{tol}})$ by registry type and orbital domain.

Registry type	Orbital domain	$(N, A_{\mathrm{tol}}, R_{\mathrm{tol}})$
PAY	LEO	$(5, 1.2\times10^{-5}, 1.8)$
PAY	VLEO	$(10, 2.0\times10^{-5}, 1.8)$
PAY	MEO/GEO/HEO	$(10, 1.6\times10^{-5}, 1.8)$
R/B or UNK	Any	$(10, 1.0\times10^{-5}, 1.8)$
DEB	LEO	$(15, 1.2\times10^{-5}, 1.28)$
DEB	Other	$(10, 1.6\times10^{-5}, 1.8)$
Missing metadata	Fallback	$(10, 1.39\times10^{-5}, 1.28)$

Two robust screens follow. Perigee radius $r_p = (1-e)a$ is detrended by a centered median and screened in two MAD passes: $(N_{\mathrm{TLE}}, N_{\mathrm{MAD}}) = (5, 1799)$ and $(10, 2139)$. Here N_{TLE} sets the detrending half-width $\lfloor N_{\mathrm{TLE}}/2 \rfloor$ in record indices, excluding the central record; N_{MAD} is the nominal centered rolling-MAD window length in records. The global and local MAD multipliers are $(3.5, 24)$ for PAY/VLEO and R/B or DEB in MEO/HEO, $(2.5, 18)$ for UNK, and $(3.5, 18)$ otherwise. Inclination is screened once with $(5, 1799)$ and multiplier 2.98 for PAY/LEO, 2.5 for UNK, and 3.5 for the other type/domain combinations. Finally, records with $B^* < 0$ are removed.

A.1.4 Nearest-Record Propagation

At each requested time t, we select the retained TLE with the latest epoch at or before t and propagate that single record using SGP4. Let $\mathcal{S}_i(t)$ index the retained records in the trajectory segment containing t, and let $t_{i,k}$ denote the epoch of record $\mathcal{E}_{i,k}$ for object i. The selected index is

$$k_i^*(t) = \underset{\substack{k \in \mathcal{S}_i(t) \\ t_{i,k} \le t}}{\arg\max}\, t_{i,k}. \tag{13}$$

Thus, a record becomes eligible at its epoch and remains selected until a newer retained record becomes eligible within the same segment. Equal-epoch duplicates are resolved by the thinning rule above.

The sampled position–velocity state is

$$\mathbf{x}_i(t) = \mathcal{P}\big(t; \mathcal{E}_{i,k_i^*(t)}\big) = \begin{bmatrix} \mathbf{r}_i(t) \\ \mathbf{v}_i(t) \end{bmatrix}, \tag{14}$$

where $\mathcal{P}$ is SGP4 propagation using `sgp4.api.Satrec` with UTC Julian dates. It returns TEME position and velocity in km and km/s; both components come from the same selected record. We apply this rule independently at every point of the 60-second grid. A valid state requires an eligible past record and a successful, finite SGP4 evaluation. The trajectory follows one propagated arc between record updates and can have state discontinuities when the selected record changes.

A.1.5 Reconstruction Timing and Source Metadata

Forecasting windows are extracted from trajectories constructed once offline and shared across forecast origins. Nearest-record propagation constrains the selected TLE epoch to be at or before

each reconstructed state time. The preceding record filtering remains an offline procedure: centered robust screens and the next-record mean-motion check use neighboring records. Real-time catalog availability is unmeasured because TLE publication timestamps are unavailable.

Record selection is relative to each sample time, so future target states can use TLEs with epochs after the forecast origin but at or before their own timestamps. Encoding, alignment, and future-reference generation use the supplied state history; future target states enter training loss and evaluation.

Evaluation uses the 2,000-object manifest, object partition, and window indices. The retained construction metadata contain an incomplete source-to-trajectory mapping and lack a provider snapshot identifier, release-specific filter counts, and the construction-time SGP4 version. These gaps limit regeneration from raw records and verification of the selected source TLE for each reconstructed state.

A.2 Dynamics Reference and Residual Model

A.2.1 Global Dynamics Reference

The Global Dynamics Reference (GDR) supplies a common orbital baseline for historical discrepancies and future predictions. For $\rho = \|\mathbf{r}\|_2$, its acceleration is

$$\mathbf{a}_{\mathrm{dyn}}(\mathbf{r}) = -\frac{\mu}{\rho^3}\mathbf{r} + \frac{3J_2\mu R_e^2}{2\rho^5}\begin{bmatrix} x(5z^2/\rho^2 - 1) \\ y(5z^2/\rho^2 - 1) \\ z(5z^2/\rho^2 - 3) \end{bmatrix}. \tag{15}$$

The constants are $\mu = 398600.8\ \mathrm{km}^3/\mathrm{s}^2$, $R_e = 6378.135$ km, and $J_2 = 0.001082616$. The terminal historical state is the integration anchor, $\mathbf{c}_L = \mathbf{x}_L$. Fixed-step RK4 with a 30-second internal step propagates backward over the history and forward over the forecast grid.

A.2.2 Local Dynamics Alignment

Local Dynamics Alignment (LDA) estimates phase drift from the history and adjusts the future reference before neural correction. We express the historical phase discrepancy in the right-handed frame of the raw GDR:

$$\mathbf{e}_R = \frac{\mathbf{r}^c}{\|\mathbf{r}^c\|_2}, \qquad \mathbf{e}_N = \frac{\mathbf{r}^c \times \mathbf{v}^c}{\|\mathbf{r}^c \times \mathbf{v}^c\|_2}, \qquad \mathbf{e}_T = \mathbf{e}_N \times \mathbf{e}_R. \tag{16}$$

We unwrap the in-plane phase discrepancy ϕ_j of the measured historical position and estimate its slope over the last $W = 1280$ samples, with offsets $q_j = j - L$. With $\bar{q} = W^{-1}\sum_j q_j$ and $\bar{\phi} = W^{-1}\sum_j \phi_j$ over this interval, the fitted slope and applied future angle are

$$\widehat{\beta} = \frac{\sum_{j=L-W+1}^{L}(q_j - \bar{q})(\phi_j - \bar{\phi})}{\sum_{j=L-W+1}^{L}(q_j - \bar{q})^2}, \qquad \Delta\phi_{L+h} = \phi_L + h\widehat{\beta}. \tag{17}$$

Here h is measured in one-minute samples, $\widehat{\beta}$ has units rad/sample, and ϕ_L is the measured terminal phase. Since the raw reference satisfies $\mathbf{c}_L = \mathbf{x}_L$, this phase is zero modulo 2π, up to numerical precision. Thus the applied rotation is equivalent to $h\widehat{\beta}$; unwrapping can change the stored terminal phase by an integer multiple of 2π without changing the rotation.

For unit axis $\mathbf{n}$, Rodrigues rotation is

$$\mathcal{R}(\mathbf{n}, \psi)\mathbf{u} = \mathbf{u}\cos\psi + (\mathbf{n} \times \mathbf{u})\sin\psi + \mathbf{n}(\mathbf{n}^\top \mathbf{u})(1 - \cos\psi). \tag{18}$$

Position and velocity are rotated separately about the time-dependent raw-GDR normal $\mathbf{e}_{N,L+h}$. No derivative or transport correction is added to velocity.

A.2.3 Dynamics-Conditioned Residual Forecasting

The residual model combines the history of reference discrepancies with the aligned orbital state at each future lead. Let s_r and s_v be the median historical radius and speed of a window, and let

$D = \mathrm{diag}(s_r, s_r, s_r, s_v, s_v, s_v)$. Historical inputs use the raw-reference residual, while future queries and reconstruction use the aligned reference:

$$\mathbf{d}^{\mathrm{hist}} = \mathbf{x}_{1:L} - \mathbf{c}^{\mathrm{raw}}_{1:L}, \qquad \mathbf{u}^{\mathrm{hist}} = \mathrm{diag}(\boldsymbol{\sigma})^{-1} D^{-1} \mathbf{d}^{\mathrm{hist}},$$
$$\boldsymbol{\xi}_h = \left[D^{-1} \mathbf{c}^{\mathrm{align}}_{L+h}, \tau_h, \sin(2\pi\tau_h), \cos(2\pi\tau_h) \right], \qquad \widehat{\mathbf{x}}_{L+h} = \mathbf{c}^{\mathrm{align}}_{L+h} + D\,\mathrm{diag}(\mathbf{B})\mathbf{y}_h, \tag{19}$$

where $\tau_h = h/H$, $\boldsymbol{\xi}_h \in \mathbb{R}^9$ contains the raw query features, and $\mathbf{q}_h = \mathcal{Q}_\theta(\boldsymbol{\xi}_h) \in \mathbb{R}^{96}$ is their learned embedding. The head predicts an unconstrained $\mathbf{y}_h \in \mathbb{R}^6$ from the pooled history and $\mathbf{q}_h$. The explicit reference notation corresponds to $\mathbf{c}^{\mathrm{raw}}_j = \mathbf{c}_j$ and $\mathbf{c}^{\mathrm{align}}_{L+h} = \mathbf{b}^{+}_{L+h}$ in the main text. The head output $\mathbf{y}_h$ denotes the normalized correction $\widehat{\widetilde{\mathbf{d}}}_{L+h}$, and the reconstructed forecast $\widehat{\mathbf{x}}_{L+h}$ denotes $\widehat{\mathbf{y}}_{L+h}$.

Input and output scaling. The encoder scales historical raw-reference discrepancies by $\boldsymbol{\sigma}$, while the prediction head scales future aligned-reference corrections by $\mathbf{B}$. Both scales are estimated from training objects and then frozen; the window scale D uses the observed history. The corresponding target in output coordinates is

$$\mathbf{y}^*_h = \mathrm{diag}(\mathbf{B})^{-1} D^{-1} \big(\mathbf{x}_{L+h} - \mathbf{c}^{\mathrm{align}}_{L+h} \big), \qquad \widehat{\mathbf{x}}_{L+h} - \mathbf{x}_{L+h} = D\,\mathrm{diag}(\mathbf{B})(\mathbf{y}_h - \mathbf{y}^*_h). \tag{20}$$

The network learns a mapping between these input and output coordinates. Loss and reconstruction use the inverse output normalization $\mathcal{N}^{-1}_{\mathrm{out}}(\mathbf{y}) = D\,\mathrm{diag}(\mathbf{B})\mathbf{y}$. The inverse input transform is $\mathcal{N}^{-1}_{d}(\mathbf{u}) = D\,\mathrm{diag}(\boldsymbol{\sigma})\mathbf{u}$.

Training-set scale estimation. We estimate the input scales as standard deviations of $D^{-1}\mathbf{d}^{\mathrm{hist}}$ over the training scaling windows; the encoder divides by these scales without subtracting a residual mean. Their frozen values are

$$\begin{aligned} \boldsymbol{\sigma} = (&0.0011322433, 0.0011513765, 0.0014482787, \\ &0.0011446613, 0.0011641523, 0.0014648231). \end{aligned} \tag{21}$$

For output channel k, one object-balanced non-overlapping training window from each of the 1,600 training objects supplies $z_k = (x^{\mathrm{future}}_k - c^{\mathrm{align}}_k)/D_{kk}$, without centering. The empirical scale is

$$B_k = \max\left\{ 1.1 Q_{0.995}(|z_k|), 1.05 \max|z_k|, 10^{-8} \right\}, \tag{22}$$

with

$$\begin{aligned} \mathbf{B} = (&0.0126868915, 0.0322591513, 0.0307293441, \\ &0.0127822878, 0.0318624191, 0.0309926290). \end{aligned} \tag{23}$$

The scale $\mathbf{B}$ rescales the unconstrained output correction.

Historical Residual Encoding and Reference-Conditioned Prediction. We encode joint position–velocity residual patches with a Transformer and mean-pool the encoded sequence into a history summary. A shared prediction head combines this summary with the embedded future-reference query to produce one correction per lead. All 2,880 corrections are predicted in parallel, without autoregressive feedback or cross-attention decoding. Table 9 gives the complete layer specification.

A.3 Experimental Configuration

Table 8 summarizes the training and runtime settings; Table 9 specifies the network dimensions.

Initialization. Learned positions use truncated-normal initialization with standard deviation 0.02. Transformer attention and linear weights use Xavier uniform initialization with zero biases, and the final output layer uses a normal distribution with standard deviation 10^{-3} and zero bias. Other modules use PyTorch defaults.

Table 8: DynaBR training and runtime configuration for the reported model.

Field	Value
History / horizon	2,048 / 2,880 samples
Sampling cadence	60 s
Reference integration	WGS72-J_2, RK4, 30 s step
Alignment interval	Last 1,280 history samples
Model	Patch 64; width 96; 4 heads; 2 layers; FFN 384
Optimizer	AdamW, lr 5×10^{-4}, weight decay 10^{-4}
Batch / schedule	Effective batch 16; 50 epochs; validate every 5; no scheduler
Seed	17
Selection	Validation position mean, position p95, velocity mean, earliest epoch
Environment	Python 3.12.13; PyTorch 2.11.0+cu130; NumPy 2.4.4; RTX 5090

Table 9: DynaBR network specification. The model has 350,982 trainable parameters. B denotes batch size; the bold vector $\mathbf{B}$ in Equation (22) is the output scale.

Component	Specification
History patch embedding	Input $[B, 2048, 6]$; 32 non-overlapping length-64 patches; flatten to $[B, 32, 384]$ and project to $[B, 32, 96]$
Position encoding	Learned absolute positions, shape $[1, 32, 96]$
History encoder	Two pre-norm Transformer layers; four heads; feed-forward width 384; GELU; zero dropout
History pooling	Final LayerNorm, then mean pooling to $[B, 96]$
Future query embedding	Input $[B, 2880, 9]$; $9 \to 96 \to 96$ GELU MLP; output $[B, 2880, 96]$
Correction head	Concatenate each query with the broadcast history summary to $[B, 2880, 192]$; LayerNorm and a $192 \to 384 \to 6$ GELU MLP
Arithmetic	Float64 reference integration, residual subtraction, output restoration and final addition; float32 neural computation

A.3.1 Training Objectives

We train DynaBR on physical position and velocity errors so that the objective acts on the reconstructed trajectory. For batch element b and horizon h, the loss is

$$\mathcal{L} = \frac{1}{BH} \sum_{b=1}^{B} \sum_{h=1}^{H} \left[\rho_{0.1}\left(\frac{\|\widehat{\mathbf{r}}_{b,h} - \mathbf{r}_{b,h}\|_2}{10 \text{ km}} \right) + \rho_{0.1}\left(\frac{\|\widehat{\mathbf{v}}_{b,h} - \mathbf{v}_{b,h}\|_2}{0.01 \text{ km/s}} \right) \right], \tag{24}$$

where the PyTorch smooth-L_1 convention is

$$\rho_\beta(u) = \begin{cases} u^2/(2\beta), & |u| < \beta, \\ |u| - \beta/2, & |u| \geq \beta. \end{cases} \tag{25}$$

The loss averages the position- and velocity-norm penalties over batch elements and forecast steps.

Direct learned baselines minimize six-channel MSE after applying a shared $(x - \boldsymbol{\mu})/\boldsymbol{\sigma}_x$ normalizer estimated only from training-history states across the 50-epoch window schedule. They share DynaBR's training-window schedule, seed, optimizer type and weight decay, and validation-only lexicographic selection rule (Table 8). Model-specific learning rates and microbatch sizes are given in Table 10.

A.3.2 Baseline Configurations

Table 10 specifies the learned, dynamics-only, and pretrained forecasters. Perturbed Orbital Dynamics (PertDyn) denotes raw J_2 propagation without LDA. The aligned-reference-only estimator adds LDA to this reference and omits the neural correction.

Table 10: Configurations of the evaluated forecasting adapters. Trainable direct baselines generate the complete 2,880-step six-channel forecast in one call unless noted otherwise.

Method	Model configuration	Training or forecast configuration
DLinear	Moving average 25; shared six-channel head	Microbatch 16; lr 10^{-3}
TCN	Width 64; kernel 3; dilations $1, \ldots, 512$; dropout 0.1	Ten causal residual blocks; microbatch 8; lr 5×10^{-4}
LSTM / GRU	Two unidirectional layers; width 256; dropout 0.1	Direct horizon head; microbatch 4 / 8; lr 5×10^{-4}
PatchTST	Patch 64, stride 32; width 96; 4 heads; 3 layers; FFN 384; dropout 0.1	Affine RevIN; shared head; microbatch 8; lr 10^{-4}
iTransformer	Width 96; 4 heads; 3 layers; FFN 384; dropout 0.1	Built-in normalization; microbatch 16; lr 10^{-4}
Informer	Width 96; 2 encoder / 1 decoder layers; FFN 384; factor 5; distillation	Label length 512; zero future placeholders; microbatch 2; lr 10^{-4}
TimesNet	Width 64; FFN 128; 2 layers; top-$k = 5$; 6 kernels; dropout 0.1	Microbatch 1; lr 10^{-4}
KepDyn	Universal-variable two-body propagation	$\mu = 398600.8$ km^3/s^2
PertDyn	Raw GDR from the terminal history state	Constants and integration in Equation (15); no LDA
TimesFM-3	`google/timesfm-3.0-pytorch`, version 3.0.0	Median; native stitching; iterative CPM RevIN; `use_znorm=False`
Chronos-2	`amazon/chronos-2`, version 2.3.2	Median; recursive $1024 + 1024 + 832$; max context 8192; cross-learning off
DynaBR	Configuration in Table 8	Physical smooth-L_1 objective in Equation (24)

A.4 Evaluation Settings and Metrics

We use validation objects for forecast-prefix, reference-alignment, component, and conjunction analyses, and the test partition in Section A.1 for overall and target-type comparisons. Learned models use one training seed. After validation-based model selection, we compute supplementary descriptive test summaries for the PertDyn control, cross-object standard deviations, and category breakdown using the fixed models.

Pointwise and object-balanced errors. We report the magnitude of physical forecast errors and give each object equal weight in the mean, irrespective of the number of its windows. For object o, window w, and forecast step h, define

$$e^r_{o,w,h} = \|\widehat{\mathbf{r}}_{o,w,h} - \mathbf{r}_{o,w,h}\|_2, \qquad e^v_{o,w,h} = \|\widehat{\mathbf{v}}_{o,w,h} - \mathbf{v}_{o,w,h}\|_2. \tag{26}$$

For the cumulative prefix $h = 1, \ldots, K$,

$$\bar{e}^r_o(K) = \frac{1}{W_o K} \sum_{w=1}^{W_o} \sum_{h=1}^{K} e^r_{o,w,h}, \qquad \bar{e}^r(K) = \frac{1}{O} \sum_{o=1}^{O} \bar{e}^r_o(K), \tag{27}$$

where O is the number of objects and W_o is the number of windows for object o. Velocity metrics use the same aggregation. The first forecast step, $h = 1$, is one sampling interval after the last historical state.

Tail error and cross-object dispersion. The pooled p95 is the NumPy 0.95 quantile over all pointwise norm errors in the prefix. Its sample count is

$$N_{\text{pool}}(K) = \sum_{o=1}^{O} W_o K. \tag{28}$$

The reported standard deviation is

$$s_r(K) = \left[\frac{1}{O} \sum_{o=1}^{O} (\bar{e}^r_o(K) - \bar{e}^r(K))^2 \right]^{1/2}, \tag{29}$$

with $\mathrm{ddof} = 0$. This is the population standard deviation of object-mean errors.

A cumulative-prefix mean averages all leads through K, whereas an endpoint error uses only $h = K$. The prefix tables report cumulative metrics, and the case study uses the endpoint $h = 2880$.

Uncertainty estimates. Object-bootstrap intervals describe variation across objects, with all five windows retained whenever an object is sampled. Intervals in the prefix comparison use 10,000 paired object-bootstrap replicates with percentile endpoints. Paired method differences use the same resampled object indices. Case-study intervals use the analogous object bootstrap with 1,000 replicates.

A.5 Supplementary Results

A.5.1 Validation Forecast Prefixes

We examine how the benefit of residual forecasting changes with lead time. Each model produces one 2,880-step forecast, from which we evaluate cumulative prefixes using Equation (27). Tables 11 and 12 report the position and velocity means; Table 13 reports the one-hour contrasts with PertDyn.

Table 11: Validation position object mean (km) over cumulative prefixes of one fixed 2,880-step forecast; one seed for DynaBR and TimesNet.

Method	1 h	6 h	12 h	24 h	48 h
KepDyn	16.707	109.199	217.706	433.527	863.338
PertDyn	**0.328**	2.168	4.047	7.634	14.937
TimesNet	238.975	207.118	191.532	188.366	238.060
TimesFM-3	20.255	54.350	67.212	94.710	338.347
DynaBR	0.395	**0.766**	**0.945**	**1.148**	**1.759**

DynaBR has lower position and velocity means than PertDyn from 6 through 48 hours. At one hour, its position error is higher; the paired 95% interval for the velocity difference includes zero, and the interval for the velocity ratio includes one. TimesNet's means decrease from 1 to 24 hours before increasing at 48 hours.

Table 12: Validation velocity object mean (km/s) for the same cumulative prefixes, windows, and seed coverage as Table 11.

Method	1 h	6 h	12 h	24 h	48 h
KepDyn	0.018400	0.110726	0.220394	0.439028	0.874349
PertDyn	0.000354	0.002130	0.003984	0.007539	0.014777
TimesNet	0.263711	0.239635	0.227733	0.226462	0.275382
TimesFM-3	0.019275	0.059577	0.070587	0.096756	0.321291
DynaBR	**0.000329**	**0.000688**	**0.000860**	**0.001058**	**0.001645**

Table 13: Short-horizon comparison on validation objects (1-hour cumulative prefix). Velocity contrasts are paired by object and use the bootstrap in Section A.4. Dashes indicate that no interval is reported for that entry.

Quantity	Estimate	95% interval
DynaBR position mean (km)	0.394971	–
PertDyn position mean (km)	0.328306	–
Relative position-error increase	20.31%	–
Velocity difference (km/s)	-2.500×10^{-5}	$[-5.423 \times 10^{-5}, 2.154 \times 10^{-6}]$
Velocity ratio, DynaBR / PertDyn	0.929	$[0.866, 1.007]$

A.5.2 Reference Alignment and Learned Correction

To separate deterministic phase alignment from neural correction, we compare the raw reference, the aligned reference alone, and DynaBR. Table 14 reports the validation reference-only comparison and the test comparison with the learned correction.

Table 14: Position errors for reference alignment and learned correction. Panel (a) compares dynamics-only references on validation objects. Panel (b) reports test means and DynaBR's relative mean-error reduction against each reference. Position errors are in km.

(a) Validation: reference alignment Method	Position mean	Position p95
Raw reference (PertDyn)	14.937025	65.825044
Aligned reference only	2.561635	6.529060
(b) Test: learned correction relative to each reference **Method**	**Position mean**	**DynaBR reduction**
Raw reference (PertDyn)	17.6619	94.48%
Aligned reference only	2.2423	56.53%
DynaBR	0.974719	–

Alignment reduces both the mean and tail of validation position error, including a 90.08% reduction in p95. On the test partition, the learned correction further reduces the mean relative to the aligned reference alone. The comparison thus distinguishes the contribution of phase adjustment from the additional correction learned by the residual model.

A.5.3 Contributions of Framework Components

We next examine which information pathways support residual forecasting. The historical-residual and future-conditioning ablations retain the aligned reference; an additional ablation removes LDA. The validation ablations use the training-window schedule, physical loss, and checkpoint-selection rule in Section A.3, changing one component at a time:

- *Without LDA*: raw GDR states replace aligned states in future queries, correction targets, and reconstruction; historical residuals already use the raw GDR.
- *Without historical residuals*: the encoder receives raw states $D^{-1}\mathbf{x}_{1:L}$; residual subtraction and division by $\boldsymbol{\sigma}$ are removed, with architecture and dimensions unchanged.

- *Without future dynamics conditioning*: the query retains only the three lead-time features. The aligned reference remains the correction target and reconstruction base.

Table 15: Single-seed validation ablation with the physical smooth-L_1 objective. The epoch column identifies the checkpoint selected by the common validation-only criterion for each variant.

Configuration	Pos. mean (km)	Pos. p95 (km)	Vel. mean (km/s)	Epoch
DynaBR	**1.759**	**2.481**	**0.001645**	50
w/o LDA	14.533	66.218	0.014376	50
w/o Hist. Residuals	2.293	6.457	0.001908	45
w/o Future Conditioning	1.867	3.095	0.002065	40

Each intervention increases position mean, position p95, and velocity mean relative to the complete model in Table 15. Removing LDA causes the largest deterioration across these metrics. In these ablations, residual-history encoding and future-reference conditioning each improve accuracy.

A.5.4 Test Target-Type Breakdown

We evaluate whether the relative performance of the methods is consistent across target categories. Objects are grouped by matching normalized international designators to a frozen CelesTrak SATCAT-compatible snapshot. Registry codes PAY, R/B, and DEB or FRAG map to satellite, rocket body, and debris; UNKNOWN and unmatched records map to unknown/other. Of 200 test objects, 142 have a registry match and 58 do not. The overall mean is $\sum_g (n_g/200)\bar{e}_g$, where n_g and $\bar{e}_g$ are the size and mean error of group g; all four groups contribute. Tables 16 and 17 report the category means under the test protocol in Section A.4.

Table 16: Test position object mean (km) over the fixed 48-hour prefix, five windows per object, and one seed for trainable methods. The weighted mean of the four groups reproduces the overall column.

Method	Overall (200)	Satellite (57)	Rocket body (22)	Debris (63)	Unknown/other (58)
DynaBR	0.975	1.125	0.381	1.514	0.467
PertDyn	17.662	21.564	6.764	28.766	5.899
KepDyn	884.857	779.178	904.592	982.264	875.424
TimesNet	331.812	323.901	319.469	286.648	393.324
TimesFM-3	389.496	381.680	451.199	363.650	401.847

Table 17: Test velocity object mean (km/s) for the target categories in Table 16.

Method	Overall (200)	Satellite (57)	Rocket body (22)	Debris (63)	Unknown/other (58)
DynaBR	0.000900	0.001089	0.000346	0.001362	0.000422
PertDyn	0.017558	0.021362	0.006810	0.028504	0.006005
KepDyn	0.886383	0.762574	0.926117	0.984137	0.886804
TimesNet	0.384993	0.395595	0.342041	0.362986	0.414771
TimesFM-3	0.363239	0.354267	0.432859	0.336780	0.374390

DynaBR has the lowest position and velocity means among the compared methods in every reported category, including unknown/other. Category difficulty is method-dependent: debris has the largest DynaBR mean errors, whereas the largest errors of some baselines occur in other categories.

A.6 Forecast-Error Dispersion and Conjunction Sensitivity

We examine the spatial dispersion of forecast errors and its effect on a controlled conjunction calculation. The analysis first estimates an object-balanced covariance in the radial–transverse–normal (RTN) frame, then projects it into a common encounter plane to isolate sensitivity to the forecast-error distribution.

A.6.1 Empirical RTN Dispersion and Equivalent Radius

For each validation forecast defined in Section A.4, we rotate the TEME position residual into the RTN frame of the corresponding target state, giving $\mathbf{e}_{o,w,h}$. The 48-hour lead is $h = 2880$.

With $W_o = 5$, define the object first and second moments

$$\mathbf{m}_{o,h} = \frac{1}{W_o} \sum_{w} \mathbf{e}_{o,w,h}, \qquad \mathbf{Q}_{o,h} = \frac{1}{W_o} \sum_{w} \mathbf{e}_{o,w,h} \mathbf{e}_{o,w,h}^{\top}. \tag{30}$$

The object-balanced bias, centered covariance, and dispersion are

$$\boldsymbol{\mu}_h = \frac{1}{O} \sum_{o} \mathbf{m}_{o,h}, \qquad \boldsymbol{\Sigma}_h = \frac{1}{O} \sum_{o} \mathbf{Q}_{o,h} - \boldsymbol{\mu}_h \boldsymbol{\mu}_h^{\top}, \qquad S_h = \sqrt{\operatorname{tr}(\boldsymbol{\Sigma}_h)}. \tag{31}$$

After subtracting the bias outer product, S_h measures the root-mean-square norm of centered RTN error vectors under the object-balanced weighting.

An equivalent radius matches the second moment of this covariance to an isotropic three-dimensional Gaussian. At nominal 95% coverage,

$$a = \sqrt{\frac{\chi^2_{3,0.95}}{3}}\, S_{48}. \tag{32}$$

This radius summarizes the covariance scale under an isotropic-Gaussian approximation.

A.6.2 Controlled Collision-Probability Sensitivity

To examine how forecast-error covariance affects the probability integral, we hold the encounter geometry constant across methods. The nominal 48-hour geometry uses states $[6790, 0, 0, 0, 7.65, 0]$ and $[6790.4, 0.15, 0, 7.2, 8.75, 0.4]$ in km and km/s for the International Space Station (ISS) and the debris object, respectively. ISS position uncertainty is neglected. Therefore $\Delta\mathbf{r} = (0.4, 0.15, 0)$ km, $\Delta\mathbf{v} = (7.2, 1.1, 0.4)$ km/s, and $\Delta\mathbf{r}^{\top}\Delta\mathbf{v} = 3.045$ km^2/s. These states specify the nominal geometry at the 48-hour lead, at an epoch offset from closest approach, as indicated by the nonzero position–velocity dot product.

Let $\widehat{\mathbf{v}} = \Delta\mathbf{v}/\|\Delta\mathbf{v}\|$. The encounter plane is perpendicular to $\widehat{\mathbf{v}}$. Its first axis is $\mathbf{e}_1 \propto \widehat{\mathbf{v}} \times [0, 0, 1]^{\top}$ (with a fixed alternate reference axis only near parallelism), and $\mathbf{e}_2 = \widehat{\mathbf{v}} \times \mathbf{e}_1$. Each RTN covariance is rotated through the nominal secondary-state frame into TEME and projected by $P = [\mathbf{e}_1^{\top}; \mathbf{e}_2^{\top}]$:

$$\mathbf{C} = P\, R_{\mathrm{RTN}\to\mathrm{TEME}} \boldsymbol{\Sigma}_{48} R_{\mathrm{RTN}\to\mathrm{TEME}}^{\top} P^{\top}. \tag{33}$$

For hard-body radius $R = 0.2$ km and independently swept miss vector $\boldsymbol{\rho}(d) = [d, 0]^{\top}$,

$$P_c(d) = \int_{\|\mathbf{u}\|_2 \le R} \mathcal{N}(\mathbf{u}; \boldsymbol{\rho}(d), \mathbf{C})\, d\mathbf{u}. \tag{34}$$

Log-space Gauss–Legendre quadrature evaluates the integral. The crossing distance d_{crit} solves $P_c(d) = 10^{-4}$; "No crossing" means $P_{c,\max} < 10^{-4}$ over the specified sweep. Uncertainty in the dispersion and crossing distance is estimated by the object bootstrap in Section A.4.

A.6.3 Dispersion and Encounter Sensitivity Results

We compare the reduction in spatial error dispersion with the resulting change in the controlled probability curve. Figure 5 shows both quantities, Table 18 reports the 48-hour estimates, and Table 19 provides the projected covariance scales and crossing intervals.

A.7 Historical Phase Trends and Future Reference Error

We test whether historical phase drift predicts future reference discrepancy on 32 validation objects selected by a fixed hash ordering of their identifiers, using the earliest registered window per object. Each history contains 2,048 samples at a 60-second cadence; LDA fits the final 1,280 using Equation (17). Raw GDR, GDR with LDA, and a reversed-sign control applying $\phi_L - h\widehat{\beta}$ share the same WGS72–J_2 reference, terminal state, and solver. Each produces one 2,880-step forecast without neural correction. Slopes and predictions are fixed before future evaluation.

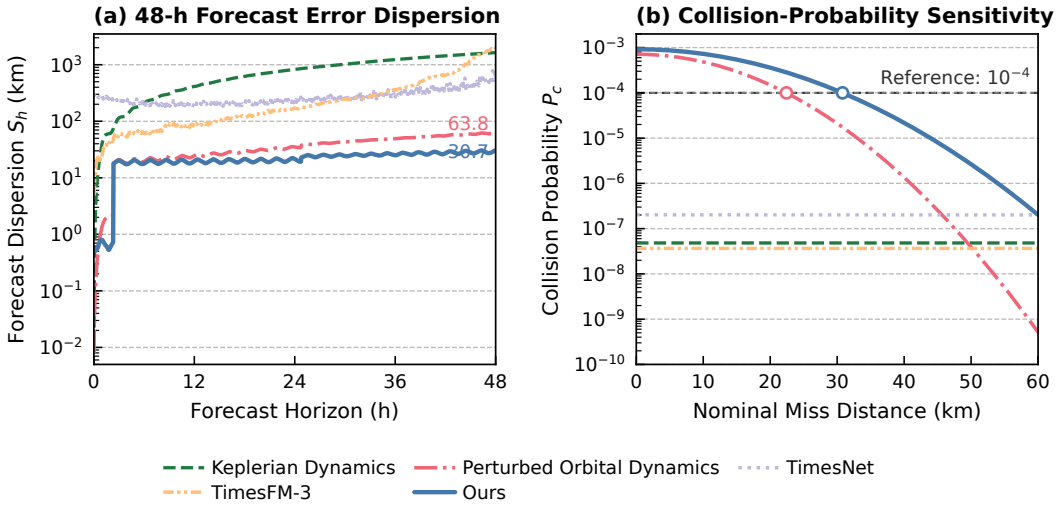


Figure 5: Validation forecast-error dispersion and controlled collision-probability sensitivity analysis. Panel (a) shows S_h over all 2,880 forecast steps for 200 objects and 1,000 windows. Panel (b) uses the common nominal geometry, a 0.2-km hard-body radius, and a $P_c = 10^{-4}$ scenario reference.

Table 18: Validation-derived 48-hour RTN dispersion and controlled encounter sensitivity. Dispersion and crossing intervals are 95% percentile intervals from 1,000 object-bootstrap replicates; all five windows remain attached to a resampled object.

Method	S_{48} (km)	95% interval (km)	$P_{c,\max}$	d_{crit} (km)	Status
KepDyn	1637.467	$[1597.144, 1676.952]$	4.83×10^{-8}	–	No crossing
PertDyn	63.777	$[46.020, 80.973]$	7.14×10^{-4}	22.421	Crossing
TimesNet	687.946	$[576.519, 807.919]$	2.02×10^{-7}	–	No crossing
TimesFM-3	2000.695	$[1864.082, 2117.333]$	3.66×10^{-8}	–	No crossing
DynaBR	30.700	$[2.955, 47.799]$	9.20×10^{-4}	30.799	Crossing

Table 19: Covariance-scale and encounter-plane diagnostics at 48 hours. Equivalent radii are approximate; crossing intervals use the bootstrap in Section A.4. All quantities are in km.

Quantity	PertDyn	DynaBR
Equivalent 95% radius a	103.0	49.6
Projected major standard deviation	63.119	30.496
Projected minor standard deviation	0.432	0.706
Crossing-distance 95% interval	$[20.886, 23.948]$	$[7.931, 34.012]$

Endpoint evaluation. We report mean position error $32^{-1}\sum_{o=1}^{32} e^{r}_{o,1,K}$ at exactly 1, 4, 8, 16, and 24 hours, with 24 hours primary and one window per object. Pointwise 95% percentile intervals use 10,000 paired object-bootstrap resamples, recomputing relative reduction from the two object means in each resample.

Table 20: Mean position error (km) at each forecast endpoint on 32 validation windows. Reduction is the percentage decrease from raw GDR to LDA, with its paired 95% interval. Wins count objects with lower endpoint error under LDA.

Horizon	Raw GDR	LDA	Reversed LDA	Reduction (%) [95% CI]	Wins
1 h	0.808	0.385	1.439	52.36 [38.76, 60.44]	26/32
4 h	2.527	0.452	5.217	82.11 [76.80, 85.46]	30/32
8 h	5.281	0.857	10.668	83.77 [73.97, 89.21]	30/32
16 h	10.689	1.408	21.195	86.83 [81.00, 90.30]	28/32
24 h	15.734	1.698	32.053	89.21 [84.06, 91.99]	29/32

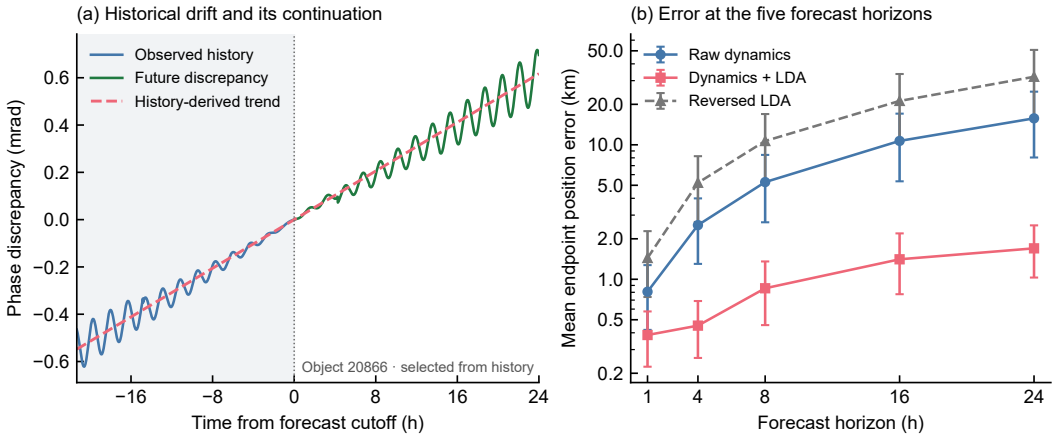


Figure 6: Historical phase discrepancy and LDA. (a) The object ranked 16th by absolute historical slope is selected before future evaluation. Shading marks the fitting history; the dashed trend is extrapolated from it. (b) Mean endpoint position errors with pointwise 95% object-bootstrap intervals on a logarithmic error axis.