STEP: Learning STructured Embeddings for Progressive Time Series

Lucas Thil
LIX, CNRS, École Polytechnique
Institut Polytechnique de Paris
IRT SystemX
Palaiseau, France
thil@lix.polytechnique.fr

Jesse Read
LIX, CNRS, École Polytechnique
Institut Polytechnique de Paris
Palaiseau, France

Rim Kaddah
IRT SystemX
Palaiseau, France

Guillaume Doquet
Safran Tech
Châteaufort, France

Abstract

We present a novel method for learning structured representations of progressive time series, that is, data capturing irreversible state transitions such as degradation or task completion. Our approach uses a self-supervised contrastive objective to learn a low-dimensional latent space in which each observation becomes a point on a manifold anchored between two fixed orthogonal prototype vectors, and a trajectory becomes a path across that manifold. From this structure we read a *latent compass*, the polar coordinates (θ, r) of the latent vector, in which θ tracks the progression of the underlying state (e.g., from healthy to failed) and r exposes residual, non-progression structure, without a supervised progression target. We evaluate the approach on industrial degradation and robotic tasks, validating three key capabilities: (1) end-state prediction, (2) health indicators that track the degradation state, and (3) phase separation. Our method is competitive with or improves over black-box counterparts while exposing the progression through a fixed set of coordinates. A simple linear regressor on top of the learned coordinates is competitive with deep architectures, quantitative evidence that the underlying state is encoded in a geometrically accessible form. Code is available at `https://github.com/LucasStill/STEP`.

1 Introduction

Time series from critical systems such as aircraft engines, batteries, or robots often capture *progressive* trajectories: a system evolves from an initial state (e.g., healthy) to a distinct end state (e.g., degraded, or task completed). In this paper, a **progressive time series** is a sequence of noisy, high-dimensional measurements x_t of an unobserved state s_t, $t = 1, \dots, T$, that follows a non-regressive transition: as t increases, s_t moves away from its initial distribution and does not return to it under nominal operation. The measurements themselves may oscillate because of sensor noise or changing operating conditions.

The canonical application is prognostics and health management (PHM) [Lee et al., 2014], where one estimates the Remaining Useful Life (RUL, the number of steps before a terminal event such as failure, $T - t$) or a State of Health (SOH) that evolves monotonically with s_t. Two difficulties arise. First, s_t is never observed, and x_t need not reflect its progressive nature directly. Second, the

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

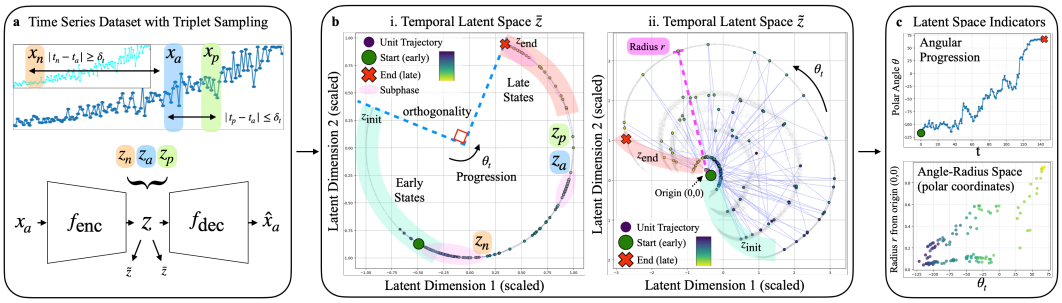


Figure 1: **Approach overview. a) Model learning:** an autoencoder is trained with triplets in which the anchor x_a and positive x_p are temporally close ($|t_p - t_a| \leq \delta_t$) and the negative x_n is temporally distant or from a different unit. **b) Latent space.** (i) The standardized view $\tilde{z}$ shows a progression between the fixed orthogonal prototypes z_{init} and z_{end}, with sub-phase clusters along the manifold. (ii) The unit-normalized view $\bar{z}$ projects trajectories onto the unit circle. **c) Latent compass:** the angle θ_t and radius r_t are deterministic functions of z_t and serve as features for downstream tasks (RUL regression, phase clustering).

only training signal is time itself: in expectation the system degrades from t to $t+1$, and the terminal state is reached at T.

Existing approaches face a trade-off. Hand-crafted health indicators (HIs) and Self-Organizing Maps (SOMs) are transparent but inflexible [Tsui et al., 2015, Bougacha et al., 2020]. Deep models predict RUL or SOH accurately, but their latent spaces are hard to inspect, which matters in safety-critical settings, and they are trained against proxy targets such as capped RUL, which can be noisy or sparse in real fleets.

We seek a latent representation z_t that (i) supports strong downstream performance, e.g., RUL prediction, and (ii) exposes progression through fixed-formula coordinates of z_t rather than through post-hoc explanations. We say *structured* rather than *interpretable* on purpose: we show that these coordinates are low-dimensional, predictive and linearly accessible, not that they are physically meaningful or mechanistically faithful health measurements.

To this end we propose **STEP** (**ST**ructured **E**mbeddings for **P**rogressive Time Series). STEP learns a low-dimensional latent space in which trajectories run between two fixed orthogonal prototype vectors $z_{\text{init}}, z_{\text{end}}$. Two readouts follow directly from z_t: the polar coordinates of z_t in the prototype plane, a *latent compass* whose angle θ_t tracks progression and whose radius r_t exposes residual structure; and a full-dimensional Mahalanobis distance from the healthy state. Both are computable at inference without oracle knowledge of T. Our core contributions are:

1. A new paradigm for learning representations of progressive time series, framing state evolution as geometric paths in a contrastively learned latent space without a supervised progression target.
2. A readout tool (that we call a 'latent compass', essentially high-level features) that exposes the progression of the underlying state through a fixed set of coordinates.
3. Demonstration of competitive performance on downstream tasks (RUL regression and health indicators) across diverse degradation benchmarks. The learned coordinates enable simple linear models to match or outperform complex architectures, evidence that the state is encoded in a geometrically accessible form.

The paper is organized by first covering related work (Sec. 2), presenting our method (Sec. 3), detailing experimental protocols (Sec. 4), and discussing results (Sec. 5).

2 Related Work

Supervised RUL prediction. A dominant strand of PHM frames prognostics as supervised RUL regression with CNN/LSTM hybrids [Zheng et al., 2017, Peng et al., 2021, Li et al., 2020], uncertainty-aware variants [De Pater and Mitici, 2022, Costa and Sánchez, 2022], or autoencoder and SSL

feature extraction [Pillai and Vadakkepat, 2021, Listou Ellefsen et al., 2019]. These models reach strong benchmark accuracy but return a scalar RUL without exposing the degradation progression. Threshold-based and parametric methods are transparent but less adaptable [Kim et al., 2017].

Survival analysis and time-to-event modeling. RUL estimation is a time-to-event problem. Survival models describe the hazard of the terminal event, from proportional hazards [Cox, 1972] to deep survival models [Katzman et al., 2018, Lee et al., 2018], and naturally exploit right-censored units and competing risks. They output an event-time distribution given covariates; STEP instead learns a representation whose geometry orders states along the trajectory, and uses complete run-to-failure trajectories for training (Sec. 6).

Health indicators (HIs). A parallel line builds scalar HIs that correlate monotonically with degradation [Rombach et al., 2024, Thil et al., 2025, Leyli-Abadi et al., 2026], often from reconstruction error or novelty scores, and evaluates them with monotonicity, trendability and prognosability [Coble, 2010]. Many target end-of-life regimes [Zhao et al., 2017, Fu et al., 2025] rather than the whole trajectory, and rely on predefined modes or labels that are often noisy or unavailable [Fink et al., 2020]. Closest in spirit, hierarchical controlled differential models separate slow degradation from fast operating fluctuations by conditioning on control inputs [Zhao and Fink, 2027].

Contrastive and topology-based representation learning. Contrastive methods learn time-series representations from positive/negative pairs [Zhang et al., 2022, Eldele et al., 2021, Sermanet et al., 2018, Baydogan and Runger, 2016], with refinements such as combined losses [Cai et al., 2025], long-range dependencies [Park et al., 2024], soft pair assignments [Liu and Chen, 2023] and SoftCLT's soft temporal contrast [Lee et al., 2024]. They do not target a globally ordered progression geometry. SOMs [Côme et al., 2010] and GTM [Bishop et al., 1998] are transparent but assume a fixed discrete grid; learned topologies [Mladenov et al., 2023] and triplet-based stage representations [Schroff et al., 2015, Fan et al., 2024] are closest to our setting.

Progressive representations beyond PHM. Task progression also provides long-horizon signals in robotics [Chen et al., 2025, Ma et al., 2023], and recent latent-planning work argues for low-dimensional bottlenecks that straighten progressive trajectories [Wang et al., 2026]. Hyperspherical, cosine-based metric learning [Li et al., 2017, Kumar BG et al., 2016] provides the geometric tools we build on.

3 STEP (STructured Embeddings for Progressive Time Series)

The central idea behind our approach is to make progression readable directly from the geometry of the latent space. Each observation x_t is encoded into a point z_t on a learned manifold (latent space), and thus an entire trajectory becomes a path across that manifold, and two polar coordinates of z_t, namely the angle θ_t and the radius r_t, together form a *latent compass*: θ_t measures how far along the progression a point is (e.g., from healthy to failed), while r_t captures residual structure beyond the progression (e.g., the concentric operating-condition rings of Fig. 2). Reading the compass thus tracks progression through a fixed, low-dimensional set of coordinates.

We learn this by combining three forces in a composite objective. A contrastive cosine triplet loss orders observations by state similarity, producing a coherent, ordered manifold such that the derived coordinates allow localization of the observed states. A prototype loss anchors the empirical start and end of every trajectory to a fixed orthogonal basis $\{z_{\text{init}}, z_{\text{end}}\}$, fixing a consistent frame across runs and datasets. A reconstruction term acts as a mild regularizer, preventing latent collapse. The full objective is

$$\mathcal{L}_{\text{total}} = \lambda_1 \mathcal{L}_{\text{recon}} + \lambda_2 \mathcal{L}_{\text{trip}} + \lambda_3 \mathcal{L}_{\text{proto}}. \tag{1}$$

with weights $\lambda_1, \lambda_2, \lambda_3$. Together, these terms make progression readable from a few fixed coordinates. The next subsections explain each component, and empirical results follow in Sec. 4 across multiple use cases.

3.1 Setting and Goal

We observe U independent progressive time series $\{(x_1^{(u)}, \ldots, x_{T_u}^{(u)})\}_{u=1}^{U}$ with $x_t^{(u)} \in \mathbb{R}^d$. Each x_t is a noisy observation of an unobserved state s_t that moves non-regressively from s_1 (e.g., healthy) to s_T (e.g., failed or task completed); a Markov-chain formalization is given in Appendix A. There is no ground-truth label for s_t: at training time only the start and end of each complete trajectory are observable surrogates for the progression boundaries. We want an encoder $z_t = f_{\text{enc}}(x_t)$ whose geometry tracks s_t rather than t: two observations in similar states should be close in z-space regardless of how much time each unit has accumulated. Sec. 5.2 tests this on held-out units.

3.2 A Contrastive Ordering Objective

We use a triplet loss with cosine similarity. For a triplet (x_a, x_p, x_n) with latent codes (z_a, z_p, z_n),

$$\mathcal{L}_{\text{trip}} = \frac{1}{N} \sum_{i=1}^{N} \max\big(0,\ \cos(z_a, z_n) - \cos(z_a, z_p) + m\big), \tag{2}$$

where m is the margin and $\cos(z_a, z_p) = \frac{z_a \cdot z_p}{\|z_a\| \|z_p\|}$. Since $\cos(z_a, z_p) = \cos(\theta_{ap})$, Eq. (2) encourages $\theta_{ap} < \theta_{an}$, ordering embeddings angularly along the trajectory.

We construct triplets using temporal proximity within a trajectory as a *sampling proxy* for state similarity. The positive x_p is drawn from the same unit as the anchor x_a, with $0 < |t_p - t_a| \leq \delta_t$; the negative x_n is drawn either from the same unit with $|t_n - t_a| \geq \delta_t$ or from a different unit entirely. Temporal proximity is a sampling heuristic: Sec. 5.2 shows that the resulting geometry tracks state rather than collapsing to elapsed time. In practice, negatives are drawn from a different coarse remaining-life band of the complete training trajectories ($>$125, 75–125, 30–75, $\leq$30 cycles; Appendix B). STEP thus uses no remaining-life regression target, but its pairing does use this coarse end-of-life information.

Per-observation encoder. f_{enc} processes each window x_t independently; time enters only through sampling. This is the modular split of most SSL methods (SimCLR [Chen et al., 2020], BYOL [Grill et al., 2020], SoftCLT [Lee et al., 2024], LeJEPA [Balestriero and LeCun, 2025]): the loss shapes the geometry of single-input representations and any sequence modeling is left to the downstream head. The same frozen encoder therefore serves several tasks, and every readout is a deterministic function of a single z_t.

We add a mean squared error term $\mathcal{L}_{\text{recon}} = \frac{1}{N} \sum_i \|x_i - f_{\text{dec}}(f_{\text{enc}}(x_i))\|_2^2$ with a deliberately small MLP decoder, so that a high-capacity decoder cannot bypass the latent structure. Alone, this term yields a weakly ordered latent (Sec. 5.4).

3.3 A Fixed Reference Frame

The triplet objective orders the manifold but leaves it free to rotate, reflect and rescale across runs, so the same observation can land at different coordinates from one training to the next, and θ and r cannot be compared across runs or datasets. We therefore fix two standard basis vectors as anchors, $z_{\text{init}} = \mathbf{e}_1$ and $z_{\text{end}} = \mathbf{e}_2$, and softly pull the first k and last k steps of each training trajectory, X_{init} and X_{end}, toward them:

$$\mathcal{L}_{\text{proto}} = \mathbb{E}_{x \in X_{\text{init}}} \left[\|f_{\text{enc}}(x) - z_{\text{init}}\|^2 \right] + \mathbb{E}_{x \in X_{\text{end}}} \left[\|f_{\text{enc}}(x) - z_{\text{end}}\|^2 \right]. \tag{3}$$

The latent space thus contains an initial-state region (around $\mathbf{e}_1$) and an absorbing terminal region (around $\mathbf{e}_2$), with trajectories navigating between them. Intermediate states are never conditioned on. The pull is a soft expectation: observations cluster *near* the anchors, leaving slack for unit-level variability such as different initial wear (the spread of the healthy zone in Fig. 1), and the radial direction is left free. Anchoring is not always needed for accuracy but matters when trajectories span heterogeneous regimes: removing it (λ_3=0) degrades FD004 from 12.10 to 19.01 RMSE but slightly improves the homogeneous FD003 (13.35 to 11.59; single runs, Appendix D.5).

Together, the contrastive ordering and the prototype-anchored basis induce a *state-based topology* over the dataset, in which each observation projects to a single *step* along the trajectory map, from start to end.

3.4 Readouts

We read two standard views of the latent, the standardized $\tilde{z} = (z - \mu)/\sigma$ and the unit-normalized $\bar{z} = z/\|z\|$, and two health indicators. The latent compass is the pair of polar coordinates of z_t in the anchor plane,

$$\theta_t = \arctan 2\big(z_t \cdot \mathbf{e}_2,\ z_t \cdot \mathbf{e}_1\big), \qquad r_t = \big\|(z_t \cdot \mathbf{e}_1,\ z_t \cdot \mathbf{e}_2)\big\|, \tag{4}$$

which uses only the two anchor dimensions It uses only the two anchor dimensions, so it is a visual summary. For $\dim(z) > 2$ the anchor plane holds only 30–54% of the latent norm. The full-dimensional readout is the Mahalanobis distance from the healthy state, $h_t = \sqrt{(z_t - \mu_0)^\top \Sigma_0^{-1} (z_t - \mu_0)}$, with (μ_0, Σ_0) estimated on the healthiest training windows (Appendix B.2). Both are computed from z_t alone at inference time.

4 Experimental Framework

Datasets. **C-MAPSS** [Saxena et al., 2008] has four turbofan run-to-failure subsets (FD001–004) with one or six operating conditions (OCs) and one or two failure modes (FMs) (Appendix Table 7); training units are complete, test units are right-censored with known RUL. **N-CMAPSS** [Arias Chao et al., 2021] simulates turbofans under real flight profiles; we follow the protocol-matched CruiseBench pipeline [Cheng and Miao, 2026] (DS02-006, 9 units, cruise mask, window 256), with the RUL cap of 34 cycles that CruiseBench prescribes for DS02-006; results at caps 80 and 125 are in Appendix Table 8. **NASA Li-ion battery aging** [Saha and Goebel, 2007]: the four classic cells B0005/6/7/18, evaluated leave-one-cell-out (Appendix C). For qualitative analysis we use NVIDIA GR00T's Can2Drawer [NVIDIA et al., 2025] and LeRobot [Cadene et al., 2024] manipulation episodes (joint positions), and a mouse neural-activity recording [Vinograd et al., 2024] (Appendix F). Sensor channels are min-max normalized per feature (after a per-regime z-score on FD002/FD004), with the scaler fit on training windows only, and the setting columns are not given to the encoder.

Protocol. The backbone model is trained without RUL targets and frozen, then a downstream head is fit on its latent features z. On C-MAPSS the head is a transformer (Tr.) or a linear regressor (LR.) on $\tilde{z}$, θ, r and the progression score (Appendix B.2). We report mean $\pm$ std over 5 seeds, RUL RMSE and the asymmetric PHM08 score [Saxena et al., 2008], and unit-level bootstrap 95% confidence intervals in Appendix D. HI quality is measured by *trendability*, the mean over units of the absolute Spearman correlation $|\rho(\mathrm{HI}, \mathrm{RUL})|$ (higher is better), with monotonicity and prognosability in the appendix [Coble, 2010].

Baselines. We compare STEP with two variants of the same backbone that differ only in their training objective, BaseAE (reconstruction only) and SoftCLT [Lee et al., 2024], using identical windows, features and RUL heads. For health indicators, we also build three classical HIs directly on the raw sensor features, without a trained encoder. These are the projection on the first principal component, the best-matching-unit index on a one-dimensional SOM, and a similarity indicator given by the Mahalanobis distance from the healthy state. On C-MAPSS, we further report published RUL models. On N-CMAPSS, we reproduce a supervised GRU and TSMixer [Chen et al., 2023] in the same pipeline, and add a randomly initialized encoder as a null baseline.

5 Results and Discussion

5.1 Latent Geometry

Fig. 2 compares the latent geometry of the three objectives on the same backbone. AE, trained for reconstruction only, leaves FD002 diffuse and temporally incoherent; SoftCLT forms soft clusters loosely aligned with OCs but no global ordering. STEP produces an ordered manifold between the anchors: at w=1 six rings appear, one per OC, and θ advances with degradation; at w=10 the rings merge and the progression sharpens. The 2D picture thus depends on window and dimension (FD004 merges at higher $\dim(z)$, FD003 keeps its separation), whereas RUL accuracy does not (Sec. 5.5), and the full-dimensional readout stays trendable at every dimension. Using θ and r as diagnostics

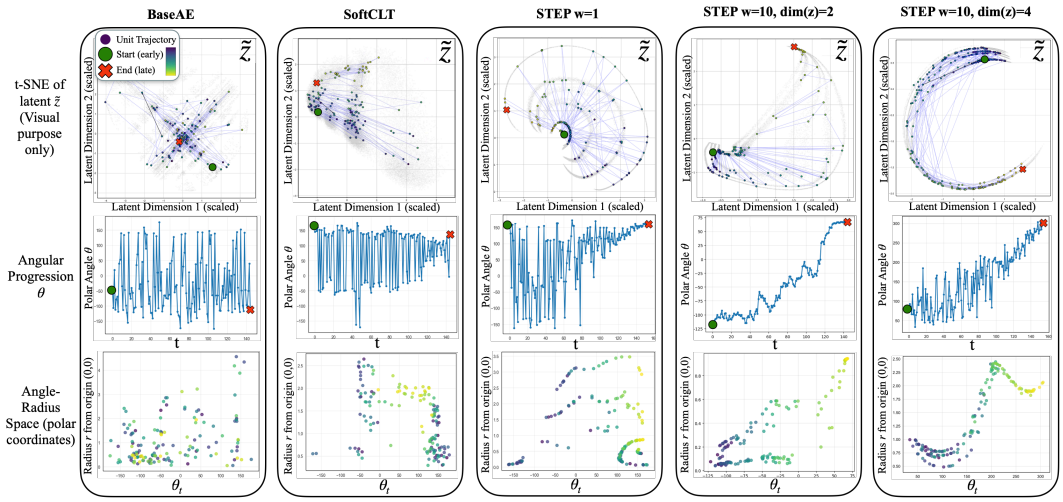
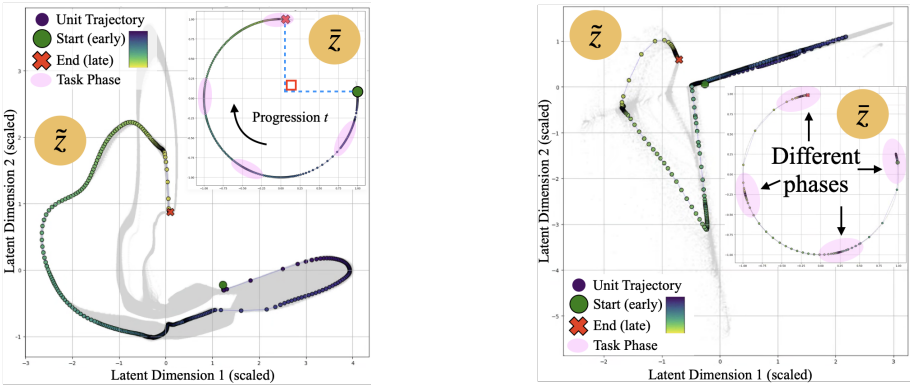


Figure 2: **Latent compass on C-MAPSS FD002** (6 operating conditions). Gray: t-SNE of the whole dataset; colored: engine 20 over time. AE shows no coherent structure and SoftCLT a limited one. STEP with window w=1 separates six concentric rings, one per operating condition, with θ advancing along the degradation. With w=10 the rings merge while the angular progression sharpens. With $\dim(z)$=4 the embedding is richer and the rings less pronounced. The rings are a dataset- and setting-dependent visual effect; quantitatively, the radius is not a reliable mode label (Sec. 5.2).



(a) GR00T [NVIDIA et al., 2025] (pick-and-place). (b) LeRobot [Cadene et al., 2024] (shelf grabbing).

Figure 3: **STEP latents on two robotics datasets** (qualitative). $\tilde{z}$ follows task progression; $\bar{z}$ groups into task phases (locating, grabbing, moving, placing) across two embodiments and tasks.

for AE and SoftCLT is motivated in Appendix B.1. On robotics (Fig. 3), $\bar{z}$ groups episodes into task phases (locating, grabbing, moving, placing) without annotation; this evidence is qualitative.

5.2 Does θ Track State Beyond Elapsed Time?

A natural concern is that θ merely encodes the clock, since triplets are built from temporal proximity. We test this on *held-out* units of the multi-condition subsets FD002 and FD004 with controls on elapsed cycles, not on t/T, which needs the oracle T and carries no information about absolute RUL.

Remaining life beyond elapsed time. If θ only encoded elapsed cycles, controlling for them would remove its relation to RUL. The partial correlation $|\rho(\theta, \mathrm{RUL} \mid t)|$ is 0.30 ($p < 0.001$; mean over FD002 and FD004) for STEP, against 0.03 for the AE angle: the AE angle is close to a clock, STEP's is not.

Table 1: **RUL RMSE on C-MAPSS** (lower is better). HI marks methods that also provide health indicators. I-GLIDE derives them from data deemed healthy, whereas STEP maps the full trajectory from start to end of life. Top: published results as reported by their authors, typically single or best runs. Bottom: the same backbone trained with different objectives, frozen, with a transformer (Tr.) or linear (LR.) head, mean $\pm$ std over 5 seeds. Bold marks the best mean per subset. SoftCLT rows use the same backbone, features and heads as STEP, scored through a latent-to-head bridge that reproduces the published Tr.STEP values to within 0.4 RMSE. PHM08 scores and all SoftCLT latent sizes are in Appendix Tables 9 and 13.

Model	**HI**	**FD001**	**FD002**	**FD003**	**FD004**
MLP [Zheng et al., 2017]		37.56	80.03	37.39	77.37
SVR [Zheng et al., 2017]		20.96	42.00	21.05	45.35
RVR [Zheng et al., 2017]		23.80	31.30	22.37	34.34
CNN [Zheng et al., 2017]		18.45	30.29	19.82	29.16
SSL [Listou Ellefsen et al., 2019]		12.56	22.73	12.10	22.66
CNN-LSTM [Peng et al., 2021]		11.17	–	9.99	–
MS-DCNN [Li et al., 2020]		11.44	19.35	11.67	22.22
VAE + RNN [Costa and Sánchez, 2022]		11.44	24.12	14.88	26.54
MLE4X+CCF [Pillai and Vadakkepat, 2021]		11.57	18.84	11.83	20.78
RVE [Costa and Sánchez, 2022]		13.42	14.92	12.51	16.37
Prob. CNN [De Pater and Mitici, 2022]		12.42	13.72	12.16	15.95
Wang et al. [Wang et al., 2025]	✓	–	–	–	15.42
I-GLIDE [Thil et al., 2025]	✓	**9.47**	16.18	**8.29**	**12.32**
Tr.AE z=32		17.15±0.18	18.13±0.21	20.72±1.41	20.76±0.68
Tr.SoftCLT z=4		15.34±1.17	21.23±0.98	14.64±1.07	22.77±1.27
Tr.SoftCLT z=32		15.50±0.61	19.73±1.25	14.87±1.06	23.09±2.08
LR.SoftCLT z=32		14.84±0.36	16.85±0.53	15.97±3.11	19.08±0.95
Tr.STEP z=2	✓	13.18±0.79	12.13±0.10	13.30±0.46	15.31±0.84
Tr.STEP z=4	✓	13.12±0.23	11.78±0.40	12.89±0.73	13.31±0.32
Tr.STEP z=32	✓	13.27±0.38	11.57±0.34	12.66±0.51	13.13±0.69
LR.STEP z=32	✓	13.09±0.42	**11.44**±0.04	12.48±0.48	13.26±0.39

Invariance to operating condition. Within matched-θ bins, a 6-way OC classifier recovers the OC from the AE angle at balanced accuracy 0.52, but from STEP's angle only at chance (0.17; chance $= 1/6$). STEP's progression axis is thus not confounded by the operating regime.

What the radius does and does not carry. The operating condition is not lost, though. It stays partly readable from the radius (balanced accuracy 0.37 on FD002 and 0.39 on FD004) and almost fully from the full latent (0.97 and 0.98). STEP thus moves operating-condition information off the angle and into the other dimensions, which is what the rings of Fig. 2 show, even if the radius alone is not a reliable mode label. The full latent also separates failure modes. Clustering the terminal embeddings with K-means splits the two failure modes of FD003 (silhouette 0.75–0.97) and FD004 (0.60–0.66) without supervision, despite a single z_{end} (Appendix Table 14).

5.3 RUL Estimation on C-MAPSS

Table 1 reports five-seed results. On the multi-condition FD002, STEP obtains the lowest mean RMSE among compared methods (11.57 $\pm$ 0.34 at z=32, against 13.72 for the best published baseline). On FD004 it is competitive (13.13 $\pm$ 0.69) but behind I-GLIDE (12.32). SoftCLT, trained with the same backbone and scored with the same heads and features, is 5-8 RMSE behind STEP on FD002 at matched latent dimension (best: 16.85 ± 0.53 with a linear head at z=32, against 11.48 ± 0.31 for STEP through the same evaluation path), and widening its latent from z=4 to z=32 does not improve results. On the single-condition subsets FD001 and FD003, STEP is behind the best supervised models, so its main value there lies in the health indicators rather than in RMSE. I-GLIDE also provides health indicators, but it learns only from data it first identifies as healthy, while STEP learns from the whole trajectory.

Replacing the transformer head with a linear regressor on the same features changes RMSE by at most 0.2 on every subset. A linear head can only succeed if the RUL-relevant state is linearly accessible in the representation. This is evidence that STEP's coordinates are predictive and geometrically

Table 2: **Health-indicator trendability** (mean per-unit $|\rho(\text{HI}, \text{RUL})|$, higher is better). **(a)** C-MAPSS, z=32, 5 seeds: same backbone trained with reconstruction only (AE) or with STEP, read with the 2D compass angle or the full-dimensional Mahalanobis readout. **(b)** NASA battery, held-out cells (leave-one-cell-out). **(c)** 2D compass versus full-dimensional readout for STEP.

(a) C-MAPSS, AE vs. STEP

	2D compass		Full (Mahalanobis)	
	AE	STEP	AE	STEP
FD001	.25±.02	.68±.16	.35±.01	.70±.01
FD002	.34±.01	.54±.14	.40±.01	.65±.03
FD003	.27±.01	.72±.08	.46±.02	.69±.02
FD004	.37±.06	.64±.02	.44±.01	.70±.02

(b) Battery

HI	Trend.
PCA	0.93
Similarity	0.97
SOM	0.76
STEP	**0.95**

(c) 2D vs. full

	2D	Full
C-MAPSS	.64	.68±.02
N-CMAPSS	.66	.93±.02

Table 3: **N-CMAPSS** (CruiseBench protocol [Cheng and Miao, 2026], window 256, 5 seeds). STEP encoders are trained with no RUL label, then frozen; a RUL head is fit on their features. Sup.: RUL is the encoder's training target. Trend.: HI trendability. Absolute RMSE is dominated by the RUL cap and is comparable only within this protocol.

Model	Sup.	RMSE	Trend.
GRU, supervised (repr.)	✓	**1.81**±0.37	–
TSMixer [Chen et al., 2023], supervised (repr.)	✓	2.41±0.07	–
STEP, CNN encoder		2.19±0.08	0.91±0.01
STEP, GRU encoder		2.30±0.06	**0.92**±0.01
STEP, LSTM encoder		2.38±0.06	0.89±0.02
STEP, PatchTST [Nie et al., 2023] encoder		2.87±0.21	0.86±0.03
Reconstruction only (null)		9.39±1.32	0.06
Random-init encoder (null)		3.44±0.53	0.14

accessible. With the same backbone, the reconstruction-only AE is 4 to 8 RMSE worse, so the gain comes from the learned representation through STEP's training objective.

5.4 Health-Indicator Quality Across Benchmarks

Health indicators on C-MAPSS. With the same backbone, reconstruction alone gives a weak indicator on C-MAPSS (trendability 0.35–0.46). Adding the STEP terms roughly doubles it on every subset (0.65–0.70, Table 2a), while also improving RUL accuracy (Table 1). STEP also beats the strongest classical indicator, the similarity HI computed directly on the sensors, by +0.16 trendability over all of C-MAPSS (95% CI $[0.15, 0.17]$). The gain is largest on the multi-condition subsets (+0.19 on FD002, +0.24 on FD004), smaller on FD001 (+0.05), and the two are tied on FD003 (Appendix Table 10).

Beyond C-MAPSS. On held-out NASA battery cells (leave-one-cell-out, fixed end of life, no capacity-derived inputs; Appendix C), STEP reaches a trendability of 0.95, competitive with the similarity indicator (0.97) and PCA (0.93) and above SOM (0.76), close to raw capacity (0.98), which defines end of life here. On N-CMAPSS (Table 3), reconstruction alone gives unusable features (RMSE 9.4, trendability 0.06), and the STEP terms turn them into strong ones on all four encoder families (RMSE 2.2–2.9, trendability 0.86–0.92): the objective is not tied to one architecture. STEP does not reach the supervised GRU (1.81) but improves on the reproduced supervised TSMixer (2.41) at the benchmark cap; at caps 80 and 125 both supervised models lead STEP by 5–12% (Appendix Table 8). Unlike the supervised regressors, STEP also provides a trendable health indicator.

2D compass versus full readout. The full-dimensional readout recovers signal that the 2D compass discards, most where health is a low-variance direction (N-CMAPSS 0.66 to 0.93); on C-MAPSS the compass already captures most of it (0.64 vs. 0.68). The compass is the visual summary, the Mahalanobis readout the quantitative one.

5.5 Ablations and Robustness

Loss terms. The triplet term is what shapes the geometry. Without it, reconstruction alone orders the latent only weakly on C-MAPSS, or sometimes is completely disorganized and can't recover the underlying structure (states are unordered) and fails on N-CMAPSS (Tables 2a and 3). The prototype term helps most when trajectories span several regimes, as on FD004, and slightly constrains the homogeneous FD003 (Sec. 3.3). On N-CMAPSS, its benefit depends on the encoder (Appendix D). We do not ablate reconstruction, since removing it risks collapsing the latent, as it serves as a regularization term in preserving the content.

Triplet sampling. Mixing negatives from the same unit and from other units works best, and drawing negatives uniformly at random works worse. The positive window matters little on homogeneous subsets but a lot on FD004, where widening it raises trendability from 0.47 to 0.63. In practice, the negative margin should be about three times the positive one, and no per-dataset tuning was needed.

Dimension and window. RUL accuracy stays stable as the latent grows from $\dim(z)$=2 to 32 (Table 1) and across window sizes. The settings that blur the rings in the 2D picture therefore cost no accuracy, and the downstream regressor benefits from a longer context window. Larger latents are harder to train, as trajectories scatter in the sparser space. The decaying prototype weight keeps training stable from $\dim(z)$=4 upward (Appendix B).

Sensor noise. To test robustness, we retrain at $\dim(z)$=16 with the sensor noise halved or multiplied by 2.5. Going from the lowest to the highest noise level, RMSE rises only moderately, from 11.34 to 13.20 on FD001 and from 14.23 to 15.12 on FD002 (single runs, Appendix Table 15). Noisy readings do not break the progressive assumption, which concerns the hidden state rather than the sensors themselves. STEP's representation is thus robust to measurement noise, not only to its own training conditions.

6 Conclusion

We have presented STEP, a method for learning structured representations of progressive time series by framing state evolution as geometric paths in a contrastively learned latent space, anchored between two fixed prototypes. Health indicators can be read directly from this geometry, as a two-dimensional latent compass for visualization and a full-dimensional readout for quantitative use. On held-out units, the angle tracks the hidden state rather than elapsed time, and it separates progression from operating conditions, which remain available in the rest of the latent. Across degradation benchmarks, STEP roughly doubles the trendability of a reconstruction-only backbone, outperforms classical indicators on C-MAPSS and matches them on battery data. These gains do not come at the cost of accuracy, since a simple linear model on STEP's coordinates is competitive with deep architectures for RUL prediction. The objective transfers across four encoder families, and on robotic tasks the same geometry recovers task phases without annotation.

Limitations. STEP is designed for progressive sequences and assumes that the underlying state does not go back. It therefore does not model reversible transitions such as maintenance, other interventions, or regime switching with recovery. Training also relies on complete run-to-failure trajectories, since both the sampler and the prototype term use their start and end, so censored units are left out, whereas survival objectives handle them natively. Finally, STEP separates operating conditions from progression but does not model how control inputs shape the degradation rate, which control-aware dynamics address [Zhao and Fink, 2027].

Future work. A natural extension is to systems that return to earlier states, for example after maintenance repairs a degraded component. More broadly, the geometry STEP learns could serve as a map for system discovery, where intermediate stages emerge as new anchors, and learning how the system moves along this map, including under control inputs, would reveal the mechanisms that drive its progression.

Acknowledgments and Disclosure of Funding

This work was carried out within the IRT SystemX of smart industrial digital twins (JNI3) project and was granted access to the HPC resources of IDRIS under the allocation 2025-AD011017210 made by GENCI. No competing interests to be declared.

References

Manuel Arias Chao, Chetan Kulkarni, Kai Goebel, and Olga Fink. Aircraft engine run-to-failure dataset under real flight conditions for prognostics and diagnostics. *Data*, 6(1):5, 2021.

Randall Balestriero and Yann LeCun. Lejepa: Provable and scalable self-supervised learning without the heuristics. *arXiv preprint arXiv:2511.08544*, 2025. doi: 10.48550/arXiv.2511.08544.

Mustafa Gokce Baydogan and George Runger. Time series representation and similarity based on local autopatterns. *Data Mining and Knowledge Discovery*, 30(2):476–509, 2016.

Christopher M. Bishop, Markus Svensén, and Christopher K. I. Williams. Gtm: The generative topographic mapping. *Neural Computation*, 10(1):215–234, 1998. doi: 10.1162/089976698300017953.

Omar Bougacha, Christophe Varnier, and Noureddine Zerhouni. A review of post-prognostics decision-making in prognostics and health management. *International Journal of Prognostics and Health Management*, 11(15):31, 2020.

Remi Cadene, Simon Alibert, Alexander Soare, Quentin Gallouedec, Adil Zouitine, Steven Palma, Pepijn Kooijmans, Michel Aractingi, Mustafa Shukor, Dana Aubakirova, Martino Russi, Francesco Capuano, Caroline Pascal, Jade Choghari, Jess Moss, and Thomas Wolf. Lerobot: State-of-the-art machine learning for real-world robotics in pytorch. `https://github.com/huggingface/lerobot`, 2024.

Huili Cai, Xiang Zhang, and Xiaofeng Liu. Semi-supervised end-to-end contrastive learning for time series classification. (arXiv:2310.08848), March 2025. doi: 10.48550/arXiv.2310.08848. URL `http://arxiv.org/abs/2310.08848`. arXiv:2310.08848 [cs].

Qianzhong Chen, Justin Yu, Mac Schwager, Pieter Abbeel, Yide Shentu, and Philipp Wu. Sarm: Stage-aware reward modeling for long horizon robot manipulation. (arXiv:2509.25358), October 2025. doi: 10.48550/arXiv.2509.25358. URL `http://arxiv.org/abs/2509.25358`. arXiv:2509.25358 [cs].

Si-An Chen, Chun-Liang Li, Nate Yoder, Sercan O. Arik, and Tomas Pfister. TSMixer: An all-MLP architecture for time series forecasting. *Transactions on Machine Learning Research*, 2023.

Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In *International Conference on Machine Learning (ICML)*, 2020.

P. Cheng and Q. Miao. CruiseBench: A real-flight-aligned N-CMAPSS benchmark for engine RUL prediction. *arXiv preprint arXiv:2607.19380*, 2026.

Jamie Baalis Coble. Merging data sources to predict remaining useful life–an automated method to identify prognostic parameters. 2010.

Etienne Côme, Marie Cottrell, Michel Verleysen, and Jérôme Lacaille. Aircraft engine health monitoring using self-organizing maps. In *10th Industrial Conference, ICDM 2010,*, volume 6171, pages 405–417. Springer, 2010.

Nahuel Costa and Luciano Sánchez. Variational encoding approach for interpretable assessment of remaining useful life estimation. *Reliability Engineering & System Safety*, 222:108353, June 2022. ISSN 09518320. doi: 10.1016/j.ress.2022.108353. URL https://linkinghub.elsevier.com/retrieve/pii/S0951832022000321.

David R. Cox. Regression models and life-tables. *Journal of the Royal Statistical Society: Series B (Methodological)*, 34(2):187–220, 1972.

Ingeborg De Pater and Mihaela Mitici. Novel metrics to evaluate probabilistic remaining useful life prognostics with applications to turbofan engines. *PHM Society European Conference*, 7(1): 96–109, June 2022. ISSN 2325-016X, 2325-016X. doi: 10.36001/phme.2022.v7i1.3320. URL https://papers.phmsociety.org/index.php/phme/article/view/3320.

Emadeldeen Eldele, Mohamed Ragab, Zhenghua Chen, Min Wu, Chee Keong Kwoh, Xiaoli Li, and Cuntai Guan. Time-series representation learning via temporal and contextual contrasting. In *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence*, pages 2352–2359. International Joint Conferences on Artificial Intelligence Organization, 2021.

Yalun Fan, Weilin Li, Liang Guo, Yaoxiang Yu, Tao Luo, and Hongli Gao. A metric learning-based health indicator for early degradation point detection of machines. In *2024 Global Reliability and Prognostics and Health Management Conference (PHM-Beijing)*, page 1–8, Beijing, China, October 2024. IEEE. ISBN 979-8-3503-5401-0. doi: 10.1109/PHM-Beijing63284.2024.10874536. URL https://ieeexplore.ieee.org/document/10874536/.

Olga Fink, Qin Wang, Markus Svensén, Pierre Dersin, Wan-Jui Lee, and Melanie Ducoffe. Potential, challenges and future directions for deep learning in prognostics and health management applications. *Engineering Applications of Artificial Intelligence*, 92:103678, 2020. ISSN 09521976. doi: 10.1016/j.engappai.2020.103678. URL https://linkinghub.elsevier.com/retrieve/pii/S0952197620301184.

Ying Fu, Ye Kwon Huh, and Kaibo Liu. Degradation modeling and prognostic analysis under unknown failure modes. *IEEE Transactions on Automation Science and Engineering*, 2025.

Jean-Bastien Grill, Florian Strub, Florent Altché, Corentin Tallec, Pierre H. Richemond, Elena Buchatskaya, Carl Doersch, Bernardo Avila Pires, Zhaohan Daniel Guo, Mohammad Gheshlaghi Azar, et al. Bootstrap your own latent – a new approach to self-supervised learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020.

Amber Hu, Henry D Smith, and Scott Linderman. Sing: Sde inference via natural gradients. *Advances in Neural Information Processing Systems*, 2025.

Jared L. Katzman, Uri Shaham, Alexander Cloninger, Jonathan Bates, Tingting Jiang, and Yuval Kluger. DeepSurv: Personalized treatment recommender system using a Cox proportional hazards deep neural network. *BMC Medical Research Methodology*, 18(1):24, 2018.

Nam-Ho Kim, Dawn An, and Joo-Ho Choi. Prognostics and health management of engineering systems. *Switzerland: Springer International Publishing*, 2017.

Vijay Kumar BG, Gustavo Carneiro, and Ian Reid. Learning local image descriptors with deep siamese and triplet convolutional networks by minimising global loss functions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5385–5394, 2016.

Changhee Lee, William R. Zame, Jinsung Yoon, and Mihaela van der Schaar. DeepHit: A deep learning approach to survival analysis with competing risks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32, 2018.

Jay Lee, Fangji Wu, Wenyu Zhao, Masoud Ghaffari, Linxia Liao, and David Siegel. Prognostics and health management design for rotary machinery systems—reviews, methodology and applications. *Mechanical Systems and Signal Processing*, 42(1–2):314–334, January 2014. ISSN 08883270. doi: 10.1016/j.ymssp.2013.06.004. URL https://linkinghub.elsevier.com/retrieve/pii/S0888327013002860.

Seunghan Lee, Taeyoung Park, and Kibok Lee. Soft contrastive learning for time series. (arXiv:2312.16424), March 2024. doi: 10.48550/arXiv.2312.16424. URL http://arxiv.org/abs/2312.16424. arXiv:2312.16424 [cs].

Milad Leyli-Abadi, Lucas Thil, Sebastien Razakarivony, Guillaume Doquet, and Jesse Read. A machine learning framework for turbofan health estimation via inverse problem formulation, 2026. URL https://arxiv.org/abs/2604.08460.

Chao Li, Xiaokong Ma, Bing Jiang, Xiangang Li, Xuewei Zhang, Xiao Liu, Ying Cao, Ajay Kannan, and Zhenyao Zhu. Deep speaker: an end-to-end neural speaker embedding system. *arXiv preprint arXiv:1705.02304*, 2017.

Han Li, Wei Zhao, Yuxi Zhang, and Enrico Zio. Remaining useful life prediction using multi-scale deep convolutional neural network. *Applied Soft Computing*, 89:106113, April 2020. ISSN 15684946. doi: 10.1016/j.asoc.2020.106113. URL https://linkinghub.elsevier.com/retrieve/pii/S1568494620300533.

André Listou Ellefsen, Emil Bjørlykhaug, Vilmar Æsøy, Sergey Ushakov, and Houxiang Zhang. Remaining useful life predictions for turbofan engine degradation using semi-supervised deep architecture. *Reliability Engineering & System Safety*, 183:240–251, March 2019. ISSN 09518320. doi: 10.1016/j.ress.2018.11.027. URL https://linkinghub.elsevier.com/retrieve/pii/S0951832018307506.

Jiexi Liu and Songcan Chen. Timesurl: Self-supervised contrastive learning for universal time series representation learning. (arXiv:2312.15709), December 2023. doi: 10.48550/arXiv.2312.15709. URL http://arxiv.org/abs/2312.15709. arXiv:2312.15709 [cs].

Yecheng Jason Ma, Vikash Kumar, Amy Zhang, Osbert Bastani, and Dinesh Jayaraman. Liv: Language-image representations and rewards for robotic control. In *International Conference on Machine Learning*, pages 23301–23320. PMLR, 2023.

Emanuil Mladenov, Miguel Martínez-García, Yu Zhang, Shaheryar Khan, and Faizan Patankar. Unsupervised flight fault propagation analysis using a variational autoencoder. In *2023 IEEE 13th International Conference on Pattern Recognition Systems (ICPRS)*, page 1–4, Guayaquil, Ecuador, 2023. IEEE. ISBN 979-8-3503-3337-4. doi: 10.1109/ICPRS58416.2023.10179004. URL https://ieeexplore.ieee.org/document/10179004/.

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.

NVIDIA, Johan Bjorck, Nikita Cherniadev Fernando Castañeda, Xingye Da, Runyu Ding, Linxi "Jim" Fan, Yu Fang, Dieter Fox, Fengyuan Hu, Spencer Huang, Joel Jang, Zhenyu Jiang, Jan Kautz, Kaushil Kundalia, Lawrence Lao, Zhiqi Li, Zongyu Lin, Kevin Lin, Guilin Liu, Edith Llontop, Loic Magne, Ajay Mandlekar, Avnish Narayan, Soroush Nasiriany, Scott Reed, You Liang Tan, Guanzhi Wang, Zu Wang, Jing Wang, Qi Wang, Jiannan Xiang, Yuqi Xie, Yinzhen Xu, Zhenjia Xu, Seonghyeon Ye, Zhiding Yu, Ao Zhang, Hao Zhang, Yizhou Zhao, Ruijie Zheng, and Yuke Zhu. GR00T N1: An open foundation model for generalist humanoid robots. In *ArXiv Preprint*, March 2025.

Junwoo Park, Daehoon Gwak, Jaegul Choo, and Edward Choi. Self-supervised contrastive learning for long-term forecasting. (arXiv:2402.02023), March 2024. doi: 10.48550/arXiv.2402.02023. URL http://arxiv.org/abs/2402.02023. arXiv:2402.02023 [cs].

Cheng Peng, Yufeng Chen, Qing Chen, Zhaohui Tang, Lingling Li, and Weihua Gui. A remaining useful life prognosis of turbofan engine using temporal and spatial feature fusion. *Sensors*, 21(2): 418, January 2021. ISSN 1424-8220. doi: 10.3390/s21020418. URL https://www.mdpi.com/1424-8220/21/2/418.

Shanmugasivam Pillai and Prahlad Vadakkepat. Two stage deep learning for prognostics using multi-loss encoder and convolutional composite features. *Expert Systems with Applications*, 171:114569, June 2021. ISSN 09574174. doi: 10.1016/j.eswa.2021.114569. URL https://linkinghub.elsevier.com/retrieve/pii/S0957417421000105.

Katharina Rombach, Gabriel Michau, Wilfried Bürzle, Stefan Koller, and Olga Fink. Learning informative health indicators through unsupervised contrastive learning. *IEEE Transactions on Reliability*, page 1–13, 2024. ISSN 0018-9529, 1558-1721. doi: 10.1109/TR.2024.3397394. URL https://ieeexplore.ieee.org/document/10531793/.

Bhaskar Saha and Kai Goebel. Battery data set. NASA Prognostics Data Repository, NASA Ames Research Center, Moffett Field, CA, 2007.

Abhinav Saxena, Kai Goebel, Don Simon, and Neil Eklund. Damage propagation modeling for aircraft engine run-to-failure simulation. In *2008 International Conference on Prognostics and Health Management*, page 1–9, Denver, CO, USA, October 2008. IEEE. ISBN 978-1-4244-1935-7. doi: 10.1109/PHM.2008.4711414. URL http://ieeexplore.ieee.org/document/4711414/.

Florian Schroff, Dmitry Kalenichenko, and James Philbin. Facenet: A unified embedding for face recognition and clustering. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 815–823, 2015.

Pierre Sermanet, Corey Lynch, Yevgen Chebotar, Jasmine Hsu, Eric Jang, Stefan Schaal, Sergey Levine, and Google Brain. Time-contrastive networks: Self-supervised learning from video. In *2018 IEEE international conference on robotics and automation (ICRA)*, pages 1134–1141. IEEE, 2018.

Lucas Thil, Jesse Read, Rim Kaddah, and Guillaume Doquet. I-glide: Input groups for latent health indicators in degradation estimation. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 395–411. Springer, 2025.

Kwok L Tsui, Nan Chen, Qiang Zhou, Yizhen Hai, and Wenbin Wang. Prognostics and health management: A review on data driven approaches. *Mathematical Problems in Engineering*, 2015 (1):793161, 2015.

Amit Vinograd, Aditya Nair, Joseph H. Kim, Scott W. Linderman, and David J. Anderson. Causal evidence of a line attractor encoding an affective state. *Nature*, 634(8035):910–918, October 2024. ISSN 0028-0836, 1476-4687. doi: 10.1038/s41586-024-07915-x. URL https://www.nature.com/articles/s41586-024-07915-x.

Tongzhou Wang and Phillip Isola. Understanding contrastive representation learning through alignment and uniformity on the hypersphere. In *International conference on machine learning*, pages 9929–9939. PMLR, 2020.

Ying Wang, Oumayma Bounou, Gaoyue Zhou, Randall Balestriero, Tim GJ Rudner, Yann LeCun, and Mengye Ren. Temporal straightening for latent planning. *arXiv preprint arXiv:2603.12231*, 2026.

Yuhui Wang, Andi Wang, Di Wang, and Dong Wang. Deep learning-based sensor selection for failure mode recognition and prognostics under time-varying operating conditions. *IEEE Transactions on Automation Science and Engineering*, 22:9252–9274, 2025. ISSN 1545-5955, 1558-3783. doi: 10.1109/tase.2024.3504595. URL https://ieeexplore.ieee.org/document/10781431/.

Xiang Zhang, Ziyuan Zhao, Theodoros Tsiligkaridis, and Marinka Zitnik. Self-supervised contrastive pre-training for time series via time-frequency consistency. (arXiv:2206.08496), October 2022. doi: 10.48550/arXiv.2206.08496. URL http://arxiv.org/abs/2206.08496. arXiv:2206.08496 [cs].

Mengjie Zhao and Olga Fink. Disentangling slow and fast temporal dynamics in degradation inference with hierarchical differential models. *Reliability Engineering & System Safety*, 277: 112943, January 2027. ISSN 09518320. doi: 10.1016/j.ress.2026.112943. URL https://linkinghub.elsevier.com/retrieve/pii/S0951832026007532.

Zeqi Zhao, Bin Liang, Xueqian Wang, and Weining Lu. Remaining useful life prediction of aircraft engine based on degradation pattern learning. *Reliability Engineering & System Safety*, 164:74–83, August 2017. ISSN 09518320. doi: 10.1016/j.ress.2017.02.007. URL https://linkinghub.elsevier.com/retrieve/pii/S0951832017302454.

Shuai Zheng, Kosta Ristovski, Ahmed Farahat, and Chetan Gupta. Long short-term memory network for remaining useful life estimation. In *2017 IEEE International Conference on Prognostics and Health Management (ICPHM)*, page 88–95, Dallas, TX, USA, 2017. IEEE. ISBN 978-1-5090-5710-8. doi: 10.1109/ICPHM.2017.7998311. URL http://ieeexplore.ieee.org/document/7998311/.

Appendix Contents

A Notation and Problem Formulation **15**

B Model Architecture and Training **16**

B.1 Theoretical Basis for Polar-Coordinate Diagnostics 16

B.2 Derived Indicators 17

C Datasets **17**

D Additional Results **18**

D.1 Five-Seed C-MAPSS Results, with PHM08 18

D.2 STEP versus the Classical Similarity Indicator on C-MAPSS 18

D.3 Held-Out Controls for θ and the Radius 18

D.4 Loss-Term and Sampling Ablations 19

D.5 Prototype Ablation ($\lambda_3 = 0$) 19

D.6 SoftCLT Baseline 19

D.7 Multi-Mode Failure Discovery 20

D.8 Sensor-Noise Robustness 20

D.9 Feature Ablation for the RUL Head 20

E Supplementary Diagnostics **20**

E.1 θ versus Normalized Time on FD002 20

E.2 Latent Dimension 21

F Mouse Neural Activity (Qualitative) **21**

A Notation and Problem Formulation

Table 4: Notation used throughout the paper.

Symbol	Description
x_t	Observation (window) at time t
s_t	Unobserved latent (e.g. health) state at time t
z_t	Learned latent: $z_t = f_{\text{enc}}(x_t)$
$\tilde{z}$	Standardized latent, $(z - \mu)/\sigma$
$\bar{z}$	Unit-normalized latent, $z/\|z\|$
θ_t, r_t	Latent compass: angle and radius in the anchor plane $(\mathbf{e}_1, \mathbf{e}_2)$ (Eq. 4)
h_t	Full-dimensional readout: Mahalanobis distance of z_t from the healthy state
$\hat{x}_t$	Reconstruction: $\hat{x}_t = f_{\text{dec}}(z_t)$
(x_a, x_p, x_n), (z_a, z_p, z_n), (t_a, t_p, t_n)	Anchor, positive, negative samples, codes and time indices
δ_t	Temporal sampling margin
$z_{\text{init}} = \mathbf{e}_1, z_{\text{end}} = \mathbf{e}_2$	Fixed prototype vectors
$X_{\text{init}}, X_{\text{end}}$	First / last k steps of each training unit
$\mathcal{L}_{\text{recon}}, \mathcal{L}_{\text{trip}}, \mathcal{L}_{\text{proto}}$	Reconstruction, cosine triplet and prototype losses

RUL formulation. Each unit gives a trajectory $(x_1, \ldots, x_T)$, sampled independently and of varying length. The degradation dataset is $\mathcal{D} = \{(x_t^{(u)}, y_t^{(u)} = T_u - t)\}$, where y is the RUL. We seek indicators computed from x_t that are predictive of y_t.

Progressive latent Markov process. The mechanism behind a progression can be written as a non-regressive Markov chain $\{s_t\}_{t=1}^{T_u}$. Without assuming a discrete state space, if discretized into $K \geq 2$ states $s_t \in \{1, \ldots, K\}$, we have $s_1 \neq s_T$, s_T is absorbing, and indices can be ordered so that $s_t \geq s_{t-1}$. STEP does not fit an explicit Markov model; this formulation motivates learning an ordered latent space.

B Model Architecture and Training

The backbone is a transformer encoder with a small MLP decoder. The low-capacity decoder prevents the decoder from bypassing the latent structure (akin to posterior collapse in VAEs). The N-CMAPSS experiments replace the encoder by a CNN, GRU, LSTM or PatchTST [Nie et al., 2023] encoder with the same objective. The C-MAPSS STEP autoencoder has 240,562 parameters; the N-CMAPSS encoder bodies have approximately 28k (CNN), 11k (GRU), 15k (LSTM) and 169k (PatchTST) parameters (counted from the architecture definitions, not logged).

Table 5: Backbone components.

Component	**Description**	**Specification**
Input projection	Linear projection	$\mathbb{R}^{\text{input_dim}} \to \mathbb{R}^{d_{\text{model}}}$
Transformer encoder	Stack of transformer layers	4 layers, 8 heads, FF= $2\times d_{\text{model}}$, GELU, dropout 0.1
Global pooling	Sequence aggregation	Adaptive average pooling
Latent projection	MLP to latent space	$\mathbb{R}^{d_{\text{model}}} \to \mathbb{R}^{32} \to \mathbb{R}^{\text{latent_dim}}$
Decoder	MLP reconstruction	$\mathbb{R}^{\text{latent_dim}} \to \mathbb{R}^{64} \to \mathbb{R}^{128} \to \mathbb{R}^{\text{input_dim}}$

Table 6: Default training hyperparameters ($d_{\text{model}} = 32$).

Parameter	**Value**	**Parameter**	**Value**
Window size w	10	Triplet margin m	1.0
Latent dimension	2 (also 4, 8, 16, 32)	λ_1 (recon)	1.0
Batch size	256	λ_2 (triplet)	1.0
Learning rate	1×10^{-3}	λ_3 (prototype)	$0.1 \to 0.05$
Seeds	5 (probes: 3)	Prototype windows k	8 initial, 8 final

Optimization and model selection. The encoder is trained with AdamW (learning rate 10^{-3} with cosine decay, weight decay 0.01, batch 256) for 100 epochs, without early stopping. The transformer RUL head is trained on frozen features with Adam (learning rate 10^{-3}, batch 32) for 20 epochs. The linear head is ordinary least squares.

Prototype-weight schedule. The triplet weight is constant (λ_2=1). The prototype weight starts at λ_3=0.1 and decays geometrically by a factor 0.95 per epoch to a floor of 0.05, so anchoring mainly shapes the early geometry. (The submitted version called this "β-scheduling".)

Sampler. Positives are drawn from the same unit as the anchor within δ_t=20 cycles. Negatives are drawn at random from windows in a different coarse life band (remaining life >125, 75–125, 30–75, ≤30 cycles), which lets valid pairs be sampled without scanning each trajectory. The prototype term uses the first and last k=8 windows of each training unit. SoftCLT is trained with its own augmentation-based objective (Appendix D.6).

B.1 Theoretical Basis for Polar-Coordinate Diagnostics

Using θ and r as diagnostics follows the alignment/uniformity view of contrastive learning [Wang and Isola, 2020]: positive pairs should be close (alignment), and features should spread over the

hypersphere (uniformity). For progressive time series, alignment appears as a smooth, monotone θ, and uniformity as distinct states occupying distinct regions. The origin is invariant to the data distribution, and movement relative to it has a direction (θ) and a magnitude (r). Methods not designed for circular geometry (AE, SoftCLT) can be read through the same lens: erratic θ/r indicates poor alignment or uniformity.

B.2 Derived Indicators

Besides θ, r (Eq. 4) and h_t, the ablations use the angular monotonicity and a progression score:

$$\mathcal{M} = \frac{1}{T-1}\sum_{t=1}^{T-1}\mathbb{I}(\theta_{t+1} > \theta_t), \qquad \hat{u} = \frac{z_T - z_1}{\|z_T - z_1\|}, \quad s_i = z_i \cdot \hat{u}, \quad s'_i = \frac{s_i - s_{\min}}{s_{\max} - s_{\min}}, \tag{5}$$

and the line deviation $d_i = \|v_i - p_i\|$ with $u = z_T - z_1$, $v_i = z_i - z_1$, $p_i = \frac{v_i \cdot u}{\|u\|^2} u$.

HI metrics. Trendability is the mean over units of $|\rho_{\text{Spearman}}(\text{HI}_t, \text{RUL}_t)|$. Monotonicity is $|\#\{\Delta\text{HI} > 0\} - \#\{\Delta\text{HI} < 0\}|/(T-1)$ per unit, and prognosability is $\exp\big(-\text{std}(\text{HI}_{\text{fail}})/\,\text{mean}\,|\text{HI}_{\text{start}} - \text{HI}_{\text{fail}}|\big)$, following Coble [2010].

Classical indicators. All are computed on the raw sensor features of the last frame of each window, with no encoder. *PCA-based*: projection on the first principal component. *SOM-based*: index of the best-matching unit on a 1-D Kohonen chain of 24 units initialized along the first principal component. *Similarity*: Mahalanobis distance from the healthy feature distribution. "Healthy" windows are the 15% of training windows with the largest remaining life, the same reference as the STEP Mahalanobis readout.

C Datasets

Table 7: NASA C-MAPSS subsets.

	FD001	FD002	FD003	FD004
Train units	100	260	100	248
Test units	100	259	100	249
OCs	1	6	1	6
FMs	1	1	2	2

N-CMAPSS. We use the CruiseBench [Cheng and Miao, 2026] protocol on DS02-006 (9 units): cruise mask, window 256 with stride 10, 18 channels, min-max scaling fit on the development units, RUL cap 34, the cap CruiseBench prescribes for DS02-006 (caps 80 and 125 in Table 8). The supervised GRU and TSMixer are reproduced in the same pipeline. HI trendability in Table 3 is the healthy-referenced Mahalanobis readout computed per test window; the 0.93 of Table 2c uses the same readout computed per unit with window 20.

NASA battery. We evaluate the four classic cells B0005, B0006, B0007 and B0018 with leave-one-cell-out cross-validation (train on three cells, test on the held-out one). End of life is fixed at 1.4 Ah, and each trajectory is clipped at the first cycle below this threshold. Each discharge cycle is represented by its voltage and temperature curves, resampled on a fixed 64-point time grid and truncated at the shortest training-cell discharge, so that neither capacity nor discharge duration is given to the model. The classical indicators receive the same input, and raw capacity serves as a reference. The encoder is a GRU (dim(z)=8). Its hidden size and number of epochs are selected by an inner leave-one-cell-out loop on the training cells only, and the held-out cell is scored once. The sampler bands and the healthy reference use the state of health (capacity over its initial maximum) of the training cells only. We report trendability on the held-out cell, mean ± sample std over 4 folds × 3 seeds. In an earlier version that also gave the encoder discharge duration, trendability was inflated, because at constant 2 A current duration is proportional to capacity.

Table 8: N-CMAPSS DS02-006 (CruiseBench protocol) RUL RMSE at caps 34 (benchmark cap), 80 and 125; mean ± sample std over 5 seeds. Each cap is a full rerun, since the cap also enters encoder training through the clipped remaining life. HI trendability at caps 34/80/125.

	RUL RMSE (cap)			Trend.
Model	34	80	125	34/80/125
STEP, CNN	2.19±0.08	6.62±0.06	6.75±0.10	.91/.90/.88
STEP, GRU	2.30±0.06	6.84±0.14	6.92±0.10	.92/.67/.68
STEP, LSTM	2.38±0.06	6.98±0.20	6.90±0.14	.89/.85/.91
STEP, PatchTST	2.87±0.24	7.11±0.35	7.20±0.07	.86/.82/.86
GRU, supervised	1.81±0.37	6.33±0.52	6.36±0.30	–
TSMixer, supervised	2.41±0.08	6.18±0.22	6.03±0.10	–
Recon-only (null)	9.39±1.32	18.13±2.44	16.87±3.10	.06/.09/.09
Random-init (null)	3.44±0.53	10.01±2.71	9.66±2.07	.14/.15/.15

D Additional Results

D.1 Five-Seed C-MAPSS Results, with PHM08

Table 9: Tr.STEP on C-MAPSS over 5 seeds: mean ± sample std of RUL RMSE and PHM08 score (lower is better).

Subset	$\dim(z)$	RMSE (mean ± std)	PHM08
FD001	2	13.18 ± 0.79	193 ± 30
FD001	4	13.12 ± 0.23	181 ± 13
FD001	32	13.27 ± 0.38	179 ± 11
FD002	2	12.13 ± 0.10	340 ± 13
FD002	4	11.78 ± 0.40	305 ± 12
FD002	32	11.57 ± 0.34	288 ± 12
FD003	2	13.30 ± 0.46	258 ± 25
FD003	4	13.66 ± 0.89	265 ± 45
FD003	32	12.66 ± 0.51	263 ± 39
FD004	2	15.31 ± 0.84	688 ± 105
FD004	4	13.31 ± 0.32	585 ± 20
FD004	32	13.13 ± 0.69	563 ± 103

D.2 STEP versus the Classical Similarity Indicator on C-MAPSS

Table 10: Trendability of the STEP Mahalanobis readout versus the input-space similarity HI, 3 seeds, paired trajectory bootstrap (B=4000). This harness scores trajectories differently from Table 2a, so absolute values are not comparable across the two tables.

Subset	n_{traj}	STEP [95% CI]	Similarity HI [95% CI]	Δ [95% CI]
Pooled	709	0.809 [0.796, 0.821]	0.647 [0.635, 0.659]	+0.162 [0.151, 0.173]
FD001	100	0.845 [0.810, 0.878]	0.794 [0.775, 0.812]	+0.051 [0.026, 0.075]
FD002	260	0.861 [0.854, 0.868]	0.671 [0.658, 0.684]	+0.190 [0.179, 0.200]
FD003	100	0.664 [0.604, 0.721]	0.665 [0.627, 0.700]	−0.001 [−0.040, 0.036]
FD004	249	0.798 [0.781, 0.815]	0.556 [0.532, 0.578]	+0.243 [0.230, 0.256]

D.3 Held-Out Controls for θ and the Radius

Held-out units of FD002 and FD004, the partial correlation is the mean over the two subsets (STEP -0.32 and -0.29; AE 0.02 and 0.03). The angle probe measures balanced accuracy within matched-θ bins.

Table 11: Held-out C-MAPSS units. Top: partial correlation of the angle with RUL controlling for elapsed cycles. Bottom: 6-way OC classification, balanced accuracy (chance 0.17); the angle probe controls for progression level.

	AE	STEP
$\lvert\rho(\theta, \mathrm{RUL} \mid t)\rvert$	0.03	0.30 ($p < 0.001$)
OC from angle θ	0.52	0.17
	FD002	FD004
OC from full STEP latent (linear probe)	0.97	0.98
OC from STEP radius r	0.37	0.39

D.4 Loss-Term and Sampling Ablations

Loss terms. C-MAPSS reconstruction-only versus full STEP is in Table 2a. On N-CMAPSS, reconstruction alone gives unusable features while the full objective gives strong ones on every encoder family (Table 3).

Triplet sampling. Table 12 varies the temporal-proximity threshold τ for positives and the negative mode. This harness uses a different RUL head from Table 1, so its RMSEs are only comparable within the table.

Table 12: Triplet-sampling ablation, STEP, 3 seeds, FD001 and FD004. Trendability (anchor-plane angle) / RUL RMSE.

τ	Negatives	FD001	FD004
4	mixed	0.634 / 17.03	0.471 / 23.74
8	mixed	0.638 / 17.19	0.511 / 23.43
12	mixed	0.639 / 17.19	0.572 / 21.70
20	mixed	0.659 / 17.15	0.628 / 20.44
40	mixed	0.695 / 17.14	0.579 / 21.05
12	random	0.664 / 17.60	0.564 / 22.92
12	cross-unit	0.667 / 17.61	0.507 / 23.29
12	far	0.616 / 17.44	0.586 / 22.34

D.5 Prototype Ablation ($\lambda_3 = 0$)

Removing prototype anchoring degrades the heterogeneous FD004 (12.10 $\rightarrow$ 19.01 RMSE, $+57\%$) and slightly improves the homogeneous FD003 (13.35 $\rightarrow$ 11.59). These are single runs from the submitted version, at $\dim(z)$=4.

D.6 SoftCLT Baseline

Table 13: SoftCLT on C-MAPSS, same backbone, features and heads as STEP, 5 seeds (mean $\pm$ sample std RMSE). †One of five seeds diverged. PHM08 is 220–4400 for SoftCLT against 174–536 for STEP, since late predictions are penalised exponentially.

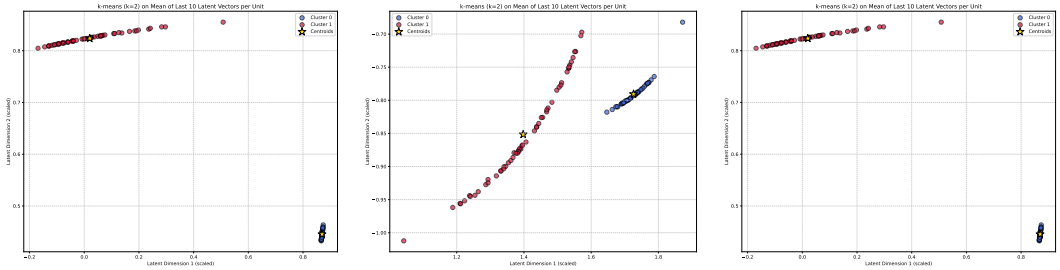
Head	z	FD001	FD002	FD003	FD004
Transformer	2	14.57$\pm$1.13	22.75$\pm$7.14†	14.18$\pm$0.65	21.93$\pm$0.94
Transformer	4	15.34$\pm$1.17	21.23$\pm$0.98	14.64$\pm$1.07	22.77$\pm$1.27
Transformer	32	15.50$\pm$0.61	19.73$\pm$1.25	14.87$\pm$1.06	23.09$\pm$2.08
Linear	2	15.19$\pm$1.73	22.78$\pm$7.13†	15.02$\pm$1.22	21.04$\pm$0.70
Linear	4	15.57$\pm$0.73	19.30$\pm$0.56	14.54$\pm$1.72	21.79$\pm$0.46
Linear	32	14.84$\pm$0.36	16.85$\pm$0.53	15.97$\pm$3.11†	19.08$\pm$0.95

D.7 Multi-Mode Failure Discovery

A single z_{end} prototype could in principle merge qualitatively different terminal states. Clustering the terminal embeddings with unsupervised K-means (k=2) instead recovers the two failure modes of FD003 and FD004 without supervision (Table 14).

Table 14: Unsupervised K-means (k=2) on terminal STEP embeddings. Distinct failure modes are recovered without supervision despite a single z_{end} prototype.

Dataset	$\dim(z)$	Silhouette	Separation ratio	Misplaced (%)
FD003	4	0.75	6.3×	0%
FD003	16	0.93	21.0×	0%
FD003	32	0.97	52.8×	0%
FD004	8	0.60	3.6×	0%
FD004	16	0.62	1.9×	0%
FD004	32	0.66	3.5×	0%



(a) $\dim(z)$=4. (b) $\dim(z)$=16. (c) $\dim(z)$=32.

Figure 4: **K-means (k=2) on terminal latent embeddings (mean of last 10 steps) recovers the two FD003 failure modes** at every latent dimension.

D.8 Sensor-Noise Robustness

We rescale the sensor noise by $\times 0.5$ and $\times 2.5$ and retrain. RMSE degrades gracefully (Table 15), since non-regression is assumed of the latent state, not of the raw sensors.

Table 15: RUL RMSE at $\dim(z)$=16 with sensor noise rescaled (single runs).

Noise	FD001	FD002	FD003	FD004
$\times 0.5$	11.34	14.23	13.23	18.36
$\times 2.5$	13.20	15.12	14.52	24.70

D.9 Feature Ablation for the RUL Head

We vary the features given to the RUL head (Table 16).

Performance varies little across feature sets. I-GLIDE, by contrast, needed to combine several HIs and was unstable when they were used in isolation.

E Supplementary Diagnostics

E.1 θ versus Normalized Time on FD002

The figures below are descriptive. Normalized time t/T needs the oracle end of life and carries no information about absolute RUL, so comparisons against it are weaker than the held-out controls on elapsed cycles (Table 11), which we consider the primary evidence.

Table 16: Feature combinations for the RUL head, $\dim(z)=2$, single runs from the submitted version, no feature smoothing, observation window 75. Bold: best per subset.

Model	Features	FD001	FD002	FD003	FD004
Random Forest	PS, θ	14.70	12.90	13.43	18.02
Random Forest	PS, θ, r	13.04	13.04	12.98	18.13
Random Forest	θ, r	12.97	12.98	13.17	18.18
Random Forest	θ	14.41	12.86	13.37	18.67
Random Forest	z	13.63	13.13	12.59	17.98
Random Forest	z, PS, θ, r	13.04	12.89	12.84	18.20
Linear regression	z, PS, θ, r	15.30	14.38	17.96	19.48
Transformer	z	11.67	13.03	**11.48**	17.11
Transformer	θ, z	**10.65**	13.19	13.19	17.53
Transformer	PS, θ	14.12	**12.57**	13.93	17.44
Transformer	PS, θ, r	13.36	12.87	12.07	**16.93**
Transformer	θ	14.04	13.00	16.10	17.62
Transformer	θ, r	12.89	12.61	12.16	17.43
Transformer	z, PS, θ, r	12.73	12.66	12.02	17.08

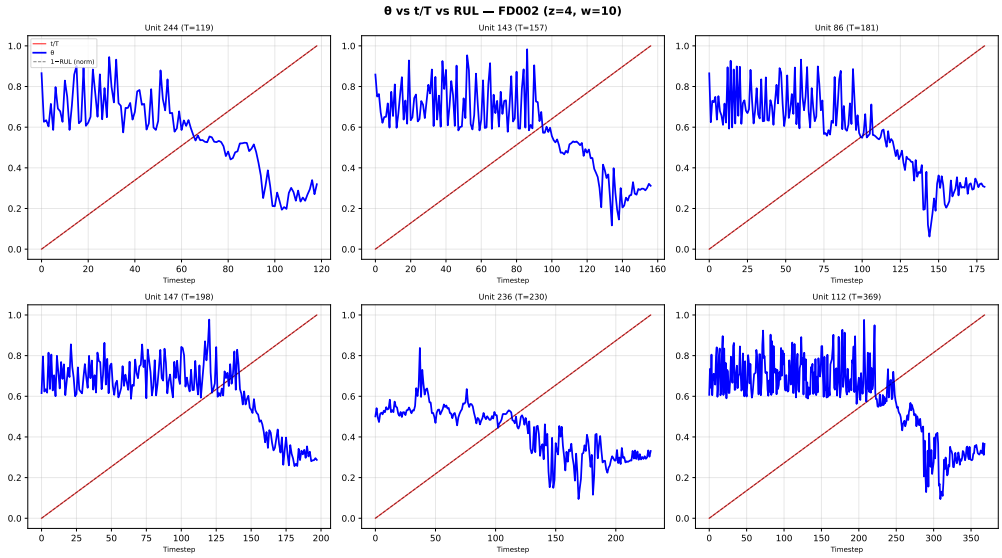


Figure 5: **Per-unit θ (blue) and t/T (red) on six FD002 engines.** t/T is a diagonal by construction; θ stays in a high, noisy healthy regime and then drops near failure, with transition timing that varies across engines.

E.2 Latent Dimension

Note that the 2D compass uses only the anchor plane at every dimension; the per-dimension evidence below concerns the stability of that plane's geometry, and Table 2c the full-dimensional readout.

F Mouse Neural Activity (Qualitative)

The submitted version included a forecasting experiment on this single-trajectory recording, following the forward-simulation protocol of latent dynamics baselines [Hu et al., 2025]. That protocol lets the encoder see the windows it later rolls out; under a strict temporal split (encoder and normalization fit on the first 70%, rollout on the last 30%), the 15-step R^2 drops from 0.32 to -0.19 (5 seeds), below a persistence baseline (0.90). We therefore removed the forecasting experiment. We keep only the qualitative latent topology below, for which the encoder is trained on the whole recording and no out-of-sample claim is made: STEP shows phase changes aligned with intruder entry and exit, as does SoftCLT, with a less ordered manifold.

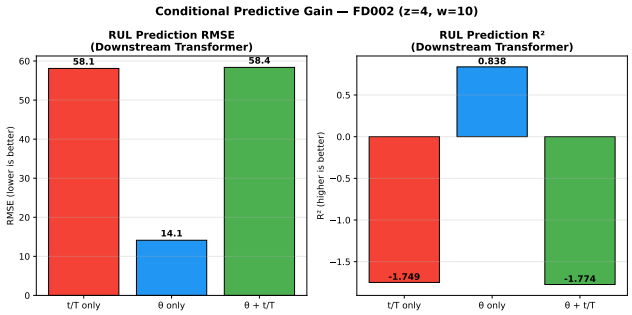


Figure 6: **Downstream RUL from θ versus t/T (FD002).** A head on θ alone reaches RMSE 14.1 (R^2=0.84), one on t/T alone 58.1; adding t/T to θ does not help. Since t/T carries no absolute-scale information, this shows that θ is predictive, not by itself that it is decoupled from elapsed time.

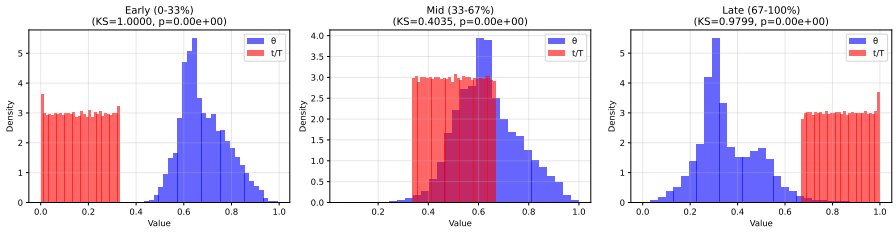


Figure 7: **Phase-wise marginal distributions of θ and t/T on FD002** (KS=1.00 early, 0.40 mid, 0.98 late). Marginal differences do not by themselves test functional decoupling.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The abstract and introduction state the scope (progressive time series), the method, and the evidence: HI trendability on three degradation benchmarks, held-out controls on elapsed time, five-seed RUL results, and qualitative robotics results. Interpretability claims are limited to structured, predictive, linearly accessible coordinates (Sec. 1).

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: Sec. 6 lists the progressive assumption, the need for complete run-to-failure training trajectories (sampler and prototypes), ambiguous or multiple end states, the limited meaning of the coordinates (radius not a mode label, 2D view dataset dependent), and the scope of evidence (no SOTA on FD001/FD003/FD004, qualitative robotics, no forecasting).

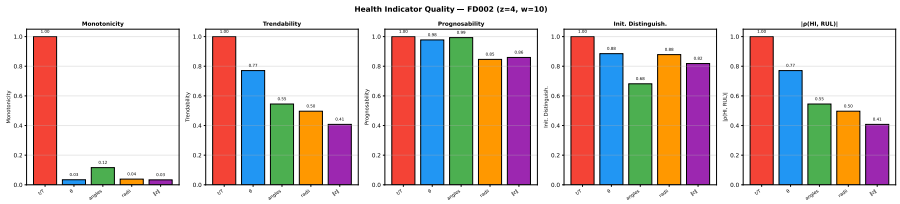
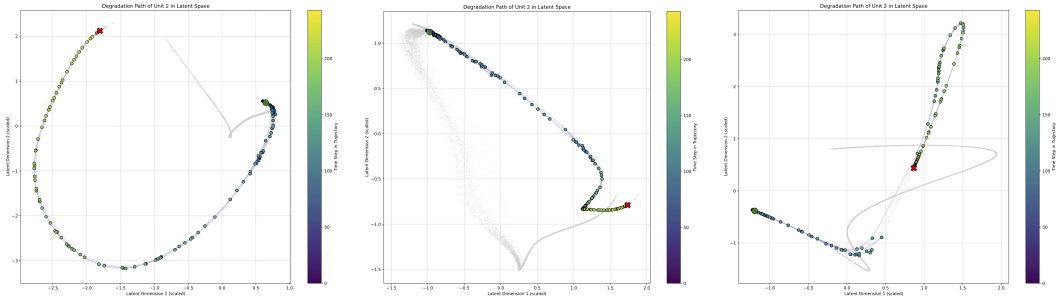


Figure 8: Standard PHM HI metrics on FD002. t/T scores 1.0 on monotonicity and trendability by construction; θ reaches prognosability 0.98 while being computable at test time.



(a) $\dim(z)$=4. (b) $\dim(z)$=16. (c) $\dim(z)$=32.

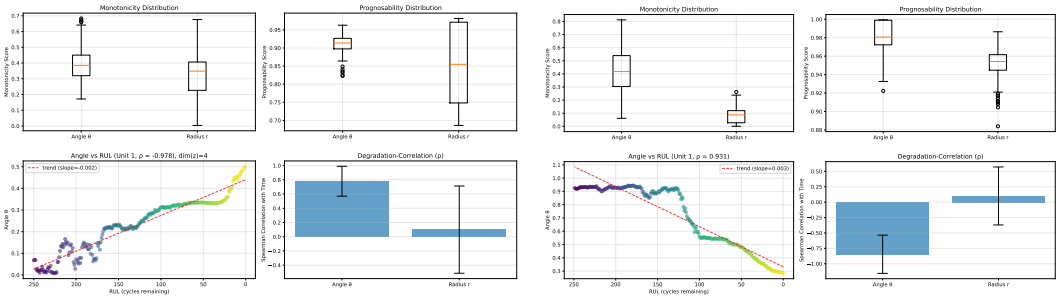
Figure 9: **Anchor-plane geometry on FD003 across latent dimensions.** The direction from the healthy (green) to the end-of-life (red) anchor is preserved; per-unit $|\rho(\theta, \mathrm{RUL})|$ is 0.98 / 0.93 / 0.96 .

Guidelines:

- The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
- The authors are encouraged to create a separate "Limitations" section in their paper.
- The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
- The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
- The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
- The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
- If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
- While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

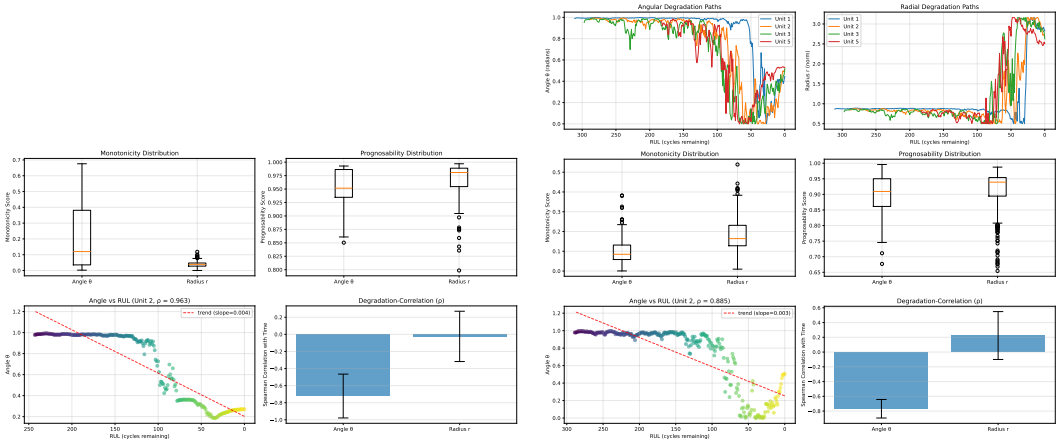
3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?



(a) FD003, $\dim(z)$=4.

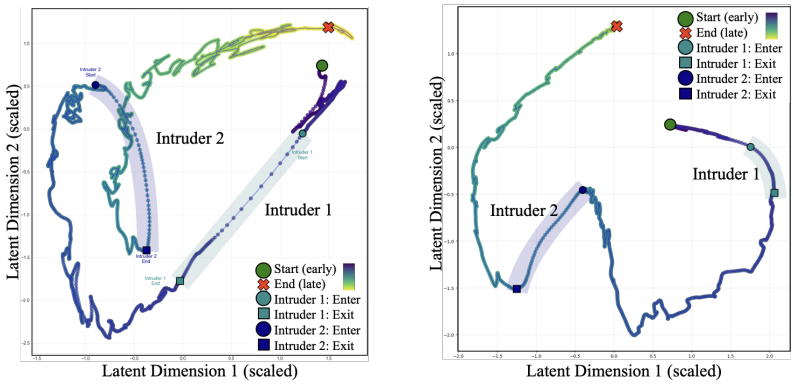
(b) FD003, $\dim(z)$=16.



(c) FD003, $\dim(z)$=32.

(d) FD004, $\dim(z)$=32.

Figure 10: **PHM diagnostics across latent dimensions**: angular and radial degradation paths, monotonicity and prognosability, angle versus RUL, and Spearman correlation.



(a) SoftCLT

(b) STEP

Figure 11: **Mouse brain activity** [Vinograd et al., 2024], $\bar{z}$ (t-SNE), qualitative.

Answer: [N/A]

Justification: The paper makes no formal theoretical claims; the Markov formulation in Appendix A is motivational.

Guidelines:

- The answer [N/A] means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.

- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: Architecture, losses, sampler, hyperparameters, protocols and baselines are given in Secs. 3–4 and Appendices B–C; code is released.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes]

   Justification: Code is available at https://github.com/LucasStill/STEP with scripts to reproduce the experiments. All datasets are public (C-MAPSS, N-CMAPSS, NASA battery, GR00T, LeRobot, and the mouse recording of Vinograd et al.).

Guidelines:

- The answer [N/A] means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

   Justification: Splits follow the standard C-MAPSS protocol and the CruiseBench protocol for N-CMAPSS; battery cell selection, hyperparameters, and head training are described in Sec. 4 and Appendices B–C.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: Main results are mean $\pm$ std over 5 seeds (Tables 1–3); unit-level bootstrap 95% confidence intervals are reported for HI comparisons and in Appendix D. Single-run results (prototype ablation, noise robustness, feature ablation) are labelled as such.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
   - The assumptions made should be given (e.g., Normally distributed errors).
   - It should be clear whether the error bar is the standard deviation or the standard error of the mean.

- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: Appendix B reports the hardware and time per run. Models are small (latent dimension 2–32, $d_{\text{model}} = 32$); each run fits on a single GPU.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: Yes it conforms with the code of ethics.

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: Better health indicators can support predictive maintenance in safety-critical systems. A risk is over-trust: the learned coordinates are predictive features, not validated physical health measures, and Sec. 6 states that any safety-critical use requires domain validation.

    Guidelines:

    - The answer [N/A] means that there is no societal impact of the work performed.
    - If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.

- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

    Answer: [N/A]

    Justification: All datasets are open-sourced and freely accessible, the architectures and methods do not create any need for safeguards.

    Guidelines:

    - The answer [N/A] means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: All datasets and models used are properly credited to their authors, and results accurately reported.

    Guidelines:

    - The answer [N/A] means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.

- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [Yes]

    Justification: We release the code with documentation under the licence stated in the repository.

    Guidelines:

    - The answer [N/A] means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [N/A]

    Justification: No crowdsourcing nor human subjects were involved in this work.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [N/A]

    Justification: No crowdsourcing nor human subjects were involved in this work.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.

- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

    Answer: [N/A]

    Justification: Nothing to declare, LLMs are not part of this work.

    Guidelines:

    - The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.