STALINO: STATE-ADAPTIVE LATENT NEURAL OPERATOR FOR TRAFFIC FORECASTING

Anonymous authors
Paper under double-blind review

ABSTRACT

Accurate traffic forecasting supports congestion management and route guidance, and its errors are most costly when congestion forms or dissolves. However, road sensors report flow, which does not reveal the traffic regime: the same flow occurs in free flow and in congestion, where disturbances travel in opposite directions. This raises two challenges: (1) identifying the regime from flow that is dominated by recurring demand, and (2) sharing information within a regime without erasing the differences between regimes. To tackle these challenges, we propose the State-Adaptive Latent Neural Operator (StaLiNO), which forecasts the deviation of flow from a learned daily and weekly base field and shares information through latent traffic states. Regimes are identified by the Reference-Relative Regime Assignment module, which assigns sensors to latent states by their deviation trajectories and fixes these assignments across the input window, so that each state tracks the same sensors. Sharing within regimes is handled by the Regime-Separated Latent Evolution module, which evolves every latent state independently and returns it to the sensors through the same assignments. We further prove that such sharing lowers reconstruction risk exactly when its distortion is below the noise it removes, and that mixing latent states removes noise that does not grow with the number of sensors while adding distortion that does. Experiments on four real-world traffic datasets demonstrate that StaLiNO achieves state-of-the-art accuracy and remains stable when sensors lose their history. Latent evolution improves accuracy with tracked states and degrades it with states regrouped at every step, and the learned states separate free-flow from congested traffic without access to occupancy.

1 INTRODUCTION

Traffic forecasting supports ramp metering, incident response, and route guidance, and its errors are most costly when congestion forms or dissolves. Road sensors sample a traffic field that evolves over the network; the Lighthill–Whitham–Richards (LWR) model describes this field as a conserved density transported at a state-dependent characteristic speed (Lighthill & Whitham, 1955; Richards, 1956). Spatiotemporal graph networks (Wu et al., 2019; Shao et al., 2022) and decomposition-based forecasters (Fang et al., 2023; Cao et al., 2025) learn this evolution from data. Neural operators offer a complementary view by learning solution operators of physical fields (Kovachki et al., 2023), and recent work extends them to irregular domains and road graphs (Wu et al., 2024; Ge et al., 2026).

A property of traffic sets it apart from the fields these operators usually model. A loop detector reports flow, and flow is a non-invertible projection of the traffic state: on the fundamental diagram, each flow below capacity occurs once on the free-flow branch and once on the congested branch. The two branches have characteristic speeds of opposite sign, so a disturbance moves downstream in free flow and upstream in congestion (Treiber & Kesting, 2013). Two sensors that report the same flow can thus follow different dynamics, as Figure 1 illustrates. A spatial operator should therefore pool information within a regime, and the regime must be inferred because it is never observed.

Existing spatial operators do not infer this regime before they share information. ① **State-independent kernels.** Graph convolutions, learned static adjacencies, and the Laplacian spectral bases of graph neural operators (Wu et al., 2019; 2020; Ge et al., 2026) weight a neighbor in the same way in free flow and in congestion, so a model whose spatial exchange relies on them averages across regimes at a congestion front. ② **Per-step states.** Dynamic-graph forecasters (Li et al.,

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

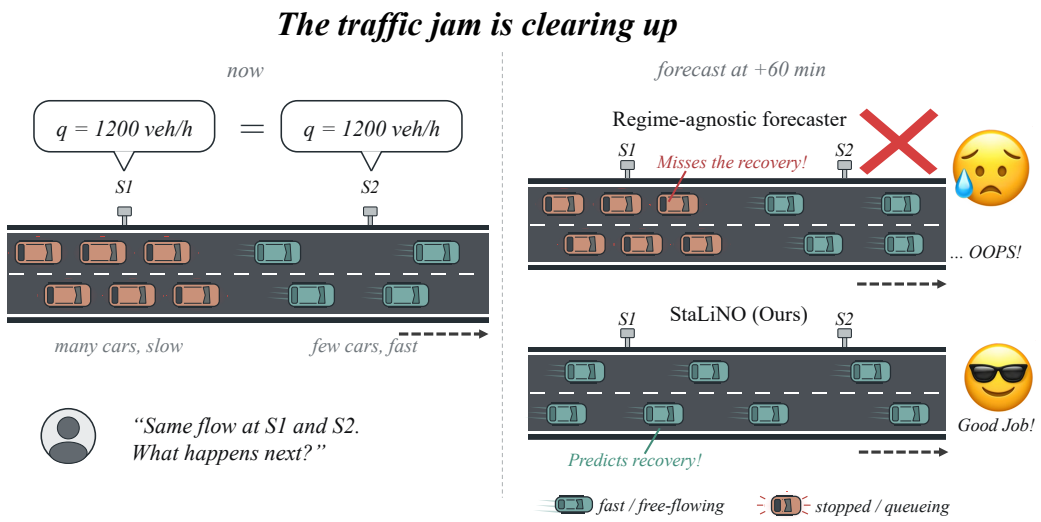


Figure 1: Schematic of why current flow alone does not determine the forecast. Sensors S_1 and S_2 have the same observed flow, but S_1 is in a queue and S_2 is in free flow. A regime-agnostic forecaster misses the illustrated recovery, whereas StaLiNO uses reference-relative trajectories to distinguish the states.

2023a) and latent neural operators (Wu et al., 2024; Alkin et al., 2024; Wang & Wang, 2024) adapt their kernels to the input, but they regenerate dependencies or latent states at every step, and latent operators additionally let their states interact through attention. In traffic, a flow snapshot does not identify the regime, and a state whose members change at every step has no trajectory for a temporal model to follow. Appendix G reviews related work in detail.

We therefore learn the forecasting operator on the deviation of flow from a calendar-locked base field and share information through latent states that are inferred from each sensor's deviation trajectory and tracked across the input window. This raises two challenges. **(a) Identifying regimes from flow.** Recurring demand dominates the flow signal, and a flow level alone maps to two regimes. Grouping sensors by raw flow or by a single snapshot therefore merges free-flow and congested sensors with similar levels. From flow alone, a regime becomes identifiable relative to what is expected at that location and time and from how the deviation has recently evolved. **(b) Sharing within regimes without erasing their differences.** Pooling sensors into a shared state removes noise but introduces bias whenever the pooled sensors differ, and letting latent states interact, as token attention does in latent operators, pools them again at a coarser level. This calls for latent states whose membership stays fixed over the window, so that a temporal model follows one group of sensors, and that remain separate while they evolve. StaLiNO addresses each challenge with one module, while local road operators and a stable pole–residue temporal operator (Cao et al., 2024) retain location-specific detail. Our contributions are as follows.

- We propose StaLiNO, a latent neural operator that forecasts the deviation of flow from a calendar-locked base field and shares information through latent traffic states, accounting for the traffic regime that flow does not reveal. A derivation from the LWR model shows why the spatial kernel depends on this regime and why the base field makes the regime identifiable from flow.
- The Reference-Relative Regime Assignment module identifies regimes from flow by assigning sensors to latent states according to their deviation trajectories and fixing these assignments across the input window, while the Regime-Separated Latent Evolution module shares information within regimes by pooling each state from its observed sensors and evolving every state independently.
- A bias–noise analysis shows that sharing through latent states lowers reconstruction risk exactly when its distortion is below the noise it removes, and that mixing latent states

removes noise that does not grow with the number of sensors while adding distortion that does, which explains why StaLiNO groups sensors by regime and keeps the states apart.

- Experiments on four real-world traffic datasets demonstrate that StaLiNO achieves state-of-the-art accuracy and remains stable when sensors lose their history. Further analyses show that latent evolution improves accuracy with tracked states and degrades it with states regrouped at every step, and that the learned states separate free-flow from congested traffic without access to occupancy.

2 METHOD

2.1 PROBLEM FORMULATION AND OPERATOR VIEW

We consider N sensors on a road network with adjacency matrix $A \in \mathbb{R}^{N\times N}$. The input consists of the flow $X \in \mathbb{R}^{T_1\times N}$ observed over the past T_1 steps up to timestamp t, validity indicators $\chi \in \{0,1\}^{T_1\times N}$ that mark missing values, and the calendar features κ_s of every historical and future timestamp s. We learn an operator $\mathcal{F}_\theta$ that maps (X, χ, κ, A) to the forecast $\widehat{Y} \in \mathbb{R}^{T_2\times N}$ of all sensors over the next T_2 steps, where $\hat{y}_{t+h,i}$ denotes the forecast for sensor i at step $h \in \{1, \dots, T_2\}$.

Traffic regimes behind flow. On a homogeneous segment, the LWR model conserves the density ρ under a flux $q(\rho)$ (Lighthill & Whitham, 1955; Richards, 1956). Write the density as a base state plus a deviation, $\rho = \bar{\rho} + \eta$, where $\bar{\rho}$ also satisfies the conservation law. For a quadratic flux, subtracting the two conservation laws gives

$$\partial_t \rho + \partial_\xi q(\rho) = 0, \qquad \partial_t \eta + \partial_\xi\big(q'(\bar{\rho})\,\eta\big) + \tfrac{1}{2} q''\,\partial_\xi\big(\eta^2\big) = 0, \tag{1}$$

where ξ denotes road position. The deviation is carried at the characteristic speed of the base state and interacts with itself through the quadratic term; along a characteristic, the deviation at ξ after a horizon h stays close to its current value at $\xi - ch$. Two facts follow. First, detectors observe q rather than ρ. Because q is concave with its maximum at a critical density, the map $\rho \mapsto q(\rho)$ is two-to-one below capacity, and $q'(\rho)$ changes sign at the critical density, so the unobserved branch decides whether a disturbance travels downstream or upstream. Second, the evolution of η depends on the base state, so a deviation is read against the base field at that location and time. The first fact makes the spatial kernel of the forecasting operator regime-dependent, and the second makes the base field the reference against which the regime is identified from flow. Detectors do not observe $\bar{\rho}$ either, so StaLiNO learns its observable counterpart, a calendar-locked flow field, and forecasts the deviation of flow from it. The derivation of Eq. (1) and the translation bound along characteristics are given in Appendix C.

Operator view. StaLiNO forecasts the deviation field with a stack of field layers that follow the neural operator template (Kovachki et al., 2023): local operators act at every sensor and time step, and a global kernel operator $\mathcal{K}_\ell$ exchanges information across the road domain Ω. Let λ be the measure on Ω that the sensors sample, and let $S_m(\xi; r) \geq 0$ with $\sum_m S_m(\xi; r) = 1$ be membership functions of M latent states that depend on the recent trajectory of the flow deviation r. For a value field V, we define

$$(\mathcal{K}_\ell V)(\xi, \cdot) = \sum_{m=1}^{M} S_m(\xi; r)\ \Phi_\ell\left[\frac{\int_\Omega S_m(\xi'; r)\, V(\xi', \cdot)\,\mathrm{d}\lambda(\xi')}{\int_\Omega S_m(\xi'; r)\,\mathrm{d}\lambda(\xi')}\right] W_V, \tag{2}$$

where Φ_ℓ evolves one latent trajectory in time. When Φ_ℓ is the identity, $\mathcal{K}_\ell$ is a kernel integral operator with kernel $k_r(\xi, \xi') = \sum_m S_m(\xi; r) S_m(\xi'; r) / \int_\Omega S_m\,\mathrm{d}\lambda$, whose rank is at most M and whose shape follows the current regime. Replacing λ by the sensors observed at each step gives the pooling that StaLiNO computes. Figure 2 shows the two modules that realize Eq. (2), detailed in Sections 2.2 and 2.3.

2.2 REFERENCE-RELATIVE REGIME ASSIGNMENT

Base field. This module addresses challenge (a) by grouping sensors according to how their flow departs from a base field. The base field is stored in a daily table $P_d \in \mathbb{R}^{L_d\times N}$ and a weekly table

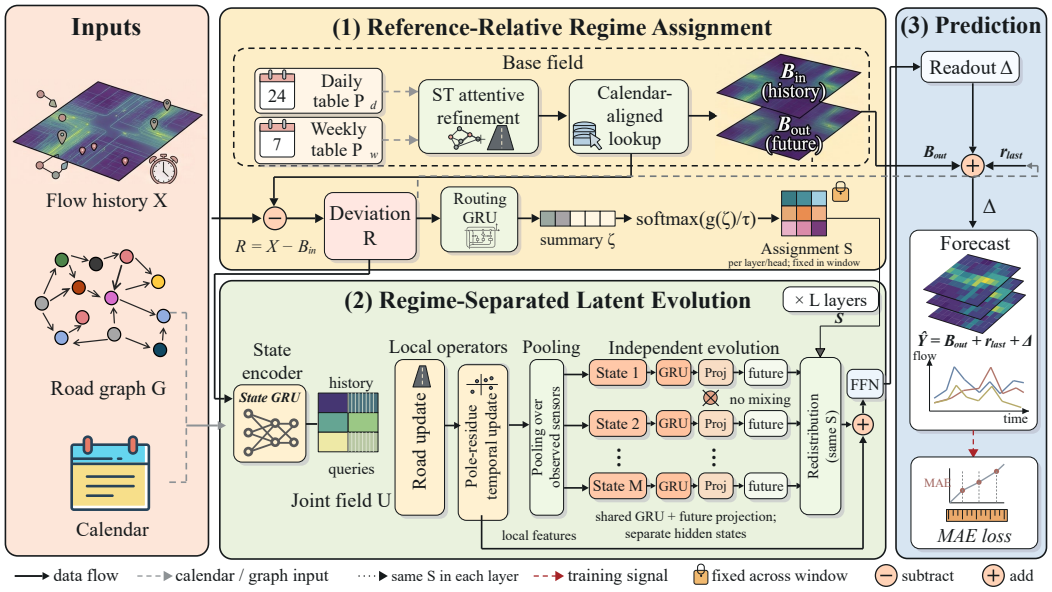


Figure 2: Overview of StaLiNO. (1) Reference-Relative Regime Assignment: a learned daily and weekly base field is subtracted from the observations, and each sensor is softly assigned to M latent states from its deviation trajectory; the assignments stay fixed across the window. (2) Regime-Separated Latent Evolution: each field layer applies local road and temporal operators, pools each latent state from its observed sensors at every historical step, evolves every latent sequence independently, and returns the evolved states through the same assignments. (3) The forecast adds a learned correction to the future base field and the latest valid deviation. The figure uses the following abbreviations: spatiotemporal (ST), gated recurrent unit (GRU), feed-forward network (FFN), projection (Proj), and mean absolute error (MAE).

$P_w \in \mathbb{R}^{L_w \times N}$, which hold the value of every sensor at each calendar slot, with $L_d = 288$ and $L_w = 7L_d$ for five-minute data. Both tables are initialized from slot means of the training data and refined by road aggregation followed by temporal attention over slots and spatial attention over sensors, which gives $\widetilde{P}_d$ and $\widetilde{P}_w$. For timestamp s with daily slot $i_d(s)$ and weekly slot $i_w(s)$, the base field and the deviation field are

$$b_{s,i} = \widetilde{P}_d[i_d(s), i] + \widetilde{P}_w[i_w(s), i], \qquad R = X - B^{\text{in}}, \tag{3}$$

where $B^{\text{in}} \in \mathbb{R}^{T_1 \times N}$ collects $b_{s,i}$ over the historical steps and the deviations of missing observations are set to zero. Calendar positions are known in advance, so the same lookup gives $B^{\text{out}} \in \mathbb{R}^{T_2 \times N}$ for the prediction steps.

Trajectory-conditioned assignments. A deviation observed at one step still does not identify the regime: a sensor entering congestion and a sensor recovering from it can show the same deviation. We therefore assign sensors by their deviation trajectories. For sensor i, let $\mathbf{a}_i$ collect its deviation history, calendar features, and validity indicators. A routing GRU encodes $\mathbf{a}_i$ into a summary $\zeta_i \in \mathbb{R}^C$ once per input window. Each layer and head map the stacked summaries $\zeta \in \mathbb{R}^{N \times C}$ to soft assignments through a membership projection g_θ and a learned temperature $\tau > 0$:

$$S = \text{softmax}_M\big(g_\theta(\zeta)/\tau\big) \in \mathbb{R}^{N \times M}. \tag{4}$$

We omit the layer and head indices of S for readability. Row i of S evaluates the memberships $S_m(\xi_i; r)$ of Eq. (2) at sensor i. Because ζ_i summarizes deviations from the base field, sensors are grouped by how they currently depart from their recurring behavior rather than by their flow levels or road positions. The assignments stay fixed across the input window, so each latent state follows the same group of sensors and gives the latent evolution a coherent sequence to model.

2.3 REGIME-SEPARATED LATENT EVOLUTION

Joint field and local operators. This module addresses challenge (b) by evolving the latent states without mixing them, while local operators retain location-specific detail. A state GRU, separate from the routing encoder, summarizes each sensor's history, and for each prediction step a query adds a step embedding and the calendar features of the target time to this summary. The lifted historical deviations and the T_2 queries form the joint field $U^0 \in \mathbb{R}^{(T_1+T_2)\times N\times C}$, so historical and future positions are updated by the same layers. Each field layer first refines every sensor locally. The road update uses the sigmoid linear unit (SiLU) activation. At every joint position s, it gives $\widetilde{U}^\ell_s = U^\ell_s + \mathrm{SiLU}(U^\ell_s W^\ell_{\mathrm{self}} + AU^\ell_s W^\ell_{\mathrm{nbr}} + b^\ell_g)$. This update exchanges information only between adjacent sensors and acts as a residual correction, so sharing beyond the road neighborhood passes through the regime-dependent latent kernel. A temporal update then follows the pole–residue formulation of the Laplace neural operator (Cao et al., 2024): each layer learns K complex poles p_{ck} and residues γ_{ck} per channel together with their conjugates, which define the transfer function $\mathcal{H}_c(\mathrm{i}\omega) = \sum_{k=1}^{2K} \gamma_{ck}/(\mathrm{i}\omega - p_{ck})$ with $\mathrm{Re}(p_{ck}) < -m_p$ for a margin $m_p > 0$, so every mode decays. The temporal update uses the Gaussian error linear unit (GELU) activation. Let $\mathcal{F}_t$ denote the discrete Fourier transform (DFT) along the joint sequence. The update is

$$\mathcal{T}_\ell(U) = \left[\mathrm{Re}\,\mathcal{F}_t^{-1}\big(\mathcal{H}^\ell \odot \mathcal{F}_t(U)\big) + T^\ell_{\mathrm{tr}}(U)\right] W^\ell_{\mathrm{mix}}, \qquad \bar{U}^\ell = \widetilde{U}^\ell + \mathrm{GELU}\Big(\mathcal{T}_\ell(\widetilde{U}^\ell)\Big), \tag{5}$$

where the transient term T^ℓ_{tr} cancels the wrap-around of frequency-domain filtering, so that the update equals the response of the same stable system started from rest. Appendix D derives the transient term.

Pooling within latent states. Let $\mathcal{O}_s = \{i : \chi_{s,i} = 1\}$ contain the sensors observed at historical position s, and let $v_{s,i} \in \mathbb{R}^d$ be a value projection of $\bar{U}^\ell_{s,i}$ with head width d. Each latent state is the assignment-weighted average of its observed sensors:

$$z_{s,m} = \frac{\sum_{i\in\mathcal{O}_s} S_{im} v_{s,i}}{\sum_{i\in\mathcal{O}_s} S_{im}}, \qquad m = 1,\dots,M. \tag{6}$$

This is the quadrature of the integral in Eq. (2) over the sensors observed at step s. A missing sensor leaves the average instead of pulling it toward zero, and a step without observed sensors gives $z_{s,m} = 0$. Repeating the pooling at every historical step yields $Z \in \mathbb{R}^{T_1\times M\times d}$, which contains one sequence per latent state.

Independent evolution and redistribution. The evolution Φ_ℓ acts on each latent sequence separately. A GRU whose parameters are shared across latent states and heads within a layer encodes the sequence, and a linear projection of its final state generates all future steps:

$$z'_{1:T_1,m} = \mathrm{GRU}_{\mathrm{lat}}(z_{1:T_1,m}), \qquad z'_{T_1+1:T_1+T_2,m} = \mathrm{reshape}(W_f z'_{T_1,m} + b_f). \tag{7}$$

Shared parameters apply one temporal model to all latent states, while separate hidden states keep their dynamics apart; no operation mixes different latent states. The evolved states return through the same assignments: at joint position s, with $Z'_s \in \mathbb{R}^{M\times d}$ stacking the evolved states, a head outputs SZ'_sW_V, so each sensor receives its own mixture of latent trajectories. We concatenate the outputs of all heads and project them into the local width. We add the result to the local features. An FFN with layer normalization (LN) produces $U^{\ell+1}$.

2.4 WHEN SHARING HELPS AND WHY LATENT STATES SHOULD NOT MIX

For a fixed assignment and with the latent evolution set aside, pooling followed by redistribution applies the kernel $P_S = SD_S^{-1}S^\top$ at each step, where $D_S = \mathrm{diag}(d_1,\dots,d_M)$ holds the state masses $d_m = \sum_i S_{im} > 0$. This kernel has rank at most M. We analyse it on L positions for one feature channel, with observations $v = \mu + \epsilon$, a fixed signal $\mu \in \mathbb{R}^L$, $\mathbb{E}[\epsilon] = 0$, and $\mathrm{Cov}(\epsilon) = \sigma^2 I$. The signal μ is what the operator should reconstruct and encodes how the deviation at each position evolves. The reconstruction risk of an operator P is $\mathcal{R}(P) = \mathbb{E}\|Pv - \mu\|_2^2$, so keeping the observations gives $\mathcal{R}(I) = \sigma^2 L$. We measure the inconsistency within states and the noise dimensions that sharing retains by

$$\mathcal{J}_S(\mu) = \min_{z\in\mathbb{R}^M} \sum_{i=1}^{L}\sum_{m=1}^{M} S_{im}(\mu_i - z_m)^2, \qquad \nu_{\mathrm{eff}}(S) = \|P_S\|_F^2. \tag{8}$$

Theorem 2.1 (Bias–noise trade-off of sharing)**.** *For a fixed soft assignment,*

$$\mathcal{R}(P_S) - \mathcal{R}(I) = \|(I - P_S)\mu\|_2^2 - \sigma^2\big[L - \nu_{\mathrm{eff}}(S)\big], \tag{9}$$

where the distortion satisfies $\|(I - P_S)\mu\|_2^2 \le \mathcal{J}_S(\mu)$*, with equality for hard assignments, and* $\nu_{\mathrm{eff}}(S) \le \min(M, L)$*. Sharing therefore lowers the risk if and only if its distortion is below the removed noise* $\sigma^2[L - \nu_{\mathrm{eff}}(S)]$*, and in particular whenever*

$$\mathcal{J}_S(\mu) < \sigma^2\big[L - \nu_{\mathrm{eff}}(S)\big]. \tag{10}$$

The latent-attention alternative inserts a map $A_Z \in \mathbb{R}^{M\times M}$ between pooling and redistribution, giving $P_A = SA_ZD_S^{-1}S^\top$. The next result isolates what this map costs.

Proposition 2.2 (Cost of mixing latent states)**.** *Let* S *be a hard assignment with state means* $z^* = D_S^{-1}S^\top\mu$*. Then*

$$\begin{aligned}\mathcal{R}(P_A) - \mathcal{R}(P_S) &= \sum_{m=1}^{M} d_m\big((A_Zz^*)_m - z_m^*\big)^2 - \sigma^2\big(M - \|P_A\|_F^2\big)\\ &\ge \sum_{m=1}^{M} d_m\big((A_Zz^*)_m - z_m^*\big)^2 - \sigma^2M.\end{aligned} \tag{11}$$

Under uniform mixing, $A_Z = M^{-1}\mathbf{1}\mathbf{1}^\top$*, the first term is the weighted dispersion* $\sum_m d_m(z_m^* - \bar{z})^2$ *of the state means around their average* $\bar{z}$*.*

Theorem 2.1 states when sharing helps: the distortion from pooling different signals must stay below the noise that pooling removes. Grouping sensors by reference-relative trajectories keeps the within-state inconsistency, which bounds this distortion, small, as challenge (a) requires. A state-independent road kernel pays the same kind of bias for every neighbor pair that straddles a regime boundary. Proposition 2.2 states why the states should not interact: mixing removes at most $M\sigma^2$ of noise energy, independent of the number of sensors, whereas the dispersion it adds grows with the state masses d_m, so StaLiNO evolves every state independently, as challenge (b) requires. Appendix B gives the proofs and the analysis of state-independent kernels, and the risk decomposition and bounds of Theorem 2.1 are machine-checked in Lean 4.

2.5 Prediction and Training

The last field layer produces representations at the prediction positions, and a shared linear readout maps them to corrections $\Delta_{h,i}$. The forecast adds these corrections to two anchors:

$$\hat{y}_{t+h,i} = b_{t+h,i} + r_i^{\mathrm{last}} + \Delta_{h,i}, \tag{12}$$

where $b_{t+h,i}$ is the future base field from B^{out} and r_i^{last} is the latest valid deviation of sensor i, set to zero when the sensor has no valid history. The base field supplies the recurring change expected at the target time, the latest deviation anchors the forecast to the current state, and the correction models how the deviation evolves. All components are trained jointly by minimizing MAE over observed targets. Appendix E details the base-field refinement, the joint field, and the training configuration.

3 Experiments

We answer four questions. **Q1:** How accurate is StaLiNO compared with spatiotemporal and decomposition-based forecasters? **Q2:** Which parts of the latent operator matter, and do they depend on each other? **Q3:** Do the latent states identify traffic regimes, and when does sharing across sensors help? **Q4:** How robust is StaLiNO to missing and perturbed inputs, and what does it cost? Appendix F gives the protocols and complete results of all experiments.

3.1 Experimental Setup

Datasets. We use four datasets from the Caltrans Performance Measurement System (PeMS), PEMS03, PEMS04, PEMS07, and PEMS08, which contain 358, 307, 883, and 170 sensors sampled every five minutes and are split chronologically into training, validation, and test sets at a 6:2:2

Table 1: StaLiNO attains the lowest MAE on all four PeMS datasets. Bold and underline denote the best and second-best values; MAPE is in percent.

Category	Method	PEMS03			PEMS04			PEMS07			PEMS08		
		MAE	RMSE	MAPE	MAE	RMSE	MAPE	MAE	RMSE	MAPE	MAE	RMSE	MAPE
Spatiotemporal forecasting	STGCN	17.58	30.14	17.17	22.70	35.55	14.74	25.52	38.88	11.24	18.05	27.93	11.40
	DCRNN	18.27	30.36	19.00	24.77	38.21	17.22	25.34	38.59	11.69	17.97	27.97	11.54
	GWNet	19.92	33.11	19.44	25.51	39.82	17.34	26.82	42.91	12.26	19.23	31.14	12.73
	ASTGCN	17.83	29.66	19.43	23.02	35.30	16.48	28.15	42.63	13.94	18.70	28.20	13.14
	MTGNN	17.09	25.91	17.40	20.01	31.95	14.18	24.02	35.84	12.51	15.09	23.98	10.31
	STGODE	16.55	27.92	16.85	20.96	32.84	13.84	23.00	37.58	10.21	16.88	25.89	10.72
	ST-WA	15.19	26.66	15.90	19.10	31.10	**12.65**	20.80	34.22	8.85	15.58	24.67	10.08
	DGCRN	<u>14.83</u>	25.93	<u>15.47</u>	18.92	30.68	12.82	20.56	**33.26**	9.05	14.65	24.31	9.47
	STPGNN	14.99	25.96	15.59	18.87	30.22	13.16	21.77	35.46	9.41	14.77	23.92	9.66
Decomposition based	D^2STGNN	15.14	26.65	**15.31**	<u>18.50</u>	<u>29.99</u>	12.86	<u>19.66</u>	<u>33.27</u>	8.51	14.41	24.25	9.42
	STWave	14.86	26.21	15.92	18.57	30.31	<u>12.69</u>	19.76	33.77	**8.28**	<u>13.85</u>	<u>23.70</u>	9.31
	CycleNet-D	17.53	27.56	20.06	23.44	36.26	17.00	25.66	39.93	11.91	18.90	29.28	13.45
	CycleNet-W	18.34	28.58	20.31	24.32	38.24	18.08	24.99	40.38	11.05	18.40	29.87	12.01
	STDN	16.07	27.64	17.80	18.75	31.07	13.27	23.00	36.10	10.31	14.86	24.75	10.34
	HyperD	14.86	<u>23.83</u>	16.50	18.77	30.57	13.16	<u>19.66</u>	33.60	<u>8.31</u>	14.09	23.96	<u>9.30</u>
Proposed	StaLiNO (Ours)	**14.49**	**23.40**	16.12	**18.27**	**29.94**	13.17	**19.63**	33.33	8.33	**13.81**	**23.65**	**9.17**

ratio. The past 12 steps, one hour, are used to forecast the next 12 steps, and we report MAE, root mean squared error (RMSE), and mean absolute percentage error (MAPE) over the prediction window.

Baselines. **(a) Spatiotemporal forecasters:** STGCN (Yu et al., 2018), DCRNN, GWNet (Wu et al., 2019), ASTGCN (Guo et al., 2019), MTGNN (Wu et al., 2020), STGODE (Fang et al., 2021), ST-WA (Cirstea et al., 2022), DGCRN (Li et al., 2023a), and STPGNN (Kong et al., 2024). **(b) Decomposition-based forecasters:** D^2STGNN (Shao et al., 2022), STWave (Fang et al., 2023), CycleNet-D and CycleNet-W (Lin et al., 2024), STDN (Cao et al., 2025), and HyperD (Shao et al., 2026).

Implementation. StaLiNO uses three field layers, four heads, hidden width 64, $M = 16$ latent states per head, and $K = 8$ temporal modes, and keeps the checkpoint with the lowest validation MAE. The main StaLiNO results are averages over five independent runs, all variants share this training protocol, and normalization and base-field initialization use only the training split.

3.2 Main Results

① **StaLiNO attains the lowest MAE on all four datasets.** To answer **Q1**, Table 1 compares StaLiNO with all baselines. StaLiNO also obtains the lowest RMSE on PEMS03, PEMS04, and PEMS08. The MAE margin over the strongest baseline is largest on PEMS03, at 2.3%.

3.3 Mechanism of the Latent Operator

② **Latent evolution improves tracked states and degrades per-step states.** To answer **Q2**, Table 2 varies how sensors are assigned to latent states and what the latent step between pooling and redistribution does: persistence repeats each state's last pooled value, and mixing is attention among states. On PEMS03, the latent GRU reduces MAE by 2.3% with trajectory assignments and raises it by 2.5% with token assignments, and PEMS08 shows the same interaction. Trajectory assignments alone trail token assignments by 1.5%, so their value lies in making evolution possible: a state whose members change at every step has no coherent sequence to model. Persistence stays above StaLiNO on both datasets, so the learned evolution matters in addition to the tracking.

③ **Mixing latent states raises error under both assignment schemes.** Attention among latent states raises PEMS03 MAE by 2.4% with token assignments and by 2.0% with trajectory assignments, and it raises PEMS08 MAE under both schemes, as Proposition 2.2 predicts. A trained latent-attention model has a normalized attention entropy of 0.988, so its mixing is close to uni-

Table 2: Retrained variants of the latent operator. Token assignments are computed per space–time token from current features, trajectory assignments once per window from each sensor's deviation history; Δ is the PEMS03 MAE change relative to StaLiNO.

			PEMS03			PEMS08
Variant	Assignment	Latent step	MAE	RMSE	Δ	MAE
No latent sharing	n/a	n/a	15.089	24.369	+4.1%	13.917
Token sharing	token	none	14.612	23.665	+0.9%	13.866
+ mixing	token	attention	14.958	24.121	+3.2%	13.99
+ evolution	token	GRU	14.983	24.513	+3.4%	14.00
Trajectory sharing	trajectory	none	14.834	23.861	+2.4%	13.891
+ persistence	trajectory	persistence	14.727	23.741	+1.6%	13.931
+ mixing and evolution	trajectory	attention, GRU	14.772	23.963	+2.0%	13.975
+ evolution (StaLiNO)	trajectory	GRU	**14.488**	**23.404**	ref	**13.814**

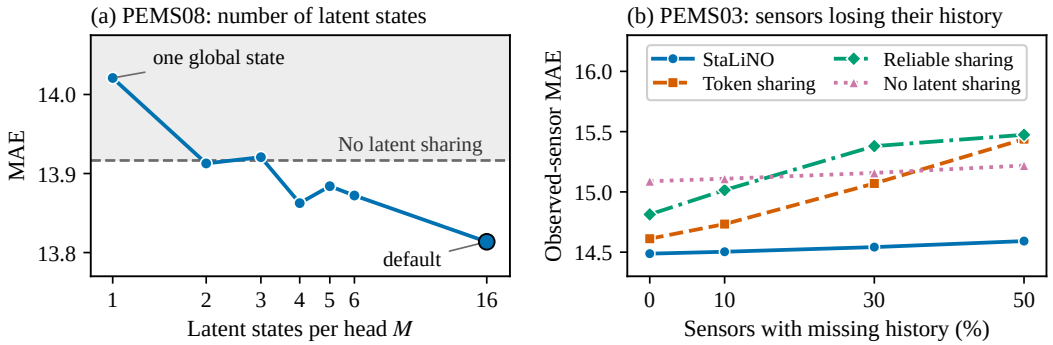


Figure 3: (a) PEMS08 MAE against the number of latent states per head; the dashed line marks the variant without latent sharing. (b) Observed-sensor MAE on PEMS03 when a fraction of sensors lose their complete history at test time.

form and costs the weighted dispersion of the state means; mixing raises the mean cosine similarity between different states from 0.44 to 0.96. The removable noise is at most $M\sigma^2$, whereas the dispersion grows with the state masses, which average $N/M \approx 22$ sensors on PEMS03 and 11 on PEMS08, consistent with the larger cost on PEMS03.

3.4 Traffic Regimes and Selective Sharing

④ **Sharing helps once the states separate sensors with different dynamics.** To answer **Q3**, Figure 3a varies the number of latent states on PEMS08 against the variant without latent sharing. A single state per head, which pools all sensors regardless of regime, is 0.7% worse than no sharing; two or three states only match it, and four or more improve on it, with $M = 4$ within 0.35% of $M = 16$. This is the trade-off of Theorem 2.1: with one state, the inconsistency $\mathcal{J}_S$ is the full spread of the deviation field, and separating sensors whose deviations evolve differently lowers the distortion below the noise that sharing removes.

⑤ **The latent states align with the free-flow and congested branches without access to occupancy.** PEMS04 and PEMS08 also record occupancy, which StaLiNO never receives. We label each sensor and window as congested when its occupancy exceeds the critical occupancy of its fundamental diagram and compare this label with the dominant latent state in each head. Averaged over heads, the adjusted mutual information (AMI) between state and label is 0.48 on PEMS04 and 0.46 on PEMS08, and the homogeneity of the states, which reaches one when every state holds a single regime, is 0.63 and 0.61. The two flow-based controls reach at most 0.38 in AMI and 0.41 in homogeneity. The learned states therefore align more closely with the occupancy-derived regime labels than groups formed from raw flow.

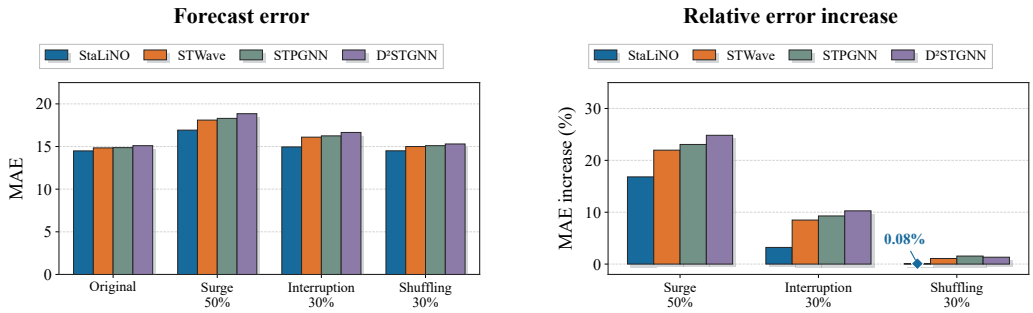


Figure 4: StaLiNO attains the lowest MAE under all short input perturbations on PEMS03. Left: absolute MAE. Right: MAE increase relative to each model's clean-input error. Each perturbation affects three consecutive historical steps. Bars show StaLiNO values and the midpoints of baseline ranges; a diamond marks the 0.08% shuffling increase of StaLiNO.

⑥ **The assignments rely on deviations from the base field.** Without the base field, the routing encoder receives raw flow, in which recurring demand masks the deviations that distinguish the regimes. PEMS08 MAE then rises by 11.4%, the largest effect in the PEMS08 component study. The base field thus provides the reference that challenge (a) requires, beyond the recurring change it supplies to the forecast in Eq. (12).

3.5 Robustness and Computational Cost

⑦ **Pooling over observed sensors keeps their forecasts stable when other sensors lose their history.** To answer **Q4**, we hide the complete history of selected sensors at test time without retraining. As the fraction of PEMS03 sensors without history rises from 0% to 50%, the observed-sensor MAE of StaLiNO rises by 0.7%, against 5.7% for token sharing and 4.5% for reliable sharing (Figure 3b). Because Eq. (6) normalizes over the sensors observed at each step, missing values never enter the states that observed sensors receive.

⑧ **StaLiNO stays accurate under short input perturbations.** We also perturb a three-step segment in the history of selected PEMS03 sensors: surges scale the observations by 1.5, interruptions remove them, and shuffling permutes them. Under every perturbation, StaLiNO keeps the lowest MAE in Figure 4, and its MAE rises by 3.2% under interruptions, less than that of any baseline, and by only 0.08% under shuffling. Interrupted observations leave the pooling of Eq. (6), and an interrupted sensor is forecast from the base field and its latest valid deviation, so a short gap does not inject corrupted values into the latent states or the forecast anchors.

⑨ **Latent sharing improves accuracy with a small parameter increase.** On PEMS03 with an NVIDIA RTX 4090 and a batch of 16 windows, StaLiNO needs 23.0 ms and 0.50 GB of peak memory per batch. Relative to the variant without latent sharing, latent sharing and its routing encoder add 4.2 ms and 0.07 million parameters. Removing them raises MAE by 4.1%.

4 Conclusion

We proposed StaLiNO, a latent neural operator for traffic forecasting. Reference-Relative Regime Assignment infers latent states from deviation trajectories relative to a learned daily and weekly base field. The assignments remain fixed across the input window. Regime-Separated Latent Evolution evolves each state independently before redistribution through the same assignments. Our analysis characterizes the conditions for latent sharing to lower reconstruction risk. Experiments on four PeMS benchmarks demonstrate improved accuracy from tracked latent evolution. StaLiNO remains accurate under missing and perturbed inputs. Its states align with the free-flow and congested branches without occupancy inputs.

REPRODUCIBILITY STATEMENT

Section 3.1 and Appendices E and F give the data splits, preprocessing, architecture, hyperparameters, validity conventions, and evaluation protocols, including the masking and perturbation procedures. Appendices B–D contain complete proofs, and Appendix B.7 maps each formalized result to its Lean 4 module.

AI USE STATEMENT

In this work, we used ChatGPT 5.6 Sol to assist with English prose editing and the search for relevant literature. We also used it to review the presentation of the methodology and experiments and discuss the interpretation of results. The authors reviewed and revised AI-assisted material before including it in the manuscript. We take responsibility for the final content of this work, including its text, citations, claims, numerical results, and figures.

REFERENCES

Benedikt Alkin, Andreas Fürst, Simon Schmid, Lukas Gruber, Markus Holzleitner, and Johannes Brandstetter. Universal Physics Transformers: A Framework for Efficiently Scaling Neural Operators. In *Advances in Neural Information Processing Systems*, volume 37, 2024. URL https://proceedings.neurips.cc/paper_files/paper/2024/hash/2cd36d327f33d47b372d4711edd08de0-Abstract-Conference.html.

Lei Bai, Lina Yao, Can Li, Xianzhi Wang, and Can Wang. Adaptive Graph Convolutional Recurrent Network for Traffic Forecasting. In *Advances in Neural Information Processing Systems*, volume 33, 2020. URL https://proceedings.neurips.cc/paper/2020/hash/ce1aad92b939420fc17005e5461e6f48-Abstract.html.

California Department of Transportation. Traffic Operations Manual, Chapter 215: Ramp Metering Systems, Part 4: Ramp Metering Operations. Technical report, California Department of Transportation, April 2025. URL https://dot.ca.gov/-/media/dot-media/programs/traffic-operations/documents/trafficops/202504-ch-215-pt-4-ramp-metering-operations-a11y.pdf.

Lingxiao Cao, Bin Wang, Guiyuan Jiang, Yanwei Yu, and Junyu Dong. Spatiotemporal-Aware Trend-Seasonality Decomposition Network for Traffic Flow Forecasting. *Proceedings of the AAAI Conference on Artificial Intelligence*, 39(11):11463–11471, 2025. doi: 10.1609/aaai.v39i11.33247. URL https://ojs.aaai.org/index.php/AAAI/article/view/33247.

Qianying Cao, Somdatta Goswami, and George Em Karniadakis. Laplace Neural Operator for Solving Differential Equations. *Nature Machine Intelligence*, 6:631–640, 2024. URL https://www.nature.com/articles/s42256-024-00844-4.

Razvan-Gabriel Cirstea, Bin Yang, Chenjuan Guo, Tung Kieu, and Shirui Pan. Towards Spatio-Temporal Aware Traffic Time Series Forecasting. In *2022 IEEE 38th International Conference on Data Engineering*, pp. 2900–2913, 2022. doi: 10.1109/ICDE53745.2022.00262. URL https://doi.org/10.1109/ICDE53745.2022.00262.

Yuchen Fang, Yanjun Qin, Haiyong Luo, Fang Zhao, Bingbing Xu, Liang Zeng, and Chenxing Wang. When Spatio-Temporal Meet Wavelets: Disentangled Traffic Forecasting via Efficient Spectral Graph Attention Networks. In *2023 IEEE 39th International Conference on Data Engineering*, pp. 517–529, 2023. doi: 10.1109/ICDE55515.2023.00046. URL https://doi.org/10.1109/ICDE55515.2023.00046.

Yuchen Fang, Yuxuan Liang, Bo Hui, Zezhi Shao, Liwei Deng, Xu Liu, Xinke Jiang, and Kai Zheng. Efficient Large-Scale Traffic Forecasting with Transformers: A Spatial Data Management Perspective. In *Proceedings of the 31st ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 2025. doi: 10.1145/3690624.3709177. URL https://doi.org/10.1145/3690624.3709177.

Zheng Fang, Qingqing Long, Guojie Song, and Kunqing Xie. Spatial-Temporal Graph ODE Networks for Traffic Flow Forecasting. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*, pp. 364–373, 2021. doi: 10.1145/3447548.3467430. URL https://doi.org/10.1145/3447548.3467430.

Yan Ge, Zhengyang Zhou, Qihe Huang, Yuxuan Liang, and Yang Wang. Enabling Arbitrary Inference in Spatiotemporal Dynamic Systems: A Physics-Inspired Perspective. In *International Conference on Learning Representations*, 2026.

Shengnan Guo, Youfang Lin, Ning Feng, Chao Song, and Huaiyu Wan. Attention Based Spatial-Temporal Graph Convolutional Networks for Traffic Flow Forecasting. *Proceedings of the AAAI Conference on Artificial Intelligence*, 33(1):922–929, 2019. doi: 10.1609/aaai.v33i01.3301922. URL https://ojs.aaai.org/index.php/AAAI/article/view/3881.

Jiahao Ji, Jingyuan Wang, Zhe Jiang, Jiawei Jiang, and Hu Zhang. STDEN: Towards Physics-Guided Neural Networks for Traffic Flow Prediction. *Proceedings of the AAAI Conference on Artificial Intelligence*, 36(4):4048–4056, 2022. doi: 10.1609/aaai.v36i4.20322. URL https://ojs.aaai.org/index.php/AAAI/article/view/20322.

Jiawei Jiang, Chengkai Han, Wayne Xin Zhao, and Jingyuan Wang. PDFormer: Propagation Delay-Aware Dynamic Long-Range Transformer for Traffic Flow Prediction. *Proceedings of the AAAI Conference on Artificial Intelligence*, 37(4):4365–4373, 2023a. doi: 10.1609/aaai.v37i4.25556. URL https://ojs.aaai.org/index.php/AAAI/article/view/25556.

Renhe Jiang, Zhaonan Wang, Jiawei Yong, Puneet Jeph, Quanjun Chen, Yasumasa Kobayashi, Xuan Song, Shintaro Fukushima, and Toyotaro Suzumura. Spatio-Temporal Meta-Graph Learning for Traffic Forecasting. *Proceedings of the AAAI Conference on Artificial Intelligence*, 37(7):8078–8086, 2023b. doi: 10.1609/aaai.v37i7.25976. URL https://ojs.aaai.org/index.php/AAAI/article/view/25976.

Weiyang Kong, Ziyu Guo, and Yubao Liu. Spatio-Temporal Pivotal Graph Neural Networks for Traffic Flow Forecasting. *Proceedings of the AAAI Conference on Artificial Intelligence*, 38(8):8627–8635, 2024. doi: 10.1609/aaai.v38i8.28707. URL https://ojs.aaai.org/index.php/AAAI/article/view/28707.

Nikola Kovachki, Zongyi Li, Burigede Liu, Kamyar Azizzadenesheli, Kaushik Bhattacharya, Andrew Stuart, and Anima Anandkumar. Neural Operator: Learning Maps between Function Spaces with Applications to PDEs. *Journal of Machine Learning Research*, 24(89):1–97, 2023. URL https://jmlr.org/papers/v24/21-1524.html.

Shiyong Lan, Yitong Ma, Weikang Huang, Wenwu Wang, Hongyu Yang, and Pyang Li. DSTAGNN: Dynamic Spatial-Temporal Aware Graph Neural Network for Traffic Flow Forecasting. In *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pp. 11906–11917, 2022. URL https://proceedings.mlr.press/v162/lan22a.html.

Fuxian Li, Jie Feng, Huan Yan, Guangyin Jin, Fan Yang, Funing Sun, Depeng Jin, and Yong Li. Dynamic Graph Convolutional Recurrent Network for Traffic Prediction: Benchmark and Solution. *ACM Transactions on Knowledge Discovery from Data*, 17(1):9:1–9:21, 2023a. doi: 10.1145/3532611. URL https://doi.org/10.1145/3532611.

Zongyi Li, Nikola Kovachki, Chris Choy, Boyi Li, Jean Kossaifi, Shourya Otta, Mohammad Amin Nabian, Maximilian Stadler, Christian Hundt, Kamyar Azizzadenesheli, and Animashree Anandkumar. Geometry-Informed Neural Operator for Large-Scale 3D PDEs. In *Advances in Neural Information Processing Systems*, volume 36, pp. 35836–35854, 2023b. doi: 10.52202/075280-1556. URL https://proceedings.neurips.cc/paper_files/paper/2023/hash/70518ea42831f02afc3a2828993935ad-Abstract-Conference.html.

M. J. Lighthill and G. B. Whitham. On Kinematic Waves II. A Theory of Traffic Flow on Long Crowded Roads. *Proceedings of the Royal Society of London. Series A. Mathematical and Physical Sciences*, 229(1178):317–345, 1955. doi: 10.1098/rspa.1955.0089. URL https://doi.org/10.1098/rspa.1955.0089.

Shengsheng Lin, Weiwei Lin, Xinyi Hu, Wentai Wu, Ruichao Mo, and Haocheng Zhong. CycleNet: Enhancing Time Series Forecasting through Modeling Periodic Patterns. In *Advances in Neural Information Processing Systems*, volume 37, 2024. doi: 10.52202/079017-3373. URL https://proceedings.neurips.cc/paper_files/paper/2024/hash/bfe7998398779dde03cad7a73b1f81b6-Abstract-Conference.html.

Kun Liu, Renjun Jia, Ruifeng Yang, Xirui Zeng, Yuqi Liang, and Cen Chen. Are Global Dependencies Necessary? Scalable Time Series Forecasting via Local Cross-Variate Modeling. In *International Conference on Learning Representations*, 2026. URL https://proceedings.iclr.cc/paper_files/paper/2026/hash/a832b65da26c4867a8581cb13d46ad6d-Abstract-Conference.html.

Jiawei Lu, Chongnan Li, Xin Bruce Wu, and Xuesong Simon Zhou. Physics-Informed Neural Networks for Integrated Traffic State and Queue Profile Estimation: A Differentiable Programming Approach on Layered Computational Graphs. *Transportation Research Part C: Emerging Technologies*, 153:104224, 2023. doi: 10.1016/j.trc.2023.104224. URL https://www.sciencedirect.com/science/article/abs/pii/S0968090X23002139.

Sepehr Mousavi, Shizheng Wen, Levi E. Lingsch, Maximilian Herde, Bogdan Raonic, and Siddhartha Mishra. RIGNO: A Graph-Based Framework for Robust and Accurate Operator Learning for PDEs on Arbitrary Domains. In *Advances in Neural Information Processing Systems*, volume 38, 2025. doi: 10.52202/085713-5018. URL https://proceedings.nips.cc/paper_files/paper/2025/hash/dcb91f43033bb1d367d1848806dee98d-Abstract-Conference.html.

Paul I. Richards. Shock Waves on the Highway. *Operations Research*, 4(1):42–51, 1956. doi: 10.1287/opre.4.1.42. URL https://pubsonline.informs.org/doi/10.1287/opre.4.1.42.

Louis Serrano, Lise Le Boudec, Armand Kassaï Koupaï, Thomas X Wang, Yuan Yin, Jean-Noël Vittaut, and Patrick Gallinari. Operator Learning with Neural Fields: Tackling PDEs on General Geometries. In *Advances in Neural Information Processing Systems*, volume 36, 2023. URL https://proceedings.neurips.cc/paper_files/paper/2023/hash/df54302388bbc145aacaa1a54a4a5933-Abstract-Conference.html.

Minlan Shao, Zijian Zhang, Yili Wang, Yiwei Dai, Xu Shen, and Xin Wang. HyperD: Hybrid Periodicity Decoupling Framework for Traffic Forecasting. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pp. 15689–15697, 2026. doi: 10.1609/aaai.v40i18.38599. URL https://ojs.aaai.org/index.php/AAAI/article/view/38599.

Zezhi Shao, Zhao Zhang, Wei Wei, Fei Wang, Yongjun Xu, Xin Cao, and Christian S. Jensen. Decoupled Dynamic Spatial-Temporal Graph Neural Network for Traffic Forecasting. *Proceedings of the VLDB Endowment*, 15(11):2733–2746, 2022. doi: 10.14778/3551793.3551827. URL https://www.vldb.org/pvldb/vol15/p2733-shao.pdf.

Transportation Research Board. *Highway Capacity Manual, 6th Edition: A Guide for Multimodal Mobility Analysis*. The National Academies Press, Washington, DC, 2016. doi: 10.17226/24798. URL https://doi.org/10.17226/24798.

Martin Treiber and Arne Kesting. *Traffic Flow Dynamics: Data, Models and Simulation*. Springer, 2013.

Tian Wang and Chuang Wang. Latent Neural Operator for Solving Forward and Inverse PDE Problems. In *Advances in Neural Information Processing Systems*, volume 37, 2024.

Haixu Wu, Huakun Luo, Haowen Wang, Jianmin Wang, and Mingsheng Long. Transolver: A Fast Transformer Solver for PDEs on General Geometries. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, 2024.

Zonghan Wu, Shirui Pan, Guodong Long, Jing Jiang, and Chengqi Zhang. Graph WaveNet for Deep Spatial-Temporal Graph Modeling. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence*, pp. 1907–1913, 2019. doi: 10.24963/ijcai.2019/264. URL https://www.ijcai.org/proceedings/2019/264.

Zonghan Wu, Shirui Pan, Guodong Long, Jing Jiang, Xiaojun Chang, and Chengqi Zhang. Connecting the Dots: Multivariate Time Series Forecasting with Graph Neural Networks. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pp. 753–763, 2020. doi: 10.1145/3394486.3403118. URL https://doi.org/10.1145/3394486.3403118.

Bing Yu, Haoteng Yin, and Zhanxing Zhu. Spatio-Temporal Graph Convolutional Networks: A Deep Learning Framework for Traffic Forecasting. In *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence*, pp. 3634–3640, 2018. doi: 10.24963/ijcai.2018/505. URL https://www.ijcai.org/proceedings/2018/505.

Jinlei Zhang, Shuai Mao, Lixing Yang, Wei Ma, Shukai Li, and Ziyou Gao. Physics-Informed Deep Learning for Traffic State Estimation Based on the Traffic Flow Model and Computational Graph Method. *Information Fusion*, 101:101971, 2024. doi: 10.1016/j.inffus.2023.101971. URL https://www.sciencedirect.com/science/article/pii/S1566253523002877.

Yihuai Zhang, Ruiguo Zhong, and Huan Yu. Mitigating Stop-and-Go Traffic Congestion with Operator Learning. *Transportation Research Part C: Emerging Technologies*, 170:104928, 2025. doi: 10.1016/j.trc.2024.104928. URL https://www.sciencedirect.com/science/article/pii/S0968090X24004492.

A NOTATION

Table 3: Main notation.

Symbol	Meaning
N, A	number of sensors; road adjacency matrix
T_1, T_2, t	input length; prediction length; last observed timestamp
X, χ, κ_s	observed flow $\mathbb{R}^{T_1 \times N}$; validity indicators; calendar features
$\widehat{Y}$, $\mathcal{F}_\theta$	forecast $\mathbb{R}^{T_2 \times N}$; forecasting operator
ρ, q, ξ	traffic density; flux; road position
$b_{s,i}$, B^{in}, B^{out}	base field; its historical and future values
R, r	deviation field on the sensors and on the road domain
r_i^{last}, $\Delta_{h,i}$	latest valid deviation of sensor i; learned correction
ζ_i, S, τ	trajectory summary; soft assignment $\mathbb{R}^{N \times M}$; temperature
M, d_m	latent states per head; state mass $\sum_i S_{im}$
U^ℓ, C, d	joint field at layer ℓ; hidden width; head width
$\mathcal{O}_s$, $z_{s,m}$	sensors observed at step s; pooled latent state
Φ_ℓ, $S_m(\xi; r)$	latent evolution; membership function of state m
p_{ck}, γ_{ck}, m_p	poles; residues; stability margin
P_S, P_A	sharing kernel $SD_S^{-1}S^\top$; mixed kernel $SA_Z D_S^{-1} S^\top$
$\mathcal{R}$, μ	reconstruction risk; signal to reconstruct
$\mathcal{J}_S$, ν_{eff}, σ^2	within-state inconsistency; retained noise dimensions; noise variance

B THEORY OF LATENT SHARING

This section proves Theorem 2.1 and Proposition 2.2, relates the analysis to state-independent graph kernels, and describes the structure and cost of the sharing kernel.

B.1 SETTING AND RISK DECOMPOSITION

We use the setting of Section 2.4: a fixed assignment $S \in \mathbb{R}^{L \times M}$ with nonnegative entries, unit row sums, and positive column masses d_m, and observations $v = \mu + \epsilon$ with zero-mean isotropic noise. The risk difference in Eq. (9) has two terms. The first is the distortion from combining different signals, and the subtracted term is the reduction in noise energy, so sharing improves reconstruction exactly when the noise reduction outweighs the distortion. The following subsections bound both terms and prove Theorem 2.1.

B.2 OPTIMAL REPRESENTATIVES AND AGGREGATION BIAS

For a feature vector $u \in \mathbb{R}^L$, the pooled representative is $z_m^* = d_m^{-1} \sum_i S_{im} u_i$, and redistribution gives $(P_S u)_i = \sum_m S_{im} z_m^*$.

Proposition B.1 (Representative optimality and bias bound)**.** *The pooled representatives minimize the weighted within-state distortion. Moreover,*

$$\mathcal{J}_S(u) = u^\top (I - P_S) u, \qquad \|(I - P_S)u\|_2^2 \le \mathcal{J}_S(u). \tag{13}$$

Proof. The identity $\sum_i S_{im}(u_i - z_m^*) = 0$ eliminates the cross term when expanding around each pooled representative. Therefore,

$$\sum_{i,m} S_{im}(u_i - z_m)^2 = \sum_{i,m} S_{im}(u_i - z_m^*)^2 + \sum_m d_m (z_m - z_m^*)^2. \tag{14}$$

The last term is nonnegative and vanishes at $z = z^*$, proving optimality. Expanding the attained minimum and using the unit row sums gives $\mathcal{J}_S(u) = \|u\|_2^2 - \sum_m d_m (z_m^*)^2$. Since $u^\top P_S u = \sum_m d_m (z_m^*)^2$, this is the first identity in Eq. (13). For each position, the residual is $u_i - (P_S u)_i = \sum_m S_{im}(u_i - z_m^*)$. Jensen's inequality for the normalized weights gives $(u_i - (P_S u)_i)^2 \le \sum_m S_{im}(u_i - z_m^*)^2$, and summing over positions proves the bias bound. □

B.3 Noise Retention and Proof of Theorem 2.1

Proposition B.2 (Contraction and noise retention). *The sharing kernel satisfies* $0 \preceq P_S \preceq I$ *and*

$$\nu_{\text{eff}}(S) = \|P_S\|_F^2 \le \operatorname{tr}(P_S) \le \min(M, L). \tag{15}$$

Proof. The matrix P_S is symmetric. Its quadratic form is $u^\top P_S u = \sum_m d_m (z_m^*)^2 \ge 0$, while Proposition B.1 gives $u^\top (I - P_S)u = \mathcal{J}_S(u) \ge 0$. Hence $0 \preceq P_S \preceq I$. Applying Jensen's inequality to $(P_S u)_i = \sum_m S_{im} z_m^*$ and summing gives $\|P_S u\|_2^2 \le \sum_m d_m (z_m^*)^2 = u^\top P_S u$. Applying this inequality to every standard basis vector yields $\|P_S\|_F^2 \le \operatorname{tr}(P_S)$. Furthermore, $0 \le S_{im} \le 1$ implies

$$\operatorname{tr}(P_S) = \sum_m \frac{\sum_i S_{im}^2}{d_m} \le \sum_m \frac{\sum_i S_{im}}{d_m} = M. \tag{16}$$

Since $P_S \preceq I$, each diagonal entry is at most one, which also gives $\operatorname{tr}(P_S) \le L$. □

Proof of Theorem 2.1. Let ϵ have finite second moments, $\mathbb{E}[\epsilon] = 0$, and $\mathbb{E}[\epsilon\epsilon^\top] = \sigma^2 I$. For any fixed matrix $T \in \mathbb{R}^{r \times L}$ and target $y \in \mathbb{R}^r$, expanding the squared error gives

$$\mathbb{E}\|Tv - y\|_2^2 = \|T\mu - y\|_2^2 + \sigma^2 \|T\|_F^2, \tag{17}$$

because the cross term vanishes for zero-mean noise and $\mathbb{E}\|T\epsilon\|_2^2 = \operatorname{tr}(T\mathbb{E}[\epsilon\epsilon^\top]T^\top) = \sigma^2\|T\|_F^2$. Taking $T = P_S$ and $y = \mu$, then applying Proposition B.1, gives

$$\mathcal{R}(P_S) = \|(I - P_S)\mu\|_2^2 + \sigma^2 \nu_{\text{eff}}(S) \le \mathcal{J}_S(\mu) + \sigma^2 \nu_{\text{eff}}(S). \tag{18}$$

The identity operator has risk $\mathcal{R}(I) = \sigma^2 L$, and subtracting this value gives Eq. (9). Proposition B.2 supplies the bound on $\nu_{\text{eff}}(S)$. For a hard assignment, each position i belongs to one state $m(i)$ and $(P_S\mu)_i = z^*_{m(i)}$, so $\|(I - P_S)\mu\|_2^2 = \sum_i (\mu_i - z^*_{m(i)})^2 = \mathcal{J}_S(\mu)$. The right-hand side of Eq. (9) is negative exactly when the distortion is below $\sigma^2[L - \nu_{\text{eff}}(S)]$, and Eq. (10) implies this condition because the distortion is at most $\mathcal{J}_S(\mu)$. □

B.4 Cost of Mixing Latent States

Proof of Proposition 2.2. For a hard assignment, each position i belongs to exactly one state $m(i)$, and $S^\top S = D_S$. Write $y = A_Z z^*$. Then $(P_A\mu)_i = y_{m(i)}$, and because $\sum_{i:m(i)=m}(\mu_i - z_m^*) = 0$ for every m,

$$\|(I - P_A)\mu\|_2^2 = \sum_{m=1}^{M} \sum_{i:m(i)=m} (\mu_i - y_m)^2 = \mathcal{J}_S(\mu) + \sum_{m=1}^{M} d_m \big(y_m - z_m^*\big)^2, \tag{19}$$

where the first term equals $\mathcal{J}_S(\mu)$ by Proposition B.1. Setting $A_Z = I$ gives $\|(I - P_S)\mu\|_2^2 = \mathcal{J}_S(\mu)$. Moreover, $P_S^2 = S D_S^{-1} S^\top S D_S^{-1} S^\top = P_S$, so P_S is an orthogonal projection and $\|P_S\|_F^2 = \operatorname{tr}(P_S) = \operatorname{tr}(D_S^{-1} S^\top S) = M$. Applying Eq. (17) to P_A and to P_S and subtracting gives the identity in Eq. (11), and $\|P_A\|_F^2 \ge 0$ gives the inequality. For $A_Z = M^{-1}\mathbf{1}\mathbf{1}^\top$, every entry of y equals $\bar{z}$. □

For a soft assignment, Jensen's inequality applied to $(P_A\mu)_i = \sum_m S_{im} y_m$, followed by the completed square in Eq. (14) with $z = y$, gives $\|(I - P_A)\mu\|_2^2 \le \mathcal{J}_S(\mu) + \sum_m d_m (y_m - z_m^*)^2$. The mixing term thus enters the bias bound in the same form for soft assignments.

B.5 State-Independent Kernels

Graph layers aggregate neighbors with weights P_{ij} that are fixed by the road graph or learned once for all inputs, which is limitation ① in Section 1. The same risk decomposition applies to such kernels.

Proposition B.3 (Bias of a row-stochastic kernel). *Let* $P \in \mathbb{R}^{L \times L}$ *have nonnegative entries and unit row sums. Then*

$$\mathcal{R}(P) = \|(I - P)\mu\|_2^2 + \sigma^2 \|P\|_F^2, \qquad \|(I - P)\mu\|_2^2 \le \sum_{i=1}^{L} \sum_{j=1}^{L} P_{ij} (\mu_i - \mu_j)^2. \tag{20}$$

Proof. The first identity is Eq. (17) with $T = P$ and $y = \mu$. Because each row sums to one, $((I - P)\mu)_i = \sum_j P_{ij}(\mu_i - \mu_j)$, and Jensen's inequality gives $((I - P)\mu)_i^2 \le \sum_j P_{ij}(\mu_i - \mu_j)^2$. Summing over i proves the bound. □

A state-independent kernel assigns the same weight P_{ij} to a neighbor pair in free flow and in congestion. The signal μ encodes how the deviation at each sensor evolves, so it differs across a congestion front even where the current flows of the two sides coincide. Adjacent sensors on opposite branches therefore have a large $(\mu_i - \mu_j)^2$ and keep a nonzero weight, and every such pair contributes to the bound. For a hard assignment, $(P_S)_{ij} = 1/d_m$ when i and j belong to the same state m and zero otherwise, so pairs placed in different states do not contribute; applying Eq. (20) to P_S gives $2\mathcal{J}_S(\mu)$, which Proposition B.1 tightens to $\mathcal{J}_S(\mu)$.

B.6 Structure and Cost of the Sharing Kernel

For a fully observed time slice with fixed sensor assignment $S \in \mathbb{R}^{N\times M}$, the pooling-and-redistribution kernel $P_S = SD_S^{-1}S^\top$ factors through the M latent states and satisfies $\operatorname{rank}(P_S) \le M$. To include a linear map within the latent space, let $S \in \mathbb{R}^{L\times M}$ be fixed, let $A_Z \in \mathbb{R}^{M\times M}$ be a fixed latent map, and let $V \in \mathbb{R}^{L\times d}$ contain the value features. Pooling gives $Z = D_S^{-1}S^\top V$, and redistribution after the latent map gives $O = SA_ZZW_V$.

Proposition B.4 (Conditional interaction factorization)**.** *The spatial map factors through the M latent states as*

$$O = P_A V W_V, \qquad P_A = SA_Z D_S^{-1} S^\top, \qquad \operatorname{rank}(P_A) \le M. \tag{21}$$

Proof. Substituting Z into $O = SA_ZZW_V$ proves the factorization by associativity. The image of P_A lies in the column space of S, so $\operatorname{rank}(P_A) \le \operatorname{rank}(S) \le M$. The row identity $O_i = \sum_m S_{im}(A_ZZW_V)_m$ shows how each position receives its own mixture of the transformed states. □

Taking $L = N$ and $A_Z = I_M$ gives the per-step kernel P_S. Per head, pooling and redistribution across the joint window cost $\mathcal{O}((T_1 + T_2)NMd)$, and the recurrent evolution with direct future projection costs $\mathcal{O}((T_1 + T_2)Md^2)$. For token assignments, $L = (T_1 + T_2)N$ and A_Z is the latent attention matrix conditioned on the input; fixing that input gives the same factorization, with pooling and redistribution cost $\mathcal{O}(LMd)$ and an additional $\mathcal{O}(M^2d)$ for dense latent attention.

B.7 Formalization

The accompanying Lean project provides machine-checked statements for the aggregation and risk bounds, the kernel structure, the traffic dynamics, and the pole constraints. It uses Lean 4.19.0 and Mathlib v4.19.0. The soft-aggregation results assume nonnegative assignments, unit row sums, and positive column masses; the risk results additionally assume zero-mean noise with finite second moments and isotropic covariance; the traffic results use classical derivatives and the field assumptions of Appendix C. Table 4 maps each formalized result to its module.

Table 4: Correspondence between mathematical results and Lean source modules, relative to the project's `StaLiNO/` directory.

Result	Lean module
Representative optimality, bias bound, noise retention (Propositions B.1–B.2)	`SoftAggregation.lean`
Risk decomposition and sharing criterion (Theorem 2.1)	`Risk.lean`
Conditional interaction factorization and rank (Proposition B.4)	`Structure.lean`
LWR–Burgers transformation and reference dynamics (Propositions C.1–C.2)	`TrafficAlgebra.lean`, `TrafficPDE.lean`
Characteristic transport and local translation (Propositions C.3–C.4)	`Transport.lean`
Pole location and exponential decay (Proposition D.1)	`StablePoles.lean`

C TRAFFIC-FIELD BACKGROUND

This section derives the deviation dynamics behind Eq. (1) and the characteristic-translation argument used in Section 2.1. Throughout, ξ denotes road position, and a classical field is continuously differentiable in (ξ, t) with a continuous second spatial derivative.

C.1 FROM DENSITY CONSERVATION TO STATE-DEPENDENT PROPAGATION

Consider the LWR model with quadratic flux $q(r) = v_f r(1-r/\rho_{\max})$, where $v_f > 0$ and $\rho_{\max} > 0$. Its derivative $q'(r) = v_f - 2v_f r/\rho_{\max}$ is the characteristic speed, which is positive below the critical density $\rho_{\max}/2$ and negative above it.

Proposition C.1 (LWR–Burgers transformation)**.** *Let $\rho : \mathbb{R}^2 \to \mathbb{R}$ be a classical density field and $\nu \geq 0$ be constant. Set $a = -2v_f/\rho_{\max}$ and $w = a\rho + v_f = q'(\rho)$. Then*

$$\rho_t + \partial_\xi q(\rho) = \nu \rho_{\xi\xi} \quad \Longleftrightarrow \quad w_t + w w_\xi = \nu w_{\xi\xi}. \tag{22}$$

Proof. Expanding the quadratic flux gives $w^2/2 = aq(\rho) + v_f^2/2$. The affine transformation also gives $w_t = a\rho_t$ and $w_{\xi\xi} = a\rho_{\xi\xi}$. Differentiating the flux identity therefore yields

$$w_t + w w_\xi - \nu w_{\xi\xi} = a\big[\rho_t + \partial_\xi q(\rho) - \nu \rho_{\xi\xi}\big]. \tag{23}$$

Since $a \neq 0$, either residual vanishes exactly when the other does. □

The case $\nu = 0$ recovers the inviscid LWR conservation law. In the transformed equation, the field value w itself is the local advection speed.

C.2 DYNAMICS RELATIVE TO A BASE FIELD

Write the transformed field as $w = \bar{w} + \eta_w$, where $\bar{w}$ is a base state and η_w its deviation.

Proposition C.2 (Base-field decomposition)**.** *Suppose $\bar{w}$ and $\bar{w} + \eta_w$ are classical solutions of the Burgers equation with the same constant $\nu \geq 0$. Then*

$$\partial_t \eta_w + \bar{w}\, \partial_\xi \eta_w + \eta_w\, \partial_\xi \bar{w} + \eta_w\, \partial_\xi \eta_w = \nu\, \partial_\xi^2 \eta_w. \tag{24}$$

Proof. Substitute $w = \bar{w} + \eta_w$ into $w_t + w w_\xi = \nu w_{\xi\xi}$ and subtract $\bar{w}_t + \bar{w}\bar{w}_\xi = \nu \bar{w}_{\xi\xi}$. The product rule gives Eq. (24); its two base-dependent terms combine as $\partial_\xi(\bar{w}\eta_w)$. □

The term $\bar{w}\, \partial_\xi \eta_w$ advects the deviation at the speed of the base state, and $\eta_w\, \partial_\xi \bar{w}$ couples it to spatial variation in that state. With $\nu = 0$, substituting $\eta_w = a\eta$, $\bar{w} = q'(\bar{\rho})$, and $q'' = a$ and dividing by a recovers the density form in Eq. (1). The deviation dynamics therefore depend on both the deviation and the base state in which it occurs.

C.3 CHARACTERISTICS AND LOCAL TRANSLATION

For a constant speed c and forecast interval $h \geq 0$, define the transport residual of a jointly continuously differentiable field u as $\mathcal{D}_c[u] = u_t + c u_\xi$. A characteristic joins $(\xi - ch, t)$ to $(\xi, t+h)$ along $\gamma(s) = (\xi - c(t+h) + cs, s)$ for $s \in [t, t+h]$.

Proposition C.3 (Characteristic translation bound)**.** *If $|\mathcal{D}_c[u](\gamma(s))| \leq \varepsilon$ for all $s \in [t, t+h]$, then*

$$|u(\xi, t+h) - u(\xi - ch, t)| \leq \varepsilon h. \tag{25}$$

In particular, $\mathcal{D}_c[u] = 0$ implies $u(\xi, t+h) = u(\xi - ch, t)$.

Proof. The chain rule gives $\frac{d}{ds} u(\xi - c(t+h) + cs, s) = \mathcal{D}_c[u](\gamma(s))$. The derivative is bounded in absolute value by ε on the segment, so the mean value theorem bounds the difference between its endpoint values by εh. □

For the deviation in Proposition C.2, replacing the varying base speed by c produces four explicit contributions:

$$\mathcal{D}_c[\eta_w] = (c - \bar{w})\, \partial_\xi \eta_w - \eta_w\, \partial_\xi \bar{w} - \eta_w\, \partial_\xi \eta_w + \nu\, \partial_\xi^2 \eta_w. \tag{26}$$

Proposition C.4 (Local translation of a Burgers deviation)**.** *Under the assumptions of Proposition C.2, let $h \geq 0$ and fix $c \in \mathbb{R}$. Suppose the following bounds hold on $\gamma([t, t+h])$ for nonnegative constants δ, E, G_1, B_1, G_2:*

$$|c - \bar{w}| \leq \delta, \quad |\eta_w| \leq E, \quad |\partial_\xi \eta_w| \leq G_1, \quad |\partial_\xi \bar{w}| \leq B_1, \quad |\partial_\xi^2 \eta_w| \leq G_2. \tag{27}$$

Then

$$|\eta_w(\xi, t+h) - \eta_w(\xi - ch, t)| \leq h\big(\delta G_1 + E B_1 + E G_1 + \nu G_2\big). \tag{28}$$

Proof. Eq. (26) follows by rearranging Eq. (24). The triangle inequality and the assumed bounds give $|\mathcal{D}_c[\eta_w]| \leq \delta G_1 + E B_1 + E G_1 + \nu G_2$ along the characteristic, and Proposition C.3 gives the result. □

With c close to the base speed $\bar{w} = q'(\bar{\rho})$, the deviation at ξ after h steps is close to its current value at $\xi - ch$. Because $q'(\bar{\rho})$ changes sign at the critical density, the location that informs a forecast lies upstream in free flow and downstream in congestion, so the relevant spatial context depends on the regime.

D Temporal Operator

Pole parameterization. For real parameters a_k, β_k and a positive margin m_p, the poles of the temporal operator in Eq. (5) are

$$p_k = -\operatorname{softplus}_{20}(a_k) - m_p + \mathrm{i}\beta_k, \qquad m_p > 0, \tag{29}$$

where $\operatorname{softplus}_{20}(a) = \log(1 + e^a)$ for $a \leq 20$ and a for $a > 20$, matching the implementation; we use $m_p = 10^{-3}$. Each pole p_k and its residue γ_k are paired with $\overline{p_k}$ and $\overline{\gamma_k}$, which gives the transfer function $\mathcal{H}_c(\mathrm{i}\omega) = \sum_{k=1}^{2K} \gamma_{ck}/(\mathrm{i}\omega - p_{ck})$.

Proposition D.1 (Pole location and exponential decay)**.** *For the parameterization in Eq.* (29) *with any $m_p > 0$, both p_k and $\overline{p_k}$ have real part smaller than $-m_p$. Consequently, $\mathrm{i}\omega - p_k \neq 0$ for every real frequency ω, and, for every time lag $\vartheta \geq 0$,*

$$\left|e^{\vartheta p_k}\right| \leq e^{-m_p \vartheta} \leq 1, \qquad \gamma_k e^{\vartheta p_k} + \overline{\gamma_k} e^{\vartheta \overline{p_k}} = 2\,\mathrm{Re}\big(\gamma_k e^{\vartheta p_k}\big) \in \mathbb{R}. \tag{30}$$

Proof. The quantity $\log(1+e^a)$ is positive for every real a, and the linear branch is positive because it is selected when $a > 20$. Thus $\mathrm{Re}(p_k) < -m_p$, and conjugation preserves the real part. Since the real part of $\mathrm{i}\omega - p_k$ is $-\mathrm{Re}(p_k) > 0$, this denominator cannot vanish. For $\vartheta \geq 0$, $|e^{\vartheta p_k}| = e^{\vartheta\,\mathrm{Re}(p_k)}$ gives the decay bound. Conjugation commutes with the complex exponential, so the second term of the paired response is the conjugate of the first, and their sum is twice its real part. □

Transient construction. Frequency-domain filtering alone treats the joint sequence as periodic, so its response at the first position is driven by the end of the sequence. For a temporal input U, let $T = T_1 + T_2$, $\vartheta_j = j/T$ for $j = 0, \ldots, T-1$, and $\alpha_{nic} = [\mathcal{F}_t(U)]_{nic}$, where n indexes the T angular DFT frequencies ω_n on this normalized grid. For one layer with all $2K$ conjugate-paired modes, the transient is

$$d_{ick} = -\frac{1}{T} \sum_{n=0}^{T-1} \alpha_{nic} \frac{\gamma_{ck}}{\mathrm{i}\omega_n - p_{ck}}, \qquad [T_{\mathrm{tr}}(U)]_{jic} = \mathrm{Re} \sum_{k=1}^{2K} d_{ick} e^{p_{ck}\vartheta_j}. \tag{31}$$

At $j = 0$, every exponential equals one, so the transient equals the negative of the filtered response at that position and their sum is zero. At later positions, the same coefficients multiply decaying modes, which gives the response of the stable system started from rest.

E Architecture and Implementation Details

E.1 Base-Field Refinement

The road aggregation uses the rectified linear unit (ReLU) activation. Each periodic table P_p, $p \in \{d, w\}$, is refined by a road aggregation followed by temporal attention over calendar slots and spatial attention over sensors:

$$F_p = \mathrm{ReLU}\left(P_p A^\top W_p + b_p\right), \qquad \widetilde{P}_p = \mathrm{Linear}\left([\mathrm{Attn}_t(F_p) \,\|\, \mathrm{Attn}_s(F_p^\top)^\top]\right), \tag{32}$$

where A includes self-loops and is row-normalized. The weekly slot is $i_w(s) = i_d(s) + L_d w(s)$ for day of week $w(s)$. When the starting weekday is unavailable, weekly indices follow a consistent seven-day cycle from the series origin. Periodic means are initialized from valid training observations, and unobserved calendar slots are initialized to zero.

E.2 Joint Field and Future Queries

The state GRU encodes the same sequence $\mathbf{a}_i$ as the routing GRU. A node embedding e_i adds location identity. A road update incorporates connected sensors. LN follows the road update:

$$g_i = \mathrm{GRU}(\mathbf{a}_i) + e_i, \qquad \bar{G} = \mathrm{LN}\big(G + \mathrm{SiLU}\big(GW_{\mathrm{self}}^{\mathrm{hist}} + AGW_{\mathrm{nbr}}^{\mathrm{hist}} + b_{\mathrm{hist}}\big)\big), \tag{33}$$

where $G \in \mathbb{R}^{N \times C}$ stacks the summaries g_i. For prediction step h, the query adds a step embedding e_h^{step} and the calendar features κ_{t+h} of the target timestamp, and the queries are appended to the lifted deviations:

$$q_{h,i} = \bar{g}_i + e_h^{\mathrm{step}} + \kappa_{t+h} W_{\mathrm{cal}}, \qquad U^0 = \mathrm{Concat}_{\mathrm{time}}(\mathcal{E}(R), Q) \in \mathbb{R}^{(T_1+T_2)\times N \times C}, \tag{34}$$

where $\mathcal{E}$ lifts each deviation and its validity indicator into the hidden space.

E.3 Assignments, Validity, and Readout

The routing GRU summarizes deviations, validity indicators, and calendar features once per input window. Each layer projects these summaries into head-specific channels; the membership projection g_θ is shared across heads within a layer, and the temperatures are learned per head. Missing deviations are set to zero and accompanied by a validity indicator. Pooling uses the sensors observed at each step and sets the pooled feature of a state without observed sensors to zero; the latent GRU receives a pooled-source validity indicator alongside the pooled features, and a state with no observed sensor throughout the window has its shared update set to zero. For the readout, let $\mathcal{I}_i \subseteq \{t - T_1 + 1, \ldots, t\}$ contain the valid historical timestamps of sensor i. If $\mathcal{I}_i$ is nonempty, $r_i^{\mathrm{last}} = x_{\ell_i,i} - b_{\ell_i,i}$ with $\ell_i = \max \mathcal{I}_i$; otherwise $r_i^{\mathrm{last}} = 0$. The correction is $\Delta_{h,i} = w_{\mathrm{out}}^\top U_{T_1+h,i}^{\mathrm{out}} + b_{\mathrm{out}}$, where U^{out} is the output of the last field layer.

E.4 Training and Hardware

StaLiNO uses $M = 16$ latent states per head, hidden width 64, four heads, three field layers, and eight temporal modes before conjugate pairing. AdamW uses batch size 32, learning rate 10^{-3}, and weight decay 10^{-4}. Inference uses a batch size of 16. Training uses a 20-epoch budget and keeps the checkpoint with the lowest validation MAE. All variants are retrained with the same protocol. We use Python 3.12.3, PyTorch 2.5.1 with CUDA 12.4, and an NVIDIA RTX 4090 with 24 GB memory.

F Experimental Protocols and Supplementary Results

F.1 Data, Metrics, and Baselines

PEMS03, PEMS04, PEMS07, and PEMS08 contain 358, 307, 883, and 170 sensors sampled every five minutes, with chronological training, validation, and test splits at a 6:2:2 ratio. The input and prediction lengths are both 12 steps. Normalization statistics and base-field initialization use the

training split. MAE, RMSE, and MAPE aggregate valid errors over the 12 prediction steps; zero-valued targets are excluded according to the dataset validity rule. Predictions are transformed back to the original scale and clipped to nonnegative values. The baselines are listed with their references in Section 3.1.

F.2 Latent-Operator Variants on PEMS03

Table 5 extends Table 2 with further ways of allocating latent states. Persistence repeats each state's last valid pooled value into the future, latent mixing adds a pre-normalized residual attention layer with 3,216 parameters before the latent GRU, and no latent sharing removes the shared update and its routing encoder while keeping the local road and temporal updates. Sparse assignments, layer-wise and window-wise budgets, prototype assignments, and regularized states vary how many states are active and how they are allocated, and reliable sharing weights redistribution by observation completeness. None of them improves on trajectory assignments with independent evolution.

Table 5: All retrained latent-operator variants on PEMS03. MAPE is in percent.

Variant	MAE	RMSE	MAPE
No latent sharing	15.089	24.369	16.63
Token sharing	14.612	23.665	15.81
Token sharing + mixing	14.958	24.121	15.99
Token sharing + evolution	14.983	24.513	16.15
Trajectory sharing	14.834	23.861	15.93
Trajectory sharing + persistence	14.727	23.741	15.84
Trajectory sharing + mixing and evolution	14.772	23.963	15.72
Sparse assignments	14.844	23.882	16.02
Layer-wise budget	14.673	23.728	15.71
Window-wise budget	14.671	23.818	15.82
Prototype assignments	14.732	23.847	15.88
Regularized states	14.796	24.132	16.27
Reliable sharing	14.813	24.069	16.58
StaLiNO	**14.488**	**23.404**	16.12

F.3 Component Study on PEMS08

Table 6 reports the component study on PEMS08. Removing the base field disables the daily and weekly tables and their refinement, so the routing encoder and the latent operator receive raw flow, while calendar features and the local paths remain active. This variant has the largest error, and replacing the latent evolution by persistence or adding latent mixing also raises the MAE.

Table 6: PEMS08 component study. MAPE is in percent.

Variant	MAE	RMSE	MAPE
w/o base field	15.386	24.428	10.64
No latent sharing	13.917	23.939	9.31
Persistence instead of evolution	13.931	23.573	9.38
With latent mixing	13.975	23.750	9.56
StaLiNO	**13.814**	23.649	**9.17**

F.4 Test-Time Input Perturbations

For each test window and affected-sensor fraction $r \in \{0.1, 0.3, 0.5\}$, we sample $\lceil rN \rceil$ sensors and a contiguous three-step segment within the 12-step history. Surges multiply the selected observations by 1.5; interruptions set them to zero and mark them invalid; shuffling permutes the selected observations and their validity indicators within the segment. Calendar features and targets are unchanged. Each condition uses three realizations determined by the window start and realization index, and all models share the same perturbations. Figure 4 in Section 3.5 compares StaLiNO with

STWave, STPGNN, and D^{2}STGNN under surges of 50% of the sensors and under interruptions and shuffling of 30%.

F.5 Complete History Loss

We hide all historical observations of selected sensors and rerun inference with models trained without artificial sensor dropout. The missing rates are 0, 10, 30, and 50% on PEMS03. Each test window uses three mask realizations shared across models, with nested hidden-sensor sets as the rate increases, and errors are pooled over the sensors that keep their history. Table 7 lists the values behind Figure 3b, and StaLiNO keeps the lowest observed-sensor MAE at every rate.

Table 7: Observed-sensor MAE on PEMS03 when a fraction of sensors lose their complete history at test time. Bold and underline denote the best and second-best values at each rate.

Variant	0%	10%	30%	50%
Token sharing	<u>14.612</u>	<u>14.733</u>	<u>15.070</u>	15.438
No latent sharing	15.089	15.109	15.158	<u>15.219</u>
Reliable sharing	14.813	15.015	15.380	15.474
StaLiNO	**14.488**	**14.503**	**14.542**	**14.592**

F.6 Number of Latent States

Table 8 gives the full metrics behind Figure 3a. All budgets use the main PEMS08 protocol, under which the variant without latent sharing reaches MAE 13.917. Every budget with four or more states improves on this variant.

Table 8: Number of latent states per head on PEMS08. The gap is relative to $M = 16$; MAPE is in percent.

M	MAE	RMSE	MAPE	MAE gap
1	14.021	23.856	9.79	+1.50%
2	13.913	23.608	9.40	+0.72%
3	13.921	**23.560**	9.27	+0.77%
4	<u>13.863</u>	<u>23.565</u>	**9.15**	+0.35%
5	13.884	23.602	9.57	+0.51%
6	13.872	23.567	9.48	+0.42%
16 (default)	**13.814**	23.649	<u>9.17</u>	–

F.7 Computational Profiles

Inference uses a preloaded batch of 16 test windows, 10 warm-up passes, and 30 timed forward passes with CUDA synchronization; latency is the median, and peak memory includes the model and input batch. Epoch time includes training, validation, and checkpoint saving, and parameter counts include the complete model. Table 9 reports the complete model and its variants.

Table 9: Computational profiles on PEMS03 with an RTX 4090 and batch size 16.

Variant	Params (M)	Latency (ms)	Memory (GB)	Epoch (s)
No latent sharing	15.055	18.85	0.50	61.3
Persistence instead of evolution	15.107	22.95	0.50	76.6
With latent mixing	15.126	22.84	0.50	77.6
StaLiNO	15.122	23.04	0.50	77.7

F.8 Latent Mixing Diagnostics

We examine a latent-attention model on 64 evenly spaced test windows across its three layers and four heads. The normalized entropy of the attention rows is 0.988, where one corresponds to uniform weights over states, which places the model close to the uniform-mixing case of Proposition 2.2. The mean pairwise cosine similarity between different states' projected values increases from 0.44 before attention to 0.96 after mixing, so mixing makes the states more alike, as the dispersion cost in Proposition 2.2 predicts.

F.9 Regime-Identification Protocol

PEMS04 and PEMS08 record occupancy and speed alongside flow. StaLiNO receives flow only. The Highway Capacity Manual (HCM) places capacity at the upper end of level of service (LOS) E and describes LOS F as flow breakdown (Transportation Research Board, 2016). Caltrans also identifies a site-specific breakdown point from PeMS flow and occupancy data (California Department of Transportation, 2025). We use this capacity distinction to construct a binary reference label. For each sensor, we divide its training occupancy into 50 equal-width bins. We define the critical occupancy o_i^{c} as the center of the bin with the largest mean flow. We label a sensor and test window as congested when its occupancy at the last input step exceeds o_i^{c}. We label the remaining pairs as free-flow.

Each head computes its own assignments, and the membership columns of different heads are not aligned, because each head projects the trajectory summaries into its own channels before the membership projection. Averaging assignments across heads would therefore combine unrelated states. We instead evaluate every head h of the first field layer separately: the dominant state of sensor i in a window is $\arg\max_m S_{im}^{(h)}$, and we score the agreement between the dominant state and the binary label over all sensor–window pairs of the test split. The reported values average the scores over the four heads.

We use two scores. AMI corrects mutual information for chance agreement, which matters when 16 states are compared with two classes. Homogeneity, $1 - H(\mathrm{label} \mid \mathrm{state})/H(\mathrm{label})$, measures whether each state contains a single regime, independently of how many states a regime occupies, and reaches one when every state contains pairs of one regime only. The two controls each use 16 groups: k-means on the flow at the last input step and k-means on the 12-step flow window.

Table 10: Agreement between groupings and the occupancy-based congestion label. The scores of StaLiNO average the four heads of the first field layer, and all groupings use 16 groups.

	PEMS04		PEMS08	
Grouping	AMI	Homogeneity	AMI	Homogeneity
k-means on current flow	0.34	0.29	0.25	0.41
k-means on flow window	0.38	0.34	0.37	0.38
StaLiNO states	**0.48**	**0.63**	**0.46**	**0.61**

G Related Work

Spatiotemporal traffic forecasting. Graph-based forecasters combine spatial message passing on the road graph with recurrent, convolutional, or attention-based temporal models (Yu et al., 2018; Wu et al., 2019; Guo et al., 2019). Later work learns the graph from data (Wu et al., 2020; Bai et al., 2020; Lan et al., 2022), generates it dynamically from the current input (Li et al., 2023a; Jiang et al., 2023b), models propagation delay and long-range dependencies (Jiang et al., 2023a; Cirstea et al., 2022; Kong et al., 2024), introduces continuous-time dynamics (Fang et al., 2021), or organizes irregular sensors into spatial patches for scale (Fang et al., 2025). These kernels are either independent of the traffic state or regenerated from the input at each step, so they do not track a group of sensors over time. Liu et al. (2026) question whether global cross-variate dependencies are necessary for forecasting; our analysis gives a traffic-specific account of when sharing across sensors helps.

Decomposition and periodic structure. Decomposition-based forecasters separate traffic into components before modeling them: D^2STGNN decouples diffusion and inherent signals (Shao et al., 2022), STWave separates stable trends from fluctuating events (Fang et al., 2023), CycleNet learns recurrent cycles and forecasts the remaining residual (Lin et al., 2024), STDN decomposes trend and seasonality (Cao et al., 2025), and HyperD decouples daily and weekly periodicity from the residual (Shao et al., 2026). StaLiNO contributes the operator that models the deviation from its base field.

Physics-guided traffic modeling. The LWR model describes traffic as a conserved density with a state-dependent characteristic speed (Lighthill & Whitham, 1955; Richards, 1956; Treiber & Kesting, 2013). Physics-informed networks use it to estimate traffic states and queue profiles from sparse observations (Lu et al., 2023; Zhang et al., 2024), STDEN models traffic through a latent potential field and a differential equation (Ji et al., 2022), operator learning has been used to design controllers that suppress stop-and-go waves (Zhang et al., 2025), and PhySTA builds a graph neural operator for arbitrary spatiotemporal inference (Ge et al., 2026). StaLiNO uses the LWR model to shape the structure of the forecasting operator.

Neural operators and latent representations. Neural operators learn maps between function spaces through local operators and global kernel integrals (Kovachki et al., 2023). Operators for irregular domains encode physical samples into a latent or regional representation and decode back to the physical points (Li et al., 2023b; Mousavi et al., 2025), and latent operators evolve the field in a compact latent space (Serrano et al., 2023; Alkin et al., 2024; Wang & Wang, 2024). Transolver assigns mesh points to learned physical states at every layer and lets the state tokens interact through attention (Wu et al., 2024). The Laplace neural operator represents temporal responses through poles and residues (Cao et al., 2024), which StaLiNO adopts with a stability margin. StaLiNO differs from these latent operators in how its states are formed and used: the states are inferred from each sensor's deviation trajectory, held fixed across the window so that each follows the same sensors, and evolved without interaction, which Table 2 and Proposition 2.2 support for traffic.