FIST: FREQUENCY-CONDITIONED INTERACTION WITH SPARSE TOPOLOGY FOR MULTIVARIATE FORECASTING

Anonymous authors
Paper under double-blind review

ABSTRACT

Multivariate forecasting faces a fundamental trade-off: ignoring other variables may miss useful information, while indiscriminate interaction can introduce irrelevant dependencies. More importantly, a dependency useful for one temporal pattern may be unhelpful for another, yet cross-variable interaction is often shared across temporal components. We introduce FIST, a frequency-conditioned forecasting framework that follows a simple principle: preserve each variable's temporal dynamics before selectively introducing information from others. FIST first encodes variables independently, then partitions the learned representation into latent frequency bands and constructs a sparse interaction graph for each band. This design naturally supports independent, shared-graph, and band-conditioned variants under the same temporal backbone. A restricted linear analysis shows that sharing one interaction operator incurs an approximation cost when dependency patterns differ across bands, which controlled experiments verify. Across 165 forecasting runs on four datasets, however, band-specific interaction yields modest and setting-dependent gains after representation learning. These results distinguish the existence of component-specific dependencies from their actual predictive value, and clarify when frequency-conditioned interaction is useful for multivariate forecasting. Code is available at: https://anonymous.4open.science/r/FigNet-D2BC.

1 INTRODUCTION

Multivariate forecasting must decide how much information each variable should borrow from others. Channel-independent models such as DLinear and PatchTST show that strong forecasts can be obtained by modeling each variable largely on its own Zeng et al. (2023). In contrast, Crossformer, iTransformer, and graph-based models explicitly capture cross-variable dependencies Zhang & Yan (2023); Liu et al. (2024). Both choices have clear advantages: ignoring other variables may miss useful information, while excessive interaction can introduce irrelevant dependencies.

A more subtle challenge is that the usefulness of a dependency can change with the temporal pattern being modeled. Two variables may share a smooth long-term trend but exhibit unrelated short-term fluctuations; in another case, they may co-vary only at faster time scales. Thus, a variable that is informative for one temporal component may contribute little to another. Yet cross-variable interaction is often learned as a single structure shared across different temporal components. This raises a simple question: Should different temporal components rely on different cross-variable interactions?

Recent methods have made dependency modeling increasingly selective, including frequency-informed channel relations, patch-specific graph filtering, and channel–patch routing Qiu et al. (2025); Hu et al. (2025). However, it remains difficult to isolate whether any improvement comes from better temporal representations, from cross-variable interaction itself, or specifically from making the interaction component-dependent. A convincing answer therefore requires not only a flexible interaction mechanism, but also a controlled way to compare independent, shared, and component-specific interaction under the same temporal model.

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.

We introduce **FIST**, a frequency-conditioned forecasting framework built around a simple *preserve-then-interact* principle. FIST first learns the temporal representation of each variable independently, and only then introduces information from other variables. The learned representation is partitioned into latent frequency bands, and each band constructs its own sparse interaction graph. Learned affinities determine which variables exchange information, while explicit self-connections and gated residual aggregation preserve each variable's own representation. Importantly, the frequency bands are defined over learned token representations rather than directly over the physical frequencies of the raw signals.

This design makes the central question directly testable. Setting the interaction graph to identity gives an independent forecaster; sharing one sparse graph across bands gives a shared-interaction model; and learning separate graphs gives the band-conditioned variant. All three use the same temporal backbone and forecasting heads, allowing the effect of component-dependent interaction to be studied without changing the underlying temporal model. This controlled design is a key part of FIST rather than merely an ablation setting.

The motivation can also be made precise in a restricted linear setting. When different temporal bands follow different cross-variable mappings, forcing them to share one interaction operator introduces an approximation cost. When the mappings are the same, however, sharing is sufficient and can be easier to estimate from limited data. This leads to a clear prediction: band-specific interaction should help when dependency patterns genuinely differ across temporal components, but not necessarily otherwise. Our controlled regression experiments confirm these regimes.

The end-to-end forecasting results reveal a more nuanced picture. Across four datasets, multiple forecast horizons, paired random seeds, and two temporal backbones, band-conditioned interaction provides only modest and setting-dependent gains over a shared sparse graph. The independent control is also competitive in several settings. These results highlight an important distinction: the existence of component-specific dependencies does not guarantee that modeling them will improve forecasting after representation learning. A strong temporal encoder may already capture part of the useful structure before cross-variable interaction is applied.

Our study therefore focuses on a more practical question than simply whether richer interaction is more expressive: *when does component-specific dependency modeling provide predictive information beyond a strong temporal representation?*

Our contributions are threefold:

- We identify an underexplored challenge in multivariate forecasting: cross-variable dependencies can be useful for some temporal components but not others, making a single shared interaction structure potentially restrictive.
- We propose FIST, which preserves variable-specific temporal representations before introducing sparse, frequency-conditioned interaction, while naturally supporting independent, shared-graph, and band-conditioned variants under the same temporal backbone.
- We combine a restricted linear analysis, controlled dependency experiments, and matched forecasting comparisons to characterize when component-specific interaction is useful and when its advantage diminishes after representation learning.

2 Related Work

Temporal representations for forecasting. Forecasting models represent temporal dynamics in different ways. DLinear and PatchTST demonstrate the effectiveness of channel-independent linear and patch-based modeling (Zeng et al., 2023; Nie et al., 2023), while Autoformer, TimesNet, and TimeMixer exploit decomposition, periodicity, and multi-scale structure (Wu et al., 2021; 2023; Wang et al., 2024). These results motivate learning strong channel-local representations before cross-variable interaction. FIST follows this principle and evaluates both GRU and MLP temporal backbones.

Cross-variable dependency modeling. Crossformer and iTransformer explicitly model dependencies across variables (Zhang & Yan, 2023; Liu et al., 2024), while TSMixer and SOFTS perform feature mixing or shared-core aggregation (Chen et al., 2023; Han et al., 2024). Graph-based

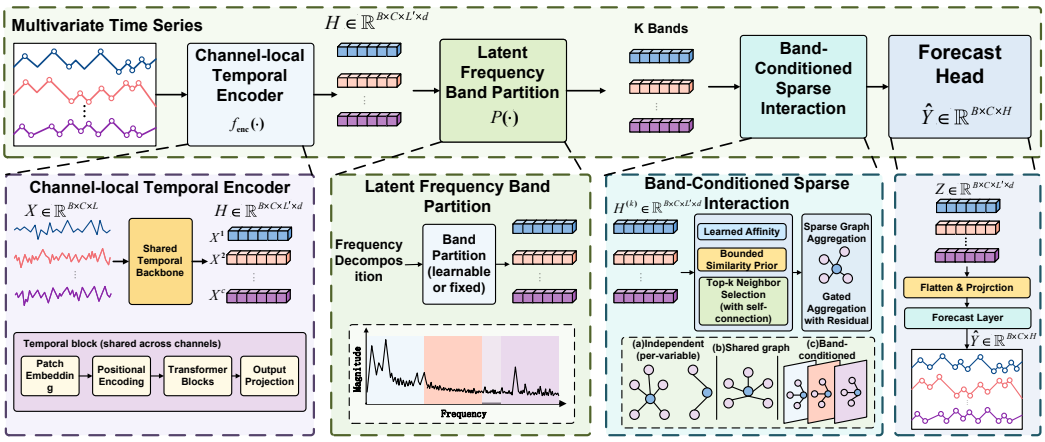

Figure 1: FIST with an explicit channel-local backbone and frequency-conditioned interaction. The temporal encoder operates within the observed history. Each graph layer recomputes band affinities, mixes spectra with real channel weights, and adds a gated residual. The trend head and normalization inversion complete the forecast.

methods such as DCRNN, Graph WaveNet, AGCRN, and MTGNN instead represent cross-variable relations through graph propagation or learned topology (Li et al., 2018; Wu et al., 2019; Bai et al., 2020; Wu et al., 2020). More recent methods make dependency selection increasingly adaptive: DUET uses frequency-informed channel relations, TimeFilter filters patch-specific connections, and xCPD routes channel–patch dependencies (Qiu et al., 2025; Hu et al., 2025; Li et al., 2026). FIST differs by learning a separate sparse channel graph for each latent temporal band and comparing it directly with independent and shared-graph controls.

Frequency-domain forecasting. FEDformer and FreTS incorporate frequency representations into forecasting (Zhou et al., 2022; Yi et al., 2023b), FourierGNN performs graph operations in Fourier space (Yi et al., 2023a), and FreDF introduces frequency-domain supervision (Wang et al., 2025). In contrast, FIST uses latent temporal frequency specifically to condition cross-variable interaction. Its temporal Fourier partition is distinct from a graph Fourier basis, and the interaction mechanism is evaluated separately from the similarity prior and auxiliary spectral objective.

3 Frequency-Conditioned Sparse Interaction

Let $X \in \mathbb{R}^{L\times C}$ contain an observed history and $Y \in \mathbb{R}^{H\times C}$ its future targets. All operations below are per sample; the batch dimension is omitted. H denotes the forecast horizon, N the number of tokens, D the feature width, and B the number of spectral bands. FIST separates temporal representation from channel communication: it encodes histories locally, selects neighbors within each latent band, and combines the resulting messages for prediction (Figure 1). This separation makes graph sharing a controlled architectural choice.

3.1 Channel-Local Temporal Representation

The backbone supplies a useful forecasting representation before cross-channel mixing. Keeping its operations local also makes an identity graph a valid independent-channel control.

Normalization and decomposition. For each channel c, the history supplies a mean μ_c and standard deviation $s_c = \sqrt{\mathrm{Var}(X_{:,c}) + \epsilon}$. Following reversible instance normalization (Kim et al., 2022), we form $\widetilde{X}_{t,c} = a_c(X_{t,c} - \mu_c)/s_c + b_c$, retaining the statistics for output inversion. The statistics are detached from differentiation. A replicate-padded moving average with width κ gives the trend $T = \mathrm{MA}_\kappa(\widetilde{X})$ and residual $S = \widetilde{X} - T$. The learned affine parameters, temporal processing, and trend forecast do not pool observations across channels.

Patching and encoding. Patches have length P and stride $s \le P$. Right replication padding gives

$$N = \left\lceil \frac{L-P}{s} \right\rceil + 1, \qquad p_{\text{right}} = (N-1)s + P - L. \tag{1}$$

A shared linear patch embedding and a learned position embedding produce $Z \in \mathbb{R}^{C\times N\times D}$. Each channel then passes through residual temporal blocks

$$U = Z + \text{Drop}(\mathcal{T}(\text{LN}_1(Z))), \qquad V = U + \text{Drop}(\text{FFN}(\text{LN}_2(U))). \tag{2}$$

Here $\mathcal{T}$ acts on one channel's token sequence. The CPU experiments specify a bidirectional GRU with a linear merge; a token-wise MLP provides a second backbone. The implementation also exposes a bidirectional Mamba option (Gu & Dao, 2024), which requires its dependency explicitly and is not evaluated in the CPU study. Bidirectional processing is confined to the available history and does not access future targets.

3.2 Graphs indexed by latent temporal frequency

For the input V to each graph layer, define

$$Z_f = \text{rFFT}_N(V; \text{ortho}) \in \mathbb{C}^{C\times F\times D}, \qquad F = \lfloor N/2 \rfloor + 1. \tag{3}$$

Partition the F bins into $B \le F$ contiguous, nonempty bands $\mathcal{F}_b$. Band sizes differ by at most one; the first $F \bmod B$ bands receive the extra bins. The default $L = 96$, $P = 16$, and $s = 8$ give $N = 11$, $F = 6$, and band sizes $(2, 2, 1, 1)$ for $B = 4$. Invalid requests for more bands than bins raise an error.

These frequencies belong to the learned token sequence. Patching, subsampling, position embeddings, and nonlinear temporal encoding prevent an automatic identification with frequencies in the original measurement units. All frequency interpretations in the method are therefore latent unless independently calibrated.

Spectral similarity. To bias neighbor selection toward compatible band representations, flatten a channel's features into $v_{b,i} \in \mathbb{C}^{|\mathcal{F}_b|D}$ and define

$$r_{b,ij} = \frac{|v_{b,i}^\top \overline{v_{b,j}}|}{\sqrt{\max(\|v_{b,i}\|_2^2, \epsilon)\max(\|v_{b,j}\|_2^2, \epsilon)}}, \qquad K_{b,ij} = \log(1 + \text{clip}(r_{b,ij}, 0, 1)). \tag{4}$$

The range of K is $[0, \log 2]$. It is a coherence-inspired similarity of learned complex vectors, not a statistical estimate of raw-signal magnitude-squared coherence. The formula adds no parameters, but its input representation is learned. Zero-energy vectors are handled by the denominator guard. Away from the numerical guard, one nonzero complex scalar per channel gives normalized magnitude one; with multiple feature dimensions it need not be. Phase differences that vary across a band can cancel in the inner product, so this prior need not preserve every lagged relation.

Learned affinities. Similarity alone need not identify useful predictive neighbors. We therefore combine it with learned, directed affinities. Let $\bar{z}_{b,i}$ be the mean complex feature over the band's bins and $e_b \in \mathbb{R}^{2D}$ a learned band embedding. With shared affine query and key maps,

$$u_{b,i} = [\Re \bar{z}_{b,i}; \Im \bar{z}_{b,i}] + e_b, \quad q_{b,i} = W_q u_{b,i} + b_q, \quad k_{b,i} = W_k u_{b,i} + b_k, \tag{5}$$

$$S_{b,ij} = q_{b,i}^\top k_{b,j}/\sqrt{D} + \alpha K_{b,ij}. \tag{6}$$

The scalar α is learned and initialized to one. The similarity acts as a bias at every forward pass, rather than an initialization of a static graph. Removing it sets its contribution to zero.

Exact support and the independent limit. Sparsity controls how many other channels can contribute while preserving a self-path. The integer k counts *off-diagonal* neighbors. For receiving channel i, let

$$\mathcal{N}_b(i) = \{i\} \cup \text{Top}_k\{S_{b,ij} : j \ne i\}, \qquad A_{b,ij} = \frac{\exp S_{b,ij}\, \mathbf{1}[j \in \mathcal{N}_b(i)]}{\sum_{j' \in \mathcal{N}_b(i)} \exp S_{b,ij'}}. \tag{7}$$

Stable descending sorting resolves ties by channel index, retaining exactly $k+1$ support entries. An entry $A_{b,ij}$ sends information from j to i. The limit $k = 0$ is $A_b = I$; $k = C - 1$ permits dense interaction. Selecting one unconstrained top-ranked channel would not guarantee independence. Gradients flow through retained scores and messages; changes of the selected support are discrete.

3.3 SPECTRAL MIXING, RESIDUALS, AND FORECASTING

After selecting neighbors, each graph mixes only the spectral features in its assigned band. The same real-valued adjacency acts on both quadratures:

$$M_b = A_b Z_f^b, \qquad M = \mathrm{irFFT}_N([M_1; \ldots; M_B]; \mathrm{ortho}), \qquad V' = \mathrm{LN}\left(V + \sigma(\gamma)\,\mathrm{Drop}(M)\right). \tag{8}$$

There are no additional complex feature weights or complex activations in this operator. Each graph layer has its own parameters and scalar gate γ, initialized to -2. A finite sigmoid gate is not an exact switch to independence; the identity-graph control provides that limit. Real graph weights preserve the real-valued DC and, for even N, Nyquist coefficients. Subsequent graph layers repeat the spectrum, graph construction, and residual update using their current input.

A final feature normalization, flattened seasonal head, and linear trend head give, for each channel,

$$\widehat{Y}_{:,c} = \mathrm{RevIN}_c^{-1}(W_s\,\mathrm{vec}(\mathrm{LN}(V'_c)) + b_s + W_t T_{:,c} + b_t)\,. \tag{9}$$

The forecasting heads share parameters across channels. Thus setting all graphs to identity leaves each output dependent only on its corresponding channel's history.

Training objective. With $F_H = \lfloor H/2 \rfloor + 1$, the implemented auxiliary objective is a complex-modulus loss:

$$\mathcal{L} = \frac{\|Y - \widehat{Y}\|_F^2}{HC} + \frac{\lambda}{F_H C} \sum_{f,c} \left| \mathrm{rFFT}_H(Y - \widehat{Y}; \mathrm{ortho})_{f,c} \right|. \tag{10}$$

This term is nonquadratic and differs from MSE; it is not magnitude-only matching. We do not claim equal energy across bands or guaranteed high-frequency compensation. A correctly weighted, squared Fourier loss would reproduce time-domain MSE by Parseval's identity. Section A states the single-sided weighting precisely. We use $\lambda = 0.1$ and test $\lambda = 0$ separately.

4 WHEN BAND CONDITIONING CAN MATTER

The design in Section 3 is motivated by a sharing constraint: one graph must serve several bands even when their predictive relations differ. A restricted linear model makes the resulting approximation cost explicit. This calculation concerns the interaction operator after fixing a representation; it does not compare the expressive power of complete neural architectures.

Proposition 1 (Shared-operator approximation cost)**.** *For each band, let $y_b = T_b x_b + \varepsilon_b$, where $\mathbb{E}[x_b x_b^*] = I$, $\mathbb{E}[\varepsilon_b x_b^*] = 0$, and second moments are finite. Let $p_b \geq 0$ with $\sum_b p_b = 1$. Among linear predictors using the same matrix T in every band, the excess squared risk over the band-specific oracle is*

$$\min_T \sum_b p_b \mathbb{E}\|y_b - T x_b\|_2^2 - \sum_b p_b \mathbb{E}\|\varepsilon_b\|_2^2 = \sum_b p_b \|T_b - \bar{T}\|_F^2, \quad \bar{T} = \sum_b p_b T_b. \tag{11}$$

The proof is a weighted variance decomposition (Section A). The right-hand side measures operator heterogeneity: it vanishes when all positive-weight bands share the same mapping and is positive when their mappings differ. This separates two empirical questions. First, can band-specific estimators avoid the sharing penalty when the representation and relations are known? Second, does this advantage remain after learning a representation from finite forecasting data? The proposition answers neither the estimation question nor the representation-learning question. Sharing can reduce estimation variance, while an overly restrictive sparse support can omit useful channels. We therefore test heterogeneous, homogeneous, and independent controls before evaluating learned forecasters.

Computational cost. For one graph layer, the spectral transform costs $O(CDN \log N)$, the spectral inner products $O(C^2FD)$, the query/key projections $O(BCD^2)$, and their pairwise affinities $O(BC^2D)$. The exact stable sort in the reference implementation adds $O(BC^2 \log C)$ in the worst case. Gather-based propagation costs $O(CFD(k+1))$ after support selection; a dense masked implementation costs $O(C^2FD)$. The C^2 affinity construction remains in both cases. Thus sparse message propagation does not make the entire layer linear in channel count. We measure the complete graph layer and compare equivalent dense-masked and gather-based implementations in Section 5.6.

Table 1: Forecasting performance at $H = 96$: mean $\pm$ standard deviation over five seeds. The first seven rows share a GRU backbone and registered parameter budget. External baselines use the same data windows and their specified configurations.

	ETTh1		Exchange	
Variant	MSE	MAE	MSE	MAE
Identity	0.3980 ± 0.0033	0.4077 ± 0.0016	0.0875 ± 0.0024	0.2060 ± 0.0027
Shared dense	0.3991 ± 0.0031	0.4083 ± 0.0016	0.0876 ± 0.0025	0.2060 ± 0.0028
Shared sparse	0.3987 ± 0.0033	0.4081 ± 0.0017	0.0874 ± 0.0025	0.2058 ± 0.0028
Band dense	0.3991 ± 0.0032	0.4083 ± 0.0017	0.0876 ± 0.0026	0.2060 ± 0.0028
Band sparse	0.3989 ± 0.0033	0.4082 ± 0.0018	0.0876 ± 0.0025	0.2061 ± 0.0028
Without prior	0.3990 ± 0.0033	0.4083 ± 0.0018	0.0876 ± 0.0025	0.2061 ± 0.0028
MSE only	0.3995 ± 0.0034	0.4089 ± 0.0018	0.0878 ± 0.0025	0.2065 ± 0.0028
DLinear	0.3938 ± 0.0024	0.4090 ± 0.0036	0.0785 ± 0.0011	0.1993 ± 0.0025
iTransformer	0.4022 ± 0.0042	0.4172 ± 0.0034	0.0920 ± 0.0067	0.2132 ± 0.0076

5 Experiments

We evaluate forecasting performance, component contributions, prediction stability, and computational cost. Comparisons with external baselines and matched controls establish the empirical results; ablations, prediction diagnostics, and a controlled dependency study examine the role of selective interaction.

5.1 Experimental Settings

Datasets and protocol. We use ETTh1/ETTm2 (7 variables), Exchange (8), and Weather (21) (Zhou et al., 2021; Lai et al., 2018; Nie et al., 2023), predicting all variables from $L = 96$ observations. Horizon $H = 96$ is evaluated on all datasets; separate $H = 336$ models are trained on ETTh1 and Exchange. Chronological splits are 12/4/4 months for ETT and 70/10/20 for the others. Standardization uses training data only; targets never cross split boundaries.

Baselines and controls. External baselines are DLinear (Zeng et al., 2023) and the released iTransformer architecture (Liu et al., 2024), evaluated on ETTh1/Exchange at $H = 96$. Internal controls vary graph sharing, sparsity, the prior, or the objective under a common GRU backbone and forecast heads; identity disables cross-variable communication. Other settings compare identity, shared-sparse, and band-sparse graphs, including an MLP backbone on ETTh1 at $H = 96$. External baselines have different parameter budgets.

Implementation and metrics. Runs use PyTorch 2.5.1 on single-thread CPU. FIST uses width 16, one temporal block, one graph layer, four bands, and two off-diagonal neighbors. Adam uses learning rate 10^{-3}, batch size 64, at most 16 epochs, and validation-MSE stopping patience five. MSE and MAE aggregate all test elements and are reported as mean $\pm$ sample standard deviation over five paired seeds (2025–2029). Relative MSE reductions are averaged across seeds with descriptive 95% Student-t intervals. Training-window strides, baseline settings, and statistical details appear in Section B.

5.2 Forecasting Performance

Tables 1 and 2 report forecasting accuracy across datasets, horizons, and backbones. Models within each setting share data windows and validation-based checkpoint selection. Lower MSE and MAE indicate better forecasts.

Comparison with forecasting baselines. Band-sparse FIST attains MSEs of 0.3989/0.0876 on ETTh1/Exchange, compared with 0.4022/0.0920 for iTransformer; MAE follows the same ordering. DLinear has lower MSE on both datasets, while FIST has slightly lower MAE on ETTh1. The rankings thus depend on the metric and dataset under this fixed training budget.

Table 2: Core interaction comparison: MSE, mean ± standard deviation over five seeds. Each horizon is trained separately. MLP denotes the second backbone. Full MAE results appear in Section D.

Dataset, H, encoder	Identity	Shared sparse	Band sparse
Weather, 96, GRU	0.1838 ± 0.0012	0.1837 ± 0.0011	0.1837 ± 0.0012
ETTm2, 96, GRU	0.1892 ± 0.0025	0.1893 ± 0.0024	0.1889 ± 0.0019
ETTh1, 336, GRU	0.4953 ± 0.0035	0.4958 ± 0.0035	0.4941 ± 0.0027
Exchange, 336, GRU	0.3583 ± 0.0209	0.3581 ± 0.0222	0.3581 ± 0.0213
ETTh1, 96, MLP	0.3978 ± 0.0031	0.3985 ± 0.0031	0.3985 ± 0.0031

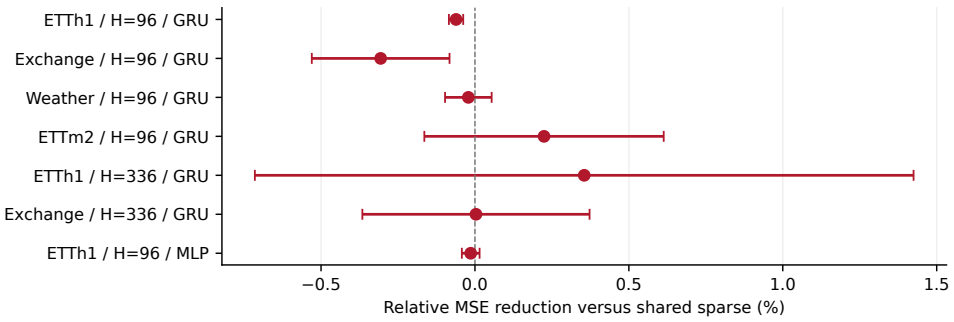

Figure 2: Band-sparse MSE reduction relative to shared-sparse. Means and descriptive 95% Student-t intervals use five paired seeds on fixed test data. Positive values favor band conditioning.

Contribution of band conditioning. Band-sparse has lower mean MSE than shared-sparse in three of seven settings, with mean paired reductions from -0.31% to $+0.35\%$ (Figure 2). None of the seven descriptive intervals lies entirely above zero. ETTm2 at $H = 96$ and ETTh1 at $H = 336$ show positive means; ETTh1/Exchange at $H = 96$ favor sharing. The competitive identity control indicates limited additional predictive value from interaction here. Complete metrics appear in Section D.

5.3 Ablation Study

Figure 3 isolates components on ETTh1/Exchange at $H = 96$, holding the backbone, forecast heads, and training protocol fixed.

Interaction structure and sparsity. On ETTh1, sparsifying a shared graph reduces mean paired MSE by 0.107%, while replacing shared-sparse with band-sparse increases it by 0.061%. Identity is the strongest internal control on ETTh1; shared-sparse performs best on Exchange. Sparsity and band conditioning therefore have distinct effects on prediction.

Similarity prior and training objective. Adding the prior and auxiliary spectral objective to their respective band-sparse controls reduces mean paired MSE on ETTh1 by 0.019% and 0.149%. Exchange shows weaker or less consistent effects, with both intervals crossing zero. These comparisons separate gains due to supervision or the prior from those attributable to band conditioning.

5.4 Stability and Model Diagnostics

Training and prediction behavior. Validation trajectories largely overlap across interaction configurations, with earlier checkpoint selection on ETTm2 than Weather (Figure 6). Channel-level errors reveal both gains and degradations within a dataset (Figure 8). Forecasts selected by a fixed chronological and column-order rule often overlap and retain visible errors, including a missed Weather decrease (Figure 7). These diagnostics complement the complete-test metrics.

Sensitivity to learned interactions. We cyclically shift graphs across bands or replace them with identity at fixed checkpoints (Section F). Band shifting slightly improves MSE on ETTh1 but has

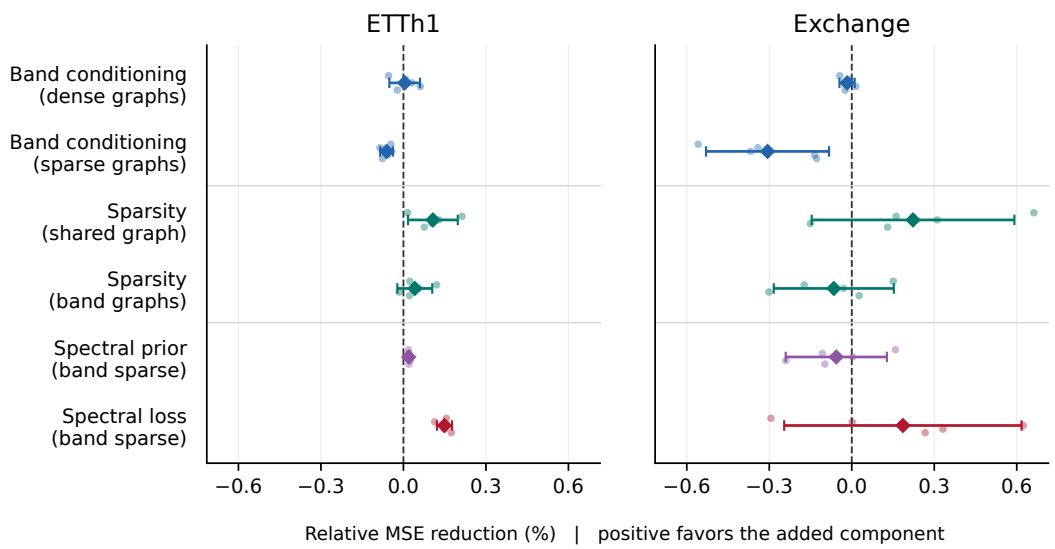

Figure 3: Component ablations at $H = 96$. Each row changes its named factor. Dots show five paired seed differences; diamonds and bars show means and unadjusted 95% Student-t intervals. Positive values favor the added component.

Table 3: Controlled coefficient regression with 128 training examples. Test complex MSE per entry, mean $\pm$ sample standard deviation over ten realizations. These are fitted linear operators, not end-to-end forecasting models.

Operator	Homogeneous	Heterogeneous	Independent
Diagonal	0.37144 ± 0.00108	0.37150 ± 0.00089	0.01005 ± 0.00003
Shared dense	0.01013 ± 0.00004	0.28375 ± 0.00119	0.01013 ± 0.00004
Shared sparse	0.01003 ± 0.00003	0.34842 ± 0.00101	0.01005 ± 0.00003
Band dense	0.01051 ± 0.00004	0.01051 ± 0.00004	0.01051 ± 0.00004
Band sparse	0.01009 ± 0.00003	0.01009 ± 0.00003	0.01020 ± 0.00003

seed-dependent effects on Exchange (Figure 10). Different graphs alone therefore do not establish predictive specialization. These interventions and configuration comparisons assess selected sources of variation; input-corruption robustness and hyperparameter sensitivity require separate experiments.

5.5 Controlled Dependency Analysis

We isolate dependency heterogeneity through coefficient regression with four bands and 12 channels. Mappings are shared, band-specific, or diagonal; the noise level is 0.01 per entry. Diagonal, shared, and band-specific ridge fits, including sparse variants, are evaluated over ten realizations with 2,048 test examples each. Generation and validation details appear in Section C.

The role of dependency heterogeneity. Band-specific fits approach the noise level under heterogeneous mappings (Table 3); sharing incurs population excess risk 0.27 per entry (Equation (11)). Shared fits are more sample-efficient under homogeneous mappings, and diagonal fits suffice under independence (Figure 4). These regimes explain a potential sharing cost, while the forecasting results measure its relevance after representation learning.

5.6 Computational Efficiency

On ETTh1 at $H = 96$, graph variants register 34,000 parameters, versus 18,624 for DLinear and 224,224 for iTransformer (Section D). Timing isolates the graph-layer implementation.

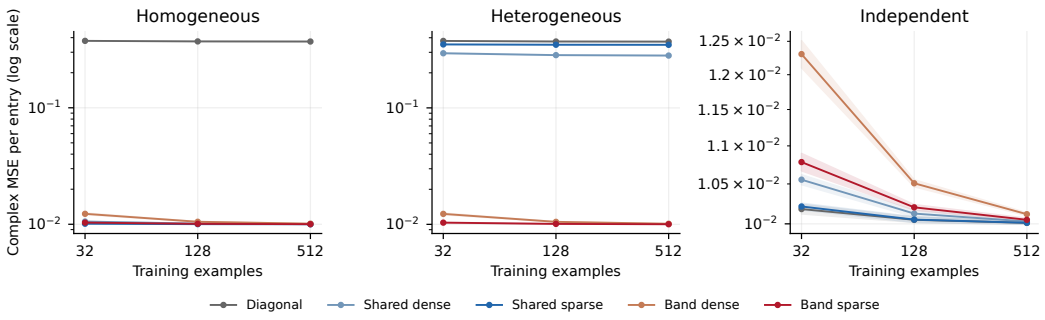

Figure 4: Controlled regression across training sample sizes, under shared, band-specific, or diagonal mappings. Shading denotes one standard deviation over ten realizations.

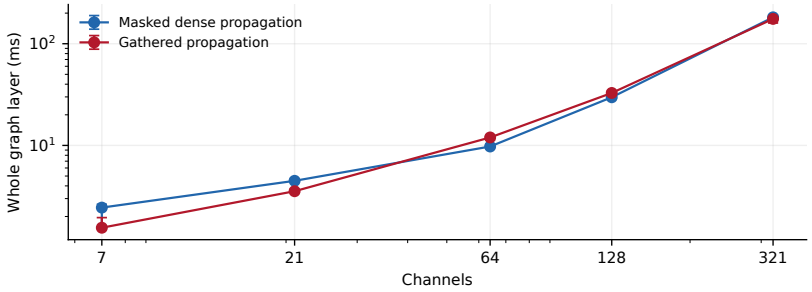

Figure 5: Complete graph-layer CPU latency for masked dense propagation and gathered messages with identical sparse supports. Batch size 4, 32 tokens, width 32, four bands, and four off-diagonal neighbors are fixed. Both implementations include full pairwise graph construction.

At $C = 321$, median complete-layer latency is 181.79 ms for masked dense propagation and 175.47 ms for gathered propagation (Figure 5). Each measurement uses five warm-up calls and 20 timed calls without gradients; error bars show interquartile ranges. Both implementations use the same initialization and inputs. Pairwise affinity construction remains quadratic in the number of channels. Eleven executable checks verify support selection, channel independence, and numerical agreement between propagation backends, among other implementation properties (Section G).

Scope and limitations. The compact budget and dataset coverage limit forecasting conclusions. Broader evaluation requires recent dependency-selection baselines, larger budgets, and higher-dimensional data. Latent frequency resolution and the linear assumptions further constrain interpretation; learned graphs describe predictive associations, without establishing causal structure.

6 Conclusion

We introduced FIST, which models cross-channel interactions with sparse graphs conditioned on latent temporal frequency. Linear analysis and controlled regression characterize when band-specific interactions can reduce the cost of sharing a single operator. Matched forecasting experiments, however, show small and setting-dependent gains after representation learning. These results clarify when frequency-conditioned interaction is useful and motivate further study with finer frequency resolution and larger-scale evaluation.

AI Use Statement

Generative AI tools were used for language polishing and improving the readability of the manuscript. All AI-assisted edits were reviewed by the authors.

REFERENCES

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*, 2020. URL https://proceedings.neurips.cc/paper/2020/hash/ce1aad92b939420fc17005e5461e6f48-Abstract.html.

Si-An Chen, Chun-Liang Li, Nathanael C. Yoder, Sercan O. Arik, and Tomas Pfister. TSMixer: An all-MLP architecture for time series forecasting. *Transactions on Machine Learning Research*, 2023. URL https://openreview.net/forum?id=wbpxTuXgm0.

Albert Gu and Tri Dao. Mamba: Linear-time sequence modeling with selective state spaces. In *Conference on Language Modeling*, 2024. URL https://arxiv.org/abs/2312.00752.

Lu Han, Xu-Yang Chen, Han-Jia Ye, and De-Chuan Zhan. SOFTS: Efficient multivariate time series forecasting with series-core fusion. In *Advances in Neural Information Processing Systems*, 2024. URL https://proceedings.neurips.cc/paper_files/paper/2024/hash/754612bde73a8b65ad8743f1f6d8ddf6-Abstract-Conference.html.

Yifan Hu, Guibin Zhang, Peiyuan Liu, Disen Lan, Naiqi Li, Dawei Cheng, Tao Dai, Shu-Tao Xia, and Shirui Pan. TimeFilter: Patch-specific spatial-temporal graph filtration for time series forecasting. In *International Conference on Machine Learning*, pp. 24893–24911, 2025. URL https://proceedings.mlr.press/v267/hu25ac.html.

Taesung Kim, Jinhee Kim, Yunwon Tae, Cheonbok Park, Jang-Ho Choi, and Jaegul Choo. Reversible instance normalization for accurate time-series forecasting against distribution shift. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=cGDAkQo1C0p.

Guokun Lai, Wei-Cheng Chang, Yiming Yang, and Hanxiao Liu. Modeling long- and short-term temporal patterns with deep neural networks. In *International ACM SIGIR Conference on Research and Development in Information Retrieval*, 2018. URL https://arxiv.org/abs/1703.07015.

Dongyuan Li, Shun Zheng, Chang Xu, Jiang Bian, and Renhe Jiang. Routing channel-patch dependencies in time series forecasting with graph spectral decomposition. In *International Conference on Learning Representations*, 2026. URL https://arxiv.org/abs/2603.13702.

Yaguang Li, Rose Yu, Cyrus Shahabi, and Yan Liu. Diffusion convolutional recurrent neural network: Data-driven traffic forecasting. In *International Conference on Learning Representations*, 2018. URL https://openreview.net/forum?id=SJiHXGWAZ.

Yong Liu, Tengge Hu, Haoran Zhang, Haixu Wu, Shiyu Wang, Lintao Ma, and Mingsheng Long. iTransformer: Inverted transformers are effective for time series forecasting. In *International Conference on Learning Representations*, 2024. URL https://arxiv.org/abs/2310.06625.

Yuqi Nie, Nam H. Nguyen, Phanwadee Sinthong, and Jayant Kalagnanam. A time series is worth 64 words: Long-term forecasting with transformers. In *International Conference on Learning Representations*, 2023. URL https://arxiv.org/abs/2211.14730.

Xiangfei Qiu, Xingjian Wu, Yan Lin, Chenjuan Guo, Jilin Hu, and Bin Yang. DUET: Dual clustering enhanced multivariate time series forecasting. In *ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 2025. URL https://arxiv.org/abs/2412.10859.

Hao Wang, Licheng Pan, Zhichao Chen, Degui Yang, Sen Zhang, Yifei Yang, Xinggao Liu, Haoxuan Li, and Dacheng Tao. FreDF: Learning to forecast in the frequency domain. In *International Conference on Learning Representations*, 2025. URL https://arxiv.org/abs/2402.02399.

Shiyu Wang, Haixu Wu, Xiaoming Shi, Tengge Hu, Huakun Luo, Lintao Ma, James Y. Zhang, and Jun Zhou. TimeMixer: Decomposable multiscale mixing for time series forecasting. In *International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=7oLshfEIC2.

Haixu Wu, Jiehui Xu, Jianmin Wang, and Mingsheng Long. Autoformer: Decomposition transformers with auto-correlation for long-term series forecasting. In *Advances in Neural Information Processing Systems*, 2021. URL https://proceedings.neurips.cc/paper/2021/hash/bcc0d400288793e8bdcd7c19a8ac0c2b-Abstract.html.

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

Zonghan Wu, Shirui Pan, Guodong Long, Jing Jiang, and Chengqi Zhang. Graph WaveNet for deep spatial-temporal graph modeling. In *International Joint Conference on Artificial Intelligence*, 2019. URL https://www.ijcai.org/proceedings/2019/264.

Zonghan Wu, Shirui Pan, Guodong Long, Jing Jiang, and Chengqi Zhang. Connecting the dots: Multivariate time series forecasting with graph neural networks. In *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2020.

Kun Yi, Qi Zhang, Wei Fan, Hui He, Liang Hu, Pengyang Wang, Ning An, Longbing Cao, and Zhendong Niu. FourierGNN: Rethinking multivariate time series forecasting from a pure graph perspective. In *Advances in Neural Information Processing Systems*, 2023a. URL https://arxiv.org/abs/2311.06190.

Kun Yi, Qi Zhang, Wei Fan, Shoujin Wang, Pengyang Wang, Hui He, Ning An, Defu Lian, Longbing Cao, and Zhendong Niu. Frequency-domain MLPs are more effective learners in time series forecasting. In *Advances in Neural Information Processing Systems*, 2023b. URL https://proceedings.neurips.cc/paper/2023/hash/f1d16af76939f476b5f040fd1398c0a3-Abstract.html.

Ailing Zeng, Muxi Chen, Lei Zhang, and Qiang Xu. Are transformers effective for time series forecasting? In *AAAI Conference on Artificial Intelligence*, 2023. URL https://arxiv.org/abs/2205.13504.

Yunhao Zhang and Junchi Yan. Crossformer: Transformer utilizing cross-dimension dependency for multivariate time series forecasting. In *International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=vSVLM2j9eie.

Haoyi Zhou, Shanghang Zhang, Jieqi Peng, Shuai Zhang, Jianxin Li, Hui Xiong, and Wancai Zhang. Informer: Beyond efficient transformer for long sequence time-series forecasting. In *AAAI Conference on Artificial Intelligence*, pp. 11106–11115, 2021. URL https://github.com/zhouhaoyi/ETDataset.

Tian Zhou, Ziqing Ma, Qingsong Wen, Xue Wang, Liang Sun, and Rong Jin. FEDformer: Frequency enhanced decomposed transformer for long-term series forecasting. In *International Conference on Machine Learning*, 2022. URL https://proceedings.mlr.press/v162/zhou22g.html.

A EXACT OBJECTIVES AND THE LINEAR CALCULATION

Proof of Proposition 1. The residual is $y_b - Tx_b = (T_b - T)x_b + \varepsilon_b$. Orthogonality of the input and noise eliminates the cross term, and whitening gives

$$\mathbb{E}\|y_b - Tx_b\|_2^2 = \|T_b - T\|_F^2 + \mathbb{E}\|\varepsilon_b\|_2^2. \tag{12}$$

Writing $T_b - T = (T_b - \bar{T}) + (\bar{T} - T)$ and using $\sum_b p_b(T_b - \bar{T}) = 0$ yields

$$\sum_b p_b\|T_b - T\|_F^2 = \sum_b p_b\|T_b - \bar{T}\|_F^2 + \|T - \bar{T}\|_F^2. \tag{13}$$

The minimum occurs at $T = \bar{T}$. This weighted variance identity is used to characterize one sharing constraint; it is not claimed as a new general theorem about neural networks. With nonidentity input covariances, the risk contains covariance-weighted quadratic forms, and the unweighted Frobenius expression need not apply. Finite-sample estimation and learned representations add issues outside the proposition.

Single-sided Fourier squared loss. For real forecast errors $e \in \mathbb{R}^{H\times C}$ and orthonormal rFFT coefficients E, define $w_f = 1$ at DC and at Nyquist when H is even, and $w_f = 2$ at all other retained bins. Then

$$\frac{1}{HC}\sum_{f,c} w_f|E_{f,c}|^2 = \frac{1}{HC}\sum_{t,c}|e_{t,c}|^2. \tag{14}$$

For even H, the unweighted one-sided energy is instead

$$\sum_{f=0}^{H/2}|E_f|^2 = \tfrac{1}{2}\|e\|_2^2 + \tfrac{1}{2}(|E_0|^2 + |E_{H/2}|^2), \tag{15}$$

with the channel sum understood. A different FFT normalization introduces the corresponding global scale. Neither expression establishes high-frequency compensation. In contrast, the complex-modulus term in Equation (10) scales linearly when e is multiplied by a positive constant; MSE scales quadratically. Its exact gradient and frequency weighting differ from squared error, without a guarantee of superior forecasting.

B FORECASTING DATA AND COMPLETE CONFIGURATION

Dataset	H	C	Target boundaries	Train/val/test windows	Train stride
ETTh1	96	7	8640/11520/14400	2,113/2,785/2,785	4
Exchange	96	8	5311/6071/7588	1,280/665/1,422	4
Weather	96	21	36887/42157/52696	2,294/5,175/10,444	16
ETTm2	96	7	34560/46080/57600	2,149/11,425/11,425	16
ETTh1	336	7	8640/11520/14400	2,053/2,545/2,545	4
Exchange	336	8	5311/6071/7588	1,220/425/1,182	4

Table 4: Chronological boundaries are zero-based exclusive endpoints. ETT uses its standard 20-month protocol interval; later file rows are not used. Validation and test strides are one.

Public sources. ETT data are downloaded from the maintainers' ETDataset repository (Zhou et al., 2021). Exchange uses the public multivariate-series release associated with LSTNet (Lai et al., 2018); the CSV conversion preserves all eight numeric columns and their order. Weather is downloaded from the Time-Series-Library dataset release. Dataset URLs and original/downloaded file SHA256 checksums are in `data/manifest.json`. Numeric means and standard deviations are fitted on the original, unwindowed training interval, not on overlapping windows. Constant channels use a unit scaling denominator. Nonfinite input data cause an explicit error.

Windows and horizons. Let the training, validation, and test target boundaries be t_1, t_2, t_3. Training target starts are $L, L + s_{\text{train}}, \ldots, t_1 - H$; validation starts are $t_1, \ldots, t_2 - H$, and test starts are $t_2, \ldots, t_3 - H$. Each example uses $[t - L, t)$ as input and $[t, t + H)$ as target. Thus preceding history may overlap an earlier split, but no target interval crosses a boundary. Separate models are fit for $H = 96$ and $H = 336$. Evaluating a prefix of a model trained for $H = 720$ is a different protocol and is not used in the reported study.

Setting	CPU study value
Lookback / patch / stride	96 / 16 / 8
Width / temporal layers / graph layers	16 / 1 / 1
Temporal encoder	Bidirectional GRU; token MLP in the second-backbone comparison
Bands / band widths / off-diagonal neighbors	4 / $(2, 2, 1, 1)$ / 2
Dropout / moving-average width	0.1 / 25
Initial graph gate / initial prior coefficient	-2 / 1
Frequency loss weight	0.1; 0 for the MSE-only control
Optimizer / learning rate / gradient clipping	Adam / 10^{-3} / global norm 1
Batch / epoch cap / early-stop patience	64 / 16 / 5
Checkpoint selection	Lowest element-weighted validation MSE
Seeds	2025, 2026, 2027, 2028, 2029
Runtime	Python 3.12; PyTorch 2.5.1 CPU; one thread per training job

Table 5: Actual configuration for the newly executed study. These values must not be conflated with an unexecuted full-width Mamba configuration.

Architecture and optimization. The FFN expands the feature dimension by four. The GRU has hidden size D in each direction, followed by a $2D \to D$ merge. The MLP temporal block has widths $D \to 2D \to D$; its flattened forecast head combines information across the token axis. Graph variants instantiate the same parameters in the same order. Identity graphs do not use the query/key scores to exchange observations, so equal registered parameter counts do not imply equal effective capacity. Shared graph scores average the band's $C \times C$ score matrices before support selection. The independent, shared-sparse, and band-sparse variants have identical registered parameters for each task.

External baselines. DLinear uses a replicate-padded moving average of width 25 and two shared linear $L \to H$ maps, with weights initialized to $1/L$. It is trained using MSE. iTransformer uses the released source files, identified by SHA256 in the supplementary code, with width 128, two encoder layers, eight heads, FFN width 128, dropout 0.1, and input normalization enabled. It predicts all channels and does not equate channel count to embedding width. All models receive numeric channel histories only; calendar covariate tokens are not supplied to iTransformer. It uses the same CPU study windows, Adam learning rate 10^{-3}, and validation criterion. These are explicit controlled-budget configurations; the iTransformer result is not claimed to reproduce every setting in its original paper. The frozen study matrix has no task-specific test-driven search.

Metrics and uncertainty. For each run, $\text{MSE} = \sum e^2/n$ and $\text{MAE} = \sum |e|/n$, where n counts all forecast elements. RMSE is computed after global MSE aggregation and provides no independent ranking evidence. Error sums for every test window and channel, together with target start indices, are saved. The first 64 forecast windows are also saved as a diagnostic sample; they are not the dataset used to compute the main scores. Complete predictions can be regenerated from the selected checkpoints and data files.

For paired seeds r, the relative improvement against comparator v is $d_r = 100(\text{MSE}_{v,r} - \text{MSE}_{\text{band},r})/\text{MSE}_{v,r}$. The reported interval is $\bar{d} \pm t_{4,0.975} s_d/\sqrt{5}$. This is a small-sample, normality-based summary over training seeds. It does not quantify population uncertainty in the time series.

Forecast horizons and overlapping windows are not counted as extra independent seeds. The statistics generator fails when a required result or seed is missing rather than inserting a guessed value.

C CONTROLLED OPERATOR EXPERIMENT

For every training size $n \in \{32, 128, 512\}$ and seed $s \in \{4100, \ldots, 4109\}$, a local NumPy random generator constructs mutually independent training, validation, and test samples. Inputs have real and imaginary parts distributed as $\mathcal{N}(0, 1/2)$, giving $\mathbb{E}[xx^*] = I$. Noise quadratures have standard deviation $0.1/\sqrt{2}$. The matrices are row-stochastic, with diagonal weight 0.4 and one off-diagonal weight 0.6 in the dependent conditions. Heterogeneous band $b \in \{0, 1, 2, 3\}$ uses a cyclic offset $b + 1$; homogeneous bands all use offset one.

Real operators are estimated by stacking the real and imaginary observations. The ridge penalty is selected separately for each estimator from $\{10^{-4}, 10^{-2}, 10^{-1}, 1, 10\}$ using 256 held-out validation examples. Shared estimators pool all bands; band estimators fit each band separately. Sparse estimators first fit a dense operator, choose the largest-magnitude off-diagonal coefficient in each row, and refit the retained self-plus-neighbor support. This is a support-selection diagnostic with different effective degrees of freedom, not a parameter-matched neural ablation. The test set contains 2,048 examples. Ground-truth support is used for evaluation only, never for selecting a learned support.

Condition	Operator	$n = 32$	$n = 128$	$n = 512$
Homogeneous	Diagonal	0.37547 ± 0.00083	0.37144 ± 0.00108	0.37058 ± 0.00142
Homogeneous	Shared dense	0.01055 ± 0.00006	0.01013 ± 0.00004	0.01003 ± 0.00002
Homogeneous	Shared sparse	0.01011 ± 0.00004	0.01003 ± 0.00003	0.01001 ± 0.00002
Homogeneous	Band dense	0.01230 ± 0.00020	0.01051 ± 0.00004	0.01012 ± 0.00002
Homogeneous	Band sparse	0.01034 ± 0.00006	0.01009 ± 0.00003	0.01002 ± 0.00002
Heterogeneous	Diagonal	0.37573 ± 0.00127	0.37150 ± 0.00089	0.37061 ± 0.00124
Heterogeneous	Shared dense	0.29442 ± 0.00230	0.28375 ± 0.00119	0.28133 ± 0.00115
Heterogeneous	Shared sparse	0.35061 ± 0.00122	0.34842 ± 0.00101	0.34785 ± 0.00116
Heterogeneous	Band dense	0.01230 ± 0.00020	0.01051 ± 0.00004	0.01012 ± 0.00002
Heterogeneous	Band sparse	0.01034 ± 0.00006	0.01009 ± 0.00003	0.01002 ± 0.00002
Independent	Diagonal	0.01018 ± 0.00006	0.01005 ± 0.00003	0.01001 ± 0.00002
Independent	Shared dense	0.01055 ± 0.00006	0.01013 ± 0.00004	0.01003 ± 0.00002
Independent	Shared sparse	0.01021 ± 0.00004	0.01005 ± 0.00003	0.01001 ± 0.00002
Independent	Band dense	0.01231 ± 0.00020	0.01051 ± 0.00004	0.01012 ± 0.00002
Independent	Band sparse	0.01078 ± 0.00011	0.01020 ± 0.00003	0.01005 ± 0.00002

Table 6: Complete coefficient-regression results. Complex MSE per entry, mean $\pm$ sample standard deviation over ten generator realizations. The diagonal estimator learns a separate self coefficient rather than fixing it to one.

Structure recovery scope. The released raw results include off-diagonal average precision where positive edges exist. For the independent condition, there are no positive off-diagonal edges, so that metric is undefined and stored as null. This structural metric concerns recovery by the controlled linear estimator under the known generator. It is not a recovery claim for FIST's latent, input-dependent graphs. A one-seed cyclic band shift is also retained as an inference-time diagnostic; it is not counted among the ten-seed trained-estimator comparisons.

D COMPLETE FORECASTING RESULTS

Variant	MSE	MAE	Parameters	Selected epochs
Identity	0.3980 ± 0.0033	0.4077 ± 0.0016	34,000	9, 14, 11, 14, 9
Shared dense	0.3991 ± 0.0031	0.4083 ± 0.0016	34,000	9, 14, 11, 14, 9
Shared sparse	0.3987 ± 0.0033	0.4081 ± 0.0017	34,000	9, 14, 11, 14, 9
Band dense	0.3991 ± 0.0032	0.4083 ± 0.0017	34,000	9, 14, 11, 14, 9
Band sparse	0.3989 ± 0.0033	0.4082 ± 0.0018	34,000	9, 14, 11, 14, 9
Without prior	0.3990 ± 0.0033	0.4083 ± 0.0018	34,000	9, 14, 11, 14, 9
MSE only	0.3995 ± 0.0034	0.4089 ± 0.0018	34,000	9, 14, 11, 14, 9
DLinear	0.3938 ± 0.0024	0.4090 ± 0.0036	18,624	7, 8, 7, 10, 13
iTransformer	0.4022 ± 0.0042	0.4172 ± 0.0034	224,224	6, 10, 6, 5, 4

Table 7: ETTh1, $H = 96$, GRU backbone for graph variants. Mean and sample standard deviation over seeds 2025–2029. Selected epochs are given in seed order.

Variant	MSE	MAE	Parameters	Selected epochs
Identity	0.0875 ± 0.0024	0.2060 ± 0.0027	34,002	7, 10, 10, 8, 9
Shared dense	0.0876 ± 0.0025	0.2060 ± 0.0028	34,002	7, 10, 10, 8, 9
Shared sparse	0.0874 ± 0.0025	0.2058 ± 0.0028	34,002	7, 10, 10, 8, 9
Band dense	0.0876 ± 0.0026	0.2060 ± 0.0028	34,002	7, 10, 10, 8, 9
Band sparse	0.0876 ± 0.0025	0.2061 ± 0.0028	34,002	7, 10, 10, 8, 9
Without prior	0.0876 ± 0.0025	0.2061 ± 0.0028	34,002	7, 10, 10, 8, 9
MSE only	0.0878 ± 0.0025	0.2065 ± 0.0028	34,002	7, 10, 10, 8, 9
DLinear	0.0785 ± 0.0011	0.1993 ± 0.0025	18,624	16, 9, 8, 9, 9
iTransformer	0.0920 ± 0.0067	0.2132 ± 0.0076	224,224	9, 6, 8, 3, 3

Table 8: Exchange, $H = 96$, GRU backbone for graph variants. Mean and sample standard deviation over seeds 2025–2029. Selected epochs are given in seed order.

Variant	MSE	MAE	Parameters	Selected epochs
Identity	0.1838 ± 0.0012	0.2249 ± 0.0012	34,028	16, 16, 16, 15, 15
Shared sparse	0.1837 ± 0.0011	0.2249 ± 0.0010	34,028	16, 16, 16, 15, 15
Band sparse	0.1837 ± 0.0012	0.2248 ± 0.0011	34,028	16, 16, 16, 15, 15

Table 9: Weather, $H = 96$, GRU backbone for graph variants. Mean and sample standard deviation over seeds 2025–2029. Selected epochs are given in seed order.

Variant	MSE	MAE	Parameters	Selected epochs
Identity	0.1892 ± 0.0025	0.2733 ± 0.0026	34,000	2, 4, 4, 4, 4
Shared sparse	0.1893 ± 0.0024	0.2734 ± 0.0025	34,000	2, 4, 4, 4, 4
Band sparse	0.1889 ± 0.0019	0.2731 ± 0.0023	34,000	3, 4, 4, 4, 4

Table 10: ETTm2, $H = 96$, GRU backbone for graph variants. Mean and sample standard deviation over seeds 2025–2029. Selected epochs are given in seed order.

Variant	MSE	MAE	Parameters	Selected epochs
Identity	0.4953 ± 0.0035	0.4617 ± 0.0052	99,760	9, 14, 4, 12, 7
Shared sparse	0.4958 ± 0.0035	0.4619 ± 0.0051	99,760	9, 14, 4, 12, 7
Band sparse	0.4941 ± 0.0027	0.4594 ± 0.0043	99,760	9, 6, 4, 12, 7

Table 11: ETTh1, $H = 336$, GRU backbone for graph variants. Mean and sample standard deviation over seeds 2025–2029. Selected epochs are given in seed order.

Variant	MSE	MAE	Parameters	Selected epochs
Identity	0.3583 ± 0.0209	0.4327 ± 0.0131	99,762	7, 9, 9, 12, 6
Shared sparse	0.3581 ± 0.0222	0.4326 ± 0.0140	99,762	7, 9, 9, 12, 6
Band sparse	0.3581 ± 0.0213	0.4326 ± 0.0134	99,762	7, 9, 9, 12, 6

Table 12: Exchange, $H = 336$, GRU backbone for graph variants. Mean and sample standard deviation over seeds 2025–2029. Selected epochs are given in seed order.

Variant	MSE	MAE	Parameters	Selected epochs
Identity	0.3978 ± 0.0031	0.4068 ± 0.0020	31,280	9, 14, 11, 14, 9
Shared sparse	0.3985 ± 0.0031	0.4072 ± 0.0020	31,280	9, 14, 11, 14, 9
Band sparse	0.3985 ± 0.0031	0.4073 ± 0.0020	31,280	9, 14, 11, 14, 9

Table 13: ETTh1, $H = 96$, MLP backbone for graph variants. Mean and sample standard deviation over seeds 2025–2029. Selected epochs are given in seed order.

Dataset, H, encoder	Versus shared sparse	Versus identity
ETTh1, 96, GRU	$-0.06\ [-0.08, -0.04]$	$-0.23\ [-0.27, -0.20]$
Exchange, 96, GRU	$-0.31\ [-0.53, -0.08]$	$-0.17\ [-0.55, +0.21]$
Weather, 96, GRU	$-0.02\ [-0.10, +0.05]$	$+0.01\ [-0.05, +0.07]$
ETTm2, 96, GRU	$+0.22\ [-0.16, +0.61]$	$+0.17\ [-0.23, +0.58]$
ETTh1, 336, GRU	$+0.35\ [-0.71, +1.42]$	$+0.25\ [-0.84, +1.33]$
Exchange, 336, GRU	$+0.00\ [-0.37, +0.37]$	$+0.06\ [-0.15, +0.27]$
ETTh1, 96, MLP	$-0.01\ [-0.04, +0.02]$	$-0.19\ [-0.21, -0.18]$

Table 14: Mean paired relative MSE reduction (%), with 95% Student-t intervals across five training seeds. Positive values favor band-sparse. These are descriptive, unadjusted comparisons.

E ADDITIONAL FORECASTING DIAGNOSTICS

The following views use the saved results of the same completed runs. They add no training configurations and do not change checkpoint selection. All four datasets use the GRU backbone and $H = 96$. The plotting script records its input checksums and the numerical values underlying each panel in `results/extended_visual_analysis.json`.

Training behavior. Figure 6 shows validation MSE at every recorded epoch. Each trajectory ends where that run actually stopped, and a marker identifies the selected checkpoint. The early minima on ETTm2 and later minima on Weather illustrate why an epoch cap alone does not describe the selected models. The curves largely overlap across the three graph configurations. We plot validation MSE rather than the training objective so that the vertical axis has the same meaning across variants.

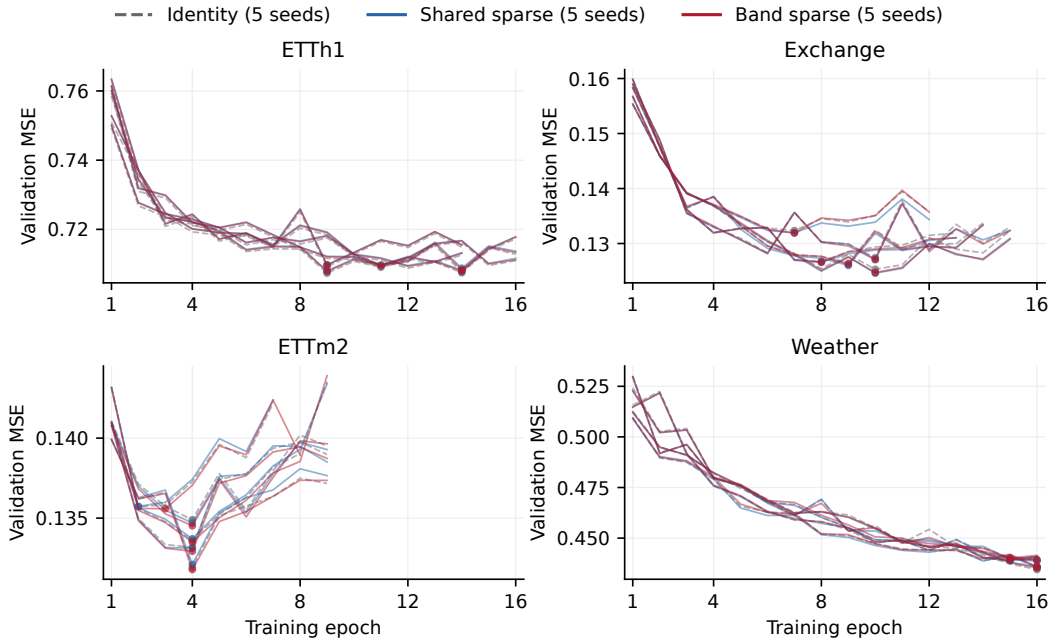

Figure 6: Validation trajectories for identity, shared-sparse, and band-sparse graphs. Each line is one of five seeds; circles mark the validation-selected epoch. Curves end at the last executed epoch, with no extrapolation or imputation. Axis ranges differ across datasets. These curves describe training under the fixed budget rather than a convergence guarantee.

Forecasts under an explicit selection rule. For every dataset, Figure 7 uses the first chronological test window and the first numeric channel in the source column order. The rule is fixed and independent of forecast error. Lines are predictions averaged over seeds; shading represents the band-sparse model's seed variation. These illustrative means are not ensembles evaluated in the main tables, which report metrics for individual runs before averaging. The examples retain visible errors, including the missed decrease in the Weather channel, and do not substitute for complete-test evaluation.

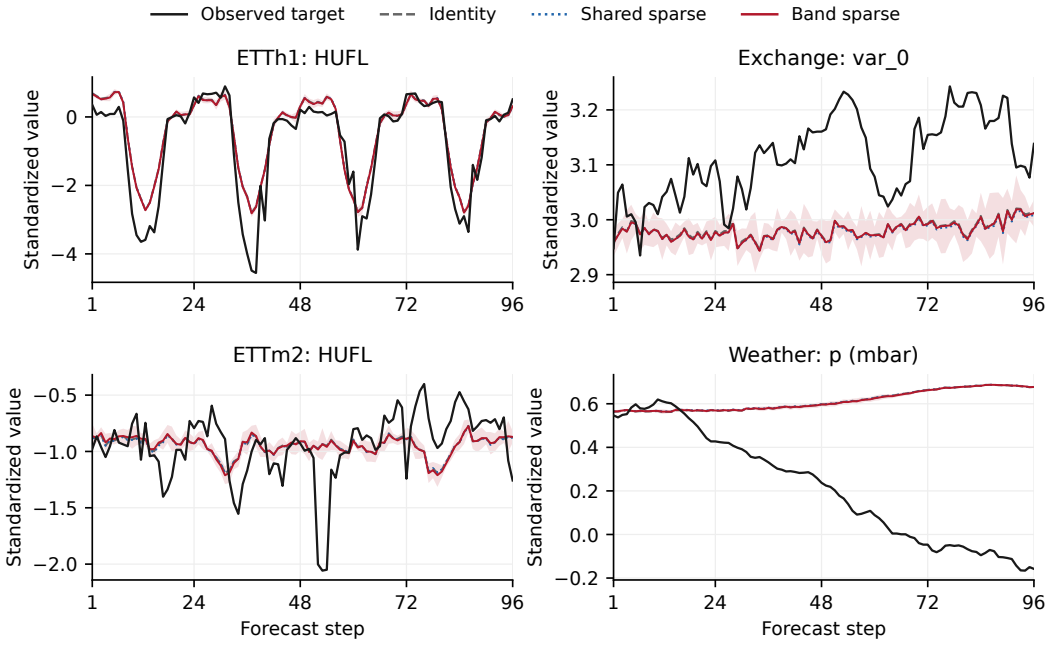

Figure 7: Forecasts for a specified test window and channel on each dataset, with all 96 forecast steps shown. Values use training-set standardization. Prediction lines are means over five seeds; red shading is one sample standard deviation for band-sparse, not a predictive interval. The three models often overlap closely. Selection is chronological and column-based, without choosing favorable cases.

Channel-level error differences. For each seed r and channel c, let

$$m_{v,r,c} = \frac{1}{WH} \sum_{w=1}^{W} \sum_{h=1}^{H} (\widehat{Y}_{v,r,w,h,c} - Y_{w,h,c})^2. \tag{16}$$

Figure 8 displays $\frac{1}{5} \sum_r 100(m_{v,r,c} - m_{\text{band},r,c})/m_{v,r,c}$ for identity and shared-sparse comparators, using every test window. This preserves differences hidden by the global average: on ETTm2, for example, improvements and degradations occur in different channels. A mean of channel-wise ratios is not the relative change of the globally aggregated MSE; the two summaries should not be interchanged.

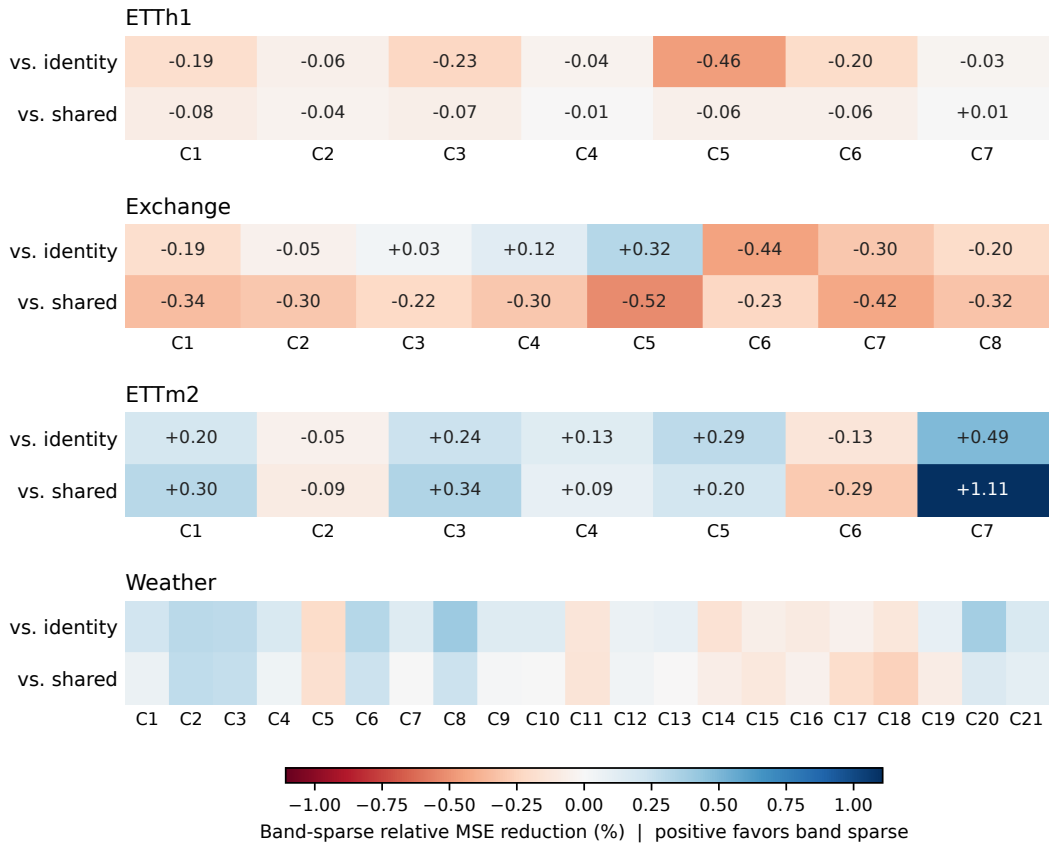

Figure 8: Mean paired relative MSE reduction for each channel at $H = 96$. Blue favors band-sparse; red favors the comparator. Both comparator rows and all four datasets share one symmetric, unclipped color scale. C1, C2, etc. follow the numeric column order in the data manifest and run metadata. All channels and all test windows are included; these descriptive means do not encode statistical significance.

F Learned Graphs and Intervention Diagnostics

We diagnose the five frozen band-sparse checkpoints on each of ETTh1 and Exchange at $H = 96$. For graph statistics, we average over the first 16 chronological test windows, all six band pairs, and off-diagonal supports. This window set is a descriptive sample, not an independent structural replication. A cyclic score-matrix shift changes which graph serves each band while leaving its spectral coefficients in place. An identity intervention removes cross-channel mixing while retaining trained gates, residuals, and normalizations. Both interventions are evaluated over all test windows without retraining. They introduce distribution shift and do not prove recovery of a ground-truth or causal graph.

Dataset	Jaccard	Self mass	Gate	Band shift	Identity
ETTh1	0.303 ± 0.010	0.407 ± 0.010	0.104 ± 0.006	-0.107 ± 0.049	-0.040 ± 0.056
Exchange	0.205 ± 0.020	0.330 ± 0.019	0.130 ± 0.003	0.216 ± 0.459	0.155 ± 0.485

Table 15: Graph statistics and frozen-checkpoint interventions, mean $\pm$ standard deviation across five seeds. The last two columns are relative MSE increases (%); positive values indicate that the intervention worsens prediction. Jaccard excludes self-loops.

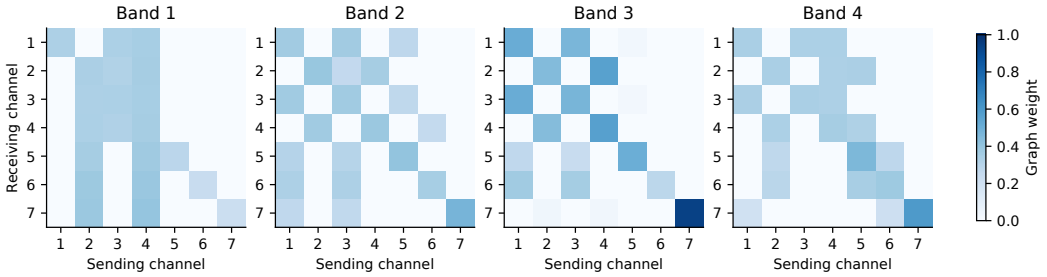

Figure 9: Band adjacencies from a specified test window of the first-seed ETTh1 model. The panels use the same color scale. Rows receive information from columns, self-loops are retained, and two off-diagonal neighbors are allowed. Different visual supports do not by themselves establish predictive usefulness or physical structure recovery.

Band overlap and predictive sensitivity. Figure 10 expands the aggregate graph table into band-pair and seed-level views. For supports E_b with self-loops removed, the overlap is $|E_b \cap E_{b'}|/|E_b \cup E_{b'}|$, computed within each window and then averaged over the first 16 windows and five seeds. Matrix diagonals are one by construction. The intervention panel uses complete-test errors from the same five checkpoints, with positive changes denoting worse prediction. The heterogeneous responses show why visually different band graphs are insufficient evidence that their assignment to bands is useful. In particular, the ETTh1 band shift reduces rather than increases error in these checkpoints.

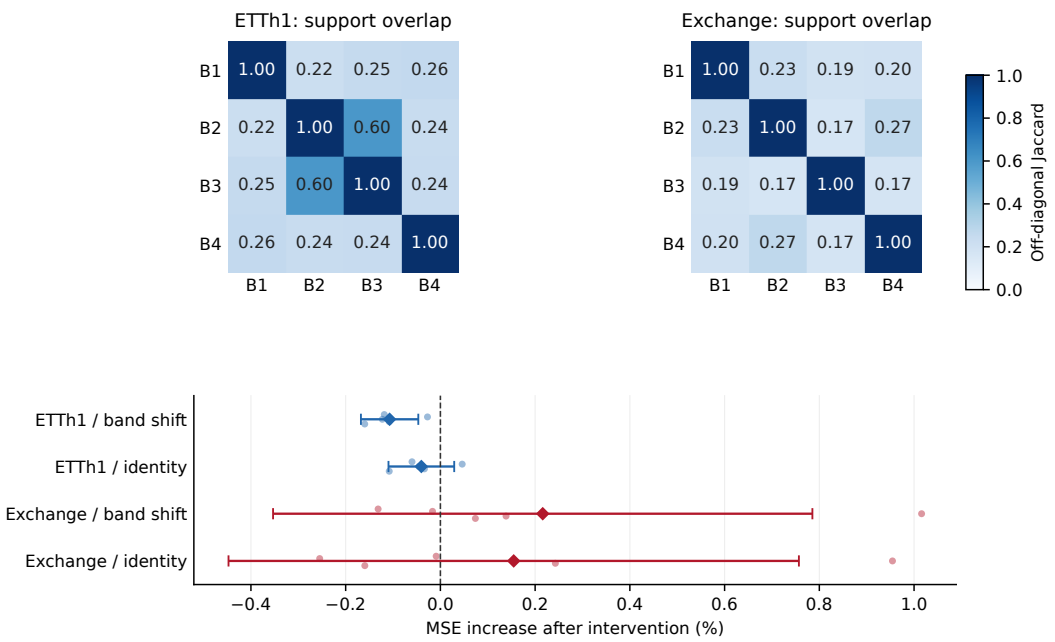

Figure 10: Structure and sensitivity of the trained band graphs at $H = 96$. Top: mean pairwise off-diagonal support Jaccard, on a common $[0, 1]$ scale. Bottom: MSE increase after a cyclic band shift or an identity substitution at inference. Dots show all five seeds; diamonds and bars give means and unadjusted 95% Student-t intervals. Models are frozen, and interventions are evaluated on every test window.

G Implementation and reproducibility checks

Unknown configuration keys and missing Mamba dependencies raise errors. Each run names its backbone. All-channel prediction and target shapes must match exactly. Real adjacency weights, scalar gates, and complex-modulus frequency loss match the equations above.

Eleven numerical tests check dense/gathered values and gradients, odd/even Parseval equality, channel independence under input perturbation, exact support under tied scores, finite zero-energy similarity, the scalar/vector distinction, invalid bands, training-only scaling, target boundaries, and metric invariance to uneven batches. These checks establish implementation properties, not forecasting superiority.

The earlier implementation had different support semantics, silent band-count capping, and discrepancies with its manuscript's loss, prior, propagation, and gate equations. Its six-dataset scores are therefore not transferred to this revised architecture. Historical measurements are preserved in the revision record; present tables use only newly executed results.

Remaining evaluation scope. The code supplies a larger-budget launcher, an explicit Mamba option, and a TimeFilter adapter; these unexecuted full runs are not empirical evidence. DUET and xCPD still require verified integrations and matched tuning. High-dimensional graph timing does not establish forecasting accuracy. No uncompleted experiment is assigned a numerical result.