STReasoner: Empowering LLMs for Spatio-Temporal Reasoning in Time Series via Spatial-Aware Reinforcement Learning

Juntong Ni[1], **Shiyu Wang***, **Qi He**[2], **Ming Jin**[3], **Wei Jin**[1*]
[1]Emory University, [2]Microsoft, [3]Griffith University
{juntong.ni, wei.jin}@emory.edu

Abstract

Spatio-temporal reasoning in time series involves the explicit synthesis of temporal dynamics, spatial dependencies, and textual context. This capability is vital for high-stakes decision-making in systems such as traffic networks, power grids, and disease propagation. However, the field remains underdeveloped because most existing works prioritize predictive accuracy over reasoning. To address the gap, we introduce **ST-Bench**, a benchmark consisting of four core tasks, including etiological reasoning, entity identification, correlation reasoning, and in-context forecasting, developed via a *network SDE-based multi-agent data synthesis pipeline*. We then propose **STReasoner**, which empowers LLM to integrate time series, graph structure, and text for explicit reasoning. To promote spatially grounded logic, we introduce *S-GRPO*, a reinforcement learning algorithm that rewards performance gains specifically attributable to spatial information. Experiments show that STReasoner achieves average accuracy gains between 17% and 135% at only 0.004× the cost of proprietary models and generalizes robustly to real-world data. Our code is available at https://github.com/LingFengGold/STReasoner.

1 Introduction

Time series data are ubiquitous in real-world systems and often exhibit complex spatio-temporal structures (Wang et al., 2020), from traffic networks (Liu et al., 2023) and climate systems (Lin et al., 2018) to disease propagation across regions (Liu et al., 2024d). As the community races to bring large language models (LLMs) to these domains (Jin et al., 2023b), it is critical to ensure that LMs can support decisions grounded in these data (Merrill et al., 2024).

*Corresponding Authors.

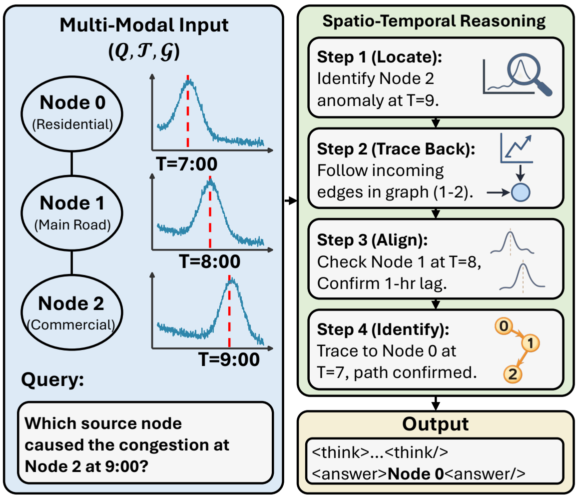


Figure 1: A traffic flow example of spatio-temporal reasoning. $\mathcal{Q}$: query, $\mathcal{T}$: time series, $\mathcal{G}$: graph.

However, most existing approaches primarily focus on predictive accuracy (Liang et al., 2025; Liu et al., 2024a,c; Li et al., 2024b), offering limited support for decision-making that requires explicit reasoning rather than point forecasts. Thus, we focus on *spatio-temporal reasoning*: the ability to answer natural-language queries about *what happened, where, when, and why* in a system whose state evolves over time and is coupled across space (e.g., a graph). Such queries require linking observations across *multiple nodes* and *multiple time steps* through the system's spatial dependencies and temporal dynamics, often involving propagation and time lag. As shown in Figure 1, when traffic congestion is observed, the query "Which source node caused the congestion at Node 2 at 9:00?" cannot be answered from a single-node prediction at 9:00; it requires tracing upstream dependencies in the traffic graph and connecting earlier upstream patterns to the downstream congestion under delay. Answering it thus integrates *textual intent*, *temporal evidence*, and *spatial structure*. Spatio-temporal reasoning of this form is essential

Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 15369–15416
July 2-7, 2026 ©2026 Association for Computational Linguistics

in many other critical systems, such as tracking viral spread across urban populations (Liu et al., 2024d), assessing cascading impacts on power grids under extreme weather (Panteli and Mancarella, 2015), and tracing flood propagation along river networks (Paola et al., 2006).

Despite its importance, spatio-temporal reasoning in time series remains underexplored. LLMs and vision language models (VLMs) demonstrate strong reasoning ability in textual and visual domains (Guo et al., 2025; Huang et al., 2025), but they are not explicitly trained or evaluated for reasoning over *spatio-temporal time series*. Existing spatio-temporal forecasting methods focus on predicting future values (Chen and Liang, 2025; Li et al., 2024b; Yuan et al., 2024), without explicitly reasoning, which limits their usefulness for decision-making. Meanwhile, time series language/reasoning models primarily consider univariate or multivariate series (Guan et al., 2025; Luo et al., 2025; Kong et al., 2025; Xie et al., 2024; Jin et al., 2023a; Wang et al., 2024c,b,a) and fail to consider *spatial relationship*. As a result, spatio-temporal reasoning in time series is inherently distinct from these paradigms.

Spatio-temporal reasoning in time series faces three key fundamental challenges that hinder its development: **(a) Data Challenges.** Existing spatio-temporal datasets rarely provide paired natural language descriptions to explain spatial entities, dependency, or temporal dynamics. Furthermore, current time series reasoning datasets often lack spatial interactions such as explicit graph structures. **(b) Evaluation Gaps.** There are no standardized, multi-dimensional benchmarks that decompose spatio-temporal reasoning into distinct reasoning tasks, making it difficult to systematically evaluate models' capabilities across different spatio-temporal reasoning demands. **(c) Modeling Limitations.** The effective architecture and optimization objective for spatio-temporal reasoning are largely underexplored. *First*, it remains unclear how to effectively fuse time series, graph structure, and textual information for reasoning without sacrificing numerical precision or global context. *Second*, it is unclear how to optimize models for spatio-temporal reasoning. Existing RL methods rely on result-only rewards (Guo et al., 2025) and fail to enforce spatial grounding, allowing models to exploit superficial temporal patterns rather than perform genuine spatial attribution.

To address these challenges, we propose three key solutions. **First**, we introduce an *SDE-based multi-agent data synthesis pipeline* (Xu et al., 2025), which generates time series with controllable spatial dependencies and temporal evolution, together with aligned textual descriptions. **Second**, building on this pipeline, we build *ST-Bench* and organize spatio-temporal reasoning into four tasks: *etiological spatial reasoning*, *spatial entity identification*, *spatial correlation reasoning*, and *in-context forecasting*, along with a real-world dataset for zero-shot evaluation. **Third**, we propose *STReasoner*, which integrates a dedicated time series encoder with LLM to process numerical time series and textual inputs jointly. To explicitly incentivize spatial reasoning during training, we propose *spatial-aware group relative policy optimization (S-GRPO)*, which assigns additional reward only when spatial information improves performance.

We conduct a comprehensive empirical study to evaluate spatio-temporal reasoning capabilities across existing language models and STReasoner. Our contributions can be summarized as:

(a) To the best of our knowledge, we take an early step toward formally studying the problem of spatio-temporal reasoning in time series.
(b) We propose an SDE-based multi-agent pipeline to synthesize spatio-temporal data with aligned textual descriptions. Based on this pipeline, we construct ST-Bench to support evaluation.
(c) We propose STReasoner, a unified spatio-temporal reasoning model, together with a spatial-aware RL objective.
(d) Through systematic evaluation, we show that existing LMs struggle with spatio-temporal reasoning, while STReasoner achieves strong performance and robust zero-shot generalization at substantially lower cost.

2 Related Work

Recent work has improved the reasoning ability of LLMs and extended them to non-text modalities (Wei et al., 2022; Guo et al., 2025; Huang et al., 2025). Meanwhile, time series language and reasoning models support QA and multi-step reasoning, but largely focus on univariate or multivariate series without spatial dependency (Jin et al., 2023a; Xie et al., 2024; Wang et al., 2025b, 2024b, 2025c; Wang, 2024). Spatio-temporal reasoning has also been explored in videos and trajectories (Feng et al., 2025c; Li et al., 2025). In contrast, spatio-temporal reasoning in time series involves numerical signals on discrete entities, with graph-defined spatial dependencies and induced temporal dynamics. A complete discussion of related work is provided in Appendix A.

3 Problem and Dataset Construction

3.1 Problem Definition

We define spatio-temporal reasoning in time series as follows: given a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ with N nodes, a set of time series $\mathcal{T} = \{T_1, \ldots, T_N\}$, where each $T_i = \{x_{i,1}, x_{i,2}, \ldots, x_{i,T}\}, \quad x_{i,t} \in \mathbb{R}$ represents a sequence of T observed values over time for the node $v_i \in \mathcal{V}$ and a natural language query Q, the goal is to generate a text-based response that requires explicit reasoning over temporal dynamics and spatial dependencies. The model produces an intermediate reasoning sequence R and a final answer A. Formally, the task is defined as $f : (Q, \mathcal{T}, \mathcal{G}) \to (R, A)$, where f denotes a model that interprets the query Q and analyzes the time series $\mathcal{T}$ under the structural constraints imposed by the graph $\mathcal{G}$ to generate the reasoning process R and the final answer A.

3.2 Dataset Synthesis Pipeline

Motivation. High-quality datasets for spatio-temporal reasoning in time series are scarce, especially those that simultaneously provide controllable spatial dependencies, rich temporal dynamics, and aligned textual descriptions. Synthetic data, therefore, becomes a practical solution (Xie et al., 2024; Fu et al., 2024; Tremblay et al., 2018). However, naive generation methods often lack precise control over *temporal dynamics* and *spatial dependency*, leading to weak alignment between data and semantics (Zhang et al., 2024). To address this, we adopt *network stochastic differential equations (Network SDEs)* (Iafrate and Iacus, 2024; Liu et al., 2025f) as the core generative mechanism.

Network SDEs. Network SDEs enable fine-grained control over *temporal dynamics* through node-wise drift and diffusion terms, and explicit modeling of *spatial dependency* through structured coupling over a graph. This makes them well-suited for simulating spatio-temporal processes with controllable dynamics and well-aligned semantics. To model the spatio-temporal time series, we assume that each discrete observation $x_{i,t}$ is sampled from an underlying continuous-time latent process $X_i(t)$ associated with node $v_i \in \mathcal{V}$. The latent dynamics over the graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ are modeled with network SDEs. For each node i, the state $X_i(t)$ evolves as

$$
\begin{aligned}
dX_i(t) &= f_i(X_i(t), t)\, dt + g_i(X_i(t), t)\, dW_i(t) \\
&+ \sum\nolimits_{j \neq i} \lambda_i A_{ji}(t) \big(X_j(t - \tau_{ji}) - X_i(t) \big)\, dt \quad (1)
\end{aligned}
$$

where f_i denotes the node-specific drift function capturing intrinsic temporal dynamics, g_i controls stochastic diffusion, and $W_i(t)$ is a Wiener process. The network coupling term models spatial dependencies induced by $\mathcal{G}$: $A_{ji}(t)$ is a time-varying adjacency weight associated with the directed edge $(v_j, v_i) \in \mathcal{E}$, λ_i controls the coupling strength at node i, and τ_{ji} represents the propagation delay from node j to node i. Discrete observations $x_{i,t}$ in the time series T_i are obtained by sampling $X_i(t)$ at the specified time points.

Network SDEs-based multi-agent spatio-temporal data synthesis pipeline. A key remaining challenge is how to align spatio-temporal dynamics with structured textual descriptions. To address this issue, we propose a network SDEs-based multi-agent data synthesis pipeline, illustrated in Figure 2. The framework consists of six agents. The ***Scenario Generation Agent*** produces high-level textual descriptions of the spatio-temporal system. The ***Scenario Parsing Agent*** converts the text into structured, machine-readable specifications, including nodes, edges, temporal patterns, and spatial dependencies. The ***Judge Agent (Scenario Validation)*** evaluates scenario logic and provides feedback to refine the outputs of the first two agents. Based on the parsed specifications, the ***SDEs Parameters Agent*** instantiates the node-wise drift and diffusion functions $f_i(\cdot)$ and $g_i(\cdot)$, together with coupling strengths λ_i, while the ***Time-Varying Adjacency Agent*** assigns dynamic adjacency weights $A_{ji}(t)$ and propagation delays τ_{ji}. The ***Judge Agent (Parameter Validation)*** assesses whether the simulated data are consistent with the scenario description and provides feedback to refine the SDE and adjacency parameters. Finally, the ***Simulation*** module generates spatio-temporal time series by integrating Equation 1 with the instantiated parameters.

To ensure alignment between text and data, we design the network SDEs to be expressive enough to faithfully realize the scenarios described by the first two agents. Specifically, we distinguish demand source nodes from propagation nodes to

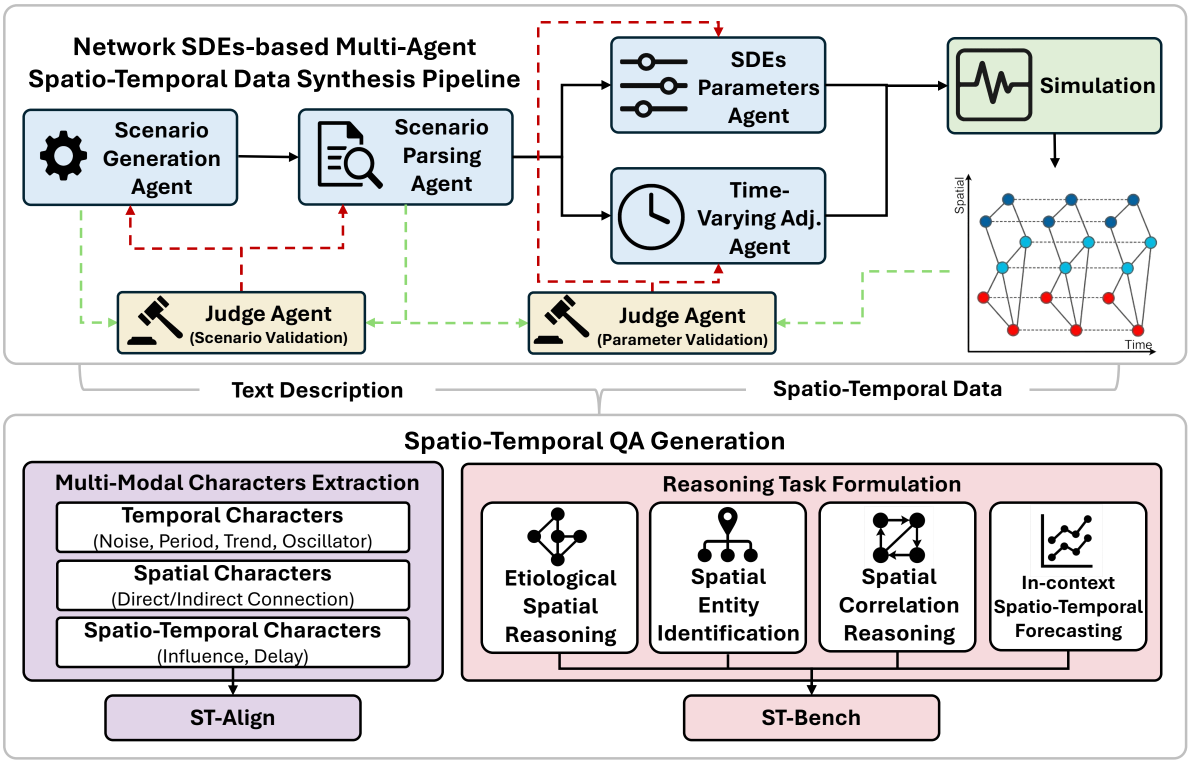


Figure 2: Overall framework of the Network SDEs-based multi-agent spatio-temporal data synthesis pipeline (upper) and spatio-temporal QA generation (lower). Detailed prompts for each agent are provided in Appendix K, and the pseudo algorithm is given in Algorithm 1.

control where self-generated dynamics can occur, restrict exogenous temporal patterns to demand sources, model spatial influence through time-varying adjacency matrices, and introduce explicit time lags to capture delayed propagation. These design choices enable the network SDEs formulation to directly encode the semantic constraints specified in the textual scenario, thereby enforcing consistency between textual descriptions, structured specifications, and simulated spatio-temporal data. As a result, the pipeline yields well-aligned textual descriptions and spatio-temporal data, which we use to generate spatio-temporal QA datasets. More details are provided in Appendix B.1.

Spatio-Temporal QA Generation. As shown in Figure 2, we generate two spatio-temporal QA datasets: *ST-Bench* and *ST-Align*. ST-Bench focuses on training and evaluating complex spatio-temporal reasoning, while ST-Align supports alignment pretraining via basic understanding questions.

ST-Bench is a benchmark designed to assess spatio-temporal reasoning through four tasks: ***T1***:*Etiological Spatial Reasoning* focuses on inferring global system-level semantics from observed spatio-temporal dynamics. ***T2***:*Spatial Entity Identification* examines whether a model can recognize semantic roles of nodes from spatio-temporal observations. ***T3***:*Spatial Correlation Reasoning* evaluates causal reasoning over spatial structures, including direct causal influence and multi-hop propagation. Finally, ***T4***:*In-context Spatio-Temporal Forecasting* assesses the ability to predict future temporal evolution under spatial dependencies. Prompts for QA generation are provided in Appendix I.

ST-Align is designed to align textual representations with spatio-temporal data during training through basic understanding questions: *temporal* questions focusing on node-level temporal dynamics, *spatial* questions probing graph connectivity and influence paths, and *spatio-temporal* questions that jointly reason over propagation delays and time-varying spatial interactions. Specifically, we generate QA pairs using predefined templates (see Appendix J) based on the outputs of the *SDEs Parameters Agent*, the *Time-Varying Adjacency Agent*, and the simulated spatio-temporal data,

3.3 Dataset Splits and Usage

We first synthesize 1,200 spatio-temporal samples using the *Network SDE-based Multi-Agent Spatio-Temporal Data Synthesis Pipeline* illustrated in Fig-

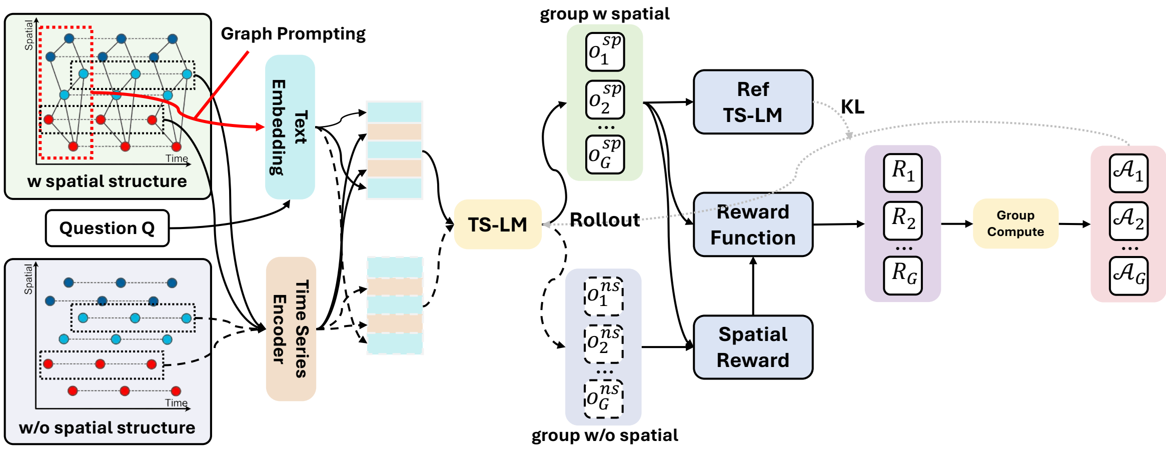


Figure 3: An illustration of our proposed STReasoner with the S-GRPO algorithm.

ure 2. We provide a showcase of the synthesized data sample in Table 8. To ensure data reliability, all synthesized samples are manually inspected, as described in Appendix B.2, resulting in 1,064 retained samples. We split the samples into 80% for training and 20% for testing. We generate **ST-Align** using the training set, which contains 153,700 QA pairs. From the same training set, we further construct **ST-Bench**, and divide it in a 6:2 ratio to form **ST-SFT** and **ST-RL** for SFT and RL, respectively. The testing split is used exclusively to construct **ST-Test** for held-out evaluation on **ST-Bench**. More details can be found in Appendix B.3.

4 Our Solution: STReasoner

4.1 Model Architecture

As shown in Figure 3, STReasoner processes multimodal inputs by separately encoding time series data and textual inputs, including the natural language query and the graph structure prompted in text. Following prior work on time series encoding for LLMs (Zhang et al., 2025c; Chow et al., 2024; Xie et al., 2024; Jin et al., 2023a), we patchify the input time series (Nie, 2022) and encode each patch using a lightweight five-layer MLP (Time Series Encoder). The resulting time series embeddings are interleaved with text tokens according to node order to form a single input sequence for the LLM, thereby casting the model as a TS-LM. Specifically, node-associated time series embeddings are inserted as special placeholders in the textual stream, yielding an input of the form $[\texttt{Node}_1 : \langle \mathrm{TS}_1 \rangle, \texttt{Node}_2 : \langle \mathrm{TS}_2 \rangle, \dots, \texttt{Graph}, \texttt{Question}]$. This representation preserves the contextual information of each time series. We adopt value-preserving normalization following ChatTS (Xie et al., 2024) to normalize time series features while preserving the original numerical information.

4.2 Model Training

Training models for spatio-temporal reasoning in time series presents challenges distinct from those in purely textual or visual domains. First, the newly introduced time series encoder produces embeddings that are not naturally aligned with the text embedding space of pretrained LLMs. Second, pretrained LLMs lack inherent spatio-temporal priors, as they are only minimally exposed to structured time series data during pretraining. Third, models may rely on superficial temporal patterns and fail to consistently ground their reasoning in spatial structure, due to the absence of explicit training signals that encourage spatial reasoning. To address these challenges, we adopt a three-stage training paradigm: (1) large-scale alignment pretraining to bridge textual and time series modalities, (2) injection of spatio-temporal reasoning priors to provide a reliable reasoning cold start, and (3) spatial-aware RL to refine reasoning behavior.

Stage 1: Large-Scale Alignment Pretraining. In the first stage, we perform large-scale alignment pretraining using **ST-Align**. This dataset consists of QA pairs that probe fundamental characteristics of spatio-temporal data, including temporal, spatial, and spatio-temporal attributes. The goal of this stage is to establish an initial alignment between textual representations and time series embeddings within the STReasoner. Leveraging large-scale synthetic data, the model learns to associate textual descriptions with attributes encoded in spatio-temporal time series, providing a foundation for subsequent reasoning-oriented training.

Stage 2: Spatio-Temporal Reasoning Priors Injection. To provide a reliable SFT cold start, we construct chain-of-thought (CoT) annotations via rejection sampling (Ahn et al., 2024; Jin et al., 2025). Specifically, for each question in **ST-SFT**, we sample five candidate responses using an instructed LLM (Claude-4.5-Sonnet) and retain only those that lead to correct final answers. The selected reasoning trajectories are aggregated to form the **ST-CoT** dataset, which is then used to perform SFT. This stage injects explicit spatio-temporal reasoning priors into the model, establishing a strong initialization for subsequent RL.

Stage 3: Reinforcing Reasoning with Spatial-Aware GRPO. While GRPO (Shao et al., 2024) has proven effective in text-based reasoning, it lacks explicit reward for spatial reasoning. To address this, we propose *Spatial-Aware GRPO* (**S-GRPO**), which introduces a contrastive reward mechanism that explicitly encourages spatial reasoning, as illustrated in Figure 3. The core idea behind **S-GRPO** is to compare the model's performance on the same question when spatial structure is provided or not. For each input question, we generate two groups of responses: $\{o_i^{\text{sp}}\}_{i=1}^{G}$ with explicit spatial structure, and $\{o_i^{\text{ns}}\}_{i=1}^{G}$ without spatial structure, where G denotes the number of responses per group, we then define the final reward R_i used for calculating advantages as:

$$R_i = \begin{cases} r_i^{\text{sp}} + \alpha, & \text{if } r_i^{\text{sp}} > \beta\, r_i^{\text{ns}}, \\ r_i^{\text{sp}}, & \text{otherwise.} \end{cases}$$

where r_i^{sp} and r_i^{ns} denote the rewards of responses o_i^{sp} and o_i^{ns}, respectively, containing both the accuracy reward and the format reward (see Appendix C) (Guo et al., 2025), α, β are hyperparameters controlling the magnitude of the spatial reward and tolerance. This contrastive design encourages the model to perform better when the spatial structure is presented than when it is removed. The model is only granted this positive reward if its current reasoning strategy for a given question demonstrates a reliance on spatial information. Then, the advantage $\mathcal{A}_i$ is computed over the rewards within each group: $\mathcal{A}_i = \frac{R_i - \text{mean}(\{R_j\})}{\text{std}(\{R_j\})}$. Following DeepSeek R1 (Guo et al., 2025), the final policy update is as follows:

$$\begin{aligned} &\mathcal{J}_{\text{S-GRPO}}(\theta) = \\ &\mathbb{E}_{q,\{o_i\}}\Bigg[\frac{1}{G}\sum_{i=1}^{G}\Bigg(\min\Bigg(\frac{\pi_\theta(o_i \mid q)}{\pi_{\theta_{\text{old}}}(o_i \mid q)}\mathcal{A}_i, \\ &\text{clip}\left(\frac{\pi_\theta(o_i \mid q)}{\pi_{\theta_{\text{old}}}(o_i \mid q)}, 1-\epsilon, 1+\epsilon\right)\mathcal{A}_i\Bigg) \\ &-\beta\,\mathbb{D}_{\text{KL}}(\pi_\theta \,\|\, \pi_{\text{ref}})\Bigg)\Bigg] \end{aligned} \tag{2}$$

By explicitly comparing the model's performance under inputs with and without spatial structure, **S-GRPO** introduces a contrastive training signal that drives the model to prefer reasoning strategies that leverage spatial patterns.

5 Experiment

Baselines. We compare **STReasoner** with models from four categories. (1) **Proprietary Models**: GPT-5.2 (OpenAI, 2025) and Claude-4.5-Sonnet (Anthropic, 2025). (2) **Open-Source Models**: Qwen3-8B and Qwen3-VL-8B-Instruct (Yang et al., 2025a). (3) **Time Series Language / Reasoning Models (LMs/RMs)**: Time-MQA-7B (Kong et al., 2025), ChatTS-8B (Xie et al., 2024), and Time-R1-7B (Luo et al., 2025). We provide implementation details of STReasoner in Appendix D.

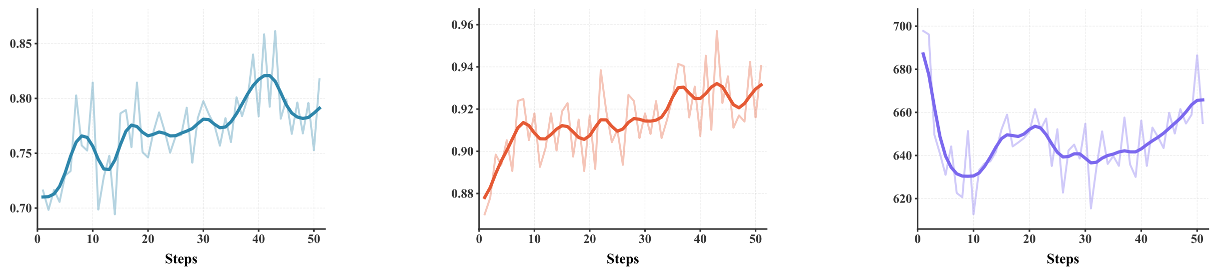
5.1 Main Results

Performance of Existing LLMs. As shown in Table 1, proprietary models demonstrate strong zero-shot performance on all tasks, especially forecasting tasks. However, their large model sizes lead to substantially higher computational costs. In contrast, most lightweight open-source models and time-series LMs/RMs perform significantly worse than proprietary models. Among them, time series RMs consistently outperform time series LMs, indicating that these tasks require explicit reasoning capability. From a cost–performance perspective, prompting time series as images is generally more effective than text, as it achieves better performance while incurring lower token cost.

Superior Performance. **STReasoner** achieves the best overall performance among open-source and time series models across all four tasks. Compared with proprietary models, **STReasoner** achieves these gains at an average cost of **0.004**× that of proprietary models, delivering average improvements of **17%**, **135%**, and **40%** on the T1:

Table 1: Model comparison on four tasks, reporting **ACC** (%) for **T1: Etiological**, **T2: Entity**, and **T3: Correlation** tasks, and **MAE** for **T4: Forecasting**. We also report **Tokens** (M) and **Est. Cost** ($), where token usage refers to input token only, and cost is estimated based on publicly available API pricing. **Prompting** indicates the input modality used by each model. † indicates that STReasoner outperforms the best baseline with a p-value less than 0.05. The best performance is highlighted in **red**, and the second-best is underlined.

Category	Models	Prompting	T1: Etiological		T2: Entity		T3: Correlation		T4: Forecasting		Est. Cost
			ACC	Tokens	ACC	Tokens	ACC	Tokens	MAE	Tokens	
Proprietary Models	GPT-5.2	Text	83.09	0.66	38.78	4.86	58.79	6.84	63.99	0.49	22.48
	GPT-5.2	Image	86.47	0.24	40.54	1.38	65.08	1.89	64.70	0.31	6.69
	Claude-4.5-Sonnet	Text	78.64	0.81	41.93	5.67	77.87	8.05	63.74	0.74	45.80
	Claude-4.5-Sonnet	Image	79.61	0.39	41.85	2.12	76.04	2.84	**63.36**	0.48	17.48
Open-Source Models	Qwen3-8B	Text	21.26	0.98	5.28	7.31	5.53	10.26	94.03	0.72	3.85
	- SFT	Text	82.13	0.98	62.65	7.31	79.84	10.26	66.49	0.72	3.85
	- SFT+S-GRPO	Text	89.37	0.98	65.41	7.31	81.34	10.26	66.35	0.72	3.85
	Qwen3-VL-8B-Instruct	Image	68.60	0.21	31.16	1.20	53.52	1.65	74.21	0.26	0.66
	- SFT	Image	90.82	0.21	61.31	1.20	80.78	1.65	68.53	0.26	0.66
	- SFT+S-GRPO	Image	91.79	0.21	69.43	1.20	83.92	1.65	67.29	0.26	0.66
Time Series LMs	Time-MQA-7B	Text	19.32	0.98	14.24	7.31	17.90	10.26	84.70	0.72	3.85
	ChatTS-8B	TS Encoder	56.52	0.08	19.51	0.50	41.08	0.72	85.14	0.08	0.27
Time Series RMs	Time-R1-7B	Text	60.39	0.98	29.65	7.31	48.62	10.26	68.15	0.72	3.85
Spatio-Temporal RMs	STReasoner-8B (Ours)	TS Encoder	**95.65**†	0.08	**75.71**†	0.50	**87.12**†	0.72	65.59	0.08	**0.27**



(a) Accuracy Reward (b) Spatial Reward (c) Response Length

Figure 4: RL training curves over steps.

Etiological, T2: Entity, and T3: Correlation tasks, respectively, while achieving comparable performance on T4: Forecasting with only **2.6%** difference. These results validate the effectiveness of the time series encoder and the proposed three-stage training strategy for spatio-temporal reasoning.

Study of Time Series Modality. To study the effect of time series modality design and the use of a separate time series encoder, we apply the same training strategy, including SFT and S-GRPO, to models that prompt time series as text or images. Image-based prompting performs well on tasks that rely on global shapes, such as Etiological, Entity, and Correlation reasoning, but underperforms text-based prompting on tasks that require precise numerical information, namely forecasting. Overall, models using only text modality or image-based multimodality perform significantly worse than STReasoner. This result shows that explicitly encoding time series through a dedicated encoder is necessary to preserve both global shape and numerical information for spatio-temporal reasoning.

5.2 Zero-Shot Results on Real-World Data

To evaluate the practical zero-shot performance of **STReasoner** on real-world spatio-temporal data, we construct a causal QA dataset based on CausalRivers (Stein et al., 2025). Specifically, given the underlying causal graph, we define a causal effect from node A to node B if there exists a directed edge from A to B, and generate one QA instance for each such edge by asking whether node A has a causal effect on node B. We evaluate **STReasoner** in a strict zero-shot setting, where no finetuning is performed on this dataset. As shown in Table 2, despite being trained only on synthetic data, **STReasoner** significantly outperforms two large

Table 2: Model comparison on the real-world dataset.

Model	Prompting	ACC	Tokens	Cost
GPT-5.2	Text	22.32	3.49M	$6.15
Claude-4.5-Sonnet	Text	83.18	0.78M	$2.35
STReasoner	TS Encoder	**98.82**	**0.26M**	**$0.05**

Table 3: Ablation results of different training stages.

Models	Etiol. ACC	Entity ACC	Corr. ACC	Fore. MAE
STReasoner `Align+SFT+S-GRPO`	**95.65**	**75.71**	**87.12**	**65.593**
`Align+SFT+GRPO`	91.79	69.60	86.12	69.961
`SFT+S-GRPO`	91.30	67.76	83.98	69.014
`Align+SFT`	88.41	63.32	80.97	66.653
`SFT`	90.34	61.47	81.47	71.096
`S-GRPO`	47.34	23.20	39.20	91.921
`Align`	3.38	8.79	3.77	75.360

proprietary models on this real-world benchmark. These results indicate strong generalization ability and validate the effectiveness of our synthetic data construction and training strategy.

5.3 Ablation Study

Does `Align` help? `Align` aims to align textual instructions with time series representations, providing a foundation for subsequent reasoning training. As shown in Table 3, the model trained with `Align` alone performs poorly across all tasks. This is because the `Align` objectives are not directly optimized for downstream reasoning tasks and therefore do not induce reasoning capability on their own. Comparing `SFT+S-GRPO` with **STReasoner**, the addition of `Align` improves the final performance by an average of **6.3%**. This demonstrates that alignment provides complementary benefits when combined with `SFT` and `S-GRPO`.

Does `SFT` help? `SFT` plays a critical role in establishing instruction-following and basic reasoning ability. Models that skip `SFT` and directly apply `S-GRPO` exhibit significantly worse performance, which is due to sparse and unstable reward signals during rollout under weak reasoning initialization.

Does `S-GRPO` help? After `SFT`, RL becomes effective. Comparing `Align+SFT+GRPO` with `Align+SFT` shows that RL further improves accuracy. Moreover, comparing **STReasoner** with `Align+SFT+GRPO` shows that replacing `GRPO` with the proposed `S-GRPO` yields an average improvement of **5.10%**, demonstrating the effectiveness of spatial-aware RL for spatio-temporal reasoning.

Table 4: Scaling analysis across model sizes.

Model	Etiol. ACC	Entity ACC	Corr. ACC	Fore. MAE
Qwen3-4B	82.53	52.85	76.07	72.323
STReasoner-4B	90.34	64.66	83.29	70.321
Qwen3-8B	88.41	63.32	80.97	66.653
STReasoner-8B	95.65	75.71	87.12	65.593
Qwen3-14B	93.24	62.90	81.16	66.041
STReasoner-14B	96.69	76.60	87.61	65.478

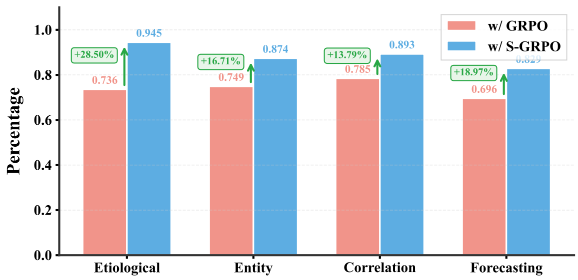


Figure 5: Percentage of spatial reasoning responses.

5.4 Effect of Model Scale

Model scale is a key factor that can influence performance, making it important to distinguish improvements brought by the proposed method from those due to increased model capacity. To ensure a fair comparison, all results in the main experiments are obtained using an 8B backbone. We further conduct scaling experiments using Qwen3 as the backbone at 4B, 8B, and 14B, where all models are trained with `Align+SFT` under the same setting.

As shown in Table 4, STReasoner consistently outperforms the corresponding Qwen3 baseline at all model scales. While increasing the baseline size from 4B to 14B leads to moderate improvements, the gains are limited compared with those achieved by applying the proposed method. Notably, STReasoner-8B already surpasses Qwen3-14B on multiple tasks, suggesting that the proposed method provides stronger benefits than simply scaling up the model.

5.5 Effect of Spatial Reward Analysis

Training Curves of RL. Figure 4 illustrates the RL training dynamics of **STReasoner**. As shown in Figure 4a, the accuracy reward exhibits an overall upward trend, indicating that the model gradually improves its ability to generate correct answers

Table 5: Effect of spatial input (w/ vs. w/o graph) during RL training.

Step	1	11	21	31	41	51
ACC (w/ graph)	82.4	86.9	90.9	93.0	94.2	95.65
ACC (w/o graph)	81.9	83.9	85.7	87.1	88.6	89.4
Δ	0.5	3.0	5.2	5.9	5.6	6.2

during RL training. Similarly, Figure 4b shows a steady increase in the spatial reward, suggesting that the model increasingly relies on spatially grounded reasoning behaviors under **S-GRPO**. Notably, the response length in Figure 4c decreases in the early stage of RL training, followed by a gradual increase and stabilization. This pattern likely reflects a transition in the learned reasoning policy: the model initially departs from sub-optimal SFT reasoning patterns and eventually converges to a more stable and effective reasoning strategy.

Effect of S-GRPO on Spatial Reasoning Usage. To further assess whether **S-GRPO** encourages models to actively rely on spatial structure during reasoning, we measure the percentage of responses that explicitly use spatial information on questions requiring spatial reasoning by GPT-5.2. The evaluation prompt can be found in Appendix L. As shown in Figure 5, **STReasoner** trained with **S-GRPO** consistently exhibits a substantially higher spatial reasoning usage ratio across all tasks compared with vanilla GRPO. These results indicate that **S-GRPO** does not merely improve final accuracy, but more importantly, shifts the model's reasoning behavior toward spatially grounded strategies.

Effect of Spatial Information Usage During RL. To further examine whether **S-GRPO** improves the model's ability to leverage spatial knowledge, we evaluate model checkpoints at different RL training steps under two input settings: w/ and w/o graph structure. Specifically, we report accuracy on Task 1 at steps 1–51 and measure the performance gap Δ between the two settings. As shown in Table 5, overall accuracy improves steadily during RL training. More importantly, the performance gain from spatial input becomes increasingly significant. The gap Δ grows from 0.5 at step 1 to 6.2 at step 51, indicating that the model progressively relies more on spatial structure. This result provides direct evidence that **S-GRPO** enhances the model's ability to utilize spatial information, rather than merely improving overall accuracy.

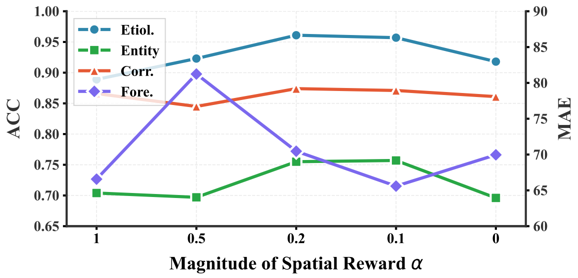


Figure 6: S-GRPO Sensitivity Analysis.

Sensitivity Analysis. Figure 6 presents a sensitivity analysis on the spatial reward magnitude α. Intermediate values ($\alpha = 0.2$ and 0.1) yield consistently strong performance, while extreme settings ($\alpha = 1$, 0.5, and 0) degrade one or more metrics. A large α causes the spatial reward to dominate optimization and bias the model away from answer correctness, whereas removing α fails to encourage spatial-aware reasoning.

5.6 Case Study

We present several case studies for each task in Appendix H. As shown in Table 9, STReasoner first parses the graph structure and, with the help of the time series encoder, accurately identifies key values and their corresponding time steps at each node. By aligning these temporal patterns with the directed topology, the model detects transport lags and attenuation along the propagation path, and finally infers the etiological scenario that best matches the observed spatio-temporal dynamics.

6 Conclusion

Spatio-temporal reasoning in time series is essential for real-world decision-making, but progress is limited by the lack of paired spatio-temporal data and text. We formally define spatio-temporal reasoning and propose an SDE-based data synthesis method that produces diverse spatio-temporal data with precise descriptions. We also build ST-Bench to evaluate multiple tasks. We propose STReasoner, the first TS-LM for complex spatio-temporal reasoning, trained in three stages on synthetic data, and introduce S-GRPO to improve spatial reasoning. Experiments show clear gains, indicating that our method effectively links spatio-temporal data with natural language understanding.

7 Limitations

Due to limited prior work on spatio-temporal reasoning in time series, STReasoner, while effective, still has several limitations. **First**, although our experiments show that carefully designed synthetic data can support effective alignment and spatio-temporal reasoning, broader and more diverse real-world datasets are still essential for fully assessing and improving STReasoner. **Second**, while a simple MLP-based time series encoder is sufficient for the structured signals considered in this work, more expressive architectures and multimodal integration strategies may be required for complex real-world scenarios.

8 Potential Risks

Table 6: Energy consumption and CO_2 emissions by training stages.

Stage	Run Time (h)	GPU-hours	GPU Energy (kWh)	Total Energy (kWh, incl. PUE)	CO_2e (kg)
Align	13.41	107.27	37.54	45.05	15.14
SFT	8.18	65.44	22.90	27.48	9.23
RL	5.38	43.07	15.07	18.09	6.08
Total	26.97	215.77	75.52	90.62	30.45

A potential risk of this work is its environmental impact. Training and deploying large-scale models can incur substantial computational cost and energy consumption, which may contribute to increased carbon emissions. While our approach relies on synthetic data and relatively lightweight components, future extensions to larger models or datasets should carefully consider efficiency and sustainability.

Acknowledgments

Juntong Ni and Wei Jin are supported by the U.S. National Science Foundation under Award Number 2504088, and the National Institute of Allergy and Infectious Diseases of the NIH under Award Numbers R01AI189829 and R01AI197111. The content is the sole responsibility of the authors and does not necessarily represent the views of the NIH. This project is also partially supported by the NAIRR Pilot Program.

References

Janice Ahn, Rishu Verma, Renze Lou, Di Liu, Rui Zhang, and Wenpeng Yin. 2024. Large language models for mathematical reasoning: Progresses and challenges. *arXiv preprint arXiv:2402.00157*.

Anthropic. 2025. Introducing claude sonnet 4.5. https://www.anthropic.com/news/claude-sonnet-4-5.

Maciej Besta, Nils Blach, Ales Kubicek, Robert Gerstenberger, Michal Podstawski, Lukas Gianinazzi, Joanna Gajda, Tomasz Lehmann, Hubert Niewiadomski, Piotr Nyczyk, and 1 others. 2024. Graph of thoughts: Solving elaborate problems with large language models. In *Proceedings of the AAAI conference on artificial intelligence*, volume 38, pages 17682–17690.

Yifu Cai, Arjun Choudhry, Mononito Goswami, and Artur Dubrawski. 2024. Timeseriesexam: A time series understanding exam. *arXiv preprint arXiv:2410.14752*.

Jialin Chen, Aosong Feng, Ziyu Zhao, Juan Garza, Gaukhar Nurbek, Cheng Qin, Ali Maatouk, Leandros Tassiulas, Yifeng Gao, and Rex Ying. 2025. Mtbench: A multimodal time series benchmark for temporal reasoning and question answering. *arXiv preprint arXiv:2503.16858*.

Wei Chen and Yuxuan Liang. 2025. Learning with calibration: Exploring test-time computing of spatio-temporal forecasting. *arXiv preprint arXiv:2506.00635*.

Zixu Cheng, Jian Hu, Ziquan Liu, Chenyang Si, Wei Li, and Shaogang Gong. 2025. V-star: Benchmarking video-llms on video spatio-temporal reasoning. *arXiv preprint arXiv:2503.11495*.

Winnie Chow, Lauren Gardiner, Haraldur T Hallgrímsson, Maxwell A Xu, and Shirley You Ren. 2024. Towards time series reasoning with llms. *arXiv preprint arXiv:2409.11376*.

Zhixuan Chu, Hui Ding, Guang Zeng, Shiyu Wang, and Yiming Li. 2024. Causal interventional prediction system for robust and explainable effect forecasting. In *Proceedings of the 33rd ACM International Conference on Information and Knowledge Management*, pages 4431–4438.

Yuchen Fang, Shiyu Wang, Yuxuan Liang, Zhou Ye, Yang Xiang, Yan Zhao, and Kai Zheng. 2026. Efficient high-dimensional time series forecasting with transformers: A channel reordering perspective. In *Proceedings of the ACM Web Conference 2026*.

Jie Feng, Shengyuan Wang, Tianhui Liu, Yanxin Xi, and Yong Li. 2025a. Urbanllava: A multi-modal large language model for urban intelligence with spatial reasoning and understanding. *arXiv preprint arXiv:2506.23219*.

Jie Feng, Jun Zhang, Tianhui Liu, Xin Zhang, Tianjian Ouyang, Junbo Yan, Yuwei Du, Siqi Guo, and Yong Li. 2025b. Citybench: Evaluating the capabilities of large language models for urban tasks. In *Proceedings of the 31st ACM SIGKDD Conference on Knowledge Discovery and Data Mining V. 2*, pages 5413–5424.

Kaituo Feng, Kaixiong Gong, Bohao Li, Zonghao Guo, Yibing Wang, Tianshuo Peng, Junfei Wu, Xiaoying Zhang, Benyou Wang, and Xiangyu Yue. 2025c. Video-r1: Reinforcing video reasoning in mllms. *arXiv preprint arXiv:2503.21776*.

Fanzhe Fu, Junru Chen, Jing Zhang, Carl Yang, Lvbin Ma, and Yang Yang. 2024. Are synthetic time-series data really not as good as real data? *arXiv preprint arXiv:2402.00607*.

Haotian Gao, Zheng Dong, Jiawei Yong, Shintaro Fukushima, Kenjiro Taura, and Renhe Jiang. 2025. How different from the past? spatio-temporal time series forecasting with self-supervised deviation learning. *arXiv preprint arXiv:2510.04908*.

Parker Glenn, Parag Dakle, Liang Wang, and Preethi Raghavan. 2024. Blendsql: A scalable dialect for unifying hybrid question answering in relational algebra. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 453–466.

Shengbo Gong, Xianfeng Tang, Carl Yang, and 1 others. 2025. Beyond chunks and graphs: Retrieval-augmented generation through triplet-driven thinking. *arXiv preprint arXiv:2508.02435*.

Tong Guan, Zijie Meng, Dianqi Li, Shiyu Wang, Chao-Han Huck Yang, Qingsong Wen, Zuozhu Liu, Sabato Marco Siniscalchi, Ming Jin, and Shirui Pan. 2025. Timeomni-1: Incentivizing complex reasoning with time series in large language models. *arXiv preprint arXiv:2509.24803*.

Yanchu Guan, Dong Wang, Zhixuan Chu, Shiyu Wang, Feiyue Ni, Ruihua Song, and Chenyi Zhuang. 2024. Intelligent agents with llm-based process automation. In *Proceedings of the 30th ACM SIGKDD conference on knowledge discovery and data mining*, pages 5018–5027.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, and 1 others. 2025. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*.

Kethmi Hirushini Hettige, Jiahao Ji, Cheng Long, Shili Xiang, Gao Cong, and Jingyuan Wang. 2025. A modular multitask reasoning framework integrating spatio-temporal models and llms. *arXiv preprint arXiv:2506.20073*.

Congcong Hu, Yuang Shi, Fan Huang, Yang Xiang, Zhou Ye, Ming Jin, and Shiyu Wang. 2026. Eventcast: Hybrid demand forecasting in e-commerce with llm-based event knowledge. *arXiv preprint arXiv:2602.07695*.

Wenxuan Huang, Bohan Jia, Zijie Zhai, Shaosheng Cao, Zheyu Ye, Fei Zhao, Zhe Xu, Yao Hu, and Shaohui Lin. 2025. Vision-r1: Incentivizing reasoning capability in multimodal large language models. *arXiv preprint arXiv:2503.06749*.

Francesco Iafrate and Stefano Iacus. 2024. Ergodic network stochastic differential equations. *arXiv preprint arXiv:2412.17779*.

Yue Jiang, Qin Chao, Yile Chen, Xiucheng Li, Shuai Liu, and Gao Cong. 2024. Urbanllm: Autonomous urban activity planning and management with large language models. *arXiv preprint arXiv:2406.12360*.

Bowen Jin, Hansi Zeng, Zhenrui Yue, Jinsung Yoon, Sercan Arik, Dong Wang, Hamed Zamani, and Jiawei Han. 2025. Search-r1: Training llms to reason and leverage search engines with reinforcement learning. *arXiv preprint arXiv:2503.09516*.

Ming Jin, Shiyu Wang, Lintao Ma, Zhixuan Chu, James Y Zhang, Xiaoming Shi, Pin-Yu Chen, Yuxuan Liang, Yuan-Fang Li, Shirui Pan, and 1 others. 2023a. Time-llm: Time series forecasting by reprogramming large language models. *arXiv preprint arXiv:2310.01728*.

Ming Jin, Qingsong Wen, Yuxuan Liang, Chaoli Zhang, Siqiao Xue, Xue Wang, James Zhang, Yi Wang, Haifeng Chen, Xiaoli Li, and 1 others. 2023b. Large models for time series and spatio-temporal data: A survey and outlook. *arXiv preprint arXiv:2310.10196*.

Takeshi Kojima, Shixiang Shane Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. 2022. Large language models are zero-shot reasoners. *Advances in neural information processing systems*, 35:22199–22213.

Yaxuan Kong, Yiyuan Yang, Yoontae Hwang, Wenjie Du, Stefan Zohren, Zhangyang Wang, Ming Jin, and Qingsong Wen. 2025. Time-mqa: Time series multi-task question answering with context enhancement. *arXiv preprint arXiv:2503.01875*.

Patrick Langer, Thomas Kaar, Max Rosenblattl, Maxwell A Xu, Winnie Chow, Martin Maritsch, Aradhana Verma, Brian Han, Daniel Seung Kim, Henry Chubb, and 1 others. 2025. Opentslm: Time-series language models for reasoning over multivariate medical text-and time-series data. *arXiv preprint arXiv:2510.02410*.

Max SY Lau, C Jessica E Metcalf, Zewen Liu, Bryan T Grenfell, and Wei Jin. 2026. Toward ai foundation models for epidemics: Promise, challenges, and paths forward. *Proceedings of the National Academy of Sciences*, 123(13):e2526192123.

Yongjia Lei, Haoyu Han, Ryan A Rossi, Franck Dernoncourt, Nedim Lipka, Mahantesh M Halappanavar, Jiliang Tang, and Yu Wang. 2025. Mixture of structural-and-textual retrieval over text-rich graph knowledge bases. In *Findings of the Association for Computational Linguistics: ACL 2025*, pages 18306–18321.

Bo Li, Yuanhan Zhang, Dong Guo, Renrui Zhang, Feng Li, Hao Zhang, Kaichen Zhang, Peiyuan Zhang, Yanwei Li, Ziwei Liu, and 1 others. 2024a. Llava-onevision: Easy visual task transfer. *arXiv preprint arXiv:2408.03326*.

Wenbin Li, Di Yao, Ruibo Zhao, Wenjie Chen, Zijie Xu, Chengxue Luo, Chang Gong, Quanliang Jing, Haining Tan, and Jingping Bi. 2025. Stbench: Assessing the ability of large language models in spatio-temporal analysis. In *Companion Proceedings of the ACM on Web Conference 2025*, pages 749–752.

Zhonghang Li, Lianghao Xia, Jiabin Tang, Yong Xu, Lei Shi, Long Xia, Dawei Yin, and Chao Huang. 2024b. Urbangpt: Spatio-temporal large language models. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 5351–5362.

Zhonghang Li, Lianghao Xia, Yong Xu, and Chao Huang. 2023. Gpt-st: generative pre-training of spatio-temporal graph neural networks. *Advances in neural information processing systems*, 36:70229–70246.

Yuxuan Liang, Haomin Wen, Yuqi Nie, Yushan Jiang, Ming Jin, Dongjin Song, Shirui Pan, and Qingsong Wen. 2024. Foundation models for time series analysis: A tutorial and survey. In *Proceedings of the 30th ACM SIGKDD conference on knowledge discovery and data mining*, pages 6555–6565.

Yuxuan Liang, Haomin Wen, Yutong Xia, Ming Jin, Bin Yang, Flora Salim, Qingsong Wen, Shirui Pan, and Gao Cong. 2025. Foundation models for spatio-temporal data science: A tutorial and survey. In *Proceedings of the 31st ACM SIGKDD Conference on Knowledge Discovery and Data Mining V. 2*, pages 6063–6073.

Yuxuan Liang, Yutong Xia, Songyu Ke, Yiwei Wang, Qingsong Wen, Junbo Zhang, Yu Zheng, and Roger Zimmermann. 2023. Airformer: Predicting nationwide air quality in china with transformers. In *Proceedings of the AAAI conference on artificial intelligence*, volume 37, pages 14329–14337.

Hunter Lightman, Vineet Kosaraju, Yuri Burda, Harrison Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. 2023. Let's verify step by step. In *The Twelfth International Conference on Learning Representations*.

Yijun Lin, Nikhit Mago, Yu Gao, Yaguang Li, Yao-Yi Chiang, Cyrus Shahabi, and José Luis Ambite. 2018. Exploiting spatiotemporal patterns for accurate air quality forecasting using deep learning. In *Proceedings of the 26th ACM SIGSPATIAL international conference on advances in geographic information systems*, pages 359–368.

Chenxi Liu, Kethmi Hirushini Hettige, Qianxiong Xu, Cheng Long, Shili Xiang, Gao Cong, Ziyue Li, and Rui Zhao. 2025a. St-llm+: Graph enhanced spatio-temporal large language models for traffic prediction. *IEEE Transactions on Knowledge and Data Engineering*.

Chenxi Liu, Sun Yang, Qianxiong Xu, Zhishuai Li, Cheng Long, Ziyue Li, and Rui Zhao. 2024a. Spatial-temporal large language model for traffic prediction. In *2024 25th IEEE International Conference on Mobile Data Management (MDM)*, pages 31–40. IEEE.

Haoxin Liu, Chenghao Liu, and B Aditya Prakash. 2025b. A picture is worth a thousand numbers: Enabling llms reason about time series via visualization. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 7486–7518.

Haoxin Liu, Shangqing Xu, Zhiyuan Zhao, Lingkai Kong, Harshavardhan Prabhakar Kamarthi, Aditya Sasanur, Megha Sharma, Jiaming Cui, Qingsong Wen, Chao Zhang, and 1 others. 2024b. Time-mmd: Multi-domain multimodal dataset for time series analysis. *Advances in Neural Information Processing Systems*, 37:77888–77933.

Lei Liu, Shuo Yu, Runze Wang, Zhenxun Ma, and Yanming Shen. 2024c. How can large language models understand spatial-temporal data? *arXiv preprint arXiv:2401.14192*.

Xu Liu, Yutong Xia, Yuxuan Liang, Junfeng Hu, Yiwei Wang, Lei Bai, Chao Huang, Zhenguang Liu, Bryan Hooi, and Roger Zimmermann. 2023. Largest: A benchmark dataset for large-scale traffic forecasting. *Advances in Neural Information Processing Systems*, 36:75354–75371.

Yezi Liu, William Youngwoo Chung, Hanning Chen, Calvin Yeung, and Mohsen Imani. 2025c. Are hypervectors enough? single-call llm reasoning over knowledge graphs. *arXiv preprint arXiv:2512.09369*.

Yong Liu, Xingjian Su, Shiyu Wang, Haoran Zhang, Haixuan Liu, Yuxuan Wang, Zhou Ye, Yang Xiang, Jianmin Wang, and Mingsheng Long. 2026. Timer-s1: A billion-scale time series foundation model with serial scaling. *arXiv preprint arXiv:2603.04791*.

Zewen Liu, Juntong Ni, Max SY Lau, and Wei Jin. 2025d. Cape: Covariate-adjusted pre-training for epidemic time series forecasting. *arXiv preprint arXiv:2502.03393*.

Zewen Liu, Juntong Ni, Xianfeng Tang, Max SY Lau, Wenpeng Yin, and Wei Jin. 2025e. Can large language models adequately perform symbolic reasoning over time series? *arXiv preprint arXiv:2508.03963*.

Zewen Liu, Guancheng Wan, B Aditya Prakash, Max SY Lau, and Wei Jin. 2024d. A review of graph neural networks in epidemic modeling. In *Proceedings of the 30th ACM SIGKDD conference on knowledge discovery and data mining*, pages 6577–6587.

Zewen Liu, Xiaoda Wang, Bohan Wang, Zijie Huang, Carl Yang, and Wei Jin. 2025f. Graph odes and beyond: A comprehensive survey on integrating differential equations with graph neural networks. In *Proceedings of the 31st ACM SIGKDD Conference on Knowledge Discovery and Data Mining V. 2*, pages 6118–6128.

Zijia Liu, Peixuan Han, Haofei Yu, Haoru Li, and Jiaxuan You. 2025g. Time-r1: Towards comprehensive temporal reasoning in llms. *arXiv preprint arXiv:2505.13508*.

Yucong Luo, Yitong Zhou, Mingyue Cheng, Jiahao Wang, Daoyu Wang, Tingyue Pan, and Jintao Zhang. 2025. Time series forecasting as reasoning: A slow-thinking approach with reinforced llms. *arXiv preprint arXiv:2506.10630*.

Mike A Merrill, Mingtian Tan, Vinayak Gupta, Thomas Hartvigsen, and Tim Althoff. 2024. Language models still struggle to zero-shot reason about time series. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 3512–3533.

Juntong Ni, Zewen Liu, Shiyu Wang, Ming Jin, and Wei Jin. 2025a. Timedistill: Efficient long-term time series forecasting with mlp via cross-architecture distillation. *arXiv preprint arXiv:2502.15016*.

Juntong Ni, Shiyu Wang, Zewen Liu, Xiaoming Shi, Xinyue Zhong, Zhou Ye, and Wei Jin. 2025b. U-cast: Learning hierarchical structures for high-dimensional time series forecasting. *arXiv preprint arXiv:2507.15119*.

Y Nie. 2022. A time series is worth 64words: Long-term forecasting with transformers. *arXiv preprint arXiv:2211.14730*.

Kanghui Ning, Zijie Pan, Yushan Jiang, Anderson Schneider, Yuriy Nevmyvaka, and Dongjin Song. 2025. Towards interpretable and trustworthy time series reasoning: A bluesky vision. *arXiv preprint arXiv:2510.16980*.

Shuai Niu, Jing Ma, Hongzhan Lin, Liang Bai, Zhihua Wang, Richard Yi Da Xu, Guo Li, Xian Yang, and 1 others. 2025. Promedts: A self-supervised, prompt-guided multimodal approach for integrating medical text and time series. In *Findings of the Association for Computational Linguistics: ACL 2025*, pages 5915–5928.

OpenAI. 2025. Gpt-5 system card. Technical report. Technical report.

Mathaios Panteli and Pierluigi Mancarella. 2015. Influence of extreme weather and climate change on the resilience of power systems: Impacts and possible mitigation strategies. *Electric Power Systems Research*, 127:259–270.

Chris Paola, Efi Foufoula-Georgiou, William E Dietrich, Miki Hondzo, David Mohrig, Gary Parker, Mary E Power, Ignacio Rodriguez-Iturbe, Vaughan Voller, and Peter Wilcock. 2006. Toward a unified science of the earth's surface: Opportunities for synthesis among hydrology, geomorphology, geochemistry, and ecology. *Water resources research*, 42(3).

Pengrui Quan, Brian Wang, Kang Yang, Liying Han, and Mani Srivastava. 2025. Benchmarking spatiotemporal reasoning in llms and reasoning models: Capabilities and challenges. *arXiv preprint arXiv:2505.11618*.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, YK Li, Yang Wu, and 1 others. 2024. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*.

Guangming Sheng, Chi Zhang, Zilingfeng Ye, Xibin Wu, Wang Zhang, Ru Zhang, Yanghua Peng, Haibin Lin, and Chuan Wu. 2025. Hybridflow: A flexible and efficient rlhf framework. In *Proceedings of the Twentieth European Conference on Computer Systems*, pages 1279–1297.

Xiaoming Shi, Shiyu Wang, Yuqi Nie, Dianqi Li, Zhou Ye, Qingsong Wen, and Ming Jin. 2024. Time-moe: Billion-scale time series foundation models with mixture of experts. *arXiv preprint arXiv:2409.16040*.

Gideon Stein, Maha Shadaydeh, Jan Blunk, Niklas Penzel, and Joachim Denzler. 2025. Causalrivers–scaling up benchmarking of causal discovery for real-world time-series. *arXiv preprint arXiv:2503.17452*.

Mingtian Tan, Mike A Merrill, Zack Gottesman, Tim Althoff, David Evans, and Tom Hartvigsen. 2025. Inferring events from time series using language models. *arXiv preprint arXiv:2503.14190*.

Jonathan Tremblay, Aayush Prakash, David Acuna, Mark Brophy, Varun Jampani, Cem Anil, Thang To, Eric Cameracci, Shaad Boochoon, and Stan Birchfield. 2018. Training deep networks with synthetic data: Bridging the reality gap by domain randomization. In *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, pages 969–977.

Chengsen Wang, Qi Qi, Jingyu Wang, Haifeng Sun, Zirui Zhuang, Jinming Wu, Lei Zhang, and Jianxin Liao. 2025a. Chattime: A unified multimodal time series foundation model bridging numerical and textual data. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pages 12694–12702.

Jiahao Wang, Mingyue Cheng, and Qi Liu. 2025b. Can slow-thinking llms reason over time? empirical studies in time series forecasting. *arXiv preprint arXiv:2505.24511*.

Liang Wang, Ivano Lauriola, and Alessandro Moschitti. 2023a. Accurate training of web-based question answering systems with feedback from ranked users. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 5: Industry Track)*, pages 660–667.

Senzhang Wang, Jiannong Cao, and S Yu Philip. 2020. Deep learning for spatio-temporal data mining: A survey. *IEEE transactions on knowledge and data engineering*, 34(8):3681–3700.

Shiyu Wang. 2024. Neuralreconciler for hierarchical time series forecasting. In *Proceedings of the 17th ACM International Conference on Web Search and Data Mining*, pages 731–739.

Shiyu Wang, Zhixuan Chu, Yinbo Sun, Yu Liu, Yuliang Guo, Yang Chen, Huiyang Jian, Lintao Ma, Xingyu Lu, and Jun Zhou. 2024a. Multiscale representation enhanced temporal flow fusion model for long-term workload forecasting. In *Proceedings of the 33rd ACM International Conference on Information and Knowledge Management*, pages 4948–4956.

Shiyu Wang, Jiawei Li, Xiaoming Shi, Zhou Ye, Baichuan Mo, Wenze Lin, Shengtong Ju, Zhixuan Chu, and Ming Jin. 2024b. Timemixer++: A general time series pattern machine for universal predictive analysis. *arXiv preprint arXiv:2410.16032*.

Shiyu Wang, Wei Lu, Jiawei Li, Xiaoming Shi, Xinyue Zhong, Zhou Ye, Ming Jin, and Qingsong Wen. 2025c. Frt: Flow-based reconcile transformer for hierarchical time series. In *Proceedings of the 31st ACM SIGKDD Conference on Knowledge Discovery and Data Mining V. 2*, pages 4995–5006.

Shiyu Wang, Yinbo Sun, Xiaoming Shi, Shiyi Zhu, Lin-Tao Ma, James Zhang, Yifei Zheng, and Jian Liu. 2023b. Full scaling automation for sustainable development of green data centers. *arXiv preprint arXiv:2305.00706*.

Shiyu Wang, Yinbo Sun, Yan Wang, Fan Zhou, Lin-Tao Ma, James Zhang, and YangFei Zheng. 2023c. Flow-based end-to-end model for hierarchical time series forecasting via trainable attentive-reconciliation. In *International Conference on Database Systems for Advanced Applications*, pages 167–176. Springer.

Shiyu Wang, Haixu Wu, Xiaoming Shi, Tengge Hu, Huakun Luo, Lintao Ma, James Y Zhang, and JUN ZHOU. 2024c. Timemixer: Decomposable multiscale mixing for time series forecasting. In *International Conference on Learning Representations (ICLR)*.

Shiyu Wang, Fan Zhou, Yinbo Sun, Lintao Ma, James Zhang, and Yangfei Zheng. 2022. End-to-end modeling of hierarchical time series using autoregressive transformer and conditional normalizing flow-based reconciliation. In *2022 IEEE International Conference on Data Mining Workshops (ICDMW)*, pages 1087–1094. IEEE.

Xiaoda Wang, Ching Chang, Defu Cao, Kaiqiao Han, Fang Sun, Yue Huang, Minxiao Wang, Chang Xu, Xiao Luo, Runze Yan, and 1 others. 2026. Position: Beyond prediction: Toward verifiable physiological waveform reasoning with foundation models and agentic llms. *Authorea Preprints*.

Ye Wang, Ziheng Wang, Boshen Xu, Yang Du, Kejun Lin, Zihan Xiao, Zihao Yue, Jianzhong Ju, Liang Zhang, Dingyi Yang, and 1 others. 2025d. Time-r1: Post-training large vision language model for temporal video grounding. *arXiv preprint arXiv:2503.13377*.

Yilin Wang, Peixuan Lei, Jie Song, Yuzhe Hao, Tao Chen, Yuxuan Zhang, Lei Jia, Yuanxiang Li, and Zhongyu Wei. 2025e. Itformer: Bridging time series and natural language for multi-modal qa with large-scale multitask dataset. *arXiv preprint arXiv:2506.20093*.

Yuxuan Wang, Haixu Wu, Yuezhou Ma, Yuchen Fang, Ziyi Zhang, Yong Liu, Shiyu Wang, Zhou Ye, Yang Xiang, Jianmin Wang, and 1 others. 2025f. Accuracy law for the future of deep time series forecasting. *arXiv preprint arXiv:2510.02729*.

Zihan Wang, Kangrui Wang, Qineng Wang, Pingyue Zhang, Linjie Li, Zhengyuan Yang, Xing Jin, Kefan Yu, Minh Nhat Nguyen, Licheng Liu, and 1 others. 2025g. Ragen: Understanding self-evolution in llm agents via multi-turn reinforcement learning. *arXiv preprint arXiv:2504.20073*.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, and 1 others. 2022. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837.

Zonghan Wu, Shirui Pan, Guodong Long, Jing Jiang, and Chengqi Zhang. 2019. Graph wavenet for deep spatial-temporal graph modeling. *arXiv preprint arXiv:1906.00121*.

Tian Xie, Zitian Gao, Qingnan Ren, Haoming Luo, Yuqian Hong, Bryan Dai, Joey Zhou, Kai Qiu, Zhirong Wu, and Chong Luo. 2025. Logic-rl: Unleashing llm reasoning with rule-based reinforcement learning. *arXiv preprint arXiv:2502.14768*.

Zhe Xie, Zeyan Li, Xiao He, Longlong Xu, Xidao Wen, Tieying Zhang, Jianjun Chen, Rui Shi, and Dan Pei. 2024. Chatts: Aligning time series with llms via synthetic data for enhanced understanding and reasoning. *arXiv preprint arXiv:2412.03104*.

Gelei Xu, Xueyang Li, Yixiong Chen, Yuying Duan, Shuqing Wu, Alexander Yu, Ching-Hao Chiu, Juntong Ni, Ningzhi Tang, Toby Jia-Jun Li, and 1 others. 2025. A comprehensive survey of agentic ai in healthcare. *Authorea Preprints*.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, and 1 others. 2025a. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*.

Ning Yang, Hengyu Zhong, Haijun Zhang, and Randall Berry. 2025b. Vision-llms for spatiotemporal traffic forecasting. *arXiv preprint arXiv:2510.11282*.

Yiyuan Yang, Zichuan Liu, Lei Song, Kai Ying, Zhiguang Wang, Tom Bamford, Svitlana Vyetrenko, Jiang Bian, and Qingsong Wen. 2025c. Time-ra: Towards time series reasoning for anomaly with llm feedback. *arXiv preprint arXiv:2507.15066*.

Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Tom Griffiths, Yuan Cao, and Karthik Narasimhan. 2023. Tree of thoughts: Deliberate problem solving with large language models. *Advances in neural information processing systems*, 36:11809–11822.

Bing Yu, Haoteng Yin, and Zhanxing Zhu. 2017. Spatio-temporal graph convolutional networks: A deep learning framework for traffic forecasting. *arXiv preprint arXiv:1709.04875*.

Yuan Yuan, Jingtao Ding, Jie Feng, Depeng Jin, and Yong Li. 2024. Unist: A prompt-empowered universal model for urban spatio-temporal prediction. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 4095–4106.

Haochuan Zhang, Chunhua Yang, Jie Han, Liyang Qin, and Xiaoli Wang. 2025a. Tempogpt: Enhancing time series reasoning via quantizing embedding. *arXiv preprint arXiv:2501.07335*.

Junru Zhang, Lang Feng, Xu Guo, Yuhan Wu, Yabo Dong, and Duanqing Xu. 2025b. Timemaster: Training time-series multimodal llms to reason via reinforcement learning. *arXiv preprint arXiv:2506.13705*.

Juyuan Zhang, Jiechao Gao, Wenwen Ouyang, Wei Zhu, and Hui Yi Leong. 2025c. Time-llama: Adapting large language models for time series modeling via dynamic low-rank adaptation. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 4: Student Research Workshop)*, pages 1145–1157.

Qianru Zhang, Haixin Wang, Cheng Long, Liangcai Su, Xingwei He, Jianlong Chang, Tailin Wu, Hongzhi Yin, Siu-Ming Yiu, Qi Tian, and 1 others. 2024. A survey of generative techniques for spatial-temporal data mining. *arXiv preprint arXiv:2405.09592*.

Weichen Zhang, Zile Zhou, Zhiheng Zheng, Chen Gao, Jinqiang Cui, Yong Li, Xinlei Chen, and Xiao-Ping Zhang. 2025d. Open3dvqa: A benchmark for comprehensive spatial reasoning with multimodal large language model in open space. *arXiv preprint arXiv:2503.11094*.

Yaowei Zheng, Junting Lu, Shenzhi Wang, Zhangchi Feng, Dongdong Kuang, and Yuwen Xiong. 2025. Easyr1: An efficient, scalable, multi-modality rl training framework.

Yaowei Zheng, Richong Zhang, Junhao Zhang, Yanhan Ye, Zheyan Luo, Zhangchi Feng, and Yongqiang Ma. 2024. Llamafactory: Unified efficient fine-tuning of 100+ language models. *arXiv preprint arXiv:2403.13372*.

Luca Zhou, Pratham Yashwante, Marshall Fisher, Alessio Sampieri, Zihao Zhou, Fabio Galasso, and Rose Yu. 2025. Cats-bench: Can language models describe numeric time series? *arXiv preprint arXiv:2509.20823*.

A Related Work

A.1 LMs Reasoning

Improving LMs' ability to perform complex and multi-step reasoning has become a major focus of recent research (Wei et al., 2022; Kojima et al., 2022; Yao et al., 2023; Lightman et al., 2023; Besta et al., 2024; Gong et al., 2025). DeepSeek-R1 (Guo et al., 2025) shows that large-scale RL with result-only rewards can induce self-emergent reasoning behaviors in LMs, and several subsequent studies have attempted to extend this paradigm to other modalities and tasks (Jin et al., 2025; Huang et al., 2025; Xie et al., 2025; Wang et al., 2025g; Feng et al., 2025c; Liu et al., 2025g; Wang et al., 2026; Lei et al., 2025; Liu et al., 2025c; Glenn et al., 2024; Wang et al., 2023a; Guan et al., 2024). Despite these advances, LM-based reasoning for spatio-temporal time series remains underexplored.

A.2 Time Series Reasoning

Time series language models leverage the implicit knowledge within LMs and contextual textual information to support downstream tasks and simple time series QA, typically without explicit reasoning (Jin et al., 2023a; Liu et al., 2024b; Xie et al., 2024; Chow et al., 2024; Cai et al., 2024; Merrill et al., 2024; Liang et al., 2024; Kong et al., 2025; Yang et al., 2025c; Zhang et al., 2025a; Wang et al., 2025a; Chen et al., 2025; Wang et al., 2025e; Langer et al., 2025; Zhou et al., 2025; Zhang et al., 2025c; Niu et al., 2025; Liu et al., 2025b,e; Hu et al., 2026; Chu et al., 2024; Wang et al., 2023b). In contrast, time series reasoning models aim to develop genuine reasoning capabilities rather than pattern matching, often employing RL to encourage explicit multi-step reasoning and to produce interpretable, step-by-step explanations (Wang et al., 2025b; Tan et al., 2025; Luo et al., 2025; Guan et al., 2025; Zhang et al., 2025b; Ning et al., 2025). However, existing studies in this line of research mainly focus on univariate or multivariate time series and largely overlook spatio-temporal reasoning over spatially structured time series data.

A.3 Spatio-Temporal Reasoning

Spatio-temporal reasoning has been studied in data modalities such as videos (Li et al., 2024a; Zhang et al., 2025d; Feng et al., 2025c; Cheng et al., 2025; Wang et al., 2025d), sensor-driven or event-based settings (Quan et al., 2025), and trajectories (Li et al., 2025; Feng et al., 2025b,a). In contrast, spatio-temporal reasoning in time series involves reasoning over *numerical signals associated with discrete entities*, where spatial relations are explicitly defined by a graph and temporal dependencies arise through time-lagged interactions, rather than spatial relationships being implicitly encoded in visual layouts or physical coordinates.

The closest prior work, STReason (Hettige et al., 2025), introduces a modular, training-free framework that decomposes spatio-temporal queries into programs executed by external tools, but it relies on textual encoding of time series, does not explicitly model or optimize graph structure, and mainly targets tasks such as anomaly detection or prediction that require limited reasoning. In contrast, our work trains a unified model that directly integrates time series, graph structure, and text, introduces spatial-aware reinforcement learning to enforce genuine spatio-temporal reasoning, and proposes ST-Bench to systematically evaluate spatio-temporal reasoning across multiple complementary tasks.

A.4 Spatio-Temporal Forecasting.

A large body of work studies spatio-temporal modeling for numeric prediction in general domains (Chen and Liang, 2025; Gao et al., 2025; Wang et al., 2022; Liu et al., 2024b, 2026; Wang et al., 2025f) or specific application domains, such as traffic (Yu et al., 2017; Wu et al., 2019), air quality (Liang et al., 2023), and epidemiology (Liu et al., 2025d; Lau et al., 2026). The key difference between spatio-temporal forecasting and multivariate time series forecasting (Ni et al., 2025b,a; Wang et al., 2023c; Fang et al., 2026; Shi et al., 2024) is the explicitly usage of the known graph structure. These spatio-temporal forecasting models are carefully designed to capture spatial dependencies and temporal dynamics for forecasting accuracy, but they operate purely on numerical inputs and outputs. As a result, they do not produce interpretable, long-form answers or support multi-step reasoning expressed in natural language. They cannot function as meaningful baselines for evaluating language-based, multi-task spatio-temporal reasoning.

A.5 Spatio-Temporal Language Models.

Several recent works integrate LMs with spatio-temporal data (Liang et al., 2025), including GPT-ST (Li et al., 2023), UrbanGPT (Li et al., 2024b), UrbanLLM (Jiang et al., 2024), UniST (Yuan et al., 2024), STG-LLM (Liu et al., 2024c), ST-LLM (Liu et al., 2024a), ST-LLM+ (Liu et al., 2025a) and ST-Vision-LLM (Yang et al., 2025b). Although these methods incorporate language models, their primary objective remains numeric forecasting or short-form question answering. They generally lack explicit mechanisms for structured, multi-step reasoning and do not generate coherent, long-form explanations grounded in graph structure and temporal dynamics. The closest prior work, Urban-LLM (Jiang et al., 2024), targets urban question answering, but its responses are typically brief and surface-level, without sustained reasoning chains. Therefore, these models are not directly comparable to our setting, which focuses on unified, language-based spatio-temporal reasoning across diverse tasks.

B Dataset Details

B.1 Spatio-Temporal Data and Textual Description Pair Generation

As shown in Figure 2, we first use the proposed network SDE–based multi-agent spatio-temporal data synthesis pipeline (detailed in Algorithm 1) to generate aligned spatio-temporal data (that is, time series and graphs) together with aligned textual descriptions. The prompt for each agent is provided in Appendix K. We use Claude-4.5-Sonnet (Anthropic, 2025) as the default LLM agent. In total, we generate **1200 pairs** of spatio-temporal data and textual descriptions. Specifically, we generate spatio-temporal data with **3, 5, and 10 nodes**, covering **10 key domains**: “Transportation,” “Energy,” “Environment&Pollution,” “Ecology,” “Public Health,” “Hydrology,” “Oceanography,” “Agriculture,” “Mobility,” and “Climate.”

To better capture realistic spatio-temporal dynamics, our data synthesis pipeline incorporates three key design choices:

1. Demand Source and Propagation Nodes. We divide nodes into demand source nodes and propagation nodes. Concretely, we force the drift function f_i of demand source nodes to be sinusoidal or mean-reverting, and assign only the mean-reverting drift function f_i to propagation nodes. The drift function of demand source nodes could be time-varying. We also tried more diverse assignments of drift functions, but the results were not good. With this design, trends from demand source nodes propagate through the graph to other propagation nodes, while propagation nodes themselves only have mean-reverting trends and therefore are mainly influenced by their neighbors. This setup simulates real-world scenarios. For example, in a traffic setting with three nodes representing a residential area, a highway, and a commercial area, the residential and commercial areas are demand source nodes because traffic naturally originates there (for instance, people driving out of garages), while the highway is a propagation node whose traffic mainly comes from other regions and does not originate locally.

2. Time-Varying Adjacency Matrix. To simulate different spatial dependencies at different times, we set the graph adjacency matrix to be time-varying. Using the same example, in the morning traffic should mainly flow from residential areas to roads and then to commercial areas, while in the evening the direction is reversed. Therefore, the edge weights in the adjacency matrix should change over time, such as increasing the weight of residential-to-road edges in the morning and decreasing them at night, and doing the opposite for road-to-residential edges.

3. Propagation Time Lags. In addition, we introduce time lags to simulate propagation delays, since interactions between nodes are not instantaneous. Each edge is assigned a time lag to reflect the speed of propagation.

Through these three components, we expect our pipeline to simulate spatio-temporal data with dynamics that are close to those observed in real-world systems.

B.2 Spatio-Temporal Data and Textual Description Pair Validation

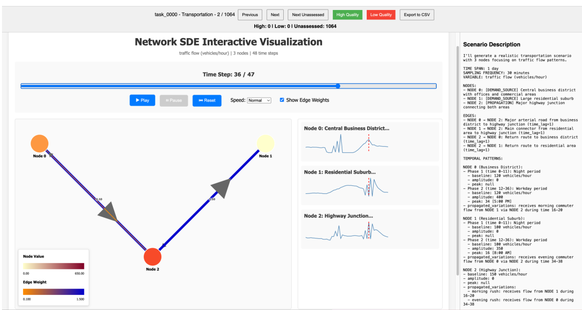


Figure 7: Screenshot of the human evaluation interface.

To validate the usefulness of the generated data pairs and assess whether the textual descriptions are accurately aligned with the corresponding spatio-temporal data, we build a human evaluation interface, shown in Figure 7. We manually inspect each pair by jointly examining the spatio-temporal data and its textual description, and then assign a quality label based on alignment correctness and overall coherence.

In total, 1064 spatio-temporal data and textual description pairs are labeled as "High Quality," while 136 pairs are labeled as "Low Quality," indicating that the majority of generated samples meet the desired quality criteria. We retain only the 1064 samples labeled as "High Quality."

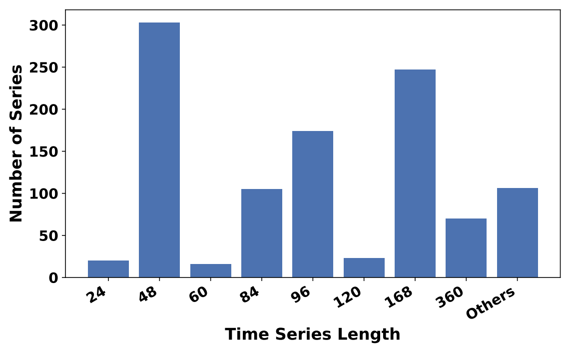


Figure 8: Distribution of Time Series Lengths.

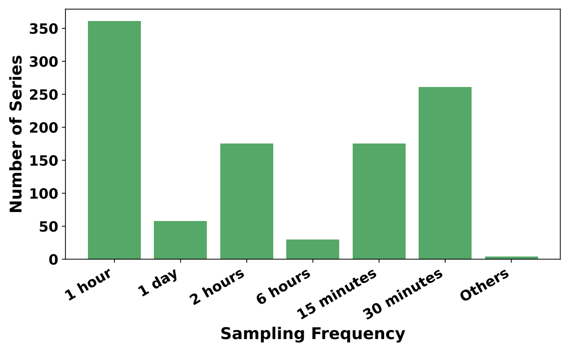


Figure 9: Distribution of Sampling Frequencies.

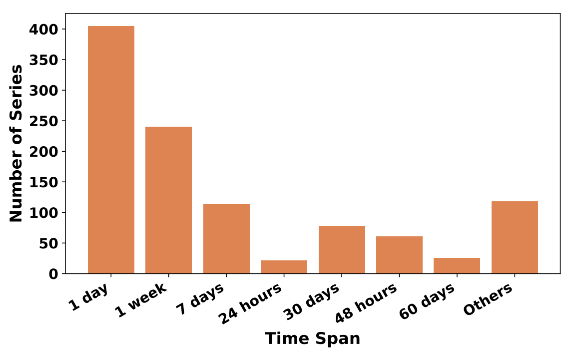


Figure 10: Distribution of Time Spans.

We further summarize the statistical characteristics of the selected 1064 samples in Figures 8–10.

Table 7: Data statistics of ST-Bench for each task.

Task	ST-CoT	ST-RL	ST-Test	Total
T1:Etiological	1,576	405	207	2,188
T2:Entity	4,974	2,451	1,194	8,619
T3:Correlation	11,124	3,231	1,592	15,947
T4:Forecasting	650	506	280	1,436
Total	18,324	6,593	3,273	28,190

Figure 8 shows the distribution of time series lengths, covering both short sequences (tens of steps) and long sequences spanning thousands of steps. Figure 9 illustrates the sampling frequency distribution, which ranges from high-frequency measurements (15–30 minutes) to low-frequency observations (daily). Figure 10 reports the temporal coverage of the data, including samples that span from hours and days to multiple months. In addition, the distributions of 10 domains and the number of nodes per graph (3,5,10 nodes) are approximately uniform, avoiding over-concentration on specific scenarios or graph sizes. Overall, these statistics indicate that the dataset exhibits broad coverage and high diversity along multiple dimensions, providing a challenging and representative testbed for spatio-temporal reasoning.

B.3 QA Dataset Generation

Based on the selected high-quality 1064 spatio-temporal data and aligned textual description pairs, we use Claude-4.5-Sonnet to generate two types of QA datasets: reasoning QA (ST-Bench) and alignment QA (ST-Align).

ST-Bench [1] is designed for Stage 2 and Stage 3 training and for evaluating the model's spatio-temporal reasoning ability. We split ST-Bench into three subsets with a 6:2:2 ratio: ST-CoT for supervised fine-tuning (SFT), ST-RL for reinforcement learning (RL), and ST-Test for evaluation. The prompt templates used to generate these QA pairs are provided in Appendix I. Dataset statistics for each task in ST-Bench are reported in Table 7.

ST-Align is used in Stage 1 alignment training to align time series embeddings with graph structure and textual representations. We first split all spatio-temporal samples with an 8:2 ratio and use only the training split to generate alignment QA. The corresponding prompts are provided in Appendix J. ST-Align consists of three categories of

[1] CC-BY 4.0

questions: Temporal Characters, Spatial Characters, and Spatio-Temporal Characters. For **temporal characters**, each node may contain multiple drift patterns. For each pattern, we generate one QA instance for every matched template key. If the drift type is sinusoidal, we additionally generate three QA instances that query the amplitude A, frequency ω, and phase ϕ, respectively. For **spatial characters**, we generate two questions for each ordered node pair (src, tgt): (1) `edge_relationship`, which asks whether a direct edge exists from src to tgt, and (2) `indirect_connection`, which asks whether an indirect path exists between them. This results in a total of $2 \times N^2$ QA instances, where N is the number of nodes. For **spatio-temporal characters**, the total number of questions is $N + E + 2M$, where N is the number of nodes (one question per node querying its type), E is the number of edges (one question per edge querying its propagation delay), and M is the number of time-varying edge modulations. For each modulation, we generate two questions that query the modulation multiplier and the effective coupling strength.

C Task-Grounded Reward Design

For each response, we compute a format reward r_{format} to enforce a strict output schema:

$$r_{\text{format}} = \mathbb{I}(\text{valid output format}) .$$

where a valid output format requires the response to follow the schema <think>...</think><answer>...</answer>. The task reward r_{task} is defined according to the type of ground-truth answer. If the ground truth y is a single discrete label (e.g., multiple-choice), we assign

$$r_{\text{task}} = \begin{cases} 1, & \hat{y} = y, \\ 0, & \text{otherwise.} \end{cases}$$

$\hat{y}$ is the choice from the answer field. If the ground truth is a numeric sequence $y = \{y_t\}_{t=1}^{T}$ (e.g. T4:Forecasting), we first extract all predicted numbers $\hat{y} = \{\hat{y}_t\}$ from the <answer> field. If no valid numbers are produced, $r_{\text{task}} = 0$. Otherwise, predictions are padded or truncated to length T, and we compute a relative-error–based score:

$$r_{\text{task}} = \frac{1}{T}\sum_{t=1}^{T} 1 - \min\left(1, \frac{|\hat{y}_t - y_t| + \epsilon}{|y_t| + \epsilon}\right),$$

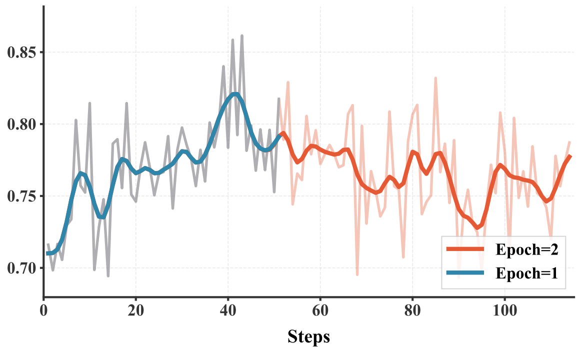


Figure 11: Scaling Up RL Training from 1 epoch to 2 epochs.

where $\epsilon = 10^{-9}$. If the predicted sequence length exactly matches the ground truth, we add a small bonus of 0.1 and clip the final score to $[0, 1]$.

Finally, the per-sample reward is a weighted combination of format and task rewards:

$$r = (1 - \lambda)\, r_{\text{task}} + \lambda\, r_{\text{format}},$$

where $\lambda = 0.5$ in all experiments.

D Implementation Details

We use Qwen3-8B (Yang et al., 2025a) as the base model of **STReasoner**. All experiments are conducted using 8 NVIDIA A100 (80GB) GPUs. Stage 1 (Alignment) is trained on ST-Align for 1,000 steps, and Stage 2 (SFT) is trained on ST-CoT for 400 steps. Stages 1 and 2 are implemented using LlamaFactory (Zheng et al., 2024). We use a cosine learning rate scheduler with a warm-up ratio of 0.2. The `per_device_train_batch_size` is set to 2, and `gradient_accumulation_steps` is set to 32. The learning rate for both the Qwen3-8B and the separate time series encoder is set to 1×10^{-5}. Stage 3 (RL) is trained on ST-RL for one epoch and is implemented using EasyR1 (Zheng et al., 2025) and `verl` (Sheng et al., 2025). The rollout group size G is set to 8. The spatial reward hyperparameters are set to $\alpha = 0.1$ and $\beta = 0.8$. We use a linear learning rate scheduler with a warm-up ratio of 0.2. The `rollout_batch_size` is set to 128, and the learning rate is set to 1×10^{-7}.

E Performance with Confidence Intervals

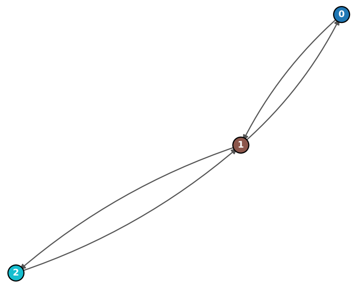
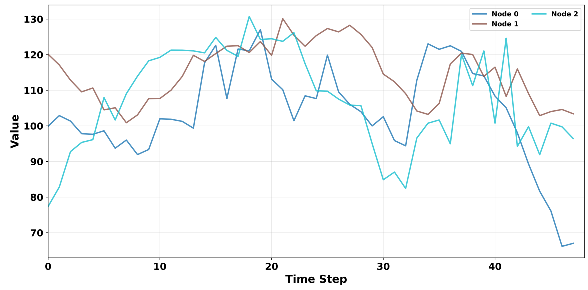
We report the results as mean $\pm$ 95% confidence intervals over five runs. The T1:Etiological achieves an accuracy of 95.39 ± 0.74, the T2:Entity reaches 75.78 ± 0.49, and the T3:Correlation attains 87.20 ± 0.47 accuracy. For the T4:Forecasting, the model obtains a mean absolute error of 65.61 ± 0.06.

F Scaling Up RL Training

Figure 11 shows that extending RL training from 1 to 2 epochs yields further improvements in the accuracy reward at early stages, but the gains gradually saturate and become less stable. This suggests diminishing returns from prolonged RL training under the same reward setting.

G Synthesized Data Pair Showcase

Table 8: Showcase of the synthesized spatio-temporal data and text description pair.

Spatio-Temporal Data	
Scenario Generation Agent	I'll revise the transportation scenario to fix the timing inconsistencies while maintaining all other constraints. **TIME SPAN:** 1 day **SAMPLING FREQUENCY:** 30 minutes (48 total points) **VARIABLE:** traffic flow (vehicles/hour) **NODES:** NODE 0: [DEMAND_SOURCE] Central business district with offices and commercial areas; NODE 1: [PROPAGATION] Major highway junction connecting business district to residential area; NODE 2: [DEMAND_SOURCE] Large residential suburb area. **EDGES:** NODE 0 → NODE 1: Primary outbound route from business district, time_lag=1; NODE 1 → NODE 2: Main highway section leading to residential area, time_lag=1; NODE 2 → NODE 1: Inbound commuter route from residential area, time_lag=1; NODE 1 → NODE 0: Primary inbound route to business district, time_lag=1. **TEMPORAL PATTERNS:** NODE 0: Phase 1 (time 0–13) Night period, baseline: 100 vehicles/hour, amplitude: 0, peak: null; Phase 2 (time 14–34) Evening exodus, baseline: 100 vehicles/hour, amplitude: 300 vehicles/hour, peak: 17 (5:00 PM); propagated_variations: receives morning commute flow from NODE 2 via NODE 1 during time 16–20 [FIXED: accounts for Node 2's peak at t=14 + 2 time_lags]. NODE 1: baseline: 120 vehicles/hour, amplitude: 0, peak: null; propagated_variations: receives evening exodus from NODE 0 during time 18–22 [FIXED: accounts for Node 0's peak at t=17 + 1 time_lag]; receives morning commute from NODE 2 during time 15–19 [FIXED: accounts for Node 2's peak at t=14 + 1 time_lag]. NODE 2: Phase 1 (time 0–13) Night period, baseline: 110 vehicles/hour, amplitude: 0, peak: null; Phase 2 (time 14–34) Morning commute, baseline: 110 vehicles/hour, amplitude: 280 vehicles/hour, peak: 14 (7:00 AM); propagated_variations: receives evening exodus flow from NODE 0 via NODE 1 during time 19–23 [FIXED: accounts for Node 0's peak at t=17 + 2 time_lags]. **Edge Modulation:** Time 14–19 [FIXED: aligned with Node 2's peak activity at t=14], edges affected: NODE 2 → NODE 1, NODE 1 → NODE 0, effect: strong, description: Morning rush hour intensifies inbound traffic flow. Time 17–22 [FIXED: aligned with Node 0's peak activity at t=17], edges affected: NODE 0 → NODE 1, NODE 1 → NODE 2, effect: strong, description: Evening rush hour intensifies outbound traffic flow. **Note:** The baselines (100, 120, 110 vehicles/hour) remain within the same order of magnitude. All propagation times now properly account for cumulative time_lags along the paths.
Other Agents	Due to space limitations, you can find the output of other agents in JSON format (Scenario Parsing Agent in Listing 1, SDEs Parameters Agent in Listing 2, Time-Varying Adj. Agent in Listing 3)

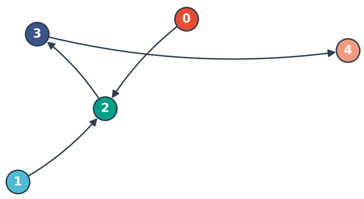
H Reasoning Case Study

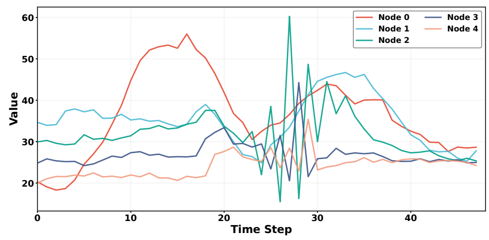
Table 9: Task 1: Etiological Spatial Reasoning example with options and reasoning process.

Task 1: Etiological Spatial Reasoning

User

You are a spatial temporal analysis expert. Node 0 time series with length of 48: <ts><ts/>; Node 1 time series with length of 48: <ts><ts/>; Node 2 time series with length of 48: <ts><ts/>; Node 3 time series with length of 48: <ts><ts/>; Node 4 time series with length of 48: <ts><ts/>; Graph Structure: Node 0→Node 2; Node 1→Node 2; Node 2→Node 3; Node 3→Node 4, please analyze the spatial temporal data and answer the following question: Which etiological scenario can be inferred from the spatio-temporal data?





Options:
A. Factory ventilation system with chemical processes spreading through connected chambers based on pressure differentials.
B. Coastal emission system with marine and port activities dispersing through atmospheric layers toward inland regions.
C. Agricultural runoff network with pesticide sources flowing through water channels, affected by seasonal discharge patterns.
D. Urban pollution network with industrial and traffic sources dispersing through nodes, modulated by time-based transport patterns.

STReasoner

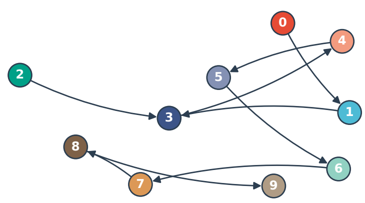
<think>
Let me analyze the spatial-temporal data systematically.
Graph Structure Analysis:
Node 0 → Node 2 → Node 3 → Node 4; Node 1 → Node 2. This is a converging pattern where Node 0 and Node 1 feed into Node 2, which then propagates through Node 3 to Node 4.
Temporal Pattern Analysis:
1) Node 0 (source). Starts at ~ 20, increases to a peak at 56.01 (timestep 20), then decreases to ~ 28 by the end; a clear rise-and-fall event centered near the midpoint.
2) Node 1 (source). Starts at ~ 34.67, rises to a peak at 46.70 (timestep 18), then decreases to ~ 27.79; similar event with slightly different timing and magnitude.
3) Node 2 (convergence). Starts at ~ 30 and rises to ~ 36, with spikes (e.g., 42.19 at timestep 19 and 60.27 at timestep 21) that occur after the peaks in Nodes 0 and 1; then declines to ~ 25. This reflects aggregation from two upstream sources and delayed amplification at the junction.
4) Node 3 (downstream). Starts at ~ 25, increases to ~ 44 (timesteps 20–21), showing a delayed response versus upstream nodes; then decreases back to ~ 25.
5) Node 4 (furthest downstream). Baseline around 20–22, increases to ~ 35 (timesteps 22–23), the most delayed and damped response; returns toward baseline (~ 24).
Key Observations: Multiple sources (Nodes 0 and 1) converge at Node 2; there is a clear transport lag along 0/1→2→3→4, and peak magnitudes attenuate downstream. The pattern looks like a single event propagation, not a seasonal cycle.
Scenario Evaluation: **A)** Ventilation/pressure-driven spread would more likely show sharper, more immediate responses rather than gradual lagged attenuation across the chain. **B)** Coastal/atmospheric dispersion would typically show different signatures (e.g., broader regional mixing and potentially periodic drivers) rather than a clean directed network propagation with two discrete sources. **C)** Agricultural runoff is often tied to seasonal or weather-driven variability; this trace is a single event-like pulse rather than seasonal discharge behavior. **D)** Urban pollution fits: two sources (industrial and traffic) merge, then disperse with time-based transport delays and spatial attenuation through the network.
Therefore, the inferred scenario is **D**.
</think>
<answer>D</answer>

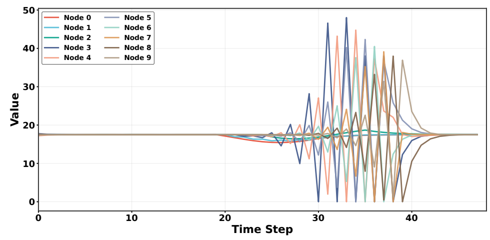
Table 10: Task 2: Spatial Entity Identification example with options and reasoning process.

Task 2: Spatial Entity Identification

User

You are a spatial temporal analysis expert. Node 0 time series with length of 48: <ts><ts/>; Node 1 time series with length of 48: <ts><ts/>; Node 2 time series with length of 48: <ts><ts/>; ... Node 8 time series with length of 48: <ts><ts/>; Node 9 time series with length of 48: <ts><ts/>; Graph Structure: Node 0→Node 1; Node 1→Node 3; Node 2→Node 3; Node 3→Node 4; Node 4→Node 5; Node 5→Node 6; Node 6→Node 7; Node 7→Node 8; Node 8→Node 9, please analyze the spatial temporal data and answer the following question: Which (name, description) pair should Node 9 correspond to?





Options:
A. Endpoint Alpha, Last measurement checkpoint.
B. Exit Node, Terminal collection station.
C. Final Point, Final monitoring point.
D. Terminal Hub, Network termination sensor.

STReasoner

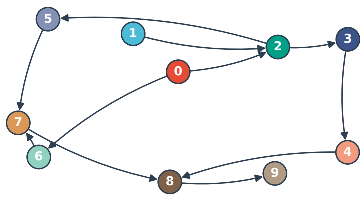
<think>
Let me analyze the spatial-temporal data systematically.
First, let me understand the graph structure: Node 0 → Node 1 → Node 3 ← Node 2; Node 3 → Node 4 → Node 5 → Node 6 → Node 7 → Node 8 → Node 9.
This forms a linear chain from Node 3 to Node 9, with Node 9 being the last node in the sequence.
Now let me analyze the temporal patterns.
1. Nodes 0–2. These nodes show relatively stable values around 17.5, with minor fluctuations. They act as early nodes in the network.
2. Node 3. Node 3 shows a dramatic change around timestep 32–36, with extreme spikes (values such as 48.03, 39.12, and 35.01) followed by drops to 0.00, and then gradually recovers back to approximately 17.5.
3. Nodes 4–8. Each of these nodes exhibits a similar disturbance pattern that propagates through the network with increasing delay: Node 4 shows a spike around timesteps 33–37 (44.77, 39.21, 34.65); Node 5 around timesteps 34–38 (42.33, 38.41, 33.90); Node 6 around timesteps 35–39 (40.50, 37.60, 35.22); Node 7 around timesteps 36–40 (39.12, 37.22, 35.64); Node 8 around timesteps 37–41 (37.98, 36.51, 34.61).
4. Node 9. Node 9 receives the disturbance last, around timesteps 38–42 (36.92, 34.12, 32.26, 31.23). The magnitude is lower and more dampened compared to earlier nodes.
Key observations. The disturbance propagates sequentially through the network. Node 9 lies at the end of the chain (Node 8 → Node 9), shows the final and most dampened version of the disturbance, and has no outgoing connections, indicating that it is a terminal node.
Looking at the options: Option A, "Endpoint Alpha, Last measurement checkpoint," suggests a specific endpoint. Option B, "Exit Node, Terminal collection station," suggests an exit or collection point. Option C, "Final Point, Final monitoring point," clearly indicates the last monitoring location. Option D, "Terminal Hub, Network termination sensor," suggests network termination.
Given that Node 9 is the last node in the sequential chain, receives the propagated disturbance last, has no outgoing connections, and represents the final monitoring position in the network, the most appropriate description is "Final Point, Final monitoring point" (Option C).
</think>
<answer>C</answer>

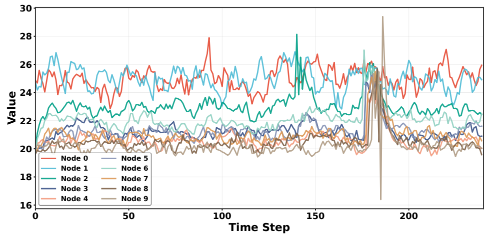
Table 11: Task 3: Spatial Correlation Reasoning example with options and reasoning process.

Task 3: Spatial Correlation Reasoning

User

You are a spatial temporal analysis expert. Node 0 time series with length of 240: <ts><ts/>; Node 1 time series with length of 240: <ts><ts/>; Node 2 time series with length of 240: <ts><ts/>; ...; Node 8 time series with length of 240: <ts><ts/>; Node 9 time series with length of 240: <ts><ts/>; Graph Structure: Node 0→Node 2; Node 0→Node 6; Node 1→Node 2; Node 2→Node 3; Node 2→Node 5; Node 3→Node 4; Node 4→Node 8; Node 5→Node 7; Node 6→Node 7; Node 7→Node 8; Node 8→Node 9, please analyze the spatial temporal data and answer the following question: Which statement best describes the relationship between Node 2 and Node 4 during time steps 177-180 (1 time step = 1 hour)?





Options:
A. There is no significant thermal signal transmission between Node 2 and Node 4.
B. Thermal signal travels directly from Node 2 to Node 4 without intermediate nodes.
C. Thermal signal flows from Node 4 back to Node 2 through a feedback loop.
D. Thermal signal propagates from Node 2 to Node 4 via Node 3, following a sequential central-to-eastern pathway.

STReasoner

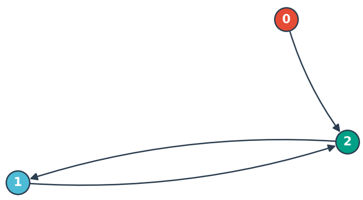
<think> Let me analyze the spatial-temporal data for the relationship between Node 2 and Node 4 during time steps 177-180. First, let me identify the graph structure path from Node 2 to Node 4: Node 2 → Node 3 → Node 4. This means Node 3 is an intermediate node between Node 2 and Node 4. Now let me extract the values at time steps 177-180 (indices 176-179 in 0-indexed arrays).
Node 2: Time 177: 21.91; Time 178: 22.01; Time 179: 22.11; Time 180: 22.21. Node 3: Time 177: 21.02; Time 178: 21.07; Time 179: 21.11; Time 180: 21.15. Node 4: Time 177: 20.04; Time 178: 20.05; Time 179: 20.06; Time 180: 20.06.
Let me also check what happens before and after to understand the temporal pattern. Node 2 (earlier): Time 176: 21.81; Time 177: 21.91 (increase of 0.10); Time 178: 22.01 (increase of 0.10); Time 179: 22.11 (increase of 0.10); Time 180: 22.21 (increase of 0.10). Node 3 (earlier): Time 176: 21.01; Time 177: 21.02 (increase of 0.01); Time 178: 21.07 (increase of 0.05); Time 179: 21.11 (increase of 0.04); Time 180: 21.15 (increase of 0.04). Node 4 (earlier): Time 176: 20.03; Time 177: 20.04 (increase of 0.01); Time 178: 20.05 (increase of 0.01); Time 179: 20.06 (increase of 0.01); Time 180: 20.06 (stable).
Looking at the pattern: (1) Node 2 shows a clear increasing thermal signal starting at time 177; (2) Node 3 shows an increasing pattern with a slight delay and dampening; (3) Node 4 shows a gradual increase with further delay. The signal appears to propagate from Node 2 → Node 3 → Node 4, which matches the graph structure. This is a sequential propagation through an intermediate node (Node 3).
Let me evaluate the options. A. No significant thermal signal – FALSE, there are clear changes in values. B. Direct transmission without intermediate nodes – FALSE, the graph shows Node 2→Node 3→Node 4, and the temporal pattern shows Node 3 responds first before Node 4. C. Flow from Node 4 back to Node 2 – FALSE, the temporal pattern shows Node 2 changes first, then Node 3, then Node 4. D. Propagates from Node 2 to Node 4 via Node 3 – TRUE, this matches both the graph structure (Node 2→Node 3→Node 4) and the temporal pattern where changes in Node 2 are followed by changes in Node 3, then Node 4.
</think>
<answer>D</answer>

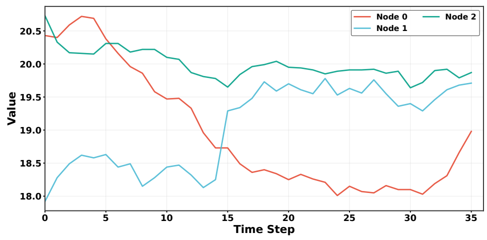
Table 12: Task 4: In-Context Spatio-Temporal Forecasting example with options and reasoning process.

Task 4: In-Context Spatio-Temporal Forecasting

User

You are a spatio-temporal analysis expert. Node 0 time series with length of 36: <ts><ts/>; Node 1 time series with length of 36: <ts><ts/>; Node 2 time series with length of 36: <ts><ts/>; Graph Structure: Node 0→Node 2; Node 1→Node 2; Node 2→Node 1, please analyze the spatial temporal data and answer the following question: Given the context Maximum solar heating period enhances thermal transfer, predict the value of node 2 for the next 3 steps. Historical observation window: 30-35.





STReasoner

<think>
Let me analyze the spatial-temporal data step by step.
Step 1: Understand the Graph Structure. The graph structure consists of directed dependencies among nodes. Node 0 influences Node 2, and Node 1 also influences Node 2, indicating two upstream sources. In addition, Node 2 feeds back to Node 1, forming a local feedback loop between Node 1 and Node 2.
Step 2: Analyze the Historical Window (positions 30–35). Within the historical observation window, Node 0 has values 19.27, 19.41, 19.51, 19.61, 19.68, and 18.97. Node 1 records values of 19.39, 19.52, 19.61, 19.68, 19.78, and 19.71. Node 2 shows a smooth increasing sequence with values 19.69, 19.74, 19.78, 19.82, 19.85, and 19.87.
Step 3: Identify Trends. Considering the full time series, Node 0 exhibits an increasing trend until approximately position 23, where it reaches around 20.72, followed by a decreasing phase with a sharp drop at position 35. Node 1 shows a strong overall increasing trend from 17.92 to around 19.78 at position 33, followed by a slight decrease at the end. Node 2 maintains a steady and monotonic increase from about 19.64 to 19.87 by position 35.
Step 4: Context Analysis. The context statement "Maximum solar heating period enhances thermal transfer" suggests a heating-dominated regime. However, the observed data indicate that Node 0 has reached saturation and started to decline, while Node 1 continues to increase but at a slower rate. Node 2 follows a stable increasing trajectory that reflects combined influence from both upstream nodes.
Step 5: Predict Next 3 Steps for Node 2. Given that Node 2 is jointly influenced by Node 0 and Node 1, the sharp decline in Node 0 at position 35 and the slight decrease in Node 1 suggest that the growth rate of Node 2 will slow down. Although Node 2 has been steadily increasing, it is likely approaching saturation. Therefore, the predicted values for Node 2 are approximately 19.88 at position 36, 19.89 at position 37, and 19.90 at position 38, reflecting continued but diminishing growth.
</think>
<answer>[19.88, 19.89, 19.90]</answer>

Algorithm 1 Multi-Agent Pipeline for Network SDE Data Synthesis

Require: N (number of nodes)
Ensure: $\mathcal{S}$ (structured scenario), Θ (network SDE), $\mathbf{X}$ (time series)

$s \leftarrow \textsc{Agent}_1(N)$ ▷ Agent 1: scenario generation
$T \leftarrow \textsc{InferSeqLen}(s)$
$\mathcal{S} \leftarrow \textsc{Agent}_2(s)$ ▷ Agent 2: scenario parsing with Judge 1 refinement
for $\ell = 1 \ldots L$ **do**
  $j \leftarrow \textsc{Judge}_1(s, \mathcal{S})$
  if j = approve **then**
    break
  end if
  $s \leftarrow \textsc{Agent}_1(s, j)$
  $\mathcal{S} \leftarrow \textsc{Agent}_2(s)$
end for
$\Theta \leftarrow \textsc{Agent}_3(\mathcal{S})$ ▷ Agents 3–4: SDE and adjacency with Judge 2 refinement
$A \leftarrow \textsc{Agent}_4(\mathcal{S})$
for $k = 1 \ldots K$ **do**
  $\tilde{\mathbf{X}} \leftarrow \textsc{Simulate}(\Theta, A, T)$
  $j \leftarrow \textsc{Judge}_2(\mathcal{S}, \tilde{\mathbf{X}})$
  if j = approve **then**
    break
  end if
  $\Theta \leftarrow \textsc{Agent}_3(\mathcal{S})$
  $A \leftarrow \textsc{Agent}_4(\mathcal{S})$
end for
$\mathbf{X} \leftarrow \textsc{Simulate}(\Theta, A, T)$
return $\mathcal{S}, \Theta, \mathbf{X}$

I ST-Bench Prompt Template

Etiological Reasoning Dataset Generation Prompt

```
You are given spatio-temporal context. Produce a multiple-choice item where the
correct option is a concise macro summary of the Scenario Context.

Scenario: {scenario}

Requirements:
1) "observation": A concise macro summary of the Scenario in 12-20 words.
  - It must describe the system at a high level (e.g., an interconnected hydroponics
  circulation system, a wastewater treatment facility, etc.). Facility names are not
   important; do not invent new names.
   - It must explicitly mention the key node variables provided.
2) "options": list of exactly four scenario summaries (each <20 words) without
labels.
   - The FIRST entry must be identical to "observation" (verbatim match).
  - The other three must be fluent but incorrect (they must contradict the Scenario
   or mention entities/processes not present in the Scenario/Involved Nodes).

Output JSON format:
{{
```

```
    "observation": "observed phenomena",
   "options": ["Correct summary", "Distractor 1", "Distractor 2", "Distractor 3"]
}}

Strictly based on given context, do not introduce external knowledge. Return only a
valid JSON object, without any explanations outside JSON.
```

Entity Reasoning Dataset Generation Template

```
You are given a node ID and its correct name and description from a spatio-temporal
network simulation.
Your task is to generate a multiple-choice question to identify which
(name, description) pair corresponds to the target node.

Target Node:
- ID: {node_id}
- Correct Name: {node_name}
- Correct Description: {node_description}

Requirements:
1) "options": list of exactly four strings containing (name, description) pairs.
Do NOT prefix with labels.
   - The FIRST entry must be the correct pair (verbatim match to the given name
   and description).
   - The remaining three must be fluent but !!!incorrect!!!.
       - They should describe plausible but different node roles or locations.
       - Maintain the same style and variable domain (e.g., “traffic flow”,
       “industrial output”, “water pressure”).
       - Avoid contradictions or unrealistic content.

Output JSON format:
{{
    "question": "Which (name, description) pair should Node {node_id}
    correspond to?",
    "options": ["Correct pair", "Distractor 1", "Distractor 2", "Distractor 3"]
}}

Strictly return only a valid JSON object. Do not add explanations outside JSON.
```

Correlation Reasoning (Direct Causal) Dataset Generation Template

```
You are a question designer for a spatio-temporal reasoning test.
Your task is to create a multiple-choice question based on a given piece of
true evidence of a causal influence.

True Evidence:
- Source Node: {source_node_name} (ID: {source_node_id})
- Target Node: {target_node_name} (ID: {target_node_id})
- Time Steps: {time_period} (1 time step = {sampling_frequency})
- Correct Description of Event: "{correct_description}"
```

```
Your Task:
1.  Create a "question" that asks which statement best describes the influence on
Node {target_node_id} during the specified time steps.
   - Explicitly use the phrase "time steps {time_period}" in the question text and
    append "(1 time step = {sampling_frequency})".
2.  Create an "options" list containing exactly four strings. The FIRST entry MUST
be the Correct Description provided above, verbatim.
3.  The remaining three entries must be plausible but incorrect distractors
describing different sources or incorrect effects.

Output Format:
Return ONLY a valid JSON object.
{{
    "question": "Your generated question.",
   "options": ["The correct description verbatim", "Distractor 1", "Distractor 2"
    , "Distractor 3"]
}}
```

Correlation Reasoning (Multi Hop) Dataset Generation Template

```
You are an expert in spatio-temporal network analysis. Your task is to analyze a set
of direct causal events and identify a multi-hop propagation path to create a
multiple-choice question.

Given Direct Causal Events (Adjacency Modulations):
{adjacency_modulations}

Time Step Note: 1 time step = {sampling_frequency}

Your Task:
1.  Analyze the events: Find a sequence of events that form a logical multi-hop path.
The time periods of the events should be overlapping or consecutive.
2.  Synthesize a description: Create a concise, high-level description for the
entire multi-hop event. This will be your correct answer.
3.  Identify Nodes and Time: State the start node, end node, and the overall time
window for the multi-hop event in terms of time steps.
4.  Generate a Question: Create a "question" asking for the most appropriate
description of the relationship between the start and end nodes during those
time steps.
    - Explicitly reference the interval as "time steps X-Y" and append
    "(1 time step = {sampling_frequency})" in the question text.
5.  Generate Options: Create an "options" list with exactly four strings.
The FIRST entry must be your synthesized description. The other three should be
plausible but incorrect distractors.

Output Format:
Return ONLY a valid JSON object.
{{
    "question": "Your generated question about the multi-hop relationship.",
    "options": ["Your synthesized correct description", "Distractor 1",
        "Distractor 2", "Distractor 3"]
```

```
}}
"""
```

Forecasting Reasoning Dataset Generation Template

```
Analyze the provided context and generate a detailed forecast description.

Scenario Details:
- Target Node: {target_node_name} (ID: {target_node_id})
- Variable: {target_node_variable}
- Observation Window: Steps {history_window}
- Prediction Window: Steps {prediction_window}
- Key Event: {events}
- Statistical Hints: {referenced_stats}

Task:
Based *only* on the information above, provide a JSON object describing the forecast.

Output Format:
```json
{
  "observation_window": "{history_window}",
  "prediction_window": "{prediction_window}",
  "prediction_length": {prediction_length},
  "target_node_id": {target_node_id},
  "context_description": "{events}",
 "summary": "text describing the expected behaviour during the prediction window",
  "confidence": "low/medium/high"
}
```
```

J ST-Align Prompt Template

Alignment Dataset Generation Template

```
{
  "temporal": {
   "drift_type": "Question: What is the evolution pattern of node {node_id} during
        the time range {time_range}? Answer: {drift_type}",
    "baseline": "Question: What is the long-term baseline value of node {node_id}
        during the time range {time_range}? Answer: {baseline}",
    "kappa": "Question: What is the mean reversion speed (kappa) of node {node_id}
        during the time range {time_range}? Answer: {kappa}",
     "sigma": "Question: What is the random fluctuation intensity (sigma) of node
        {node_id} during the time range {time_range}? Answer: {sigma}",
    "lambda": "Question: What is the coupling strength (lambda) of node {node_id}
        during the time range {time_range}? Answer: {lambda}",
   "diffusion_shape": "Question: What is the diffusion shape (diffusion_shape) of
      node {node_id} during the time range {time_range}? Answer: {diffusion_shape}",
     "sinusoidal_A": "Question: What is the sinusoidal amplitude (A) of
        node {node_id} during the time range {time_range}? Answer: {A}",
     "sinusoidal_omega": "Question: What is the sinusoidal frequency (omega) of
```

```
        node {node_id} during the time range {time_range}? Answer: {omega}",
    "sinusoidal_phi": "Question: What is the sinusoidal phase (phi) of node
        {node_id} during the time range {time_range}? Answer: {phi}"
  },
  "spatial": {
   "edge_relationship": "Question: Is there a direct  connection from node {src} to
    node {tgt}? Answer: {answer}",
   "indirect_connection": "Question: Is there an indirect path (through one or more
        intermediate nodes) from node {src} to node {tgt}? Answer: {answer}"
  },
  "spatial_temporal": {
   "node_type": "Question: What is the type of node {node_id}? demand_source or
        propagation? Answer: {node_type}",
   "edge_lag": "Question: What is the time lag between node {src} and node {tgt}?
        Answer: {lag}",
   "edge_modulation": "Question: What is the modulation multiplier (multiplier) of
     edge {edge} during the time range {time_range}? Answer: {multiplier}",
    "effective_coupling_strength": "Question: What is the effective coupling
        strength of edge {edge} during the time range {time_range}? Answer:
        {multiplier * base_adjacency}"
  }
}
```

K Agent Prompt Template

Scenario Generation Prompt (Agent 1)

```
You are Agent 1: Scenario Generation Agent.

Your task:
Generate a realistic scenario description for a spatial-temporal dataset
with {num_nodes} interconnected nodes.

CORE PRINCIPLES:
- Create synthetic but realistic scenarios with SPECIFIC, CONCRETE details
- Provide information that enables accurate time series generation
- Describe the physical system clearly (what flows, where, when, why)

NODE TYPE DEFINITIONS:
1. DEMAND_SOURCE (1 or 2 nodes):
Definition: Nodes that independently generate or consume the monitored variable.
Characteristics:
- Must specify baseline and amplitude values
- Must have exactly ONE self_generated peak (exogenous cycle)
- Any additional variations must be explicitly marked as propagated
from other nodes

2. PROPAGATION
Definition: Relay nodes that transmit flows without independent generation.
Characteristics:
- Must specify a baseline value (nonzero, low). This represents a small
```

```
(much smaller than the demand_source nodes), ambient background level and
ensures physical realism (e.g., a river junction is never completely dry).
- Amplitude must equal 0
- peak must be null
- All variations must be propagated from other nodes

BASELINE CONSISTENCY RULE:
- All nodes (both DEMAND_SOURCE and PROPAGATION) must have baseline values within
the same order of magnitude
- Baseline values should be similar across all nodes
- This ensures network coupling effects are meaningful and nodes can effectively
influence each other

BASELINE REALISM RULE:
- The `baseline` value must reflect a realistic, physically plausible state for
the node, often representing its value during a "calm" or "initial" period.
- This ensures the simulation starts from a sensible state and that the
mean-reverting behavior is anchored to a meaningful physical value.

REQUIREMENTS:
1. Number nodes as NODE 0, NODE 1, NODE 2, ... (0-indexed)
2. All nodes monitor the SAME variable.
3. Specify spatial relationships at different time
4. Specify TIME SPAN and SAMPLING FREQUENCY such that total points are
smaller than {max_seq_len}
5. Temporal dynamics rules:
- DEMAND_SOURCE nodes follow the above constraints (single exogenous peak +
possible propagated variations)
- PROPAGATION nodes follow the above constraints (only propagated variations,
no self-generated peaks)
6. Edge Consistency Rule:
- Any propagated variation described in TEMPORAL PATTERNS must correspond to an
explicitly declared directed edge in the EDGES section.
- No hidden or undeclared propagation is allowed.
- The graph must be connected, ensuring that the effects from demand_source nodes
can propagate to all other nodes.
7. Direction Integrity Rule:
- If a demand_source node generates an outbound peak (e.g., evening exodus from
downtown), the corresponding outbound edge (e.g., NODE 2 → NODE 1) must be explicitly
listed in EDGES.
- Temporal patterns cannot contradict or introduce flows that are missing from the
graph structure.
8. Demand Source Connectivity Rule:
- Every DEMAND_SOURCE node must have at least one outgoing edge, i.e., it must appear
as the source node in at least one directed edge in the EDGES section.
9. Propagated Event Timing Consistency Rule (CRITICAL):
- When describing Edge Modulation for a propagating event (e.g., morning rush hour
traveling through multiple edges), you MUST account for cumulative time lags.
- Key Principle: An event cannot activate an edge before it physically arrives
at that edge's source node.
```

```
- Design Strategy: Create staggered, overlapping time windows that shift
forward by the time_lag amount for each successive edge in the chain.
10. Time Lag Design Guideline:
- Use time_lag>=1 only when the physical travel/transmission time is significant
relative to sampling frequency
- For long chains (>3 nodes), consider small sampling frequency to keep cumulative
delays

AVOID:
- Vague phrases ("depends on conditions", "may vary")
- Real geographic names (cities, countries)
- Specific calendar dates (use relative time: "weekdays", "weekend")
- Special events or holidays
- Assigning multiple independent peaks to a single demand_source node
- More than 2 demand_source nodes

OUTPUT FORMAT (STRICT):
- TIME SPAN: [exact duration, e.g., "1 year","1 day", "1 week"]
- SAMPLING FREQUENCY: [exact interval, e.g., "1 day", "1 week", "1 hour"]
- VARIABLE: [single variable name, e.g., "traffic flow (vehicles/hour)",
"power demand (MW)", "water temperature (°C)", "network bandwidth (Gbps)",
"migration intensity (individuals/day)"]
- NODES:
  - NODE 0: [type: DEMAND_SOURCE or PROPAGATION] [description]
  - NODE 1: [type: DEMAND_SOURCE or PROPAGATION] [description]
  - ...
- EDGES:
  - NODE 0 → NODE 1: [relationship description, including any time lag,
  e.g., "with a 5-day delay"]
  - NODE 1 → NODE 2: [relationship description]
  - NODE 1 → NODE 0: [relationship description]
  - NODE 2 → NODE 1: [relationship description]
  - ...
- TEMPORAL PATTERNS:
For each node, describe its periodicity characteristics, which may vary over time.
- NODE 0:
    - Can have multiple phases, each with a time period and behavior.
    - Example Phase 1 (time 0-239):
      - Behavior: stable, mean-reverting around a baseline
      - baseline: [numerical value with unit, e.g., "100 individuals/day"]
      - amplitude: 0
      - peak: null
    - Example Phase 2 (time 240-260):
      - Behavior: sinusoidal increase/decrease
      - baseline: [numerical value, same as other phases]
      - amplitude: [numerical value, < 5*baseline]
      - peak: [time step number, e.g., "250"]
    - propagated_variations: [if any, describe which nodes and when,
   e.g., "receives flow from NODE 0 with 3-step delay, peaking around step 63"]
- NODE 1:
```

```
    - baseline: [numerical value with unit, MUST be similar to NODE 0's baseline,
    e.g., if NODE 0 is 100, use 90-110 range]
    - amplitude: [numerical value, must be 0 for PROPAGATION nodes]
    - peak: [time step number or null]
    - propagated_variations: [describe propagation sources and timing]
- ... (repeat for all nodes, maintaining similar baseline values)
- Edge Modulation: Describe how edge influences vary over time
    (must have at least one modulation)
    Format for each time-dependent modulation:
    - Time [time period, e.g., "50-70" or "7-9"]:
      - Edges affected: [e.g., "NODE 0 → NODE 1"]
      - Effect: [strong/moderate]
      - Description: [brief explanation]
```

Scenario Parsing Prompt (Agent 2)

```
You are Agent 2: Scenario Parsing Agent.

Your task: Convert a natural language scenario description into a STRICT,
STRUCTURED JSON object.

INPUT: Natural language scenario description (from Agent 1)

OUTPUT: A single valid JSON object with NO markdown, NO explanations, NO comments,
NO trailing commas.

---

JSON SCHEMA (STRICT):
{
  "time_span": "string (e.g., '7 days')",
  "sampling_frequency": "string (e.g., '1 hour')",
  "seq_len": "integer (number of time steps)",
  "variable": "string (exactly ONE variable monitored by all nodes)",
  "nodes": [
    {"id": 0, "type": "demand_source or propagation", "name": "string",
    "description": "string"},
    {"id": 1, "type": "demand_source or propagation", "name": "string",
    "description": "string"},
    ...
  ],
  "edges": [
    {
      "source": 0,
      "target": 1,
      "relationship": "string describing directional influence",
      "time_lag": "integer (optional, number of time steps for delay)"
    },
    {
      "source": 1,
      "target": 2,
```

```
      "relationship": "string describing directional influence",
      "time_lag": "integer (optional)"
    },
    ...
  ],
  "drift_patterns": {
    "repeat": "boolean (optional, indicates if the pattern sequence repeats)",
    "repeat_period": "integer (optional, defines the cycle duration in steps
    if repeat is true, e.g., 24 for a daily cycle)",
    "nodes": [
      {
        "id": "integer (node ID)",
        "patterns": [
          {
            "time_range": "[start_time, end_time] (integer array)",
            "behavior": "string (e.g., 'mean_reverting', 'sinusoidal')",
            "baseline": "number (long-term average level, must be > 0)",
            "amplitude": "number (peak deviation from baseline, >= 0)",
        "peak": "integer or null (time step of the peak for sinusoidal behavior)"
          }
        ],
        "propagated_variations": [
          {
            "time": "string (time location or range)",
            "origin": "propagated",
            "source": "integer (node_id)",
            "delay": "string (optional, e.g., '3 days', '2 hours')",
            "description": "string (short explanation)"
          }
        ]
      }
    ]
  },
  "adjacency_modulation": {
    "patterns": [
      {
        "time_period": "string (e.g., '50-70', '7-9')",
        "effect": "string (strong/moderate)",
     "applies_to": "string or array of strings (e.g., '0->1' or ['0->1', '1->2'])",
        "description": "string (explanation of why this modulation occurs)"
      }
    ]
  },
  "spatial_layout": {
    "0": {"x": number, "y": number},
    "1": {"x": number, "y": number},
    ...
  }
}
```

```
PARSING RULES:
0. Calculate seq_len:
- Extract the numeric values from time_span and sampling_frequency
- Convert both to the same unit (e.g., hours, days)
- Calculate: seq_len = time_span / sampling_frequency
1. Node Classification:
- If description mentions "generate", "originate", "consume", "demand", "source"
→ demand_source
- If description mentions "relay", "connector", "junction", "pass through",
"transmit" → propagation
- Each node must be classified based on its physical role
2. Edge Construction:
- Extract all directional influences from scenario description
- For each edge, extract these attributes:
 * source: source node ID
 * target: target node ID
 * relationship: brief description of the connection
 * time_lag: (optional) integer representing delay in time steps
 (e.g., if scenario says "5 day delay" and sampling is "1 day", time_lag should be 5)
3. Drift Patterns:
- This section describes the time-varying behavior of each node.
- For each node, parse its temporal description into a list of `patterns`.
- Each pattern in the list must describe a specific behavior over a `time_range`,
and include:
 * `baseline`: The typical long-term average value. This must be > 0.
 * `amplitude`: The peak deviation from the baseline. This must be >= 0.
 * `peak`: The time step where the peak occurs (for `sinusoidal` behavior).
 Must be null for other behaviors.
- Parse any `propagated_variations` described for the node.
- If the patterns repeat (e.g., a daily cycle), set `repeat: true` and define the
cycle's duration in `repeat_period` (e.g., `24` for a 24-hour cycle).
- Coverage Constraint: If `repeat` is true, the `time_range` of all patterns for
a node must completely and contiguously cover the range from `0` to `repeat_period`.
- Constraints per Node Type (CRITICAL):
 - For demand_source nodes: Can have patterns with `amplitude` > 0.
 - For propagation nodes: All patterns MUST have `amplitude: 0` and `peak: null`.
 Their variation comes only from `propagated_variations`.
4. Adjacency Modulation:
- Extract concrete time-dependent edge effects from scenario
- CRITICAL: For propagating events (e.g., traffic flowing through a chain of nodes),
each edge in the path should have its own modulation entry with a properly staggered
time_period that accounts for the cumulative time_lag
- Describe modulation patterns with:
 * time_period: when the modulation occurs (e.g., "50-70", "7-9" - just numbers
 representing time steps)
   - For event chains, ensure each edge's time_period starts AFTER the event could
    have arrived from the previous edge
   - If edge A->B has time_lag=1 and modulation starts at t=15, then edge B->C should
    have modulation starting at t>=16
 * effect: strength of the modulation (strong/moderate)
```

```
   - strong: significant enhancement of edge influence
   - moderate: moderate enhancement of edge influence
 * applies_to: which edge(s) are affected
   - Can be a single edge string (e.g., "0->1")
  - Can be an array of edge strings (e.g., ["0->1", "1->2"]) ONLY when they truly share
   the exact same time window (be cautious with this for event chains)
 * description: explanation of why this modulation happens

5. Spatial Layout:
   - Generate simple 2D coordinates for visualization
   - Arrange nodes logically (e.g., source on left, propagation in middle,
   sink on right)

6. Output Format:
   - Valid JSON only (RFC 8259)
   - Double quotes for strings
   - No trailing commas
   - No markdown code blocks
   - No extra text

Example: [See source code]

INPUT SCENARIO:
{scenario}
```

SDE Parameters Generation Prompt (Agent 3)

```
You are Agent 3: SDE Parameters Generation Agent.

Your task: Generate hierarchical SDE parameters from a structured scenario JSON.

INPUT: Structured JSON from Agent 2 (scenario parsing agent)

OUTPUT: Hierarchical SDE parameters as strict JSON (NO markdown, NO comments)

SDE MODEL (per node i):
dX_i(t) = [ drift_i(t, X_i) + lambda_i * capital_sigma_j A_ji(t) * (X_j - X_i) ] dt
+ sigma_i * g_i(X_i) dW_i(t)

Components:
- drift_i: drift term (type-dependent)
- lambda_i: coupling strength
- A_ji(t): time-varying adjacency (from Agent 4)
- sigma_i: base volatility
- g_i(X_i): diffusion shape function

DRIFT TYPES:
1. mean_reverting (default):
   - Formula: drift = kappa * (mu_t - X_t)
   - Parameters: kappa (mean reversion speed), baseline (mu_t)
   - Constraint: 0.01 < kappa < 0.5
```

```
   - Usage: REQUIRED for propagation nodes, allowed for demand_source nodes
2. constant:
   - Formula: drift = alpha
   - Parameters: alpha (constant drift rate)
   - Constraint: alpha \in R
   - Usage: ONLY allowed for demand_source nodes
3. sinusoidal:
   - Formula: drift = kappa * (baseline + A*sin(omega*t + phi) - X_t)
   - Parameters: A (amplitude), omega (frequency), phi (phase shift)
   - Constraint: A > 0, omega > 0, phi \in R (ALL SCALARS, NOT ARRAYS)
   - CRITICAL: Single harmonic only - no multi-frequency superposition
   - Usage: ONLY allowed for demand_source nodes
4. logistic:
   - Formula: drift = r * X_t * (1 - X_t/baseline)
   - Parameters: r (growth rate), baseline (carrying capacity)
   - Constraint: 0 < r < 0.1, baseline > 0
   - Usage: Allowed for both demand_source and propagation nodes

CRITICAL CONSTRAINTS (STRICTLY ENFORCED):
1. Node Type Constraints:
   - propagation nodes: MAY use mean_reverting or logistic drift (small r)
  - demand_source nodes: MAY use mean_reverting, sinusoidal, constant, or logistic
2. Parameter Ranges (for stability):
   - 0.01 < kappa < 0.5 (mean reversion speed)
   - 0.8 < lambda < 1.5 (coupling strength - high values for
   realistic network dynamics)
   - sigma < 0.01*baseline (volatility, must be less than 1% of the baseline)
   - For sinusoidal: A, omega, phi must be scalars (not arrays)
   - For logistic: 0 < r < 0.1, baseline > 0
3. Propagation Node Special Rules:
   - Use LOW kappa (0.05-0.2) for weak self-reversion
   - Use HIGH lambda (1.0-1.5) for strong neighbor coupling
   - This ensures propagation nodes relay upstream flows effectively
4. Diffusion Shapes:
   - "constant": g(X) = 1
   - "sqrt": g(X) = sqrt(|X| + 1e-6)
   - "linear": g(X) = 1 + alpha*|X|

HIERARCHICAL OUTPUT STRUCTURE:
{{
  "global_defaults": {{
    "drift_type": "mean_reverting",
    "node_type": "demand_source",
    "kappa": 0.25,
    "baseline": 50.0,
    "lambda": 1.0,
    "sigma": 2.0,
    "diffusion_shape": "constant"
  }},
  "group_params": {{
```

```
    "demand_sources": {{
      "node_type": "demand_source",
      "drift_type": "sinusoidal",
      "baseline": 100.0,
      "A": 30.0,
      "omega": 0.2618,
      "phi": 0.0,
      "kappa": 0.25,
      "lambda": 1.0,
      "sigma": 2
    }},
    "propagation_nodes": {{
      "node_type": "propagation",
      "drift_type": "mean_reverting",
      "baseline": 50.0,
      "kappa": 0.1,
      "lambda": 1.2,
      "sigma": 2
    }}
  }},
  "node_overrides": {{
    "0": {{
      "group": "demand_sources",
      "drift_patterns": [
        {{
          "time_range": [0, 239],
          "drift_type": "mean_reverting",
          "baseline": 100,
          "kappa": 0.2
        }},
        {{
          "time_range": [240, 260],
          "drift_type": "sinusoidal",
          "baseline": 100,
          "A": 90,
          "omega": 0.0172,
          "phi": -2.557,
          "kappa": 0.35
        }}
      ],
      "description": "Node with time-varying drift"
    }},
    "1": {{
      "group": "propagation_nodes",
      "description": "Connector highway - pure relay"
    }},
    "2": {{
      "group": "demand_sources",
      "baseline": 80.0,
      "phi": 3.14159,
```

```
      "A": 30.0,
      "omega": 0.2618,
      "description": "Business district - midday peak (phase shifted)"
    }}
  }},
}}

INPUT JSON:
{structured_scenario}

RETURN ONLY VALID JSON (no markdown, no comments).
```

Agent Time-Varying Adjacency Generation Prompt

```
You are Agent 4: Time-Varying Adjacency Generation Agent.

Your task: Generate time-varying modulation rules from structured scenario JSON.

INPUT: Structured JSON from Agent 2 (scenario parsing agent)

OUTPUT: Time modulation configuration as strict JSON (NO markdown, NO comments)

---

TASK SPECIFICATION:

Generate time_modulation: Rules for how edge weights vary over time
- Derived STRICTLY from "adjacency_modulation" field in input JSON
- Each pattern specifies time ranges and edge-specific multipliers

NOTE: Base adjacency matrix is handled separately by the system.
You only need to generate time modulation rules.

TIME MODULATION CONSTRUCTION:
Extract from input JSON "adjacency_modulation.patterns" field:
For each pattern in the input JSON:
1. Extract "time_period" (e.g., "7-9", "25-55", "150-240") and convert to
[start, end]
2. Extract "effect" (strong/moderate) and map to multiplier
3. Extract "applies_to" (e.g., "0->1" or "0->1, 1->2") and parse edges
4. Extract "description"

Effect to Multiplier mapping:
- strong → multiplier: 10-20
- moderate → multiplier: 5-10

Output format (simplified, no daily/seasonal distinction):
{{
  "time_modulation": {{
    "patterns": [
      {{
```

```
        "time_range": [start, end],
        "description": "...",
        "edge_modulations": {{
          "source->target": {{"multiplier": value, "description": "..."}},
          ...
        }}
      }}
    ]
  }}
}}

MULTIPLIER INTERPRETATION:
Final edge weight at time t:
- weight(t) = base_adjacency[i][j] * multiplier(t)
- Base adjacency is handled by the system (you don't generate it)
Multiplier values from effect mapping:
- strong effect: 10-20
- moderate effect: 5-10
- No modulation: 1.0 (edge weight unchanged, default when time is outside all
pattern ranges)

CRITICAL RULES:
1. Time Modulation:
   - Extract patterns from input JSON "adjacency_modulation.patterns" array
   - Do NOT invent new patterns or time ranges
   - Map "effect" field to multiplier: strong=10-20, moderate=5-10
   - Output as unified "patterns" array (no daily/seasonal/weekly distinction)

2. Edge Specification:
   - Format: "source->target" (e.g., "0->1", "1->2")
   - Use "all_edges" if input JSON applies_to = "all_edges"
   - Otherwise, parse input JSON applies_to field (e.g., "0->1, 1->2"
   → separate entries)

3. Time Ranges:
   - Parse "time_period" from input JSON (e.g., "7-9", "25-55", "150-240")
   - Convert to [start, end] integer array
   - No distinction between hourly/daily/seasonal - just numerical ranges

4. Output Format:
   - Valid JSON only (RFC 8259)
   - No markdown code blocks
   - No comments
   - No trailing commas

---

OUTPUT JSON SCHEMA:

{{
```

```
  "time_modulation": {{
    "patterns": [
      {{
        "time_range": [7, 9],
      "description": "Morning rush hour strengthens residential to highway flow",
        "edge_modulations": {{
          "0->1": {{"multiplier": 15, "description": "Strong effect on
          commuter flow"}}
        }}
      }},
      {{
        "time_range": [17, 19],
        "description": "Evening rush hour moderately strengthens highway to
        business flow",
        "edge_modulations": {{
          "1->2": {{"multiplier": 10, "description": "Moderate effect on
          highway flow"}}
        }}
      }}
    ]
  }}
}}

EXAMPLE: [See source code]

INPUT JSON:
{structured_scenario}

RETURN ONLY VALID JSON (no markdown, no comments).
```

Agent Judge Scenario Parsing Prompt

```
You are Judge Agent 1, a meticulous diagnostic expert responsible for
two-level validation.

You will receive:
1. Original Scenario Text: Natural language description from Agent 1
2. Parsed Structured JSON: Structured data from Agent 2

Your mission is to determine if they are consistent, logical, and ready
for simulation. Most importantly, if there is an error, you must diagnose the source:
is it Agent 1's scenario logic or Agent 2's parsing accuracy?

DIAGNOSTIC PROCESS (FOLLOW THIS ORDER):
STEP 1: PARSING FIDELITY ASSESSMENT (Evaluating Agent 2)
Assume the Original Scenario Text is correct. Compare the Parsed JSON against
it meticulously.
Check for:
1. Node Count Accuracy: Does the JSON contain exactly {expected_num_nodes} nodes
as required?
2. Entity Completeness: Are all nodes and edges from the text present in JSON?
```

```
3. Type Accuracy: Are node types (demand_source/propagation) correctly assigned?
4. Attribute Accuracy:
   - Are all edge relationships correctly represented?
   - Are time_lag values correctly extracted as integers?
5. Value Extraction:
   - Time span and sampling frequency correctly extracted?
   - Baseline, amplitude, and peak values match the text?
   - Propagated variations correctly parsed with source nodes and timings?
6. Structure Completeness:
   - Are adjacency_modulation patterns (time periods, effects, edges)
   fully captured?
  - Are drift_patterns accurately representing the temporal evolution described?
If you find ANY discrepancy between the text and JSON, this is Agent 2's error.
- Set `error_source: "agent2"`
- List specific parsing mismatches in `issues`
- Stop here and do NOT proceed to Step 2

STEP 2: SCENARIO LOGIC ASSESSMENT (Evaluating Agent 1)
Only proceed if you are confident the JSON is a FAITHFUL representation of the text.
Now analyze the scenario's internal logic using ONLY the structured JSON:
1. Propagated Event Timing Consistency (CRITICAL):
- Identify event propagation chains in adjacency_modulation
(e.g., edges forming a path like 0->1->3->2)
- For each edge in a chain, verify its modulation time_range respects preceding
edges' time_lag
- Calculation Example:
 * Event path: 0 -> 1 -> 3 -> 2
 * Edge (0->1) has time_lag=1, modulation starts at t_start_1=15
 * Event arrives at Node 1 at t_arrival_1 = t_start_1 + time_lag = 15 + 1 = 16
 * Therefore, edge (1->3) modulation MUST start at t >= 16
 * If edge (1->3) modulation starts at t=15, this is IMPOSSIBLE
- This error indicates Agent 1 failed to account for propagation delays
2. Graph-Temporal Consistency:
- Does every propagated_variation have a corresponding incoming edge?
- Does every demand_source node have at least one outgoing edge?
- Are propagated_variation timings consistent with edge time_lags?
- Is the graph connected to ensure that the effects from demand_source nodes can
propagate to all other nodes?
3. Physical Realism:
- Are all baseline values within similar order of magnitude?
- Do demand_source nodes have amplitude > 0 and exactly one self-generated peak?
- Do propagation nodes have amplitude = 0 and peak = null?
4. Cumulative Delay vs Event Duration:
- Calculate total time lag along critical paths
- Compare to the duration of edge_modulation events describing that path
- If cumulative lag >> event duration, the scenario is unrealistic
If you find logical inconsistencies in the scenario design, this is Agent 1's error.
- Set `error_source: "agent1"`
- Provide specific, actionable suggestions for scenario redesign
```

```
OUTPUT FORMAT (STRICT JSON):
You MUST respond with a single JSON object. No markdown blocks, no extra text.
{{
  "approved": boolean,
  "error_source": "agent1" | "agent2" | null,
  "feedback": "Brief one-sentence summary for control loop routing",
  "issues": [
    {{
      "type": "Parsing Fidelity" | "Scenario Logic",
      "field": "specific field or path (e.g., 'edges[2].time_lag',
      'adjacency_modulation.path_0->1->3')",
      "problem": "Detailed description of the issue",
      "suggestion": "Clear, actionable fix for the responsible agent"
    }}
  ],
  "overall_comment": "Comprehensive assessment explaining the decision"
}}
Rules:
- If approved=true, set error_source=null and issues=[]
- If approved=false, error_source MUST be either "agent1" or "agent2"
- Step 1 errors always result in error_source="agent2"
- Step 2 errors always result in error_source="agent1"

Original Scenario Description:
{scenario}

Parsed Structured JSON:
{parsed_json}

Begin your two-step diagnostic analysis now.
```

Agent Judge Parameter Validation Prompt

```
You are a validation agent responsible for checking if the SDE parameters and
generated time series are reasonable and consistent with the scenario.

Scenario Description (from Agent 2):
{structured_scenario}

SDE Parameters Generated:
{sde_parameters}

Time-Varying Adjacency:
{time_varying_adjacency}
{previous_assessment_section}
Your Task:
Analyze the attached time series visualization and assess:
1. Verification of Fixes: If a previous assessment is provided, first verify if the
suggested changes have been implemented and if the previous issues are resolved.
2. Time Series Patterns: Do the visualized curves match the scenario's
```

```
`drift_patterns`? Check for correct transitions between behaviors
(e.g., stable `mean_reverting` or `logistic` trends vs. dynamic `sinusoidal` peaks).
3. Parameter Plausibility: Within each `drift_pattern`, are the SDE parameter
values (kappa, lambda, sigma) reasonable?
4. Baseline Consistency: Is the baseline value consistent across different patterns
for the same node, as described in the scenario?
5. Drift Type Correctness: Does the sequence of assigned drift types in the
parameters match the intended behaviors in the scenario?
6. Coupling Effects: Are the time-varying coupling strengths (edge weights) from
the adjacency matrix correctly reflected in the simulation? For example, during a
'strong' modulation period, is there a visible and significant influence between
the connected nodes?
7. Dependency Flow: Do propagation nodes show clear dependency on the demand source
nodes they are connected to?
8. Simulation Stability: Are there any unrealistic behaviors like explosive growth,
flatlining, or excessive, non-physical oscillations?
9. Noise Level (Sigma): Is the level of noise (random fluctuations) appropriate for
the scenario? If the noise is so large that it completely hides the underlying trend
(e.g., the sinusoidal peak), suggest reducing `sigma`. If the curve looks too smooth
and artificial, suggest increasing `sigma`.

CRITICAL CONSTRAINTS FOR ADJACENCY ISSUES:
- You can give suggestions to increase or decrease the multiplier within the range of
10-20 for 'strong' and 5-10 for 'moderate'.
- If a propagated peak or pattern is not obvious, suggest INCREASING the `multiplier`
for the relevant edge. If a curve is excessively unstable or over-reacts to another
node, suggest DECREASING the `multiplier`.

Response Format:
Provide your assessment in the following JSON format. Focus ONLY on suggesting
specific parameter changes, do not give implementation advice.
```json
{{
  "approved": true/false,
  "parameter_issues": [
    {{
      "node_id": "node identifier",
      "parameter": "parameter name (e.g., 'kappa', 'baseline', 'drift_type')",
      "current_value": "current value",
      "problem": "description of the problem",
      "suggested_value": "suggested new value or range"
    }}
  ],
  "adjacency_issues": [
    {{
      "edge": "edge identifier (e.g., '0->1')",
      "problem": "description of the problem",
      "suggestion": "how to fix it"
    }}
  ],
```
```

~~~~
  "visual_assessment": "analysis of the time series patterns and their consistency
   with the scenario",
   "overall_comment": "overall assessment and guidance for revision"
}}
```

If everything looks good, set "approved": true and keep issue lists empty.
If there are problems, set "approved": false and provide detailed suggestions.
~~~~

L Spatial Effect Prompt Template

Spatial Effect Prompt

```
Does this response correctly reason with spatial/graph structure?

YES requires BOTH:
1. References graph structure (edges, connections, node relationships)
2. Uses it to reason about the answer (e.g., "A influences B because of edge A→B",
"propagation follows the graph", "connected nodes show similar patterns")

NO if:
- Only describes the graph without reasoning with it
- Only analyzes nodes independently
- Mentions structure but reasoning is incorrect or irrelevant

Question: {question}
Response: {response}

Label: yes or no
```

Listing 1: Scenario parsing Agent


```
{
  "time_span": "1 day",
  "sampling_frequency": "30 minutes",
  "seq_len": 48,
  "variable": "traffic flow (vehicles/hour)",
  "nodes": [
    {
      "id": 0,
      "type": "demand_source",
      "name": "Central Business District",
      "description": "Central business district with offices and commercial areas"
    },
    {
      "id": 1,
      "type": "propagation",
      "name": "Highway Junction",
      "description": "Major highway junction connecting business district to
          residential area"
    },
    {
      "id": 2,
      "type": "demand_source",
      "name": "Residential Suburb",
      "description": "Large residential suburb area"
    }
  ],
```

```
  "edges": [
    {
      "source": 0,
      "target": 1,
      "relationship": "Primary outbound route from business district",
      "time_lag": 1
    },
    {
      "source": 1,
      "target": 2,
      "relationship": "Main highway section leading to residential area",
      "time_lag": 1
    },
    {
      "source": 2,
      "target": 1,
      "relationship": "Inbound commuter route from residential area",
      "time_lag": 1
    },
    {
      "source": 1,
      "target": 0,
      "relationship": "Primary inbound route to business district",
      "time_lag": 1
    }
  ],
  "drift_patterns": {
    "repeat": true,
    "repeat_period": 48,
    "nodes": [
      {
        "id": 0,
        "patterns": [
          { "time_range": [0,13], "behavior": "mean_reverting", "baseline": 100, "amplitude": 0, "peak": null },
          { "time_range": [14,34], "behavior": "sinusoidal", "baseline": 100, "amplitude": 300, "peak": 17 },
          { "time_range": [35,48], "behavior": "mean_reverting", "baseline": 100, "amplitude": 0, "peak": null }
        ],
        "propagated_variations": [
          { "time": "16-20", "origin": "propagated", "source": 2,
            "description": "Receives morning commute flow from residential area via highway junction" }
        ]
      },
      {
        "id": 1,
        "patterns": [
          { "time_range": [0,48], "behavior": "mean_reverting", "baseline": 120, "amplitude": 0, "peak": null }
        ],
        "propagated_variations": [
          { "time": "18-22", "origin": "propagated", "source": 0,
            "description": "Receives evening exodus from business district" },
          { "time": "15-19", "origin": "propagated", "source": 2,
            "description": "Receives morning commute from residential area" }
        ]
      },
      {
        "id": 2,
        "patterns": [
          { "time_range": [0,13], "behavior": "mean_reverting", "baseline": 110, "amplitude": 0, "peak": null },
          { "time_range": [14,34], "behavior": "sinusoidal", "baseline": 110, "amplitude": 280, "peak": 14 },
          { "time_range": [35,48], "behavior": "mean_reverting", "baseline": 110, "amplitude": 0, "peak": null }
        ],
        "propagated_variations": [
```

```
          { "time": "19-23", "origin": "propagated", "source": 0,
            "description": "Receives evening exodus flow from business district via highway junction" }
        ]
      }
    ]
  },
  "adjacency_modulation": {
    "patterns": [
      { "time_period": "14-19", "effect": "strong", "applies_to": ["2->1","1->0"],
        "description": "Morning rush hour intensifies inbound traffic flow" },
      { "time_period": "17-22", "effect": "strong", "applies_to": ["0->1","1->2"],
        "description": "Evening rush hour intensifies outbound traffic flow" }
    ]
  },
  "spatial_layout": {
    "0": { "x": 0, "y": 0 },
    "1": { "x": 1, "y": 0 },
    "2": { "x": 2, "y": 0 }
  },
  "domain": "Transportation",
  "task_id": "task_0030"
}
```

Listing 2: SDEs Parameters Agent

```
{
  "global_defaults": {
    "drift_type": "mean_reverting",
    "node_type": "demand_source",
    "kappa": 0.25,
    "baseline": 50.0,
    "lambda": 1.0,
    "sigma": 0.5,
    "diffusion_shape": "sqrt"
  },
  "group_params": {
    "demand_sources": {
      "node_type": "demand_source",
      "drift_type": "sinusoidal",
      "baseline": 100.0,
      "A": 30.0,
      "omega": 0.1309,
      "phi": 0.0,
      "kappa": 0.25,
      "lambda": 0.9,
      "sigma": 0.5
    },
    "propagation_nodes": {
      "node_type": "propagation",
      "drift_type": "mean_reverting",
      "baseline": 120.0,
      "kappa": 0.25,
      "lambda": 0.8,
      "sigma": 0.3
    }
  },
  "node_overrides": {
    "0": {
      "group": "demand_sources",
      "drift_patterns": [
        { "time_range": [0,13], "drift_type": "mean_reverting", "baseline": 100, "kappa": 0.25, "sigma": 0.5 },
        { "time_range": [14,34], "drift_type": "sinusoidal", "baseline": 100, "A": 300, "omega": 0.1309, "phi": -1.7262, "kappa": 0.25, "sigma": 0.8 },
        { "time_range": [35,48], "drift_type": "mean_reverting", "baseline": 100, "kappa": 0.25, "sigma": 0.5 }
      ],
      "description": "Central business district with offices and commercial areas"
    },
```

```
    "1": {
      "group": "propagation_nodes",
      "drift_patterns": [
        { "time_range": [0,48], "drift_type": "mean_reverting", "baseline": 120, "
            kappa": 0.25, "lambda": 0.8, "sigma": 0.3 }
      ],
      "description": "Major highway junction connecting business district to
          residential area"
    },
    "2": {
      "group": "demand_sources",
      "drift_patterns": [
        { "time_range": [0,13], "drift_type": "mean_reverting", "baseline": 110, "
            kappa": 0.25, "sigma": 0.5 },
        { "time_range": [14,34], "drift_type": "sinusoidal", "baseline": 110, "A":
            280, "omega": 0.1309, "phi": -1.2217, "kappa": 0.25, "sigma": 0.8 },
        { "time_range": [35,48], "drift_type": "mean_reverting", "baseline": 110, "
            kappa": 0.25, "sigma": 0.5 }
      ],
      "description": "Large residential suburb area"
    }
  }
}
```

Listing 3: Time-Varying Adj. Agent

```
{
  "time_modulation": {
    "patterns": [
      {
        "time_range": [14,19],
        "description": "Morning rush hour intensifies inbound traffic flow",
        "edge_modulations": {
          "2->1": { "multiplier": 15, "description": "Strong inbound commuter flow"
              },
          "1->0": { "multiplier": 15, "description": "Strong inbound business
              district flow" }
        }
      },
      {
        "time_range": [17,22],
        "description": "Evening rush hour intensifies outbound traffic flow",
        "edge_modulations": {
          "0->1": { "multiplier": 15, "description": "Strong outbound business
              district flow" },
          "1->2": { "multiplier": 15, "description": "Strong outbound residential
              flow" }
        }
      }
    ]
  },
  "base_adjacency": [
    [0.0, 0.1, 0.0],
    [0.1, 0.0, 0.1],
    [0.0, 0.1, 0.0]
  ]
}
```