ISOMORPH: A SUPPLY CHAIN DIGITAL TWIN FOR SIMULATION, DATASET GENERATION, AND FORECASTING BENCHMARKS

Anonymous authors
Paper under double-blind review

ABSTRACT

Open time-series forecasting (TSF) benchmarks cover retail, energy, weather, and traffic, but supply-chain logistics remains underserved. We introduce ISOMORPH, the first public digital twin of a multi-echelon logistics network with fully interpretable, user-configurable parameters and modular topology, demand process, and control rules. The simulator advances a directed routing graph in discrete time: demand arrives at the destination, is served from stock or recorded as backlog, and triggers replenishment through the network. By construction, the state vector tracks per-node on-hand inventory together with outstanding orders, in-transit shipments, and a smoothed demand estimate across the network, so the dynamics close as a Markov chain on a high-dimensional but tractable state space whose transition kernel acts linearly on the empirical distribution of the state. The released data reproduces the bullwhip effect—the standard empirical signature of supply-chain dynamics—at magnitudes consistent with industry evidence, and three conservation laws structurally encoded in the Markov chain serve as verification tools when users extend the simulator with their own control rules. We release datasets at two catalogue scales (C=50 and C=200) with a 33-rollout scenario library at C=50, exhibiting dynamics largely absent from fixed TSF benchmarks: variance amplification, cascading bottlenecks, regime shifts, and cross-channel coupling through shared macro shocks. Zero-shot evaluation of three foundation models (Chronos, Moirai, TimesFM) against three in-domain-trained baselines (ARIMA, ETS, PatchTST) spans four target observables: demand and three network-dependent quantities: backlog, fill rate, and edge utilization. Comparison with ETTh1, Electricity, and Weather under a common protocol shows that the ISOMORPH targets introduce forecasting regimes that differ from those represented in standard real-world TSF benchmarks. This positions ISOMORPH as a complementary, regenerable logistics-domain benchmark with explicit network-state targets and controlled variation in the underlying data-generating system. The same pairing produces forecast confidence bands across an ensemble of scenario configurations, providing a form of forward UQ from parameter uncertainty. This demonstrates how ISOMORPH can serve as a configurable digital-twin testbed for evaluating and developing foundation models' uncertainty quantification under controlled variation in the underlying system configuration. Code (MIT) is released as anonymized supplementary material.

1 INTRODUCTION

Public time-series forecasting (TSF) benchmarks cover retail, energy, weather, and traffic fairly extensively. However, supply chain logistics, one of the most critical applied forecasting domains, remains underserved: no public benchmark pairs demand series with the network state that produces them. The closest existing resources address the domain in a limited manner. SupplyGraph (Wasi et al., 2024) provides eight months of per-product demand with a static product relationship graph from a single company. However, it has no logistics network characteristics that include inventory state, transportation edges, or capacities. Deepbullwhip (Arief, 2026) simulates a four-stage serial chain to study how demand swings amplify upstream, but releases no supply chain forecasting dataset.

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

M5 (Makridakis et al., 2022) aggregates retail demand hierarchically by store and category. It records what customers bought, without the warehouses, shipments, or inventory behind those sales. Across all three, the same piece is missing: demand series paired with the topology, capacities, and control decisions that determine whether the network can deliver.

Therefore, to close this gap, one needs a different kind of resource. A real-world dataset, even from a willing operator, observes one realization of a logistics network. Capacities, routing rules, and control decisions sit implicitly inside the recorded series; they cannot be inspected, compared against, modified, or replayed under alternative settings. A simulator-based dataset makes those rules explicit: the network, the demand process, and the control policy are part of the deliverable, and the dataset is one realization of a configuration that other users can reproduce, rescale, or modify. Adjacent domains have used this perspective. BuildingsBench (Emami et al., 2023) pairs an EnergyPlus simulator with a curated load-forecasting benchmark for buildings, the most closely related work. No equivalent exists for supply-chain logistics.

This paper introduces ISOMORPH (***I**nterpretable **S**upply-chain **O**perations: a **M**arkov **O**perator for **R**outing and **P**acking in **H**igh dimensions*), a digital twin of a multi echelon logistics network. The user supplies a directed graph of source, intermediate, and destination nodes with edge transit times and per edge container capacities; the simulator runs the network forward at a user chosen time resolution, with shipments routed by Dijkstra, packed greedily into containers, and replenished under an (s, S) policy—each stocking location orders up to a target level S as soon as its on-hand inventory falls to or below a reorder point s (§B.1.3). The simulator's parameters fall into two groups: *structural parameters* (topology, catalogue size C, time resolution, and horizon), which fix what dataset is being generated; and *scenario knobs* (demand structure, inventory, edge transport), which govern how the network operates within that structure. We release a family of datasets generated from one set of structural parameters—a 13-node U.S. network at daily resolution (three sources, nine warehouses across five tiers, and a destination served by two last-mile edges) at two cardinalities (C=50 and C=200)—together with rollouts from perturbed scenario knobs. The simulator is fully open-source and modular: any user can swap the topology, time resolution, demand process, or control rules to generate releases tailored to their own setting. Each release includes the full state trajectory of the network, covering per item demand and on hand inventory as well as per edge shipments and utilization, so quantities like per edge saturation events and destination fill rate drops can be recovered directly from the released data without rerunning the simulator.

A central challenge in simulating supply-chain dynamics is that the system mixes continuously evolving variables (inventory levels, smoothed demand) with abrupt regime changes (demand shocks, capacity bottlenecks, replenishment triggers), producing nonlinear, history-dependent behavior. Without the right state-space formulation, the Markov property fails and conservation guarantees become unverifiable. Our approach is to formulate the network as a Markov chain on the right state space: unlike existing resources that are limited to per-node demand or inventory (Wasi et al., 2024; Naik et al., 2025), the state vector carries outstanding orders, in-transit shipments, and a smoothed demand estimate alongside per-node inventory. The resulting state dimension is at least $C(3|\mathcal{N}| + 1)$ (for $|\mathcal{N}|$ nodes and catalogue size C). This closes the dynamics to be Markovian with a transition kernel that acts linearly on the state distribution, and structurally encodes three conservation laws—the discrete realization of coupled transport and queueing equations—directly into the transition function (§3), enabling tractable simulation over arbitrary horizons, topologies, catalogue sizes, and scenario knobs. The full ISOMORPH release provides datasets at catalogue sizes $C = 50$ and $C = 200$, corresponding to more than $2{,}000$ and $8{,}000$ state dimensions, respectively.

To demonstrate ISOMORPH's use as a forecasting benchmark, we evaluate three pretrained TSF foundation models (Chronos, Moirai, TimesFM) in a zero-shot setting, together with three in-domain-trained baselines (ARIMA, ETS, PatchTST) on four targets: demand, and three network-state quantities that the released trajectories expose—backlog at the destination, fill rate, and edge utilization. The first three are 50-channel item-level series, the last a 16-channel edge-level series (§5.1). Forecasting performance varies substantially across these targets (Table 6).

Against ETTh1, Electricity, and Weather under the same protocol, the ISOMORPH targets show higher normalized forecast errors for several network-state quantities while edge utilization remains comparable to ETTh1 (Table 2). These results indicate that ISOMORPH provides forecasting regimes that differ from those represented in standard real-world TSF benchmarks. This complements

existing benchmark suites by extending their coverage to a regenerable logistics domain with explicit network-state targets and controlled variation in the underlying data-generating system.

Contributions

1. **Simulator.** We introduce ISOMORPH (§2), the first public digital twin of a multi-echelon logistics network designed for TSF benchmarking, with every parameter— demand, inventory, transport, routing, and control — physically interpretable and user-configurable. The simulator is open-source and modular: topology, time resolution, demand process, replenishment policy, and routing rule are swappable without modifying the simulator code; and captures the day-to-day operational logic of large supply chain entities. Every parameter is documented with defaults in App. G.
2. **Empirical and structural validation.** We validate the released data on two complementary axes. *Empirically*, the data reproduce the bullwhip effect at amplification levels within the range reported by Cachon et al. (2007), and the scenario library covers the propagation timing and part of the variability and intermittency statistics of a public single-company dataset (App. K). The comparison is at the level of output statistics, since the inputs of real networks are not publicly observable. *Structurally*, three pathwise conservation laws – per-node mass, global mass, and backlog conservation (Proposition C.1) – are encoded in the transition function by the right state space design. When users add their own control rules to the simulator, these conservation laws—or adapted versions—can verify fidelity of the modified implementation.
3. **Released datasets with complex supply-chain dynamics.** We release two baseline datasets generated by ISOMORPH (§4) at cardinalities $C{=}50$ and $C{=}200$ ($T{=}7{,}300$ time units at daily resolution, i.e. 20 years, with chronological $70/15/15$ splits and loaders) on a 13-node U.S. network, together with a 33-rollout scenario library at $C{=}50$ built around that baseline, of which 29 form the forecasting evaluation cohort and 20 the forward-UQ ensemble. The generated trajectories exhibit coupled supply-chain dynamics, including variance amplification along the network (bullwhip), congestion that concentrates on individual edges as capacity is tightened, long-horizon regime shifts in demand, and cross-channel coupling through a shared macro shock $G(t)$ that simultaneously affects all items and propagates through the network.
4. **Forward UQ on supply-chain dynamics.** The pairing of ISOMORPH with zero-shot foundation models enables forward uncertainty quantification (UQ) under perturbations that standard TSF datasets cannot expose: *parameter uncertainty* (numerical values of scenario knobs such as demand-shock intensities and drift coefficients) and *model uncertainty* (structural choices such as distributional families and replenishment policies). We instantiate the first, performing forward UQ over an ensemble of $K{=}20$ scenario configurations drawn from the cohort (§5.2) and assessing the resulting predictive intervals with empirical coverage and weighted CRPS on all four targets. This establishes ISOMORPH as a configurable digital-twin testbed for evaluating and developing foundation models' uncertainty quantification under controlled variation in the underlying system configuration.

Related work on TSF benchmarks, supply-chain simulators and datasets, and simulator-based ML benchmarks is reviewed in App. A.

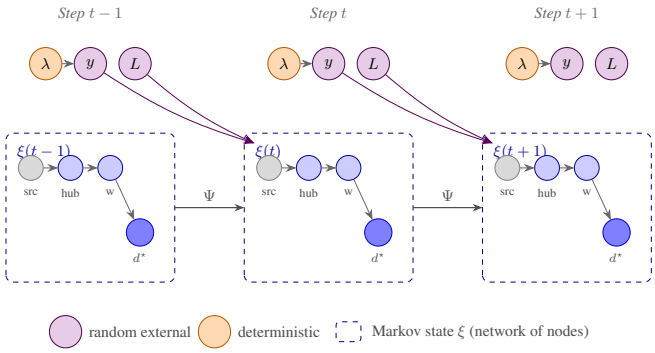
2 THE ISOMORPH DIGITAL TWIN

2.1 OVERVIEW

ISOMORPH simulates the flow of *multiple item types* (a catalogue of C goods) through *a directed network* of factories, intermediate warehouses, and a customer-facing destination, advancing the state of every location and link forward in *discrete time* at user-prescribed time resolution (day/hour/minute). In a single rollout, random customer demand arrives at the destination each step, is served from available stock, and triggers replenishment through the network under an (s, S) policy—order up to level S when on-hand inventory reaches the reorder point s (§B.1.3)—and the cycle repeats. At the end of each step t the simulator collects the state of the entire network—stock levels at each location for each of the C items, unfilled orders, in-transit shipments, and a smoothed demand estimate—into a single vector ξ_t (formal definition in §B.1.1, Eq. 1).



(a) Released network



(b) Markov chain dynamics across three time slices

Figure 1: The ISOMORPH model. **(a)** Released network: 13 U.S. cities (3 sources, 9 intermediate warehouses across 5 tiers, and destination NewYork) connected by 16 directed edges, each carrying a user-set transit time τ_e; the map shows geographic placement, but the graph structure and routing are defined by the edges and their transit times, not by geographic distance. **(b)** Dynamic Bayesian network across three time slices. Top row: random inputs—demand y_t (violet, drawn from deterministic intensity λ_t, orange) and source lead times L_t (violet). Bottom row (dashed boxes): Markov state ξ_t transitioned by Ψ (Algorithm 1), see also Fig. 2 Each slice shows a representative subset of the $|\mathcal{N}| = 13$ nodes (cities).

The network is a user-supplied directed graph $\mathcal{G} = (\mathcal{N}, \mathcal{E})$ whose locations are embedded in two-dimensional geography (e.g. U.S. cities); however, the metric on this graph is defined by user-set integer transit times τ_e on each link, not by geographic distance. Locations split into three roles: factories $\mathcal{N}_{\text{src}}$ with unbounded upstream supply, intermediate warehouses $\mathcal{N}_{\text{int}}$, and a customer-facing destination $d^\star$. Each directed link $e = (u, v)$ carries, in addition to τ_e, a per-container volume V_e and a per-step container count K_e, giving volume capacity $K_e V_e$ per step. Dijkstra's algorithm routes shipments on the weight $\tau_e/(K_e V_e)$, so that capacity enters path selection alongside transit time, and the user controls both the topology and the effective distances by choosing which locations to connect and what τ_e, K_e and V_e to assign (Figure 1(a); released values in Appendix G.3, Table 11).

The dynamics is modeled as a *Markov chain*: the sequence of state variables $\{\xi_t\}$ satisfies $\xi_{t+1} = \Psi(\xi_t, y_t, L_t)$, where Ψ is a deterministic transition function driven by two random inputs—exogenous Poisson customer demand y_t and Gaussian factory lead times L_t, the latter drawn conditioned on ξ_t at source locations whose stock falls below a reorder threshold. The demand y_t is generated from a five-component intensity combining seasonality, drift, bursts, and a shared macro shock (§B.1.2). Figure 1(b) shows the Markov chain unrolled over three time slices. The physical rules that constitute Ψ are organized into two groups: replenishment and service rules (§B.1.3), which govern when each location reorders stock and how the destination serves or backlogs demand; and inter-warehouse flows (§B.1.4), which specify source arrivals, Dijkstra-based dispatch, and inter-warehouse pulls, see Figure 2. The state space, the random inputs, the transition map Ψ, and the resulting transition kernel are specified in §B.1.1 (Eqs. 9–10).

3 The Right State Space and Its Consequences

3.1 Markov Property

Without the auxiliary components Out_t, IT_t, and $\tilde{\lambda}_t$ in the state, next-step inventory would depend on a window of past dispatches and lead-time draws, not on ξ_t alone: the dynamics would require history and the Markov property would fail. By including them, the system closes into a Markov chain $\xi_{t+1} = \Psi(\xi_t, y_t, L_t)$ whose transition probability acts linearly on the empirical distribution of the state (§B.1.1), i.e. the right space for scalable simulation, forward UQ, and downstream stochastic control. The resulting state representation contains $C(3|\mathcal{N}| + 1)$ fixed scalar dimensions (for $|\mathcal{N}|$ nodes and catalogue size C), together with a variable-length in-transit list IT_t. In this high-dimensional space of

$$\text{dimension} \geq C(3|\mathcal{N}| + 1),$$

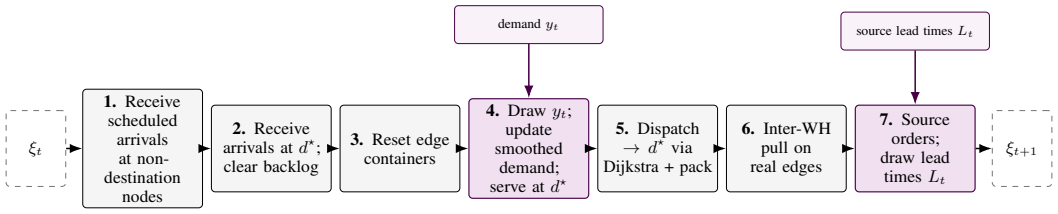


Figure 2: One-step transition map $\Psi : \xi_t \mapsto \xi_{t+1}$ as a fixed sequence of seven sub-steps (Algorithm 1). Random external inputs (violet, top) enter at exactly two sub-steps—demand y_t at sub-step (4) and source lead times L_t at sub-step (7); the remaining five sub-steps are deterministic bookkeeping given (ξ_t, y_t, L_t). Sub-steps (1) and (2) integrate inbound arrivals due at t, (3) replenishes per-step edge capacity, (4) consumes the demand draw and updates the destination's stock and backlog, (5) and (6) move units along edges via Dijkstra routing and greedy first-fit packing, and (7) places fresh source orders with stochastic lead times.

the dynamics become Markovian: these components contain all information required to determine the next step's state without reliance on historical trajectories, enabling efficient simulation of the digital twin over arbitrary horizons. The full ISOMORPH release provides datasets at catalogue sizes $C = 50$ and $C = 200$, corresponding to more than $2{,}000$ and $8{,}000$ fixed scalar state dimensions, respectively.

3.2 Discrete conservation laws

The same state completeness that enables the Markov property also reveals conservation structure structurally encoded in the transition function Ψ. The network-internal stock $I_{i,t}$ in (26) sums over on-hand, in-transit, and scheduled-arrival components—all present in ξ_t by design—so the conservation laws of Proposition C.1 are expressible as closed-form functions of the current state. Proposition C.1 states these three identities (per-node mass, global mass, and backlog conservation); its proof and the continuous-time fluid limit are given in App. C.

4 Released datasets

4.1 What the user configures and what the simulator records

The released datasets are configured through two kinds of parameters. *Structural parameters* are fixed within a release and define what dataset is being generated: the graph $\mathcal{G}$, the catalogue size C, and the time horizon T. *Scenario knobs* govern the operating regime of the network and fall into three categories: demand structure, inventory, and edge transport. The complete parameter tables and their mechanisms within the simulator are in App. G.

Structural parameters. All releases in this paper use the following configuration:

- *Graph*: the 13-node U.S. network of Figure 1(a)—three sources {San Francisco, St. Louis, Orlando}, nine intermediate warehouses organized in five tiers (hub, three mid-level tiers, and last-mile), and destination NewYork. All three sources converge at Nashville and fan out through Atlanta; every shipment to NewYork passes through one of two last-mile edges (Philadelphia→NewYork or Baltimore→NewYork).
- *Cardinality*: two regimes, C=50 and C=200. The two regimes share the same demand parameters, reorder thresholds (s, S), routing, and packing rules; the edge capacities differ. Upstream edge container volumes scale linearly with C, so they are four times larger at C=200 when the per-step container count K_e is held fixed. The two last-mile edges are sized by back-solving the realized mean intensity (§G.4). The first 50 items at C=200 coincide with the entire C=50 release under the same seed, so C=50 is suitable for faster model development with fewer item channels while C=200 tests scalability to a larger catalogue.
- *Horizon*: T=7,300 time units (20 years at daily resolution), comparable to daily-resolution TSF benchmarks such as Exchange (T=7,588).

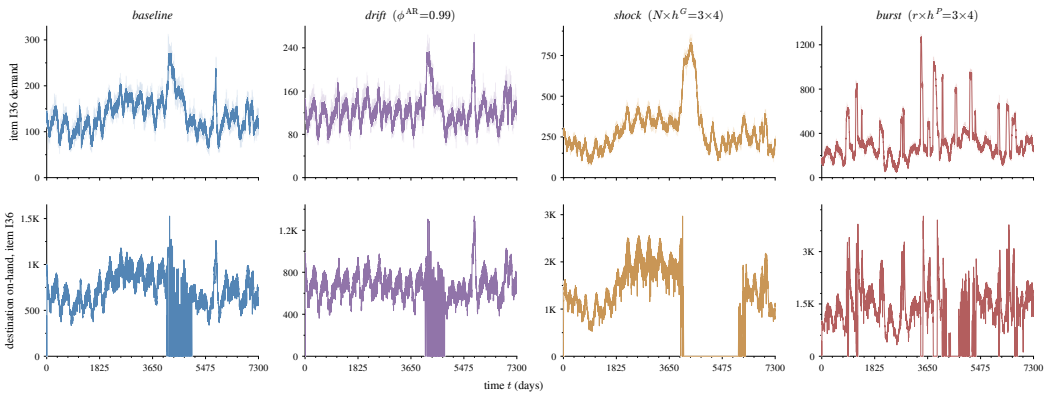


Figure 3: Diversity of dynamics generated by the simulator for a single item (I36) across four demand-side scenarios, over the full released horizon (T=7,300 daily steps; x-axis in days). **Top row:** realized per-item demand $y_{i,t}$ (light) with the deterministic intensity $\lambda_{i,t}$ of Eq. (14) overlaid (solid). **Bottom row:** destination on-hand stock for the same item, i.e. the network's response to that demand. Columns are baseline, drift (ϕ^{AR}=0.99), shock ($N\times h^G$=3×4), and burst ($r\times h^P$=3×4). The four columns differ in both the demand structure and the resulting stockout signature: shock produces one sustained network-wide surge (peak intensity roughly 2.5× baseline) followed by an extended depletion window, burst produces repeated sharp per-item spikes and correspondingly frequent stock crashes, and drift compresses the demand range relative to baseline. Note the differing y-axis scales across columns.

Scenario knobs, output sequences, and forecast targets are detailed in App. D.

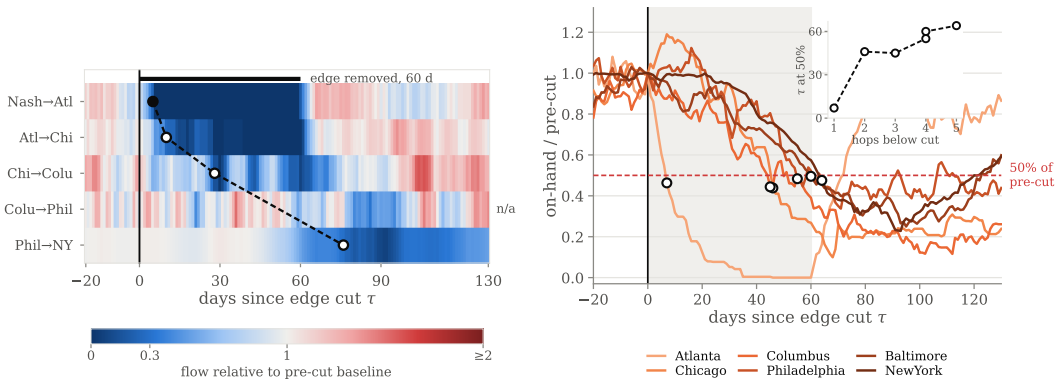
4.2 Reproducing known supply-chain behaviors with Isomorph

Bullwhip effect. The bullwhip effect describes the amplification of demand variability as material moves upstream from the destination toward the source. We compute the corresponding amplification ratio at each non-source node and summarize the results by tier in Table 1, averaged over the 29-rollout cohort of §5.1. The five upstream tiers (1.07–1.45) fall within the $[0.97, 1.90]$ range spanning the 50th–90th percentiles of Cachon et al. (2007), and the destination (1.93 ± 0.41) has a confidence interval that straddles its upper bound. Per-node values are in Table 24 (App. L.2). App. K further compares dimensionless statistics of the scenario library (variability, intermittency, autocorrelation, propagation lag) with SupplyGraph, a public single-company dataset. Propagation timing and all delivery-stage statistics fall within the simulator range; the observed sales series is more intermittent than the Poisson demand of the released baseline, a property of the demand module rather than of the network dynamics. Public datasets record sales and shipments but not the topology, capacities, or ordering rules that produced them, so such comparisons show that observed statistical regimes are reachable; they cannot show that the released network matches a particular operator.

Ripple and cascading disruption. The dynamic edge-cut configuration produces a disruption that propagates through successive echelons of the network (Figure 4). After the Nashville→Atlanta edge is removed, downstream edge flows and node inventories decline progressively with distance from the cut. The propagation is not instantaneous or uniform: transit times and intermediate inventory buffers shape both the timing and magnitude of the response. When the edge is restored, recovery begins but can extend beyond the disruption window. This scenario provides an example of multi-echelon disruption propagation that can be inspected directly from the released network-state trajectories.

Table 1: Bullwhip amplification by tier at C=50, averaged over the 29-rollout cohort. $\bar{B} = \mathrm{Var}(\mathrm{inflow}) / \mathrm{Var}(\mathrm{outflow})$, computed per (node, item) and averaged over items, then over nodes in each tier, then over the 29-rollout cohort of §5.1; the monthly column reports mean $\pm$ 95% Student-t CI over the 29 rollouts. The three source nodes are excluded because they have no inflow on the released edges. The "Daily" column is inflated near the destination because shipments arrive in bursts within a month; the "Monthly" column smooths this out and is the granularity comparable to Cachon et al. (2007)'s industry-level evidence.

Tier	Nodes	Daily $\bar{B}$	Monthly $\bar{B}$
Destination	NewYork	22.64	1.93 ± 0.41
Tier-5 (LM)	Baltimore, Philadelphia	12.68	1.45 ± 0.10
Tier-4	Columbus, Richmond	1.36	1.24 ± 0.06
Tier-3	Charlotte, Chicago, Memphis	1.17	1.22 ± 0.03
Tier-2	Atlanta	1.30	1.14 ± 0.01
Hub	Nashville	0.97	1.07 ± 0.01



(a) Flow on each downstream corridor edge, measured as the 7-day mean flow relative to its own pre-cut 14-day baseline, after removing the Nashville→Atlanta edge for 60 days. The filled circle marks the removed edge, and open circles indicate the first time each downstream edge falls below 30% of its baseline. Flow declines progressively farther downstream, while Columbus→Philadelphia remains largely unaffected because Philadelphia has an alternate Richmond branch.

(b) On-hand inventory at downstream nodes, shown relative to each node's pre-cut level, following the same edge cut. Open circles mark the first time inventory falls below 50% of its pre-cut level. Inventory depletion propagates downstream with increasing graph distance from the cut, and recovery begins after restoration but does not return the destination to its pre-cut level within the 130-day window.

Figure 4: **Ripple and cascading disruption across multiple echelons in ISOMORPH.** The Nashville→Atlanta edge is removed for 60 days in the `ripple_scenario` rollout. (a) Flow along downstream corridor edges, normalized by their pre-cut baselines; the progressive decline illustrates the propagation of the disruption through the network. (b) On-hand inventory relative to the pre-cut level at successive downstream nodes. Recovery begins after the edge is restored, with the timing shaped by network distance and intermediate inventory.

5 COMBINING THE DIGITAL TWIN ISOMORPH WITH TSF FOUNDATION MODELS

5.1 EXPERIMENTAL SETUP: TSF BENCHMARKING

Forecast targets. We evaluate four targets, each a different functional of the network state and each recoverable exactly from every released rollout by the conservation laws of Proposition C.1:

- *Demand* $y_{i,t}$, the per-item demand at the destination. It is generated by Eq. (14) *before* the network acts on it, so it carries no information about the network state and a model that forecasts it well

Table 2: Zero-shot foundation-model MASE across ISOMORPH targets and standard TSF benchmarks. ISOMORPH rows: mean $\pm$ 95% Student-t CI over n=29 rollouts. Benchmark rows: single-run point estimates. Agg. is the geometric mean across horizons. Full per-horizon, per-normalizer tables for each target are Table 5 (demand), Table 29 (backlog), Table 30 (fill rate), and Table 31 (edge utilization).

Model	h=1	h=7	h=14	h=30	Agg.
ISOMORPH *demand (50 items, daily)*					*ours*
Chronos	0.774 ± 0.009	0.843 ± 0.042	0.939 ± 0.090	1.156 ± 0.199	0.910 ± 0.068
Moirai	0.792 ± 0.016	0.867 ± 0.044	0.962 ± 0.086	1.166 ± 0.176	0.932 ± 0.067
TimesFM	0.746 ± 0.014	0.817 ± 0.048	0.915 ± 0.097	1.130 ± 0.205	0.885 ± 0.073
ISOMORPH *edge utilisation (16 edges, daily)*					*ours*
Chronos	0.736 ± 0.026	0.757 ± 0.016	0.758 ± 0.018	0.773 ± 0.019	0.755 ± 0.018
Moirai	0.721 ± 0.024	0.747 ± 0.015	0.747 ± 0.016	0.758 ± 0.018	0.743 ± 0.016
TimesFM	0.812 ± 0.072	0.831 ± 0.108	0.857 ± 0.151	0.844 ± 0.114	0.831 ± 0.097
ISOMORPH *backlog (50 items, daily)*					*ours*
Chronos	2.109 ± 1.164	3.876 ± 2.343	4.860 ± 2.616	6.361 ± 3.946	3.690 ± 2.042
Moirai	1.570 ± 0.910	3.431 ± 2.174	4.566 ± 2.467	6.427 ± 3.988	3.242 ± 1.873
TimesFM	5.157 ± 2.579	6.236 ± 3.229	8.038 ± 4.165	11.914 ± 6.788	7.152 ± 3.740
ISOMORPH *fill rate (50 items, daily)*					*ours*
Chronos	2.902 ± 0.951	3.837 ± 1.204	4.592 ± 1.457	5.578 ± 2.166	3.979 ± 1.266
Moirai	2.308 ± 0.993	2.939 ± 1.249	3.391 ± 1.284	4.185 ± 1.853	2.976 ± 1.251
TimesFM	4.902 ± 2.266	5.979 ± 2.877	7.192 ± 3.878	9.066 ± 5.558	6.457 ± 3.399
ETTh1 (7 channels, hourly)					
Chronos	0.519	0.694	0.814	0.890	0.715
Moirai	0.582	0.792	0.894	0.949	0.791
TimesFM	0.572	0.752	0.848	0.916	0.761
Electricity (321 channels, hourly)					
Chronos	0.508	0.643	0.704	0.767	0.648
Moirai	0.754	0.862	0.931	0.990	0.880
TimesFM	0.548	0.677	0.722	0.762	0.672
Weather (21 channels, 10-min)					
Chronos	0.490	0.559	0.613	0.799	0.605
Moirai	0.448	0.529	0.593	0.812	0.581
TimesFM	0.498	0.469	0.446	0.527	0.484

has learned the intensity, not the logistics. We include it as the point of comparison with existing benchmarks.

- *Backlog* $b_{i,t}$, accumulated unmet demand at the destination (Eq. (22)): a cumulative, integrating quantity.
- *Fill rate*, served over demanded per item: a bounded ratio whose denominator can be small.
- *Edge utilization* $U_{e,t}$, per-step shipped volume over capacity $K_e V_e$: bounded in $[0, 1]$ and capacity-clipped.

The last three are network-dependent targets generated through the multi-echelon structure, making that structure an integral part of the evaluation.

Models. Three pretrained foundation models are run *zero-shot*, chosen to cover the main architectural families: **Chronos-T5** (Ansari et al., 2024) (tokenized values, encoder-decoder T5 backbone, real and KernelSynth pretraining); **Moirai-1.1-R** (Woo et al., 2024) (masked encoder, pretrained on LOTSA); and **TimesFM-2.0** (Das et al., 2024) (decoder-only, trained on Google Trends and Wikipedia pageviews). Three further forecasters are fit *in domain*, per rollout: **ARIMA**, **ETS**, and **PatchTST** (Nie et al., 2023). Including in-domain baselines is what makes the difficulty claim

testable: without them, a high zero-shot error is equally consistent with a hard domain and with an unfamiliar one. The rolling-origin protocol, MASE normalizers, and evaluation cohort are in App. E.

5.2 Experimental setup: zero-shot forward UQ

Standard fixed datasets limit UQ to perturbations of the observed data, such as additive noise, masking, or jitter. Because ISOMORPH exposes the full data-generating process, we can additionally quantify the effect of uncertainty in the underlying system configuration on foundation-model predictions. *Parameter uncertainty* varies the numerical values of scenario knobs (App. G) while keeping the model structure fixed; *model uncertainty* substitutes structural choices—such as the Poisson demand family (Eq. 14, §B.1.2), rectangular pulse shapes for shocks and bursts (Eqs. 19–18), or the (s, S) replenishment policy (§B.1.3)—while keeping knob values fixed. In both cases, the simulator generates a new rollout for each configuration, and the pretrained foundation model is applied zero-shot, allowing us to attribute prediction variability directly to the scenario configurations and produce confidence bands from multiple configurations. The experiments below instantiate parameter uncertainty; model uncertainty is left to future work (App. F). The full forward-UQ protocol and calibration metrics are given in App. E.

5.3 Results

5.3.1 TSF benchmarking

Difficulty relative to standard TSF benchmarks. Table 2 compares the four ISOMORPH targets with ETTh1, Electricity, and Weather under a common MASE evaluation, using the $m = 1$ seasonal-naive baseline for ISOMORPH and the corresponding seasonal periods for each benchmark. The ISOMORPH targets exhibit substantially higher MASE values than the standard benchmarks, particularly for backlog and fill rate, indicating that the foundation models capture substantially less of the short-lag predictive structure in ISOMORPH than in the existing benchmarks. Demand also shows consistently higher MASE than the three benchmarks, while edge utilization is closer to the difficulty of ETTh1. Thus, the released ISOMORPH trajectories provide forecasting regimes in which the predictive structure captured by current foundation models is substantially weaker than on commonly used TSF benchmarks.

5.3.2 Forward UQ

Across the four targets, zero-shot calibration is strongest on demand and edge utilization, whereas backlog and fill rate exhibit substantially larger and more horizon-dependent deviations from the nominal coverage (Table 7). These results also motivate fine-tuning foundation models on the released trajectories, particularly for backlog and fill rate, to investigate whether domain adaptation can improve uncertainty quantification under the same forward-UQ protocol.

6 Conclusion and outlook

We presented ISOMORPH, an open-source digital twin for generating multi-echelon, multi-item logistics trajectories on a user-configurable directed network. Supply-chain dynamics are inherently nonlinear and history-dependent, combining persistent demand variation, discrete replenishment and shipping decisions, inventory accumulation, and network-level flow constraints. We close these history-dependent dynamics as a Markov chain by augmenting the state with per-node on-hand inventory, outstanding orders, in-transit shipments, and a smoothed demand estimate. This yields a high-dimensional but tractable state space with a deterministic transition function driven by two random inputs. The resulting transition kernel acts linearly on the state distribution and supports simulation over user-configurable directed networks, while pathwise conservation laws provide trajectory-level verification. The released configuration is compared with real data only through output statistics (bullwhip, propagation lag, variability and intermittency); its structural inputs are not calibrated to a specific network, however users can supply such available data through the corresponding modules. Together, these results establish ISOMORPH as a regenerable logistics-domain benchmark that complements fixed real-world TSF benchmarks, while providing a generative testbed for evaluating and developing foundation models through system-level uncertainty quantification.

Reproducibility statement

The simulator code, evaluation pipeline, and data loaders are available under the MIT License, and the datasets are available under CC-BY-4.0, at https://osf.io/haxqk/overview?view_only=64ccdeabb29a4384bc9f37a42726a9d0; The simulator is fully specified in App. B (state, random inputs, transition map, and Algorithms 1–3); parameter values and scenario definitions are in App. G, the proof of Prop. C.1 in App. C, dataset schemas in App. I, evaluation details in App. M.1, and compute in App. N.

Ethics statement

ISOMORPH generates synthetic data from an explicit simulator and involves no human subjects or personal data. External datasets are used only for aggregate statistical comparison.

AI use statement

In this work, we have not used generative AI tools to generate synthetic data sets, help develop theoretical models or conceptual frameworks, formulate mathematical claims, provide critical ingredients for proving mathematical claims, propose or refine hypotheses, design or provide feedback on research methodology or experiments, implement methods, clean and reformat datasets, support qualitative and thematic data analysis, or interpret results. Assisting with translation does not apply to this work. Additionally, we used generative AI tools for drafting parts of the paper, identifying relevant literature, editing the paper to improve readability, and checking intermediate derivations and the exposition of proofs. We reviewed all AI-assisted content and checked it for correctness, and we take full responsibility for the final content of this paper, including any text, claims, or artifacts produced with the aid of generative AI.

References

Taha Aksu, Gerald Woo, Juncheng Liu, Xu Liu, Chenghao Liu, Silvio Savarese, Caiming Xiong, and Doyen Sahoo. GIFT-Eval: A benchmark for general time series forecasting model evaluation, 2024. URL https://arxiv.org/abs/2410.10393.

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

Mansur M. Arief. Deepbullwhip: An open-source simulation and benchmarking for multi-echelon bullwhip analyses, 2026. URL https://arxiv.org/abs/2604.13478.

Gérard Cachon, Taylor Randall, and Glen Schmidt. In search of the bullwhip effect. *Manufacturing & Service Operations Management*, 9:457–479, 2007.

Hong Chen and David D. Yao. *Fundamentals of Queueing Networks: Performance, Asymptotics, and Optimization*. Stochastic Modelling and Applied Probability. Springer, 2001.

Abhimanyu Das, Weihao Kong, Rajat Sen, and Yichen Zhou. A decoder-only foundation model for time-series forecasting, 2024. URL https://arxiv.org/abs/2310.10688.

Edsger W. Dijkstra. A note on two problems in connexion with graphs. *Numerische Mathematik*, 1 (1):269–271, 1959.

Samuel Dooley, Gurnoor Singh Khurana, Chirag Mohapatra, Siddartha Naidu, and Colin White. ForecastPFN: synthetically-trained zero-shot forecasting. In *Advances in Neural Information Processing Systems*, 2023.

Patrick Emami, Abhijeet Sahu, and Peter Graf. BuildingsBench: A large-scale dataset of 900k buildings and benchmark for short-term load forecasting. In *Thirty-seventh Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2023. URL https://openreview.net/forum?id=c5rqd6PZn6.

Rakshitha Wathsadini Godahewa, Christoph Bergmeir, Geoffrey I. Webb, Rob Hyndman, and Pablo Montero-Manso. Monash time series forecasting archive. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 2)*, 2021. URL https://openreview.net/forum?id=wEc1mgAjU-.

Christian D. Hubbs, Hector D. Perez, Owais Sarwar, Nikolaos V. Sahinidis, Ignacio E. Grossmann, and John M. Wassick. OR-Gym: A reinforcement learning library for operations research problems, 2020. URL https://arxiv.org/abs/2008.06319.

Thomas G. Kurtz. Solutions of ordinary differential equations as limits of pure jump Markov processes. *Journal of Applied Probability*, 7(1):49–58, 1970.

Spyros Makridakis, Evangelos Spiliotis, and Vassilios Assimakopoulos. The M4 competition: 100,000 time series and 61 forecasting methods. *International Journal of Forecasting*, 36(1): 54–74, 2020. URL https://www.sciencedirect.com/science/article/pii/S0169207019301128.

Spyros Makridakis, Evangelos Spiliotis, and Vassilios Assimakopoulos. M5 accuracy competition: Results, findings, and conclusions. *International Journal of Forecasting*, 38(4):1346–1364, 2022. URL https://www.sciencedirect.com/science/article/pii/S0169207021001874.

Avi Mandelbaum and Gennady Pats. State-dependent stochastic networks. Part I: Approximations and applications with continuous diffusion limits. *Annals of Applied Probability*, 8(2):569–646, 1998.

Nachiket N. Naik, Prathamesh Dinesh Joshi, Raj Abhijit Dandekar, Rajat Dandekar, and Sreedath Panat. BULL-ODE: Bullwhip learning with neural odes and universal differential equations under stochastic demand, 2025. URL https://arxiv.org/abs/2509.18105.

Yuqi Nie, Nam H. Nguyen, Phanwadee Sinthong, and Jayant Kalagnanam. A time series is worth 64 words: Long-term forecasting with transformers. In *International Conference on Learning Representations (ICLR)*, 2023.

Herbert Scarf. The optimality of (S, s) policies in the dynamic inventory problem. In Kenneth J. Arrow, Samuel Karlin, and Patrick Suppes (eds.), *Mathematical Methods in the Social Sciences*, pp. 196–202. Stanford University Press, 1960.

Bertnardo Mario Uskono. Global supply chain disruption & resilience, 2026. URL https://www.kaggle.com/datasets/bertnardomariouskono/global-supply-chain-disruption-and-resilience.

Azmine Toushik Wasi, MD Shafikul Islam, and Adipto Raihan Akib. SupplyGraph: A benchmark dataset for supply chain planning using graph neural networks, 2024. URL https://arxiv.org/abs/2401.15299.

D. Watson-Parris, Y. Rao, D. Olivié, Ø. Seland, P. Nowack, G. Camps-Valls, P. Stier, S. Bouabid, M. Dewey, E. Fons, J. Gonzalez, P. Harder, K. Jeggle, J. Lenhardt, P. Manshausen, M. Novitasari, L. Ricard, and C. Roesch. ClimateBench v1.0: A benchmark for data-driven climate projections. *Journal of Advances in Modeling Earth Systems*, 14(10):e2021MS002954, 2022. URL https://agupubs.onlinelibrary.wiley.com/doi/abs/10.1029/2021MS002954.

Ward Whitt. *Stochastic-Process Limits: An Introduction to Stochastic-Process Limits and Their Application to Queues*. Springer Series in Operations Research. Springer, 2002.

Gerald Woo, Chenghao Liu, Akshat Kumar, Caiming Xiong, Silvio Savarese, and Doyen Sahoo. Unified training of universal time series forecasting transformers. In *Forty-first International Conference on Machine Learning*, 2024.

Haoyi Zhou, Shanghang Zhang, Jieqi Peng, Shuai Zhang, Jianxin Li, Hui Xiong, and Wancai Zhang. Informer: Beyond efficient transformer for long sequence time-series forecasting. In *The Thirty-Fifth AAAI Conference on Artificial Intelligence, AAAI 2021, Virtual Conference*, volume 35, pp. 11106–11115. AAAI Press, 2021.

A Related work

A.1 Time-series forecasting benchmarks

Standard TSF benchmarks primarily consist of fixed real-world datasets spanning domains such as electricity, weather, retail, and traffic. Representative suites include **ETT** (Zhou et al., 2021), **M4** and **M5** (Makridakis et al., 2020; 2022), **Monash** (Godahewa et al., 2021), and the recent **GIFT-Eval** (Aksu et al., 2024). While these benchmarks provide broad coverage of forecasting tasks, they expose datasets rather than the underlying data-generating processes, limiting the ability to systematically regenerate series under controlled changes to the generating conditions. Synthetic approaches such as **KernelSynth** (Ansari et al., 2024) and **ForecastPFN** (Dooley et al., 2023) instead enable parameterized generation of time series, providing greater control over the characteristics of the resulting data. However, these approaches use generic statistical or learned generative mechanisms rather than an explicit simulator of a particular operational system. ISOMORPH complements these existing benchmarks by providing a regenerable forecasting benchmark generated from an explicit multi-echelon supply-chain network, with controllable parameters governing the underlying system dynamics.

A.2 Supply-chain simulators and datasets

Publicly available supply-chain time-series datasets remain limited. **SupplyGraph** (Wasi et al., 2024) provides SKU-level observations from a consumer-goods company across four supply-chain stages—Sales Order, Delivery to Distributor, Factory Issue, and Production—covering $C = 41$ product categories. SupplyGraph is a static observational release: its product-association graph encodes shared product groups, sub-groups, plants, or storage locations rather than a transportation network with explicit transit times and capacities, and its underlying data-generating process is not exposed. Consequently, its generating mechanisms cannot be directly compared one-to-one with those of ISOMORPH. Instead, we use SupplyGraph as a real-world reference to assess the statistical coverage of ISOMORPH, comparing observable distributional and temporal properties in App. K. This comparison serves as a sanity check rather than a validation of identical dynamics: it asks whether the scenario library of ISOMORPH can reproduce statistical regimes observed in a real supply chain. At the same time, ISOMORPH is complementary to such observational releases, as its controllable generating process produces both time-series observations and the underlying network-state trajectories and system variables that are unavailable in SupplyGraph.

Open-source supply-chain simulators provide controllable alternatives to observational datasets, but target different simulation and learning objectives. **deepbullwhip** (Arief, 2026) provides a serial-chain simulator and bullwhip benchmark, **OR-Gym** (Hubbs et al., 2020) provides a serial-chain environment for reinforcement-learning-based inventory control, and **BULL-ODE** (Naik et al., 2025) studies demand, inventory, and ordering dynamics in a single-node continuous-time setting. In contrast, ISOMORPH is designed to generate multi-item, multi-echelon network trajectories for time-series forecasting research, with a configurable directed network and user-controlled demand and network parameters. Forecasting targets can be defined at the level of individual items and/or network nodes and edges, enabling forecasting tasks across different levels of the supply-chain system. In our forecasting experiments, demand, backlog, and fill rate are modeled as 50-channel item-level targets, while edge utilization is modeled as a 16-channel edge-level target (Tables 5, 29, 30, and 31). Table 3 compares these frameworks with ISOMORPH in terms of primary use, topology, catalogue size, time model, and structural closure. The comparison highlights that ISOMORPH supports a configurable directed network, multiple catalogue items, user-chosen temporal resolution, and a closed state representation over nodes, edges, and items, whereas the released configurations of the related frameworks consider more restricted network and catalogue settings. Simpler supply-chain configurations can be obtained as reductions of ISOMORPH, allowing direct comparison with the serial-chain, single-item settings of deepbullwhip and OR-Gym. The full structural comparison and the corresponding reductions are given in Appendix J.

A.3 Related digital twins as ML dataset generators

Mechanistic simulators, that is, simulators that advance a system by applying its governing rules step by step, rather than sampling from a fitted or learned law, have been released as public machine-

Table 3: High-level comparison of ISOMORPH with three related open supply-chain simulation and learning frameworks. Entries describe the default or headline released instantiation of each work; the full structural comparison (demand process, ordering and control, lead times, capacity and routing, analytical structure, scenario generation) is Table 21 in App. J, which also reports the full state vector of each system. "Structural closure" asks whether the released state is rich enough to make the dynamics Markovian and represent the system's material-flow balances.

Axis	**ISOMORPH** (this work)	**BULL-ODE** (Naik et al., 2025)	**deepbullwhip** (Arief, 2026)	**OR-Gym** (Hubbs et al., 2020)
Primary use	Simulator + forecasting dataset	Learning study (NODE vs. UDE)	Simulator + bullwhip benchmark	RL environment for inventory control
Topology	**Directed graph** (user-configurable); released 13-node U.S. network with 16 edges	**Single node**	**Serial chain**; v0.3.0 extends to DAGs via `SupplyChainGraph`	**Serial chain**
Catalogue C	Multi-item; released at $\mathbf{C{=}50}$ and $\mathbf{C{=}200}$	$\mathbf{C{=}1}$	$\mathbf{C{=}1}$	$\mathbf{C{=}1}$
Time model	Discrete, user-chosen frequency; released daily with $T{=}7{,}300$	Continuous-time ODE ($\Delta t{=}0.2$); $T{=}30$	Discrete, weekly; $T{\leq}5{,}200$ tested	Discrete, daily; $T{=}30$
Structural closure	Markov over nodes, edges, and items, with routing and packing internal to the transition; supports three pathwise conservation identities (Prop. C.1)	Markov on a single node in continuous time; scalar conservation $\dot{I} = O - D$	Markov over nodes; per-node material balances and a delta-method CV filtering result	Markov over nodes; per-node material balances, no explicit conservation framing

learning benchmarks in other domains. We restrict attention to those whose task is forecasting the future state of an operating system, which is the setting of this paper; simulator-generated benchmarks for surrogate modeling, where the target is a solution operator rather than a trajectory, are a separate line of work. **BuildingsBench** (Emami et al., 2023) couples the EnergyPlus building simulator with a benchmark for short-term electricity load forecasting. **ClimateBench** (Watson-Parris et al., 2022), in contrast, evaluates machine-learning models trained to emulate outputs of CMIP6 climate models, representing a surrogate-modeling task rather than time-series forecasting. ISOMORPH applies this simulator–benchmark paradigm to multi-echelon supply chains, while additionally enabling zero-shot foundation-model evaluation under controlled perturbations of the underlying system parameters. To our knowledge, ISOMORPH is the first framework to provide this combination of controllable multi-echelon logistics simulation, forecasting benchmark generation, and parameter-perturbation evaluation.

B SUPPLEMENTARY MATERIAL FOR SECTION 2: MODEL DETAILS AND ALGORITHMS

The base configuration holds three properties fixed within a run—stationary topology, fixed control policy, and deterministic edge transit (§B.1.3)—and under them three pathwise conservation identities, in two families (mass, and demand events), hold on every sample path; they are stated and proved in §3.2 (Prop. C.1).

Those three assumptions are sufficient for the proof but stronger than the identities require, which matters because App. H relaxes two of them in the released simulator. Inspecting (29)–(31): the two mass identities refer to receipts, dispatches, and in-transit counts alone—never to time-invariance of $\mathcal{G}$ or to the distribution of transit times—and the backlog identity refers to the destination service rule. Accordingly, dynamic edge cut and restore (which relaxes stationary topology and generates the disruption dynamics of §4.2) and stochastic edge transit both leave all three identities intact. What is actually load-bearing is narrower: that no sub-step discards a unit—which is why the finite-warehouse-capacity extension holds excess arrivals on the inbound edge instead of dropping them—and that demand events are not coupled across items, which is why substitution is the one entry in Table 15 marked as breaking (31). Fixed control policy is the one assumption still in force in every released run. We have not re-proved Prop. C.1 under the relaxed assumptions; the closure column of App. H is an argument about the state vector, and the archived extension rollouts satisfy the identities numerically.

The simulator is realized as three algorithms in §B.2. The top-level procedure is Algorithm 1 (§B.2.1), a seven-stage per-step loop that defines Ψ, see Figure 2; it calls two subroutines:

- Algorithm 2 (§B.2.2): greedy first-fit packing of units into finite-capacity containers along a shipping path.
- Algorithm 3 (§B.2.3): sampling of the five-component demand intensity tensor.

Design rationale for the modeled dynamics. The components of ISOMORPH are individually well-known: (s, S) replenishment, which is optimal for the single-location dynamic inventory problem under fixed ordering costs (Scarf, 1960); exponential smoothing for the demand estimate that drives proactive shipping; shortest-path routing via Dijkstra's algorithm (Dijkstra, 1959); and greedy first-fit packing into containers. The design of ISOMORPH lies in their sequential coupling across warehouses and items within a single Markovian formulation on a directed graph (Algorithm 1). This formulation follows a common principle: realistic network behavior is introduced without substantially enlarging the dynamic state, allowing ISOMORPH to retain a compact Markov representation while supporting multi-echelon and multi-item dynamics. The specific structural choices underlying this principle, including capacity and routing, pathwise conservation, lead-time structure, and proactive shipping, are detailed in Appendix J. Additional mechanisms can be treated as scenario-level extensions; App. H discusses their implementation and configuration.

B.1 MATHEMATICAL MODEL

This section specifies the simulator as a mathematical object: the Markov chain structure (§B.1.1), the demand intensity (§B.1.2), the replenishment and service rules (§B.1.3), and the inter-warehouse flows (§B.1.4).

B.1.1 MARKOV CHAIN STRUCTURE

The simulator is a Markov chain on a hybrid state space: a deterministic one-step map Ψ, driven by two random external inputs (exogenous demand and source lead times), induces a transition kernel as the average of Ψ over their joint law, yielding an equivalent Dynamic Bayesian Network (DBN) representation (Figure 1).

Graph and node attributes. Let $\mathcal{G} = (\mathcal{N}, \mathcal{E})$ be a directed graph with $|\mathcal{N}| = 13$. The nodes split into three groups: source nodes $\mathcal{N}_{\text{src}}$, intermediate nodes $\mathcal{N}_{\text{int}}$, and a single destination $d^\star$, see Figure 1. Each edge $e = (u, v) \in \mathcal{E}$ carries a transit time $\tau_e \in \mathbb{Z}_{\geq 1}$ (time units), a per-container volume $V_e \in \mathbb{R}_{>0}$, and a per-step container count $K_e \in \mathbb{Z}_{\geq 0}$. The per-step volume capacity on e is $C_e := K_e V_e$.

In the released instantiation (§4), $\mathcal{N}_{\text{src}} = \{\text{San Francisco}, \text{St. Louis}, \text{Orlando}\}$, $d^\star = \text{NewYork}$, and $\mathcal{N}_{\text{int}}$ contains the nine intermediate warehouses; the per-edge values (τ_e, V_e, K_e) are listed in Table 11 (§G.3).

Items and time. The item set is $\mathcal{I} = \{1, \ldots, C\}$ with per-item unit volumes $v_i \in \mathbb{R}_{>0}$. Time is discrete with one step per user-chosen time unit, $t \in \{0, 1, \ldots, T-1\}$ and horizon T. The released runs use a *daily* step and $T = 7{,}300$ (20 years).

State variables. The state at the end of time unit t is the vector

$$\xi_t = \big(\mathrm{OH}_t,\ B_t,\ \mathrm{Out}_t,\ \mathrm{IT}_t,\ \tilde{\lambda}_t\big) \in \mathcal{X}, \tag{1}$$

with the following five components:

- On-hand inventory $\mathrm{OH}_t = (\mathrm{OH}_t^{n,i})_{n\in\mathcal{N},\, i\in\mathcal{I}} \in \mathbb{Z}_{\geq 0}^{|\mathcal{N}|\cdot C}$, where $\mathrm{OH}_t^{n,i}$ is the number of units of item i held at node n.
- Backlog $B_t = (B_t^{n,i})_{n\in\mathcal{N},\, i\in\mathcal{I}} \in \mathbb{Z}_{\geq 0}^{|\mathcal{N}|\cdot C}$, where $B_t^{n,i}$ is the number of demand units owed by node n for item i, which vanishes at every node other than $d^\star$.
- Outstanding source orders $\mathrm{Out}_t = (\mathrm{Out}_t^{n,i})_{n\in\mathcal{N},\, i\in\mathcal{I}}$, where each $\mathrm{Out}_t^{n,i} \in \{\emptyset\} \cup (\mathbb{Z}_{>0} \times \mathbb{Z}_{>0})$ is either empty or a pair (τ, q) recording q units due to arrive at time unit τ.

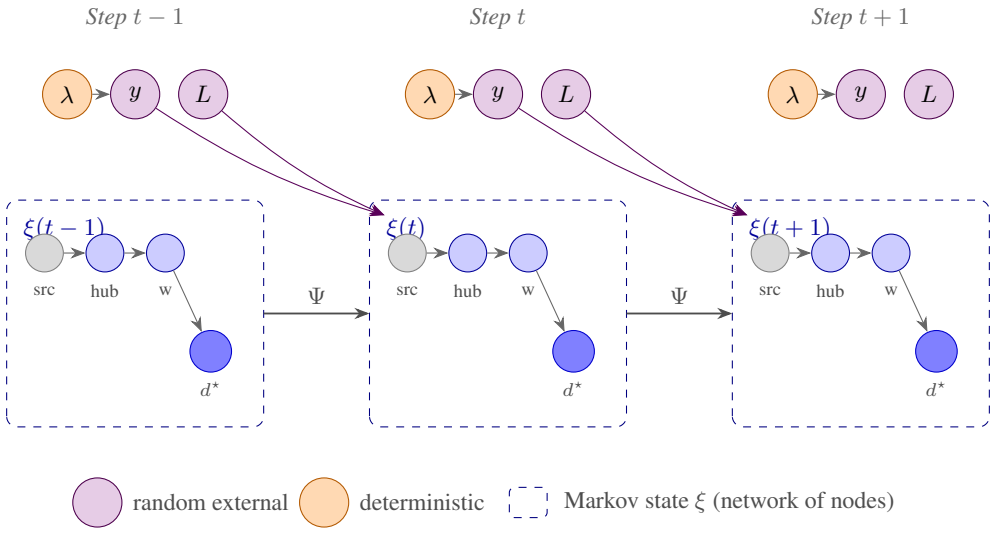


Figure 5: The ISOMORPH model as a dynamic Bayesian network across three time slices. *Random external inputs* (top row, violet): demand $y_t \sim \text{Poisson}(\lambda_{i,t})$ and source lead times L_t are the sources of randomness in the chain, and no other component is random. The intensity λ (orange) is deterministic given the simulator seed. *Markov state* (bottom row, dashed boxes): the state ξ_t is the joint state of the simulator, transitioned by the deterministic map Ψ defined in Algorithm 1. The illustrated nodes inside each slice are organized by the partition $\mathcal{N} = \mathcal{N}_{\text{src}} \sqcup \mathcal{N}_{\text{int}} \sqcup \{d^\star\}$ from §B.1.1: a representative source, two intermediate nodes (a hub and a downstream warehouse), and the destination. We do not draw all 13 cities of the released topology in each slice; the figure shows the role of each group, not the full graph. The geographic instantiation ($\mathcal{N}_{\text{src}} = \{\text{San Francisco, St. Louis, Orlando}\}$, $d^\star = \text{NewYork}$, and the nine intermediate nodes) is in Figure 1 of the main text.

- Scheduled destination arrivals $\text{IT}_t \in \mathcal{M}\big(\mathbb{Z}_{>0} \times \mathcal{I} \times \mathbb{Z}_{>0}\big)$, a finite collection of triples (τ, i, q) recording shipments en route to $d^\star$.
- Smoothed demand estimate $\tilde{\lambda}_t = (\tilde{\lambda}_t^i)_{i \in \mathcal{I}} \in \mathbb{R}_{>0}^C$, an exponentially weighted moving average of recent observed demand, updated by $\tilde{\lambda}_{t+1}^i = \alpha y_{i,t} + (1-\alpha)\tilde{\lambda}_t^i$ with $\alpha = 0.05$.

Hybrid state space. The state ξ_t combines four discrete-valued components and one continuous-valued component. The discrete part lives in

$$\mathcal{Q} = \mathbb{Z}_{\geq 0}^{|\mathcal{N}| \cdot C} \times \mathbb{Z}_{\geq 0}^{|\mathcal{N}| \cdot C} \times \big(\{\emptyset\} \cup \mathbb{Z}_{>0} \times \mathbb{Z}_{>0}\big)^{|\mathcal{N}| \cdot C} \times \text{Lists}\big(\mathbb{Z}_{>0} \times \mathcal{I} \times \mathbb{Z}_{>0}\big),$$

a countable set comprising integer counts for on-hand inventory and backlog ($|\mathcal{N}| \cdot C$ entries each), optional integer pairs for outstanding source orders ($|\mathcal{N}| \cdot C$ entries), and a finite list of triples (τ, i, q) recording shipments scheduled to arrive at $d^\star$ at time unit τ. The continuous part is the smoothed demand vector $\tilde{\lambda}_t \in \mathbb{R}_{>0}^C$, which takes values in a C-dimensional positive-real space. The full state space is the hybrid product

$$\mathcal{X} = \mathcal{Q} \times \mathbb{R}_{>0}^C, \tag{2}$$

on which probability distributions are defined in the standard way for hybrid spaces (the Borel product of the discrete σ-algebra on $\mathcal{Q}$ with the Borel σ-algebra on $\mathbb{R}_{>0}^C$).

Per-edge in-transit volumes (derived quantity). Per-edge in-transit counts are not part of the state variables above. We write $q_{i,t}^{u \to v}$ for the cumulative count of item-i units on edge (u, v) at the end of time unit t, computed as the sum of all dispatches that have been routed onto (u, v) but whose realized arrival time is strictly later than t. This is a derived quantity, reconstructible from the dispatch records produced inside the per-step update (Algorithm 2, §B.2.2); it is referenced in Proposition C.1.

Random external inputs. At each step t, the per-step update reads two random vectors (y_t, L_t) that are external inputs, supplied to the system rather than computed from its current state. They are not part of the state.

Demand. The demand vector $y_t = (y_{1,t}, \ldots, y_{C,t}) \in \mathbb{Z}_{\geq 0}^{C}$ has independent Poisson components,

$$y_{i,t} \sim \text{Poisson}(\lambda_{i,t}), \tag{3}$$

with the draws independent across both i and t. The intensity sequence $(\lambda_{i,t})_{i \in \mathcal{I},\, t \in \{0,\ldots,T-1\}}$ is constructed in §B.1.2 (see (14)) and §B.2.3 (algorithmic construction). For the present subsection, $(\lambda_{i,t})$ is treated as a fixed input to the chain.

Source lead times. Define the set of source-item pairs that need a fresh order on a state ξ as

$$\mathcal{T}(\xi) = \big\{(n,i) \in \mathcal{N}_{\text{src}} \times \mathcal{I} \;:\; \text{OH}^{n,i} < s^{n,i},\ \text{Out}^{n,i} = \emptyset\big\}, \tag{4}$$

where the parameters $(s^{n,i}, \mu^{n,i})$ are specified in §B.1.3. For each $(n,i) \in \mathcal{N}_{\text{src}} \times \mathcal{I}$, define

$$L_t^{n,i} = \max\big(1,\ \lceil X_t^{n,i} \rceil\big), \qquad X_t^{n,i} \sim \mathcal{N}\big(\mu^{n,i},\ (0.2\,\mu^{n,i})^2\big). \tag{5}$$

Writing Φ for the standard-Gaussian cumulative distribution function, the integer $L_t^{n,i}$ takes value ℓ with probability

$$\mathbb{P}\big(L_t^{n,i} = \ell\big) = \begin{cases} \Phi\left(\dfrac{1-\mu^{n,i}}{0.2\,\mu^{n,i}}\right), & \ell = 1, \\ \Phi\left(\dfrac{\ell-\mu^{n,i}}{0.2\,\mu^{n,i}}\right) - \Phi\left(\dfrac{\ell-1-\mu^{n,i}}{0.2\,\mu^{n,i}}\right), & \ell \geq 2. \end{cases} \tag{6}$$

Collect all draws into the lead-time vector

$$L_t = \big(L_t^{n,i}\big)_{(n,i) \in \mathcal{N}_{\text{src}} \times \mathcal{I}} \in \mathbb{Z}_{\geq 1}^{|\mathcal{N}_{\text{src}}| \cdot C}. \tag{7}$$

The transition map Ψ in §B.1.1 reads the components indexed by $\mathcal{T}(\xi_t)$ and no others; the remaining components are extra randomness that has no effect on ξ_{t+1}.

Independence and initialization. Conditional on ξ_t, the lead-time vector L_t has independent components and is independent of y_t. The collection $\big((y_t, L_t)\big)_{t \geq 0}$ is jointly independent across t. The initial state $\xi_0 \in \mathcal{X}$ is fixed in advance, specified by the simulator configuration. Throughout this paper, the simulation seed is treated as a fixed parameter; randomness in the chain is carried entirely by $\big((y_t, L_t)\big)_{t \geq 0}$.

One-step transition map. The transition from ξ_t to ξ_{t+1} has two layers. The first is a deterministic map Ψ that, given the current state and a realized draw of the random external inputs, computes the next state by mechanical bookkeeping. The second is the transition probability P_t obtained by averaging Ψ over the distribution of those inputs. This subsection specifies Ψ; §B.1.1 specifies P_t.

Define Ψ as the composition of the seven sub-step operations of Algorithm 1 (§B.2.1), executed in fixed order, viewed as a map

$$\Psi : \mathcal{X} \times \mathbb{Z}_{\geq 0}^{C} \times \mathbb{Z}_{\geq 1}^{|\mathcal{N}_{\text{src}}| \cdot C} \longrightarrow \mathcal{X}. \tag{8}$$

Given ξ_t and realized inputs (y_t, L_t), the next state is

$$\xi_{t+1} = \Psi(\xi_t,\ y_t,\ L_t) \tag{9}$$

deterministically: there is no further randomness in the transition once (y_t, L_t) are fixed. The seven sub-steps execute in the order specified by Algorithm 1, and the resulting state ξ_{t+1} depends on this ordering. Each sub-step is built from arithmetic operations, indicator functions, and finite enumeration over the finite sets $\mathcal{N}$, $\mathcal{I}$, $\mathcal{E}$. Writing Ψ in closed form would require inlining all seven sub-steps with their (s,S) trigger logic and packing-residual dependencies into one expression; the algorithmic specification of Algorithm 1 is more compact and is the definition of Ψ throughout this paper.

Transition probability. The transition probability defined as

$$P_t(\xi, A) = \mathbb{P}(\xi_{t+1} \in A \mid \xi_t = \xi)$$

is constructed from Ψ in three steps: fix the realized inputs and apply Ψ deterministically, ask whether the result lands in A, then average over the distribution of inputs.

Step 1 — pointwise transition. For $\xi_t = \xi$ and realizations (y, ℓ), the next state is $\xi_{t+1} = \Psi(\xi, y, \ell)$ (§B.1.1).

Step 2 — conditional law of inputs. By §B.1.1, the joint conditional distribution of (y_t, L_t) given $\xi_t = \xi$ factorizes as $\pi_t^{\mathrm{dem}}(y)\,\pi^{\mathrm{lt}}(\ell)$, with neither factor depending on ξ.

Step 3 — average over inputs. For each $t \geq 0$, $\xi \in \mathcal{X}$, and event A,

$$P_t(\xi, A) = \sum_{y \in \mathbb{Z}_{\geq 0}^{C}} \sum_{\ell \in \mathbb{Z}_{\geq 1}^{|\mathcal{N}_{\mathrm{src}}| \cdot C}} \mathbf{1}\{\Psi(\xi, y, \ell) \in A\}\ \pi_t^{\mathrm{dem}}(y)\,\pi^{\mathrm{lt}}(\ell), \tag{10}$$

where

$$\pi_t^{\mathrm{dem}}(y) = \prod_{i=1}^{C} \frac{\lambda_{i,t}^{y_i}\, e^{-\lambda_{i,t}}}{y_i!}, \qquad \pi^{\mathrm{lt}}(\ell) = \prod_{(n,i) \in \mathcal{N}_{\mathrm{src}} \times \mathcal{I}} \mathbb{P}\big(L^{n,i} = \ell^{n,i}\big), \tag{11}$$

and the lead-time mass function is given in (6).

Summing over the entire state space, $P_t(\xi, \mathcal{X}) = \sum_y \pi_t^{\mathrm{dem}}(y) \sum_\ell \pi^{\mathrm{lt}}(\ell) = 1$, since both factors are probability mass functions and Ψ takes values in $\mathcal{X}$. For each fixed ξ, $P_t(\xi, \cdot)$ is a probability distribution over the next state.

The function $P_t : \mathcal{X} \times \mathcal{B}(\mathcal{X}) \to [0, 1]$ is also called a *Markov transition kernel* in the probability literature. We use "transition probability" throughout to mean the same object in less technical language.

Markov property The Markov property compares the conditional distribution of ξ_{t+1} given the entire past against the conditional distribution given the current state alone. To formalize "the entire past," let $\mathcal{F}_t$ be the collection of events whose outcomes are fully determined by the initial state ξ_0 and the random external inputs through time unit $t - 1$:

$$\mathcal{F}_t = \sigma\big(\xi_0,\ y_0,\ \ell_0,\ \ldots,\ y_{t-1},\ \ell_{t-1}\big).$$

Concretely, $\mathcal{F}_t$ is the set of all yes-or-no questions whose answer can be read off from the values of these inputs alone — "did inventory at warehouse w exceed 100 units yesterday?", "has any backlog accumulated at $d^\star$ in the first week?", and so on. Closing this set under the natural operations on events (union, intersection, complement) gives the formal object $\mathcal{F}_t$ (a σ-algebra). Since ξ_t is computed from these inputs by repeated application of Ψ, any event involving ξ_t belongs to $\mathcal{F}_t$.

By construction, (y_t, L_t) is independent of $\mathcal{F}_t$, so back in (10)

$$\mathbb{P}\big(\xi_{t+1} \in A \,\big|\, \mathcal{F}_t\big) = P_t(\xi_t, A) \qquad \text{a.s.} \tag{12}$$

The conditional distribution of ξ_{t+1} given the past coincides with the distribution given ξ_t. The chain $(\xi_t)_{t \geq 0}$ is therefore a time-inhomogeneous Markov chain on $\mathcal{X}$ with transition probabilities $(P_t)_{t \geq 0}$, where time-inhomogeneity comes from the time-varying demand intensity $\lambda_{i,t}$ that enters π_t^{dem}. The augmented process $\big((t, \xi_t)\big)_{t \geq 0}$ on $\mathbb{Z}_{\geq 0} \times \mathcal{X}$ is time-homogeneous, with transition probability

$$\bar{P}\big((t, \xi),\ \{t + 1\} \times A\big) = P_t(\xi, A). \tag{13}$$

The state space $\mathcal{X} = \mathcal{Q} \times \mathbb{R}_{>0}^{C}$ is hybrid: transitions on the continuous component $\tilde{\lambda}$ are deterministic given y_t, while transitions on the discrete component are random.

DBN representation. The chain admits an equivalent representation as a dynamic Bayesian network (DBN). In this representation, the state ξ_t is decomposed into its per-node, per-item components

$$\big(\mathrm{OH}_t^{n,i}, B_t^{n,i}, \mathrm{Out}_t^{n,i}, \mathrm{IT}_t, \tilde{\lambda}_t^{i}\big)_{n \in \mathcal{N},\, i \in \mathcal{I}},$$

and the random external inputs (y_t, L_t) enter as separate input nodes, see also Figure 1. The network topology $\mathcal{G}$ is then embedded in the cross-node dependencies within each time slice: $\mathrm{OH}_{t+1}^{n,i}$ depends on $\mathrm{OH}_t^{u,i}$ for upstream neighbours u via the inter-warehouse flows of §B.1.4. We do not use the DBN representation in the present work — the Markov chain abstraction over ξ_t is sufficient for the formal claims of §3.2 and the algorithmic specification of §B.2. The DBN view could be useful for follow-up work on inference, structure learning, or interventional reasoning.

B.1.2 DEMAND INTENSITY

This section gives the equational form of the demand intensity sequence $(\lambda_{i,t})$ used in §B.1.1 as the rate parameter of the Poisson demand draws.

For each item i and time unit t, demand intensity is

$$\lambda_{i,t} = \bar{\lambda}_i \cdot \big[1 + S_i(t) + W_i(t) + A_i(t) + P_i(t) + g_i\, G(t)\big]_{\geq \varepsilon}, \tag{14}$$

with $[\cdot]_{\geq\varepsilon} := \max(\cdot, \varepsilon)$ and $\varepsilon = 0.08$ (a floor that prevents the intensity from collapsing to zero). The five components are as follows.

Yearly seasonal (with second harmonic, so the shape is not a perfect sine):

$$S_i(t) = a_{1,i}^{(y)}\, \sin(\omega_y t + \phi_i) + a_{2,i}^{(y)}\, \sin(2\omega_y t + 0.7\,\phi_i), \qquad \omega_y = 2\pi/365. \tag{15}$$

Weekly seasonal:

$$W_i(t) = a_i^{(w)}\, \sin(\omega_w t + \psi_i), \qquad \omega_w = 2\pi/7. \tag{16}$$

Long-horizon drift, autoregressive of order 1 (AR(1)) and clipped:

$$A_i(t) = \mathrm{clip}\big(\phi_i^{\mathrm{AR}}\, A_i(t-1) + \epsilon_{i,t},\ \pm 0.6\big), \qquad \epsilon_{i,t} \sim \mathcal{N}(0, \sigma_{\mathrm{AR},i}^2). \tag{17}$$

The AR(1) recursion produces a slowly varying random trajectory whose persistence is set by ϕ_i^{AR} and whose noise level is set by $\sigma_{\mathrm{AR},i}$; the clip at ± 0.6 keeps the drift contribution bounded.

Per-item bursts:

$$P_i(t) = \sum_k h_{i,k}^P\, \pi\left(\frac{t - t_{i,k}^P}{\Delta_{i,k}^P}\right), \tag{18}$$

where $\pi(\cdot)$ is a trapezoidal pulse: it ramps linearly up over the first 15% of its duration, plateaus for 60%, and ramps down over the last 25%. Bursts fire as a sparse random process per item.

Shared macro-shock:

$$G(t) = \sum_k h_k^G\, \pi\left(\frac{t - t_k^G}{\Delta_k^G}\right), \tag{19}$$

drawn once per run and shared across all items, with per-item sensitivity g_i. The distributions of all coefficients are given in Table 9 (§G.1).

B.1.3 REPLENISHMENT POLICY AND DESTINATION SERVICE

This section specifies the per-node replenishment policy and the destination service rule. The parameters $(s^{n,i}, S^{n,i}, \mu^{n,i})$ are used in §B.1.1 (the trigger set $\mathcal{T}$ and lead-time means) and in §B.2.1 (sub-step (vii)). Standing assumptions held fixed across all releases are recorded at the end.

Replenishment. Replenishment follows an (s, S) policy at every non-destination node: when on-hand drops below the reorder threshold s, place an order that brings on-hand back up to the order-up-to level S. The two node types realize this policy differently. Source nodes $n \in \mathcal{N}_{\mathrm{src}}$ act as factories with unlimited upstream supply: when $\mathrm{OH}^{n,i} < s^{n,i}$ and no order is outstanding, an order of size $S^{n,i} - \mathrm{OH}^{n,i}$ is placed and arrives after a random lead time $L^{n,i}$ drawn as in (5). Intermediate nodes $n \in \mathcal{N}_{\mathrm{int}}$ use the same trigger but pull inventory from upstream neighbours along real edges; the requested quantity is physically routed and packed (Algorithm 2, §B.2.2), and the arrival time equals the realized path-transit time. A closed-form lead time therefore exists at the source boundary and nowhere else.

Destination service. Demand $y_{i,t}$ is served at $d^\star$ in two ordered events within a single transition $\xi_t \to \xi_{t+1}$. First, scheduled arrivals at $d^\star$ are received: writing

$$A_{i,t} \ :=\ \mathrm{IT}_t[t{+}1,\, i] \tag{20}$$

for the units of item i scheduled to arrive at $d^\star$ between t and $t{+}1$, the arrivals are applied first to outstanding backlog, and the residual is added to on-hand:

$$B_{(2)}^{d^\star,i} \ =\ \big(B_t^{d^\star,i} - A_{i,t}\big)_+, \qquad \mathrm{OH}_{(2)}^{d^\star,i} \ =\ \mathrm{OH}_t^{d^\star,i} + \big(A_{i,t} - B_t^{d^\star,i}\big)_+. \tag{21}$$

Second, the day's demand is served from the post-arrival on-hand and any unmet portion accumulates as new backlog:

$$\mathrm{OH}_{t+1}^{d^\star,i} \;=\; \big(\mathrm{OH}_{(2)}^{d^\star,i} - y_{i,t}\big)_+, \qquad B_{t+1}^{d^\star,i} \;=\; B_{(2)}^{d^\star,i} + \big(y_{i,t} - \mathrm{OH}_{(2)}^{d^\star,i}\big)_+. \tag{22}$$

The fill-rate of item i over a horizon is the ratio of served units to demanded units on that horizon.

Standing assumptions. Three properties are held fixed within a run:

- *Stationary topology*: the graph $\mathcal{G}$ and the per-edge triple (τ_e, V_e, K_e) are set at construction and not modified.
- *Fixed control policy*: the (s, S) levels, the routing rule, and the packing primitive are held constant; no online control is applied.
- *Deterministic edge transit*: edges in $\mathcal{E}$ contribute no lead-time noise; randomness in lead times is confined to the source-level draw of (5).

B.1.4 Inter-warehouse flows

Cities communicate by shipping physical units across edges. This section gives the mathematical form of the three flow types that move material between cities. Each is a deterministic function of the state once the random external inputs (y_t, L_t) are fixed; the algorithmic resolution is in §B.2.1.

Source arrivals (boundary inflow). For $n \in \mathcal{N}_{\mathrm{src}}$, source orders are placed under the (s,S) trigger of (4) and delivered after the random lead time of (5). An order placed at time unit $t' < t$ with realized lead time $L_{t'}^{n,i}$ delivers as a scheduled arrival at time unit $t' + L_{t'}^{n,i}$; if this equals t the order arrives at t with size $S^{n,i} - \mathrm{OH}_{t'}^{n,i}$ (the order-up-to amount at the time of placement). The cumulative source-arrival mass between t and $t+1$, denoted $A_{i,t}^{\mathrm{src}}$, is defined formally in (27). Source replenishment is the mechanism by which units enter the network from outside; sources act as boundary inflow.

Outbound dispatch (warehouses to destination). For each item i and time unit t, the per-step target shipment volume to the destination is

$$\mathrm{target}_{i,t} = \max\Big\{0,\; B_t^{d^\star,i} + m\,\tilde{\lambda}_t^i - \mathrm{IT}_t^i - \mathrm{OH}_t^{d^\star,i}\Big\}, \tag{23}$$

where $\mathrm{IT}_t^i = \sum_{\tau > t} \mathrm{IT}_t[\tau, i]$ is total in-transit volume bound for $d^\star$ and m is the pipeline multiplier (a configuration parameter; $m = 7$ in this paper). The interpretation: ship enough to cover the destination's backlog plus m time units of expected demand, minus what is already on hand or in transit. Warehouses are visited in increasing Dijkstra distance to $d^\star$, where the edge weight on (u, v) is $\tau_{(u,v)}/(K_{(u,v)}V_{(u,v)})$ (favoring short paths with high per-step capacity); each warehouse's contribution is bounded by its on-hand stock and by the remaining target after earlier warehouses have shipped. The flow quantity from a warehouse is the volume placed by GreedyPack along the resolved path; arrivals are scheduled at $d^\star$ at time unit $t + \lceil \sum_e \tau_e \rceil$.

Inter-warehouse pull. Intermediate nodes $n \in \mathcal{N}_{\mathrm{int}}$ trigger replenishment when $\mathrm{OH}_t^{n,i} < s^{n,i}$ and $\mathrm{Out}_t^{n,i} = \emptyset$. The requested quantity is $S^{n,i} - \mathrm{OH}_t^{n,i}$. The flow is sourced from the first upstream supplier $u \in \mathcal{N}_{\mathrm{int}} \cup \mathcal{N}_{\mathrm{src}}$ with available stock $\mathrm{OH}_t^{u,i} > 0$, walked in increasing total path-transit-time order. The flow quantity is the volume placed by GreedyPack along the path from u to n, and arrival is scheduled at n at time unit $t + \lceil \sum_e \tau_e \rceil$.

Sequential coupling within a step. In all three flow types, the actual quantity placed depends on the residual capacity of edge containers along the path, which is itself a function of earlier flows in the same step. This makes the inter-warehouse flows sequentially coupled across items and across warehouses within the per-step update; the ordering of sub-steps in Algorithm 1 fixes this sequence. In Algorithm 1, source arrivals are realized in sub-step (vii) (order placement) and sub-step (i) (receipt), outbound dispatch in sub-step (v), and inter-warehouse pull in sub-step (vi).

B.2 Algorithms

This section gives the explicit algorithmic construction of the transition map Ψ defined abstractly in §B.1.1. §B.2.1 gives the seven-sub-step per-step procedure that defines Ψ. §B.2.2 defines the packing

routine called from §B.2.1. §B.2.3 gives the intensity-construction procedure that produces the demand rates of §B.1.2. The parameter values used to produce the released runs are in Appendix G.

B.2.1 PER-STEP ALGORITHM

Algorithm 1 realizes the transition map Ψ of §B.1.1 as a sequence of seven sub-steps, executed in the fixed order given below. Each sub-step reads state written by the previous one, and the conservation identities of Proposition C.1 depend on this ordering.

Algorithm 1 One step of the ISOMORPH simulation loop.

Input: time-unit index t; graph $\mathcal{G}$; per-node (s, S, μ_L); current state $\{\mathrm{OH}, B, \mathrm{Out}, \mathrm{IT}\}$; demand function $\mathrm{demand}(t)$; smoothed demand $\tilde{\lambda}$ with $\alpha = 0.05$; pipeline multiplier m.

(1) Receive scheduled (s, S) replenishment.
for $n \in \mathcal{N} \setminus \{d^\star\}$ and $i \in \mathcal{I}$ **do**
  if $\mathrm{Out}^{n,i}$ is (τ, q) with $\tau \le t$ **then**
    $\mathrm{OH}^{n,i} \leftarrow \mathrm{OH}^{n,i} + q$; clear $\mathrm{Out}^{n,i}$.
  end if
end for

(2) Receive arrivals at destination; backlog cleared first.
for i such that $\mathrm{IT}_t[t, i] > 0$ **do**
  $q \leftarrow \mathrm{IT}_t[t, i]$; $b \leftarrow B^{d^\star,i}$.
  $B^{d^\star,i} \leftarrow (b - q)_+$; $\mathrm{OH}^{d^\star,i} \leftarrow \mathrm{OH}^{d^\star,i} + (q - b)_+$.
end for

(3) Reset edge containers each step.
for $e \in \mathcal{E}$ **do**
  $\mathrm{containers}_e \leftarrow [V_e] \times K_e$.
end for

(4) Demand, service, and smoothed-demand update at $d^\star$.
$y \leftarrow \mathrm{demand}(t)$.
for $i \in \mathcal{I}$ **do**
  $\tilde{\lambda}_i \leftarrow \alpha\, y_i + (1 - \alpha)\, \tilde{\lambda}_i$.
  $\mathrm{served} \leftarrow \min(\mathrm{OH}^{d^\star,i},\ y_i)$; $\mathrm{unfilled} \leftarrow y_i - \mathrm{served}$.
  $\mathrm{OH}^{d^\star,i} \leftarrow \mathrm{OH}^{d^\star,i} - \mathrm{served}$; $B^{d^\star,i} \leftarrow B^{d^\star,i} + \mathrm{unfilled}$.
end for

(5) Dispatch warehouses $\to d^\star$ (round-robin over items).
for i in round-robin order **do**
  $\mathrm{target}_i \leftarrow \max\{0,\ B^{d^\star,i} + m\,\tilde{\lambda}_i - \text{in-transit}_i - \mathrm{OH}^{d^\star,i}\}$.
  $\mathrm{remaining} \leftarrow \mathrm{target}_i$.
  for w in warehouses sorted by Dijkstra distance to $d^\star$ (ascending) **do**
    if $\mathrm{remaining} \le 0$ **then break**
    end if
    $q_{\mathrm{try}} \leftarrow \min(\mathrm{OH}^{w,i},\ \mathrm{remaining})$.
    $q_{\mathrm{placed}} \leftarrow \textsc{GreedyPack}(i,\ q_{\mathrm{try}},\ \mathrm{Path}(w, d^\star))$. ▷ Alg. 2
    $\mathrm{OH}^{w,i} \mathrel{-}= q_{\mathrm{placed}}$; $\mathrm{remaining} \mathrel{-}= q_{\mathrm{placed}}$.
    Schedule arrival at $d^\star$ at time unit $t + \lceil \sum_{e \in \mathrm{Path}(w,d^\star)} \tau_e \rceil$ of size q_{placed}.
  end for
end for

(6) Inter-warehouse replenishment (pull along real edges).
for $n \in \mathcal{N}_{\mathrm{int}}$ in upstream-first order and each $i \in \mathcal{I}$ **do**
  if $\mathrm{OH}^{n,i} < s^{n,i}$ and $\mathrm{Out}^{n,i}$ is empty **then**

qty $\leftarrow S^{n,i} - \mathrm{OH}^{n,i}$.
for u upstream supplier of n, sorted by total transit time **do**
$q_{\text{placed}} \leftarrow$ GREEDYPACK$(i,\ \min(\mathrm{OH}^{u,i},\ \text{qty}),\ \mathrm{Path}(u, n))$.
if $q_{\text{placed}} > 0$ **then**
$\mathrm{OH}^{u,i} \mathrel{-}= q_{\text{placed}}$; $\mathrm{Out}^{n,i} \leftarrow (t + \lceil \sum_e \tau_e \rceil,\ q_{\text{placed}})$; **break**.
end if
end for
end if
end for

(7) Place (s, S) orders at source nodes (random lead time).
for $n \in \mathcal{N}_{\text{src}}$ and $i \in \mathcal{I}$ **do**
if $\mathrm{OH}^{n,i} < s^{n,i}$ and $\mathrm{Out}^{n,i}$ is empty **then**
Draw $X \sim \mathcal{N}(\mu^{n,i},\ (0.2\,\mu^{n,i})^2)$; $L \leftarrow \max(1, \lceil X \rceil)$.
$\mathrm{Out}^{n,i} \leftarrow (t + L,\ S^{n,i} - \mathrm{OH}^{n,i})$.
end if
end for

B.2.2 GREEDY FIRST-FIT PACKING

The same routine handles outbound shipping (sub-step 5 of Algorithm 1) and inter-warehouse pull (sub-step 6). A call attempts to place Q units of a single item along a fixed path P, one unit at a time. For each unit, the routine walks P edge by edge and selects the first container on that edge with enough residual volume. This keeps the order of units along the path consistent with the order in which they were requested.

Algorithm 2 GREEDYPACK: first-fit allocation of up to Q units of item i along a path P.

Input: item i with volume v_i; max units Q; path $P = (e_1, \ldots, e_k)$.
placed $\leftarrow 0$.
for $q = 1, \ldots, Q$ **do**
slots $\leftarrow [\,]$; ok $\leftarrow$ TRUE.
for $e \in P$ **do**
$j \leftarrow$ smallest index with $\mathrm{containers}_e[j] \geq v_i$, or $\perp$ if none.
if $j = \perp$ **then**
ok $\leftarrow$ FALSE; **break**.
end if
Append (e, j) to slots.
end for
if not ok **then**
break ▷ stop filling; do not retry a later unit
end if
for $(e, j) \in$ slots **do**
$\mathrm{containers}_e[j] \mathrel{-}= v_i$.
end for
placed $\leftarrow$ placed $+ 1$.
end for
return placed.

B.2.3 INTENSITY CONSTRUCTION ALGORITHM

Algorithm 3 constructs the intensity tensor λ used in (14). Of the five components it samples, four are drawn independently per item: yearly seasonal, weekly seasonal, AR(1) drift, and per-item burst train. The fifth, $G(t)$, is drawn once and reused across every item, with a per-item weight g_i. As noted in §B.1.2, $G(t)$ is the component that lifts every item's intensity simultaneously, and is the mechanism by which demand forecasting and bottleneck forecasting become coupled.

Algorithm 3 BUILDINTENSITY: sample the five-component intensity λ.

Input: item list $\mathcal{I}$; horizon T; seed.

Global macro-shock process $G(t)$.
$G \leftarrow 0_T$; draw N uniformly from $\{5, \dots, 11\}$.
for $k = 1, \dots, N$ **do**
  Draw t_k^G uniform on $\{0, \dots, T-1\}$; Δ_k^G uniform on $\{180, \dots, 1099\}$; $h_k^G \sim \mathcal{U}[0.20, 0.60]$.
  Add trapezoidal pulse $\pi(\cdot)$ with parameters $(t_k^G, \Delta_k^G, h_k^G)$ to G.
end for

Per-item components.
for $i \in \mathcal{I}$ **do**
  $\bar{\lambda}_i \sim \mathcal{U}[80, 250]$.
  Yearly: $S_i(t) \leftarrow a_{1,i}^{(y)} \sin(2\pi t/365 + \phi_i) + a_{2,i}^{(y)} \sin(4\pi t/365 + 0.7\,\phi_i)$ with $a_{1,i}^{(y)} \sim \mathcal{U}[0.12, 0.28]$, $a_{2,i}^{(y)} \sim \mathcal{U}[0.04, 0.10]$, $\phi_i \sim \mathcal{U}[0, 2\pi]$.
  Weekly: $W_i(t) \leftarrow a_i^{(w)} \sin(2\pi(t \bmod 7)/7 + \psi_i)$ with $a_i^{(w)} \sim \mathcal{U}[0.04, 0.10]$, $\psi_i \sim \mathcal{U}[0, 2\pi]$.
  AR(1) drift: $\phi_i^{\mathrm{AR}} \sim \mathcal{U}[0.9990, 0.9996]$, $\sigma_{\mathrm{AR},i} \sim \mathcal{U}[0.008, 0.018]$; $A_i(0) \sim \mathcal{N}(0, 0.10^2)$; $A_i(t) \leftarrow \mathrm{clip}\big[\phi_i^{\mathrm{AR}} A_i(t-1) + \mathcal{N}(0, \sigma_{\mathrm{AR},i}^2),\ \pm 0.6\big]$.
  Per-item spikes $P_i(t)$: burst rate $r_i \sim \mathcal{U}[2 \cdot 10^{-4}, 10^{-3}]$; at each t with $\mathrm{Bern}(r_i) = 1$, add a trapezoidal pulse of duration $\mathcal{U}\{30, \dots, 179\}$ and height $\mathcal{U}[0.20, 0.70]$.
  Global sensitivity: $g_i \sim \mathcal{U}[0.4, 1.2]$.
  Assemble: $\lambda_{i,t} \leftarrow \bar{\lambda}_i \cdot \max\big(0.08,\ 1 + S_i(t) + W_i(t) + A_i(t) + P_i(t) + g_i\, G(t)\big)$.
end for
return λ.

Algorithm 1 specifies the ordered sequence of seven sub-steps that defines the transition map. App. H considers alternative transition rules and summarizes the extensions implemented in the released simulator, as well as additional mechanisms that are not currently implemented. For each extension, we indicate whether the stated Markov property and the conservation identities of Proposition C.1 continue to hold.

C SUPPLEMENTARY MATERIAL FOR SECTION 3: PROOF AND CONTINUOUS-TIME FLUID LIMIT

Unlike existing resources limited to per-node demand or inventory (Wasi et al., 2024; Naik et al., 2025), the state vector ξ_t in ISOMORPH carries outstanding orders, scheduled arrivals, and smoothed demand estimates alongside per-node inventory. These auxiliary components lift the system into a space where three properties emerge: the dynamics are Markovian (§3.1), three discrete conservation laws hold exactly on every sample path (§3.2), and the discrete laws align with classical continuum transport and queueing equations (§C.2).

C.1 CONSERVATION LAWS: STATEMENT AND PROOF

Proposition C.1 (Conservation of units and demand events)**.** *For each item $i \in \mathcal{I}$ and time unit $t \in \{0, \dots, T-1\}$, let $u_{i,t}^e$ denote the units of item i placed onto edge e at sub-steps (5)–(6) of Algorithm 1. Define per-node receipts and dispatches in $[t, t+1]$ as*

$$R_t^{n,i} := \begin{cases} q, & n \neq d^\star \text{ and } \mathrm{Out}_t^{n,i} = (t{+}1, q), \\ (A_{i,t} - B_t^{d^\star,i})_+, & n = d^\star, \\ 0, & \text{otherwise}; \end{cases} \tag{24}$$

$$D_t^{n,i} := \begin{cases} \sum_{(n,v)\in\mathcal{E}} u_{i,t}^{(n,v)}, & n \neq d^\star, \\ \min\big(\mathrm{OH}_{(2)}^{d^\star,i},\ y_{i,t}\big), & n = d^\star. \end{cases} \tag{25}$$

Define the network-internal stock

$$I_{i,t} := \sum_{n\in\mathcal{N}} \mathrm{OH}_t^{n,i} + \sum_{(u,v)\in\mathcal{E}, v\neq d^\star} q_{i,t}^{u\to v} + \sum_{\tau>t} \mathrm{IT}_t[\tau, i], \tag{26}$$

where $q_{i,t}^{u\to v}$ is the per-edge in-transit count defined in §B.1.1 and $\mathrm{IT}_t[\tau, i]$ is the multiplicity of arrivals scheduled at $d^\star$ at time unit τ for item i. Define the source arrivals between t and $t+1$ as

$$A_{i,t}^{\mathrm{src}} := \sum_{n\in\mathcal{N}_{\mathrm{src}}} \mathbf{1}\Big\{\mathrm{Out}_t^{n,i} = (\tau, q),\ \tau = t+1\Big\} \cdot q, \tag{27}$$

the mass arriving at non-destination nodes from sources between time units t and $t+1$. With $A_{i,t} := \mathrm{IT}_t[t{+}1, i]$ and $\mathrm{OH}_{(2)}^{d^\star,i} = \mathrm{OH}_t^{d^\star,i} + (A_{i,t} - B_t^{d^\star,i})_+$ as defined in (20)–(21), define the total destination outflow as the sum of the arrival-clears-backlog flow and the immediate-service flow:

$$S_{i,t} \ :=\ \min\big(A_{i,t},\, B_t^{d^\star,i}\big) \ +\ \min\big(\mathrm{OH}_{(2)}^{d^\star,i},\, y_{i,t}\big). \tag{28}$$

Then the chain $(\xi_t)_{t\geq 0}$ satisfies, almost surely for all $t \in \{0, \ldots, T-2\}$ and $i \in \mathcal{I}$:

$$\textit{(a) Per-node mass conservation:} \qquad \mathrm{OH}_{t+1}^{n,i} = \mathrm{OH}_t^{n,i} + R_t^{n,i} - D_t^{n,i}, \quad \forall n \in \mathcal{N}, \tag{29}$$

$$\textit{(b) Global mass conservation:} \qquad I_{i,t+1} = I_{i,t} + A_{i,t}^{\mathrm{src}} - S_{i,t}, \tag{30}$$

$$\textit{(c) Backlog conservation:} \qquad B_{t+1}^{d^\star,i} = B_t^{d^\star,i} + \big(y_{i,t} - \mathrm{OH}_t^{d^\star,i}\big)_+. \tag{31}$$

Proof. **Part (a).** Fix $n \in \mathcal{N}$ and $i \in \mathcal{I}$. We trace changes to $\mathrm{OH}^{n,i}$ across the seven sub-steps of Algorithm 1 during the transition from t to $t+1$.

Case $n \neq d^\star$. Sub-step (1) increments $\mathrm{OH}^{n,i}$ by q when $\mathrm{Out}_t^{n,i} = (t{+}1, q)$, and by zero otherwise; this matches $R_t^{n,i}$ as defined in (24). Sub-steps (2) and (4) act at $d^\star$ alone and leave $\mathrm{OH}^{n,i}$ unchanged. Sub-step (3) resets edge containers without modifying any node's on-hand. Sub-steps (5) and (6) decrement $\mathrm{OH}^{n,i}$ by exactly the units placed onto n's outgoing edges via GREEDYPACK, i.e., $\sum_{(n,v)\in\mathcal{E}} u_{i,t}^{(n,v)} = D_t^{n,i}$. Sub-step (7) modifies $\mathrm{Out}^{n,i}$ for $n \in \mathcal{N}_{\mathrm{src}}$ but does not modify $\mathrm{OH}^{n,i}$. Collecting: $\mathrm{OH}_{t+1}^{n,i} = \mathrm{OH}_t^{n,i} + R_t^{n,i} - D_t^{n,i}$.

Case $n = d^\star$. Sub-step (1) acts at non-destination nodes. Sub-step (2) increments $\mathrm{OH}^{d^\star,i}$ by $(A_{i,t} - B_t^{d^\star,i})_+ = R_t^{d^\star,i}$ (the post-backlog residual of arriving units, per (21)). Sub-step (3) resets edge containers. Sub-step (4) decrements $\mathrm{OH}^{d^\star,i}$ by $\min(\mathrm{OH}_{(2)}^{d^\star,i}, y_{i,t}) = D_t^{d^\star,i}$ (the served quantity, second term of $S_{i,t}$ in (28)). Sub-steps (5)–(7) do not modify $\mathrm{OH}^{d^\star,i}$: the destination is excluded from the warehouse loop in (5), is not an upstream supplier in (6), and is not a source in (7). Hence $\mathrm{OH}_{t+1}^{d^\star,i} = \mathrm{OH}_t^{d^\star,i} + R_t^{d^\star,i} - D_t^{d^\star,i}$.

Part (b). Summing Part (a) over all nodes and adding the in-transit and scheduled-arrival components: routing along edges transfers units between $\sum_n \mathrm{OH}^{n,i}$ and $\sum_e q_i^{u\to v}$ without changing the total $I_{i,t}$ defined in (26). Two mechanisms change $I_{i,t}$: source arrivals (sub-steps i and vii), which add $A_{i,t}^{\mathrm{src}}$, and destination service (sub-steps ii and iv), which removes $S_{i,t}$. No other sub-step changes it. Therefore $I_{i,t+1} - I_{i,t} = A_{i,t}^{\mathrm{src}} - S_{i,t}$ almost surely.

Part (c). Immediate from (21)–(22) of §B.1.3: at sub-step (ii), backlog is updated to $B_{(2)}^{d^\star,i} = (B_t^{d^\star,i} - A_{i,t})_+$; at sub-step (iv), the unfilled remainder $(y_{i,t} - \mathrm{OH}_{(2)}^{d^\star,i})_+$ is added to $B_{(2)}^{d^\star,i}$ to give $B_{t+1}^{d^\star,i}$. No other sub-step modifies $B^{d^\star,i}$. □

All three identities are preserved exactly—not approximately—because the simulator's transitions are bookkeeping operations on integer counts, with no rounding or stochastic loss inside the network. Any deviation at any (t, i) in a released run would indicate a bug in the simulator, corruption in the released files, or a mismatch between the model and its algorithmic realization.

Conservation laws as structural invariants of the DBN. All three laws hold pathwise—for every realization of the random external inputs (y_t, L_t), and not merely in expectation. The two boundary terms of the global mass conservation align with the two random input nodes in the Markov chain (Figure 1): $A^{\mathrm{src}}_{i,t}$ is determined by the lead-time vector L_t (which controls when source orders arrive), and the service component of $S_{i,t}$ is determined by the demand vector y_t together with the on-hand state at $d^\star$. The places where randomness enters the chain are the same places where mass crosses the boundary of the conserved domain. Internal flows (routing, packing, inter-warehouse transfers) are deterministic given (y_t, L_t) and conservative by construction. The conservation laws are therefore structural invariants of the Markov chain: they constrain every sample path the simulator can produce.

Conservation laws as a fidelity check. Because every unit and every demand event is accounted for at every step, downstream metrics—fill rate, service level, backlog counts—are well defined and recoverable from the released data. When users add their own control rules to the simulator, these conservation laws, or adapted versions of them, can verify the fidelity of the modified implementation.

C.2 Continuous-time fluid limit: coupled transport and queueing

This section highlights the mathematical structure of ISOMORPH and connects it to the literature on queueing systems and their fluid limits. First, the three discrete conservation laws of Proposition C.1 are not isolated bookkeeping identities: they are the pre-limit forms of classical continuum equations that govern the fluid limit of the Markov chain. Consider a sequence of systems indexed by a system-size parameter N, in which the catalogue size, demand intensities, edge capacities, and control thresholds (s, S) are all scaled by N simultaneously. Under this Kurtz-type law-of-large-numbers scaling (Kurtz, 1970), each individual unit becomes a vanishing fraction of the total, and the rescaled density $\rho^{N,n,i}(t) = N^{-1}\mathrm{OH}^{n,i}_{\lfloor tN \rfloor}$ converges in probability to the deterministic solution of a coupled system: a transport equation on the graph $\mathcal{G}$ governing inventory, and a Skorokhod-reflected ODE at $d^\star$ governing backlog. The discrete conservation laws hold at every finite N and pass to the limit, making the continuum system their deterministic counterpart.

At the discrete level, the simulator is a jump process on a network of control volumes: each node $n \in \mathcal{N}$ is a fixed control volume that accumulates on-hand inventory $\mathrm{OH}^{n,i}_t$, and the edges $\mathcal{E}$ are conduits that carry discrete unit-valued jumps between control volumes. The fluid limit smooths these integer-valued jumps into continuous densities and fluxes.

Transport (inventory side). In the fluid limit, the per-node, per-item inventory satisfies a delay-differential equation on $\mathcal{G}$: for each node $n \in \mathcal{N}$ and item $i \in \mathcal{I}$,

$$\partial_t \rho^{n,i}(t) = \sum_{u:(u,n)\in\mathcal{E}} \phi_i^{u\to n}(t-\tau_{u,n}) - \sum_{v:(n,v)\in\mathcal{E}} \phi_i^{n\to v}(t) + r_i^n(t)\,\mathbf{1}\{n \in \mathcal{N}_{\mathrm{src}}\} - s_i^n(t)\,\mathbf{1}\{n = d^\star\}, \tag{32}$$

subject to edge capacity constraints $\sum_i v_i \phi_i^{u\to v}(t) \le \kappa_{u,v}$. This is the continuous form of the local continuity equation $\partial_t \rho + \nabla \cdot j = \sigma$ on the graph, where the delayed inflows from upstream edges play the role of flux divergence, and source replenishment r_i^n and destination service s_i^n act as boundary source/sink terms.

The per-node mass conservation (29) is the time-discretized form of (32): on-hand inventory $\mathrm{OH}^{n,i}_t$ is the local density in the control volume at node n, receipts $R^{n,i}_t$ and dispatches $D^{n,i}_t$ are the discrete inflow and outflow across the control-volume boundary, and their difference is the discrete divergence. Integrating over the entire graph–summing over all control volumes–yields the global mass conservation (30): internal fluxes cancel in pairs, leaving the boundary terms, giving the discrete analog of $\frac{d}{dt}\int_\Omega \rho = -\int_{\partial\Omega} j \cdot n + \int_\Omega \sigma$.

Queueing (demand side). In classical queueing theory, the queue length Q evolves according to the fluid model $dQ/dt = \lambda(t) - \mu(t)$, where λ is the arrival rate and μ is the service rate, subject to the constraint $Q \ge 0$ (Skorokhod reflection). In ISOMORPH, the backlog $B^{d^\star,i}_t$ at the destination $d^\star$ for each item $i \in \mathcal{I}$ plays the role of Q, demand $y_{i,t}$ plays the role of λ, and fulfilled orders $\min(\mathrm{OH}^{d^\star,i}_t, y_{i,t})$ play the role of μ. The discrete backlog conservation (31) is the time-discretized form of this fluid model, with the $(\cdot)_+$ operator enforcing non-negativity.

In the fluid limit, the rescaled backlog at the destination $d^\star$ for each item $i \in \mathcal{I}$ converges to the Skorokhod-reflected ODE

$$\partial_t b^{d^\star,i} = (\lambda_i - \rho^{d^\star,i})_+ - (A_i - b^{d^\star,i})_+, \tag{33}$$

where the first term adds unmet demand (arrivals exceeding on-hand density) and the second term drains backlog when shipments arrive at $d^\star$.

Coupling at the destination. The transport and queueing subsystems couple at $d^\star$: the service term $S_{i,t}$ is simultaneously the boundary outflow of the transport equation (32) and the service rate of the queue (33). The transport network's ability to deliver units to $d^\star$ determines the queue's service rate, and the queue's backlog state feeds back into dispatch targets through the proactive shipping formula of sub-step (5). This coupling persists in both the discrete and continuum settings. Table 4 summarizes the full correspondence.

Table 4: Correspondence between the fluid-limit equations and the discrete conservation laws of Proposition C.1. Each node $n \in \mathcal{N}$ serves as a control volume; edges carry discrete unit-valued jumps that become continuous fluxes in the fluid limit.

Fluid limit	Discrete (ISOMORPH)
(a) Per-node mass conservation (29)*: local continuity equation* (32)	
density $\rho^{n,i}(t)$	on-hand inventory $\mathrm{OH}_t^{n,i}$
flux $\phi_i^{u\to v}$	shipment flow $u_{i,t}^e$ along edge e
divergence	net outflow from control volume at node n: $D_t^{n,i} - R_t^{n,i}$ (Eqs. 25, 24)
source/sink r_i^n, s_i^n	$A_{i,t}^{\mathrm{src}}$ (27) (source inflow), $S_{i,t}$ (28) (destination outflow)
(b) Global mass conservation (30)*: integrated form*	
integrated density $\int_\Omega \rho$	network-internal stock $I_{i,t}$ (26)
boundary flux $\int_{\partial\Omega} j \cdot n$	$A_{i,t}^{\mathrm{src}}$ (27) (in), $S_{i,t}$ (28) (out)
(c) Backlog conservation (31)*: reflected ODE* (33)	
queue length $b^{d^\star,i}$	backlog $B_t^{d^\star,i}$
arrival rate λ_i	demand $y_{i,t}$
service rate	$\min(\mathrm{OH}_t^{d^\star,i},\, y_{i,t})$
Coupling at $d^\star$: $S_{i,t}$ = transport outflow = queue service rate	

Each sub-step of Algorithm 1 contributes a jump of size $O(1/N)$ at rate $O(N)$ to the rescaled generator, placing the chain in the regime where Kurtz's theorem (Kurtz, 1970) applies. The (s, S) replenishment policy introduces switching boundaries at the reorder threshold, producing either a hybrid system with Filippov solutions or, under a smoothed base-stock control, a classical ODE; both are standard in the fluid-network literature (Chen & Yao, 2001; Mandelbaum & Pats, 1998). The edge capacity constraints produce a complementarity structure–flow at capacity implies zero slack at the controller–shared with fluid models of multi-class queueing networks (Whitt, 2002). A rigorous formulation of the limit theorem and the associated diffusion correction (CLT-type Brownian fluctuations around ρ) is of independent mathematical, computational and statistical interest.

D SUPPLEMENTARY MATERIAL FOR SECTION 4: OUTPUT SEQUENCES AND OBSERVABLE SUPPLY-CHAIN PHENOMENA

The simulator provides a configurable generative process, while the released datasets instantiate this process as concrete forecasting and uncertainty- quantification benchmarks in § 5. We first describe the structural configurations, scenario knobs, and output sequences used to construct the release. We then illustrate observable supply-chain phenomena that emerge from these configurations, including demand amplification, inventory and transportation dynamics, service degradation, and disruption propagation. The resulting dataset family combines fixed structural settings with controlled variation in the operating regime, providing both long-horizon forecasting series and network-state trajectories for downstream tasks.

D.1 SCENARIO KNOBS AND OUTPUT SEQUENCES

Scenario knobs. We vary six scenario knobs to generate different demand regimes and network responses. Three knobs perturb the demand process: *drift* controls the persistence of long-lived demand-level changes, *burst* controls transient item-specific demand surges, and *shock* controls network-wide demand events shared across items. Three additional knobs perturb the network response through transportation capacity, inventory buffering, and source lead times. Each knob is varied independently around a common baseline, with additional compound scenarios combining demand-side perturbations. The resulting scenarios provide controlled variation in the temporal and cross-item structure of demand and in the network conditions under which it propagates. Complete sweep definitions, parameter settings, and scenario names are given in App. G.5.

Output sequences. Each rollout records the full network state trajectory: per-item demand, service, and backlog at the destination; per-node inventory levels; in-transit volumes; and per-edge shipments with resolved paths. Dataset schemas are given in Appendix I. For the TSF evaluation of §5, we use four target families: *demand*, *backlog*, and *fill rate* at the item level, and *edge utilization* at the edge level. Demand is taken from `daily_records.csv`; backlog from `backlog_history.csv` at $n = d^\star$; and fill rate is computed as `served_from_stock`/`demand` from `daily_records.csv`. Edge utilization is taken from `edge_utilisation.npy`, with columns ordered according to `edge_list.csv`. Thus, the first three targets are indexed by item, whereas edge utilization is indexed by edge. Diverse dynamics emerge from different scenario configurations (scenario definitions in Table 12, App. G.5; instantiation in §5.2); Figure 3 illustrates this for a single item across four scenarios and two output variables.

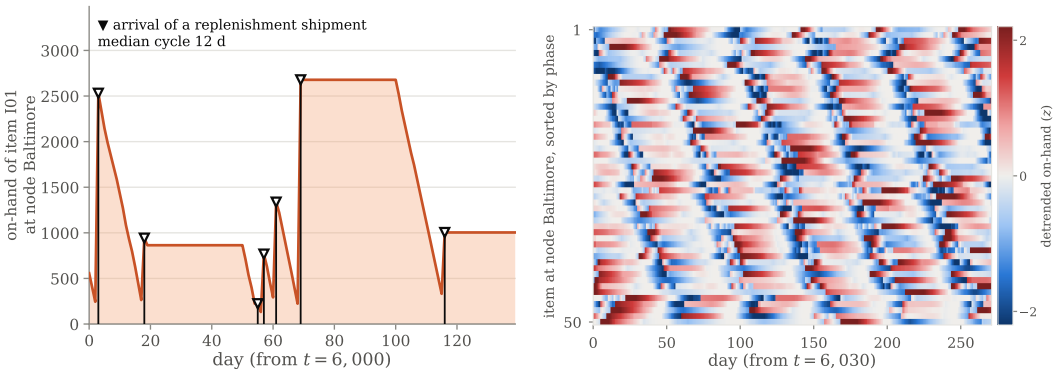
D.2 OBSERVABLE SUPPLY-CHAIN PHENOMENA

To demonstrate that ISOMORPH can generate known supply-chain behaviors, we examine a set of commonly observed phenomena that provide interpretable examples of the simulator's dynamics. These include demand amplification, inventory and transportation dynamics, service degradation, and disruption propagation. Where a corresponding public dataset is available, we compare the simulated and observed behaviors in terms of their qualitative patterns and effect sizes; otherwise, we illustrate the phenomenon using ISOMORPH alone. Since the network structure, capacities, and control rules behind those datasets are not available, these comparisons assess behavioral consistency rather than trajectory-level agreement with a particular operational system. Because some of these phenomena require configurations beyond the baseline scenarios, we therefore include dedicated rollouts in the released dataset to demonstrate them directly. The corresponding scenario definitions and directory organization are given in App. G.5 and App. I.3.

Inventory oscillation. Figure 6 illustrates the inventory oscillations generated by the (s, S) replenishment cycle. Individual items exhibit recurring sawtooth patterns, with cycle timing and amplitude determined by the replenishment and demand dynamics. Across the scenario library, the detrended $\mathrm{AC}(7)$ of on-hand inventory ranges from -0.00 to 0.98, spanning the $[0.31, 0.72]$ range observed across SupplyGraph SKUs (App. L.3). At a single node, different items exhibit similar oscillation periods with staggered phases, so aggregation across items can substantially smooth the pattern.

High edge-utilization periods. Figure 7 illustrates how transportation capacity shapes high-utilization periods across the network. At baseline capacity, no released edge spends more than 0.33% of days at or above $U_{e,t} = 0.8$; under a uniform $0.3\times$ capacity scaling, the most highly utilized edge reaches 7.2% of days above the same threshold. Capacity tightening therefore creates localized periods of elevated edge utilization. The full edge-level results are shown in Figure 7b.

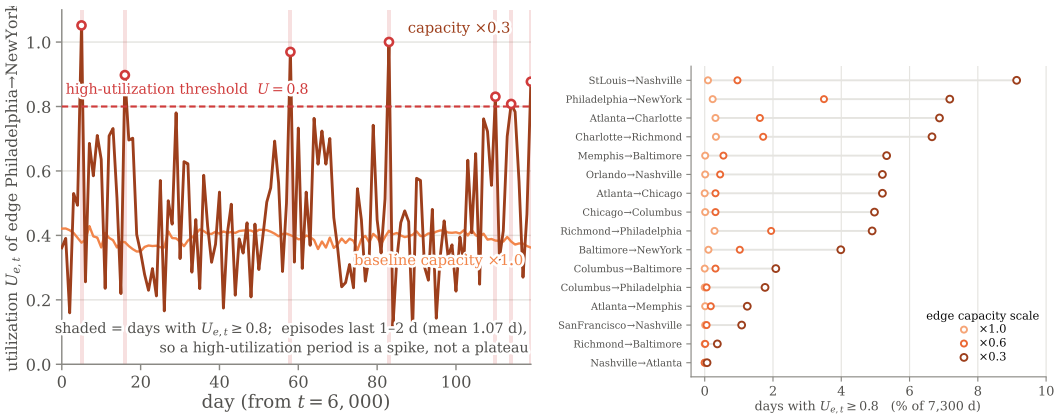
Disruption and delivery lateness. Figure 8 illustrates disruption-related service degradation in both ISOMORPH and Global Supply Chain Disruption (Uskono, 2026), a second synthetic dataset whose generating process is not documented by its publisher. In that dataset, disruption events are associated with elevated late-delivery rates on several shipping lanes. In ISOMORPH, removing a corridor edge without an alternate route produces a substantial increase in destination unfilled-on-time demand, whereas removing an edge with an alternate route produces a smaller response. A matched



(a) On-hand inventory of item I01 over 140 days.

(b) Detrended and standardized on-hand inventory for all 50 items over 300 days, sorted by the timing of the first trough.

Figure 6: **Inventory oscillation in ISOMORPH.** Both panels show the baseline $C = 50$ rollout at Baltimore. (a) The on-hand inventory of item I01 over 140 days, showing the sawtooth pattern produced by the (s, S) replenishment cycle. (b) Detrended and standardized on-hand inventory for all 50 items over 300 days, sorted by the timing of the first trough. The staggered phases across items smooth the oscillation when aggregated.



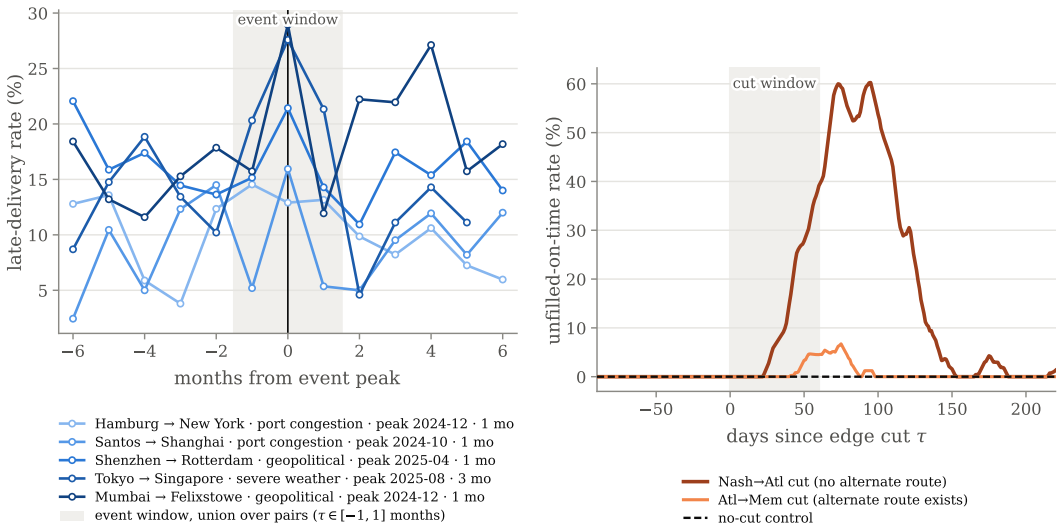
(a)

(b)

Figure 7: **High edge-utilization periods in ISOMORPH.** (a) Utilization of the Philadelphia→NewYork edge over 120 days at baseline and under 0.3× capacity. (b) Fraction of the 7,300 days with $U_{e,t} \geq 0.8$ for all 16 edges under three uniform capacity scalings. Capacity tightening produces localized periods of high edge utilization, with different edges affected according to the network topology and routing.

no-cut control remains unchanged. Both series are synthetic and measure different service-level quantities, so the comparison illustrates the corresponding disruption response rather than matching the outcome variables directly, and it carries no claim about either dataset's realism.

Bottleneck migration. Figure 9 illustrates how a temporary edge disruption can redistribute flow across alternative routes. Removing the Atlanta→Memphis edge causes Baltimore's inbound volume to shift toward the Columbus and Richmond routes before returning toward its original composition after restoration. The response is delayed because the affected route can continue supplying from buffer stock during the disruption. Thus, the network can exhibit route-level supplier migration rather than simple loss of downstream flow.



(a) Global Supply Chain Disruption, a synthetic dataset (Uskono, 2026) (10,000 orders, 2024-01 to 2026-01): monthly late-delivery rates for five lane–event-type pairs, aligned by months from each pair's peak disruption month.

(b) ISOMORPH: destination unfilled-on-time rate after 60-day cuts to Nashville→Atlanta (no alternate route) and Atlanta→Memphis (alternate route), compared with a matched no-cut control.

Figure 8: **Disruption-related service degradation in two synthetic supply-chain datasets.** (a) Global Supply Chain Disruption, a synthetic dataset: event-aligned late-delivery rates for five lane–event-type pairs. (b) ISOMORPH: destination unfilled-on-time rate following two 60-day edge cuts, compared with a matched no-cut control. The two panels use different service-level measures; the comparison illustrates the response to disruption rather than a one-to-one correspondence between the outcome variables.

E SUPPLEMENTARY MATERIAL FOR SECTION 5: ADDITIONAL RESULTS

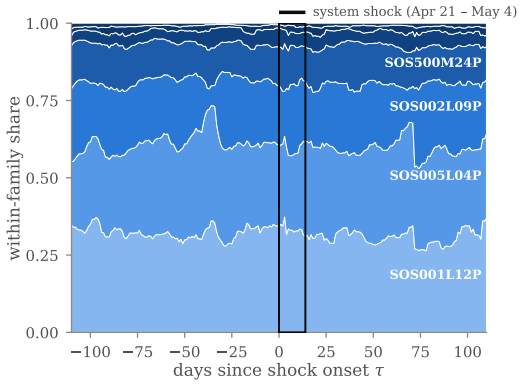
E.1 COMBINING THE DIGITAL TWIN WITH TSF FOUNDATION MODELS: OVERVIEW

Combining a digital twin with TSF foundation models enables two complementary capabilities: using ISOMORPH as a regenerable benchmark for time-series forecasting and as a testbed for evaluating foundation models as fast surrogates for digital-twin forward simulation and uncertainty quantification. We first assess forecasting performance across the released target variables and scenario configurations (§5.1), and then evaluate whether foundation models can reproduce the predictive variability induced by changes in the underlying system configuration (§5.2). Experiments probing these capabilities are in §5.3.

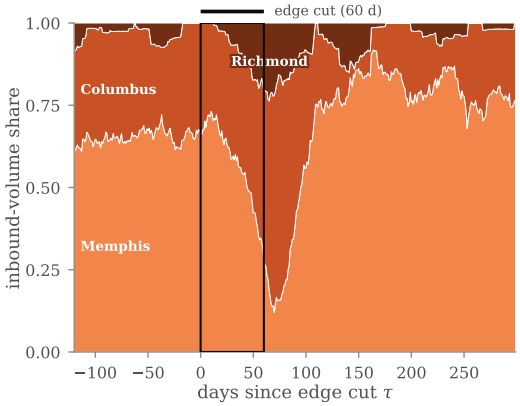
E.2 TSF BENCHMARKING: PROTOCOL DETAILS

ISOMORPH introduces the logistics domain to TSF benchmarking. Different scenario configurations produce diverse sequential patterns—oscillatory reorder cycles, persistent drift, global shocks, and localized bursts (Figure 3)—that are largely absent from existing benchmark collections. The setup below is designed to examine two aspects separately: how current forecasters perform across these generated patterns, and how forecasting difficulty varies between the demand process and network-dependent targets arising from the multi-echelon structure.

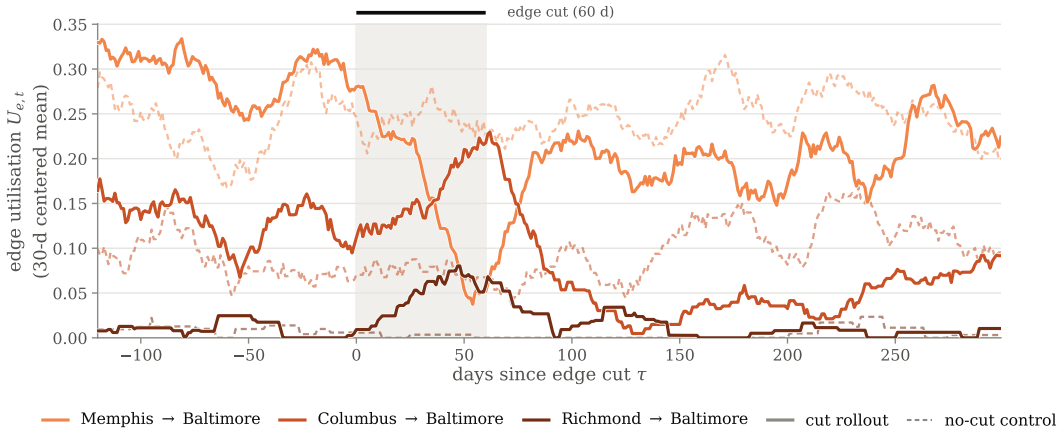
Rolling-origin protocol. All models see $L{=}512$ past steps and predict at horizons $h \in \{1, 7, 14, 30\}$, which fall within the short-to-medium range of GIFT-Eval's 97 task configurations ($h \in [6, 900]$) (Aksu et al., 2024). Each rollout is split chronologically $70/15/15$; the forecast origin slides across the test split in non-overlapping steps of 30, giving $|W|{=}36$ windows per rollout at $T{=}7{,}300$. Context windows are allowed to span the train/test boundary, since no model is refit online.



(a) SupplyGraph: seven close-substitute SKUs from the SOS family, showing within-family Sales-Order composition (7-day rolling) around the Eid-ul-Fitr system shutdown (2023-04-21 to 2023-05-04, boxed). The composition is measured as each SKU's share of total sales orders within the family, with time aligned to the onset of the shutdown. The figure shows how order share is redistributed among related SKUs following the disruption.



(b) ISOMORPH: inbound-volume composition at Baltimore under a 60-day removal of the Atlanta→Memphis edge. Each area shows the share of Baltimore's inbound volume supplied through Memphis, Columbus, or Richmond (30-day rolling), aligned to the edge-cut onset. Removing the Memphis route causes its share to fall while Columbus and Richmond increase their shares to absorb the displaced flow, with the response lagging the edge cut and recovery after restoration.



(c) The same three Baltimore-inbound edges as in (b), showing edge utilization $U_{e,t}$ (30-day centered mean) under the edge cut (solid) and a matched no-cut control (dashed). Following the Atlanta→Memphis cut, utilization of Memphis→Baltimore falls while the Columbus→Baltimore and Richmond→Baltimore alternatives increase, whereas the control remains comparatively stable.

Figure 9: **Flow redistribution following a disruption in real and simulated supply-chain data.** (a) SupplyGraph: within-family sales-order composition for seven related SKUs around a system disruption. (b) ISOMORPH: inbound-volume composition at Baltimore following a 60-day Atlanta→Memphis edge cut, showing redistribution toward alternative routes. (c) Utilization of the three Baltimore-inbound edges under the cut and a matched no-cut control. The two systems exhibit temporary redistribution among alternatives, with SupplyGraph capturing SKU-level substitution and ISOMORPH capturing route-level migration.

Error metric and seasonal normalizers. We report Mean Absolute Scaled Error (MASE) following the GIFT-Eval protocol (Aksu et al., 2024). At each window, the model's MAE over the first h steps is normalized by the seasonal-naive error on that window's own L-step context, $\frac{1}{L-m}\sum_t |y_{t+m} - y_t|$ (App. M.1). Because the released data is daily and the choice of m changes what counts as "trivially predictable," we report *three* normalizers rather than one: m=1 (the GIFT-Eval default), m=7 (weekly, matching the daily resolution and the weekly harmonic of Eq. (16)), and m=30 (monthly).

Evaluation cohort and uncertainty. Every reported interval is a 95% Student-t confidence interval over the n=29-rollout cohort of §4.1: one baseline, 24 static one-knob variants, two compound-stress scenarios, and the two dynamic edge-cut scenarios of §4.2. These intervals therefore quantify variability across operating regimes. For backlog and fill rate, the effective sample is n=24, because five under-loaded rollouts have identically zero backlog and unit fill rate, making MASE undefined. These rollouts are excluded for this reason.

E.3 FORWARD-UQ PROTOCOL AND CALIBRATION METRICS

Forward-UQ protocol (parameter uncertainty)

1. Take the scenario design of §4.1 as the perturbation structure: six one-at-a-time sweeps each move a single knob away from a shared baseline through four settings (App. G.5).
2. Use K=20 of the 29 cohort rollouts as the forward-UQ ensemble, keeping every knob family of the design represented. The ensemble reads the same directories as the forecasting evaluation and adds none of its own (App. I.3).
3. For each ensemble member k, read all four forecast targets c of §5.1 off the corresponding rollout, giving trajectories $y_t^{(k,c)}$. One rollout therefore serves every target; no re-simulation is needed per target.
4. For each foundation model $m \in \{\text{Chronos}, \text{Moirai}, \text{TimesFM}\}$, apply it zero-shot with context length L=512 to each $y^{(k,c)}$, producing forecasts $\hat{y}_t^{(k,c),m}$.
5. Score each (k, w, c) cell against the model's own predictive distribution at that cell, and aggregate over the ensemble into the two metrics below.

The ensemble is a designed set of configurations and not a sample from a probability measure on knob space. The coverage reported below is therefore coverage across the scenario families the design spans, and it does not carry the frequentist reading that sampling from an assumed prior would give it.

Calibration metrics. We report two scalar metrics, both computed per (k, w, c) cell (configuration, window, channel). *Empirical coverage* is the fraction of cells whose realized value falls inside the model's own $[P_{10}, P_{90}]$ predictive interval; a calibrated model reports 80%, below indicates overconfidence and above indicates overcoverage. ***Weighted CRPS*** is $\text{wCRPS} = \sum_i \text{CRPS}_i / \sum_i |y_i|$, with CRPS_i the sample-based continuous ranked probability score of the predictive ensemble at cell i; lower is better. The two answer different questions—coverage is a property of one nominal level, wCRPS integrates over all of them—so a model can be good on one and poor on the other, and §5.3.2 reports a case where exactly that happens. Trajectory-level envelopes are straightforward to produce from the same ensemble; we report the metrics rather than the pictures because the metrics are what establish whether an interval is correctly sized.

Foundation models as fast surrogates for forward UQ. This evaluation requires an accessible data-generating process. Because ISOMORPH exposes the generating process, uncertainty over scenario configurations can be propagated through both the simulator and a forecaster. The resulting predictive spread can then be compared with the variation exhibited by the system across the same set of configurations. The protocol is applicable to any forecaster and evaluates whether its zero-shot predictive distribution reflects the variability induced by the underlying system configurations.

E.4 ADDITIONAL TSF BENCHMARKING RESULTS

Seasonal structure of the four targets. The seasonal-naive denominator $\text{SN}(m)$, reported in the last row of each block of Tables 5, 29, 30, and 31, measures the typical error of repeating the target at lag m. Its dependence on m differs substantially across the four targets. For *demand*, $\text{SN}(m)$ changes little from $m = 1$ to $m = 7$ (16.08 to 16.87) but increases substantially at $m = 30$ (27.23). *Edge utilization* shows little dependence on the lag, with $\text{SN}(m)$ varying by less than 5% across all three lags. *Backlog* exhibits the strongest lag dependence, with $\text{SN}(m)$ increasing by factors of 4.5 and 9.7 from $m = 1$ to $m = 7$ and $m = 30$, respectively, while *fill rate* shows an intermediate increase. These differences indicate distinct temporal structures across the four targets and motivate reporting MASE at all three seasonal lags.

Table 5: Univariate MASE on **demand**, mean ± 95% Student-t CI over the n=29-rollout cohort. *In-domain trained* baselines (ARIMA, ETS, PatchTST) are fit per-rollout; foundation models (Chronos, Moirai, TimesFM) run *zero-shot*. Agg. is the geometric mean across horizons.

Model	h=1	h=7	h=14	h=30	Agg.
MASE (m=1)					
ARIMA	0.756 ± 0.015	0.831 ± 0.048	0.936 ± 0.091	1.186 ± 0.185	0.914 ± 0.075
ETS	0.746 ± 0.013	0.828 ± 0.050	0.952 ± 0.111	1.280 ± 0.271	0.932 ± 0.092
PatchTST	0.756 ± 0.018	0.809 ± 0.041	0.884 ± 0.077	1.048 ± 0.164	0.867 ± 0.068
Chronos	0.774 ± 0.009	0.843 ± 0.042	0.939 ± 0.090	1.156 ± 0.199	0.917 ± 0.074
Moirai	0.792 ± 0.016	0.867 ± 0.044	0.962 ± 0.086	1.166 ± 0.176	0.937 ± 0.072
TimesFM	0.746 ± 0.014	0.817 ± 0.048	0.915 ± 0.097	1.130 ± 0.205	0.891 ± 0.080
SN(m=1) MAE	16.02	16.06	16.10	16.12	16.08
MASE (m=7)					
ARIMA	0.733 ± 0.014	0.798 ± 0.007	0.888 ± 0.031	1.108 ± 0.084	0.871 ± 0.030
ETS	0.725 ± 0.015	0.794 ± 0.008	0.897 ± 0.042	1.169 ± 0.131	0.881 ± 0.041
PatchTST	0.732 ± 0.014	0.778 ± 0.006	0.841 ± 0.025	0.980 ± 0.077	0.828 ± 0.028
Chronos	0.754 ± 0.019	0.811 ± 0.006	0.890 ± 0.030	1.070 ± 0.093	0.873 ± 0.033
Moirai	0.769 ± 0.015	0.835 ± 0.006	0.916 ± 0.028	1.091 ± 0.080	0.895 ± 0.029
TimesFM	0.725 ± 0.015	0.785 ± 0.008	0.866 ± 0.036	1.046 ± 0.098	0.847 ± 0.035
SN(m=7) MAE	16.71	16.83	16.92	17.02	16.87
MASE (m=30)					
ARIMA	0.474 ± 0.021	0.511 ± 0.013	0.563 ± 0.004	0.691 ± 0.018	0.554 ± 0.014
ETS	0.469 ± 0.021	0.508 ± 0.012	0.566 ± 0.004	0.720 ± 0.038	0.558 ± 0.018
PatchTST	0.473 ± 0.020	0.499 ± 0.014	0.535 ± 0.005	0.614 ± 0.019	0.528 ± 0.015
Chronos	0.488 ± 0.023	0.520 ± 0.015	0.564 ± 0.005	0.665 ± 0.023	0.556 ± 0.017
Moirai	0.497 ± 0.021	0.534 ± 0.015	0.582 ± 0.005	0.683 ± 0.017	0.570 ± 0.015
TimesFM	0.469 ± 0.021	0.503 ± 0.012	0.549 ± 0.002	0.650 ± 0.026	0.539 ± 0.015
SN(m=30) MAE	26.68	27.07	27.39	27.78	27.23

Table 6: Aggregate MASE (geometric mean over $h \in \{1, 7, 14, 30\}$) at m=7 on the four ISOMORPH forecast targets, mean ± 95% Student-t CI over the evaluation cohort (n=29; n=24 for backlog and fill rate, see §5.1). Bold marks the column minimum. ARIMA diverges on backlog and fill rate. Full per-horizon, per-normalizer tables are Tables 5 and 29–31.

Model	Demand	Backlog	Fill rate	Edge utilization
ARIMA	0.871 ± 0.030	7.7e+06	4.8e+06	0.798 ± 0.033
ETS	0.881 ± 0.041	4.063 ± 2.156	6.931 ± 3.811	0.798 ± 0.021
PatchTST	$\mathbf{0.828 \pm 0.028}$	4.839 ± 1.862	7.714 ± 3.027	0.886 ± 0.038
Chronos	0.873 ± 0.033	2.291 ± 1.639	2.643 ± 0.962	0.735 ± 0.016
Moirai	0.895 ± 0.029	$\mathbf{2.228 \pm 1.653}$	$\mathbf{1.473 \pm 0.721}$	$\mathbf{0.721 \pm 0.014}$
TimesFM	0.847 ± 0.035	5.545 ± 2.951	5.021 ± 3.281	0.817 ± 0.109

Dependence of forecast difficulty on the target variable. The four targets exhibit substantially different forecasting difficulty, as measured by aggregate MASE across models and horizons (Table 6). Edge utilization has the lowest aggregate MASE (0.72–0.89; Table 31), followed by demand (0.83–0.90; Table 5), while backlog and fill rate have substantially higher errors (1.5–7.7; Tables 29 and 30). This difference is consistent with the temporal structure of the targets: backlog accumulates mismatches between arrivals and demand, leading to persistent forecast errors, while fill rate is a ratio whose variability can increase when demand is small. The harder targets also exhibit greater variation across scenario rollouts. For example, the aggregate confidence intervals are wide for both backlog and fill rate, with Moirai achieving 2.228 ± 1.653 on backlog and PatchTST 7.714 ± 3.027 on fill rate (Tables 29 and 30).

Zero-shot foundation models against in-domain baselines. Zero-shot foundation models and in-domain-trained baselines show different performance across the four targets (Table 6). Among the three network-dependent targets, the zero-shot Moirai model has the lowest aggregate MASE: 0.721 for edge utilization, 2.228 for backlog, and 1.473 for fill rate. The corresponding best aggregate MASE among the in-domain-trained baselines is 0.798, 4.063, and 6.931, respectively. For demand, which is independent of the network, PatchTST has the lowest aggregate MASE (0.828), closely followed by zero-shot TimesFM (0.847). Thus, the relative performance of zero-shot foundation models and in-domain-trained baselines depends on the target.

This comparison highlights the role of ISOMORPH as a complementary benchmark to existing TSF benchmarks such as GIFT-Eval. Standard benchmarks provide fixed real-world datasets across diverse domains, whereas ISOMORPH provides a regenerable logistics-domain dataset with forecasting regimes that are substantially less well covered by current foundation models. This makes ISOMORPH a useful additional benchmark for evaluating forecasting models beyond the regimes represented in existing datasets.

E.5 FORWARD-UQ RESULTS BY TARGET

We evaluate the zero-shot predictive intervals of the three foundation models across the four targets, comparing their empirical coverage with the nominal 80% level and their weighted CRPS across horizons (Tables 7 and 8). The results differ substantially across targets:

- **Demand.** Moirai and TimesFM remain close to the nominal 80% coverage across horizons, with coverage ranging from 80.1–84.2% for Moirai and 75.6–80.1% for TimesFM (Table 7). Their weighted CRPS values are also similar, ranging from 0.048–0.079 for Moirai and 0.046–0.077 for TimesFM (Table 8). Chronos is consistently undercovered, with coverage decreasing from 69.5% to 63.0% as the horizon increases.
- **Backlog.** Coverage varies substantially across the three models and horizons (Table 7). Moirai remains overcovered at 93.0–95.6%, while Chronos is severely undercovered at 7.4–12.0%. TimesFM shows a strongly non-monotone pattern, with coverage ranging from 30.0% to 91.1%. The corresponding weighted CRPS values remain small in absolute terms (0.0020–0.0070), but do not by themselves reflect the large differences in coverage (Table 8).
- **Fill rate.** The coverage patterns are similar to those for backlog (Table 7). Moirai is consistently overcovered, with 93.0–96.8% coverage, whereas Chronos is severely undercovered at only 0.5–3.0%. TimesFM is highly non-monotone, with coverage dropping to 13.4–16.5% at $h = 7$–14 before returning to 96.8% at $h = 30$. Weighted CRPS values range from 0.0079 to 0.025 across the three models (Table 8).
- **Edge utilization.** Moirai and TimesFM remain close to the nominal coverage, ranging from 76.7–79.2% and 78.7–80.4%, respectively (Table 7). Chronos is consistently undercovered at 34.3–36.4%. The weighted CRPS values are comparable between Moirai and TimesFM (0.266–0.296), and higher for Chronos (0.295–0.312) at most horizons (Table 8).

F SUPPLEMENTARY MATERIAL FOR SECTION 6: RELEASE SUMMARY AND OUTLOOK

F.1 RELEASED BENCHMARK AND EVALUATIONS

For the released benchmark, we instantiate a 13-node U.S. network with 16 edges, daily resolution over $T = 7{,}300$ steps, and two catalogue regimes with $C = 50$ and $C = 200$ items. Six scenario knobs vary demand and network conditions around this common configuration, generating controlled variation across the resulting trajectories. The release comprises a 33-rollout scenario library, including 29 forecasting rollouts and four additional physical-realism demonstrations; 20 of the forecasting rollouts form the forward-UQ ensemble.

We use these trajectories for two main evaluations. First, we benchmark time-series forecasting with three zero-shot foundation models and three in-domain baselines across item-level and network-level targets. The results show that ISOMORPH provides a regenerable logistics-domain benchmark with

Table 7: Empirical coverage of the $[P_{10}, P_{90}]$ predictive interval (nominal 80%), averaged over $K{=}20$ scenario rollouts, mean $\pm$ 95% Student-t CI across scenarios. A calibrated model reports 80%; below indicates overconfidence (intervals too narrow), above indicates overcoverage. The forward-UQ ensemble covers these three models.

Model	$h{=}1$	$h{=}7$	$h{=}14$	$h{=}30$
demand				
Chronos	$69.5\% \pm 0.6\%$	$68.5\% \pm 0.8\%$	$66.4\% \pm 0.8\%$	$63.0\% \pm 1.1\%$
Moirai	$80.8\% \pm 0.6\%$	$84.2\% \pm 0.6\%$	$82.6\% \pm 0.4\%$	$80.8\% \pm 0.2\%$
TimesFM	$78.4\% \pm 0.5\%$	$80.1\% \pm 0.3\%$	$78.3\% \pm 0.2\%$	$75.6\% \pm 0.6\%$
backlog				
Chronos	$7.4\% \pm 5.9\%$	$10.0\% \pm 8.0\%$	$10.5\% \pm 8.5\%$	$12.0\% \pm 9.7\%$
Moirai	$95.0\% \pm 3.5\%$	$93.6\% \pm 4.1\%$	$93.2\% \pm 4.6\%$	$93.6\% \pm 4.5\%$
TimesFM	$91.1\% \pm 4.2\%$	$30.0\% \pm 14.6\%$	$31.0\% \pm 14.4\%$	$90.5\% \pm 4.7\%$
fill_rate				
Chronos	$0.5\% \pm 0.5\%$	$1.5\% \pm 0.9\%$	$2.1\% \pm 1.4\%$	$3.0\% \pm 1.9\%$
Moirai	$95.5\% \pm 2.4\%$	$93.0\% \pm 3.9\%$	$95.6\% \pm 2.1\%$	$96.3\% \pm 2.0\%$
TimesFM	$95.8\% \pm 2.2\%$	$13.4\% \pm 6.9\%$	$16.5\% \pm 8.3\%$	$96.8\% \pm 1.9\%$
edge_util				
Chronos	$36.2\% \pm 1.5\%$	$36.4\% \pm 1.4\%$	$36.0\% \pm 1.4\%$	$34.3\% \pm 1.5\%$
Moirai	$76.7\% \pm 0.9\%$	$78.8\% \pm 1.0\%$	$77.6\% \pm 1.1\%$	$79.2\% \pm 1.0\%$
TimesFM	$78.8\% \pm 1.0\%$	$80.4\% \pm 1.1\%$	$79.4\% \pm 1.2\%$	$78.7\% \pm 0.8\%$

Table 8: Weighted CRPS $\mathrm{wCRPS} = \sum_i \mathrm{CRPS}_i / \sum_i |y_i|$ of the forward-UQ ensemble ($K{=}20$ scenario rollouts), per target, model, and horizon; lower is better. Cells are aggregated over all (scenario, window, channel) triples at each horizon. The forward-UQ ensemble covers these three models.

Model	$h{=}1$	$h{=}7$	$h{=}14$	$h{=}30$
demand				
Chronos	0.047	0.060	0.079	0.118
Moirai	0.048	0.061	0.079	0.110
TimesFM	0.046	0.059	0.077	0.113
backlog				
Chronos	0.0020	0.0032	0.0044	0.0066
Moirai	0.0008	0.0026	0.0047	0.0070
TimesFM	0.0009	0.0021	0.0032	0.0055
fill_rate				
Chronos	0.0093	0.019	0.022	0.025
Moirai	0.0083	0.017	0.020	0.023
TimesFM	0.0079	0.016	0.020	0.021
edge_util				
Chronos	0.299	0.295	0.298	0.312
Moirai	0.271	0.274	0.393	0.296
TimesFM	0.270	0.266	0.266	0.273

target-dependent forecasting difficulty and controlled variation in the underlying data-generating system. Second, we propagate variation across scenario configurations through the forecasting pipeline to evaluate forward uncertainty quantification for the foundation models. This provides a direct test of whether their predictive distributions reflect variability induced by changes in the underlying system configuration.

F.2 OUTLOOK

Outlook 1— Scope of the current release. The released datasets instantiate a single 13-node U.S. network, two catalogue sizes, and one time horizon, with variation provided by the scenario library described in App. G.5. The simulator itself accepts user-supplied directed graphs and modular transition rules, so these configurations define the scope of the present benchmark rather than the generative framework as a whole. The current release therefore provides a concrete starting point for exploring larger networks, alternative topologies, and additional operational mechanisms.

Outlook 2 — Simulator extensions. The modular transition map provides several direct extensions, catalogued in App. H. Online closed-loop control and multi-destination allocation preserve the current state-space closure and would enable forecasting tasks with adaptive control and destination-specific allocation. Other extensions include richer shock processes, time-dependent congestion, vehicle scheduling, and cross-item substitution, with the latter requiring corresponding changes to the conservation structure. These extensions provide systematic directions for increasing the operational complexity and diversity of the generated trajectories.

Outlook 3 — Forecasting and uncertainty quantification. The present evaluation focuses on zero-shot forecasting and parameter uncertainty. Extending forward UQ to structural alternatives—including the demand family, shock and burst shapes, and replenishment policy—would provide a complementary form of model uncertainty. The forecasting results also motivate recalibration methods for integrating targets such as backlog and fill rate, where predictive intervals are poorly calibrated, and multivariate forecasting across the coupled network-state targets.

Outlook 4 — Foundation-model adaptation and benchmark integration. The regenerable trajectories also provide a basis for training logistics-domain forecasting models. Fine-tuning foundation models on ISOMORPH-generated trajectories can test whether exposure to the simulated domain improves forecasting and uncertainty quantification on held-out scenarios. Incorporating ISOMORPH into multi-domain suites such as GIFT-Eval would likewise extend their coverage with a regenerable logistics-domain benchmark.

G PARAMETER VALUES

This appendix lists the parameter values used to produce the released runs. The algorithms that use these parameters are in §B.2.

G.1 DEMAND-GENERATOR COEFFICIENTS

Each item i draws its demand-process coefficients independently from the distributions of Table 9. The same distributions are used at both cardinalities $C = 50$ and $C = 200$; with shared seed, the coefficients of the first 50 items at $C = 200$ coincide with those of the entire $C = 50$ release. The per-item unit volume v_i used by the packing routine of §B.2.2 is drawn from $\mathcal{U}[1.0, 4.0]$ using the same shared seed.

G.2 INVENTORY POLICY AND SOURCE LEAD TIMES

The (s, S) levels and source lead-time means (Table 10) are configuration constants of the simulator, set per node and identical across the two releases. For each (node, item) pair the realized value is drawn once at construction as $x = \text{base} + \mathcal{U}[-\text{var}, +\text{var}]$, rounded to the nearest integer; clipping ensures $0 \le s^{n,i} < S^{n,i}$ and initial inventory lies in $[s^{n,i}, S^{n,i}]$. The lead-time mean μ^n enters the source (s, S) rule at $n \in \mathcal{N}_{\text{src}}$; at intermediate nodes the realized arrival time is determined by the routing and packing of sub-steps 5–6 in Algorithm 1.

G.3 EDGE TRANSPORT PARAMETERS

Each edge $e = (u, v)$ has a transit time τ_e in time units, a per-container volume V_e in the same units, and a per-step container count K_e, giving a per-step volume capacity $C_e = K_e V_e$. The two

Table 9: Distributions of the demand-generator coefficients of Eq. (14). All coefficients are drawn once per run from the distributions below. The notation $\mathcal{U}\{a, \dots, b\}$ denotes the discrete uniform on $\{a, a+1, \dots, b\}$.

Symbol	Role	Distribution
$\bar{\lambda}_i$	baseline per-item rate	$\mathcal{U}[80, 250]$
$a_{1,i}^{(y)}$	yearly 1st-harmonic amplitude	$\mathcal{U}[0.12, 0.28]$
$a_{2,i}^{(y)}$	yearly 2nd-harmonic amplitude	$\mathcal{U}[0.04, 0.10]$
$a_i^{(w)}$	weekly amplitude	$\mathcal{U}[0.04, 0.10]$
ϕ_i, ψ_i	seasonal phases	$\mathcal{U}[0, 2\pi]$
ϕ_i^{AR}	AR(1) coefficient	$\mathcal{U}[0.9990, 0.9996]$
$\sigma_{\mathrm{AR},i}$	AR(1) innovation scale	$\mathcal{U}[0.008, 0.018]$
$A_i(0)$	AR(1) initial value	$\mathcal{N}(0, 0.10^2)$
r_i	per-item burst rate	$\mathcal{U}[2 \cdot 10^{-4}, 10^{-3}]$
$\Delta_{i,k}^P$	per-item burst duration	$\mathcal{U}\{30, \dots, 179\}$
$h_{i,k}^P$	per-item burst height	$\mathcal{U}[0.20, 0.70]$
N	macro-shock event count	$\mathcal{U}\{5, \dots, 11\}$
t_k^G	macro-shock event start	$\mathcal{U}\{0, \dots, T-1\}$
Δ_k^G	macro-shock event duration	$\mathcal{U}\{180, \dots, 1099\}$
h_k^G	macro-shock event height	$\mathcal{U}[0.20, 0.60]$
g_i	per-item macro-shock sensitivity	$\mathcal{U}[0.4, 1.2]$

Table 10: Per-node inventory parameters (base $\pm$ var) and source lead-time means μ^n. Drawn once per (node, item) at simulator construction. "LM" marks last-mile DCs.

Node	Tier	Initial inv.	s	S	μ^n (days)
SanFrancisco	Source	4000 ± 400	400 ± 60	4000 ± 400	3 ± 1
St. Louis	Source	4000 ± 400	400 ± 60	4000 ± 400	3 ± 1
Orlando	Source	4000 ± 400	400 ± 60	4000 ± 400	3 ± 1
Nashville	Hub	8000 ± 800	1000 ± 150	8000 ± 800	3 ± 1
Atlanta	Tier-2	6000 ± 600	500 ± 80	6000 ± 600	1
Chicago	Tier-3	5000 ± 500	1000 ± 150	5000 ± 500	8 ± 1
Charlotte	Tier-3	5000 ± 500	1000 ± 150	5000 ± 500	7 ± 1
Memphis	Tier-3	3000 ± 300	500 ± 80	3000 ± 300	7 ± 1
Columbus	Tier-4	4000 ± 400	500 ± 80	4000 ± 400	2
Richmond	Tier-4	4000 ± 400	500 ± 80	4000 ± 400	2
Philadelphia	Tier-5 (LM)	3000 ± 300	500 ± 80	3000 ± 300	1
Baltimore	Tier-5 (LM)	3000 ± 300	500 ± 80	3000 ± 300	2

releases share τ_e and $K_e = 3$ on every edge; V_e scales by a factor of four on the fourteen upstream edges (Table 11). The two last-mile edges Philadelphia→NewYork and Baltimore→NewYork are back-solved at runtime; the formula is given in Appendix G.4.

G.4 Last-mile back-solve

Let $\bar{\lambda}$ denote the realized mean intensity of the run, $\bar{\lambda} := (CT)^{-1} \sum_{i,t} \lambda_{i,t}$, computed after the demand tensor is sampled. The target combined per-step volume for the two last-mile edges, with a buffer and an allowance for wasted packing space, is

$$V_{\mathrm{raw}} = \frac{C\,\bar{\lambda}\,\bar{v}\,\rho}{\eta}, \qquad \bar{v} = 2.5,\ \rho = 1.20,\ \eta = 0.93, \tag{34}$$

where $\bar{v}$ is the midpoint of the per-item volume distribution $\mathcal{U}[1.0, 4.0]$ (used in place of the realized sample mean to keep the back-solve deterministic in the demand tensor), ρ is a target load factor giving 20% headroom above the mean, and η accounts for greedy first-fit waste. The split between

Table 11: Per-edge transport parameters at the two cardinalities $C \in \{50, 200\}$. τ_e in time units, V_e in unit-equivalent volume, K_e per-step container count. Last-mile edges are back-solved (§G.4).

Edge	τ_e	V_e (C=50)	V_e (C=200)	K_e
SanFrancisco → Nashville	4	5,000	20,000	3
St. Louis → Nashville	2	5,000	20,000	3
Orlando → Nashville	2	5,000	20,000	3
Nashville → Atlanta	1	15,000	60,000	3
Atlanta → Chicago	8	4,000	16,000	3
Atlanta → Charlotte	7	4,000	16,000	3
Atlanta → Memphis	7	4,000	16,000	3
Chicago → Columbus	2	4,000	16,000	3
Charlotte → Richmond	2	4,000	16,000	3
Columbus → Philadelphia	2	4,000	16,000	3
Richmond → Philadelphia	1	4,000	16,000	3
Richmond → Baltimore	3	3,000	12,000	3
Columbus → Baltimore	3	3,000	12,000	3
Memphis → Baltimore	2	3,000	12,000	3
Philadelphia → NewYork	1	back-solved	back-solved	3
Baltimore → NewYork	2	back-solved	back-solved	3

the two edges is fixed at $0.55 : 0.45$, after which each share is divided by $K_e = 3$ and rounded to the nearest 100 to give the integer per-container volumes. The values of ρ and η are engineering choices, held fixed across the two releases; varying them is not part of this work.

G.5 Scenario sweeps

The baseline release uses the default values of every scenario knob in Tables 9–11. To study how the network's behavior and the resulting time series change under different operating conditions, we construct six one-at-a-time sweeps: each varies a single knob across five settings—a shared baseline (all knobs at default) plus four perturbations—while holding all other knobs at baseline, producing $6 \times 5 = 30$ rollouts on the C=50 regime.

Table 12 lists all six sweeps; the baseline value is bold in each row. Three sweeps perturb the demand process and three perturb the network's response:

- *Drift*: sets the AR(1) persistence to a single shared value ϕ^{AR} across all items (Eq. 17); baseline 0.9993.
- *Shock*: multiplies the macro-shock event count N and height h^G (Eq. 19); baseline $(1, 1)$.
- *Burst*: multiplies the per-item burst rate r_i and height h^P (Eq. 18); baseline $(1, 1)$.
- *Edge cap*: multiplies the per-edge container count K_e; baseline 1.
- *Buffer*: multiplies the per-node reorder thresholds (s, S) and initial inventory; baseline 1.
- *Lead time*: multiplies the source-node mean μ^n; edge transit times τ_e unchanged; baseline 1.

We select five demand-side scenarios from the six sweeps of Table 12; three are single-knob picks and two are compound scenarios that combine perturbations along multiple axes:

- `shock_xhi`: $(3, 4)$ from the shock sweep.
- `drift_mid`: $\phi^{\mathrm{AR}} \in [0.95, 0.97]$ from the drift sweep.
- `burst_xhi`: $(3, 4)$ from the burst sweep.
- `chaos_compound`: `shock_xhi` with $\phi^{\mathrm{AR}} \in [0.96, 0.98]$.
- `chaos_burst`: `burst_xhi` with $\phi^{\mathrm{AR}} \in [0.96, 0.98]$.

Supply-side sweeps leave the demand series unchanged and therefore cannot propagate to forecasts of $y_{i,t}$; we restrict the evaluation to demand-side knobs. The evaluation protocol matches §5.1 throughout.

Table 12: The six scenario sweeps. Each sweep varies a single knob across five settings while all others stay at baseline (bold). The Type column indicates whether the sweep changes the demand process or the network's response to it. Details of each sweep are in the text.

Sweep	Notation	Type	Settings **(baseline)**
Drift	ϕ^{AR}	demand	$0.71,\ 0.86,\ 0.96,\ 0.99,\ \mathbf{0.9993}$
Shock	$(N \times h^G)$	demand	$(0,1),\ (0.5,0.7),\ (\mathbf{1},\mathbf{1}),\ (2,2),\ (3,4)$
Burst	$(r \times h^P)$	demand	$(\mathbf{1},\mathbf{1}),\ (1.5,2),\ (2,3),\ (3,4),\ (5,8)$
Edge cap	ρ_K	supply	$0.3,\ 0.6,\ \mathbf{1.0},\ 1.5,\ 2.5$
Buffer	ρ_{sS}	supply	$0.1,\ 0.2,\ 0.5,\ 0.75,\ \mathbf{1.0}$
Lead time	ρ_{lt}	supply	$\mathbf{1.0},\ 2.0,\ 5.0,\ 10.0,\ 20.0$

G.6 Runtime parameters

The two releases were produced by running the simulator script with the parameters of Table 13. All values are the script's built-in defaults, except for the pipeline multiplier m. The default $m = 0$ selects a reactive shipping rule that ships against backlog plus a three-time-unit buffer of smoothed demand; $m = 7$ selects the proactive rule used in this work, which keeps seven time units of smoothed demand in the pipeline at all times.

Table 13: Runtime parameters used to produce the two releases. The pipeline multiplier $m = 7$ is the single value that differs from the script's defaults.

Parameter	Value
Horizon T	7,300 (daily steps)
Seed	2025
Pipeline multiplier m	7.0 (default 0.0)
Packing	greedy first-fit
Streaming output	enabled

H Extending the base transition map

Algorithm 1 fixes seven sub-steps, but not their contents. Because each sub-step reads and writes the state vector ξ_t of Eq. (1), a modified sub-step yields a valid simulator as long as it leaves that vector's semantics intact. Two properties are worth tracking separately when extending: the *Markov closure* of §3.1—whether ξ_{t+1} still depends on the past through ξ_t and the external inputs alone—and the *conservation laws* of Proposition C.1, which additionally require that no unit and no demand event is created or destroyed. Closure can survive a change that conservation does not, and vice versa; the two are not interchangeable, which is why Tables 14 and 15 report them as one column with per-row qualifications rather than as a single verdict.

Dynamics available in the release. Beyond the base configuration, five further dynamics are implemented and exposed as command-line flags, each with an archived example rollout under `data/output_mixture/` (Table 14). Four of them add a stochastic mechanism to the transition function; the fifth is a deterministic scenario control applied to an otherwise unchanged mechanism, which is why it is described separately below.

- *Non-infinite source supply* (`reservoir_tight`). Flags `-prod_rate_per_item` and `-reservoir_cap_per_item` replace unbounded source inventory with a bounded reservoir replenished at a fixed rate. The reservoir level joins the state as an additional per-(source, item) coordinate, so closure is preserved; conservation is preserved because units leaving the reservoir are exactly the units entering the transport system. Emits `reservoir_history` alongside the standard files.

- *Finite warehouse capacity* (`warehouse_cap2x`). Flag `-warehouse_cap_scale` caps per-node on-hand inventory. Arrivals exceeding the cap are held on the inbound edge rather than discarded; this choice is what preserves Proposition C.1. A spillover policy that discarded excess would keep the Markov property but break conservation.
- *Non-deterministic edge transit* (`tt_noise10`). Flag `-edge_tt_std_frac` makes the realized transit time on edge e a random draw around τ_e. Closure is preserved by treating the draw as an external random input alongside L_t (§B.1.1); the conservation laws are untouched, since they do not reference transit duration.
- *Supplier disruption* (`disruption_mid`). Flags `-disruption_p_onset` and `-disruption_length_days` switch source availability off for a sampled window. Both closure and conservation are unaffected: an unavailable source simply places no orders. Emits `source_availability`.

A fifth capability, *dynamic edge cut and restore*, relaxes the fixed topology assumption and is used to generate the disruption scenarios of §4.2. *Item-specific shipping volumes* are not an extension at all—they are part of the base model, since each item carries its own unit volume v_i into the packing step of Algorithm 2.

Further dynamics and their effect on the state-space design. Table 15 lists the dynamics that users and reviewers have asked for and that the released configuration does not implement. For each it reports the implementation effort we estimate and, in the column that carries the actual content, whether closure and the conservation identities would survive. That column is what a user needs before modifying the simulator. A row marked ✓ can be added and the result still checked against Proposition C.1, so the verification tools of §3.2 continue to apply. The row marked ×—substitution across items—changes the accounting itself: it couples demand events between items, so (31) must be restated before any check of a modified implementation means anything. Three rows are conditional, and each note states the condition rather than leaving it to the effort tier. The whole classification is a design argument, not a measurement: the effort tiers are our estimate, and the closure column reasons about the state vector rather than reporting a verified property of code that does not yet exist.

I Dataset schema

I.1 Release overview

Directory structure. A release contains one directory, `output_item50/` or `output_item200/`, containing six CSV files, one NumPy array, and one short text sidecar. Both releases use the same schema; they differ in C and in the row counts of files that scale with C. The scenario library is a sibling tree, `output_mixture/<scenario>/`, whose rollouts use this same schema; App. I.3 maps each directory to the experiment that reads it.

Time index. The time index t runs over $\{0, \dots, T-1\}$ with $T = 7{,}300$ in both releases. Each file stores the index as an integer column named `day`; the column name reflects the resolution of the released runs (one step is one day). The simulator itself accepts any user-chosen step length, and at other resolutions the column carries the same integer index of t.

Item identifiers. Items are zero padded strings: `I01–I50` at C=50 and `I001–I200` at C=200.

Sidecar metadata. The release also contains `demand_signals_cols.txt`, a one-line text file listing the item identifiers in the column order used by `demand_signals.npy`.

I.2 File schemas

File inventory. Table 16 lists every file in a release, the columns that uniquely identify a row, and the number of rows or the array shape. The paragraphs that follow give the columns of each file, in the order of the table.

Table 14: Dynamics available in the released simulator. "Closure" asks whether the Markov property of §3.1 and the conservation identities of Prop. C.1 survive; "✓*" marks a conditional answer explained in the note. Each row below the first has an archived example rollout under `data/output_mixture/`.

Dynamic	Example run	Closure	Note
Item-specific shipping volumes	(base model)	✓	Per-item unit volume v_i enters the packing step (Algorithm 2); not an extension but a property of the base configuration.
Non-infinite source supply	`reservoir_tight`	✓	Bounded source reservoir; relaxes unbounded upstream supply.
Finite warehouse capacity	`warehouse_cap2x`	✓*	Conservation holds because excess arrivals are retained in the transport system; a policy that discarded them would keep closure but break (30).
Non-deterministic edge transit	`tt_noise10`	✓	Stochastic travel times augment the external random inputs; relaxes deterministic edge transit.
Supplier disruption	`disruption_mid`	✓	On/off source availability; state and accounting unchanged.
Dynamic edge cut/restore	`ripple_scenario`, `bottleneck_scenario`	✓	Relaxes stationary topology; generates the disruption dynamics of §4.2.

`daily_records.csv`. One row per (t, i) pair, recording per-item demand and service at $d^\star$. The two `*_end_before_ship` columns are the destination's on-hand and backlog snapshots taken between sub-steps (4) and (5) of Algorithm 1 — after demand and service but before any dispatch. So that `served_from_stock` + `new_backlog_today` = `demand`.

`shipments.csv`. One row per dispatch committed by GREEDYPACK in sub-steps (5) and (6) of Algorithm 1. A single $(t, \text{from}, \text{to}, i)$ combination may appear in multiple rows, since round-robin dispatch can revisit an item within the same step.

`inventory_history.csv`, `backlog_history.csv`, `intransit_history.csv`. Three long form files record the network state at the end of each step. All three share the key columns (`day`, `node`, `item`) of types (int, str, str) and differ in the value column they carry (Table 19). `inventory_history.csv` and `backlog_history.csv` cover every node

Table 15: Extension pathways. For each dynamic not exercised by the released configuration: implementation effort (a coarse tier, not a time estimate) and whether the Markov property of §3.1 and the conservation identities of Prop. C.1 are preserved.

Dynamic	Effort	Closure	Note
Non-Dijkstra routing	small–large	✓	Preserved for any *fixed* routing policy; the routing module is already swappable. Effort scales with the alternative, from k-shortest paths through min-cost flow to learned routing.
Online closed-loop control	small	✓*	Preserved if the controller is a function of the current state; a controller with memory requires state augmentation. This is the one standing assumption of §B.1.3 still in force in every released run.
Multi-destination allocation	moderate	✓	Backlog becomes per-destination; (29)–(31) generalize directly.
Time-dependent congestion	moderate	✓*	Transit time becomes a function of current edge load; preserved for dependence on the current state, fails for dependence on history.
Vehicle scheduling	large	✓*	Adds vehicle-position state; conservation holds, but the state dimension grows with fleet size.
Substitution / cannibalization	large	×	Cross-item demand coupling breaks the per-item demand-event accounting of (31); the backlog identity would need restating.

Table 16: File inventory of one ISOMORPH release. "Key" is the combination of columns that uniquely identifies a row. "Rows" is given as a function of the cardinality $C \in \{50, 200\}$, the horizon $T = 7{,}300$, and the node count $|\mathcal{N}| = 13$.

File	Key	Rows / shape
`daily_records.csv`	(day, item)	$T \cdot C$
`shipments.csv`	none (event log)	variable (4.90×10^5 at C=50; 1.80×10^6 at C=200)
`inventory_history.csv`	(day, node, item)	$T \cdot \lvert\mathcal{N}\rvert \cdot C$
`backlog_history.csv`	(day, node, item)	$T \cdot \lvert\mathcal{N}\rvert \cdot C$
`intransit_history.csv`	(day, item)	$T \cdot C$ (destination only)
`service_summary.csv`	item	C
`demand_signals.npy`	—	(T, C)
`demand_signals_cols.txt`	—	1 line (C item identifiers)

Table 17: Schema of `daily_records.csv`.

Column	Type	Description
`day`	int	time index t
`item`	str	item identifier
`demand`	int	realized demand $y_{i,t}$ (Poisson sample, Eq. 3)
`served_from_stock`	int	$\min(\mathrm{OH}_t^{d^\star,i}, y_{i,t})$
`new_backlog_today`	int	$(y_{i,t} - \mathrm{OH}_t^{d^\star,i})_+$
`dest_on_hand_end_before_ship`	int	destination on-hand right after service, before sub-step (5)
`dest_backlog_end_before_ship`	int	destination backlog right after service, before sub-step (5)

$n \in \mathcal{N}$; `intransit_history.csv` restricts to the destination $d^\star$, so its `node` column is constant. Backlog is zero at every node other than $d^\star$ by construction (§B.1.3) and is recorded as zero for schema regularity. All three values are taken at the end of step t, after all seven sub-steps of Algorithm 1 have run.

`service_summary.csv`. One row per item with five columns: `item` (str), `total_demand` (int), `served_from_stock` (int), `new_backlog_added` (int), and `fill_rate_stock_only` (float). The three integer columns aggregate the corre-

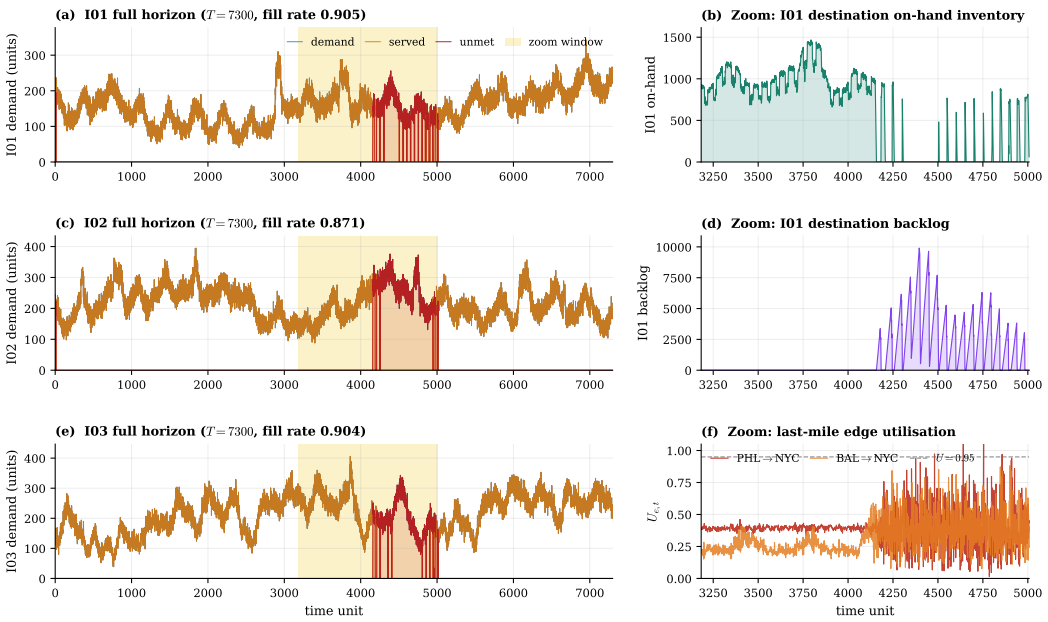


Figure 10: Baseline release at $C{=}50$, raw values at each daily time unit over the full horizon $T{=}7{,}300$. **Left column (a, c, e):** for items I01–I03, the daily demand, the part served immediately from destination inventory, and the part left unmet (red, shaded). Per-item fill rates over the whole horizon (0.905, 0.871, 0.904) appear in each panel title; the yellow band marks the five-year zoom window (days 3183–5008) selected as the window of largest rolling aggregate demand. **Right column (b, d, f):** the network's response for item I01 inside that window — **(b)** destination on-hand inventory, **(d)** destination backlog, and **(f)** utilization of the two last-mile edges (Philadelphia→NewYork, Baltimore→NewYork), with the $U{=}0.95$ saturation line marked. The three panels read together: on-hand collapses to zero from roughly day 4150, backlog rises in the same window to a peak near 10^4 units, and last-mile utilization becomes volatile rather than saturated, since there is no stock upstream to ship.

Table 18: Schema of `shipments.csv`.

Column	Type	Description
`day`	int	dispatch step t
`arrival_day`	int	arrival step at `to`, $t + \lceil \sum_{e \in P} \tau_e \rceil$
`from`	str	origin node of the dispatch
`to`	str	path endpoint
`item`	str	item identifier
`units`	int	units dispatched on this row
`path_nodes`	str	repr of `list[str]`: resolved Dijkstra path $P = (v_0, \ldots, v_k)$
`edge_times`	str	repr of `list[float]`: per-edge transit times $(\tau_{e_1}, \ldots, \tau_{e_k})$ in time units

sponding columns of `daily_records.csv` over the full horizon; the float column is `served_from_stock` divided by `total_demand`, rounded to six decimal places. The file is derived from `daily_records.csv` and included for convenience.

`demand_signals.npy`. It is a `float64` array of shape (T, C) holding the intensity values $\lambda_{i,t}$ used to sample the `demand` column of `daily_records.csv` (Eq. 14); pairing samples with rates lets users study sample-versus-rate behavior without rerunning the simulator. Columns are indexed by item identifier in alphabetical order, listed comma-separated on a single line in `demand_signals_cols.txt`.

Table 19: Value column of each history file. All three share the key (`day`, `node`, `item`).

File	Value column	Type	Description
`inventory_history.csv`	`on_hand`	int	on-hand inventory $\mathrm{OH}_t^{n,i}$
`backlog_history.csv`	`backlog`	int	backlog $B_t^{n,i}$ (zero unless $n = d^\star$)
`intransit_history.csv`	`in_transit`	int	total units of item i already scheduled to arrive at $d^\star$ on any step after t: $\sum_{\tau>t} \mathrm{IT}_t[\tau, i]$

I.3 Data directories and their associated experiments

Table 20 summarizes the released data directories and their corresponding experiments and analyses. The directory structure contains two points that are not evident from the directory names alone.

The forward-UQ ensemble does not have a separate data directory. It uses a $K{=}20$ subset of the same 29 scenario rollouts used for the forecasting evaluation (§5.2). Thus, no additional rollouts are generated specifically for forward UQ; the two experiments use the same scenario files but aggregate them differently.

Four of the 33 *rollouts are demonstrations rather than evaluation data.* The `disruption_mid`, `reservoir_tight`, `warehouse_cap2x`, and `tt_noise10` rollouts each activate one physical-realism extension (App. H). These rollouts are excluded from the forecasting cohort because the edge arrays required for the edge-utilization target are generated by a separate derivation step, which is applied to the two edge-cut rollouts but not to these four extension demonstrations.

Table 20: Released data directories and their associated experiments and analyses. The 33 rollouts in `output_mixture/` comprise the 29-rollout forecasting cohort and four extension demonstrations. The forward-UQ ensemble is a subset of the forecasting cohort rather than a separate set of rollouts.

Directory	Rollouts	Read by
`output_item50/`, `output_item200/`	2	Baseline releases at $C{=}50$ and $C{=}200$; the schema of App. I and the overview of Fig. 10.
`output_mixture/ <sweep>/seed2025/`	27	Baseline plus the 24 one-knob variants of App. G.5 and the two compound-stress scenarios.
`output_mixture/ ripple_scenario`, `bottleneck_scenario`	2	Dynamic edge-cut rollouts; the disruption dynamics of §4.2.
The 29 *rollouts above are the forecasting evaluation cohort of §5.1;* 20 *of them are the forward-UQ ensemble of §5.2.*		
`output_mixture/ disruption_mid`, `reservoir_tight`, `warehouse_cap2x`, `tt_noise10`	4	One physical-realism extension active in each (Table 14); outside the cohort.
`network_state_ forecast_datasets/ <rollout>_<target>/`	—	Target-recast copies written by the extraction step for the three network-state targets; read by the forecasting drivers of §5.1.

The scenario directories are not uniformly structured. The 27 sweep rollouts store their files under a `seed2025/` leaf, whereas the two edge-cut rollouts and four extension rollouts store their files at the top level. Accordingly, data loaders must be pointed to the directory containing the corresponding files. The extension rollouts additionally contain `reservoir_history` and `source_availability`, which are not written by the base configuration.

J DETAILED COMPARISON WITH RELATED SIMULATORS

Table 21 extends the high-level comparison in Table 3 (§A.2) to the structural and dynamical details of each simulator, including demand processes, ordering and control, lead times, capacity and routing, conservation properties, and scenario generation.

Table 21: Structural comparison of ISOMORPH with three related open supply-chain simulation and learning frameworks.

Axis	**ISOMORPH** (this work)	**BULL-ODE** (Naik et al., 2025)	**deepbullwhip** (Arief, 2026)	**OR-Gym** (Hubbs et al., 2020)
Demand process	Poisson: yearly and weekly seasonality, AR(1) drift, per-item trapezoidal bursts, and shared macro-shock $G(t)$ (Eq. (14))	AR(1) (ϕ=0.6), i.i.d. Gaussian, or lognormal multiplicative	AR(1) with weekly seasonality and structural shocks; real-data replay	Stationary Poisson (mean 20)
Ordering / control	(s, S) at every non-destination node; proactive shipping based on demand estimates and pipeline conditions	Continuous linear order-up-to embedded in the ODE (τ=5, α=0.8)	Pluggable: OUT, POUT, Smoothing OUT, Constant	Base-stock (IPA / DFO / MIP), shrinking-horizon LP, PPO
Lead times	Stochastic source replenishment; realized path transit time for intermediate shipments; deterministic edge transit by default, with stochastic-transit extension	Deterministic per-stage lead time	Deterministic per-stage lead time	Deterministic per-stage lead time
Capacity / routing	Edge-level transportation capacity; capacity-aware shortest-path routing	—	Uncapacitated upstream (default)	Per-stage production capacity
Conservation enforcement	Pathwise conservation laws over physical units and demand events (Prop. C.1); continuous-time fluid limit (§C.2)	Inventory conservation $\dot{I}=O-D$ hard-coded into the UDE residual	—	Material balances; no explicit conservation-law framing

Simpler supply-chain configurations can be obtained as reductions of ISOMORPH, allowing direct comparison with the serial-chain settings of deepbullwhip and OR-Gym. In particular, restricting the catalogue to a single item and the network to a serial chain preserves the following structural distinctions.

- **Path-dependent lead times.** Source replenishment is stochastic, while intermediate shipments incur the realized transit time along their selected paths. This contrasts with the deterministic per-stage lead times used by the other frameworks.

- **Edge capacity and routing.** Transportation capacity is represented on edges rather than solely at nodes, allowing lane-level bottlenecks to constrain throughput and influence path selection without introducing additional dynamic stock variables.
- **Pathwise conservation.** Material flow and backlog satisfy explicit pathwise conservation laws, enforcing trajectory-level accounting across the network rather than relying on local inventory balances.
- **Proactive shipping.** Outbound shipping responds to demand estimates and pipeline conditions, rather than to currently observed demand alone. An exponentially weighted demand estimate provides this history dependence without requiring a full demand window in the dynamic state.

These choices introduce network-level behavior while keeping the dynamic state compact and Markovian. BULL-ODE remains a distinct reference point because its continuous-time single-node formulation does not instantiate a finite transportation network.

K Statistical Comparison with SupplyGraph

SupplyGraph (Wasi et al., 2024) provides observations from a consumer goods company across four supply-chain stages—Sales Order, Delivery to Distributor, Factory Issue, and Production—covering $C = 41$ SKUs over 221 days. The four stages contain nearly the same aggregate volume ($\sim 7.65\times10^6$ cases), indicating that they largely represent the same material flow observed at different points in the supply chain. The agreement is not uniform across SKUs: 11–12 of the 41 differ by more than 25% between Sales Order and a given downstream stage, so part of the aggregate match is compensation across SKUs. Since SupplyGraph is an observational release and does not expose its underlying data-generating process or network dynamics, we use it as a real-world statistical reference rather than as a mechanistic validation target.

Three SupplyGraph stages have direct counterparts in ISOMORPH: Sales Order is compared with destination demand, Delivery to Distributor with shipments into destination nodes, and Factory Issue with shipments from source nodes. Production is excluded because the base configuration of ISOMORPH uses unbounded source supply. To account for the different units and scales of the two datasets, the comparison uses dimensionless distributional and temporal statistics computed over matched 221-day windows. SKUs with insufficient nonzero observations are excluded from the SupplyGraph statistics.

Table 22 summarizes marginal, intermittency, and temporal statistics for the three comparable stages. Each entry reports the SupplyGraph median and the range across the 29 ISOMORPH rollouts, with ✓ indicating that the observed value falls within the simulator range. Table 23 reports cross-stage amplification and propagation timing.

Across the 39 reported properties, 15 are contained within the ISOMORPH scenario range. The coverage is strongest at the Delivery stage, where all reported properties are contained, while the Sales Order and Factory Issue stages show systematic differences in variability, intermittency, and temporal dependence. Weekly aggregation improves agreement for several upstream statistics. The cross-stage comparison shows that ISOMORPH reproduces the observed propagation timing, although its bullwhip ratios are consistently higher than those observed in SupplyGraph at the resolutions considered.

These results should be interpreted as a statistical coverage check for one observed supply chain. Containment indicates that a corresponding statistical regime is reachable within the ISOMORPH scenario library; it does not imply that the released baseline reproduces SupplyGraph or that the observed range represents the broader population of supply chains. Nor can such a comparison establish more than this: SupplyGraph does not record the topology, capacities, lead times, or ordering rules that generated its series, and many different configurations of these inputs produce the same output statistics. The mismatches in Table 22 are of two kinds. The higher variability and intermittency of the observed Sales Order series relative to the released baseline (zero fraction 0.21 against 0; coefficient of variation 1.03 against $[0.16, 0.44]$) is a property of the Poisson demand module (§B.1.2), which a user can replace; it is the model-uncertainty axis of §5.2. The disagreement in the daily and weekly bullwhip ratios of Table 23 reflects container-level shipment batching, which

Table 22: Statistical comparison of ISOMORPH against SupplyGraph (Wasi et al., 2024). Each cell reports the SupplyGraph median and the range across the 29 ISOMORPH rollouts. A ✓ indicates that the observed value lies within the simulator range. All statistics are dimensionless and computed on matched 221-day windows.

Property	**Sales Order**	**Delivery**	**Factory Issue**
Marginal shape			
Coefficient of variation	1.03 / [0.16, 0.44] ×	0.74 / [0.20, 6.59] ✓	0.92 / [1.66, 4.94] ×
CV of nonzero values	0.79 / [0.16, 0.44] ×	0.69 / [0.20, 1.12] ✓	0.79 / [0.00, 0.64] ×
Skewness	1.44 / [0.11, 0.31] ×	1.04 / [0.15, 6.68] ✓	1.52 / [1.65, 5.06] ×
Intermittency			
Zero fraction	0.21 / [0.00, 0.00] ×	0.04 / [0.00, 0.96] ✓	0.12 / [0.68, 0.96] ×
Average demand interval (d)	1.26 / [1.00, 1.00] ×	1.04 / [1.00, 22.33] ✓	1.12 / [3.12, 23.00] ×
Temporal dependence			
Lag-1 autocorrelation	0.15 / [0.71, 0.98] ×	0.41 / [0.03, 0.64] ✓	0.35 / [-0.20, -0.03] ×
Lag-7 autocorrelation	0.46 / [0.72, 0.89] ×	0.21 / [-0.05, 0.64] ✓	0.11 / [-0.10, 0.05] ×
Weekly spectral power ratio	7.49 / [0.36, 4.80] ×	3.32 / [0.06, 5.71] ✓	2.28 / [0.19, 0.79] ×
After weekly aggregation			
Coefficient of variation	0.45 / [0.13, 0.42] ×	0.43 / [0.16, 2.61] ✓	0.54 / [0.56, 1.93] ×
Skewness	0.62 / [0.05, 0.36] ×	0.57 / [0.08, 2.33] ✓	0.45 / [0.15, 1.74] ✓
Lag-1 autocorrelation	0.36 / [0.92, 0.95] ×	0.39 / [-0.15, 0.88] ✓	0.37 / [-0.20, 0.39] ✓

Table 23: Cross-stage statistics. Bullwhip is the ratio of upstream to downstream squared coefficients of variation; values above one indicate amplification. Lead-lag is the lag maximizing the cross-correlation between the two stages.

Property	**SupplyGraph**	**ISOMORPH range**	
Bullwhip, delivery / sales (daily)	0.48	[1.38, 1454.92]	×
Bullwhip, delivery / sales (weekly)	1.02	[1.20, 313.98]	×
Bullwhip, factory / sales (daily)	0.79	[65.03, 944.53]	×
Bullwhip, factory / sales (weekly)	1.44	[10.58, 198.33]	×
Lead-lag, sales to delivery (d)	1.00	[0.00, 2.00]	✓
Lead-lag, sales to factory (d)	1.00	[-1.00, 3.00]	✓

dominates variance ratios at sub-monthly resolution (compare the daily and monthly columns of Table 1); these ratios also differ in definition (squared coefficient of variation, cross-stage, 221-day windows) from the per-node monthly variance ratio of §L.1, and the two are not directly comparable.

L REPRODUCING KNOWN SUPPLY-CHAIN BEHAVIORS WITH ISOMORPH

L.1 BULLWHIP METHOD

For each non-source node n and item i, we compute Cachon's amplification ratio(Cachon et al., 2007):

$$B_{n,i} = \frac{\mathrm{Var}\big(\mathrm{inflow}_{n,i,\cdot}\big)}{\mathrm{Var}\big(\mathrm{outflow}_{n,i,\cdot}\big)},$$

where $\mathrm{inflow}_{n,i,t}$ is the units of item i arriving at node n at time unit t and $\mathrm{outflow}_{n,i,t}$ is the units shipped out of n at the same time unit. At the destination, there is no downstream node; we use the realized customer demand $y_{i,t}$ as the boundary outflow, so $B_{\text{dest},i}$ measures how much the inbound replenishment stream amplifies the demand it is meant to serve. A ratio $B_{n,i} > 1$ indicates the classical bullwhip signature — variability grows as one moves upstream from the customer; $B_{n,i} = 1$ is no amplification, and $B_{n,i} < 1$ means n smooths variability. The three source nodes (§B.1.3)

replenish via a magic-stock mechanism rather than by inflow from the released network, so $B_{n,i}$ is not defined for them and they are excluded.

Both series are aggregated into 30-time-unit totals — matching the monthly granularity at which Cachon reports bullwhip ratios — before computing variance, after dropping the first 365 time units as a warm-up. The per-node value reported in Table 1 is the mean of $B_{n,i}$ over the C items at that node; the per-tier value is the mean over nodes in the tier.

L.2 PER-NODE BULLWHIP RATIOS

Table 1 in §4.2 reports tier-level means. For reproducibility, Table 24 below gives the per-node values that those tier means are computed from. Each entry is the mean of $B_{n,i}$ over the 50 items at node n, using the procedure described in §4.2; tier names match Table 10.

Table 24: Per-node bullwhip ratio $\bar{B}_n$ on the C=50 release. Each value is the mean over the items at that node, averaged over the 29-rollout cohort under the rules of §L.1; the same numbers averaged within tier appear in Table 1. The wide spread between Baltimore and Philadelphia in the daily column reflects sub-monthly batching of arrivals, which the monthly column smooths out.

Node	Tier	Daily $\bar{B}_n$	Monthly $\bar{B}_n$
NewYork	Destination	22.64	1.93
Baltimore	Tier-5	5.79	1.49
Philadelphia	Tier-5	19.56	1.41
Columbus	Tier-4	1.35	1.23
Richmond	Tier-4	1.36	1.26
Charlotte	Tier-3	1.14	1.31
Chicago	Tier-3	1.09	1.17
Memphis	Tier-3	1.28	1.18
Atlanta	Tier-2	1.30	1.14
Nashville	Hub	0.97	1.07

L.3 QUANTITATIVE DETAILS FOR OBSERVABLE PHENOMENA

This appendix provides quantitative details supporting the phenomenon-level figures in §4.2. The reported values are computed directly from the released trajectories and provide additional measurements for the inventory oscillation, high edge-utilization, disruption, ripple and cascading disruption, and bottleneck migration figures. The figures remain the primary visual presentation of these phenomena; the tables below provide the corresponding numerical summaries and exact values where useful.

Additional quantitative measurements. For inventory oscillation, the detrended $\mathrm{AC}(7)$ of on-hand inventory across the scenario library ranges from -0.00 to 0.98, providing the numerical range underlying Figure 6. For high edge utilization, the fraction of days with $U_{e,t} \geq 0.8$ ranges from the baseline values reported in Figure 7 to substantially larger values under uniform capacity reduction. The complete edge-level values are shown in the corresponding figure and its underlying released trajectories.

For disruption and delivery lateness, Figure 8 combines the synthetic Global Supply Chain Disruption dataset with a matched ISOMORPH edge-cut experiment. In that dataset, the Pearson correlation between monthly disruption-flag share and late-delivery rate ranges from 0.808 to 0.978 for four of the five lane–event pairs and is -0.249 for Tokyo→Singapore. In the ISOMORPH experiment, removing Nashville→Atlanta, which has no alternate route, produces a larger increase in destination unfilled-on-time demand than removing Atlanta→Memphis, for which an alternate route is available.

Ripple and cascading disruption. Table 25 provides the exact timing measurements underlying Figure 4. The Nashville→Atlanta edge (Hub→Tier-2) is removed for 60 days starting at t. For each

downstream node, we report the first day on which incoming edge utilization falls below 30% of its pre-cut level and the first day on which on-hand inventory falls below 50% of its pre-cut level, using the thresholds of the figure. The increasing delays across tiers quantify the progressive propagation of the disruption through the network.

Table 25: **Ripple and cascading disruption following a temporary edge failure.** The edge Nashville→Atlanta is removed for 60 days starting at t. For each downstream tier, the table reports the first day on which incoming edge utilization falls below 30% of its pre-cut level and the first day on which on-hand inventory falls below 50% of its pre-cut level, the thresholds marked in Figure 4; "—" means the threshold is not reached within the 130-day window.

Downstream tier	Incoming edge	First day $U_e < 0.3\,U_{e,0}$	First day $\mathrm{OH} < 0.5\,\mathrm{OH}_0$
Tier-2 (Atlanta)	Nashville→Atlanta	t+5	t+7
Tier-3 (Chicago)	Atlanta→Chicago	t+10	t+46
Tier-4 (Columbus)	Chicago→Columbus	t+28	t+45
Tier-5 (Philadelphia)	Columbus→Philadelphia	—	t+55
Destination (NewYork)	Philadelphia→NewYork	—	t+64

Bottleneck migration. Table 26 provides the exact values underlying Figure 9. The Atlanta→Memphis edge is removed for 60 days. Baltimore has three parallel inbound routes; for each period, we report the fraction of days on which each inbound edge is Baltimore's primary supplier, defined as the inbound edge with the highest daily utilization. The resulting shift in primary-supplier share quantifies the redistribution of flow toward alternative routes during the disruption and the subsequent return toward the pre-cut configuration.

Table 26: **Bottleneck migration under a temporary upstream disruption.** The edge Atlanta→Memphis is removed for 60 days. For each period, the table reports the fraction of days on which each Baltimore inbound edge is the primary supplier, defined as the inbound edge with the highest daily utilization. Pre- and post-cut are the 60 days before the cut and after restoration; "during cut" is days 31–60 of the cut, after Memphis's buffer stock is exhausted.

Inbound edge to Baltimore	Pre-cut	During cut	Post-cut
	Primary supplier share (% of days)		
Memphis→Baltimore	**78**	34	**67**
Columbus→Baltimore	18	**52**	21
Richmond→Baltimore	3	14	12
Total	100	100	100

M FOUNDATION MODEL EVALUATION DETAILS

This appendix collects the implementation details required to reproduce the tables of §5.3: the formal MASE definition (§M.1), checkpoint identifiers and parameter counts (§M.2), per-model inference hyperparameters (Table 28), and the full per-target confidence-interval grids (§M.3).

M.1 MASE: FORMAL DEFINITION

We use the official GIFT-Eval evaluation protocol (`calculate_seasonal_error` and `mase` from the gluonts evaluation module). For each rolling-origin test window w with start time t_w and channel i, the *in-context seasonal error* is

$$\mathrm{SE}_{i,w} \;=\; \tfrac{1}{L-m} \sum_{t=t_w-L}^{t_w-1-m} \big|\, y_{i,\,t+m} - y_{i,\,t} \,\big|,$$

where L is the context length and m=`get_seasonality(freq)` is the GluonTS seasonal period for each native frequency (m=1 daily, 24 hourly, 144 for 10-minute). The model's point forecast

$\hat{y}^{\text{model}}_{i,\,t_w+k}$ is the per-step median over its sample paths. The per-entry MASE at horizon h is the mean MAE over the first h forecast steps divided by the in-context seasonal error:

$$\text{MASE}_{i,w}(h) \;=\; \frac{\frac{1}{h}\sum_{k=0}^{h-1}\left|\, y_{i,\,t_w+k} - \hat{y}^{\text{model}}_{i,\,t_w+k} \,\right|}{\text{SE}_{i,w}}.$$

The MASE score for each dataset is computed by avaraging over windows and channels:

$$\text{MASE}(h) \;=\; \frac{1}{|W|\,C}\sum_{w\in W}\sum_{i=1}^{C}\text{MASE}_{i,w}(h). \tag{35}$$

M.2 Models and inference hyperparameters

Table 27 lists the released checkpoint, parameter count, and pre-train context range for each model, together with the context length L used in our evaluation. Table 28 lists the inference-time hyperparameters: the number of probabilistic sample paths drawn (reduced to the per-step median for the point forecast), the batching unit each wrapper operates on, and per-model configuration knobs.

Table 27: Foundation models evaluated. Pre-train context is the sequence length used during published pre-training; L used is the context length in our zero-shot evaluation (see §5.1).

Model	HuggingFace ID	Params	Pre-train ctx	L used
Chronos	`amazon/chronos-t5-base`	200M	512	512
Moirai	`Salesforce/moirai-1.1-R-base`	91M	512–2048	512
TimesFM	`google/timesfm-2.0-500m-pytorch`	500M	up to 2048	512

Table 28: Inference hyperparameters. *Samples* is the number of probabilistic sample paths drawn; the per-day median is the point forecast used to compute MASE. TimesFM uses a deterministic quantile head and reports no sample count. *Batch* indicates the unit batched: Chronos batches *channels* within a window, Moirai is multivariate-native and batches *windows*, TimesFM batches *channels* per call.

Model	Samples	Batch	Model-specific
Chronos	20	channel_batch=16	bf16 inference
Moirai	100	8 (C=50) / 1 (C=200)	patch_size=32; OOM-bound at C=200
TimesFM	—	per_core_batch=32	horizon_len=128 (decode 1 patch, slice to H)

M.3 Network-state forecast targets: full grids

Tables 29–31 give the complete 7-model × 4-horizon × 3-seasonal-lag grids for backlog, fill rate, and edge utilization, on the same 29-rollout evaluation cohort and under the same rolling-window protocol as Table 5 (demand). The row labeled *SN(m) MAE* is the seasonal-naive denominator in the units of the target, and is included so that a MASE value can be converted back to an absolute error: MASE alone is not comparable across targets whose scales differ by four orders of magnitude (fill rate is $O(10^{-3})$, backlog is $O(10^{2})$).

Two entries deserve explicit comment rather than being left for the reader to find. ARIMA's backlog and fill-rate columns at $h \geq 7$ are reported in scientific notation because the fitted models diverge, not because of a formatting choice; the values (10^2 to 10^{17}) are the actual computed MASE and are retained because suppressing them would hide the diagnostic. And the backlog and fill-rate tables use an effective n=24 rather than 29: five rollouts in the cohort are under-loaded to the point that backlog is identically zero and fill rate identically one, which makes MASE undefined (zero denominator), so those rollouts are dropped from those two targets and retained for demand and edge utilization. They are not dropped for poor performance.

N Computational resources

Time-series generation. Simulator runs use a single CPU core and no GPU. A C=50 rollout takes roughly 20 minutes; C=200 roughly 90 minutes. The release comprises 34 generated rollouts: the

Table 29: Univariate MASE on **backlog**, mean ± 95% Student-t CI over the n=29-rollout cohort. 5 rollouts are excluded from this target's CI because MASE is undefined when the series is constant (under-loaded scenarios yield zero backlog / unit fill rate), giving effective n=24. *In-domain trained* baselines (ARIMA, ETS, PatchTST) are fit per-rollout; foundation models (Chronos, Moirai, TimesFM) run *zero-shot*. Agg. is the geometric mean across horizons.

Model	h=1	h=7	h=14	h=30	Agg.
MASE (m=1)					
ARIMA	3.388 ± 1.780	5.1e+02	8.4e+06	4.9e+17	9.2e+06
ETS	2.531 ± 0.987	4.873 ± 2.280	7.309 ± 3.375	12.848 ± 6.732	5.834 ± 2.684
PatchTST	5.622 ± 1.630	7.209 ± 2.430	8.443 ± 2.888	10.218 ± 4.168	7.690 ± 2.642
Chronos	2.109 ± 1.164	3.876 ± 2.343	4.860 ± 2.616	6.361 ± 3.946	3.987 ± 2.306
Moirai	1.570 ± 0.910	3.431 ± 2.174	4.566 ± 2.467	6.427 ± 3.988	3.546 ± 2.103
TimesFM	5.157 ± 2.579	6.236 ± 3.229	8.038 ± 4.165	11.914 ± 6.788	7.449 ± 3.920
SN(m=1) MAE	59.59	58.79	59.13	61.66	59.79
MASE (m=7)					
ARIMA	2.760 ± 1.591	4.2e+02	7.1e+06	4.2e+17	7.7e+06
ETS	1.853 ± 0.807	3.459 ± 1.891	4.985 ± 2.638	8.532 ± 5.263	4.063 ± 2.156
PatchTST	3.455 ± 1.003	4.634 ± 1.719	5.321 ± 1.986	6.435 ± 3.309	4.839 ± 1.862
Chronos	1.047 ± 0.616	2.333 ± 1.795	2.901 ± 1.922	3.890 ± 3.315	2.291 ± 1.639
Moirai	0.958 ± 0.636	2.268 ± 1.806	2.882 ± 1.930	3.935 ± 3.327	2.228 ± 1.653
TimesFM	4.081 ± 1.987	4.740 ± 2.444	5.859 ± 3.030	8.342 ± 5.105	5.545 ± 2.951
SN(m=7) MAE	270.1	265.7	261.8	271.6	267.3
MASE (m=30)					
ARIMA	2.664 ± 1.521	4e+02	6.8e+06	4e+17	7.3e+06
ETS	1.801 ± 0.776	3.308 ± 1.803	4.744 ± 2.570	7.964 ± 5.045	3.873 ± 2.076
PatchTST	3.372 ± 1.044	4.487 ± 1.660	5.154 ± 1.955	6.202 ± 3.179	4.690 ± 1.833
Chronos	0.897 ± 0.573	2.134 ± 1.714	2.686 ± 1.852	3.553 ± 3.181	2.067 ± 1.559
Moirai	0.862 ± 0.600	2.114 ± 1.728	2.698 ± 1.863	3.593 ± 3.191	2.050 ± 1.581
TimesFM	4.422 ± 1.974	4.924 ± 2.347	5.993 ± 2.940	8.343 ± 4.895	5.744 ± 2.868
SN(m=30) MAE	598.2	576.7	559.3	575.5	577.4

C=50 and C=200 baselines, 24 one-knob scenario settings at C=50 (the six sweeps contribute four non-baseline settings each, and the per-sweep baseline is shared with the main C=50 release rather than re-simulated), the two compound scenarios `chaos_compound` and `chaos_burst`, the two dynamic edge-cut scenarios, and the four extension demonstrations of App. H. The forward-UQ ensemble of §5.2 reuses 20 of these rollouts and generates none of its own. Wall-clock on the order of 12 CPU hours.

Foundation-model inference. All zero-shot inference ran on a single NVIDIA RTX 2080 Ti, one job per GPU. The pipeline covers the TSF evaluation of §5 (6 forecasters × 4 targets × 29 rollouts) and the forward-UQ scoring of §5.2 over 3 foundation models × 20 of those rollouts. Aggregate wall-clock is on the order of 30 GPU hours: roughly 22 hours for the TSF evaluation (each job runs the full $|W|$=36 rolling-origin protocol of §5.1 at stride 30) and roughly 7 hours for the UQ ensemble.

Table 30: Univariate MASE on **fill_rate**, mean ± 95% Student-t CI over the n=29-rollout cohort. 5 rollouts are excluded from this target's CI because MASE is undefined when the series is constant (under-loaded scenarios yield zero backlog / unit fill rate), giving effective n=24. *In-domain trained* baselines (ARIMA, ETS, PatchTST) are fit per-rollout; foundation models (Chronos, Moirai, TimesFM) run *zero-shot*. Agg. is the geometric mean across horizons.

Model	h=1	h=7	h=14	h=30	Agg.
MASE (m=1)					
ARIMA	3.763 ± 1.766	8.1e+02	9e+06	4e+16	5.7e+06
ETS	3.923 ± 1.527	7.747 ± 3.368	11.946 ± 5.754	20.657 ± 10.791	9.306 ± 4.245
PatchTST	10.648 ± 3.439	12.503 ± 3.687	13.814 ± 3.971	14.518 ± 3.783	12.783 ± 3.723
Chronos	2.902 ± 0.951	3.837 ± 1.204	4.592 ± 1.457	5.578 ± 2.166	4.110 ± 1.383
Moirai	2.308 ± 0.993	2.939 ± 1.249	3.391 ± 1.284	4.185 ± 1.853	3.132 ± 1.312
TimesFM	4.902 ± 2.266	5.979 ± 2.877	7.192 ± 3.878	9.066 ± 5.558	6.612 ± 3.457
SN(m=1) MAE	0.0027	0.0027	0.0031	0.0033	0.0029
MASE (m=7)					
ARIMA	3.015 ± 1.591	6.9e+02	7.6e+06	3.3e+16	4.8e+06
ETS	2.946 ± 1.446	5.634 ± 3.000	8.799 ± 5.074	15.801 ± 9.512	6.931 ± 3.811
PatchTST	6.554 ± 2.788	7.663 ± 3.162	8.300 ± 3.322	8.495 ± 2.831	7.714 ± 3.027
Chronos	2.107 ± 0.792	2.470 ± 0.868	2.798 ± 0.929	3.349 ± 1.330	2.643 ± 0.962
Moirai	1.007 ± 0.463	1.393 ± 0.754	1.605 ± 0.683	2.091 ± 1.117	1.473 ± 0.721
TimesFM	3.875 ± 2.181	4.499 ± 2.731	5.314 ± 3.634	6.857 ± 5.261	5.021 ± 3.281
SN(m=7) MAE	0.0066	0.0073	0.0081	0.0079	0.0075
MASE (m=30)					
ARIMA	2.894 ± 1.513	6.6e+02	7.3e+06	3.2e+16	4.6e+06
ETS	2.833 ± 1.381	5.380 ± 2.850	8.377 ± 4.838	15.146 ± 9.049	6.631 ± 3.630
PatchTST	6.488 ± 2.987	7.433 ± 3.279	8.003 ± 3.424	8.338 ± 2.912	7.532 ± 3.156
Chronos	2.047 ± 0.764	2.391 ± 0.829	2.691 ± 0.892	3.317 ± 1.331	2.571 ± 0.933
Moirai	0.905 ± 0.435	1.278 ± 0.710	1.469 ± 0.643	2.034 ± 1.158	1.363 ± 0.695
TimesFM	3.775 ± 2.109	4.333 ± 2.613	5.092 ± 3.474	6.663 ± 5.023	4.854 ± 3.144
SN(m=30) MAE	0.0076	0.0083	0.0095	0.0092	0.0087

Table 31: Univariate MASE on **edge_util**, mean $\pm$ 95% Student-t CI over the n=29-rollout cohort. *In-domain trained* baselines (ARIMA, ETS, PatchTST) are fit per-rollout; foundation models (Chronos, Moirai, TimesFM) run *zero-shot*. Agg. is the geometric mean across horizons.

Model	h=1	h=7	h=14	h=30	Agg.
MASE (m=1)					
ARIMA	0.791 ± 0.031	0.815 ± 0.029	0.820 ± 0.028	0.834 ± 0.031	0.815 ± 0.030
ETS	0.798 ± 0.022	0.820 ± 0.028	0.827 ± 0.027	0.843 ± 0.031	0.822 ± 0.027
PatchTST	0.898 ± 0.036	0.910 ± 0.046	0.909 ± 0.045	0.921 ± 0.046	0.910 ± 0.043
Chronos	0.736 ± 0.026	0.757 ± 0.016	0.758 ± 0.018	0.773 ± 0.019	0.756 ± 0.020
Moirai	0.721 ± 0.024	0.747 ± 0.015	0.747 ± 0.016	0.758 ± 0.018	0.743 ± 0.018
TimesFM	0.812 ± 0.072	0.831 ± 0.108	0.857 ± 0.151	0.844 ± 0.114	0.836 ± 0.111
SN(m=1) MAE	0.1145	0.1140	0.1136	0.1129	0.1137
MASE (m=7)					
ARIMA	0.779 ± 0.040	0.798 ± 0.031	0.801 ± 0.029	0.814 ± 0.030	0.798 ± 0.033
ETS	0.777 ± 0.016	0.797 ± 0.024	0.802 ± 0.021	0.816 ± 0.023	0.798 ± 0.021
PatchTST	0.874 ± 0.030	0.886 ± 0.041	0.886 ± 0.039	0.897 ± 0.041	0.886 ± 0.038
Chronos	0.718 ± 0.024	0.736 ± 0.013	0.737 ± 0.013	0.750 ± 0.013	0.735 ± 0.016
Moirai	0.702 ± 0.023	0.724 ± 0.012	0.724 ± 0.012	0.734 ± 0.011	0.721 ± 0.014
TimesFM	0.794 ± 0.071	0.812 ± 0.107	0.837 ± 0.149	0.823 ± 0.113	0.817 ± 0.109
SN(m=7) MAE	0.1176	0.1173	0.1168	0.1161	0.1170
MASE (m=30)					
ARIMA	0.768 ± 0.038	0.786 ± 0.030	0.788 ± 0.026	0.799 ± 0.028	0.785 ± 0.031
ETS	0.767 ± 0.015	0.785 ± 0.023	0.789 ± 0.018	0.802 ± 0.021	0.786 ± 0.019
PatchTST	0.861 ± 0.029	0.873 ± 0.040	0.872 ± 0.038	0.883 ± 0.039	0.872 ± 0.036
Chronos	0.708 ± 0.023	0.725 ± 0.012	0.725 ± 0.011	0.738 ± 0.010	0.724 ± 0.014
Moirai	0.692 ± 0.022	0.712 ± 0.010	0.712 ± 0.009	0.721 ± 0.008	0.709 ± 0.012
TimesFM	0.784 ± 0.070	0.804 ± 0.112	0.829 ± 0.157	0.814 ± 0.118	0.808 ± 0.113
SN(m=30) MAE	0.1193	0.1192	0.1187	0.1181	0.1188