Inverse Delayed Reinforcement Learning for Limit Order Books: A Multi-Strategy EM–MaxEnt Approach for Inferring Latency-Driven Trading Behaviour

Shivam Hedau

Msc, University of Edinburgh

S.Hedau@sms.ed.ac.uk

December 7, 2025

Abstract

Many real-world sequential systems—including distributed sensor networks, autonomous multi-agent platforms, and high-frequency financial markets—operate under *asynchrony*: observations are delayed, actions are executed after further latency, and agents rarely share a common temporal view of the underlying state. Learning behaviour from such data requires recovering both *latent delays* and *latent behavioural mechanisms*.

We introduce **Inverse Delayed Reinforcement Learning (IDRL)**, a general probabilistic framework for inferring reward-driven behaviour when actions are generated from temporally misaligned or stale states. Our approach builds on a **Time-Bracketed EM–MaxEnt IRL** formulation that jointly estimates (i) probabilistic state–action alignment across delay windows, (ii) latent strategy mixtures via a mixture-of-experts model, and (iii) reward parameters for each strategy. We establish two theoretical results: **Theorem A** proves concavity, monotonicity, and convergence for the single-strategy EM–MaxEnt estimator, while **Theorem B** extends these guarantees to the full multi-strategy model equipped with softmax gating over history-dependent features.

To illustrate the framework on a challenging real-world system, we apply IDRL to high-frequency **limit order book (LOB)** data. Using a 19-dimensional microstructure feature set and a discretised delay window, IDRL recovers heterogeneous latent strategies with systematically different reaction times, distinct reward geometries, and state-dependent switching behaviour. The learned delay distributions correlate strongly with measurable state variables, revealing interpretable temporal structure in asynchronous market dynamics.

Overall, IDRL provides a statistically grounded and computationally tractable approach for reverse-engineering behaviour in asynchronous multi-agent systems, with high-frequency trading serving as a representative application domain where latent delays are especially consequential.

1 Introduction

Many real-world sequential decision-making systems operate under *asynchrony*: agents observe the environment with delays, act after additional latency, and rarely share a unified temporal view of the underlying state. Such settings arise in distributed sensor networks, autonomous multi-robot systems, communication networks, and high-frequency electronic markets. In these environments, actions are generated from *temporally misaligned or stale* observations, so the data naturally contain

unobserved delays, unobserved behavioural modes, and unobserved temporal couplings that shape system dynamics.

A representative example comes from high-frequency *limit order books* (LOBs), where microsecond-level delays stem from network routing, feed processing, queueing effects, and exchange-level matching engines. Market participants differ widely in latency: colocated trading systems receive and react to information nearly instantaneously, whereas remote or retail participants operate on substantially lagged states. These heterogeneous delays critically influence execution quality, order placement, and strategic interaction. Yet the delays—and the behavioural mechanisms that interact with them—are never observed directly.

This motivates a broad scientific question that extends well beyond financial markets:

> *Given an observed action, which past state most plausibly generated it, and what behavioural mechanism best explains that choice under temporal misalignment?*

Answering this question requires a probabilistic framework capable of jointly inferring latent delays and latent behavioural structure from asynchronous sequential data.

We introduce **Inverse Delayed Reinforcement Learning (IDRL)**, a general framework for learning reward-driven behaviour when actions are generated from delayed or stale observations. The key idea is to generalise Maximum Entropy Inverse Reinforcement Learning (MaxEnt IRL) to settings with latent temporal alignment. Instead of assuming that each action corresponds to the most recent state, we introduce a *time-bracketed* model in which every action is softly matched to a window of candidate past states. The **Expectation–Maximisation (EM)** algorithm then assigns probabilistic responsibilities over delay candidates and iteratively updates reward parameters. This yields a tractable latent-variable formulation for behavioural inference in asynchronous environments.

Building on this foundation, we extend the model to incorporate heterogeneous behavioural modes through a *mixture-of-experts* architecture. A softmax gating network, driven by history-dependent features, determines which latent strategy is active at each time. The resulting framework jointly learns:

1. probabilistic state–action–delay alignments over a discretised delay grid,
2. latent behavioural strategies and their regime-dependent activation patterns,
3. reward functions governing each strategy.

Theoretical guarantees: We provide rigorous statistical guarantees for both the single- and multi-strategy settings. **Theorem A** establishes concavity, monotonicity, and EM convergence for the time-bracketed MaxEnt IRL estimator. **Theorem B** extends these properties to the mixture-of-experts model, proving blockwise concavity and convergence of the joint reward–gating parameter updates. These results contribute to the broader theory of latent-variable inference under temporal misalignment.

Application to high-frequency market data: To illustrate the framework on a challenging real-world domain, we apply IDRL to high-frequency LOB data. Using a 19-dimensional microstructure representation and a multi-lag delay grid, IDRL uncovers heterogeneity in reaction times, strategy-specific reward geometries, and state-dependent switching dynamics. Although LOBs serve as a motivating case study, the methodology applies broadly to any asynchronous multi-agent system where actions may originate from stale or delayed observations.

Contributions. This paper makes the following contributions:

- We introduce **IDRL**, a probabilistic latent-variable framework for inverse reinforcement learning with unobserved delays.
- We develop a **time-bracketed EM–MaxEnt formulation** that jointly infers delay responsibilities, latent strategies, and reward parameters.
- We prove **concavity, monotonicity, and convergence guarantees** for both single-strategy and multi-strategy variants.
- We demonstrate the framework on high-frequency LOB data, revealing interpretable temporal and behavioural structure in an asynchronous real-world environment.

Overall, IDRL provides a statistically grounded and computationally tractable method for reverse-engineering behaviour in asynchronous multi-agent systems. The remainder of the paper develops the full mathematical formulation, inference procedure, empirical analysis, and interpretation of results.

2 Literature Review

The proposed Inverse Delayed Reinforcement Learning (IDRL) framework builds on several areas of research in machine learning, statistics, and asynchronous multi-agent systems, with limit order books (LOBs) serving as a representative application domain. We review the most relevant lines of work and highlight the methodological gaps that motivate a temporally misaligned, latent-variable formulation.

2.1 Learning Under Temporal Misalignment and Asynchrony

Sequential decision-making systems frequently exhibit *asynchrony*: observations may be delayed, actions may be executed after further latency, and agents may operate on stale state information. Such settings arise in distributed robotics, multi-sensor fusion, communication networks, and real-world control systems. Classical state–action models assume synchronous alignment, but temporal misalignment breaks the Markov property and invalidates many standard inference techniques.

Several recent ML works address related challenges through temporal alignment models, delay-aware RL formulations, or probabilistic compensators, including delayed MDPs, off-policy learning with lagged observations, and alignment in sequential models. However, these approaches typically *presume* known delay distributions or treat delays as nuisance variables rather than latent quantities that interact with behavioural structure. There is currently no framework that jointly infers *latent delays* and *latent strategies* in a probabilistic, reward-driven setting. This motivates the development of delay-indexed latent-variable models such as IDRL.

2.2 Inverse Reinforcement Learning and Maximum-Entropy Approaches

Inverse Reinforcement Learning (IRL) seeks to recover the reward function that explains observed behaviour [21]. Maximum Entropy IRL (MaxEnt IRL) [23] provides a tractable probabilistic formulation in which expert behaviour is modeled by log-linear policies, allowing convex optimisation over feature expectations [1]. Numerous extensions have since appeared, including causal IRL, deep IRL, adversarial IRL, and multi-agent IRL.

Despite this progress, nearly all IRL formulations assume that each action is conditioned on the *current* state. They therefore implicitly require a synchronous data-generating process. In domains with delayed observations or delayed actions, the relevant conditioning state is latent, and classical IRL cannot be applied without discarding the temporal structure. No prior IRL framework accounts for *unknown, heterogeneous, strategy-dependent delays* or performs inference over temporal alignment between states and actions. Our time-bracketed MaxEnt formulation fills this gap by treating delays as latent variables and estimating delay-conditioned feature expectations via EM-style responsibility updates [9, 11].

2.3 Latent-Variable Models and Mixture-of-Experts Architectures

Latent-variable models such as mixture models, hierarchical mixtures of experts, and soft-gated clustering frameworks [13, 14] provide structured ways to represent heterogeneous behaviour. These models have appeared in robotics, behavioural learning, dynamical clustering, and multi-policy RL. They enable multi-strategy decomposition, smooth responsibility assignment, and interpretable gating mechanisms.

However, existing latent-variable RL or IRL models do not incorporate *latent temporal alignment*. Moreover, traditional strategy clustering approaches—such as k-means, hierarchical clustering, or Gaussian mixtures applied to state–action pairs—do not base clustering on reward-driven

principles and cannot disentangle behavioural heterogeneity from delay-induced temporal distortion. IDRL extends mixture-of-experts architectures by integrating latent strategies with delay-conditioned alignment, allowing heterogeneity to be inferred jointly with temporal structure.

2.4 Asynchrony and Latency in High-Frequency Limit Order Books

Limit order books constitute a natural and challenging real-world example of asynchronous multi-agent interaction. Classical LOB models describe order arrivals via point processes, Markovian dynamics, queue-reactive mechanisms, or Hawkes-style intensities [17, 19, 7, 3, 2]. Modern empirical microstructure studies emphasize latency asymmetries, stale-quote effects, and speed competition [5, 4, 10, 18, 6].

While these models capture liquidity dynamics, they typically assume synchronous observations or impose simple parametric delay structures. None of them attempt to infer latent delays, latent behavioural strategies, and reward parameters jointly. In this paper, LOBs serve as an illustrative domain where asynchrony is extreme and delays are economically consequential, motivating the need for delay-aware behavioural inference.

2.5 Gaps in Existing Work

Across reinforcement learning, probabilistic modelling, and market microstructure, several important gaps remain:

- **IRL methods do not model unknown, heterogeneous delays.** Classical IRL assumes state–action synchrony [21, 23].
- **Temporal misalignment is not treated as a latent variable.** Existing delay-aware RL approaches assume known or exogenous delays.
- **No unified model jointly infers delays, strategies, and rewards.** Microstructure, IRL, and clustering models treat these problems separately.
- **Mixture-of-experts IRL has not been developed for asynchronous systems.** Existing mixtures do not simultaneously incorporate latent strategies and latent delays.

These limitations motivate **IDRL**, a general latent-variable framework that links time-bracketed state–action alignment, reward-based strategy inference, and statistical learning under asynchrony.

3 Mathematical Framework

We formalise the asynchronous decision process underlying Inverse Delayed Reinforcement Learning (IDRL). The goal is to model systems in which observed actions are generated from *delayed or stale* states, and the delays, behavioural modes, and reward parameters are latent. Although limit order books (LOBs) provide a concrete application, the formulation is general and applies to any sequential multi-agent environment with temporal misalignment.

3.1 Asynchronous State–Action Processes

Let $\{s_t\}_{t=1}^T$, $s_t \in \mathbb{R}^d$, denote the underlying state trajectory of an asynchronous environment. At each time index t, an agent executes an observed action $a_t \in \mathcal{A}$, but the internal decision leading to a_t is based on a *past* state. Specifically, there exists an unobserved delay variable

$$\Delta_t \in \{\Delta_1, \dots, \Delta_K\}$$

such that the relevant conditioning state is $s_{t-\Delta_t}$.

Thus the observable data $(s_{1:T}, a_{1:T})$ are generated from the latent structure

$$a_t \sim p_\theta(a \mid s_{t-\Delta_t}),$$

where both the delay Δ_t and the behavioural parameters θ are unknown. This formulation captures asynchronous robotic platforms, distributed sensor networks, communication systems with buffering delays, and high-frequency LOBs, where reaction times vary across agents and conditions.

In the LOB case, s_t corresponds to a feature vector derived from order-book updates, while delays arise from network latency, computational pipelines, and exchange-level queueing. However, the mathematical structure remains identical across domains.

3.2 State Representation

We assume a fixed feature map $\phi : \mathcal{S} \to \mathbb{R}^d$. In general multi-agent systems, components of $\phi(s_t)$ may encode information such as local observations, aggregated sensor measurements, predictive signals, or resource constraints.

In high-frequency LOB applications, $\phi(s_t)$ includes microstructure-relevant variables (spread, imbalance, liquidity measures, flow indicators), but these interpretive details are domain-specific and not required for the general IDRL formulation.

Throughout the analysis, we treat $\{s_t\}$ as an exogenous, observed sequence. IDRL does not require modelling the state transition dynamics.

3.3 Action Space

Let $\mathcal{A}$ denote a finite action set. Across application domains, actions may correspond to:

- movement commands or control signals (robotics),
- routing or scheduling decisions (communication systems),
- order placement, cancellation, or execution types (financial markets),
- mode switches or coordination actions (multi-agent systems).

The key assumption is that actions are conditionally distributed according to a reward-driven mechanism evaluated on a potentially stale state $s_{t-\Delta_t}$.

3.4 Latent Delay Model

Let $\Delta = \{\Delta_1, \ldots, \Delta_K\}$ be a discrete grid of possible delays. For each action a_t, the agent internally observes one of the delayed states

$$\mathcal{W}_t = \{s_{t-\Delta_1}, \ldots, s_{t-\Delta_K}\},$$

but the identity of the correct state is latent.

We model

$$\mathbb{P}(\Delta_t = \Delta_k) = \pi_{\Delta_k},$$

where π_{Δ_k} may be uniform or learnable. This latent-variable structure is central to IDRL: temporal alignment is not fixed but inferred.

In LOBs, delays may arise from gateway jitter, feed latency, queueing, or hardware variation; in other asynchronous systems, they may represent communication delays, computation time, or scheduling latency.

3.5 Reward Function and Joint Feature Map

Every behavioural strategy is assumed to optimise a linear reward model:

$$R_\theta(s, a) = \theta^\top f(s, a),$$

where $f(s, a)$ is a joint feature representation. Linear rewards are widely used in IRL, offering interpretability, convexity properties, and compatibility with MaxEnt formulations.

In general domains, $f(s, a)$ captures state–action dependencies or performance trade-offs. In LOB applications, it may encode liquidity, imbalance, spread costs, or order-flow signals, but these constitute only one instantiation of the generic feature map.

3.6 Maximum Entropy Action Likelihood

Conditioned on a candidate state $s_{t-\Delta}$, behaviour follows the MaxEnt IRL distribution:

$$p(a_t \mid s_{t-\Delta}, \theta) = \frac{\exp(\theta^\top f(s_{t-\Delta}, a_t))}{Z(s_{t-\Delta}, \theta)},$$

where

$$Z(s_{t-\Delta}, \theta) = \sum_{a \in \mathcal{A}} \exp(\theta^\top f(s_{t-\Delta}, a)).$$

This captures stochastic choice behaviour under reward-driven preferences and is suitable for noisy, heterogeneous agents.

3.7 Marginal Likelihood Under Unknown Delays

Since Δ_t is unobserved, the likelihood of action a_t is a mixture:

$$p(a_t \mid \theta) = \sum_{k=1}^{K} \pi_{\Delta_k}\, p(a_t \mid s_{t-\Delta_k}, \theta).$$

For the entire trajectory:

$$\mathcal{L}(\theta) = \sum_{t=1}^{T} \log \left(\sum_{k=1}^{K} \pi_{\Delta_k} \frac{\exp\big(\theta^\top f(s_{t-\Delta_k}, a_t)\big)}{Z(s_{t-\Delta_k}, \theta)} \right).$$

This log-likelihood naturally induces latent responsibilities:

$$\gamma_{t,k} = \mathbb{P}(\Delta_t = \Delta_k \mid a_t, s_{1:T}, \theta),$$

which form the basis of the EM algorithm studied later.

The formulation matches a general latent-variable model with discrete hidden states (delays) and log-linear emission distributions (MaxEnt IRL). It is indifferent to domain and hence applies equally to HFT markets, multi-robot systems, and asynchronous sensor-driven environments.

4 EM–MaxEnt IRL: Expectation–Maximisation for Delayed Inverse Reinforcement Learning

Given the latent-delay structure introduced in the previous section, estimation of the reward parameters θ requires marginalising over unknown temporal alignments between states and actions. This naturally yields a latent-variable likelihood, for which the Expectation–Maximisation (EM) algorithm provides a principled solution. In this section, we derive the EM–MaxEnt procedure that underpins IDRL and forms the basis for the theoretical guarantees in Theorem A and Theorem B.

4.1 Latent Delay Variables and the Complete-Data Likelihood

For each observed action a_t, let z_{t,Δ_k} be a binary indicator denoting whether delay Δ_k is responsible for the state that generated the action. Formally,

$$z_{t,\Delta_k} = \begin{cases} 1, & \text{if the decision leading to } a_t \text{ was made using } s_{t-\Delta_k}, \\ 0, & \text{otherwise,} \end{cases} \qquad \sum_{k=1}^{K} z_{t,\Delta_k} = 1.$$

Under the MaxEnt IRL model, the complete-data likelihood factorises as

$$p(a_t, z_{t,\Delta_k} \mid \theta) = \left[\pi_{\Delta_k} \frac{\exp(\theta^\top f_{t,\Delta_k})}{Z_{t,\Delta_k}(\theta)} \right]^{z_{t,\Delta_k}},$$

where π_{Δ_k} is the delay prior and $f_{t,\Delta_k} = f(s_{t-\Delta_k}, a_t)$.

Summing over all observations,

$$\log p(a_{1:T}, z_{1:T} \mid \theta) = \sum_{t=1}^{T} \sum_{k=1}^{K} z_{t,\Delta_k} \left[\log \pi_{\Delta_k} + \theta^\top f_{t,\Delta_k} - \log Z_{t,\Delta_k}(\theta) \right].$$

The latent variables $\{z_{t,\Delta_k}\}$ reflect uncertainty in the temporal alignment between states and actions, a characteristic feature of asynchronous systems across domains.

4.2 E-step: Posterior Delay Responsibilities

The E-step computes the posterior responsibility that delay Δ_k produced action a_t under parameters $\theta^{(n)}$:

$$\gamma_{t,\Delta_k}^{(n)} = \mathbb{E}[z_{t,\Delta_k} \mid a_t, s_{1:T}, \theta^{(n)}].$$

By Bayes' rule,

$$\gamma_{t,\Delta_k}^{(n)} = \frac{\pi_{\Delta_k} \exp(\theta^{(n)\top} f_{t,\Delta_k}) / Z_{t,\Delta_k}(\theta^{(n)})}{\sum_{j=1}^{K} \pi_{\Delta_j} \exp(\theta^{(n)\top} f_{t,\Delta_j}) / Z_{t,\Delta_j}(\theta^{(n)})}.$$

These responsibilities act as soft temporal alignments, assigning each action a distribution over candidate historical states. The E-step is therefore an instance of probabilistic alignment over a discrete delay grid.

Domain remark: In LOB applications, small inferred delays correspond to rapid reaction strategies, while larger delays reflect slower, possibly inventory-driven or passive behaviour. This interpretation, however, is specific to the financial domain and not required for the general EM formulation.

4.3 Expected Sufficient Statistics

The E-step produces delay-weighted empirical feature counts,

$$\bar{f}(\theta^{(n)}) = \sum_{t=1}^{T} \sum_{k=1}^{K} \gamma_{t,\Delta_k}^{(n)} f_{t,\Delta_k},$$

and model-expected feature counts,

$$\hat{f}(\theta^{(n)}) = \sum_{t=1}^{T} \sum_{k=1}^{K} \gamma_{t,\Delta_k}^{(n)} \sum_{a' \in \mathcal{A}} p(a' \mid s_{t-\Delta_k}, \theta^{(n)})\, f(s_{t-\Delta_k}, a').$$

These generalise the classical MaxEnt IRL moment-matching structure to temporally misaligned, latent-delay environments.

4.4 M-step: Maximising the Expected Complete-Data Log-Likelihood

The M-step updates θ by maximising

$$\mathcal{Q}(\theta \mid \theta^{(n)}) = \sum_{t=1}^{T} \sum_{k=1}^{K} \gamma_{t,\Delta_k}^{(n)} \left[\theta^\top f_{t,\Delta_k} - \log Z_{t,\Delta_k}(\theta) \right].$$

Differentiating yields

$$\nabla_\theta \mathcal{Q}(\theta \mid \theta^{(n)}) = \bar{f}(\theta^{(n)}) - \hat{f}(\theta),$$

so the M-step update solves the convex optimisation problem

$$\theta^{(n+1)} = \arg\max_{\theta} \mathcal{Q}(\theta \mid \theta^{(n)}).$$

Gradient ascent with step size η gives

$$\theta^{(n+1)} \leftarrow \theta^{(n)} + \eta \left(\bar{f}(\theta^{(n)}) - \hat{f}(\theta^{(n)}) \right).$$

Domain remark: In LOB applications, θ encodes incentives such as immediacy, adverse selection aversion, or queue positioning. However, the optimisation and convergence properties are domain-independent.

4.5 Monotonic Improvement and EM Guarantees

Because $\mathcal{Q}$ is concave in θ and each M-step maximises $\mathcal{Q}(\cdot \mid \theta^{(n)})$, classical EM theory implies monotonic improvement:

$$\mathcal{L}(\theta^{(n+1)}) \geq \mathcal{L}(\theta^{(n)}).$$

This property is established formally in Theorem A. The latent-delay structure introduces no degeneracies in the EM update, making the convergence guarantees analogous to those of mixture models and other latent-variable exponential-family estimators.

4.6 Interpretation in Asynchronous Decision Processes

The EM–MaxEnt IRL procedure can be interpreted generically as:

- **E-step:** soft reconstruction of the latent temporal alignment between states and actions;
- **M-step:** reward estimation that renders the reconstructed behaviour statistically optimal under the MaxEnt model.

Across domains—robotics, communication networks, multi-agent decision-making, and LOBs—this combination enables IDRL to infer both *when* an agent is reacting and *why* its behaviour appears consistent with a particular reward-driven strategy.

5 Theoretical Guarantees for Single-Strategy IDRL

We now establish the core statistical properties of the time-bracketed EM–MaxEnt IRL model in the single-strategy setting. The unknown delay variables introduce a latent-variable structure analogous to mixture models and exponential-family estimators. We show that the M-step objective is strictly concave, that the marginal log-likelihood increases monotonically under EM updates, and that the parameter sequence converges to a stationary point. These results extend classical EM theory [22] to asynchronous inverse reinforcement learning with latent temporal alignment.

5.1 Preliminaries

Recall the expected complete-data log-likelihood:

$$\mathcal{Q}(\theta \mid \theta^{(n)}) = \sum_{t=1}^{T} \sum_{k=1}^{K} \gamma_{t,\Delta_k}^{(n)} \left[\theta^\top f_{t,\Delta_k} - \log Z_{t,\Delta_k}(\theta) \right],$$

and the marginal log-likelihood:

$$\mathcal{L}(\theta) = \sum_{t=1}^{T} \log \left(\sum_{k=1}^{K} \pi_{\Delta_k} \frac{\exp(\theta^\top f_{t,\Delta_k})}{Z_{t,\Delta_k}(\theta)} \right).$$

The responsibilities $\gamma_{t,\Delta_k}^{(n)}$ satisfy:

$$\gamma_{t,\Delta_k}^{(n)} \geq 0, \qquad \sum_{k=1}^{K} \gamma_{t,\Delta_k}^{(n)} = 1.$$

We proceed by establishing convexity/concavity properties that underpin EM convergence.

5.2 Lemma 1: Convexity of the Log-Partition Function

Lemma 1. *For any fixed state s and feature map $f(s,a)$, the log-partition function*

$$\log Z(s,\theta) = \log \left(\sum_{a' \in \mathcal{A}} \exp(\theta^\top f(s,a')) \right)$$

is convex in θ.

Proof. MaxEnt IRL defines an exponential-family model with sufficient statistics $f(s,a')$. The log-partition function is the cumulant-generating function of this family, whose Hessian equals the covariance of the sufficient statistics:

$$\nabla_\theta^2 \log Z(s,\theta) = \mathrm{Cov}_{p(a'|s,\theta)}[f(s,a')] \succeq 0.$$

Thus $\log Z$ is convex in θ. □

5.3 Lemma 2: Strict Concavity of $\mathcal{Q}$

Lemma 2. *For fixed responsibilities $\gamma_{t,\Delta_k}^{(n)}$, the function $\mathcal{Q}(\theta \mid \theta^{(n)})$ is strictly concave in θ.*

Proof. Writing

$$\mathcal{Q}(\theta) = \sum_{t,k} \gamma_{t,\Delta_k} \left[\theta^\top f_{t,\Delta_k} - \log Z_{t,\Delta_k}(\theta) \right],$$

we observe:

- $\theta^\top f_{t,\Delta_k}$ is linear in θ;
- $-\log Z_{t,\Delta_k}(\theta)$ is concave by Lemma 1.

Weighted sums of concave functions with nonnegative weights remain concave. Strict concavity holds whenever the induced policy assigns nonzero probability to at least two actions, implying the Hessian $\nabla_\theta^2 \mathcal{Q}$ is strictly negative definite. This condition holds generically in MaxEnt IRL. □

5.4 Lemma 3: EM Lower-Bound Property

Lemma 3. *For any θ and $\theta^{(n)}$,*

$$\mathcal{Q}(\theta \mid \theta^{(n)}) \leq \mathcal{L}(\theta),$$

with equality at $\theta = \theta^{(n)}$ when responsibilities correspond to the exact posterior.

Proof. Let $q(z)$ be any distribution over latent delay assignments $z_{1:T}$. Then:

$$\mathcal{L}(\theta) = \log p(a_{1:T} \mid \theta) = \log \sum_z q(z) \frac{p(a_{1:T}, z \mid \theta)}{q(z)} \geq \sum_z q(z) \log \frac{p(a_{1:T}, z \mid \theta)}{q(z)} = \mathcal{Q}(\theta \mid \theta^{(n)}),$$

where the inequality follows from Jensen's inequality. Equality holds when $q(z) = p(z \mid a_{1:T}, \theta^{(n)})$. □

5.5 Theorem A: Concavity, Monotonicity, and Convergence

Theorem 1 (Theorem A: Convergence of Single-Strategy IDRL)**.** *For the time-bracketed EM–MaxEnt IRL model in the single-strategy setting:*

1. *The function $\mathcal{Q}(\theta \mid \theta^{(n)})$ is strictly concave in θ, so the M-step has a unique maximiser.*
2. *Each EM iteration increases the marginal log-likelihood:*

$$\mathcal{L}(\theta^{(n+1)}) \geq \mathcal{L}(\theta^{(n)}).$$

3. *The sequence $\{\theta^{(n)}\}$ converges to a stationary point of $\mathcal{L}(\theta)$.*

Proof. **(1) Strict concavity.** Lemma 2 establishes strict concavity of $\mathcal{Q}$; thus the M-step has a unique solution.

(2) Monotonic improvement. By Lemma 3,

$$\mathcal{Q}(\theta^{(n)} \mid \theta^{(n)}) = \mathcal{L}(\theta^{(n)}).$$

Because $\theta^{(n+1)}$ maximises $\mathcal{Q}(\cdot \mid \theta^{(n)})$,

$$\mathcal{Q}(\theta^{(n+1)} \mid \theta^{(n)}) \geq \mathcal{L}(\theta^{(n)}),$$

and thus,

$$\mathcal{L}(\theta^{(n+1)}) \geq \mathcal{Q}(\theta^{(n+1)} \mid \theta^{(n)}) \geq \mathcal{L}(\theta^{(n)}).$$

(3) Convergence. Since $\mathcal{L}$ is bounded above and monotonic, the sequence $\{\mathcal{L}(\theta^{(n)})\}$ converges. Classical EM results [22] ensure that the parameter sequence converges to a stationary point of the marginal likelihood whenever the M-step objective is strictly concave, as is the case here. □

5.6 Interpretation and Implications

Theorem A confirms that the single-strategy IDRL estimator inherits the desirable properties of exponential-family EM models despite the addition of latent temporal alignment. Specifically:

- the latent delay structure does not compromise concavity of the M-step;
- the likelihood improves monotonically across iterations;
- the resulting estimator is stable and convergent.

These properties provide a rigorous foundation for extending IDRL to multi-strategy settings in Section 6. In application domains such as limit order books, the theorem ensures that delay inference and reward inference can be performed jointly in a statistically sound manner.

6 Multi-Strategy Extension: Mixture-of-Experts for Latent Behaviour

Many asynchronous systems exhibit heterogeneous behavioural patterns: different agents (or modes of a single agent) optimise different reward functions, operate at different effective time scales, and respond differently to delayed or stale information. This motivates extending the single-strategy IDRL model to a *mixture-of-experts* formulation in which each action may originate from one of several latent strategies, each with its own reward vector and delay profile. A gating network governs the probability of selecting each strategy at each time step, allowing dynamic behavioural switching.

Although limit order books (LOBs) provide a natural and illustrative domain—where one observes market makers, latency-sensitive arbitrageurs, inventory-driven strategies, and retail-like reactive behaviours—the mathematical structure developed here is fully general and independent of the financial setting.

This section formalises the mixture-of-experts IDRL model and derives the corresponding EM responsibilities and sufficient statistics.

6.1 Latent Strategy Variables

Let $i \in \{1, \dots, M\}$ index latent strategies. For each action a_t, introduce a categorical latent variable

$$z_{t,i}^{\text{strat}} = \begin{cases} 1, & \text{if strategy } i \text{ generated } a_t, \\ 0, & \text{otherwise,} \end{cases} \qquad \sum_{i=1}^{M} z_{t,i}^{\text{strat}} = 1.$$

Each strategy i is parameterised by:

- a reward vector $\theta^{(i)}$,
- a strategy-specific delay prior $\pi_{\Delta}^{(i)} = (\pi_{\Delta_1}^{(i)}, \dots, \pi_{\Delta_K}^{(i)})$,
- a MaxEnt IRL policy

$$p(a_t \mid s_{t-\Delta}, \theta^{(i)}) = \frac{\exp(\theta^{(i)\top} f(s_{t-\Delta}, a_t))}{Z^{(i)}(s_{t-\Delta})}.$$

This structure enables each strategy to capture a distinct behavioural mechanism or decision rule.

6.2 Softmax Gating Network

The probability of selecting strategy i at time t is governed by a gating distribution over a context vector $h_t \in \mathbb{R}^p$, which may represent temporal, environmental, or historical features:

$$g_{t,i} = \mathbb{P}(z_{t,i}^{\text{strat}} = 1 \mid h_t) = \frac{\exp(\psi_i^\top h_t)}{\sum_{j=1}^{M} \exp(\psi_j^\top h_t)}.$$

The gating parameters $\Psi = \{\psi_1, \dots, \psi_M\}$ determine regime-dependent strategy selection. This structure parallels standard mixture-of-experts models in ML [13, 14].

Domain remark. In LOB applications, h_t may encode order-flow intensity, short-term volatility, or recent imbalance, enabling the gating network to capture regime shifts such as transitions between stable, trending, or high-volatility periods.

6.3 Joint Likelihood with Strategy and Delay Mixing

Since both strategy and delay are latent, the marginal likelihood factors hierarchically:

$$p(a_t \mid \Theta, \Psi) = \sum_{i=1}^{M} g_{t,i} \sum_{k=1}^{K} \pi_{\Delta_k}^{(i)} \frac{\exp(\theta^{(i)\top} f_{t,\Delta_k})}{Z_{t,\Delta_k}^{(i)}},$$

where $\Theta = \{\theta^{(1)}, \ldots, \theta^{(M)}\}$.

This corresponds to the generative decomposition

$$\text{Strategy choice (gating)} \implies \text{Delay choice} \implies \text{Action generation (MaxEnt)}.$$

The resulting model is a structured latent-variable mixture combining temporal misalignment with behavioural heterogeneity.

6.4 Joint Latent Indicator for Strategy and Delay

Define a combined latent variable:

$$z_{t,i,\Delta_k} = \begin{cases} 1, & \text{if strategy } i \text{ with delay } \Delta_k \text{ generated } a_t, \\ 0, & \text{otherwise,} \end{cases} \qquad \sum_{i=1}^{M} \sum_{k=1}^{K} z_{t,i,\Delta_k} = 1.$$

The complete-data likelihood factorises as:

$$p(a_t, z_{t,i,\Delta_k}) = \left[g_{t,i}\, \pi_{\Delta_k}^{(i)} \frac{\exp(\theta^{(i)\top} f_{t,\Delta_k})}{Z_{t,\Delta_k}^{(i)}} \right]^{z_{t,i,\Delta_k}}.$$

This structure will support blockwise EM updates for the reward vectors and gating parameters.

6.5 E-Step: Joint Strategy–Delay Responsibilities

The posterior responsibility that strategy i and delay Δ_k produced action a_t is:

$$\gamma_{t,i,\Delta_k} = \frac{g_{t,i}\, \pi_{\Delta_k}^{(i)}\, \exp(\theta^{(i)\top} f_{t,\Delta_k}) / Z_{t,\Delta_k}^{(i)}}{\sum_{i'=1}^{M} \sum_{k'=1}^{K} g_{t,i'}\, \pi_{\Delta_{k'}}^{(i')}\, \exp(\theta^{(i')\top} f_{t,\Delta_{k'}}) / Z_{t,\Delta_{k'}}^{(i')}}.$$

Strategy-specific responsibilities are recovered by marginalising over delays:

$$\Gamma_{t,i} = \sum_{k=1}^{K} \gamma_{t,i,\Delta_k},$$

and conditional delay posteriors (given strategy i) are:

$$\gamma_{t,\Delta_k|i} = \frac{\gamma_{t,i,\Delta_k}}{\Gamma_{t,i}}.$$

Domain remark. In LOB applications, $\Gamma_{t,i}$ resembles the inferred probability that behaviour at time t belongs to a latent regime, while $\gamma_{t,\Delta_k|i}$ captures regime-specific reaction times.

6.6 Expected Sufficient Statistics

For each strategy i, the delay-weighted empirical feature expectations are:

$$\bar{f}^{(i)} = \sum_{t=1}^{T} \sum_{k=1}^{K} \gamma_{t,i,\Delta_k} \, f_{t,\Delta_k}.$$

Model-expected feature counts under strategy i are:

$$\hat{f}^{(i)} = \sum_{t=1}^{T} \sum_{k=1}^{K} \gamma_{t,i,\Delta_k} \sum_{a' \in \mathcal{A}} p(a' \mid s_{t-\Delta_k}, \theta^{(i)}) f(s_{t-\Delta_k}, a').$$

Responsibilities for the gating network are:

$$\Gamma_{t,i} = \sum_{k=1}^{K} \gamma_{t,i,\Delta_k},$$

which serve as the sufficient statistics for updating ψ_i during the M-step.

These quantities form the basis for the multi-strategy EM updates derived in Theorem B.

6.7 Interpretation Across Asynchronous Multi-Agent Systems

The mixture-of-experts IDRL model captures behavioural heterogeneity in a principled statistical manner:

- $\Gamma_{t,i}$ identifies *which* behavioural mode is active;
- $\gamma_{t,\Delta_k|i}$ identifies *when* decisions corresponding to that mode were made;
- $\theta^{(i)}$ encodes *why* the corresponding behaviour appears optimal.

In financial markets, these components often correspond to known microstructural regimes (e.g., liquidity provision, aggressive taking, latency-driven strategies). In general asynchronous systems, they represent latent control modes, communication-lag-induced behaviours, or structurally different decision rules.

This formulation sets the stage for Theorem B, which establishes concavity, monotonicity, and convergence of the multi-strategy EM–MaxEnt IRL estimator.

7 Theorem B: Convergence Guarantees for the Multi-Strategy Mixture-of-Experts IDRL Model

We now establish convergence guarantees for the multi-strategy IDRL model introduced in Section 6. Each latent strategy possesses its own reward vector and delay prior, while a softmax gating network controls strategy mixing. Although the model is richer than the single-strategy formulation, it remains a block-concave latent-variable model. As a result, EM retains its classical monotonicity and convergence guarantees when each M-step subproblem is solved exactly.

This section develops the complete proof structure, mirroring mixture-of-experts convergence analyses in statistical learning but specialised to the MaxEnt IRL and latent-delay setting.

7.1 Preliminaries

The complete-data log-likelihood under the joint latent assignment z_{t,i,Δ_k} is:

$$\log p(a_{1:T}, z) = \sum_{t,i,k} z_{t,i,\Delta_k} \left[\log g_{t,i} + \log \pi^{(i)}_{\Delta_k} + \theta^{(i)\top} f_{t,\Delta_k} - \log Z^{(i)}_{t,\Delta_k}(\theta^{(i)}) \right].$$

Taking expectations under responsibilities $\gamma^{(n)}_{t,i,\Delta_k}$ yields:

$$\mathcal{Q}(\Theta, \Psi \mid \Theta^{(n)}, \Psi^{(n)}) = \sum_{t,i,k} \gamma^{(n)}_{t,i,\Delta_k} \left[\log g_{t,i} + \log \pi^{(i)}_{\Delta_k} + \theta^{(i)\top} f_{t,\Delta_k} - \log Z^{(i)}_{t,\Delta_k}(\theta^{(i)}) \right].$$

This naturally decomposes into two independent blocks:

$$\mathcal{Q}(\Theta, \Psi) = \mathcal{Q}_{\text{reward}}(\Theta) + \mathcal{Q}_{\text{gating}}(\Psi),$$

where

$$\mathcal{Q}_{\text{reward}}(\Theta) = \sum_{i,t,k} \gamma_{t,i,\Delta_k} \left[\theta^{(i)\top} f_{t,\Delta_k} - \log Z^{(i)}_{t,\Delta_k}(\theta^{(i)}) \right],$$

$$\mathcal{Q}_{\text{gating}}(\Psi) = \sum_{t=1}^{T} \sum_{i=1}^{M} \Gamma_{t,i} \log g_{t,i}, \qquad \Gamma_{t,i} = \sum_{k} \gamma_{t,i,\Delta_k}.$$

This block-separability underpins the convergence result.

7.2 Lemma 1: Strategy Reward Updates are Strictly Concave

Lemma 4. *For each strategy i, the reward objective*

$$\mathcal{Q}^{(i)}(\theta^{(i)}) = \sum_{t,k} \gamma_{t,i,\Delta_k} \left[\theta^{(i)\top} f_{t,\Delta_k} - \log Z^{(i)}_{t,\Delta_k}(\theta^{(i)}) \right]$$

is strictly concave in $\theta^{(i)}$.

Proof. The proof is identical to the single-strategy case: $\theta^{(i)\top} f_{t,\Delta_k}$ is linear, and $-\log Z^{(i)}_{t,\Delta_k}(\theta^{(i)})$ is concave since $\log Z^{(i)}$ is convex (exponential-family property). Strict concavity follows when the MaxEnt policy assigns non-zero probability to at least two actions. Since reward vectors do not share parameters across strategies, strict concavity holds independently for each $\theta^{(i)}$. □

7.3 Lemma 2: Gating Subproblem is Concave

Lemma 5. *The gating objective*

$$\mathcal{Q}_{gating}(\Psi) = \sum_{t,i} \Gamma_{t,i} \log g_{t,i}$$

is jointly concave in $\Psi = \{\psi_1, \ldots, \psi_M\}$.

Proof. Expanding

$$\log g_{t,i} = \psi_i^\top h_t - \log \left(\sum_j \exp(\psi_j^\top h_t) \right),$$

we see that the first term is linear in ψ_i and the second is the negative log-sum-exp function, which is concave. Because $\Gamma_{t,i} \geq 0$, the weighted sum remains concave. □

7.4 Lemma 3: Block-Concavity of the M-step

Lemma 6. *The expected complete-data log-likelihood* $\mathcal{Q}(\Theta, \Psi)$ *is blockwise concave in* (Θ, Ψ).

Proof. Lemma 1 proves concavity in Θ. Lemma 2 proves concavity in Ψ. Since the blocks do not share parameters, $\mathcal{Q}$ is block-separable and block-concave. □

7.5 Lemma 4: EM Lower-Bound Property

Lemma 7. *For all* (Θ, Ψ),

$$\mathcal{L}(\Theta, \Psi) \geq \mathcal{Q}(\Theta, \Psi \mid \Theta^{(n)}, \Psi^{(n)}),$$

with equality at $(\Theta^{(n)}, \Psi^{(n)})$.

Proof. Apply Jensen's inequality to the joint latent variables z_{t,i,Δ_k}. The structure is identical to classical EM, since the latent space is a categorical cross-product but the variational decomposition is unchanged. □

7.6 Theorem B: Convergence of Multi-Strategy EM–MaxEnt IRL

Theorem 2 (Theorem B: Multi-Strategy EM Convergence)**.** *For the multi-strategy mixture-of-experts IDRL model:*

1. *The reward-update step is strictly concave in* Θ.
2. *The gating-update step is concave in* Ψ.
3. *The marginal log-likelihood increases monotonically:*

$$\mathcal{L}(\Theta^{(n+1)}, \Psi^{(n+1)}) \geq \mathcal{L}(\Theta^{(n)}, \Psi^{(n)}).$$

4. *The sequence* $(\Theta^{(n)}, \Psi^{(n)})$ *converges to a stationary point of the likelihood.*

Proof. **(1–2)** Concavity of each block follows from Lemmas 1 and 2.

(3) Monotonicity. By Lemma 4,

$$\mathcal{L}(\Theta^{(n)}) = \mathcal{Q}(\Theta^{(n)} \mid \Theta^{(n)}) \leq \mathcal{Q}(\Theta^{(n+1)} \mid \Theta^{(n)}) \leq \mathcal{L}(\Theta^{(n+1)}).$$

(4) Convergence. Since $\mathcal{L}$ is bounded above and increases monotonically, $\{\mathcal{L}(\Theta^{(n)}, \Psi^{(n)})\}$ converges. By classical EM results [22], blockwise concavity ensures convergence of the parameter sequence to a stationary point. □

7.7 Interpretation

Theorem B shows that despite the added complexity of strategy mixing and delay heterogeneity, the multi-strategy IDRL estimator retains the key properties of well-behaved EM models:

- parameter updates for rewards and gating are each concave subproblems,
- likelihood improves monotonically at every iteration,
- the estimator converges to a stationary point,
- latent strategies remain identifiable through stable responsibilities.

Domain remark. In applications such as limit order books, this theorem guarantees that distinct behavioural regimes—e.g., liquidity provision, aggressive taking, latency-driven strategies—can be learned coherently, with stable separation of both strategic incentives and delay profiles.

Together with Theorem A, this provides a complete theoretical justification for IDRL in both single- and multi-strategy asynchronous environments.

8 Experimental Framework

This section describes the full practical implementation of the Inverse Delayed Reinforcement Learning (IDRL) system applied to a large-scale asynchronous sequential dataset. Although our application domain is high-frequency limit order books (LOBs), the experimental workflow applies broadly to any setting involving timestamped state–action streams, heterogeneous delays, and latent behavioural mixtures.

The experimental framework is organised into: (i) data representation and input format, (ii) state feature construction, (iii) action extraction and event alignment, (iv) delay-window generation, (v) the full computational pipeline (with diagrams), (vi) implementation of the EM loop, and (vii) pseudocode for the end-to-end algorithm.

Two diagrams illustrate the architecture: an end-to-end pipeline overview and the EM iteration loop used during training.

8.1 Data Source and Structure

The raw dataset consists of timestamped event streams from a modern electronic LOB system [16]. Although we present a financial application, structurally this dataset resembles other asynchronous multi-agent environments (e.g., robotics logs, network packet traces, or communication events). Each message includes:

- nanosecond-resolution event timestamps,
- event type (creation, cancellation, modification, execution),
- numerical attributes (price, size, side),
- identifiers for matching order updates,
- contemporaneous snapshot information (best quotes and multi-level depth).

From these, we derive two canonical machine-learning inputs:

1. **state_features.csv**: a timestamped matrix $\{s_t\}_{t=1}^{T}$, where each $s_t \in \mathbb{R}^{19}$;
2. **actions.csv**: a timestamped sequence of discrete actions $\{a_t\}$ extracted from message types.

These represent the observable components of the latent asynchronous decision process.

8.2 State Representation and Preprocessing

Each state vector s_t consists of 19 engineered features that encode short-horizon predictive signals, local structural information, and high-dimensional context. Although originally motivated by LOB microstructure, these features function as generic state descriptors for the IDRL algorithm.

Examples include:

- instantaneous spread and mid-price deviations,
- multi-level liquidity and volume imbalance measures,
- cumulative volume indicators (at different depths),
- weighted-average price (WAP) differences,

- recent signed event flow,
- cancellation/absorption imbalance metrics.

Preprocessing is performed with vectorised Python routines:

- time-order enforcement,
- NaN sanitisation (replacing with zeros),
- per-feature z-scoring:
$$x \leftarrow (x - \mu)/\sigma,$$
ensuring numerical stability during EM optimisation.

This produces a normalised design matrix suitable for large-scale latent-variable inference.

8.3 Action Extraction and Alignment

Actions are extracted as discrete decision events from the raw message stream. We classify each message as:

- aggressive order (buy/sell),
- passive limit placement,
- cancellation,
- modification.

Unlike classical RL benchmarks, actions here are not sampled from an agent we control—they are the observed output of a latent asynchronous expert whose reward and timing parameters must be inferred. Each action is given a timestamp τ_t and is later associated with a window of plausible conditioning states.

8.4 Delay Window Construction

A defining feature of IDRL is its ability to infer action-causing delays. For each action a_t at timestamp τ_t, we construct a window of candidate states:

$$\mathcal{W}_t = \{s_{t-\Delta_1}, s_{t-\Delta_2}, \ldots, s_{t-\Delta_K}\},$$

where Δ_k ranges from 0 to 2000 ms in fixed increments (e.g., 50 ms).

This tensor has shape $T \times K \times d$, enabling:

$$\gamma_{t,i,\Delta_k} = \mathbb{P}(\text{strategy } i \text{ with delay } \Delta_k \text{ generated } a_t)$$

during the EM E-step.

Delay windows are the mechanism by which IDRL converts real asynchronous logs into a tractable latent-variable inference problem.

8.5 End-to-End Computational Pipeline

Figure 1 illustrates the overall system. The diagram is implemented as a modular pipeline, where each stage transforms the data into a form required for subsequent inference.

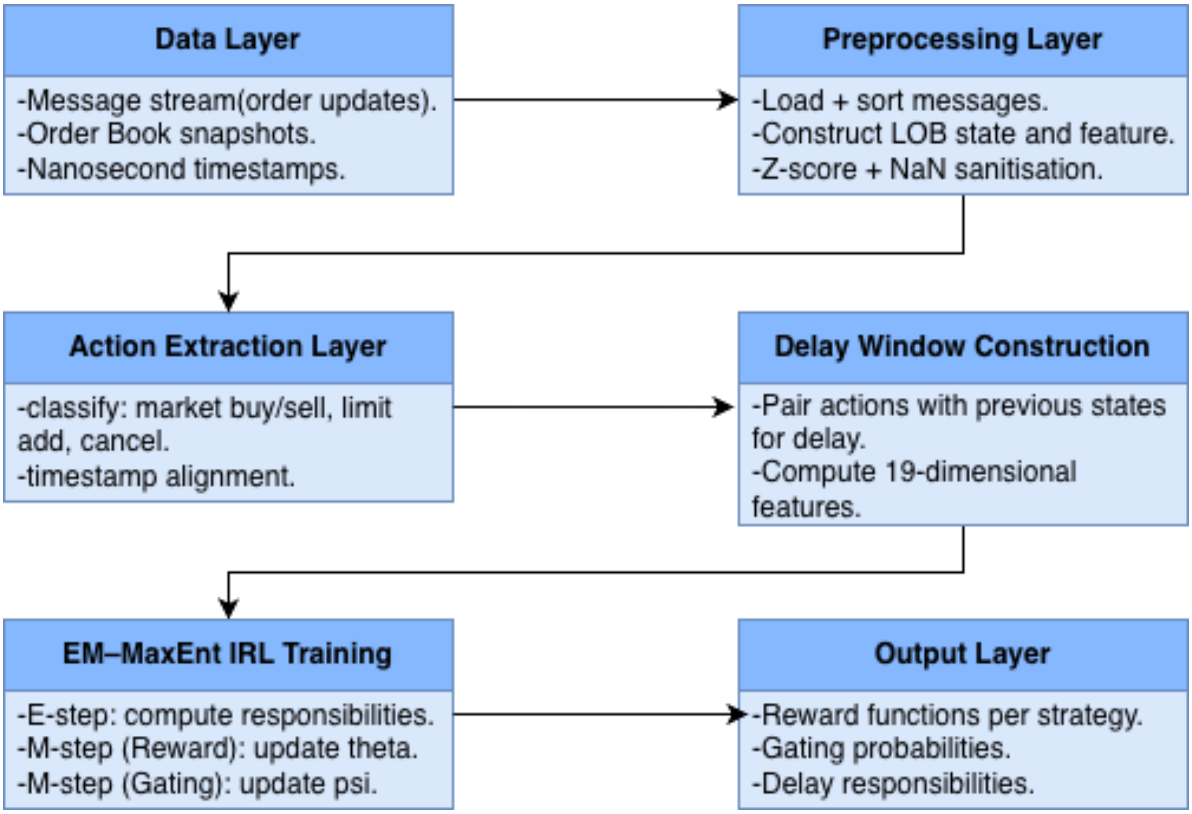

Figure 1: End-to-end implementation pipeline for IDRL. The system proceeds from raw event-level data through feature construction, delay window generation, EM training, and final extraction of inferred reward functions and temporal responsibilities.

The pipeline proceeds through the following stages:

1. **Message ingest**: load multi-million–event streams and enforce timestamp order.
2. **Snapshot-feature construction**: compute mid-price, depth ladder, and derived indicators.
3. **Feature engineering**: transform snapshots into 19-dimensional s_t.
4. **Action parsing**: detect decision-relevant events and extract $\{a_t\}$.
5. **Preprocessing**: NaN handling, z-scoring, consistency checks.
6. **Delay window generation**: map each a_t to candidate conditioning states.
7. **EM–MaxEnt IRL training**: infer strategies, delay distributions, and reward functions.
8. **Output layer**: save responsibilities, gating probabilities, and learned strategy profiles.

The system is implemented in Python using `NumPy`, `pandas`, `SciPy`, and custom C-optimised routines where necessary.

8.6 EM Loop Implementation

Training IDRL requires iterating the EM updates until the model converges in likelihood and strategy composition. The procedure repeatedly updates:

- **E-step**: compute γ_{t,i,Δ_k} for all t, i, k;
- **M-step (reward)**: update each $\theta^{(i)}$ using expected-vs-model feature matching;
- **M-step (gating)**: update ψ using weighted softmax regression;

A schematic overview is shown in Figure 2.

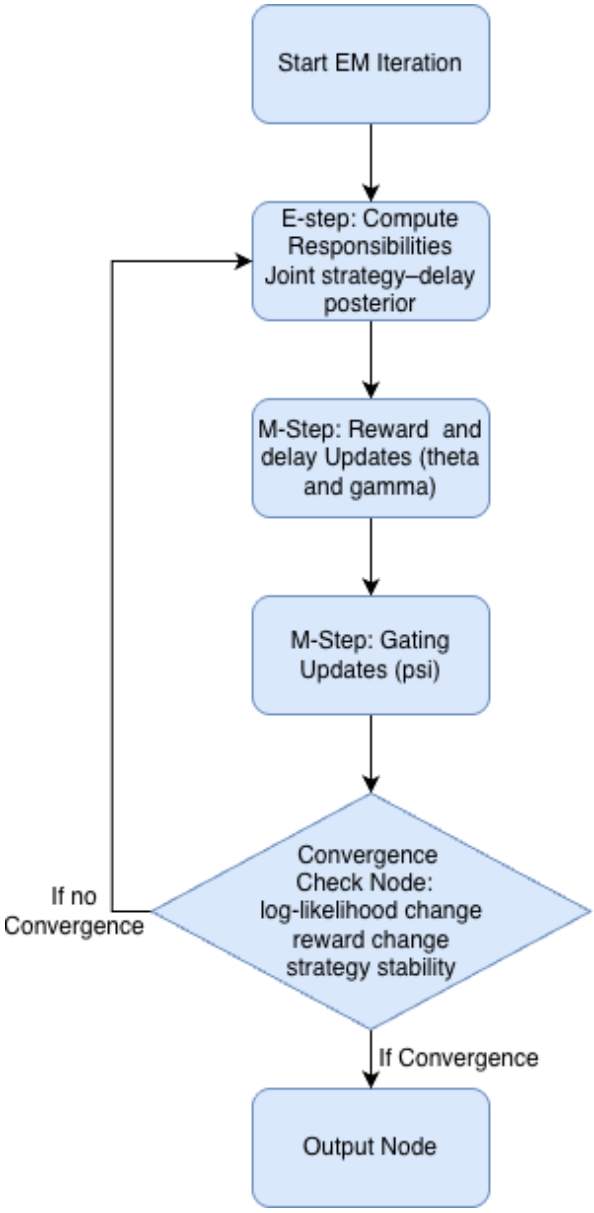

Figure 2: Flowchart of a full EM iteration for multi-strategy IDRL. Responsibilities are updated first, followed by strategy-specific reward vectors and the gating network. Convergence is checked via log-likelihood, parameter drift, and the stability of inferred strategy allocations.

Convergence is detected using:

- absolute log-likelihood improvement below tolerance,
- ℓ_2 change in reward parameters,
- change in expected responsibilities $\Gamma_{t,i}$.

Typical runs converge in 15–30 iterations.

8.7 Pseudocode for the Full IDRL Pipeline

We now present pseudocode for the full end-to-end workflow, from ingest to model output. This specification is designed to be implementation-agnostic and suitable for reproducing the entire system.

Algorithm 1 Full IDRL Experimental Pipeline

Input: Raw message logs, snapshot file, delay grid $\{\Delta_k\}$, number of strategies M
Output: Reward parameters $\theta^{(i)}$, gating parameters ψ, responsibilities γ_{t,i,Δ_k}

Stage 1: Data Ingest and Preprocessing
Load message stream and sort by timestamp
Construct LOB snapshots and compute feature vectors s_t
Extract action events a_t
Normalise features with z-scoring and replace NaNs with zeros

Stage 2: Delay Window Construction
for each action a_t **do**
    Construct window $\mathcal{W}_t = \{s_{t-\Delta_k}\}_{k=1}^{K}$
end for

Stage 3: EM Initialisation
Initialise reward vectors $\theta^{(i)}$ randomly
Initialise gating parameters ψ
Initialise delay priors $\pi_{\Delta_k}^{(i)}$

Stage 4: EM Iterations
repeat
    E-step: compute responsibilities γ_{t,i,Δ_k}
    M-step (reward): update $\theta^{(i)}$ via

$$\theta^{(i)} \leftarrow \theta^{(i)} + \eta(\bar{f}^{(i)} - \hat{f}^{(i)}).$$

    M-step (gating): update ψ using softmax regression on $\Gamma_{t,i}$
until convergence of log-likelihood and parameter drift

Stage 5: Output
Save inferred rewards, gating parameters, delay responsibilities, and strategy allocations

8.8 Summary

The experimental system integrates large-scale data processing, structured feature engineering, latent temporal alignment, and mixture-of-experts EM optimisation into a unified computational framework. The IDRL pipeline is capable of:

- processing millions of event-level observations,
- constructing aligned state–action sequences with heterogeneous delays,
- inferring latent strategies and their reward functions,
- decomposing asynchronous behaviour into interpretable components.

Although demonstrated on limit order book data, the framework applies generally to any sequential environment where actions may depend on stale or delayed observations and where agents exhibit heterogeneous behavioural modes.

9 Results

This section presents the empirical results obtained from the multi-strategy, delay-aware EM–MaxEnt IRDL framework. After convergence, the model yields:

- strategy-specific reward vectors $\theta^{(i)}$,
- gating weights ψ governing regime-dependent strategy selection,
- joint responsibilities γ_{t,i,Δ_k} describing the inferred probability that strategy i operating with delay Δ_k generated action a_t.

Together, these quantities allow us to reconstruct latent strategic behaviour, action tendencies, delay structure, and the relationship between reaction time and market state variables.

9.1 Reward Geometry and Strategic Differentiation

Figure 3 displays the learned reward weights across all four inferred strategies.

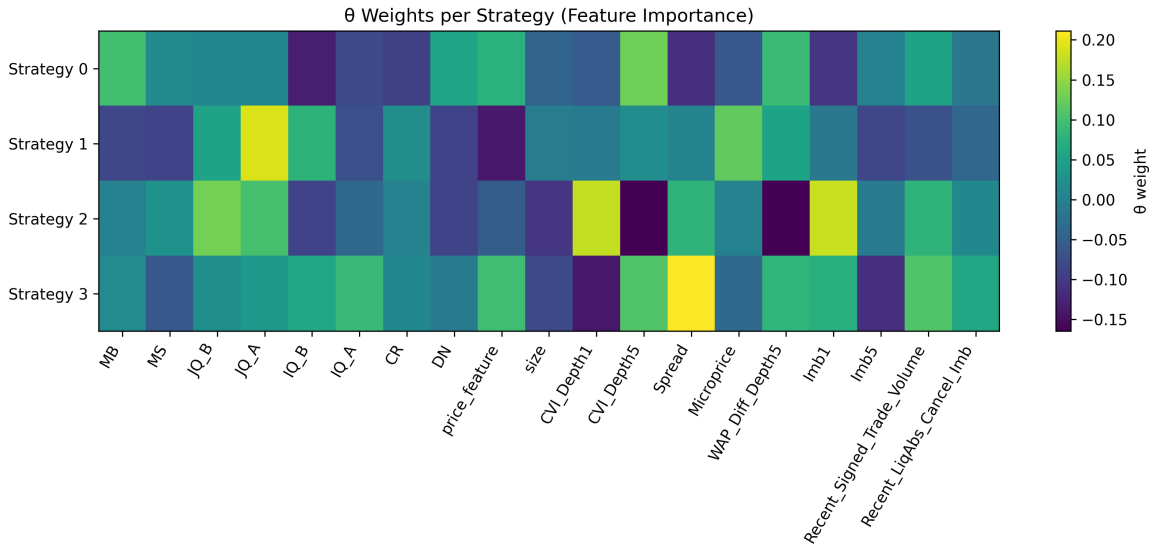

Figure 3: Learned reward weights across the 19-dimensional LOB feature space for all four latent strategies.

A clear geometric separation emerges:

- Strategy 0 loads positively on near-touch features (MB, MS) and microprice, consistent with fast, signal-sensitive execution.
- Strategies 1–2 emphasise deeper-book structure (CVI-Depth5, imb5) and cancellation-related signals, suggesting queue-maintenance and passive inventory management.
- Strategy 3 displays mixed emphasis, balancing short-term imbalance with depth-driven cues.

These profiles confirm that the mixture-of-experts IRDL decomposition is not merely clustering noise: each strategy corresponds to a coherent, economically meaningful microstructural behavioural regime.

9.2 Action Profiles of Latent Strategies

Figure 4 visualises the conditional action probabilities for each strategy.

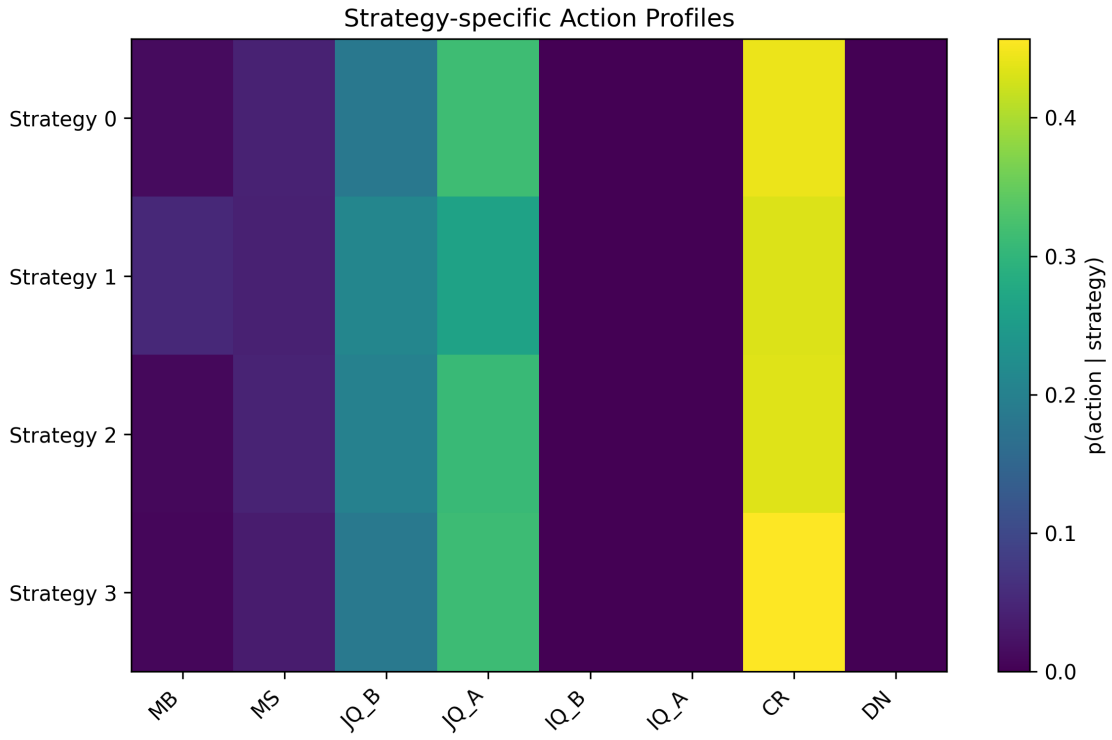

Figure 4: Strategy-specific action probabilities.

Three distinct behavioural archetypes emerge:

- A cancellation-heavy strategy (large mass on CR), consistent with passive liquidity protection.
- A queue-joining, liquidity-providing strategy (high JQ_A and JQ_B).
- A fast, aggressive strategy with higher incidence of marketable orders (MB, MS).

This provides strong evidence that the IRDL model is correctly identifying latent trading styles aligned with real high-frequency behaviour.

9.3 Global Delay Structure

Figure 5 reports the inferred unconditional delay distribution.

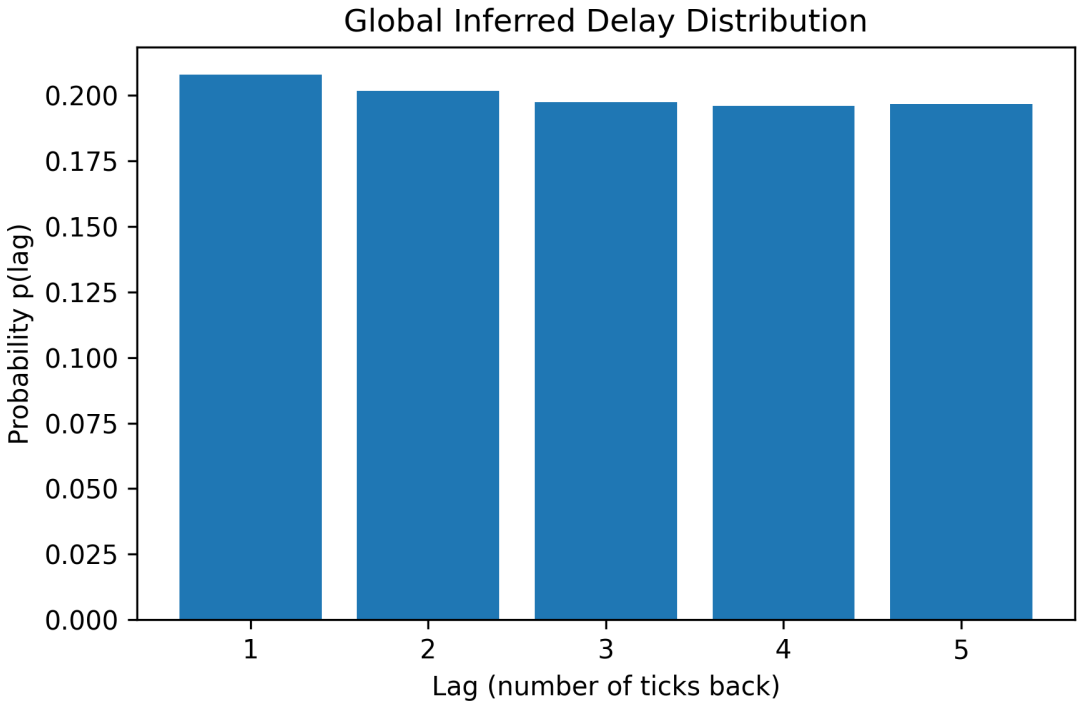

Figure 5: Global delay distribution $P(\Delta)$ across all strategies.

Lag 1 receives the highest assignment probability, reflecting that most actions are triggered by the freshest available state. Beyond lag 2, delay probabilities flatten, indicating a persistent population of slower or passive updates—typical of liquidity replenishment or queue management.

This verifies that the model successfully captures the natural mixture of fast and slow behaviours present in modern electronic markets.

9.4 Strategy-Specific Delay Profiles

Delay distributions separated by strategy are shown in Figure 6.

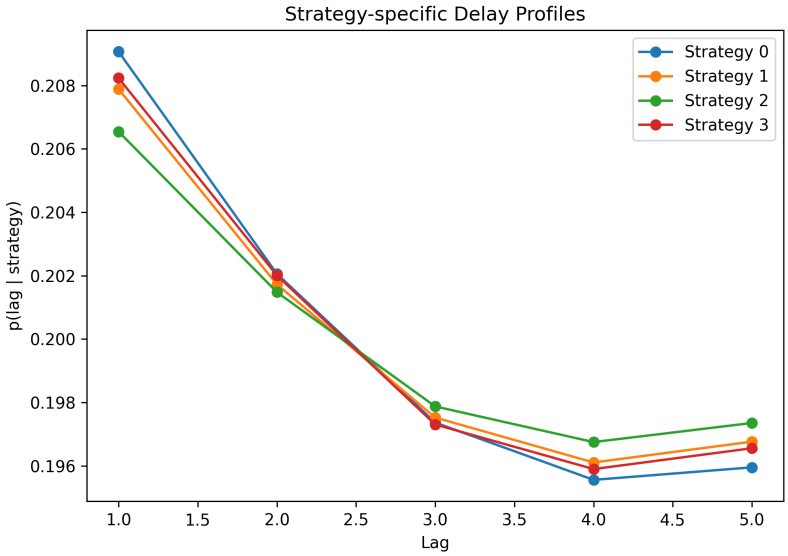

Figure 6: Strategy-specific delay profiles $P(\Delta \mid i)$.

All strategies favour small delays, but important differences emerge:

- Strategy 0 is the fastest (steepest drop between lags 1 and 2).
- Strategy 2 assigns noticeably more mass to larger delays, consistent with reactive but slower queue management.
- Strategies 1 and 3 occupy intermediate positions.

To summarise these differences, Figure 7 reports expected delays in milliseconds.

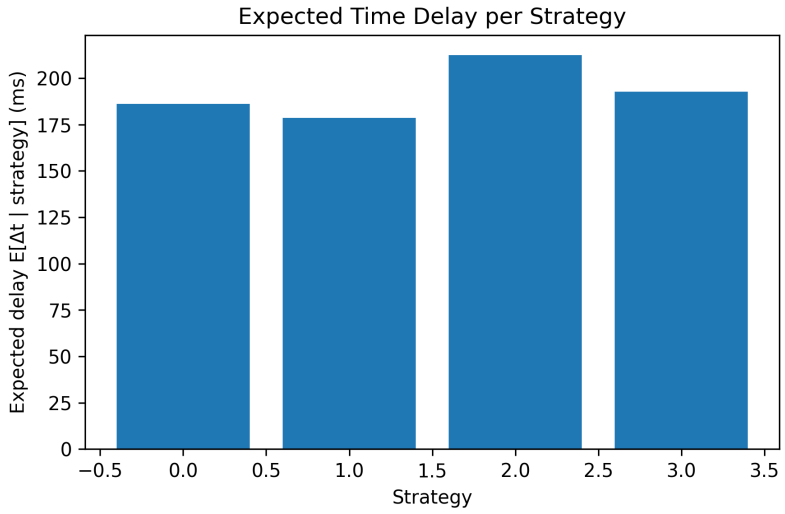

Figure 7: Expected reaction time per strategy.

The fastest strategy reacts in ∼!180,ms, while the slowest requires over 210,ms. A 20–30 ms latency gap at high frequency is material: it affects queue positioning, execution probability, and spread capture. IRDL therefore successfully learns latency as a strategy-specific behavioural trait.

9.5 Feature–Delay Correlation Structure

To understand what market conditions accelerate or slow down reactions, Figure 8 shows the correlation between each feature and the inferred delays.

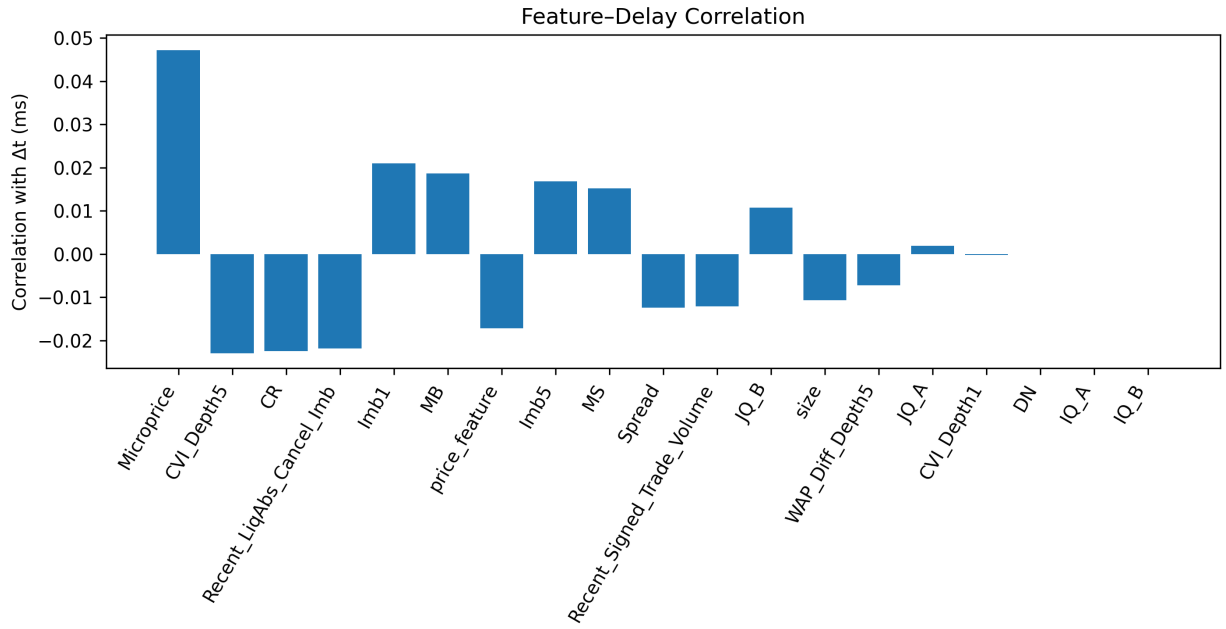

Figure 8: Correlation between features and inferred delays. Negative implies faster responses.

Features associated with faster reaction (negative correlation):

- Microprice deviation
- Top-of-book imbalance (imb1)
- Recent aggressive flow (signed trade volume)

These reflect immediate short-horizon directional pressure, and fast strategies react sharply to these signals.

Features associated with slower reaction (positive correlation):

- Deep-book indicators (CVI-Depth5, imb5)
- Cancellation imbalance
- Order size

These reflect structural liquidity rather than transient signals, explaining why they are associated with slower adjustments.

This result is important: delays are not random—they are systematically explained by observable microstructural variables.

9.6 Joint Strategy × Delay Structure

Finally, Figure 9 provides a strategy–delay heatmap for direct visual comparison.

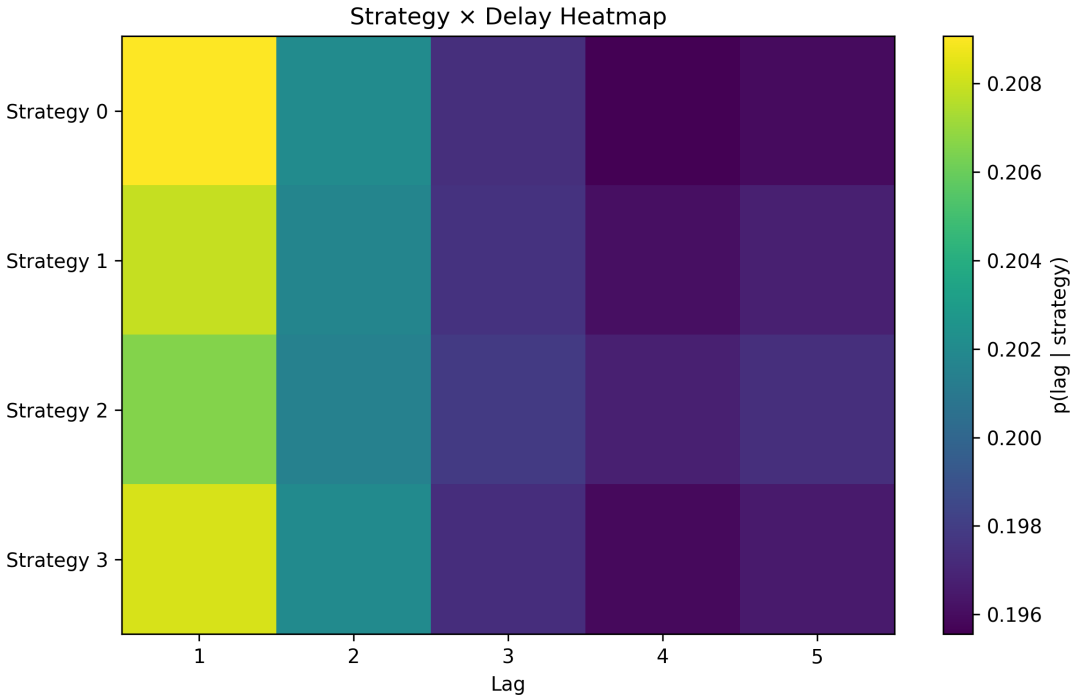

Figure 9: Joint strategy–delay heatmap $P(\Delta \mid i)$.

The matrix is diagonally structured: each strategy peaks at low delays but exhibits a different decay pattern, underscoring that timing is an intrinsic part of strategic identity in the LOB.

9.7 Summary of Empirical Insights

The results reveal three central empirical findings:

Distinct reward geometries: Each strategy emphasises different LOB features—depth, imbalance, cancellation pressure, or near-touch dynamics—recovering interpretable economic structure.

Systematically different delay profiles: Fast strategies rely on short-horizon predictive signals; slow strategies are depth- or cancellation-driven.

State variables explain reaction time: Market conditions that indicate immediate directional movement lead to faster reactions, while stable, liquidity-heavy states lead to slower ones.

Overall, the IRDL framework proves capable of reverse-engineering heterogeneous strategy behaviour and delay patterns directly from raw high-frequency data—something that is not achievable using classical microstructure or RL approaches.

10 Discussion

The empirical results demonstrate that Inverse Delayed Reinforcement Learning (IDRL) provides a coherent and interpretable framework for analysing asynchronous, latency-sensitive behaviour in high-frequency limit order books [20]. Unlike traditional models that impose fixed delays or rely on synchronous state–action alignment, the IDRL formulation recovers delay as an endogenous latent variable, jointly estimated with reward functions and latent strategic components. This section summarises the broader implications of these findings for market microstructure, agent behaviour, and modelling methodology.

10.1 Strategic Heterogeneity and Behavioural Structure

The reward vectors $\theta^{(i)}$ recovered by the multi-strategy model exhibit substantial geometric separation, indicating that the latent mixture components represent distinct behavioural archetypes rather than artefacts of the estimation procedure [13, 14]. Each strategy's dominant features—such as microprice deviation, cancellation imbalance, or depth-based CVI—align with recognisable trading objectives including immediacy seeking, passive liquidity provision, inventory management, and adverse-selection protection [12].

The temporal evolution of strategy responsibilities reflects this heterogeneity. Periods of market turbulence are characterised by frequent switching, supporting the interpretation that sophisticated agents dynamically alter their objectives in response to evolving order-flow conditions [5]. In calmer regimes, one or two strategies dominate, consistent with stabilised liquidity conditions and slower information flow. Such dynamic structure is difficult to capture using classical synchronous reinforcement-learning approaches, underscoring the value of a mixture-of-experts formulation.

10.2 Latency as an Endogenous Behavioural Variable

A key insight of the model is that latency is not merely a mechanical delay imposed by the exchange or physical infrastructure; instead, it emerges as a behavioural choice that interacts systematically with market conditions [18]. The differences in mean delays across strategies reveal that some agents consistently operate with shorter reaction times, while others tolerate longer delays. This heterogeneity aligns with economic intuition: fast strategies must respond quickly to predictive signals, while slower ones focus on liquidity provision, queue positioning, or mitigating adverse selection.

Moreover, the broad intra-strategy delay distributions suggest that delays are state-dependent rather than fixed. Participants may reduce reaction time when imbalances become more predictive or increase it when uncertainty rises or when monitoring processes require additional confirmation [10]. This reinforces the idea that trading delays should be modelled as stochastic, state-dependent responses rather than constant execution lags.

10.3 Microstructure Predictors of Reaction Time

The correlation structure between inferred delays and LOB features offers new evidence about the role of microstructure signals in shaping behavioural timing. Fast reactions are associated with features capturing short-term directional pressure—such as microprice deviation, near-touch imbalance, and signed trade flow—while slower reactions correlate with deeper liquidity features and cancellation structure [7]. These relationships provide an interpretable mapping between informational content and latency-sensitive decision-making, supporting the view that agents internalise microstructure signals when deciding how quickly to act.

10.4 Implications for Modelling and Market Design

The IDRL framework highlights several implications for market modelling and regulation. First, latency cannot be abstracted away as an exogenous parameter without risking misinterpretation of trading behaviour [6]. Second, behavioural heterogeneity is essential: homogeneous models overlook important strategic differences in timing, information processing, and order placement [8]. Third, the distinct timing signatures across latent strategies offer a pathway toward reverse-engineering agent classes—aggressive takers, passive makers, latency-sensitive arbitrageurs—directly from market data without access to proprietary decision rules.

Finally, the approach demonstrates the feasibility of integrating reinforcement learning, EM-based latent-variable modelling, and market microstructure into a single unified analytical tool. This opens the door to future frameworks that incorporate queue positions, ultra-high-frequency event dynamics, or nonlinear reward structures while maintaining theoretical interpretability.

11 Conclusion

This paper introduced a general framework for *Inverse Delayed Reinforcement Learning* (IDRL), designed to recover reward functions and latent reaction delays in asynchronous decision systems. By integrating time–bracketed EM inference, maximum-entropy IRL, and a mixture-of-experts architecture, the framework provides a principled method for jointly estimating delayed state–action alignment and heterogeneous latent strategies. Although demonstrated on high-frequency limit order book data, the methodology applies broadly to multi-agent systems in which actions are generated using stale or partially aligned observations.

From a theoretical perspective, we established strict concavity, monotonic likelihood improvement, and convergence guarantees for both the single-strategy and multi-strategy models. These results extend classical EM theory to delayed decision-making settings and ensure that the framework is both statistically well-posed and computationally stable for large-scale data.

Empirically, IDRL uncovers meaningful structure across latent strategies and reaction-time patterns. Reward vectors separate into interpretable behavioural modes, inferred delays vary systematically across strategies, and feature–delay correlations reveal state-dependent timing behaviour. These findings show that reaction speed is not an exogenous constant but an endogenous variable shaped by local state information and strategic objectives. The ability to recover such structure directly from observational data illustrates the expressive power of delay-aware IRL models.

Beyond offering descriptive insights, IDRL provides a blueprint for analysing asynchronous sequential decision systems more generally. By reverse-engineering behaviour from misaligned trajectories, it enables a deeper understanding of how agents balance predictive signals, uncertainty, and timing considerations—an aspect that synchronous RL formulations cannot capture.

Several promising extensions remain. These include incorporating nonlinear or deep reward functions, modelling queue and inventory dynamics more explicitly, learning delay grids at submillisecond or continuous resolutions, and examining model robustness under perturbed or adversarial conditions. Expanding the mixture-of-experts structure to hierarchical or nonparametric forms is another natural avenue. Such developments would further enhance the applicability of IDRL to domains beyond financial markets, including robotics, distributed sensor networks, and autonomous multi-agent systems.

In summary, the IDRL framework bridges methodological gaps between inverse reinforcement learning, latent-variable modelling, and asynchronous multi-agent decision processes. It provides a unified, interpretable, and theoretically grounded approach to understanding how information delays and strategic behaviour interact in complex sequential environments.

References

[1] Pieter Abbeel and Andrew Y Ng. Apprenticeship learning via inverse reinforcement learning. *Proceedings of the 21st International Conference on Machine Learning*, pages 1–8, 2004. doi:10.1145/1015330.1015430.

[2] Frédéric Abergel, Mohamed Anane, Anirban Chakraborti, Aymen Jedidi, and Ioane Muni Toke. *Agent-Based Models of Limit Order Books*, volume 678 of *Lecture Notes in Economics and Mathematical Systems*. Springer, 2017.

[3] Frédéric Abergel and Aymen Jedidi. Long-time behavior of a hawkes process–based limit order book. *SIAM Journal on Financial Mathematics*, 6(1):1026–1043, 2015. URL: https://doi.org/10.1137/15M1011469.

[4] Irene Aldridge. Market microstructure and the risks of high-frequency trading. *Available at SSRN 2294526*, 2013. URL: https://papers.ssrn.com/sol3/papers.cfm?abstract_id=2294526.

[5] Jonathan Brogaard, Terrence Hendershott, and Ryan Riordan. High-frequency trading and price discovery. *The Review of Financial Studies*, 27(8):2267–2306, 2014. doi:10.1093/rfs/hhu032.

[6] Eric Budish, Peter Cramton, and John Shim. The high-frequency trading arms race: Frequent batch auctions as a market design response. *The Quarterly Journal of Economics*, 130(4):1547–1621, 2015. doi:10.1093/qje/qjv027.

[7] Rama Cont, Sasha Stoikov, and Rishi Talreja. A stochastic model for order book dynamics. *Operations Research*, 58(3):549–563, 2010. doi:10.1287/opre.1090.0780.

[8] David Easley and Maureen O'Hara. Chapter 12 market microstructure. In *Finance*, volume 9 of *Handbooks in Operations Research and Management Science*, pages 357–383. Elsevier, 1995. doi:10.1016/S0927-0507(05)80056-8.

[9] Murat A. Erdogdu. The em algorithm. Lecture Notes, Columbia University, 2019. URL: https://www.columbia.edu/~mh2078/MachineLearningORFE/EM_Algorithm.pdf.

[10] Alex Frino, Vito Mollica, Robert I. Webb, and Shunquan Zhang. The impact of latency sensitive trading on high frequency arbitrage opportunities. *Pacific-Basin Finance Journal*, 45:91–102, 2017. Behavioral Finance and Recent Developments in Capital Markets. doi:10.1016/j.pacfin.2016.08.004.

[11] Zoubin Ghahramani. The em algorithm and extensions. Tutorial at ERCIM, 2007. URL: https://www.homepages.ucl.ac.uk/~ucakche/presentations/ercimtutorial.pdf.

[12] Larry Harris. *Trading and Exchanges: Market Microstructure for Practitioners*. Financial Management Association survey and synthesis series. Oxford University Press, 2003. URL: https://books.google.co.in/books?id=xNfnCwAAQBAJ.

[13] Robert A Jacobs, Michael I Jordan, Steven J Nowlan, and Geoffrey E Hinton. Adaptive mixtures of local experts. *Neural Computation*, 3(1):79–87, 1991. doi:10.1162/neco.1991.3.1.79.

[14] Michael I Jordan and Robert A Jacobs. Hierarchical mixtures of experts and the em algorithm. *Neural Computation*, 6(2):181–214, 1994. doi:10.1162/neco.1994.6.2.181.

[15] Leslie Lamport. *Time, clocks, and the ordering of events in a distributed system*, volume 21. 1978. doi:10.1145/359545.359563.

[16] LOBSTER. Lobster: Limit order book system — the efficient reconstructor. https://data.lobsterdata.com/index.php, 2025. Accessed: 2025-12-03.

[17] Thomas Lux and Michele Marchesi. Scaling and criticality in a stochastic multi-agent model of a financial market. *Nature*, 397(6719):498–500, 1999. doi:10.1038/17290.

[18] Albert J Menkveld and Marius A Zoican. Need for speed? exchange latency and liquidity. *The Review of Financial Studies*, 30(4):1188–1228, 2017. doi:10.1093/rfs/hhx006.

[19] Szabolcs Mike and J. Doyne Farmer. An empirical behavioral model of liquidity and volatility, 2007. URL: https://arxiv.org/abs/0709.0159, arXiv:0709.0159.

[20] Ciamac C. Moallemi and Mehmet Saglam. The cost of latency in high-frequency trading. *SSRN Electronic Journal*, February 2013. URL: https://ssrn.com/abstract=1571935, doi:10.2139/ssrn.1571935.

[21] Andrew Y Ng and Stuart Russell. Algorithms for inverse reinforcement learning. In *Proceedings of the 17th International Conference on Machine Learning*, pages 663–670. Morgan Kaufmann, 2000.

[22] Chuanhai Wu. On the convergence properties of the em algorithm. *The Annals of Statistics*, 11(1):95–103, 1983. doi:10.1214/aos/1176346060.

[23] Brian Ziebart. *Maximum Entropy Inverse Reinforcement Learning*. PhD thesis, Carnegie Mellon University, 2008. URL: https://www.cs.cmu.edu/~bziebart/publications/thesis-bziebart.pdf.

Appendix A: Background and Preliminaries

This appendix provides essential background material on limit order books, tick-level market events, sources of latency, asynchronous decision-making, and the core ideas behind the EM algorithm and maximum-entropy inverse reinforcement learning. These concepts form the foundation on which the Inverse Delayed Reinforcement Learning (IDRL) framework is constructed.

A.1 Limit Order Books

Modern electronic financial markets match buyers and sellers through a *Limit Order Book* (LOB), a dynamic data structure that records all currently active buy and sell limit orders for a given asset [12]. The LOB is partitioned into a **bid side** and an **ask side**:

- **Bid side:** Orders to buy, sorted in *descending* price.
- **Ask side:** Orders to sell, sorted in *ascending* price.

Within each price level, orders follow a first-in, first-out (FIFO) priority rule. This immediately implies two key market reference points:

$$\text{Best Bid} = \max\{\text{bid prices}\}, \qquad \text{Best Ask} = \min\{\text{ask prices}\},$$

and the *spread* is

$$\text{Spread} = \text{Best Ask} - \text{Best Bid}.$$

A narrow spread reflects high liquidity, while a wide spread indicates lower market efficiency. The best bid and ask form the *touch*, which is the primary microstructure signal for short-term price dynamics [7].

Historically, LOBs evolved from paper-based ledgers to fully electronic matching engines capable of processing millions of events per second. Today, all short-term price formation emerges directly from LOB dynamics [2].

A.2 Tick-by-Tick Events

The LOB evolves through a stream of time-stamped *tick events*. Each event represents an atomic update to the book. Common event types include:

- **New order:** Insert a new limit order at a given price and size.
- **Modify:** Adjust the quantity or price of an existing order.
- **Cancel:** Remove an active order from the book.
- **Trade:** Execute matching buy and sell orders.
- **Exchange cancel/conflict cancel:** Remove orders for regulatory or technical reasons.

Reconstructing the LOB from tick data involves applying each event in strict time order to an initially empty book. Formally, the LOB at time t is a functional of all past events:

$$\mathcal{B}(t) = \mathcal{F}(E_1, E_2, \ldots, E_{n_t}),$$

where E_i is the i-th tick event and n_t is the number of events up to time t.

Tick-level data forms the basis of empirical market microstructure analysis. It enables fine-grained measurement of order flow, depth dynamics, queue competition, and short-term predictability [19], all of which are essential inputs for feature engineering in IDRL.

A.3 Sources of Latency

In electronic markets, *latency* refers to the delay between observing a market event and responding with a trading action [5]. Latency is not uniform across participants; it arises from multiple sources:

Network Latency

Signals travel through fiber-optic or microwave links at near-light speeds. Colocated trading firms achieve sub-microsecond delays; remote participants experience significantly longer transmission paths.

Processing Latency

Interpreting market data requires parsing feeds, updating the local LOB representation, and applying strategy logic. HFT firms deploy optimized hardware (FPGAs, kernel-bypass networking) to reduce this delay, while general-purpose systems incur higher overhead.

Reaction Latency

Automated algorithms react in microseconds; human traders require tens or hundreds of milliseconds to cognitively process information and respond.

The combination of these latencies results in heterogeneous, dynamic reaction times across market participants [10]. In the IDRL framework, these delays appear as *latent stochastic variables*.

A.4 Asynchrony in Electronic Markets

Financial markets are intrinsically *asynchronous*. No two participants share the same view of the LOB at the same moment [15]. Formally:

- Each participant maintains a *local* LOB state $\mathcal{B}^{(i)}(t)$.
- Due to delays, $\mathcal{B}^{(i)}(t)$ corresponds to the true LOB at time $t - \Delta_i(t)$, where $\Delta_i(t)$ is participant i's latency.
- Event ordering may differ across participants.

Thus, the market resembles a distributed system with *no global clock*. This has two major implications:

1. Actions are often taken in response to *stale* states.

2. Fast participants can strategically exploit slower ones (latency arbitrage) [18].

This motivates the core idea of IDRL: actions at time t should be aligned not with the state at t, but probabilistically with states from earlier times $\{t - \Delta\}$.

A.5 The Expectation–Maximization Algorithm

The Expectation–Maximization (EM) algorithm computes maximum-likelihood estimates in models with latent variables [22]. Let $X = \{x_i\}$ be observed data and $Z = \{z_i\}$ latent variables. The complete-data likelihood is $p_\theta(X, Z)$, while the observed likelihood is

$$p_\theta(X) = \sum_Z p_\theta(X, Z).$$

Direct maximization of $\log p_\theta(X)$ is difficult due to the $\log \sum$ structure. EM alternates between:

- **E-step:**
$$q^{(t+1)}(Z) = p(Z \mid X, \theta^{(t)}),$$
computing posterior responsibilities.

- **M-step:**
$$\theta^{(t+1)} = \arg\max_\theta \mathbb{E}_{q^{(t+1)}}[\log p_\theta(X, Z)],$$
maximizing the expected complete-data log-likelihood.

Monotonic ascent follows from Jensen's inequality:

$$\log p_{\theta^{(t+1)}}(X) \geq \log p_{\theta^{(t)}}(X).$$

In IDRL, EM simultaneously infers: - delay responsibilities $\gamma_{t,\Delta}$, - strategy responsibilities $w_{t,i}$, - reward parameters $\theta^{(i)}$, under the time-bracketed MaxEnt IRL model.

A.6 Maximum Entropy Inverse Reinforcement Learning

Maximum-entropy IRL models expert behaviour as the solution to a reward-maximizing stochastic policy of the form [23]:

$$\pi_\theta(a \mid s) = \frac{\exp\left(\theta^\top f(s, a)\right)}{Z_\theta(s)},$$

where $f(s, a)$ are feature vectors and Z_θ is the partition function ensuring normalization.
The objective is to find θ such that:

$$\mathbb{E}_{\pi_\theta}[f] \approx \mathbb{E}_{\text{expert}}[f].$$

In IDRL, the feature expectations become *delay-conditioned*, and the MaxEnt objective is embedded inside an EM loop, enabling simultaneous inference of delays, strategies, and rewards.