Universität Zürich^UZH

Inverse RL Shrinkage: Learning Expert Demonstrations of Risk-Optimized Portfolios

Viacheslav Buchkov, Nikolai Kurlovich

June 25, 2025

Quantitative Asset Management & Systematic Investments
Dr. Gianluca De Nard & Dr. Patrick Walker

Abstract

De Nard and Kostovic (2025) present a novel approach to estimating the risk-optimized portfolio by learning the optimal shrinkage estimation, created by an "expert Oracle" in the supervised learning manner. Our paper shows that this Behavioral Cloning approach might be significantly improved by introducing Inverse Reinforcement Learning, where a policy is optimized to copy this "expert Oracle". Such an approach solves the issues of distributional shifts and cascading policy errors that are present in Behavior Cloning, resulting in suboptimal risk-optimized portfolios. Empirical experiments show that in fully out-of-sample construction our model improves on the existing results for linear shrinkage, providing a new data-driven approach to construct risk-optimized portfolios.

Keywords: Imitation learning; Inverse reinforcement learning; portfolio management reinforcement learning; risk optimization

Contents

List of Figures

List of Tables

1 Introduction

Accurate estimation of the covariance matrix of asset returns is central to modern portfolio optimization ever since Markowitz's seminal work introduced mean–variance analysis as a framework for balancing risk and return. However, in practical settings with large numbers of assets relative to sample size, the sample covariance matrix is notoriously ill-conditioned and yields unstable inverse estimates, often resulting in poor out-of-sample performance and extreme portfolio weights.

To mitigate these issues, a rich literature has developed shrinkage estimators that pull the sample covariance toward a structured target. Linear shrinkage combines the sample covariance with a simple target using an asymptotically optimal weight derived under an i.i.d. assumption. Nonlinear shrinkage further refines this idea by shrinking individual sample eigenvalues toward a central value. Extensions based on cross-validation have also been proposed to select shrinkage intensities without relying on asymptotics. More recently, De Nard et al. 2025 introduce a supervised "Behavioral Cloning" approach that uses an expert Oracle—whose monthly shrinkage choices minimize realized out-of-sample volatility—to train a regression model for predicting optimal shrinkage parameters.

In this paper, we show that moving from behavioral cloning as shown in De Nard et al. 2025 to Inverse Reinforcement Learning (IRL) yields materially better risk-optimized portfolios. While Behavioral Cloning fits a policy to past expert actions, it struggles under distribution shift and lacks an explicit notion of the underlying reward that generated those actions. IRL methods such as GAIL (Ho et al. 2016) and AIRL (Fu et al. 2017) recover a surrogate reward function consistent with expert behavior, then derive a policy that is robust to changing market conditions, providing superior performance.

Initial attempts to apply RL directly to covariance shrinkage demonstrate its feasibility but also reveal limitations. Matera et al. 2023 introduce a data-driven shrinkage estimator in which a policy gradient (PGA) and a recurrent PGA (RPGA) learn a single intensity parameter toward an identity target. While innovative, their framework is constrained to a one-dimensional action space and neglects richer structural priors or state-dependent multi-objective rewards. Similarly, Lu et al. 2024 extend RL to correlation matrices by incorporating a Text-Based Network (TBN) prior as a shrinkage target and optimize under a single exponential-utility reward. However, they restrict themselves to fixed TBN versus identity priors, do not explore alternative reward formulations, and their agent's policies remain prone to overfitting when market regimes shift.

More generally, reinforcement learning ideas have been applied across finance: Deng et al. 2016 use deep Q-networks for market-signal representation, Jiang et al. 2017 develop a model-free RL framework for portfolio allocation, and subsequent work has incorporated curiosity-driven exploration (Hirchoua et al. 2021) and multi-agent

RL (Shavandi et al. 2022) into trading and allocation strategies. This clearly shows that there a long record of attempts to utilize reinforcement learning for different applications in finance.
The remainder of this paper is organized as follows. Section 2 develops the theoretical framework for both the existing methods drawn from De Nard et al. 2025 and our newly proposed approaches. Section 4 describes the datasets, details the backtesting procedures and implementation, and presents all numerical results. Finally, Section 5 outlines avenues for future research, and Section 6 offers our concluding remarks.

2 Methodology

2.1 Linear Shrinkage

Ledoit and Wolf (Ledoit et al. 2003) propose estimating the population covariance matrix Σ by "shrinking" the sample covariance S toward a structured target matrix F via:

$$\tilde{\Sigma} = \delta F + (1 - \delta)S$$

where $\delta \in [0, 1]$ is the shrinkage intensity.
Under the i.i.d. assumption, an optimal δ can be derived such that it minimizes the expected quadratic loss function given by $\mathbb{E}[||\delta F + (1 - \delta) - \Sigma||_F^2]$. It turns out to be:

$$\hat{\delta} = \min\left\{1, \max\left\{\frac{\left(\sum_i \sum_j \frac{1}{T}\sum_t((r_{ij} - \bar{r}_{i\cdot})(r_{ji} - \bar{r}_{j\cdot}) - s_{ij})^2\right) - \hat{\rho}}{T\sum_i \sum_j (f_{ij} - s_{ij})^2}, 0\right\}\right\}$$

where $\hat{\rho}$ measures the covariance between the shrinkage target and the sample covariance matrix[1].
A crucial advantage of this estimator is that, since $F \succ 0$ and $\delta \geq 0$, $\widehat{\Sigma}(\hat{\delta})$ remains positive definite (and hence invertible) even when S itself is singular (e.g. when $N > T$).

2.2 Reinforcement Learning

As outlined in the original De Nard et al. 2025 paper, here we work in the 1-step Markov Decision Process (MDP) setting with $\mathcal{M} := \langle \mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R}, \zeta \rangle$, where $\mathcal{S}$ denotes the states (features that describe current market environment), $\mathcal{A}$ the set of actions (in our case, the shrinkage intensity), $\mathcal{P}$ the state transition probability matrix (in our case, independent of actions), $\mathcal{R}$ the reward function (in our case, the negative of 21-days-ahead realized portfolio standard deviation), and ζ the discount factor (in our case, equal to one for a one-period model).
We follow exactly the same RL formalism construction as in De Nard et al. 2025, meaning that our agent aims to maximize the negative of 21-days-ahead realized portfolio (excess returns) standard deviation by choosing an optimal action $\hat{\delta}^{RL-*}$at each round, i.e. each rebalancing period. We agree fully with the considerations of action-independent state transitions and 1-step optimization, which leads us to the *contextual bandits* setting, as outlined in De Nard et al. 2025:

[1]The exact analytical expression for ρ depends on the choice of the shrinkage target and therefore requires case-by-case analysis.

$$P(s_{t+1}|s_t, a_t) = P(s_{t+1}|s_t), \quad \forall \quad a_t \in \mathcal{A}$$

Moreover, one can see that while our overall goal is minimization of the long-term strategy standard deviation $\sigma_T(\{s_i\}_i^T, \{a_i\}_i^T)$, by the independence of state-visitation distribution d^{π_θ} from the taken actions a_i, one ends up with the following transformation:

$$\min_\theta \mathbb{E}_{a_i \sim \pi_\theta(\cdot|s_i)} \ \sigma_T^2(\{s_i\}_i^T, \{a_i\}_i^T) = \min_\theta \mathbb{E}_{a_i \sim \pi_\theta(\cdot|s_i)} \ \sum_i^T \sigma_i^2(s_i, a_i)$$

$$= \sum_i^T \min_\theta \mathbb{E}_{a_i \sim \pi_\theta^i(\cdot|s_i)} \ \sigma_i^2(s_i, a_i) = \sum_i^T \min_\theta \ \sigma_i^2(s_i, a_i^\theta) \tag{1}$$

where we used that expectation and the sum can be exchanged due to the same measure setting (same state-visitation distribution) and the portfolio constructions are independent from each other (at each rebalancing period we are choosing an optimal action, regardless of the past choices, implying that we can close all the positions in the portfolio and reconstruct it from scratch each time).

$$(1) \iff a_i^* = \arg\min_\theta \ \sigma_i^2(s_i, a_i^\theta) = \arg\min_\theta \ \sigma_i(s_i, a_i^\theta)$$

One should note that the future realized volatility is unknown, meaning that our optimization works with $\hat{\sigma}_i(s_i, a_i^\theta)$, an estimate of the future standard deviation. This discrepancy will prove to be crucial for the model performance and is addressed further in the Inverse RL setting, and is behind of one of the main reasons we are able to improve the De Nard et al. 2025 paper results.

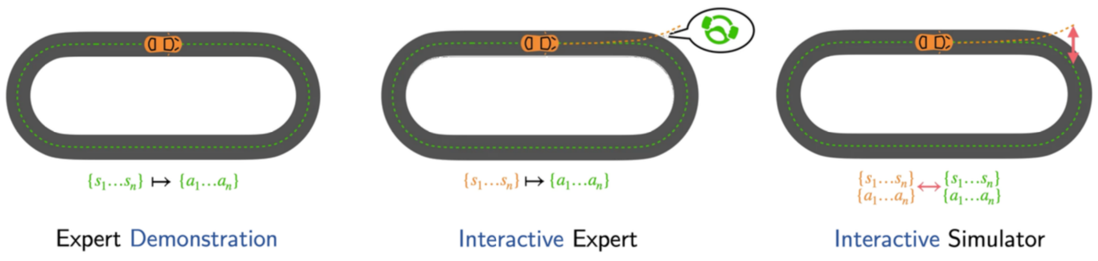
2.3 Imitation Learning

As argued in the previous section, we do not know the function that maps the chosen action to the future realized standard deviation, nor do we have the demonstrations that can be reliably enough used to optimize such a function under changing market conditions (see the Contextual Bandits Section on more details and construction of the experiment to show such a claim).

Another pitfall of the usual RL setting in this case is the reward design. As was mentioned before, our cost (reward) is (negative) 21-days-ahead realized standard deviation of the chosen portfolio. However, due to changing market environment and structural shifts, the distribution of such a cost (reward) is non-stationary through time. It makes the RL task severally more complex, as one cannot distinguish the optimal action from suboptimal one under unstable rewards. For instance, in one month the overall volatility might be elevated, meaning that an optimal action would

have a reward lower than a suboptimal action in the previous month. And as our state space is represented by the unique (market) states at each day, a usual RL approach can not be suitable for such a reward distribution.
Moreover, if one tries to scale the rewards to stabilize the distribution (as is done in Contextual Bandits Section), one ends up with still shifting scaled reward distribution, as for OOS construction the scaling can be done only using the past observed 21-day volatilities, which are not exactly representative for the future, especially under constantly changing market environment and unseen before crises (GFC, Covid Crisis etc.).
Therefore, as suggested in Equation (2.12) of De Nard et al. 2025, one might try to learn the optimal policy from the "expert demonstrations", i.e. some optimal demonstrations of **actions** without the need to connect those to some explicitly formulated reward function.
This approach is well-known in RL literature and called the Imitation Learning, where different settings use exactly the construction as described above. In our case we work in the setting of Expert Demonstrations, where we have a pre-determined set (past data, used for training) of optimal state-action pairs:

Figure 1: Categories of Imitation Learning. Source: Prof. Dr. Niao He Slides, TFoRL Spring 2025


$\{s_1 \ldots s_n\} \mapsto \{a_1 \ldots a_n\}$
Expert Demonstration
$\{s_1 \ldots s_n\} \mapsto \{a_1 \ldots a_n\}$
Interactive Expert
$\{s_1 \ldots s_n\}$ $\{a_1 \ldots a_n\}$ $\leftrightarrow$ $\{s_1 \ldots s_n\}$ $\{a_1 \ldots a_n\}$
Interactive Simulator

Let us first define the state-action pair construction and then comment on the De Nard et al. 2025 approach versus our improvement from Theoretical Foundations of Reinforcement Learning perspective.

2.4 Feature Set

For consistency and comparability with De Nard et al. 2025, we adopt exactly the same ten-feature vector throughout this paper. These features capture both cross-sectional market structure and temporal dynamics of returns and past portfolio decisions. Table 1 summarizes each feature and its interpretation.

Feature	Description
Average Correlation	Average pairwise sample correlation between stocks.
Average Volatility	Average one-year sample volatility of all individual stocks.
EWMA	Exponentially weighted moving average of previous-month returns (decay = 0.1) of the equally-weighted portfolio.
Lagged Optimal Action	Optimal action from the previous rebalancing.
Linear Shrinkage	Linear shrinkage intensity of Ledoit and Wolf (2004b).
Momentum	Fraction of days each stock had positive returns over the previous month, averaged across all stocks.
Rolling Optimal Action	One-year rolling average of the optimal action.
Rolling Optimal SD	One-year rolling standard deviation of the optimal actions.
Trace	Trace of the sample covariance matrix.
Universe Volatility	One-year sample volatility of the equally-weighted universe.

Table 1: Summary of features used.

2.5 ML Target Construction

We follow the methodology, proposed in De Nard et al. 2025. Namely, our shrinkage target $\hat{\delta}^{RL-*}$is attained as the minimizer of 21 days-ahead portfolio standard deviation. However, contrary to the original paper, our approach is based on continuous optimization, meaning that instead of a search over a pre-defined grid of values, we search within the domain of $[\delta_{min}^{RL-*}; \delta_{max}^{RL-*}]$, allowing the variable to take any value inside this interval. Even though there is no reason to have a difference in the portfolio performance with this method, which is supported by our robustness checks, we follow this approach for more generality, as it can be applied to another domains, as outlined in the Future Work section.

2.5.1 Black Box Optimization

The search for $\hat{\delta}^{RL-*}$represents a so-called "black-box optimization" problem, meaning that the search for optimal minimum standard deviation is an intractable function from the $\hat{\delta}^{RL-*}$. That said, we address the common solutions of such a problem, where we do not need any prior information on the $\sigma_{t:T}(\delta^{RL-*})$ behavior. Namely, we use the well-adapted model of GP-UCB (Gaussian Process - Upper Confidence Bound) (Lai et al. 1985).
The model works in the following setting:

- Given: domain of shrinkage values $D = \{\delta^{RL-*} \in [\delta_{min}^{RL-*}; \delta_{max}^{RL-*}]\}$, we obtain (potentially noisy) realized standard deviations $\sigma_{t:T} : D \to \mathbb{R}^+$;
- Task: Adaptively choose potential maximizers $\delta_1^{RL-*}, ..., \delta_S^{RL-*}$ and observe $\sigma_{t:T}$, where S is the number of iterations.

This approach corresponds exactly to the Bandit Optimization setting, meaning that we aim to minimize the cumulative regret over these K steps of optimization, which balances the precision of the solution and computational budget spent in our case.
To employ the efficient optimization we use the simple upper confidence bound algorithm, GP-UCB. At step s it samples the next potentially optimal point such that

$$\delta_s = \arg\max_{\delta \in D} \mu_{s-1}(\delta) + \beta_s^{1/2} K_{s-1}(\delta, \delta),$$

where $\mu_{k-1}(\delta)$ and $K_{k-1}(\delta, \delta)$ are corresponding mean and covariance matrix of a Gaussian Process with Kernel K. In our modeling, following the practical performance considerations, we use the Radial Basis Function Kernel (Gaussian Kernel), given by

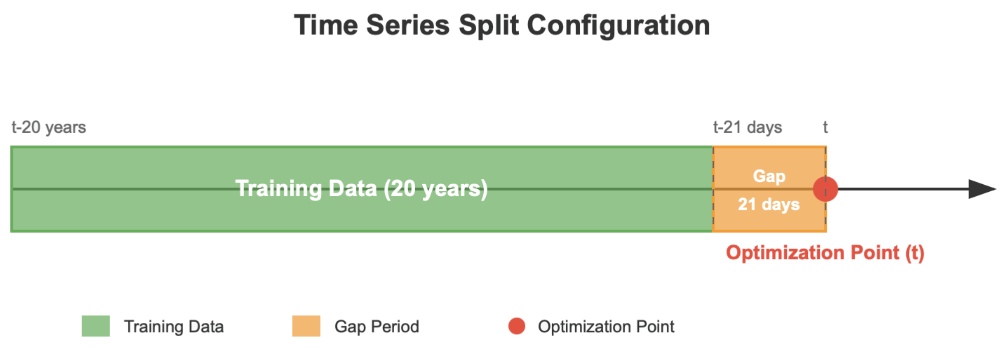
$$K(x, x') = \sigma_p^2 \exp - \frac{||x - x'||^2}{2h^2},$$

where x and x' are the two points within the domain, h is the bandwidth hyperparameter and σ_p^2 is the prior variance.
This simple algorithm allows to model the dependence of δ to the 21-days-ahead realized portfolio standard deviation without any additional prior assumptions on this complex function behavior. The GP-UCB iteration allows for sublinear asymptotical bound on the optimal standard deviation, meaning that the optimal gap decays sublinearly with each new iteration, producing $\max_s \sigma_{t:T}(\delta_s) \to \sigma_{t:T}(\delta^*)$ (Srinivas et al. 2010), which is exactly what we want to achieve.
Therefore, our **in-sample** optimization for target construction is exactly iterating over S steps with this GP-UCB algorithm, where for each day of interest within the whole sample we reinitialize a new Gaussian Process and make at most S steps for optimization, thus saving a lot of resources, compared to the Grid Search, and allowing for continuous δ modeling.
Moreover, we employ the checking of the obtained results with the well-known Golden Section Algorithm, which works in the Grid Search setting, but optimally chooses 3 points, within which one should search for the maximum. It is restricted to end after the same S iterations, and our experiments show that both algorithms indeed achieve optimum - on the random subset of points the solutions compared with the Brute Force solution.

2.5.2 Causal Window

An important aspect of the experiments construction is the application of the so-called “Causal Window“. Namely, as our target is constructed as optimal for upcoming 21 days, the actual last available point in out of sample construction is the one that is 21 days before the rebalancing days. The reason for that is the fact the for each optimal $\hat{\delta}^{RL-*}$that is in $[t - 21\ days; t]$, we need to know the realization of the next 21 days of trading, which is unavailable in the out-of-sample construction. Indeed, in practice we would have needed to witness the next days quotes to produce the optimal point calculation, i.e. to get $\hat{\delta}^{RL-*}_{t-1\ days}$, we need to know the realized returns of the portfolio, constructed at $t - 1\ days$ and held through $t + 20\ days$, which clearly contains a lookahead.
Therefore, to ensure fully OOS construction, we employ the 21 days gap, meaning that we take into model fitting for each rebalancing date t only datapoints that are within the interval $[t - 20\ years; t - 21\ days)$. The model that is used for prediction is trained on the “green“ interval of the following interval:

Figure 2: Causal Window Illustration



However, for each point within the “green“ interval we take the features that were available at the day of the portfolio construction for each of the 20 years - 21 days datapoints. All such points are obtained, using the look-behind data only for each datapoints, so it already ensures the OOS construction.
Finally, for the current rebalancing point (“red“), we use the features that are available at the most recent point t minus the trading lag of n days, except for the last optimal action, moving average and standard deviation of targets that are also using the causal window of 21 days.
The trading lag n is either zero for theoretical construction or $n > 0$ for more practical consideration to be protected against potential imperfection in OOS feature

construction. We test several lags and find that the model is robust to all reasonable lags up to 3 days, while the performance decays (as expected).
One may note that such a construction “throws away“ 21 datapoints from optimization. While it is necessary for the targets (and several target-based features), most of the market data-based features might still be useful for the model score improvement, which is a potential area of the future work - e.g., one may construct a time series / recurrent model for the prediction, which would take the 21 datapoints into account.

2.6 Behavioral Cloning

If we look at our task from the Imitation Learning perspective, one may instantly recognize the De Nard et al. 2025 approach (Equation 2.12 and 2.13) as the Behavioral Cloning. Namely, we use the Expert Demonstrations and learn the optimal policy by maximum likelihood fitting. Observing the (offline) demonstrations $\mathcal{D} = \{(s_i, a_i^*)\}_{i=1}^n$ under $a_i^* \sim \pi_E(s_i)$ expert policy, which is exactly the optimal shrinkage value, found by the previous section procedure.
One follows the regression setting procedure of fitting the MLE by Empirical Risk Minimization:

$$\min_\theta \mathbb{E}_{s\sim d^{\pi_E}, a\sim\pi_E(\cdot|s)} \left[\ell(\pi_\theta(\cdot|s), \pi_E(\cdot|s))\right]. \tag{2}$$

where d^{π_E} is the state visitation distribution under policy π_E. One might recognize in this exactly the Equation 2.12 of De Nard et al. 2025 under the L2-Loss (MSE) and state visitation distribution d^{π_E}, being independent from the policy chosen (as outlined in Reinforcement Learning Section).

$$\pi^*(s) = \arg\min_{\pi\in\Pi} \mathbb{E}_s \left[(\pi(s) - a^*(s))^2\right]$$

In order to learn a policy that mimics the oracle shrinkage intensities y_t^* computed on each date, the problem is cast as a standard supervised regression task over the feature vectors s_t.
As a small note, the task might also be viewed as the KL-Divergence minimization of our policy versus the expert policy:

$$\min_\theta \mathbb{E}_{s\sim d^{\pi_E}}[D_{KL}(\pi_E(\cdot|s)||\pi_\theta(\cdot|s))] = \mathbb{E}_{s\sim d^{\pi_E}, a\sim\pi_E(\cdot|s)} \left[\log\left(\frac{\pi_E(a|s)}{\pi_\theta(a|s)}\right)\right] \tag{2}$$

For this algorithm we can get the MLE guarantee, as outlined in Agarwal et al. 2021 as

$$J(\pi_E) - J(\hat\pi_{MLE}) \leq \frac{1}{(1-\gamma)^2}\mathcal{O}(\sqrt{\frac{\log(|\Pi|/\delta)}{|\mathcal{D}|}}) \tag{3}$$

, where $|\Pi|$ is the policy class, $|\mathcal{D}|$ is the length of the dataset and $J(\cdot)$ is the expected cumulative reward our our fitted policy.
Therefore, such an approach works reasonably well under static environments and high number of representative expert demonstrations. Let us discuss the models used for the Empirical Risk Minimization.

2.6.1 Elastic Net

First of all, we replicate the approach of De Nard et al. 2025 exactly by using the Elastic Net, trained on the 20 most recent years of optimal actions, except the causal window of 21 days.
The mapping $s_t \to y_t^*$ is modeled by a linear predictor with combined ℓ_1/ℓ_2 regularization. Concretely, letting β_0 be the intercept and $\beta \in \mathbb{R}^{d+1}$ be the coefficients of the d-dimensional feature vector s_t, Elastic Net solves

$$\hat{\beta} = \arg\min_{\beta_0,\beta} \frac{1}{T}\sum_{t=1}^{T}\big(y_t^* \;-\; \beta_0 \;-\; s_t^\top\beta\big)^2 \;+\; \lambda\Big(\alpha\|\beta\|_1 \;+\; \tfrac{1-\alpha}{2}\|\beta\|_2^2\Big),$$

where

- $\lambda > 0$ controls the overall regularization strength;
- $\alpha \in [0, 1]$ controls the balance between the two types of regularization – ℓ_1/ℓ_2.

The Elastic Net model is retrained every 21 trading days. Within each window:

1. A grid search is performed over the parameters $\lambda \in \{0.5, 1, 1.5, 2, 5\}$ and $\alpha \in \{0.1, 0.25, 0.5, 0.75, 0.9\}$;
2. For each (λ, α) candidate, out-of-sample performance is evaluated via time-series cross-validation (utilizing five folds respecting temporal order);
3. Contrary to the paper, we do not optimize over `max_iter` and `tol`, as these parameters control only the convergence condition, which is always satisfied under well-defined problem.

Such a construction differs slightly from the original De Nard et al. 2025 paper, as it ensures out-of-sample construction by the hyperparameter tuning for each window separately. While De Nard et al. 2025 have tested the robustness of their in-sample tuning, our choice of modeling is aimed to replicate the fully practically relevant approach and targets OOS construction in all of the details.

2.6.2 Random Forest

To capture potentially nonlinear dependencies between the feature vector s_t and the expert shrinkage intensity y_t^*, a Random Forest regressor is implemented. It is a non-parametric, ensemble learning method that builds a collection of decision trees on bootstrap samples of the data and averages their predictions to reduce variance and guard against overfitting. Each tree is grown by recursively splitting the feature space so as to minimize the mean-squared error in its leaves, and at each split a random subset of the features is considered.
For hyperparameters, the number of trees is set equal to 30, whereas the maximum depth of each tree is 10. We do not optimize the hyperparameters neither in-sample nor out-of-sample, as this level of parameters is driven by the balance of computational budget (former) and overfitting control (latter). While one might find a better optimization scheme, in our experiments we find robustness of orders of the results to the set hyperparameters. Moreover, only these hyperparameters are relevant for usual Random Forest construction, as the other ones (like maximum number of objects in leafs etc.) can be mapped into the choice of the maximum depth and work similarly to control the overfitting.

2.6.3 Gaussian Process

Gaussian Process Regression (GPR) is a Bayesian non-parametric method that models the mapping $s_t \mapsto y_t^*$ by placing a Gaussian prior over functions and updating it to a posterior after observing the data. Using a kernel $k(s, s')$, GPR yields a predictive mean

$$\hat{\delta}(s) \;=\; k(s,S)\left[K(S,S)+\sigma_n^2 I\right]^{-1} y^*,$$

which we take as the shrinkage estimate at each rebalancing date. This approach flexibly captures complex, nonlinear relationships and provides a natural uncertainty quantification through its posterior variance.
We consider the GP Regression as an important baseline for our study, as by placing a hyperprior over hyperparameters, it allows for tuning those on the training dataset. While still ensuring OOS construction, such an approach is alternative for Time Series Cross Validation and can be seen as another way to estimate the optimal hyperparameters (by estimating the prior standard deviation on the CV folds).
In our case, we employ the hyperprior for tuning, allowing the model to restart the optimizer 3 times at each refitting date (rebalancing date). The number of restarts is chosen for the computational speed, while we still find robustness of this hyperparameter in our experiments.
We work with the 2 types of kernels:

- DotProduct(), also known as "Linear Kernel". The GPR under this kernel is exactly a Ridge Regression, meaning that such a baseline should produce the results close to the ElasticNet from De Nard et al. 2025, but potentially gain a robustness of hyperparameters due to hyperprior.

- RBF() (as outlined in Black Box Optimization Subsection). This kernel represents a complex non-linear relationships and is aimed to test the increase of complexity on the modelling results. One may note that an infinitely wide Neural Net (outlined in the next section) can be viewed as a Gaussian Process under some (potentially complex) kernel (Jacot et al. 2018). Moreover, any complex smooth function on a compact set may be represented by the sum of kernels ("Representer Theorem", Schölkopf et al. 2001). Therefore, such a GPR represents quite a strong baseline to study the virtue of complexity in this application.

2.6.4 Deep Learning

As our target approach on Inverse RL by construction requires the Neural Net for Reward function approximation and Policy modeling, we need to include the Deep Learning models as a baseline to test, whether the improvement comes from the IRL approach or just a more expressive predictor.
Deep learning covers a family of neural network architectures; here we adopt a simple multi-layer perceptron (MLP) to learn the mapping $s_t \mapsto y_t^*$. Here we use a multi-layer perceptron (MLP) with two hidden layers of size 32 and ReLU activations, followed by a linear output. Formally:

$$h^{(1)} = \max\big(0,\, W^{(1)} s_t + b^{(1)}\big), \quad h^{(2)} = \max\big(0,\, W^{(2)} h^{(1)} + b^{(2)}\big), \quad \hat{\delta}_t = W^{(3)} h^{(2)} + b^{(3)}.$$

We train by minimizing mean-squared error via stochastic gradient descent with a cosine-annealing schedule.
We fix learning rate = 10^{-3}, hidden size = 32, layers = 2, epochs = 10, batch size = 64, weight decay = 0.0.
The choice of the NN architecture copies exactly the underlying architecture of IRL approaches. Therefore it represents a sutable baseline for testing of the hypothesis of more complex predictor.

2.7 Contextual Bandits

As formulated by the Reinforcement Learning Section (following on the results from De Nard et al. 2025), the problem at hand can be viewed as a *contextual bandit* setting. However, as outlined in Imitation Learning Section, direct Contextual Bandit optimization is not expected to yield a good results in this case because of the reward design issues.

However, to implement a suitable baseline, we test the usual Contextual Bandits setting. Therefore, we scale the negative future realized volatility by the past observed minimum and maximum of **past optimal** volatilities at each point of time to attain

$$\tilde{r}_t = \frac{-\sigma_{t:T}(\hat{\delta}^{RL-*}) - (-\sigma_{max}(\delta^{RL-*}_{t-\Delta}))}{-\sigma_{min}(\delta^{RL-*}_{t-\Delta}) - (-\sigma_{max}(\delta^{RL-*}_{t-\Delta}))} \in [0;1]$$

.
We also test the approach, where we construct the reward as $r_t = \frac{1}{\sigma_{t:T}(\hat{\delta}^{RL-*})}$. The results seem to be close to the original setting, but the reward distribution is more concentrated in this case, meaning that the former approach should be preferred.
In the bandit optimization setting, at each rebalancing date t the agent observes the current state (feature vector) s_t, selects a shrinkage intensity a_t, and immediately receives a reward $r_t = -\sigma_{t:T}(s_t, a_t)$ (the negative 21-day realized volatility). There are no long-term state transitions, so the objective reduces to minimizing cumulative regret

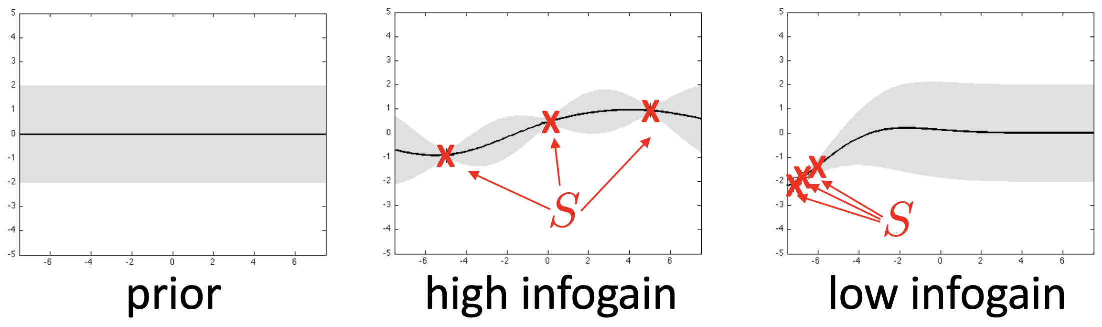
$$R_T = \sum_{t=1}^{T} \big[r^*(s_t) - r_t\big], \quad r^*(s) = \max_a \mathbb{E}[r(s,a)].$$

Under a smoothness assumption on the reward function $f(s,a)$, we model f with a Gaussian Process and use a GP-UCB rule to balance exploration and exploitation. At step t, given the posterior mean $\mu_{t-1}(s_t,a)$ and standard deviation $\sigma_{t-1}(s_t,a)$, the agent chooses

$$a_t = \arg\max_{a\in A} \big[\mu_{t-1}(s_t,a) \;+\; \sqrt{\beta_t}\,\sigma_{t-1}(s_t,a)\big],$$

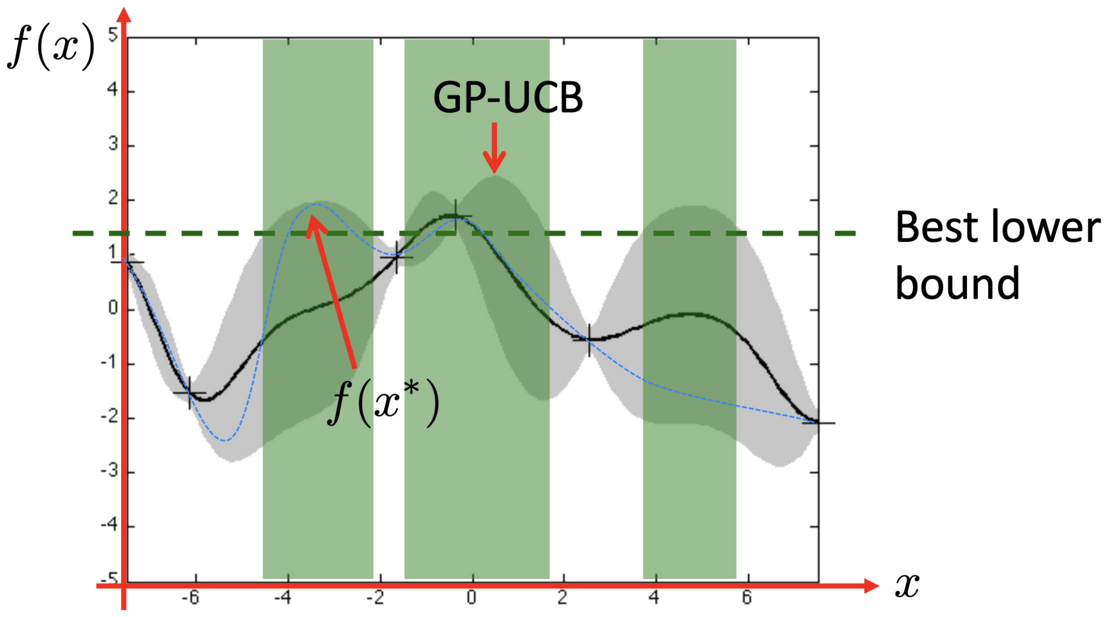
where β_t grows logarithmically in t to guarantee sublinear regret.
The search for the lowest uncertainty can be visualized by the following picture:

Figure 3: Information Gain. Source: Prof. Dr. Andreas Krause, PAI Fall 2024



The idea of the GP-UCB is to balance the exploration and exploitation by the optimal choice of the search region. Thus, we want to simultaneously maximize the function (i.e. search in the region of the maximum mean value of our f estimation) and minimize the uncertainty (as there might potentially be a true maximum, which we haven't captured by the current function yet, thus we should sample there to obtain new information). It can be viewed in the following illustration, where the "green" regions are the ones, where we should search for the optimum, according to GP-UCB:

Figure 4: GP-UCB. Source: Prof. Dr. Andreas Krause, PAI Fall 2024



The natural extension of this approach is given by Krause et al. 2011 and is called Contextual GP-UCB. The approach just adds the "context" (in our case, the market state representing features) as a new input variable to model context-dependent probabilities for each option (discrete case) / representative function f (continuous case). Namely, we model K as the product kernel of k_S and k_A, s.t. $K = k_S \times k_A$, where each separate kernel is dedicated to modeling context and action mapping into optimal reward space, respectively.
This model also produces the sublinear regret (Krause et al. 2011). Moreover, as our decision domain is compact and convex ($\delta \in [0; 1]$), under assumption of f being P-a.s. L-smooth (or weaker probability assumption as in Krause et al. 2011), we obtain the optimal choice of β_t to ensure the sublinear regret.
We implement both a discrete Kernel-UCB baseline (grid over δ) and a continuous

CGP-UCB variant (joint GP on (s, a)). Hyperparameters are set as: RBF kernel (with prior variance and lengthscale optimized under hyperprior setting) and confidence parameter $\delta = 0.1$ (so $\beta_t = 2\ln(|A|\, t^2\pi^2/(6\delta))$).
What is the difference of this approach, compared to the Imitation Learning? Here we follow more classical setup of RL, where we send the chosen action into the environment, balancing exploration and exploitation, and trying to find the optimum. Such an approach might be motivated by at least two considerations:

1. What if our optimized $\hat{\delta}^{RL-*}$is not easily attainable (represents the sharply peaked likelihood function in one points), while there is another slightly suboptimal region, but with a more plateau-like surface of likelihood? Then one might be better off in aiming to find this less optimal region, as while it increases the bias of our model, it would also decrease a variance, resulting in more robust out-of-sample prediction. Therefore, to attain such a solution we would use the model to drive the exploration of the state progression, but not the developed before target.

2. Moreover, as our current modeling is two-stage: first generation of targets and then only optimization for OOS prediction, the CGP-UCB approach aims to do both “in one go“, potentially using lower computational budget.

Finally, our progress works the in the following way:

1. We sample a “context“ in Time Series manner (i.e. we operate sequentially over each day in the training data).

2. The model selects an optimal exploration-exploitation action for this context.

3. We evaluate this action, returning a reward, and switch to the next state, meaning the next trading day.

4. When we reach a rebalancing date, we “switch-off“ the exploration mode (by setting $\beta_t = 0$) and return **pure exploitation** action.

2.8 Inverse Reinforcement Learning

2.8.1 Improvement Vs Behavioral Cloning

Our proposed approach is based on the Inverse Reinforcement Learning approach. Even though the Behavioral Cloning represents the simple and easy-to-implement approach, it was proven to be suboptimal in case of:

1. Long-Term Planning. As the approach works only for one single point of policy optimization, it suffers from the lack of awareness that the state-action

pair comes from an MDP, thus being unable to provide the H-step ahead decision-making policy.

2. Cascading Errors. As we act under the policy that is optimal for one step only, the Behavioral Cloning tends to accumulate the errors over the H horizon, resulting in suboptimal reward, compared to the Expert policy. This error grows with the horizon H.

3. Distribution Shift. At the core of the Behavioral Cloning is the assumption that each Expert policy demonstration comes from the same state visitation distribution. It means that the policy, which we aim to learn, is robust to the changing environment. In other words, **it suffers from the case of mismatching training and testing distributions**, known as *distribution shift* (Abbeel et al. 2004, Ziebart et al. 2008)

While the pitfalls 1 and 2 are irrelevant in our problem formulation, as our horizon $H = 1$ for one-step optimization and our actions do not affect state-visitation distribution (De Nard et al. 2025), the pitfall 3 is crucial in our case. Indeed, as the financial markets are constantly changing and the agents behave quite differently throughout the time, the optimal decision mapping might also change, meaning that there is the issue of *distribution shift.*
In De Nard et al. 2025 formulation, the argument can be seen in Equation (2.11). Indeed, while our optimal action is a maximizer of such a future realized volatility, as the correspondence between the optimal features and the optimal action might change over time, our policy should account for this shift during optimization, because we do not know the optimal $\hat{\sigma}(\cdot)$ in advance. In other words, **we should embed in our policy the "regularization" to know that past optimal feature $\rightarrow$ optimal action correspondence might not be optimal through time**, which can be done by sequentially modeling the policy to follow expert, but additionally learning the reward function that led to such an expert in our environment. And this is exactly what is done in the Inverse Reinforcement Learning approach.
Moreover, the virtue of IRL that is important in financial applications is the fact that it is able to recover off-distribution states from the training data. While Behavioral Cloning works solely with seen pairs of state-actions, and all of it generalization ability comes solely from the Risk Minimizer mapping of features to optimal action. However, as IRL is aimed to recover the reward - in our case, the order of shrinkage optimality -, it might be expected to be able to generalize outside of seen patterns of market behavior, which is also a known result in IRL literature.
The issue with the *distributional shift* can be clearly visible from the KL-Divergence

formulation:

$$\mathbb{E}_{s\sim d^{\pi_E}, a\sim\pi_E(\cdot|s)}\left[\log\left(\frac{\pi_E(a|s)}{\pi_\theta(a|s)}\right)\right]$$

It means that while we should aim to approximate our **policy reward** to be close to an expert, Behavioral Cloning achieves **policy behavior** to be close to it. Meaning that it has no control over how much the error of diverging costs in terms of reward, which is crucial in our case, as in some corner market conditions one may produce suboptimal results for a nearly KL-close policy.

Inverse Reinforcement Learning approach recovers the reward function and nearly optimal policy from the given demonstrations. We assume that the true reward function is unknown - i.e., we do not know how to represent the shrinkage optimality in terms of stable and comparable score between different days. However, we have the demonstrations of optimal policy π_E of the MDP.

Task: Find a reward function $r(\cdot,\cdot) : \mathcal{S}\times\mathcal{A}\rightarrow[-1,1]$ that explains the expert behavior, such that

$$\mathbb{E}\left[\sum_{t=0}^{\infty}\gamma^t r(s_t,a_t)\mid s_0\sim\mu,\pi_{\mathbb{E}}\right]\geq\mathbb{E}\left[\sum_{t=0}^{\infty}\gamma^t r(s_t,a_t)\mid s_0\sim\mu,\pi\right],\quad\forall\pi\in\Pi. \tag{4}$$

An important assumption for the IRL approach is that the Expert policy is **the best that one can do in the environment**. While such an assumption might be tricky in many usual IRL settings, causing issues in optimization, it is exactly satisfied in our case by construction of the most optimal target, which is used for demonstration. It is exactly the virtue that allows to achieve the superior results, compared to Behavioral Cloning - we can learn from the demonstration of exactly the optimality bound.

Here is the main overview of our methodology:

- We supply the model with the Expert policy demonstrations in the same manner with the Behavioral Cloning.
- Additionally, we allow the model to interact with the (training sample) environment by assessing the actions, i.e. receiving the negative realized portfolio standard deviation for chosen shrinkage levels. It is done in the same manner as the CGP-UCB, meaning that one can assess the action only once for each trading day. While in practice we could allow multiple interactions in the backtest, to keep the comparison fair we use this approach.
- We train the model on the train set only. I.e., after we start the backtest for Empirical Results, we do not retrain the model again. Thus, the model sees significantly less data (only half of the dataset) for training, and has no access

to new incoming points, contrary to the Behavioral Cloning baselines. It is done specifically to show the power of outperformance of such an approach and to guarantee absence of any look-ahead biases, as the model does not have any possibility to see any points beyond the first rebalancing date.

2.8.2 GAIL

First of the models that we use is Generative Adversarial Imitation Learning (GAIL, Ho et al. 2016).

- GAIL [6] aims to solve the **min-max game** for learning the policy given an expert policy π_E.

$$\min_{\theta} \max_{\phi} \quad \mathbb{E}_{s,a\sim\lambda^{\pi_\theta}}[\log(C_\phi(s,a))] + \mathbb{E}_{s,a\sim\lambda^{\pi_E}}[\log(1 - C_\phi(s,a))] - c \cdot H(\pi_\theta). \tag{5}$$

- We assume a differentiable parametrized policy π_θ (e.g., a Neural Net).
- The discriminator tries to separate the data generated by the learned policy from expert data.
- When $c = 0$, this is equivalent to minimize **the Jensen-Shannon divergence** between the state-action distributions between the expert policy and the learned policy.
- The min-max problem can be solved via efficient primal-dual methods by finding a conjugate function and transforming the problem into GAN-like loss.
- This model has seen many improvements from Jensen-Shannon divergence to the general f-divergence, however, we focus on the basic setup in order to produce a robust solution from widely accepted application.

Moreover, GAIL achieves more optimal bounds than Behavioral Cloning as can be seen by comparison with (3). The following Theorem shows, why theoretically our approach should yield more optimal results, compared to the Behavioral Cloning approach, even if the latter is done under suitable Risk Minimizer.
Theorem (Error Bounds of Imitation Gap) (Xu et al. 2020)
Assume $|r(s,a)| \leq 1, \forall s, a$. Suppose GAIL returns an imitated policy π such that

$$D_f(\lambda^{\pi_E}, \lambda^{\pi}) \leq \epsilon,$$

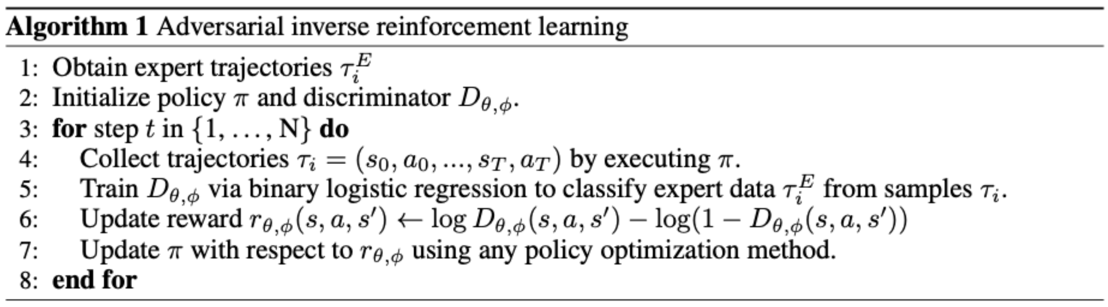
then we have

$$J(\pi_E) - J(\pi) \leq \mathcal{O}\left(\frac{1}{1-\gamma}\sqrt{\epsilon}\right)$$

2.8.3 AIRL

The final target approach that we employ is Adversarial Inverse Reinforcement Learning (AIRL, Fu et al. 2017). The approach of this model is quite similar to GAIL, however, the main distinct feature that is crucial for our analysis is the main result of AIRL paper. Authors achieve an algorithm that is **able to generalize well under significant variation in the environment seen in training data, allowing the AIRL model to recover reward function (and thus the nearly-optimal policy) that is robust to changes in dynamics** (Fu et al. 2017).
The model is trained in two stages. First the discriminator learns to "disentangle" the rewards of expert and rewards of the learned policy by simple binary classifier. And then we apply the same logic of the reward function update as in GAIL, following by the usual RL step for optimization of the optimal policy.
The algorithm is presented below:

Figure 5: AIRL. Source: Fu et al. 2017



Algorithm 1 Adversarial inverse reinforcement learning

```
1: Obtain expert trajectories $\tau_i^E$
2: Initialize policy $\pi$ and discriminator $D_{\theta,\phi}$.
3: for step $t$ in $\{1, \dots, N\}$ do
4:     Collect trajectories $\tau_i = (s_0, a_0, ..., s_T, a_T)$ by executing $\pi$.
5:     Train $D_{\theta,\phi}$ via binary logistic regression to classify expert data $\tau_i^E$ from samples $\tau_i$.
6:     Update reward $r_{\theta,\phi}(s, a, s') \leftarrow \log D_{\theta,\phi}(s, a, s') - \log(1 - D_{\theta,\phi}(s, a, s'))$
7:     Update $\pi$ with respect to $r_{\theta,\phi}$ using any policy optimization method.
8: end for
```

Therefore, as shown by the authors, the approach works well at highly varying environment because of more robust reward modeling, which is exactly the virtue that is needed in financial markets setting.

3 Baselines

3.1 Factor Models

We implement a static factor-model covariance estimator. Let $B_t \in \mathbb{R}^{N\times K}$ be the matrix of factor exposures—estimated via OLS on past excess returns—$f_t \in \mathbb{R}^K$ the corresponding factor returns, and $\varepsilon_t \in \mathbb{R}^N$ the idiosyncratic residuals. The covariance estimate is then

$$\Sigma_t \;=\; B_t \,\mathrm{Cov}(f)\, B_t^\top \;+\; \mathrm{f}\big(\mathrm{Var}(\varepsilon_t)\big),$$

where $\mathrm{Cov}(f)$ is the sample covariance of the K factors (we utilize low-risk, momentum, quality, size, value, and SPX) and $\mathrm{f}\big(\mathrm{Var}(\varepsilon_t)\big)$ - some transformation of the residual covariance[2].

3.2 Asymptotic Shrinkage

We implement the analytic linear shrinkage intensity δ from Ledoit et al. 2003 as mentioned in section 2.1. Recall that δ^* minimizes

$$\mathbb{E}\big\|\delta F + (1-\delta)S_t - \Sigma\big\|_F^2.$$

Moreover, we apply the Quadratic Inverse Shrinkage (QIS) approach from Ledoit et al. 2020 as a stronger baseline that shrinks the eigenvalues with different intensities. Even though our work is focused on the data-driven linear shrinkage improvement, we implement the framework that allows for QIS-based shrinkage testing.

3.3 Moving Average Shrinkage

This baseline smooths the oracle shrinkage intensities $\{\delta_i^*\}$ by a simple rolling average over the past $W = 252$ trading days (one year):

$$\delta_t = \frac{1}{W}\sum_{i=t-W}^{t-1} \delta_i^*.$$

It provides simple temporal regularization of the expert's choices without further learning.

[2]In this work we use Diagonal, $PCA(k=3)$ and QIS transformations

3.4 Last Optimal Shrinkage

The "last-action" rule sets the current shrinkage equal to the previous period's oracle value:

$$\delta_t = \delta^*_{t-1}.$$

This "last-action" rule is effectively a martingale baseline: if no additional information is predictive, the best guess for today is yesterday's value. By comparing our learning models against this persistence strategy, we can assess whether the feature set truly adds forecasting power beyond simple inertia.

4 Empirical Analysis

4.1 Data and Portfolio Construction Details

Following the approach in De Nard et al. 2025, we use daily stock returns from Center for Research in Security Prices (CRSP) starting on January 1, 1980 through July 31, 2024 (thus, expanding the original paper's set by two years). We filter stocks by being listed at NYSE, AMEX, and NASDAQ (CRSP exchange codes 1, 2, 3). Additionally, we use a subset of only ordinary common shares (CRSP shares codes 10 and 11).

In the same way as it is done in the original paper, we allocate all data before 12/18/2000 for initial training (or full training in case of IRL approaches), and use the remaining 5,932 days (283 months) for out-of-sample evaluation. As was emphasized throughout the paper, we ensure fully out-of-sample construction on all of the stages of the modelling, aiming to approach the fully practically implementable construction.

We rebalance the portfolio monthly, choosing month end date as the weights calculation date. If month end is a weekend or non-trading day, we will trade on the next available trading day. The weights calculation logic at the rebalancing date uses all of the past available data up to $t - n$, where n is the corresponding trading lag. In most of the models we use 1 day lag to additionally control the look-ahead bias. We test the lag robustness of the approach in Practical Implementation Section. The covariance matrix is estimated by 1 year of past data (trimming the set by the $t - 1$ *year* date).

The investable universe of $N \in \{30, 50, 100, 225, 500\}$ portfolio sizes is obtained in the same manner as in the paper - by filtering largest stocks (by market capitalization) and taking the top N. However, contrary to the original paper, we restrict the look-ahead of 'future' return for universe selection, and construct fully out-of-sample performance by considering stocks that have the 1 day ahead return present. As 1 day ahead return might be absent only due to non-existing today's price (know at the time of rebalancing) or next trading day price (the presence of which is almost always known, as any trading halts or delistings are published at least 1 day before). If the stock becomes absent at some point during our 1 month holding period, we assume that we would have known about delisting 1 day before the first NaN return and would have sold the stock from the portfolio. This is a reasonable assumption, as in the case of bankruptcy or other distressed scenarios, we would have received the whole amount of adverse returns, thus avoiding look-ahead bias from excluding such a stock. If delisting happens inside 1 month interval, we assume that this position brings us 0% return every day after the last day it was trading (which could be a risk-free rate, but for simplicity we use the more definite procedure).

Finally, for each rebalancing date we train the model on the past 20 years for each

rebalancing point. 2nd through final rebalancing is done by “throwing away“ past 1 month of observation and including 1 month of new data (adjusted by causal window from 2.5.2, where needed). We test different sizes of training data windows and find robustness of our model performance ranking to all of these sizes.
The performance measures are implemented exactly as in De Nard et al. 2025 with expanded set of measures for Final Strategy analysis in **??**.

4.2 Global Minimum Variance

Following the original paper, we employ the global minimum variance (GMV) portfolio as our target optimization for the paper. As De Nard et al. 2025 results show that outperformance holds for long-only case too, we demonstrate and compare baselines for the whole universe on GMV only. However, our experiments on long-only S&P constituents show that AIRL and GAIL approaches indeed produce an improved performance there too.
In case of GMV - in absence of short-selling constraints - we use the FOC-based solution of Quadratic Programming under linear budget constraints:

$$\hat{w} = \hat{\Sigma}^{-1}A^T(A\hat{\Sigma}^{-1}A^T)^{-1}b$$

where $\hat{\Sigma}$ is estimated covariance matrix and budget constraint is given by $Aw = b$
As outlined in the original paper, in long-only case we lose tractability and need to apply the numerical solver. We use the *CVXPY* package in python through Optimization module, provided by Dr. Cyril Bachelard in UZH Spring 2025 course “Quantitative Portfolio Management with Python“.
We include Equally-Weighted (EW) benchmark for comparison. In the interest of space and lack of new information added to original paper, we do not use Value-Weighted benchmark due to visible outperformance of all of the approached in De Nard et al. 2025.

4.3 Main Results

We can summarize our findings in the following outline:

- We confirm outperformance of De Nard et al. 2025 results to all of non data-driven (not shrinkage-predicted) approaches.
- More strong learner, Random Forest, outperforms De Nard et al. 2025 results across all of the experiments by SD and Sharpe due to better treatment of extreme shrinkages.
- Contextual Bandits approach shows worse results than the De Nard et al. 2025 paper.

- **Inverse RL achieves significantly better results in fully out-of-sample construction in both SD and SR, compared to any other model, including De Nard et al. 2025 approach**.

Table 2: GMV, (12/2000-07/2024) before TC with monthly rebalancing

	L	F-QIS	DNK	RF	GAIL	AIRL	EW
				SD			
$N = 30$	14.79	14.92	14.47	14.25	14.94	14.09	20.55
$N = 50$	14.58	14.69	14.17	14.06	14.05	14.05	20.68
$N = 100$	14.10	14.13	13.52	13.34	13.16	13.21	21.43
$N = 225$	13.10	13.00	12.44	12.21	12.35	12.43	21.73
$N = 500$	12.30	12.04	11.77	11.61	11.41	11.46	22.66
				SR			
$N = 30$	0.71	0.72	0.77	0.90	0.90	0.94	0.39
$N = 50$	0.73	0.73	0.77	0.86	0.93	0.94	0.40
$N = 100$	0.61	0.60	0.72	0.84	0.82	0.88	0.37
$N = 225$	0.68	0.70	0.78	0.89	0.90	0.93	0.35
$N = 500$	0.69	0.58	0.78	0.82	0.90	0.90	0.37

(a) Realized Standard Deviation (SD) and Sharpe Ratio (SR)

L: Linear Lediot-Wolf Asymptotic Shrinkage, F-QIS: Factor Model with QIS Residual Cov Shrinkage, DNK: De Nard et al. 2025 approach, RF: Random Forest.

As expected from theoretical results, the Imitation Learning approach produce severely improved results, with Inverse RL improving De Nard et al. 2025 Behavioral Cloning approaches and other strong baselines. We can note that AIRL approach produces quite robust outperformance versus GAIL across almost all universes. Moreover, these two approaches together improve significantly both SD (our target metric) and SR (which is of secondary importance), compared to any other considered baseline. Given that Inverse RL was trained only on data before 12/18/2000, **not adjusting** for new incoming data after 1st rebalancing through end, the results look even more stronger.
Furthermore, we present the mean, mean absolute, and mean squared deviations of the oracle intensities. We can see the clear pattern, outlined in De Nard et al. 2025,

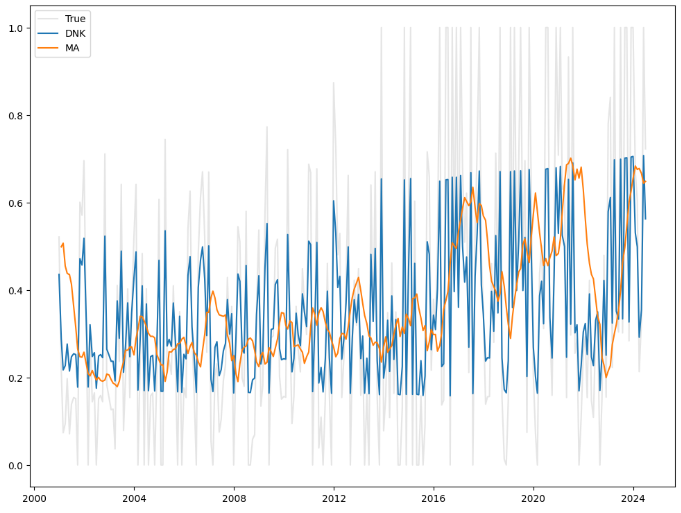
of data-drive shrinkage estimation approaches demonstrating substantially lower errors. The patter is continued by stronger Risk Minimizers and also is clearly seen in Inverse RL approach, which does not aim to learn the optimal shrinkage directly. We see that all of the considered below estimators produce unbiased errors, which is indicated by close to zero mean error.

Table 3: GMV Optimal Shrinkage Error, (12/2000-07/2024)

	DNK	RF	AIRL
	Mean Error		
$N = 30$	0.01	-0.01	0.02
$N = 50$	0.01	-0.01	-0.01
$N = 100$	0.01	0.00	-0.01
$N = 225$	0.01	0.00	-0.02
$N = 500$	0.00	0.00	0.01
	Mean Absolute Error		
$N = 30$	0.13	0.05	0.06
$N = 50$	0.13	0.05	0.06
$N = 100$	0.12	0.04	0.05
$N = 225$	0.11	0.04	0.06
$N = 500$	0.13	0.07	0.10
	Mean Squared Error		
$N = 30$	0.03	0.01	0.01
$N = 50$	0.02	0.01	0.01
$N = 100$	0.02	0.01	0.01
$N = 225$	0.02	0.00	0.01
$N = 500$	0.02	0.01	0.02

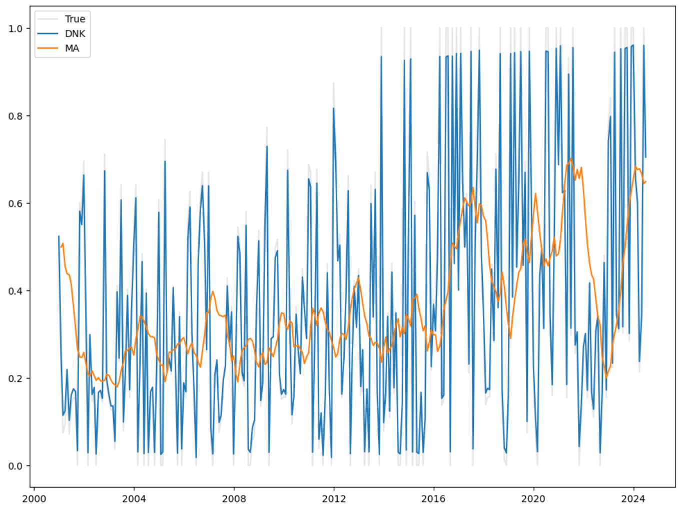
By further examining the learning error patterns, we first address the original Elastic Net approach. We confirm the results of the De Nard et al. 2025 paper that outlines suggested approach to show significantly more data-dependent approach (in “blue“), as opposed to moving average (in “orange“).

Figure 6: De Nard et al. 2025 Policy Path, $N = 100$



However, one can note that such a path appears to be quite restrictive - the predicted shrinkage turns out to be far away from the true Oracle target (in “gray“). The reason for that might be an excessive regularization. Even though we tune out-of-sample optimal shrinkage parameters, as they are always non-zero, it induces restriction on the number of non-zero regression coefficients (LASSO) and the size of the regression coefficients (LASSO and Ridge). Therefore, if one drops such a restriction by fitting a simple OLS regression without any regularization, we end up with a more pronounced path of predicted shrinkage.

Figure 7: OLS Policy Path, $N = 100$



But, unfortunately, such a simple estimator is prone to overfitting and unnessessary extrapolation. Therefore, we apply Random Forest as a robust way to optimize with the presence of sharp peaks and outliers. Due to dissecting the training data into different subspaces, we end up with the ability to predict “outliers“ quite efficiently, as in their respective leaf those are not outliers anymore. Indeed, we can see that Random Forest leads to an improved performance and protection against overfitting.

Figure 8: Random Forest Policy Path, $N = 100$

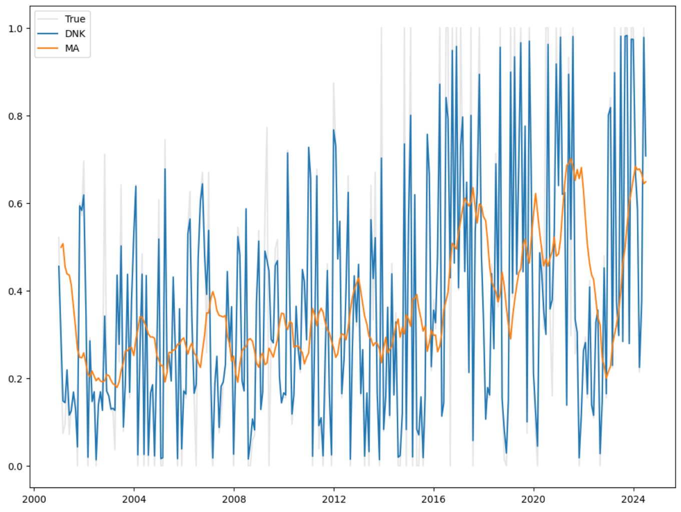
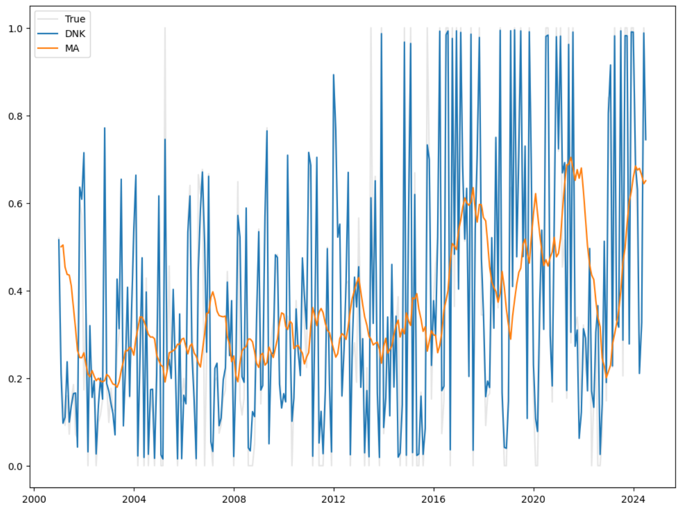


Figure 9: Inverse RL Policy Path, $N = 100$



Finally, our target approach is able to capture the extremes even more efficiently, producing the confirmation of robust policy optimization under changing environment dynamics (Fu et al. 2017). We can see that in most of the cases we are able to recover exactly the optimal shrinkage intensity up to 1 digit of rounding.

4.4 Baselines Analysis

In this section we analyze our suggested approach versus the chosen set of baselines. First of all, we analyze the performance, compared to the Factor Model without any data-driven shrinkage learning. As this approach aims to decompose the covariance matrix into the sum of low-rank and sparse matrices, we test the hypothesis of the shrinkage being already contained in the risk premia that drive the market.

Table 4: Factor Model Analysis

	F-Diag All	F-Diag Market	F-PCA	F-QIS	AIRL
			SD		
$N = 30$	15.10	16.30	15.46	14.92	14.09
$N = 50$	14.77	16.40	14.85	14.69	14.05
$N = 100$	14.43	16.59	14.09	14.13	13.21
$N = 225$	13.68	16.67	12.69	13.00	12.43
$N = 500$	13.00	16.54	11.65	12.04	11.46
			SR		
$N = 30$	0.60	0.38	0.67	0.72	0.94
$N = 50$	0.60	0.36	0.68	0.73	0.94
$N = 100$	0.51	0.34	0.74	0.60	0.88
$N = 225$	0.53	0.36	0.84	0.70	0.93
$N = 500$	0.47	0.44	0.84	0.58	0.90

(a) Realized Standard Deviation (SD) and Sharpe Ratio (SR)

F-Diag: Factor Model with Diagonal Residual Cov, F-PCA: Factor Model with 3-Factor PCA Residual Correlation Shrinkage, F-QIS: Factor Model with QIS Residual Cov Shrinkage.

We can see that the model outperforms significantly the baseline of low rank decomposition, meaning that our shrinkage intensity prediction comes beyond the simple risk premia that drive the variance and correlation of the stock returns.

Furthermore, to support the claim that our complex models, including the replication of the original approach, indeed produce some non-trivial mapping of features into the optimal shrinkage level, we test the "weak baselines" below.

Table 5: Weak Baselines Comparison

	MA	Last Optimal	DNK	AIRL
		SD		
$N = 30$	15.80	15.29	14.47	14.09
$N = 50$	15.82	15.48	14.17	14.05
$N = 100$	16.46	15.66	13.52	13.21
$N = 225$	23.27	24.44	12.44	12.43
$N = 500$	15.76	15.45	11.77	11.46
		SR		
$N = 30$	0.71	0.94	0.77	0.94
$N = 50$	0.74	0.74	0.77	0.94
$N = 100$	0.56	0.56	0.72	0.88
$N = 225$	0.31	0.49	0.78	0.94
$N = 500$	0.37	0.41	0.78	0.90

(a) Realized Standard Deviation (SD) and Sharpe Ratio (SR)

MA: Moving Average, DNK: De Nard et al. 2025 approach.

We can see that both the target model and the De Nard et al. 2025 approach provide us with improved performance, demonstrating learning beyond the trivial prediction rules. Moreover, as the weak predictors demonstrate drastically decaying quality with increasing size of the universe, it supports the need for more complex predictors.

In our CGP-UCB experiments we have witnesses severely inferior performance, compared to both the original De Nard et al. 2025 and our target IRL approach. And the reason for that is quite simple. We expected CGP-UCB to act as a regularized form of training, searching not for the exactly optimal solution, but for a slightly less optimal, but one that we can be certain about. However, as one can see from 8 and 9, our target models do a pretty good job of learning even extreme values of optimal shrinkage, which provides us with the certainty about model ability to learn exactly the extreme optimum in OOS construction. It means that CGP-UCB approach is expected to be inferior.

We present the results only for one universe due to additional computational budget for CGP-UCB, worse performance of which is expected to generalize onto other sets of features.

Table 6: Contextual Bandits Baselines, $N = 30$

	Kernel-UCB Discrete	CGP-UCB	AIRL
SD	15.87	15.11	14.09
SR	0.79	0.72	0.94

Finally, we test the performance, compared to the more complex baselines in terms of Risk Minimizer's expressive power. As one may argue looking at IRL results, the reason for the outperformance might be hidden inside the predictive power of Neural Nets, used in this approach, but not the performance of the IRL approach itself. We test this hypothesis by looking at very expressive GPR with RBF kernel and fitting a Neural Net with the arhcitecture, exactly equal to the Reward Net in the GAIL and AIRL approach, namely, 2-layer MLP with 32 hidden neurons. We see that our IRL approach indeed outperforms these models severely, **indicating that the improved performance comes not only from better expressive power of the function representation, but from the theoretical properties of Inverse RL approach**.

Table 7: Complex Non-Linear Baselines

	GPR-Ridge	GPR-RBF	DL	DNK	AIRL
		SD			
$N = 30$	14.20	15.08	14.91	14.46	14.09
$N = 50$	14.00	14.93	14.59	14.17	14.05
$N = 100$	13.24	14.96	14.01	13.52	13.21
$N = 225$	13.03	22.65	12.72	12.44	12.43
$N = 500$	11.60	14.70	14.01	11.77	11.46
		SR			
$N = 30$	0.90	0.67	0.74	0.77	0.94
$N = 50$	0.87	0.70	0.69	0.77	0.94
$N = 100$	0.81	0.54	0.62	0.72	0.88
$N = 225$	0.89	0.45	0.69	0.78	0.93
$N = 500$	0.81	0.40	0.62	0.78	0.90

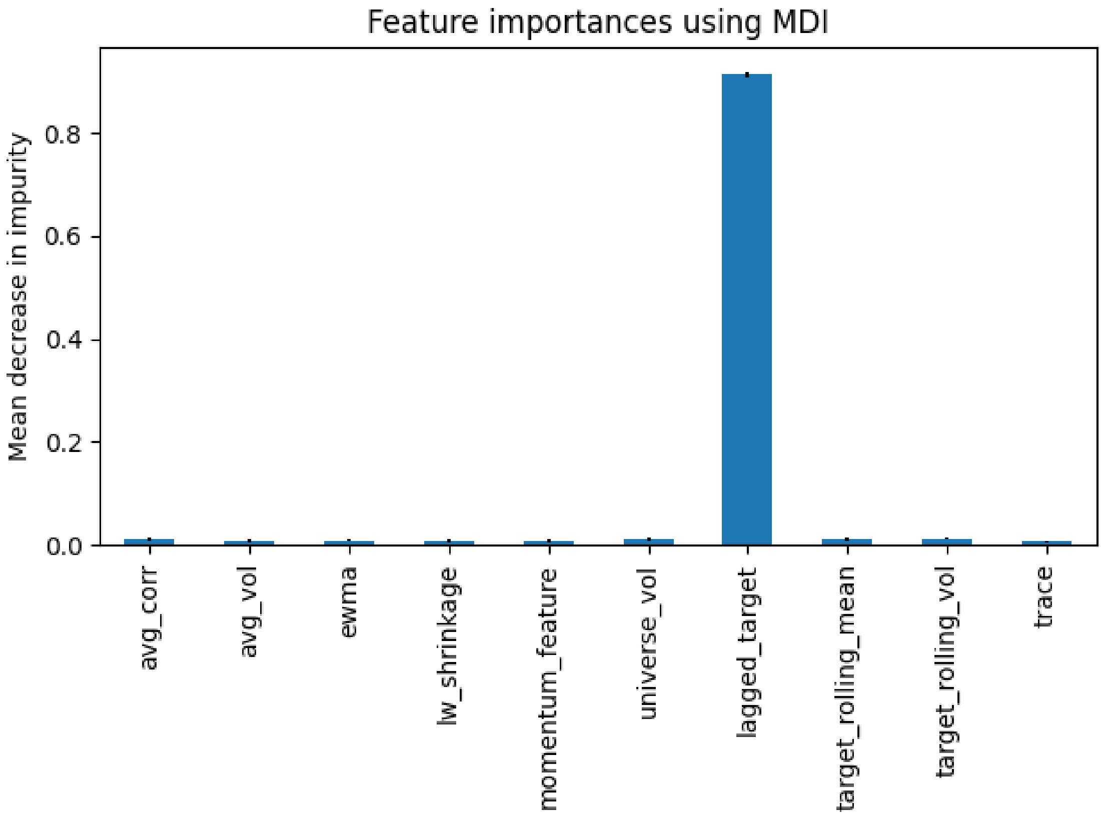
(a) Realized Standard Deviation (SD) and Sharpe Ratio (SR)

GPR-Ridge/RBF: GP Regression with DotProduct (same as Ridge Regression) / RBF Kernel, DL: FeedForward Neural Net Behavioral Cloning, DNK: De Nard et al. 2025 approach.

4.5 Feature Importance

Additionally, we analyze the most important features for the model performance. We use the Random Forest as a base model for the analysis due to its close-to-optimal performance. For the AIRL the analysis could be done in the similar manner by Integrated Gradients or Feature Ablation approaches, but as it is out-of-scope for the current results, we use a computationally cheap approach. For the feature importance construction we use the Maximum Decrease in Impurity approach.
An important note is the fact that such an approach measures the decrease in leafs impurity, if a feature is taken out of the splits, and relates to the **training** fitting rather than the testing inference. Therefore, it shows how important the feature is for the initial model fitting on training dataset.

Figure 10: Feature Importances, $N = 30$

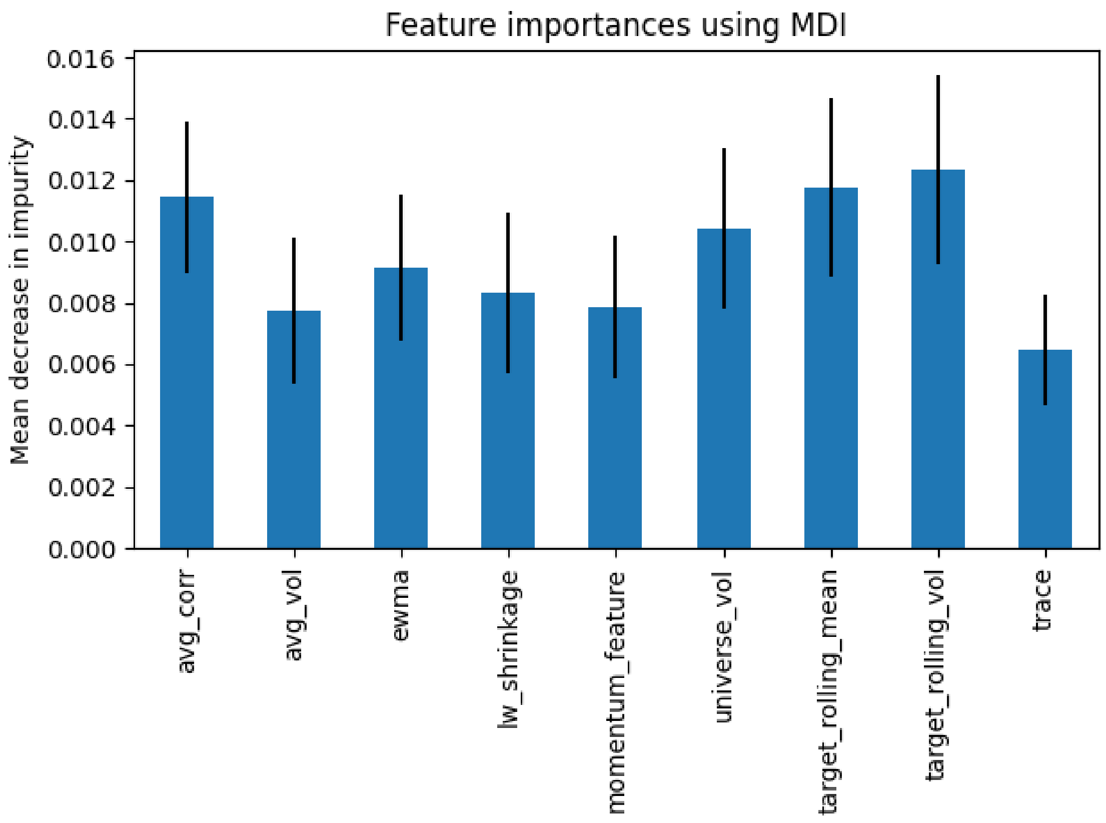


One may note excessively autoregressive structure in the prediction. It means that most of the shrinkage regimes can be indeed identified by the prevailing shrinkage in the previous step. However, as our analysis of using the last optimal value as a predictor shows, such a dependence is non-linear and non-trivial. It opens a potential road for the future work by employing the autoregressive learning structure, for instance the Long-Short Term Memory (LSTM) networks.

Moreover, one might note that while in training our lagged target is 1 day before optimal value, in inference - due to causal window (see 2.5.2) - it is 22 days before. This produces a potential overfitting errors and fixing those might give rise to the ideas for future work. Indeed, as we learn to predict well with last 1 day optimal value, but it will be unavailable in practice, we should modify the training set to include 22 day-lagged value as a prediction and drop this unattainable 1 day-lagged alternative.

Additionally, one may note that the residual feature importance beyond the lagged value is evenly distributed (and still quite high in absolute terms) across the other features, meaning that all the features are quite meaningful for shrinkage intensity prediction.

Figure 11: Feature Importances without Last Optimal, $N = 30$



4.6 Practical Implementation

As was outlined in previous sections, we test the significance of adding a 1 day lag in our methodology versus the original paper approach. We note that while the metrics indeed decay with the larger trading lags, the overall performance remains quite close to the optimal one, showing robustness in the feature delay that might be crucial in practice.

Table 8: De Nard et al. 2025 Trading Lag Analysis

	DNK No Lag	DNK 1 day lag	DNK 2 day lag	DNK 3 day lag
		SD		
$N = 30$	14.34	14.47	14.53	14.55
$N = 50$	14.17	14.17	14.19	14.24
$N = 100$	13.39	13.52	13.63	13.62
$N = 225$	12.38	12.44	12.60	12.62
$N = 500$	11.63	11.77	11.97	11.95
		SR		
$N = 30$	0.82	0.77	0.76	0.75
$N = 50$	0.78	0.77	0.78	0.77
$N = 100$	0.71	0.72	0.72	0.72
$N = 225$	0.84	0.78	0.76	0.74
$N = 500$	0.84	0.78	0.74	0.73

Table 9: Inverse RL Trading Lag Analysis

	AIRL 1 day lag	AIRL 2 day lag	AIRL 3 day lag
		SD	
$N = 30$	14.09	14.31	14.34
$N = 50$	14.05	14.15	14.17
$N = 100$	13.21	13.45	13.48
$N = 225$	12.43	12.65	12.66
$N = 500$	11.46	11.86	11.88
		SR	
$N = 30$	0.94	0.87	0.84
$N = 50$	0.94	0.91	0.85
$N = 100$	0.87	0.90	0.89
$N = 225$	0.93	0.86	0.83
$N = 500$	0.90	0.79	0.75

Moreover, we have tested the approaches, aiming to predict the optimal shrinkage with the trading lag directly (i.e. using in training features, delayed by the training lag). Such an approach demonstrated inferior performance. In the interest of space we do not include the results here, but those can be found in our open source code for replication.

5 Future Work

As was presented in this paper, Inverse RL proved to be quite a powerful framework for optimal shrinkage learning in RL-L case. We are certain that such a novel set of optimization approaches, presented by De Nard et al. 2025, is a very promising field in portfolio optimization, as it combines the theoretically optimal scheme of optimization (covariance matrix regularization, in our case).
As our paper considered only linear shrinkage, an important next logical step, following on De Nard et al. 2025, is to test the presented approaches in RL-NL (QIS) setting. Our experiments on the $N = 30$ universe show that the IRL approach outperformance holds against all of the usual baselines, but this claim should be rigourously tested, which is an area for future work.
Here are our main ideas for the future work on the presented approaches:

1. One of the first ideas for future improvement lies inside the Neural Net structure for IRL Reward Net. The logical generalization of our paper results might be learning the optimal shrinkage for each universe in one single model. Indeed, now for each N number of stocks, we needed to collect the historically optimal values of shrinkage, which is computationally expensive, while still feasible in practice. But one may go further and use several presentations of transitions for chosen set of N - with N as an additional feature for learning -, and then test the OOS performance on yet unseen N universe. One may be interested to compare two separate regimes: interpolating (for example, training on $N \in \{30, 50, 225, 500\}$ and testing on $N = 100$) and extrapolating (for example, training on $N \in \{30, 50, 100, 225\}$ and testing on $N = 500$).

2. Training with the causal window directly, using the 2.5.2 construction in training to account for this lag, which is especially crucial for lagged optimal action, as outlined in 4.5

3. As also outlined in feature analysis (4.5), which usually proves to be quite an important part of the ideas for improvement, one might be interested in considering an autoregressive structure of optimal shrinkage. The potential improvement might be in considering the RNN/LSTM neural nets in Reward Net and Policy for IRL with VAR and Bayesian VAR as baselines in the Behavioral Cloning case.

6 Conclusion

This paper contributes to the existing literature by providing the theoretical explanation and setup of the Inverse RL application for the long-term risk-optimized portfolio, introducing a procedure to fit a nearly-optimal policy for the shrinkage intensity decision-making. Our approach improves the initial ideas of De Nard et al. 2025, which are based on Behavioral Cloning, by application of a robust Imitation Learning algorithm that works well in highly varying environment (Fu et al. 2017), contrary to the distribution invariance assumption-dependent Behavioral Cloning (Abbeel et al. 2004, Ziebart et al. 2008). Our approach follows the initial ideas of De Nard et al. 2025, but improves the theoretical part of Reinforcement Learning algorithm used.
As a result we see that our RL agent learns to quantify the reward, compared to the Expert transitions (collected in the same way as in De Nard et al. 2025). The Imitation Learning approach, same as used in the original paper, robustifies the usual RL approaches by dropping the necessity to design a reward, which is a complex task in financial applications. Indeed, in our case the distribution of the reward (negative of 1-month-ahead realized portfolio standard deviation) changes over time. Therefore, the Imitation Learning approach, introduced by De Nard et al. 2025, proves to be quite handy in such a case. However, the De Nard et al. 2025 uses Behavioral Cloning approach, which suffers in non-stable environments (Abbeel et al. 2004) and can be improved by the Inverse RL approach of learning the reward function, such that in these environments we get Expert transitions to be most optimal.
Our empirical experiments show that the learned IRL policy is in line with the Behavioral Cloning policy of De Nard et al. 2025, but produces lower out-of-sample errors, better realized portfolio standard deviations and Sharpe Ratios in fully out-of-sample construction, which replicates the approach the one would use in the practical trading implementation. We focus on the linear shrinkage analysis, but show that our approach is easily generalizable to non-linear shrinkage case by applying the De Nard et al. 2025 methodology.
Finally, we show that our approach improves on more stronger baselines of Behavioral Cloning, which use Random Forest, Contextual Bandits (CGP-UCB) and Deep Learning approaches, meaning that outperformance indeed comes from theoretically more relevant Inverse RL approach.

7 References

Abbeel, Pieter and Andrew Ng (Sept. 2004). “Apprenticeship Learning via Inverse Reinforcement Learning”. In: *Proceedings, Twenty-First International Conference on Machine Learning, ICML 2004*. DOI: 10.1007/978-0-387-30164-8_417.

Agarwal, Alekh et al. (2021). *Reinforcement learning: Theory and algorithms.* https://rltheorybook.github.io.

De Nard, Gianluca and Damjan Kostovic (2025). “AI shrinkage: a data-driven approach for risk-optimized portfolios”. In: *Working paper series.*

Deng, Yue et al. (Feb. 2016). “Deep Direct Reinforcement Learning for Financial Signal Representation and Trading”. In: *IEEE Transactions on Neural Networks and Learning Systems* 28, pp. 1–12. DOI: 10.1109/TNNLS.2016.2522401.

Fu, Justin, Katie Luo, and Sergey Levine (2017). “Learning Robust Rewards with Adversarial Inverse Reinforcement Learning”. In: *CoRR* abs/1710.11248. arXiv: 1710.11248. URL: http://arxiv.org/abs/1710.11248.

Hirchoua, Badr, Brahim Ouhbi, and Bouchra Frikh (2021). “Deep reinforcement learning based trading agents: Risk curiosity driven learning for financial rules-based policy”. In: *Expert Systems with Applications* 170, p. 114553. ISSN: 0957-4174. DOI: https://doi.org/10.1016/j.eswa.2020.114553. URL: https://www.sciencedirect.com/science/article/pii/S0957417420311970.

Ho, Jonathan and Stefano Ermon (2016). “Generative Adversarial Imitation Learning”. In: *CoRR* abs/1606.03476. arXiv: 1606.03476. URL: http://arxiv.org/abs/1606.03476.

Jacot, Arthur, Franck Gabriel, and Clément Hongler (2018). “Neural tangent kernel: Convergence and generalization in neural networks”. In: *Advances in neural information processing systems* 31.

Jiang, Zhengyao, Dixing Xu, and Jinjun Liang (2017). *A Deep Reinforcement Learning Framework for the Financial Portfolio Management Problem.* arXiv: 1706.10059 [q-fin.CP]. URL: https://arxiv.org/abs/1706.10059.

Krause, A. and Cheng Ong Soon (2011). “Contextual Gaussian Process Bandit Optimization”. In: *NIPS.* DOI: https://las.inf.ethz.ch/files/krause11contextual-long.pdf.

Lai, T. L. and Herbert Robbins (1985). “Asymptotically efficient adaptive allocation rules”. In: *Advances in Applied Mathematics* 6 (1), pp. 4–22. DOI: https://doi.org/10.1016/0196-8858(85)90002-8.

Ledoit, Olivier and Michael Wolf (2003). “Improved estimation of the covariance matrix of stock returns with an application to portfolio selection”. In: *Journal of Empirical Finance* 10.5, pp. 603–621. ISSN: 0927-5398. DOI: https://doi.org/10.1016/S0927-5398(03)00007-0. URL: https://www.sciencedirect.com/science/article/pii/S0927539803000070.

Ledoit, Olivier and Michael Wolf (2020). “Quadratic Shrinkage for Large Covariance Matrices”. In: *University of Zurich, Department of Economics, Working Paper No. 335, Revised version.* DOI: https://dx.doi.org/10.2139/ssrn.3486378.

Lu, Cheng, Papa Momar Ndiaye, and Majeed Simaan (Sept. 2024). “Improved estimation of the correlation matrix using reinforcement learning and text-based networks”. In: *International Review of Financial Analysis* 96, p. 103572. DOI: 10.1016/j.irfa.2024.103572.

Matera, Giulio and Raffaele Matera (Feb. 2023). “Shrinkage estimation with reinforcement learning of large variance matrices for portfolio selection”. In: *Intelligent Systems with Applications* 17, p. 200181. DOI: 10.1016/j.iswa.2023.200181.

Schölkopf, Bernhard, Ralf Herbrich, and Alex J Smola (2001). “A generalized representer theorem”. In: *International conference on computational learning theory*. Springer, pp. 416–426.

Shavandi, Ali and Majid Khedmati (2022). “A multi-agent deep reinforcement learning framework for algorithmic trading in financial markets”. In: *Expert Systems with Applications* 208, p. 118124. ISSN: 0957-4174. DOI: https://doi.org/10.1016/j.eswa.2022.118124. URL: https://www.sciencedirect.com/science/article/pii/S0957417422013082.

Srinivas, N. et al. (2010). “Gaussian process optimization in the bandit setting: No regret and experimental design”. In: *ICML*. DOI: https://doi.org/10.48550/arXiv.0912.3995.

Xu, Tian, Ziniu Li, and Yang Yu (2020). “Error Bounds of Imitating Policies and Environments”. In: *CoRR* abs/2010.11876. arXiv: 2010.11876. URL: https://arxiv.org/abs/2010.11876.

Ziebart, Brian D. et al. (2008). “Maximum Entropy Inverse Reinforcement Learning”. In: *Proc. AAAI*, pp. 1433–1438.

8 Appendices

8.1 A. Code Source

All the code for paper replication can be found at the GitHub.

Moreover, we have implemented the easy-to-use framework for construction and backtesting of the data-driven covariance estimation for risk-optimized portfolios. which can be used to improve on the presented results and work on the future research ideas.