Reinforcement Learning and Inverse Reinforcement Learning in Goal Based Wealth Management

Rui Ding and Yizhou Li

Stony Brook University AMS Department

Sep 29th, 2021

BANK OF AMERICA

Stony Brook University

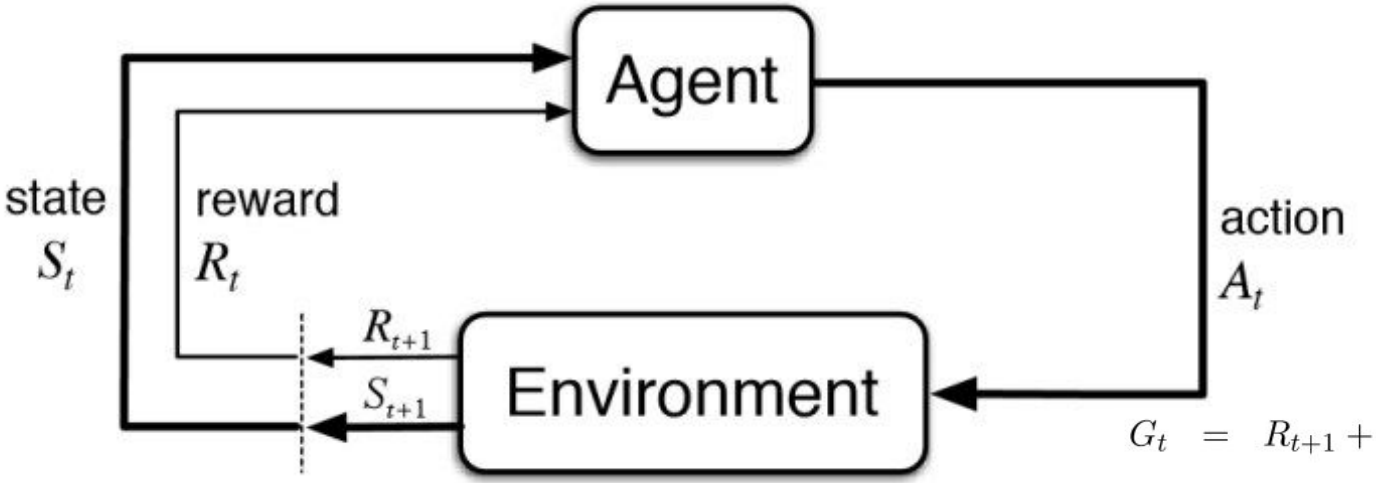
Modern Perspectives on Reinforcement Learning in Finance

Intertemporal choice is at the heart of many modern financial applications involve. These are decision making problems in which the timing of costs and benefits are spread out over time and where choices at one time influence the possibilities available at other points in time. In finance, common problems of this kind include pricing and hedging of contingent claims, investment and portfolio allocation, buying and selling a portfolio of securities subject to transaction costs, market making, asset liability management and optimization of tax consequences, to name a few.

Fundamental to these dynamic optimization problems is to determine the best actions possible that maximize the relative value between two or more payoffs at different points in time. Probably, the most common approach for solving dynamic optimization problems of this kind is dynamic programming (DP). While DP can be applied in deterministic or stochastic and discrete-time or continuous-time settings, it relies on several assumptions that are rarely true in practice, including: (i) one accurately knows the dynamics of the environment, (ii) one has enough computational resources to complete the computation of the solution and (iii) the Markov property. In many problems, we'll incur curse of dimensionality or curse of modelling.

RL provides a way of overcoming these two curses by means of building agents that act intelligently, allowing efficient solutions to problems that were considered intractable via DP. The roots of the success of RL comes from leveraging several well-known areas, including DP, Monte Carlo simulation, function approximation and machine learning (ML).

Reinforcement Learning Framework



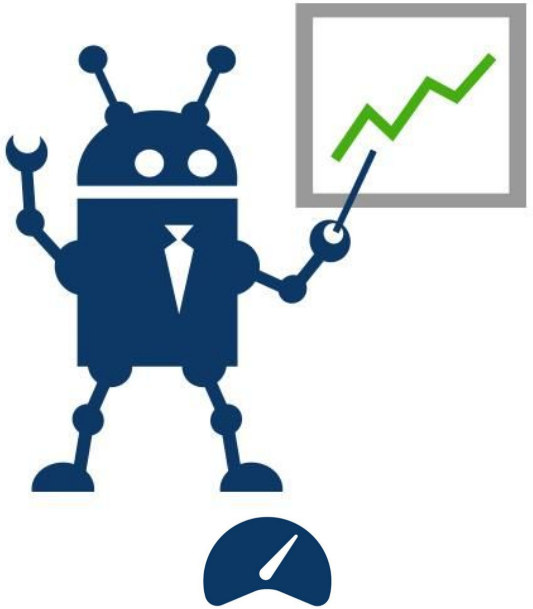
$$G_t = R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \dots$$
$$= R_{t+1} + \gamma G_{t+1}$$

In RL a Markov decision process (MDP) serves the role of providing a model of the sequential decision-making problem at hand where a decision maker interacts with a system in a sequential fashion. There are two versions of goal functions in common usage: the discounted reward goal and the average-reward (or reward-per-unit time) goal. **RL is the search for policies which maximize the expectation of the goal function.**

Reinforcement Learning in Quantitative Wealth and Investment Management

RL allows to combine “prediction" and “portfolio construction" tasks in one integrated step, thereby closely aligning ML problem with investor objectives. RL is capable of handling real-world complexity of financial planning, including effects of income taxes, mean-reverting asset classes, time-varying bond yield curves, and uncertain life expectancies. **RL allows robo-advisor to “learn” investor risk preference over time by observing her portfolio choices under different markets.**

Goal Based Investing (GBI) is an investment approach where performance is measured by the success of investments in meeting the investor financial goals. For an individual investor such goals may be retirement, education for children, or vacation home, while for institutional investors such as pension funds the goals may be aligned to the pension liabilities. The objective is to invest systematically in a consistent manner with investor’s risk profile and time horizon of the goals, and performance is measured by probabilities of success in achieving investor financial goals (at given time horizons and for specified priority levels). **Portfolio optimization for GBI can be viewed as an optimal control problem performed within a data-driven world.** GBI can be solved via reinforcement learning RL, a paradigm of learning by trial-and-error, solely from rewards or punishments. It was shown that RL can solve financial applications of intertemporal choice. Given current state-of-the-art for RL, incorporating multiple goals with their corresponding priority levels within a RL-based solution for GBI appears to be a significant modeling challenge. One possible solution may leverage hierarchical RL and curiosity-driven exploration.



Betterment

wealthfront

Literature Review: Goal Based Investing

Das et al. (“Dynamic portfolio allocation in goals-based wealth management,” 2020)

- report a dynamic programming algorithm which, given a set of efficient (or even inefficient) portfolios, constructs an optimal portfolio trading strategy that maximizes the probability of attaining an investor’s specified target wealth at the end of a designated time horizon

Das and Varma (“Dynamic Goals-Based Wealth Management using Reinforcement Learning,” 2020)

- present a reinforcement learning (RL) algorithm to solve for a dynamically optimal goal-based portfolio

Parker (“Goal-Based Portfolio Optimization,” 2016)

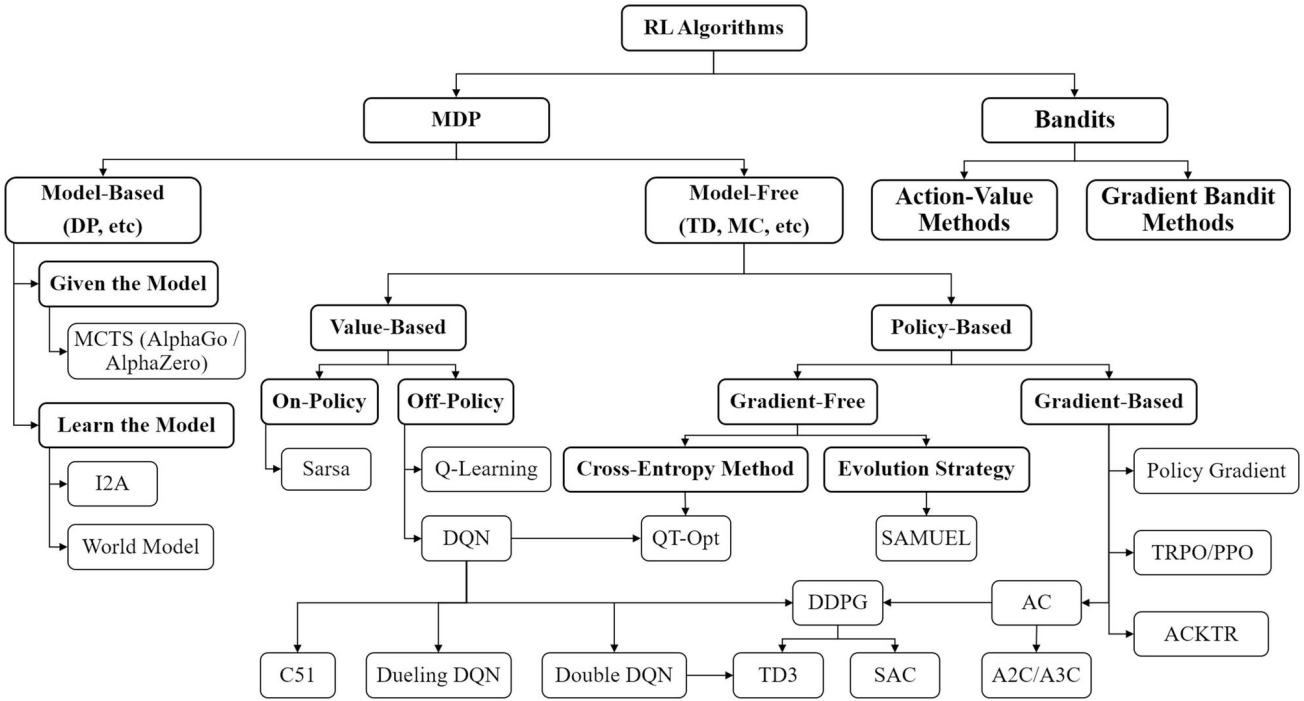
- presents an optimization procedure that seeks to minimize goal failure and maximize excess return

Parker (“Portfolio Selection in a Goal-Based Setting,” 2016)

- illustrate the deficiencies of using only modern portfolio theory (MPT) metrics and assumptions when selecting portfolios in a goal-based setting

Roncalli (“How Machine Learning Can Improve Portfolio Allocation of Robo-Advisors,” 2019)

Literature Review: Reinforcement Learning



The RL Taxonomy

Literature Review: Reinforcement Learning Cont'd

Chan et al. ("Measuring the Reliability of Reinforcement Learning Algorithms," 2020)

- propose a set of metrics that quantitatively measure different aspects of reliability, focusing on variability and risk

Dulac-Arnold et al. ("Challenges of Real-World Reinforcement Learning," 2019)

- specify the exact meaning of nine challenges, present some approaches from the literature, and specify some metrics for evaluating

Fedus et al. ("Hyperbolic Discounting and Learning over Multiple Horizons," 2019)

- demonstrate that a simple approach approximates hyperbolic discount functions while still using familiar temporal-difference learning techniques in RL

Jordan et al. ("Evaluating the Performance of Reinforcement Learning Algorithms," 2020)

- propose an evaluation methodology for RL algorithms that produces reliable measurements of performance both on a single environment and when aggregated

Laskin et al. ("Reinforcement Learning with Augmented Data," 2020)

- introduce two new data augmentations: random translate and random amplitude scale, that can enhance RL algorithm performance

Literature Review: Reinforcement Learning Cont'd

Nachum et al. ("Why Does Hierarchy (Sometimes) Work So Well in Reinforcement Learning?", 2019)

- find that most of the observed benefits of hierarchy can be attributed to improved exploration, as opposed to easier policy learning or imposed hierarchical structures

Wang et al. ("Benchmarking Model-Based Reinforcement Learning", 2019)

- benchmark a wide collection of MBRL algorithms with unified problem settings, including noisy environments

Watkins and Dayan ("Q-Learning", 1992)

- proves convergence of Q-learning, an incremental method for dynamic programming which imposes limited computational demands and works by successively improving its evaluations of the quality of particular actions at particular states

Sutton and Barto (Reinforcement Learning: An Introduction (Second Edition), 2018)

Schulman et al. ("Trust Region Policy Optimization", 2017)

- despite its approximations that deviate from the theory, TRPO tends to give monotonic improvement, with little tuning of hyperparameters

Literature Review: Deep Reinforcement Learning

Ivanov and D'yakonov ("Modern Deep Reinforcement Learning Algorithms", 2019)

- latest DRL algorithms are reviewed with a focus on their theoretical justification, practical limitations and observed empirical properties

Chakraborty ("Capturing Financial markets to apply Deep Reinforcement Learning", 2019)

- explore the usage of deep reinforcement learning algorithms to automatically generate consistently profitable, robust, uncorrelated trading signals in any general financial market

Stooke and Abbeel ("Accelerated Methods for Deep Reinforcement Learning", 2018)

- investigate how to optimize existing deep RL algorithms for modern computers, specifically for a combination of CPUs and GPUs

Xiong et al. ("Practical Deep Reinforcement Learning Approach for Stock Trading", 2018)

- train a deep reinforcement learning agent and obtain an adaptive trading strategy thus maximizing the investment returns

Zhang et al. ("A Study on Overfitting in Deep Reinforcement Learning", 2018)

- commonly used techniques in RL that add stochasticity do not necessarily prevent or detect overfitting

Literature Review: Inverse Reinforcement Learning

Abbeel and Ng (“Apprenticeship Learning via Inverse Reinforcement Learning”, 2017)

- consider learning in a Markov decision process where we are not explicitly given a reward function, but where instead we can observe an expert demonstrating the task that we want to learn to perform

Brown et al. (“Risk-Aware Active Inverse Reinforcement Learning”, 2019)

- propose a risk-aware active inverse reinforcement learning algorithm that focuses active queries on areas of the state space with the potential for large generalization error

Castro et al. (“Inverse Reinforcement Learning with Multiple Ranked Experts”, 2019)

- proposed method is particularly useful for problems where we have access to a large set of agent behaviours with varying degrees of expertise

Halperin (“The QLBS Q-Learner goes NuQLear: fitted Q iteration, inverse RL, and option portfolios”, 2019)

- the QLBS model is a discrete-time option hedging and pricing model that is based on Dynamic Programming (DP) and Reinforcement Learning (RL)

Halperin and Feldshteyn (“Market Self-Learning of Signals, Impact and Optimal Trading: Invisible Hand Inference with Free Energy (Or, How We Learned to Stop Worrying and Love Bounded Rationality)”, 2018)

- present a simple model of a non-equilibrium self-organizing market where asset prices are partially driven by investment decisions of a bounded-rational agent

G-Learner: Goal Based Wealth Management with Reinforcement Learning

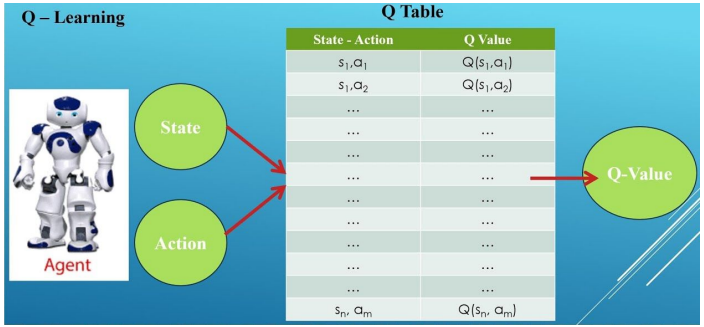
Main References:

Fox, R., A. Pakman, and N. Tishby (2015). Taming the Noise in Reinforcement Learning Via Soft Updates.

Dixon, M. F. and Halperin, I. (2020). G-Learner and GIRL: Goal Based Wealth Management with Reinforcement Learning.

G-learning (Fox et al., 2015) is a probabilistic extension of the Q-learning method of reinforcement learning.

- Q-learning is an off-policy RL method with a deterministic policy.
- G-Learning is an off-policy RL method with a stochastic policy.



In this paper, they demonstrate how G-learning, when applied to a quadratic reward and Gaussian reference policy, gives an entropy-regulated Linear Quadratic Regulator (LQR).

This critical insight provides a novel and computationally tractable tool for wealth management tasks which scales to high dimensional portfolios. **G-learning can be considered as an entropy-regularized Q-learning, which may be suitable when working with noisy data.** Because G-learning operates with stochastic policies, it amounts to a generative RL model. Previously, G-learning was applied to dynamic portfolio optimization in (Halperin and Feldshteyn, 2018), while here we extend this approach to portfolio management involving cash flows at intermediate time steps.

G-Learner Cont'd: Standard and Entropy Regularized Bellman Equations

$$V_t^{\pi}(\mathbf{x}_t) := \mathbb{E}_t^{\pi}\left[\sum_{t'=t}^{T-1}\gamma^{t'-t}\hat{R}_{t'}(\mathbf{x}_{t'},\mathbf{a}_{t'})\middle|\mathbf{x}_t\right].$$

$$V_t^{\star}(\mathbf{x}_t) = \max_{\mathbf{a}_t}\hat{R}_t(\mathbf{x}_t,\mathbf{a}_t) + \gamma\mathbb{E}_{t,\mathbf{a}_t}\left[V_{t+1}^{\star}(\mathbf{x}_{t+1})\right].$$

Let us begin by reformulating the Bellman optimality equation using a Fenchel-type representation:

$$V_t^{\star}(\mathbf{x}_t) = \max_{\pi(\cdot|y)\in\mathcal{P}}\sum_{\mathbf{a}_t\in\mathcal{A}_t}\pi(\mathbf{a}_t|\mathbf{x}_t)\left(\hat{R}_t(\mathbf{x}_t,\mathbf{a}_t) + \gamma\mathbb{E}_{t,\mathbf{a}_t}\left[V_{t+1}^{\star}(\mathbf{x}_{t+1})\right]\right). \tag{4}$$

The one-step *information cost* of a learned policy $\pi(\mathbf{a}_t|\mathbf{x}_t)$ relative to a reference policy $\pi_0(\mathbf{a}_t|\mathbf{x}_t)$ is defined as follows (Fox et al., 2015):

$$g^{\pi}(\mathbf{x}_t,\mathbf{a}_t) := \log\frac{\pi(\mathbf{a}_t|\mathbf{x}_t)}{\pi_0(\mathbf{a}_t|\mathbf{x}_t)}. \tag{5}$$

Its expectation with respect to the policy π is the Kullback-Leibler (KL) divergence of $\pi(\cdot|\mathbf{x}_t)$ and $\pi_0(\cdot|\mathbf{x}_t)$:

$$\mathbb{E}_{\pi}\left[g^{\pi}(\mathbf{x},\mathbf{a})|\,\mathbf{x}_t\right] = KL[\pi||\pi_0](\mathbf{x}_t) := \sum_{\mathbf{a}_t}\pi(\mathbf{a}_t|\mathbf{x}_t)\log\frac{\pi(\mathbf{a}_t|\mathbf{x}_t)}{\pi_0(\mathbf{a}_t|\mathbf{x}_t)}. \tag{6}$$

G-Learner Cont'd: Entropy-regularized Bellman optimality equation

The total discounted information cost for a trajectory is defined as follows:

$$I^{\pi}(\mathbf{x}_t) := \sum_{t'=t}^{T} \gamma^{t'-t} \mathbb{E}_t^{\pi} \left[g^{\pi}(\mathbf{x}_{t'}, \mathbf{a}_{t'}) | \, \mathbf{x}_t \right]. \tag{7}$$

The *free energy* function $F_t^{\pi}(\mathbf{x}_t)$ is defined as the value function (4) augmented by the information cost penalty (7) which is added using a regularization parameter $1/\beta$:

$$F_t^{\pi}(\mathbf{x}_t) := V_t^{\pi}(\mathbf{x}_t) - \frac{1}{\beta} I^{\pi}(\mathbf{x}_t) = \sum_{t'=t}^{T} \gamma^{t'-t} \mathbb{E}_t^{\pi} \left[\hat{R}_{t'}(\mathbf{x}_{t'}, \mathbf{a}_{t'}) - \frac{1}{\beta} g^{\pi}(\mathbf{x}_{t'}, \mathbf{a}_{t'}) \right]. \tag{8}$$

A Bellman equation for the free energy function $F_t^{\pi}(\mathbf{x}_t)$ is obtained from Eq.(8):

$$F_t^{\pi}(\mathbf{x}_t) = \mathbb{E}_{\mathbf{a}|\mathbf{y}} \left[\hat{R}_t(\mathbf{x}_t, \mathbf{a}_t) - \frac{1}{\beta} g^{\pi}(\mathbf{x}_t, \mathbf{a}_t) + \gamma \mathbb{E}_{t,\mathbf{a}} \left[F_{t+1}^{\pi}(\mathbf{x}_{t+1}) \right] \right]. \tag{9}$$

For a finite-horizon setting with a terminal reward $\hat{R}_T(\mathbf{x}_t, \mathbf{a}_T)$, Eq.(9) should be supplemented by a terminal condition

$$F_T^{\pi}(\mathbf{x}_t) = \hat{R}_T(\mathbf{x}_t, \mathbf{a}_T^{\star}) \tag{10}$$

G-Learner Cont'd: G-function and optimal policy

Similar to the action-value function, we define the state-action free energy function $G^{\pi}(\mathbf{x}, \mathbf{a})$ as (Fox et al., 2015)

$$
\begin{aligned}
G_t^{\pi}(\mathbf{x}_t, \mathbf{a}_t) &= \hat{R}_t(\mathbf{x}_t, \mathbf{a}_t) + \gamma \mathbb{E}\left[F_{t+1}^{\pi}(\mathbf{x}_{t+1}) \middle| \mathbf{x}_t, \mathbf{a}_t \right] \qquad (11) \\
&= \hat{R}_t(\mathbf{x}_t, \mathbf{a}_t) + \gamma \mathbb{E}_{t,\mathbf{a}} \left[\sum_{t'=t+1}^{T} \gamma^{t'-t-1} \left(\hat{R}_{t'}(\mathbf{x}_{t'}, \mathbf{a}_{t'}) - \frac{1}{\beta} g^{\pi}(\mathbf{x}_{t'}, \mathbf{a}_{t'}) \right) \right] \\
&= \mathbb{E}_{t,\mathbf{a}_t} \left[\sum_{t'=t}^{T} \gamma^{t'-t} \left(\hat{R}_{t'}(\mathbf{x}_{t'}, \mathbf{a}_{t'}) - \frac{1}{\beta} g^{\pi}(\mathbf{x}_{t'}, \mathbf{a}_{t'}) \right) \right],
\end{aligned}
$$

$$
F_t^{\pi}(\mathbf{x}_t) = \sum_{\mathbf{a}_t} \pi(\mathbf{a}_t|\mathbf{x}_t) \left[G_t^{\pi}(\mathbf{x}_t, \mathbf{a}_t) - \frac{1}{\beta} \log \frac{\pi(\mathbf{a}_t|\mathbf{x}_t)}{\pi_0(\mathbf{a}_t|\mathbf{x}_t)} \right]. \qquad (12)
$$

This functional is maximized by the following distribution $\pi(\mathbf{a}_t|\mathbf{x}_t)$:

$$
\begin{aligned}
\pi(\mathbf{a}_t|\mathbf{x}_t) &= \frac{1}{Z_t} \pi_0(\mathbf{a}_t|\mathbf{x}_t) e^{\beta G_t^{\pi}(\mathbf{x}_t, \mathbf{a}_t)} \qquad (13) \\
Z_t &= \sum_{\mathbf{a}_t} \pi_0(\mathbf{a}_t|\mathbf{x}_t) e^{\beta G_t^{\pi}(\mathbf{x}_t, \mathbf{a}_t)}.
\end{aligned}
$$

The free energy (12) evaluated at the optimal solution (13) becomes

$$
F_t^{\pi}(\mathbf{x}_t) = \frac{1}{\beta} \log Z_t = \frac{1}{\beta} \log \sum_{\mathbf{a}_t} \pi_0(\mathbf{a}_t|\mathbf{x}_t) e^{\beta G_t^{\pi}(\mathbf{x}_t, \mathbf{a}_t)}. \qquad (14)
$$

Using Eq.(14), the optimal action policy can be written as follows :

$$
\pi(\mathbf{a}_t|\mathbf{x}_t) = \pi_0(\mathbf{a}_t|\mathbf{x}_t) e^{\beta \left(G_t^{\pi}(\mathbf{x}_t, \mathbf{a}_t) - F_t^{\pi}(\mathbf{x}_t) \right)}. \qquad (15)
$$

G-Learner Cont'd: Summarizing G-Learning

In the RL setting when rewards are observed, the system Eqs.(14, 15, 16) can be reduced to one non-linear equation. Substituting the augmented free energy (14) into Eq.(16), we obtain

$$G_t^{\pi}(\mathbf{x}, \mathbf{a}) = \hat{R}(\mathbf{x}_t, \mathbf{a}_t) + \mathbb{E}_{t,\mathbf{a}}\left[\frac{\gamma}{\beta}\log\sum_{\mathbf{a}_{t+1}}\pi_0(\mathbf{a}_{t+1}|\mathbf{x}_{t+1})e^{\beta G_{t+1}^{\pi}(\mathbf{x}_{t+1},\mathbf{a}_{t+1})}\right]. \tag{18}$$

This equation provides a soft relaxation of the Bellman optimality equation for the action-value Q-function, with the G-function defined in Eq.(11) being an entropy-regularized Q-function (Fox et al., 2015). The "inverse-temperature" parameter β in Eq.(18) determines the strength of entropy regularization. In particular, if we take a "zero-temperature" limit $\beta \to \infty$, we recover the original Bellman optimality equation for the Q-function. Because the last term in (18) approximates the $\max(\cdot)$ function when β is large but finite, for a particular choice of a uniform reference distribution π_0, Eq.(18) is known in the literature as "soft Q-learning".

For finite values $\beta < \infty$, in a setting of Reinforcement Learning with observed rewards, Eq.(18) can be used to specify *G-learning* (Fox et al., 2015): an off-policy time-difference (TD) algorithm that generalizes Q-learning to noisy environments where an entropy-based regularization is appropriate.

Applications Case:
Portfolio optimization for a defined contribution retirement plan

G-Learner Cont'd: Portfolio optimization for a defined contribution retirement plan

We assume that at each time step t, there is a pre-specified target value $\hat{P}_{t+1}$ of a portfolio at time $t+1$. We assume that the target value $\hat{P}_{t+1}$ at step t exceeds the next-step value $V_{t+1} = (1+\mathbf{r}_t)(\mathbf{x}_t + \mathbf{u}_t)$ of the portfolio, and we seek to impose a penalty for under-performance relative to this target. To this end, we can consider the following expected reward for time step t:

$$R_t(\mathbf{x}_t, \mathbf{u}_t, c_t) = -c_t - \lambda \mathbb{E}_t \left[\left(\hat{P}_{t+1} - (1+\mathbf{r}_t)(\mathbf{x}_t + \mathbf{u}_t) \right)_+ \right] - \mathbf{u}_t^T \mathbf{\Omega} \mathbf{u}_t . \tag{19}$$

We therefore modify the one-step reward (19) in two ways: we replace the first term using Eq.(20), and approximate the rectified non-linearity by a quadratic function. The new one-step reward is

$$R_t(\mathbf{x}_t, \mathbf{u}_t) = -\sum_{n=1}^{N} u_{tn} - \lambda \mathbb{E}_t \left[\left(\hat{P}_{t+1} - (1+\mathbf{r}_t)(\mathbf{x}_t + \mathbf{u}_t) \right)^2 \right] - \mathbf{u}_t^T \mathbf{\Omega} \mathbf{u}_t . \tag{21}$$

$$\hat{P}_{t+1} = (1-\rho) B_t + \rho \eta \, \mathbf{1}^{\mathbf{T}} \mathbf{x_t},$$

$$\begin{aligned} R_t(\mathbf{x}_t, \mathbf{u}_t) &= -\lambda \hat{P}_{t+1}^2 - \mathbf{u}_t^T \mathbb{1} + 2\lambda \hat{P}_{t+1} (\mathbf{x}_t + \mathbf{u_t})^{\mathbf{T}} (\mathbf{1} + \bar{\mathbf{r}}_t) - \lambda \, (\mathbf{x_t} + \mathbf{u_t})^{\mathbf{T}} \, \hat{\mathbf{\Sigma}}_{\mathbf{t}} \, (\mathbf{x_t} + \mathbf{u_t}) - \mathbf{u_t^T \Omega u_t} \\ &= \mathbf{x}_t^T \mathbf{R}_t^{(xx)} \mathbf{x}_t + \mathbf{u}_t^T \mathbf{R}_t^{(ux)} \mathbf{x}_t + \mathbf{u}_t^T \mathbf{R}_t^{(uu)} \mathbf{u}_t + \mathbf{x}_t^T \mathbf{R}_t^{(x)} + \mathbf{u}_t^T \mathbf{R}_t^{(u)} + R_t^{(0)} \end{aligned}$$

G-Learner Cont'd: G-learner for retirement plan optimization

We start by specifying a functional form of the value function as a quadratic form of $\mathbf{x}_t$:

$$F_t^{\pi}(\mathbf{x}_t) = \mathbf{x}_t^T \mathbf{F}_t^{(xx)} \mathbf{x}_t + \mathbf{x}_t^T \mathbf{F}_t^{(x)} + F_t^{(0)}, \tag{24}$$

where $\mathbf{F}_t^{(xx)}$, $\mathbf{F}_t^{(x)}$, $F_t^{(0)}$ are parameters that can depend on time via their dependence on the target values $\hat{P}_{t+1}$ and the expected returns $\bar{\mathbf{r}}_t$. The dynamic equation takes the form:

$$\mathbf{x}_{t+1} = \mathbf{A}_t (\mathbf{x}_t + \mathbf{u}_t) + (\mathbf{x}_t + \mathbf{u}_t) \circ \tilde{\varepsilon}_t, \quad \mathbf{A}_t := \mathrm{diag}(1 + \bar{\mathbf{r}}_t), \quad \tilde{\varepsilon}_t := (0, \varepsilon_t) \tag{25}$$

Coefficients of the value function (24) are computed backward in time starting from the last maturity $t = T - 1$. For $t = T - 1$, the quadratic reward (23) can be optimized analytically by the following action:

$$\mathbf{u}_{T-1} = \tilde{\mathbf{\Sigma}}_{T-1}^{-1} \left(\frac{1}{2\lambda} \mathbf{R}_t^{(u)} + \frac{1}{2\lambda} \mathbf{R}_t^{(ux)} \mathbf{x}_{T-1} \right) \tag{26}$$

where we defined $\tilde{\mathbf{\Sigma}}_{T-1}$ as follows

$$\tilde{\mathbf{\Sigma}}_{T-1} := \hat{\mathbf{\Sigma}}_{T-1} + \frac{1}{\lambda} \mathbf{\Omega}. \tag{27}$$

$$G_t^{\pi}(\mathbf{x}_t, \mathbf{u}_t) = \hat{R}_t(\mathbf{x}_t, \mathbf{u}_t) + \gamma \mathbb{E}_{t,\mathbf{u}} \left[F_{t+1}^{\pi}(\mathbf{x}_{t+1}) \middle| \mathbf{x}_t, \mathbf{u}_t \right]$$

we see that the action-value function $G_t^{\pi}(\mathbf{x}_t, \mathbf{u}_t)$ should also be a quadratic function of $\mathbf{x}_t$ and $\mathbf{u}_t$:

$$G_t^{\pi}(\mathbf{x}_t, \mathbf{u}_t) = \mathbf{x}_t^T \mathbf{Q}_t^{(xx)} \mathbf{x}_t + \mathbf{u}_t^T \mathbf{Q}_t^{(ux)} \mathbf{x}_t + \mathbf{u}_t^T \mathbf{Q}_t^{(uu)} \mathbf{u}_t + \mathbf{x}_t^T \mathbf{Q}_t^{(x)} + \mathbf{u}_t^T \mathbf{Q}_t^{(u)} + Q_t^{(0)}, \tag{30}$$

G-Learner Cont'd: G-learner for retirement plan optimization

After the action-valued function is computed as per Eqs.(31), what remains is to compute the F-function for the current step:

$$F_t^{\pi}(\mathbf{x}_t) = \frac{1}{\beta} \log \int \pi_0(\mathbf{u}_t|\mathbf{x}_t) e^{\beta G_t^{\pi}(\mathbf{x}_t, \mathbf{u}_t)} d\mathbf{u}_t. \tag{32}$$

A reference policy $\pi_0(\mathbf{u}_t|\mathbf{x}_t)$ is Gaussian:

$$\pi_0(\mathbf{u}_t|\mathbf{x}_t) = \frac{1}{\sqrt{(2\pi)^n |\Sigma_p|}} e^{-\frac{1}{2}(\mathbf{u}_t - \hat{\mathbf{u}}_t)^T \Sigma_p^{-1} (\mathbf{u}_t - \hat{\mathbf{u}}_t)}, \tag{33}$$

where the mean value $\hat{\mathbf{u}}_t$ is a linear function of the state $\mathbf{x}_t$:

$$\hat{\mathbf{u}}_t = \bar{\mathbf{u}}_t + \bar{\mathbf{v}}_t \mathbf{x}_t. \tag{34}$$

The optimal policy for the given step is given by

$$\pi(\mathbf{u}_t|\mathbf{x}_t) = \pi_0(\mathbf{u}_t|\mathbf{x}_t) e^{\beta\left(G_t^{\pi}(\mathbf{x}_t, \mathbf{u}_t) - F_t^{\pi}(\mathbf{x}_t)\right)}. \tag{38}$$

Using here the quadratic action-value function (30) produces a new Gaussian policy $\pi(\mathbf{u}_t|\mathbf{x}_t)$:

$$\pi(\mathbf{u}_t|\mathbf{x}_t) = \frac{1}{\sqrt{(2\pi)^n |\tilde{\Sigma}_p|}} e^{-\frac{1}{2}(\mathbf{u}_t - \tilde{\mathbf{u}}_t - \tilde{\mathbf{v}}_t \mathbf{x}_t)^T \tilde{\Sigma}_p^{-1} (\mathbf{u}_t - \hat{\mathbf{u}}_t - \tilde{\mathbf{v}}_t \mathbf{x}_t)} \tag{39}$$

GIRL: G-Learner Inverse Reinforcement Learning

In this section, we consider the IRL problem with G-learning, and present an algorithm we call **GIRL (G-learning IRL) whose objective is to make inference of the reward function of an individual agent such as a retirement plan contributor or an individual brokerage account holder.** That is, we assume that we are given a history of dollar-nominated asset positions in an investment portfolio, jointly with an agent's decisions that include both injections or withdrawals of cash from the portfolio and asset allocation decisions. Additionally, we are given historical values of asset prices and expected asset returns for all assets in the investor universe.

Assume that we have historical data that includes a set of D trajectories ζ_i where $i = 1, \dots D$ of state-action pairs $(\mathbf{x}_t, \mathbf{u}_t)$ where trajectory i starts at some time t_{0i} and runs until time T_i.

The probability of observing trajectory ζ is given by the following expression

$$P(\mathbf{x}, \mathbf{u}|\Theta) = p_0(\mathbf{x}_0) \prod_{t=0}^{T-1} \pi_\theta(\mathbf{u}_t|\mathbf{x}_t) p_\theta(\mathbf{x}_{t+1}|\mathbf{x}_t, \mathbf{u}_t) . \tag{41}$$

Transition probabilities $p_\theta(\mathbf{x}_{t+1}|\mathbf{x}_t, \mathbf{u}_t)$ entering this expression can be obtained from the state equation

$$\mathbf{x}_{t+1} = \mathbf{A}_t(\mathbf{x}_t + \mathbf{u}_t) + (\mathbf{x}_t + \mathbf{u}_t) \circ \tilde{\varepsilon}_t, \quad \mathbf{A}_t := \text{diag}(1 + \bar{\mathbf{r}}_t), \quad \tilde{\varepsilon}_t := (0, \varepsilon_t), \tag{43}$$

$$p_\theta(\mathbf{x}_{t+1}|\mathbf{x}_t, \mathbf{u}_t) = \frac{e^{-\frac{1}{2}\mathbf{\Delta}_t^T \mathbf{\Sigma}_r^{-1} \mathbf{\Delta}_t}}{\sqrt{(2\pi)^N |\mathbf{\Sigma}_r|}} \delta\left(x_{t+1}^{(0)} - (1 + r_f) x_t^{(0)}\right), \quad \mathbf{\Delta}_t := \frac{\mathbf{x}_{t+1}^{(r)}}{\mathbf{x}_t^{(r)} + \mathbf{u}_t^{(r)}} - \vec{\mathbf{A}}_t^{(r)}, \tag{44}$$

A "proper" IRL setting would correspond to only learning parameters of the reward function from the log-likelihood.

G-Learner and GIRL Numerical Results

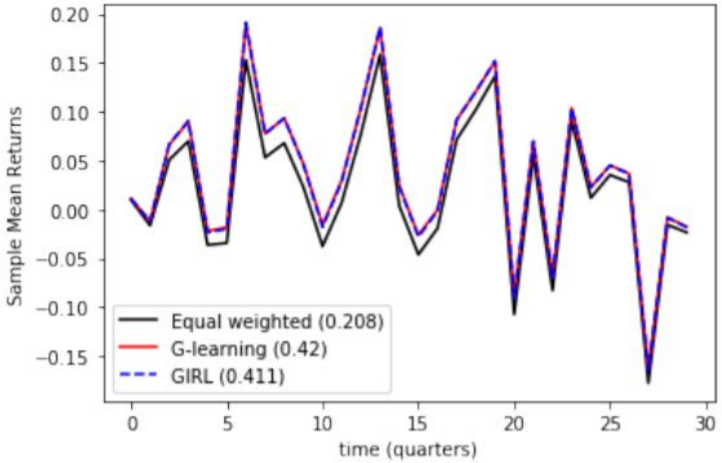


Figure 2: *The sample mean portfolio returns are shown over a 30 quarterly period horizon (7.5 years). The black line shows the sample mean returns for an equally weighted portfolio without rebalancing. The red line shows a G-learning agent, for the parameter values given in Table 1. GIRL imitates the G-learning agent and generates returns shown by the blue dashed line. Sharpe Ratios are shown in parentheses.*

Robo-advising: Learning Investors' Risk Preferences via Portfolio Choices

Main Reference: Humoud Alsabah, Agostino Capponi, Octavio Ruiz Lacedelli, and Matt Stern, Robo-advising: Learning Investors' Risk Preferences via Portfolio Choices.

The robo-advisor does not know the investor's risk preference, but learns it over time by observing her portfolio choices in different market environments. **They develop an exploration-exploitation algorithm which trades off costly solicitations of portfolio choices by the investor with autonomous trading decisions based on stale estimates of investor's risk aversion.** They show that the algorithm's value function converges to the optimal value function of an omniscient robo-advisor over a number of periods that is polynomial in the state and action space. By correcting for the investor's mistakes, the robo-advisor may outperform a stand-alone investor, regardless of the investor's opportunity cost for making portfolio decisions. It considers objective function of the form: $u(\theta_{s_t}, s_t, a_t) = \mathbb{E}[X_{s_t,a_t}] - \theta_{s_t}\mathbb{D}[X_{s_t,a_t}]$

Where D is a dispersion term capturing the uncertainty of portfolio returns which may be taken to be variance, central semideviation, or weighted mean deviations from quantiles:

$$\mathbb{D}[X_{s_t,a_t}] := Var[X_{s_t,a_t}] \quad \mathbb{D}[X_{s_t,a_t}] := \left(\mathbb{E}\left[(\mathbb{E}[X_{s_t,a_t}] - X_{s_t,a_t})_+^p\right]\right)^{1/p} \quad \mathbb{D}[X_{s_t,a_t}] := \mathbb{E}\left[\max\{(1-\alpha)(H^{-1}_{X_{s_t,a_t}}(\alpha) - X_{s_t,a_t}), \alpha(X_{s_t,a_t} - H^{-1}_{X_{s_t,a_t}}(\alpha))\}\right]$$

Algorithm 1. *Input $C(s)$ for all $s \in S$. Set $N(s) = 0$ and $\hat{\theta}_s = 0$ for all $s \in S$. Let s_1 be prevailing economic state at time 1. For $t = 1, 2, \ldots, T$:*

1. *If $N(s_t) < C(s)$, set $a_t^{\boldsymbol{R}} :=$ ask and $N(s_t) = N(s_t) + 1$. Observe the investor's action $a_t^{\boldsymbol{I}}$, and update as $\hat{\theta}_{s_t} = \hat{\theta}_{s_t} + \frac{g^{-1}(a_t^{\boldsymbol{I}};s_t) - \hat{\theta}_{s_t}}{N(s_t)}$.*

2. *Otherwise, set $a_t^{\boldsymbol{R}} := \operatorname{argmax}_a r(s_t, a, \bar{\theta})$ where $\bar{\theta} = \operatorname{argmin}_{x\in\Theta} |x - \hat{\theta}_{s_t}|$.*

Data Description

To begin with, we consider the tickers provided in the EOD metadata, there are roughly 18000 of them. The daily open, high, low, close prices and volume can be extracted from the complete dataset.

We plan on using a selected pool of the data, say for example the stocks in S&P 500, as our initial universe. The S&P 500 index, along with the equally weighted portfolio, will serve as benchmark performances in our simulation.

We plan to simulate some baseline/expert strategies(which may be the result from some rolling window optimization schemes) along with the G-learning algorithm, while using the GIRL algorithm to inverse learn the optimization hyperparameters and try to mimic those strategy behaviors. For example, some of the expert strategies may result from a Markowitz mean-variance optimization with a single hyperparameter indicating investor's risk aversion preference, which we can search in a gridpoint style. By changing the input estimates for mean and covariance matrix in the Markowitz problem, we can obtain different strategy behaviors as well.

$$Minimize \quad z = \sigma_p^2 - \lambda R_p$$

$$Where,$$

$$R_p = \sum_{i=1}^{N} x_i R_i$$

$$\sigma_p = \sum_{i=1}^{N} \sum_{j=1}^{N} x_i x_j \rho_{ij} \sigma_i \sigma_j$$

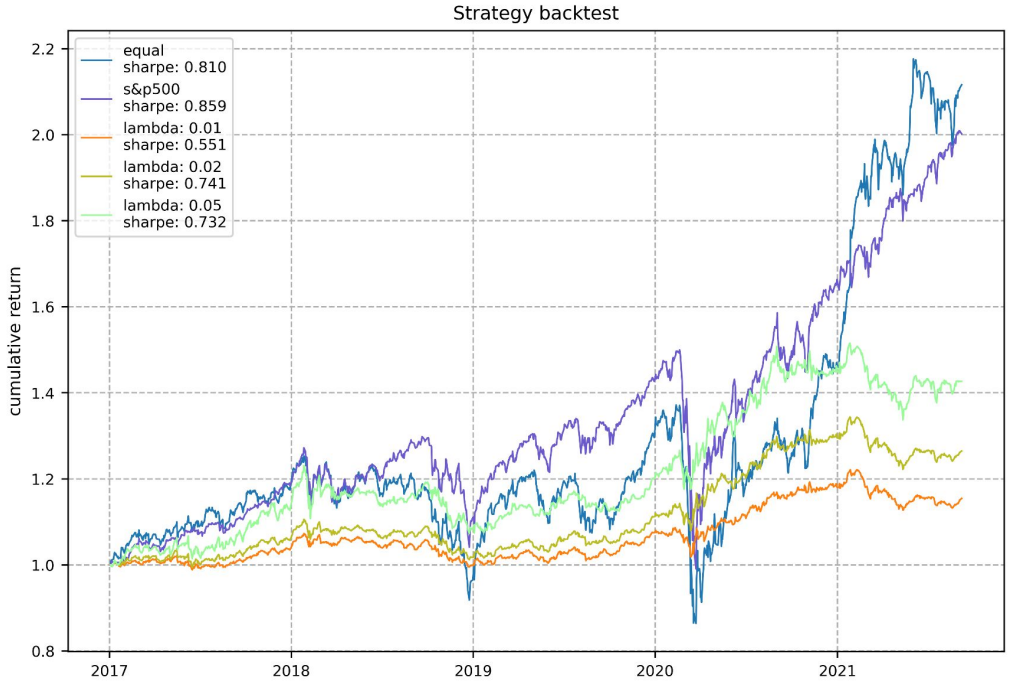
$$subject \ to: \sum_{i=1}^{N} x_i = 1$$

$$Additional \ \ Constraints$$

$$0 < \lambda < +\infty$$

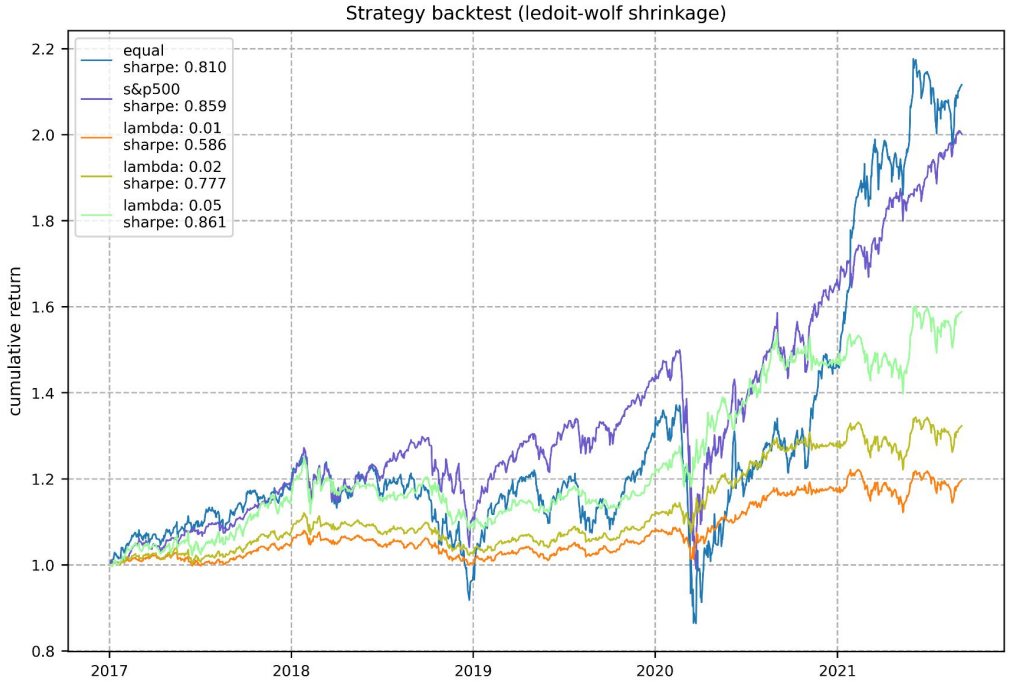
$$L_i \le x_i \le U_i$$

Expert Strategy Simulations



We selected the most liquid 30 stocks with one year (252 days) rolling window. The backtest period is from 2016-01-01 to 2021-09-07, and the portfolio positions are adjusted monthly. We chose risk aversion parameter λ = 0.01, 0.02 and 0.05. The bound constraint for each stock is (0, 0.2).

Expert Strategy Simulations



We also employed the Ledoit-wolf shrinkage covariance and mean in place of original covariance matrix and mean into portfolio optimization.


Thank You!