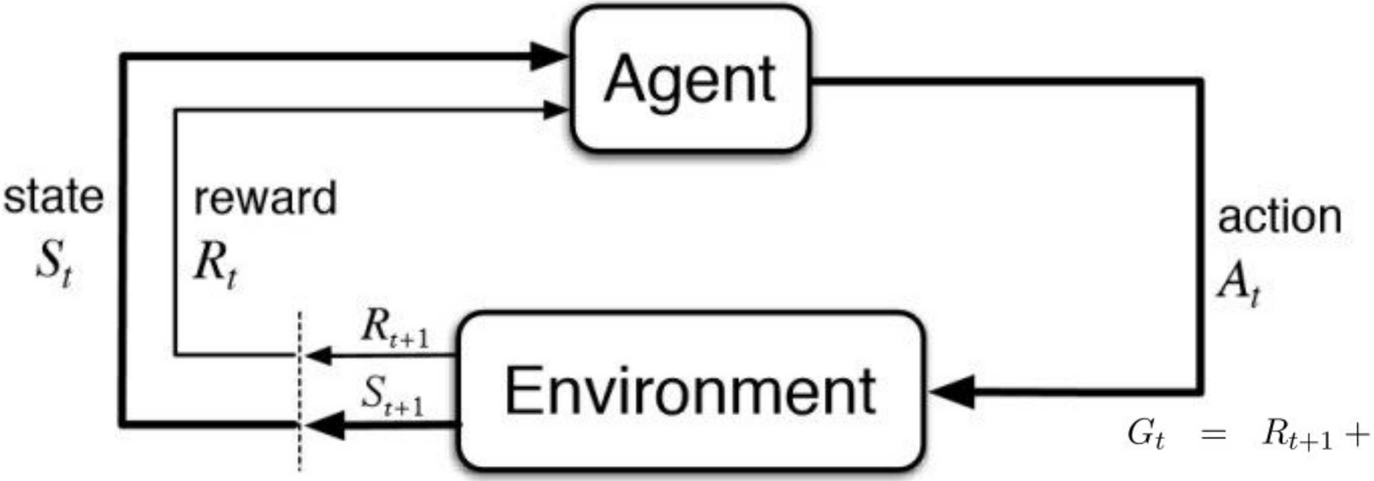
Reinforcement Learning and Inverse Reinforcement Learning in Goal Based Wealth Management

Rui Ding and Yizhou Li

Stony Brook University AMS Department

Nov 3rd, 2021

Review: Reinforcement Learning Framework



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

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.**

G-Learner: Goal Based Wealth Management with Reinforcement Learning

Main References:

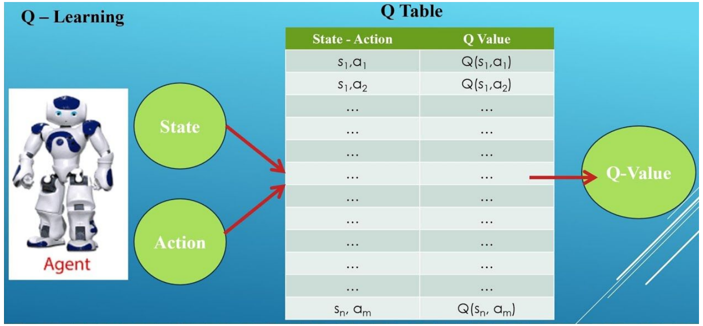
Fox, R., A. Pakman, and N. Tishby (2015). Taming the Noise in Reinforcement Learning Via Soft Updates.

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).

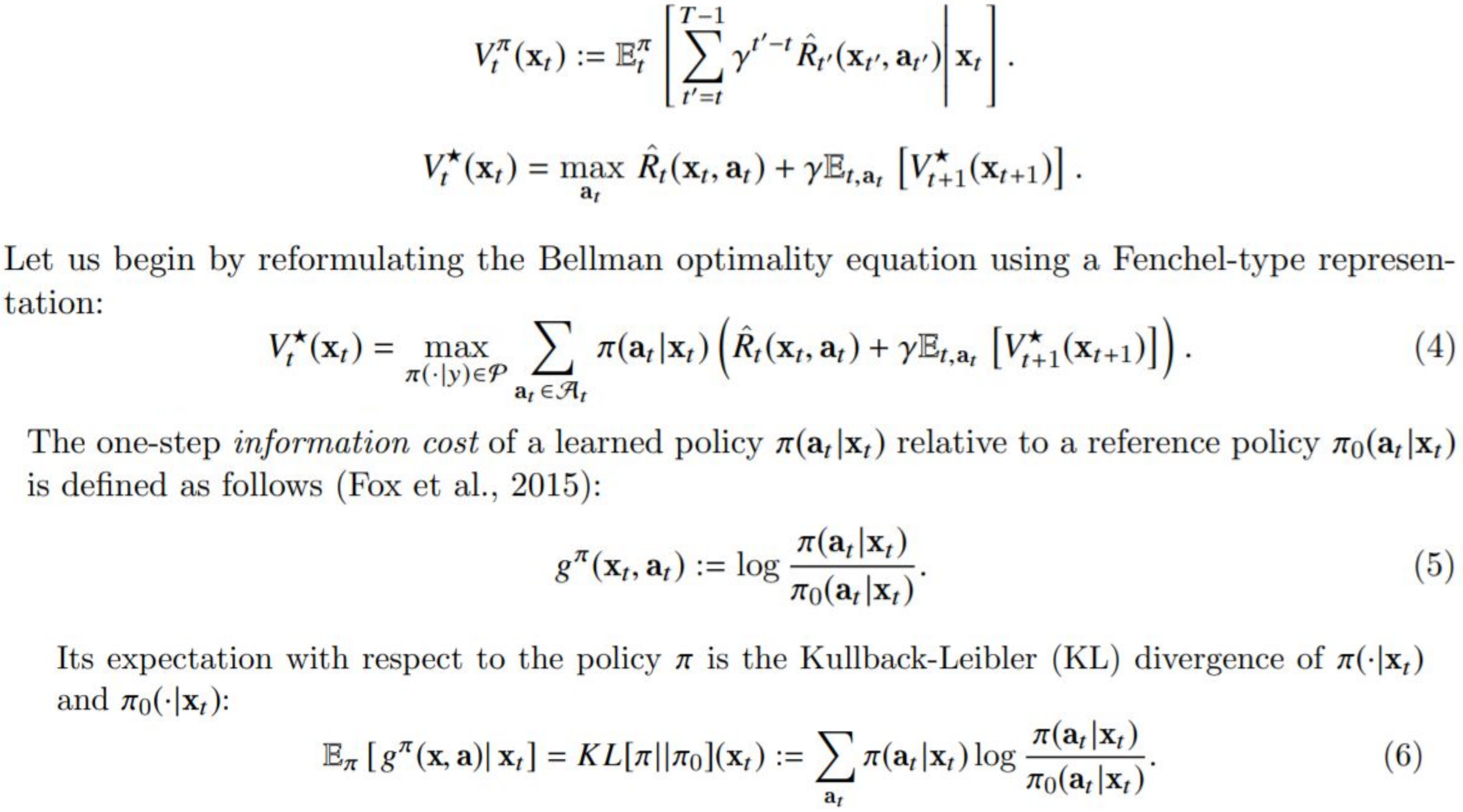


Q Table

State - Action	Q Value
s_1,a_1	$Q(s_1,a_1)$
s_1,a_2	$Q(s_1,a_2)$
…	…
…	…
…	…
…	…
…	…
…	…
…	…
…	…
…	…
s_n, a_m	$Q(s_n, a_m)$

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:

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

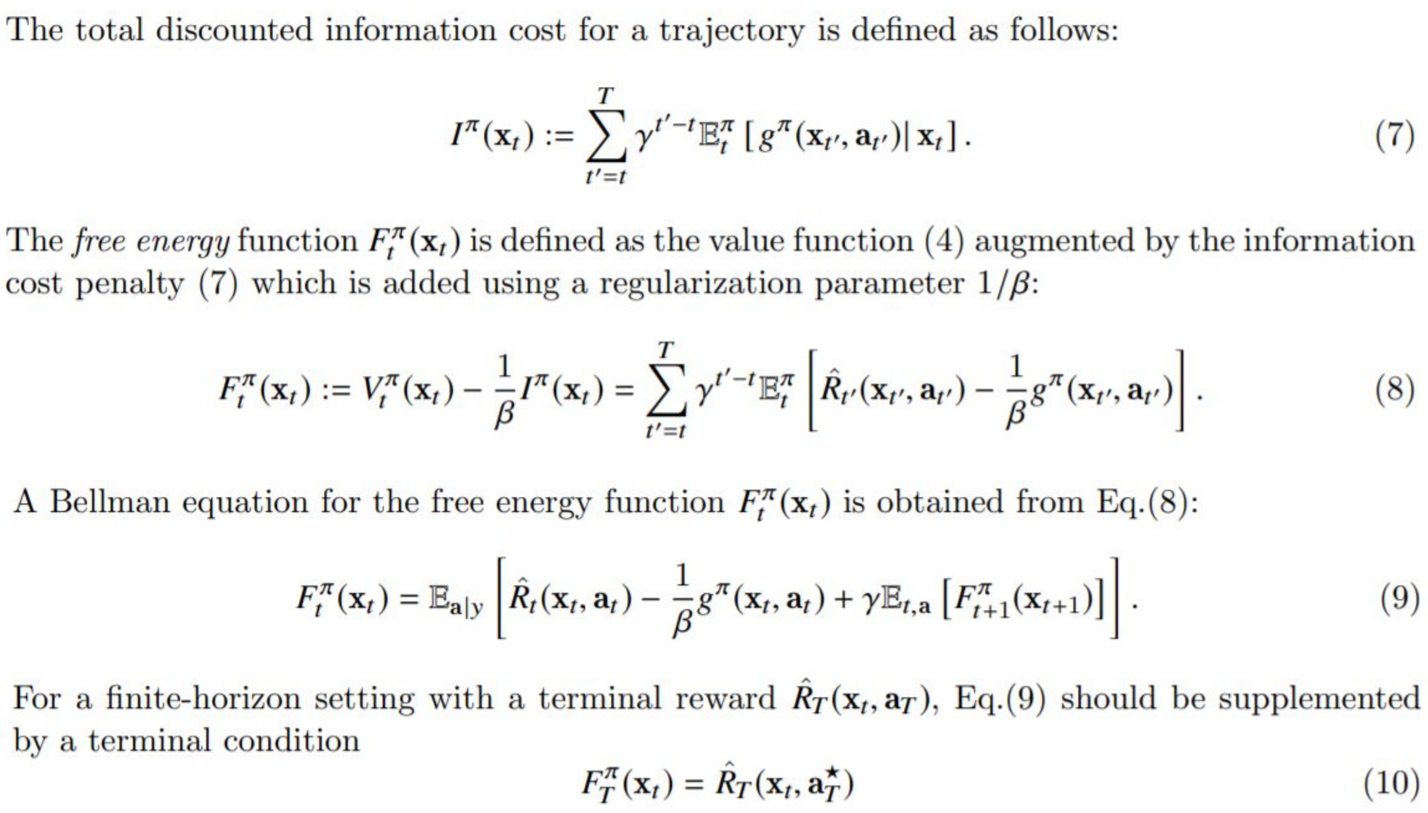
The one-step *information cost* of a learned policy $\pi(\mathbf{a}_t|\mathbf{x}_t)$ relative to a reference policy $\pi_0(\mathbf{a}_t|\mathbf{x}_t)$ is defined as follows (Fox et al., 2015):

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

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

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

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



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

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

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 $1/\beta$:

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

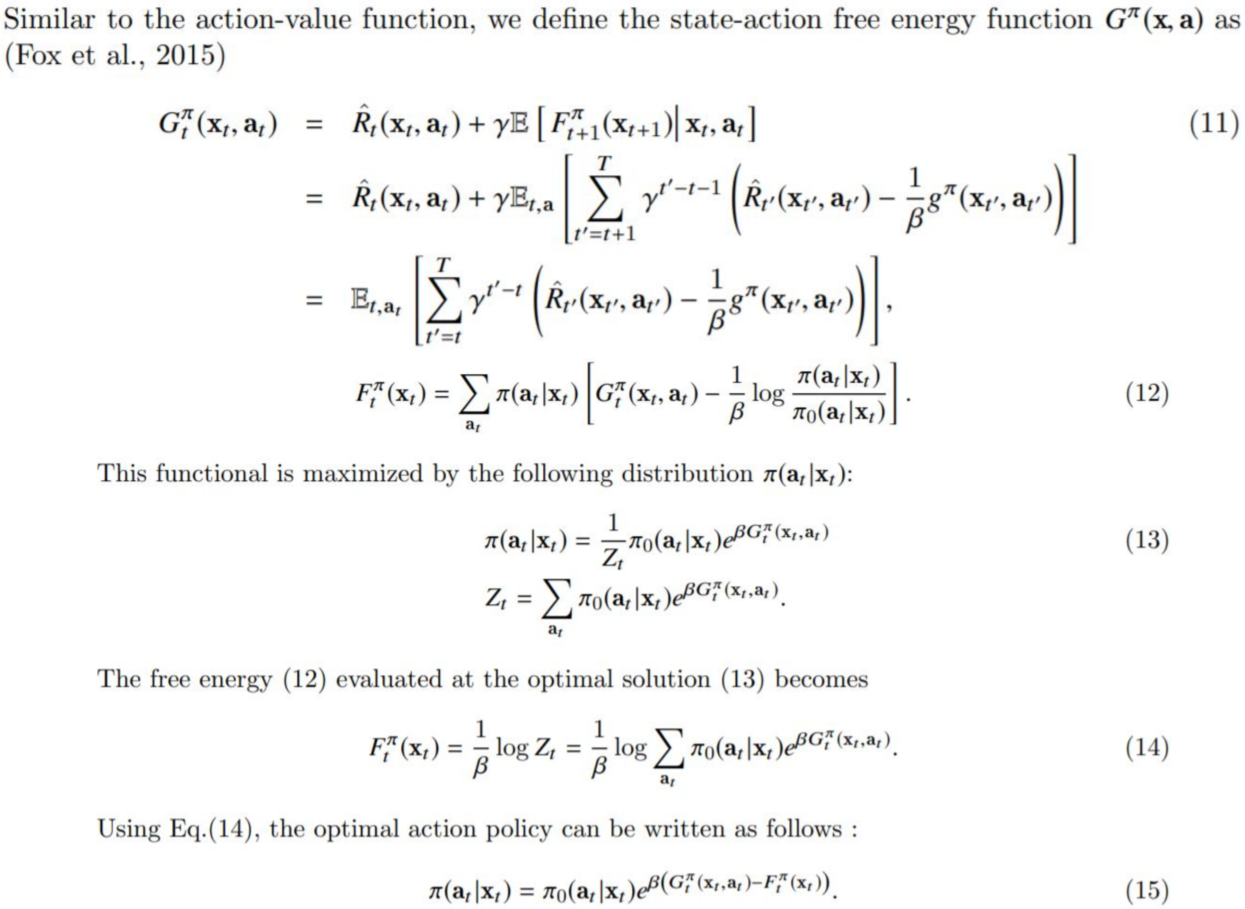
A Bellman equation for the free energy function $F_t^{\pi}(\mathbf{x}_t)$ is obtained from Eq.(8):

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

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)

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

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

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

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

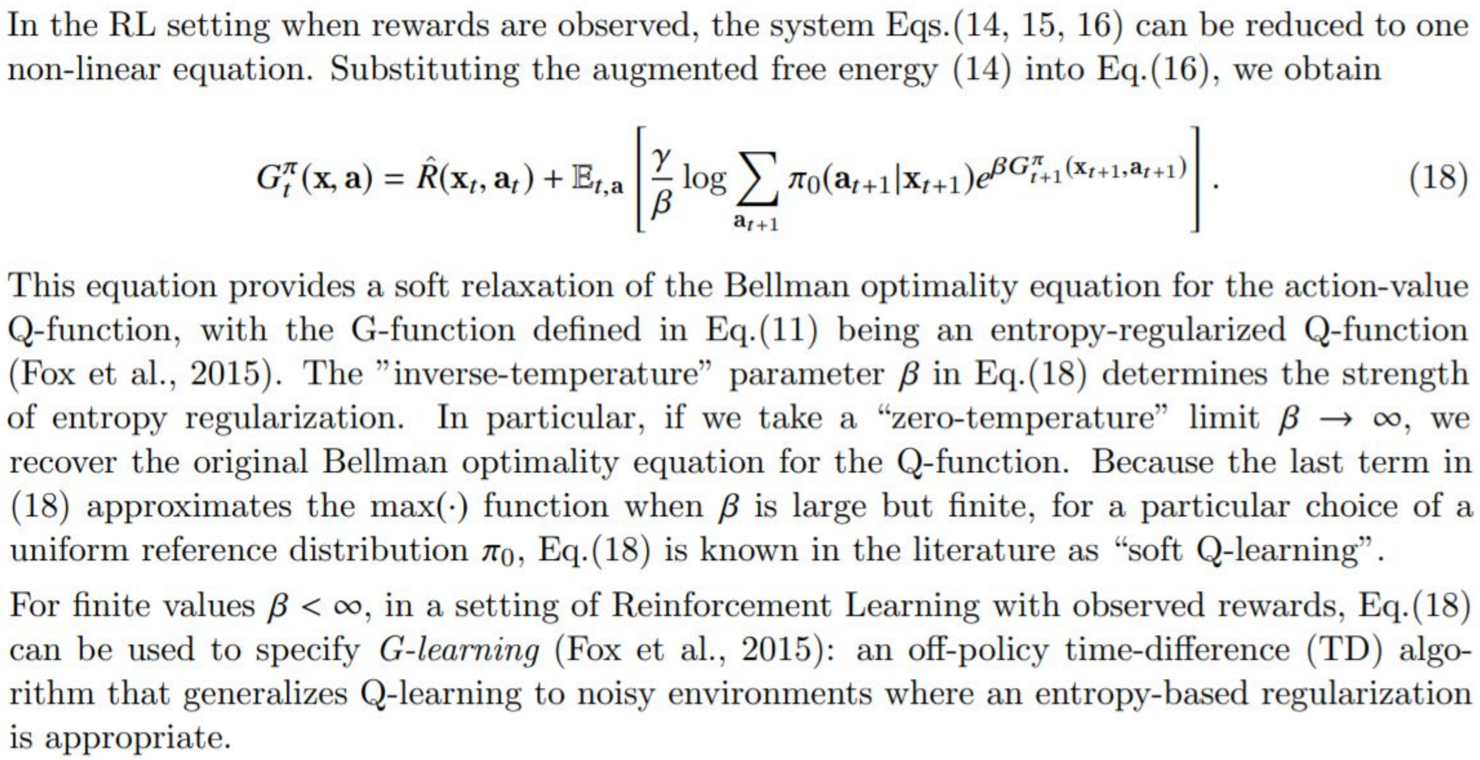
The free energy (12) evaluated at the optimal solution (13) becomes

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

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

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

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

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

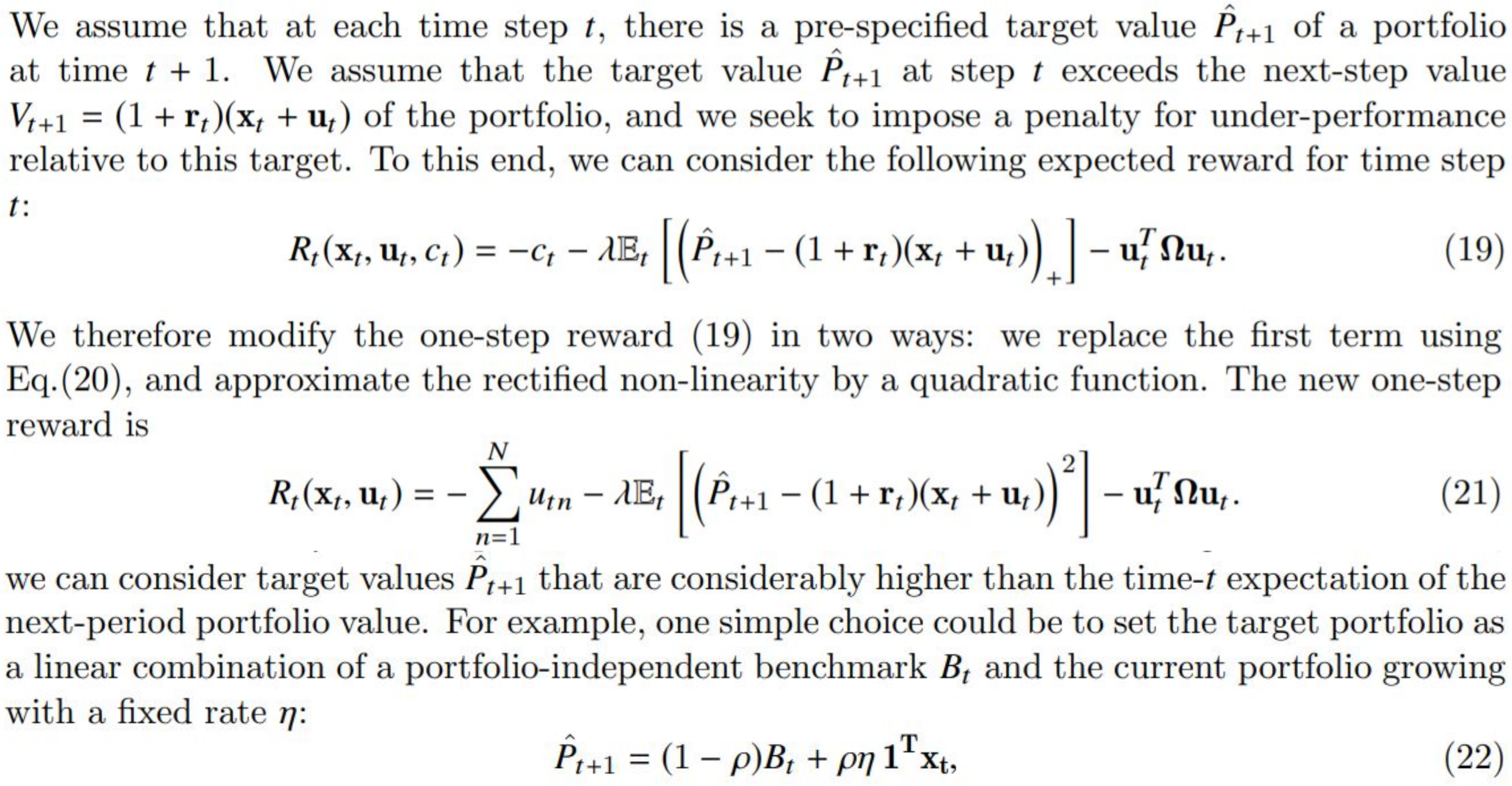
This equation provides a soft relaxation of the Bellman optimality equation for the action-value Q-function, with the G-function defined in Eq.(11) being an entropy-regularized Q-function (Fox et al., 2015). The ”inverse-temperature” parameter β in Eq.(18) determines the strength of entropy regularization. In particular, if we take a “zero-temperature” limit $\beta \rightarrow \infty$, we recover the original Bellman optimality equation for the Q-function. Because the last term in (18) approximates the max(·) 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

Case Study: Retirement Plan Optimization

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:

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

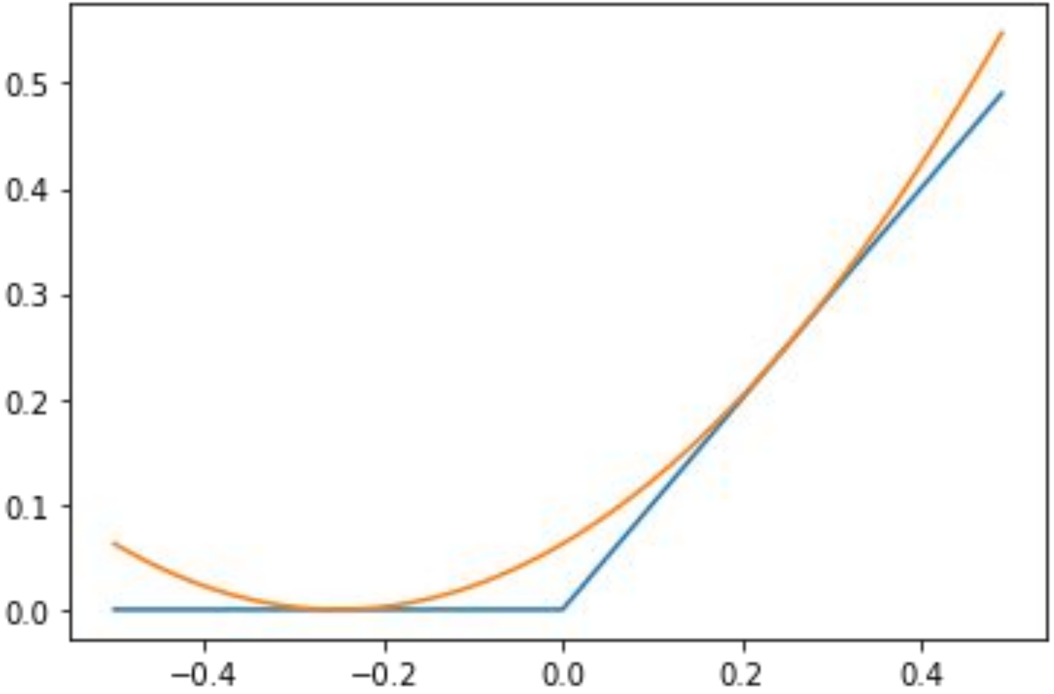
We therefore modify the one-step reward (19) in two ways: we replace the first term using Eq.(20), and approximate the rectified non-linearity by a quadratic function. The new one-step reward is

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

we can consider target values $\hat{P}_{t+1}$ that are considerably higher than the time-t expectation of the next-period portfolio value. For example, one simple choice could be to set the target portfolio as a linear combination of a portfolio-independent benchmark B_t and the current portfolio growing with a fixed rate η:

$$\hat{P}_{t+1} = (1-\rho)B_t + \rho\eta\, \mathbf{1}^{\mathbf{T}} \mathbf{x_t}, \tag{22}$$

Demonstration: Approximating the under-performance function with a shifted quadratic term



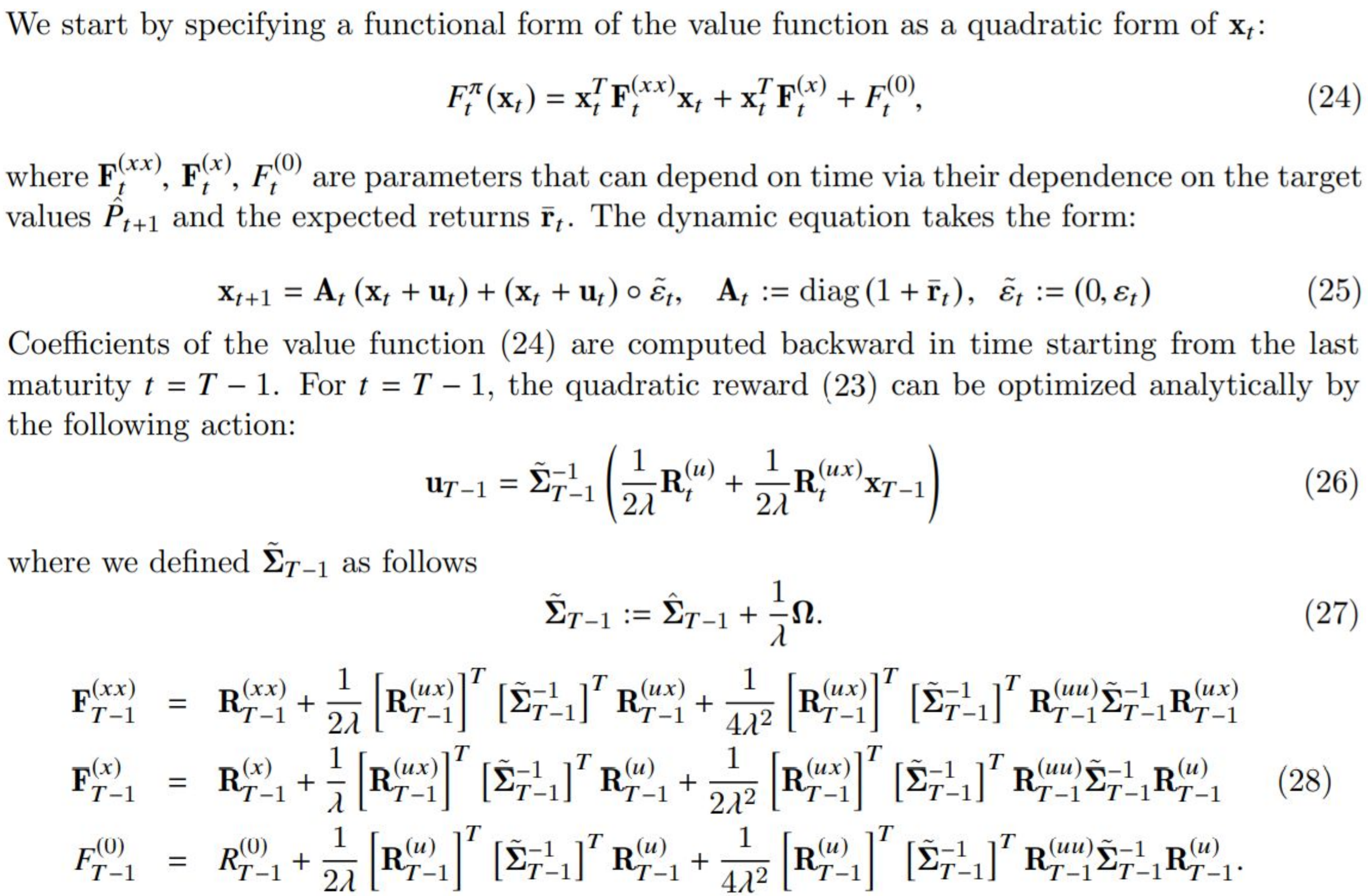
The expected reward (21) can be written in a more explicit quadratic form if we denote asset returns as $\mathbf{r}_t = \bar{\mathbf{r}}_t + \tilde{\boldsymbol{\varepsilon}}_t$ where the first component $\bar{r}_0(t) = r_f$ is the risk-free rate (as the first asset is risk-free), and $\tilde{\boldsymbol{\varepsilon}}_t = (0, \boldsymbol{\varepsilon}_t)$ where $\boldsymbol{\varepsilon}_t$ is an idiosyncratic noise with covariance $\boldsymbol{\Sigma}_r$ of size $(N-1) \times (N-1)$. Substituting this expression in Eq.(21), we obtain

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

$$\begin{aligned}
\hat{\boldsymbol{\Sigma}}_t &= \begin{bmatrix} 0 & \mathbf{0} \\ \mathbf{0} & \boldsymbol{\Sigma}_r \end{bmatrix} + (1 + \bar{\mathbf{r}}_t)(1 + \bar{\mathbf{r}}_t)^T \\
\mathbf{R}_t^{(xx)} &= -\lambda \eta^2 \rho^2 \mathbf{1}\mathbf{1}^{\mathbf{T}} + \mathbf{2}\lambda\eta\rho(\mathbf{1} + \bar{\mathbf{r}}_{\mathbf{t}})\mathbf{1}^{\mathbf{T}} - \lambda \hat{\boldsymbol{\Sigma}}_{\mathbf{t}} \\
\mathbf{R}_t^{(ux)} &= 2\lambda\eta\rho(1 + \bar{\mathbf{r}}_t)\mathbf{1}^T - 2\lambda \hat{\boldsymbol{\Sigma}}_t \\
\mathbf{R}_t^{(uu)} &= -\lambda \hat{\boldsymbol{\Sigma}}_t - \boldsymbol{\Omega} \\
\mathbf{R}_t^{(x)} &= -2\lambda\eta\rho(1 - \rho) B_t \mathbf{1} + 2\lambda(1 - \rho) B_t (1 + \bar{\mathbf{r}}_t) \\
\mathbf{R}_t^{(u)} &= -\mathbf{1} + 2\lambda(1 - \rho) B_t (1 + \bar{\mathbf{r}}_t) \\
R_t^{(0)} &= -(1 - \rho)^2 \lambda B_t^2
\end{aligned}$$

Assuming that the expected returns $\bar{\mathbf{r}}_t$, covariance matrix $\boldsymbol{\Sigma}_r$ and the benchmark B_t are fixed, the vector of free parameters defining the reward function is thus $\theta := (\lambda, \eta, \rho, \Omega)$.

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:

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

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

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

$$\begin{aligned}
\mathbf{F}_{T-1}^{(xx)} &= \mathbf{R}_{T-1}^{(xx)} + \frac{1}{2\lambda} \left[\mathbf{R}_{T-1}^{(ux)}\right]^T \left[\tilde{\mathbf{\Sigma}}_{T-1}^{-1}\right]^T \mathbf{R}_{T-1}^{(ux)} + \frac{1}{4\lambda^2} \left[\mathbf{R}_{T-1}^{(ux)}\right]^T \left[\tilde{\mathbf{\Sigma}}_{T-1}^{-1}\right]^T \mathbf{R}_{T-1}^{(uu)} \tilde{\mathbf{\Sigma}}_{T-1}^{-1} \mathbf{R}_{T-1}^{(ux)} \\
\mathbf{F}_{T-1}^{(x)} &= \mathbf{R}_{T-1}^{(x)} + \frac{1}{\lambda} \left[\mathbf{R}_{T-1}^{(ux)}\right]^T \left[\tilde{\mathbf{\Sigma}}_{T-1}^{-1}\right]^T \mathbf{R}_{T-1}^{(u)} + \frac{1}{2\lambda^2} \left[\mathbf{R}_{T-1}^{(ux)}\right]^T \left[\tilde{\mathbf{\Sigma}}_{T-1}^{-1}\right]^T \mathbf{R}_{T-1}^{(uu)} \tilde{\mathbf{\Sigma}}_{T-1}^{-1} \mathbf{R}_{T-1}^{(u)} \\
F_{T-1}^{(0)} &= R_{T-1}^{(0)} + \frac{1}{2\lambda} \left[\mathbf{R}_{T-1}^{(u)}\right]^T \left[\tilde{\mathbf{\Sigma}}_{T-1}^{-1}\right]^T \mathbf{R}_{T-1}^{(u)} + \frac{1}{4\lambda^2} \left[\mathbf{R}_{T-1}^{(u)}\right]^T \left[\tilde{\mathbf{\Sigma}}_{T-1}^{-1}\right]^T \mathbf{R}_{T-1}^{(uu)} \tilde{\mathbf{\Sigma}}_{T-1}^{-1} \mathbf{R}_{T-1}^{(u)}.
\end{aligned} \tag{28}$$

For an arbitrary time step $t = T-2, \ldots, 0$, we use Eq.(25) to compute the conditional expectation of the next-period F-function in the Bellman equation as follows:

$$\begin{aligned}\mathbb{E}_{t,\mathbf{a}}\left[F^{\pi}_{t+1}(\mathbf{x}_{t+1})\right] &= (\mathbf{x}_t + \mathbf{u}_t)^T \left(\mathbf{A}_t^T \bar{\mathbf{F}}^{(xx)}_{t+1} \mathbf{A}_t + \tilde{\boldsymbol{\Sigma}}_r \circ \bar{\mathbf{F}}^{(xx)}_{t+1}\right)(\mathbf{x}_t + \mathbf{u}_t) \\ &+ (\mathbf{x}_t + \mathbf{u}_t)^T \, \mathbf{A}_t^T \bar{\mathbf{F}}^{(x)}_{t+1} + \bar{F}^{(0)}_{t+1}, \quad \tilde{\boldsymbol{\Sigma}}_r := \begin{bmatrix} 0 & \mathbf{0} \\ \mathbf{0} & \boldsymbol{\Sigma}_r \end{bmatrix}\end{aligned} \tag{29}$$

where $\bar{\mathbf{F}}^{(xx)}_{t+1} := \mathbb{E}_t\left[\mathbf{F}^{(xx)}_{t+1}\right]$, and similarly for $\bar{\mathbf{F}}^{(x)}_{t+1}$ and $\bar{F}^{(0)}_{t+1}$. This is a quadratic function of $\mathbf{x}_t$ and $\mathbf{u}_t$, and has the same structure as the quadratic reward $\hat{R}(\mathbf{x}_t, \mathbf{a}_t)$ in Eq.(23). Plugging both expressions in the Bellman equation

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

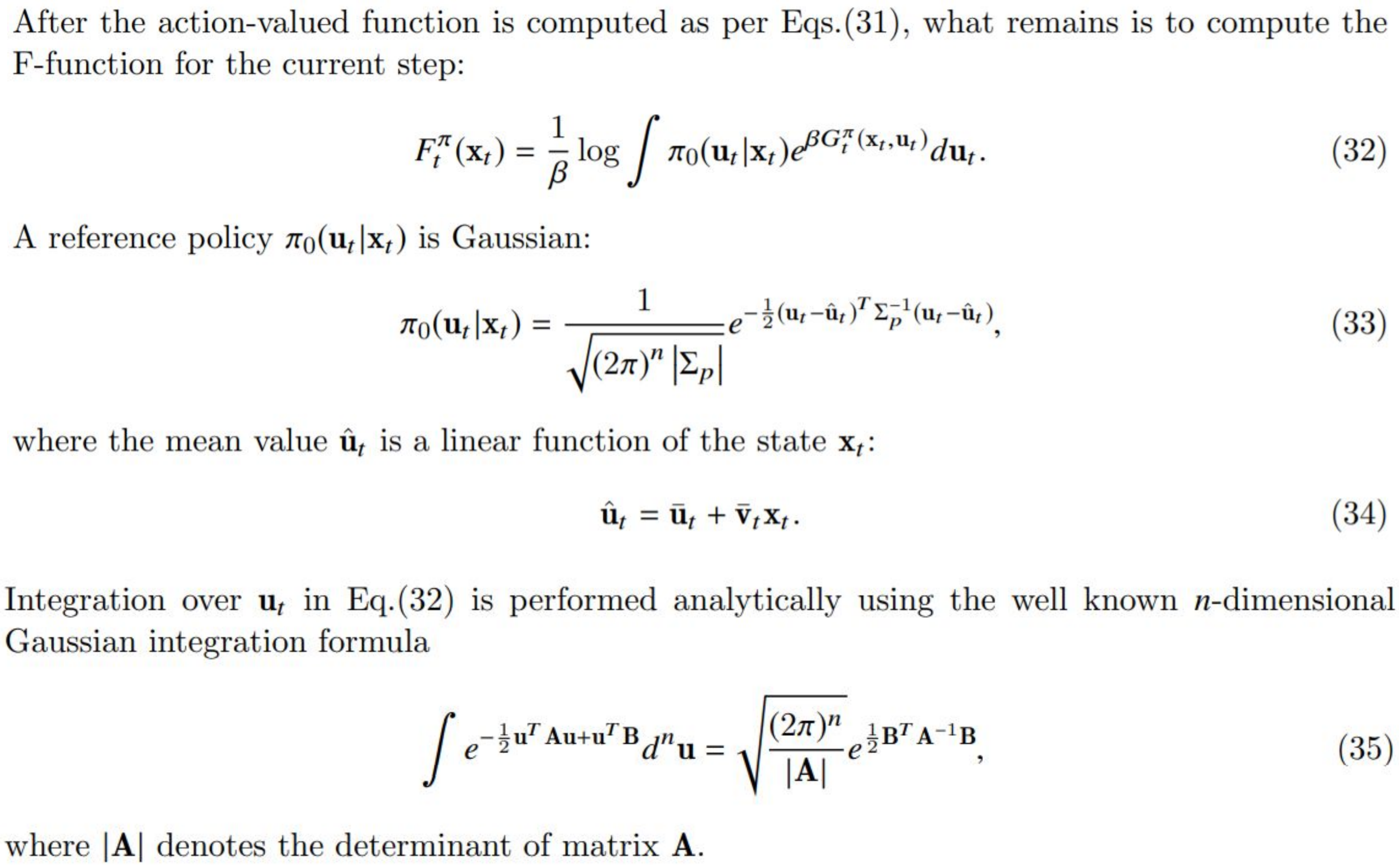
we see that the action-value function $G^{\pi}_t(\mathbf{x}_t, \mathbf{u}_t)$ should also be a quadratic function of $\mathbf{x}_t$ and $\mathbf{u}_t$:

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

where

$$\begin{aligned}\mathbf{Q}^{(xx)}_t &= \mathbf{R}^{(xx)}_t + \gamma\left(\mathbf{A}_t^T \bar{\mathbf{F}}^{(xx)}_{t+1} \mathbf{A}_t + \tilde{\boldsymbol{\Sigma}}_r \circ \bar{\mathbf{F}}^{(xx)}_{t+1}\right) \\ \mathbf{Q}^{(ux)}_t &= \mathbf{R}^{(ux)}_t + 2\gamma\left(\mathbf{A}_t^T \bar{\mathbf{F}}^{(xx)}_{t+1} \mathbf{A}_t + \tilde{\boldsymbol{\Sigma}}_r \circ \bar{\mathbf{F}}^{(xx)}_{t+1}\right) \\ \mathbf{Q}^{(uu)}_t &= \mathbf{R}^{(uu)}_t + \gamma\left(\mathbf{A}_t^T \bar{\mathbf{F}}^{(xx)}_{t+1} \mathbf{A}_t + \tilde{\boldsymbol{\Sigma}}_r \circ \bar{\mathbf{F}}^{(xx)}_{t+1}\right) - \boldsymbol{\Omega} \\ \mathbf{Q}^{(x)}_t &= \mathbf{R}^{(x)}_t + \gamma \mathbf{A}_t^T \bar{\mathbf{F}}^{(x)}_{t+1} \\ \mathbf{Q}^{(u)}_t &= \mathbf{R}^{(u)}_t + \gamma \mathbf{A}_t^T \bar{\mathbf{F}}^{(x)}_{t+1} \\ Q^{(0)}_t &= R^{(0)}_t + \gamma F^{(0)}_{t+1}.\end{aligned} \tag{31}$$

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:

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

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

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

Integration over $\mathbf{u}_t$ in Eq.(32) is performed analytically using the well known n-dimensional Gaussian integration formula

$$\int e^{-\frac{1}{2}\mathbf{u}^T \mathbf{A} \mathbf{u} + \mathbf{u}^T \mathbf{B}} d^n \mathbf{u} = \sqrt{\frac{(2\pi)^n}{|\mathbf{A}|}} e^{\frac{1}{2}\mathbf{B}^T \mathbf{A}^{-1} \mathbf{B}}, \tag{35}$$

where $|\mathbf{A}|$ denotes the determinant of matrix $\mathbf{A}$.

Performing the Gaussian integration and comparing the resulting expression with Eq.(24), we obtain for its coefficients:

$$\begin{aligned}
F_t^{\pi}(\mathbf{x}_t) &= \mathbf{x}_t^T \mathbf{F}_t^{(xx)} \mathbf{x}_t + \mathbf{x}_t^T \mathbf{F}_t^{(x)} + F_t^{(0)} \\
\mathbf{F}_t^{(xx)} &= \mathbf{Q}_t^{(xx)} + \frac{1}{2\beta}\left(\mathbf{U}_t^T \bar{\boldsymbol{\Sigma}}_p^{-1} \mathbf{U}_t - \bar{\mathbf{v}}_t^T \boldsymbol{\Sigma}_p^{-1} \bar{\mathbf{v}}_t \right) \\
\mathbf{F}_t^{(x)} &= \mathbf{Q}_t^{(x)} + \frac{1}{\beta}\left(\mathbf{U}_t^T \bar{\boldsymbol{\Sigma}}_p^{-1} \mathbf{W}_t - \bar{\mathbf{v}}_t^T \boldsymbol{\Sigma}_p^{-1} \bar{\mathbf{u}}_t \right) \\
\mathbf{F}_t^{(0)} &= \mathbf{Q}_t^{(0)} + \frac{1}{2\beta}\left(\mathbf{W}_t^T \bar{\boldsymbol{\Sigma}}_p^{-1} \mathbf{W}_t - \bar{\mathbf{u}}_t^T \boldsymbol{\Sigma}_p^{-1} \bar{\mathbf{u}}_t \right) - \frac{1}{2\beta}\left(\log\left|\boldsymbol{\Sigma}_p\right| + \log\left|\bar{\boldsymbol{\Sigma}}_p\right| \right),
\end{aligned} \tag{36}$$

where we use the auxiliary parameters

$$\begin{aligned}
\mathbf{U}_t &= \beta \mathbf{Q}_t^{(ux)} + \Sigma_p^{-1} \bar{\mathbf{v}}_t \\
\mathbf{W}_t &= \beta \mathbf{Q}_t^{(u)} + \Sigma_p^{-1} \bar{\mathbf{u}}_t \\
\bar{\boldsymbol{\Sigma}}_p &= \boldsymbol{\Sigma}_p^{-1} - 2\beta \mathbf{Q}_t^{(uu)}.
\end{aligned} \tag{37}$$

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 $\boldsymbol{\pi}(\mathbf{u}_t|\mathbf{x}_t)$:

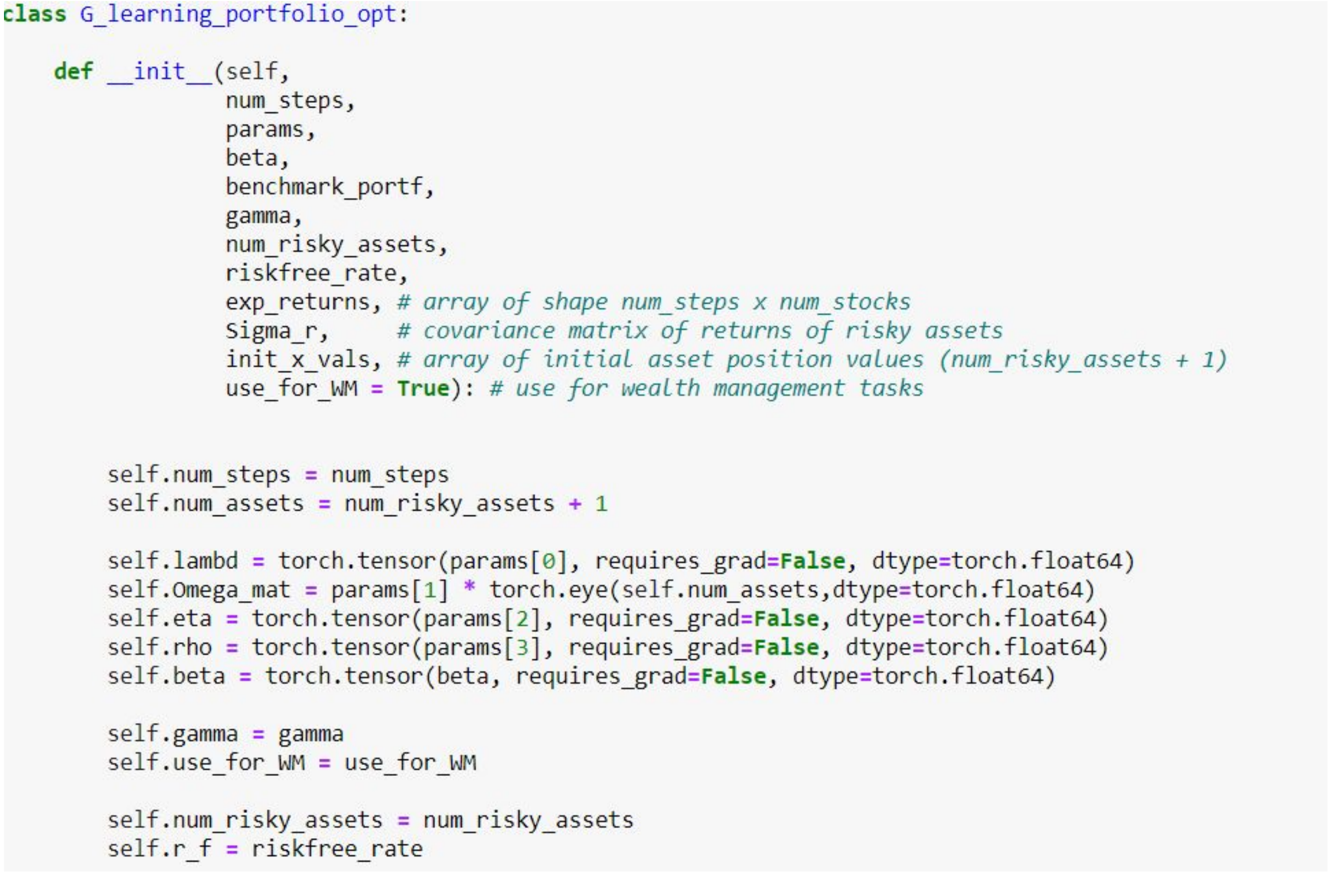
$$\pi(\mathbf{u}_t|\mathbf{x}_t) = \frac{1}{\sqrt{(2\pi)^n \left|\tilde{\Sigma}_p\right|}} e^{-\frac{1}{2}(\mathbf{u}_t - \tilde{\mathbf{u}}_t - \tilde{\mathbf{v}}_t \mathbf{x}_t)^T \tilde{\Sigma}_p^{-1} (\mathbf{u}_t - \hat{\mathbf{u}}_t - \tilde{\mathbf{v}}_t \mathbf{x}_t)} \tag{39}$$

where

$$\begin{aligned} \tilde{\Sigma}_p^{-1} &= \boldsymbol{\Sigma}_p^{-1} - 2\beta \mathbf{Q}_t^{(uu)} \\ \tilde{\mathbf{u}}_t &= \tilde{\Sigma}_p \left(\boldsymbol{\Sigma}_p^{-1} \bar{\mathbf{u}}_t + \beta \mathbf{Q}_t^{(u)} \right) \\ \tilde{\mathbf{v}}_t &= \tilde{\Sigma}_p \left(\boldsymbol{\Sigma}_p^{-1} \bar{\mathbf{v}}_t + \beta \mathbf{Q}_t^{(ux)} \right) \end{aligned} \tag{40}$$

Therefore, policy optimization for G-learning with quadratic rewards and Gaussian reference policy amounts to the Bayesian update of the prior distribution (33) with parameters updates $\bar{\mathbf{u}}_t$, $\bar{\mathbf{v}}_t$, $\boldsymbol{\Sigma}_p$ to the new values $\tilde{\mathbf{u}}_t$, $\tilde{\mathbf{v}}_t$, $\tilde{\Sigma}_p$ defined in Eqs.(40). These quantities depend on time via their dependence on the targets $\hat{P}_t$ and expected asset returns $\bar{\mathbf{r}}_t$.

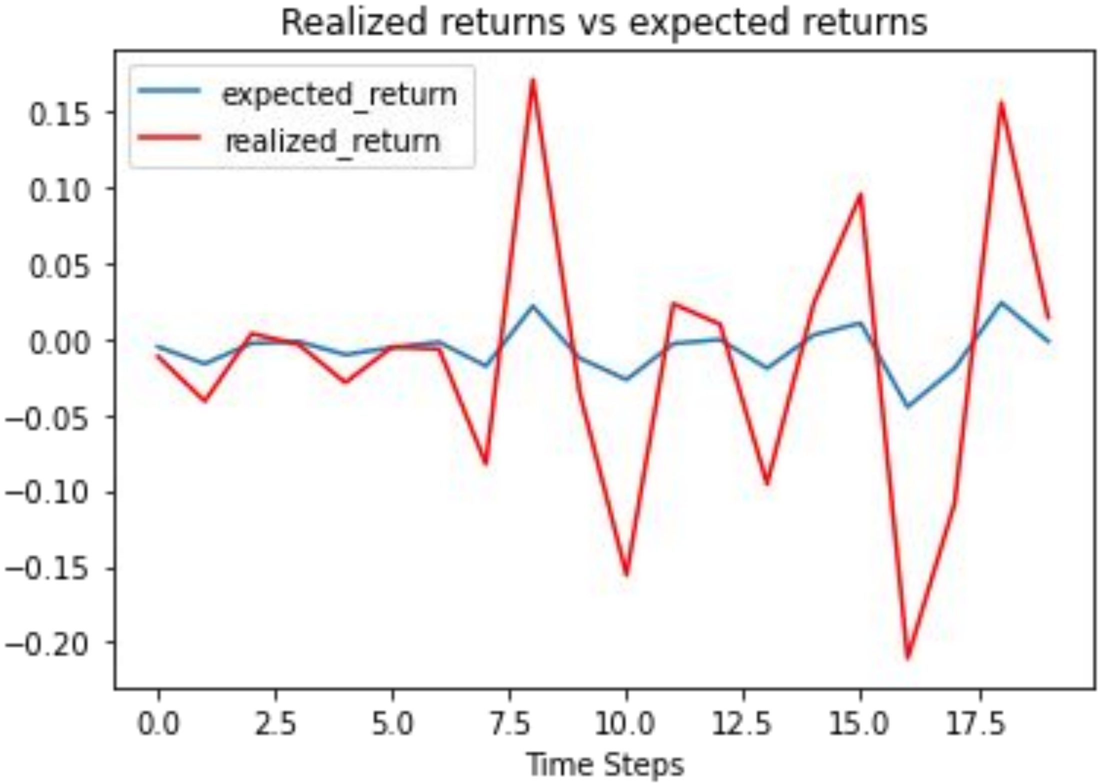
G-Learning Code Snippet:



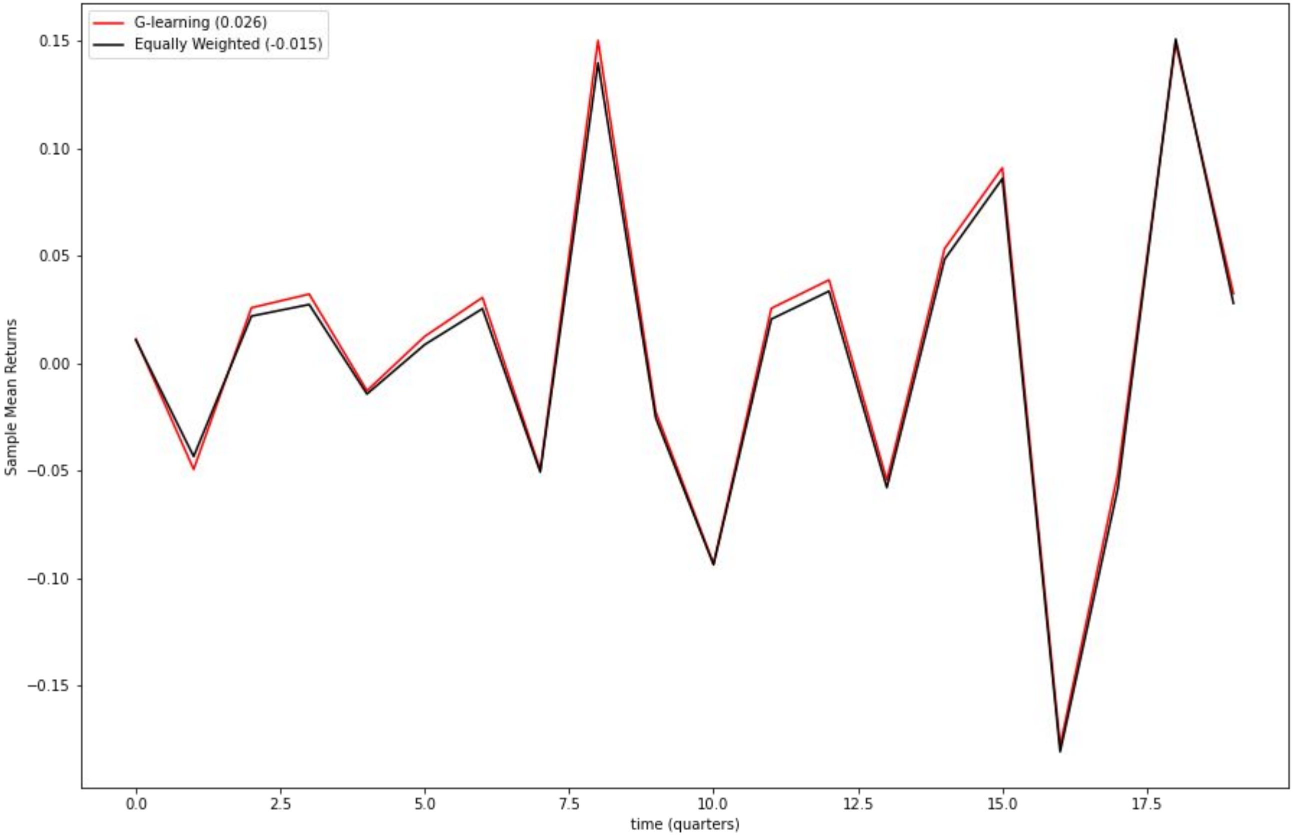
```
class G_learning_portfolio_opt:

    def __init__(self,
                 num_steps,
                 params,
                 beta,
                 benchmark_portf,
                 gamma,
                 num_risky_assets,
                 riskfree_rate,
                 exp_returns, # array of shape num_steps x num_stocks
                 Sigma_r,     # covariance matrix of returns of risky assets
                 init_x_vals, # array of initial asset position values (num_risky_assets + 1)
                 use_for_WM = True): # use for wealth management tasks


        self.num_steps = num_steps
        self.num_assets = num_risky_assets + 1

        self.lambd = torch.tensor(params[0], requires_grad=False, dtype=torch.float64)
        self.Omega_mat = params[1] * torch.eye(self.num_assets,dtype=torch.float64)
        self.eta = torch.tensor(params[2], requires_grad=False, dtype=torch.float64)
        self.rho = torch.tensor(params[3], requires_grad=False, dtype=torch.float64)
        self.beta = torch.tensor(beta, requires_grad=False, dtype=torch.float64)

        self.gamma = gamma
        self.use_for_WM = use_for_WM

        self.num_risky_assets = num_risky_assets
        self.r_f = riskfree_rate
```

Simulated market data using a factor model with market betas and idiosyncratic alphas:

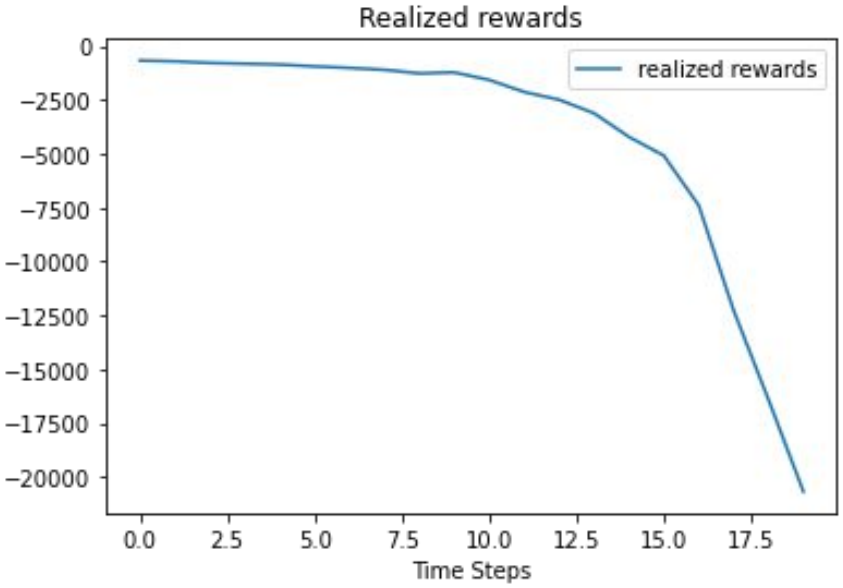
1. High Volatility Case

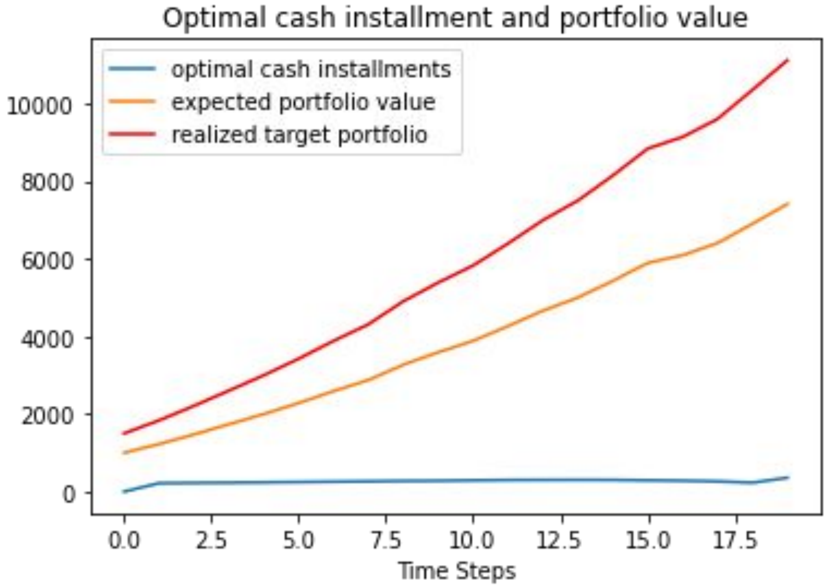


G-Learning performance on simulated data:

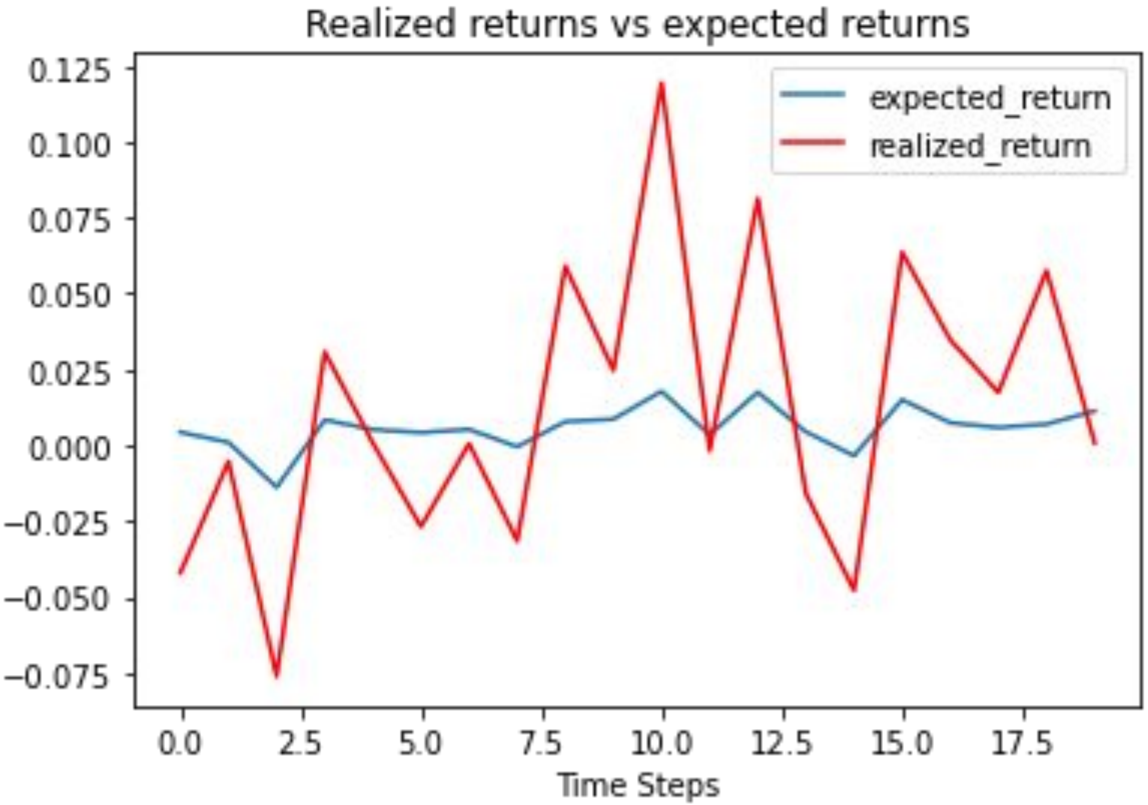


Cash Installments and Portfolio Value

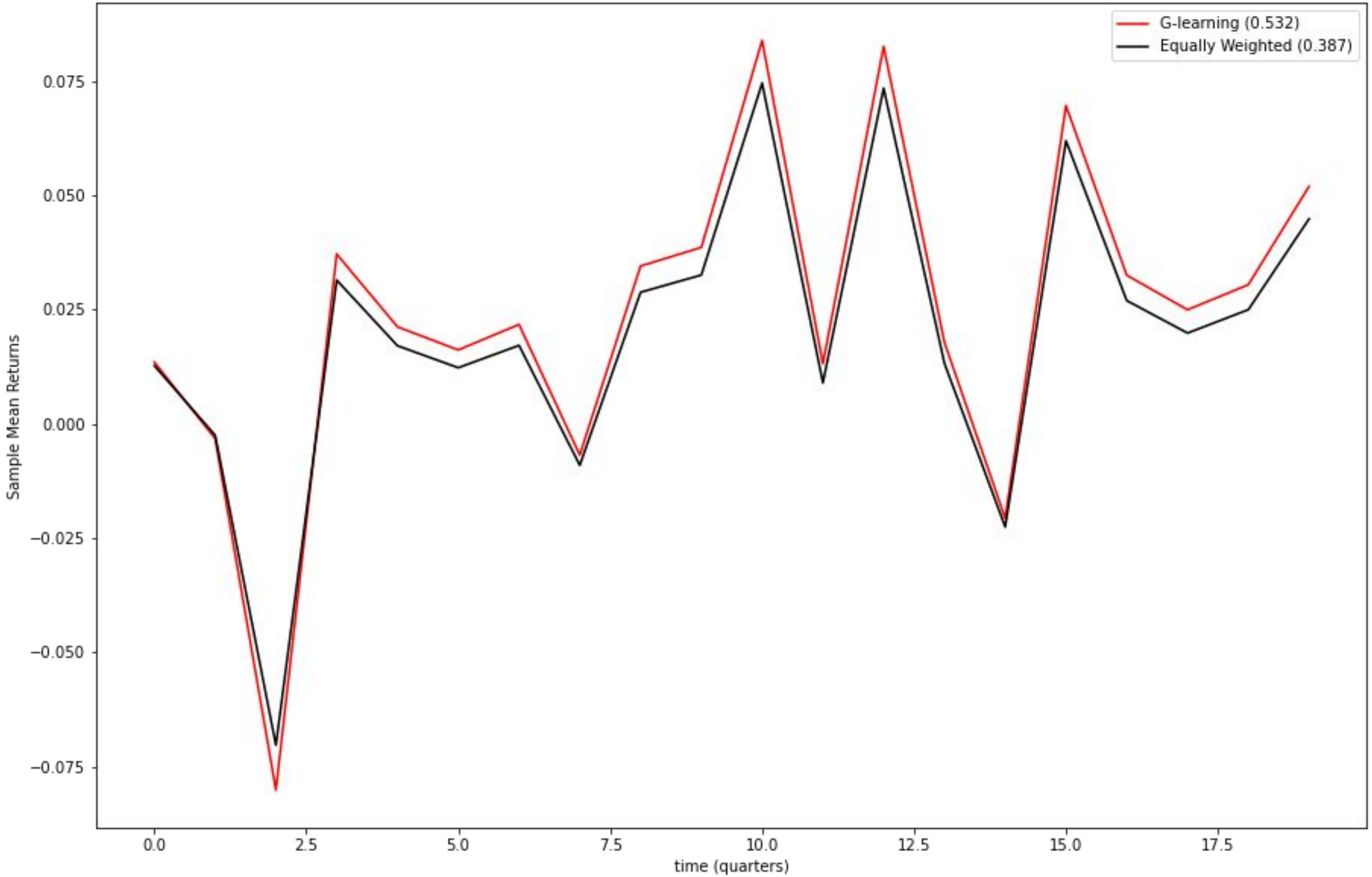




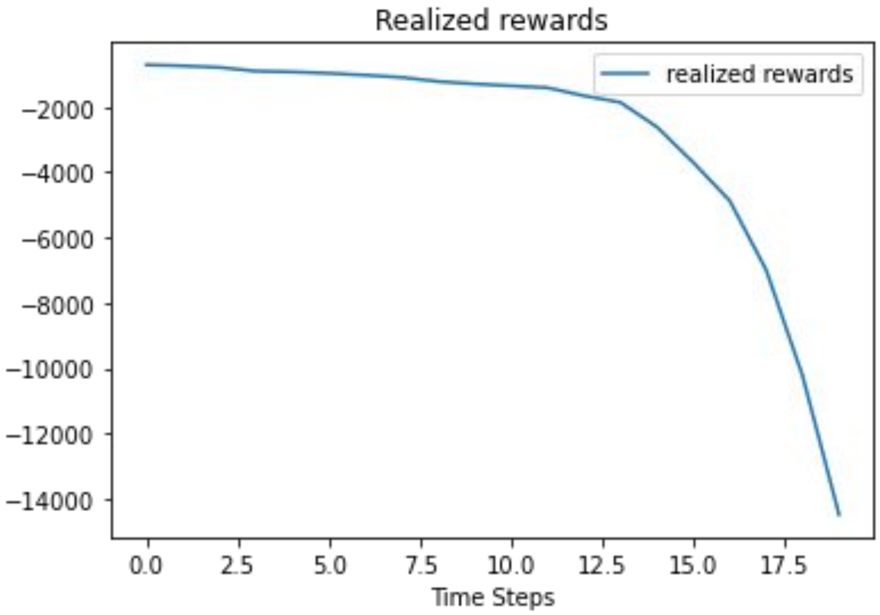
2. Low Volatility Case

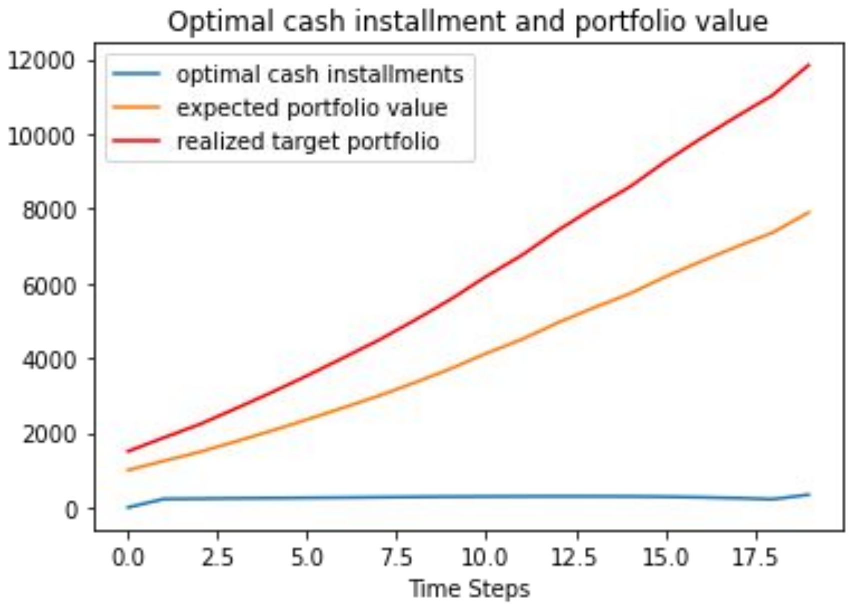


G-Learning performance on simulated data:



Cash Installments and Portfolio Value

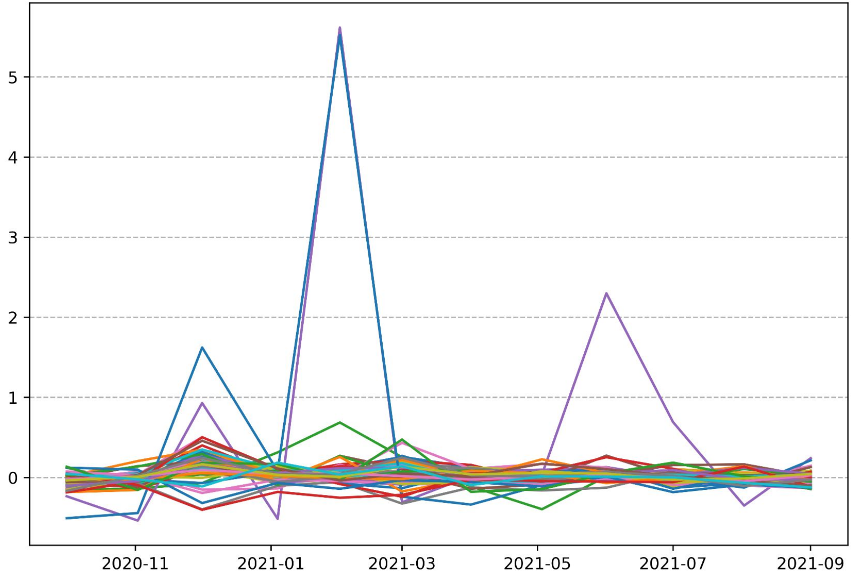




3. Using Real Market Data:

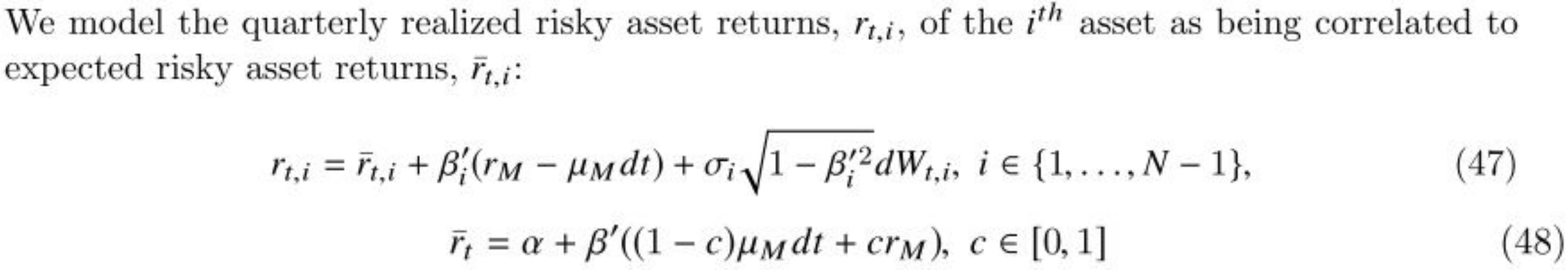
To begin with, we consider the tickers provided in the EOD metadata, there are roughly 18000 of them. The daily open, high, low, close prices and volume can be extracted from the complete dataset.

We select the 50 most liquid stocks (highest average volume) as risky assets and assume an additional risk-free asset. We update our portfolio in monthly basis with the 1 year rolling window.



Difference with simulation

In simulation, all returns are obtained by one-factor model



We model the quarterly realized risky asset returns, $r_{t,i}$, of the i^{th} asset as being correlated to expected risky asset returns, $\bar{r}_{t,i}$:

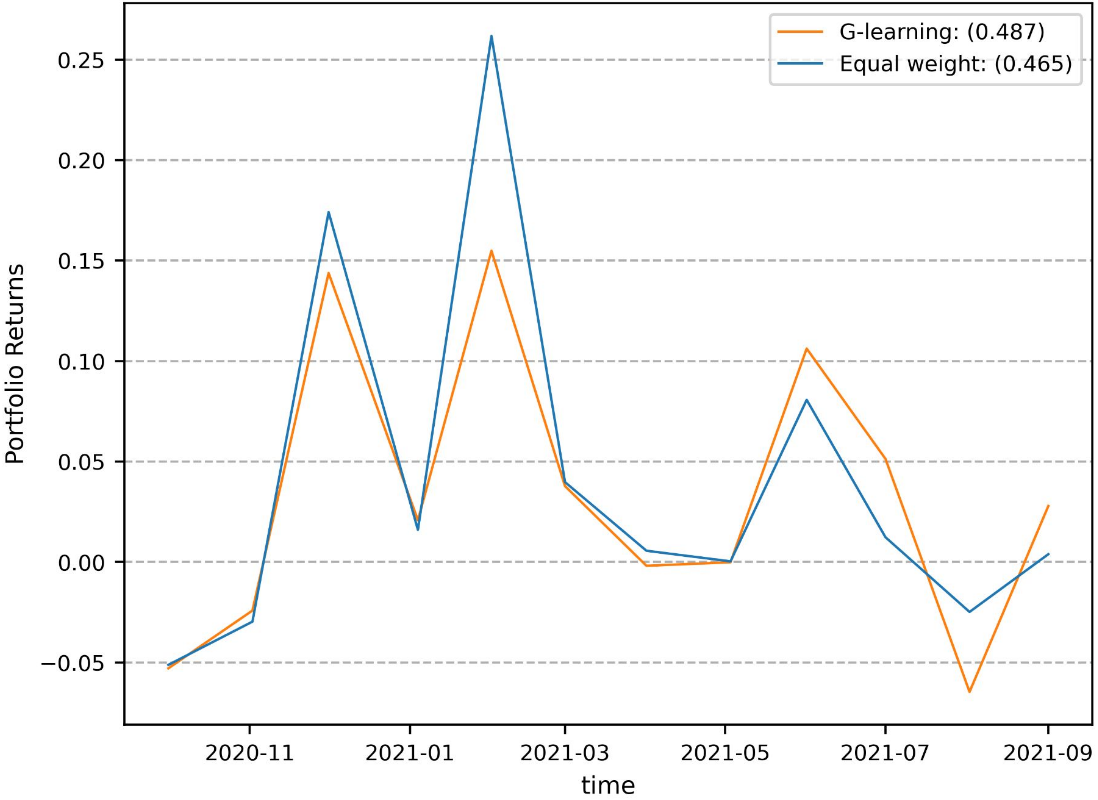
$$r_{t,i} = \bar{r}_{t,i} + \beta_i'(r_M - \mu_M dt) + \sigma_i \sqrt{1 - \beta_i'^2} dW_{t,i}, \;\; i \in \{1, \ldots, N-1\}, \tag{47}$$

$$\bar{r}_t = \alpha + \beta'((1-c)\mu_M dt + c r_M), \;\; c \in [0, 1] \tag{48}$$

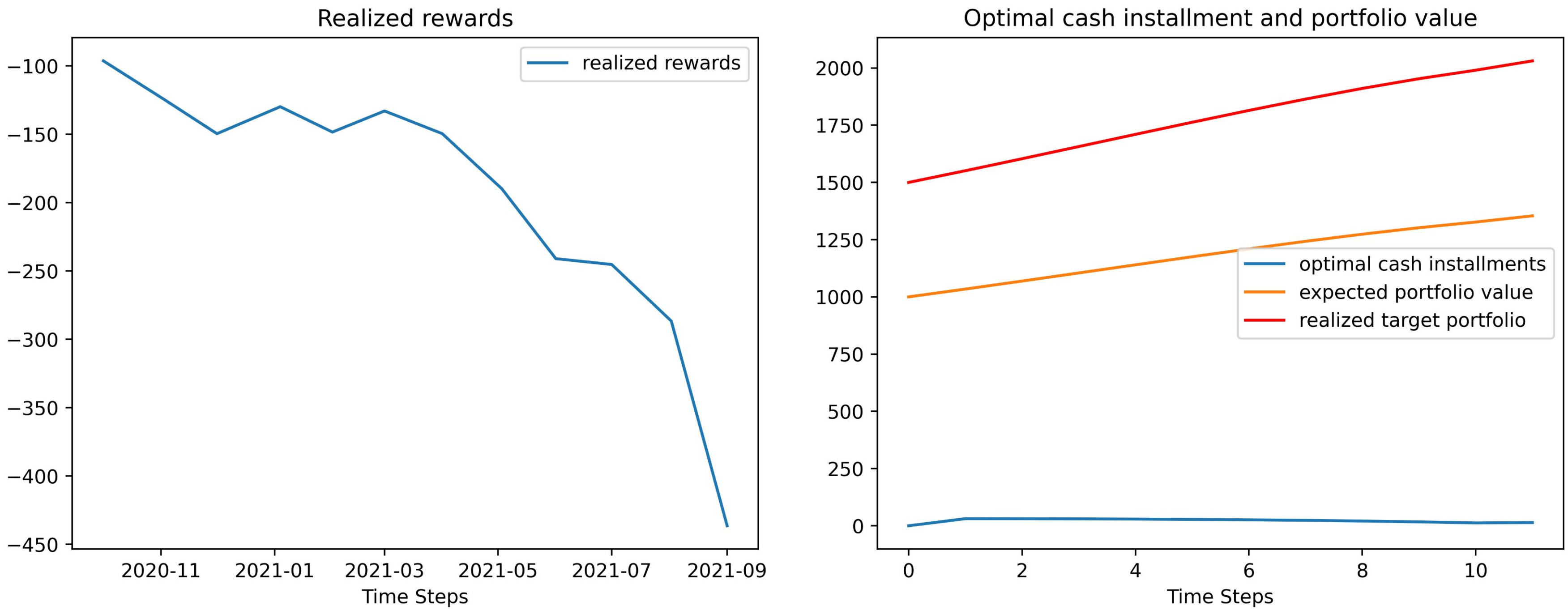
The RL algorithm assume that the expected asset returns are available as model parameters. The present model formalism is agnostic to the choice of the expected return model. In simulation case, we know that the expected returns we obtained are “correct”. However, in reality, it is a further task to have an accurate expected return model.

At this stage, we use 36-months rolling mean returns as expected returns. The G-learn algorithm is also executed in the rolling basis. At each time step t, the G-learner is trained one time with new expected return matrix, and we record the updated parameters at time t.

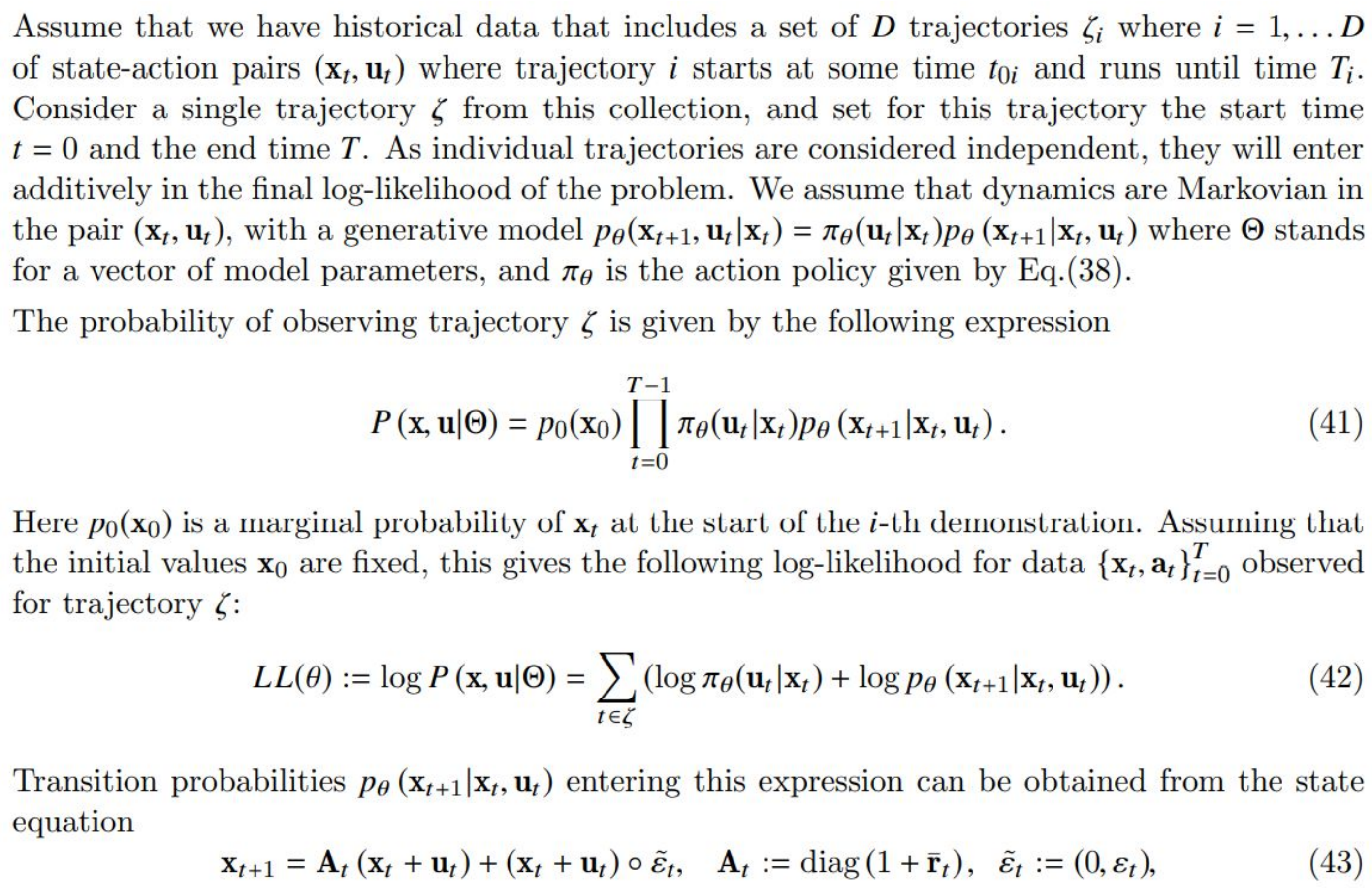
G-Learning performance on actual data:



Cash Installments and Portfolio Value



GIRL: G-Learner Inverse Reinforcement Learning



Assume that we have historical data that includes a set of D trajectories ζ_i where $i = 1, \ldots D$ 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. Consider a single trajectory ζ from this collection, and set for this trajectory the start time $t = 0$ and the end time T. As individual trajectories are considered independent, they will enter additively in the final log-likelihood of the problem. We assume that dynamics are Markovian in the pair $(\mathbf{x}_t, \mathbf{u}_t)$, with a generative model $p_\theta(\mathbf{x}_{t+1}, \mathbf{u}_t | \mathbf{x}_t) = \pi_\theta(\mathbf{u}_t | \mathbf{x}_t) p_\theta(\mathbf{x}_{t+1} | \mathbf{x}_t, \mathbf{u}_t)$ where $\mathbf{\Theta}$ stands for a vector of model parameters, and π_θ is the action policy given by Eq.(38).

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

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

Here $p_0(\mathbf{x}_0)$ is a marginal probability of $\mathbf{x}_t$ at the start of the i-th demonstration. Assuming that the initial values $\mathbf{x}_0$ are fixed, this gives the following log-likelihood for data $\{\mathbf{x}_t, \mathbf{a}_t\}_{t=0}^{T}$ observed for trajectory ζ:

$$LL(\theta) := \log P(\mathbf{x}, \mathbf{u} | \Theta) = \sum_{t \in \zeta} \left(\log \pi_\theta(\mathbf{u}_t | \mathbf{x}_t) + \log p_\theta(\mathbf{x}_{t+1} | \mathbf{x}_t, \mathbf{u}_t) \right) . \tag{42}$$

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

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

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

$$\log p_\theta\left(\mathbf{x}_{t+1}|\mathbf{x}_t,\mathbf{u}_t\right)=-\frac{1}{2}\log\left|\boldsymbol{\Sigma}_r\right|-\frac{1}{2}\boldsymbol{\Delta}_t^T\boldsymbol{\Sigma}_r^{-1}\boldsymbol{\Delta}_t. \qquad (45)$$

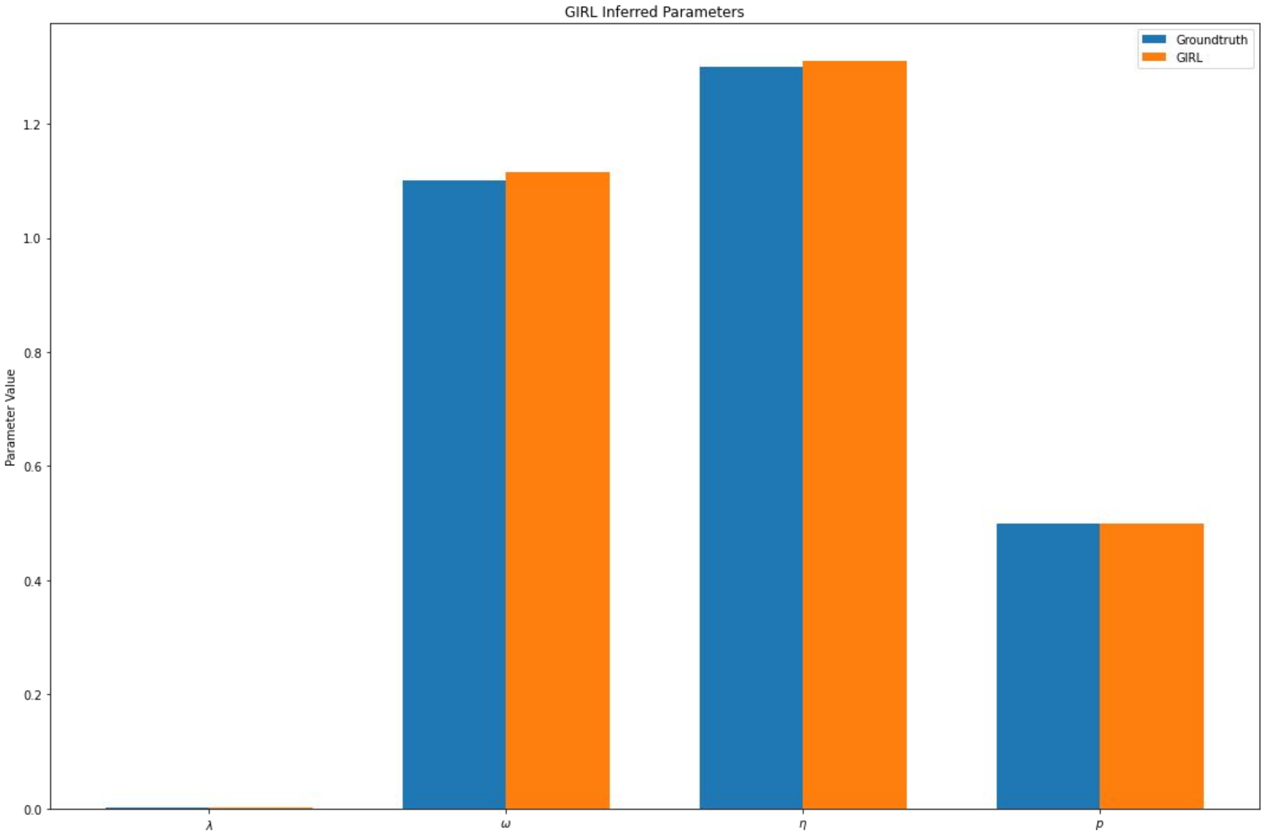
Substituting Eqs.(38), (30), (45) into the trajectory log-likelihood (42), we put it in the following form:

$$LL(\theta)=\sum_{t\in\zeta}\left(\beta\left(G_t^\pi(\mathbf{x}_t,\mathbf{u}_t)-F_t^\pi(\mathbf{x}_t)\right)-\frac{1}{2}\log\left|\boldsymbol{\Sigma}_r\right|-\frac{1}{2}\boldsymbol{\Delta}_t^T\boldsymbol{\Sigma}_r^{-1}\boldsymbol{\Delta}_t\right), \qquad (46)$$

where $G_t^\pi(\mathbf{x}_t,\mathbf{u}_t)$ and $F_t^\pi(\mathbf{x}_t)$ are defined by Eqs.(30) and (24). The log-likelihood (46) is a function of model parameter vector $\theta=\left(\lambda,\eta,\rho,\boldsymbol{\Omega},\boldsymbol{\Sigma}_r,\boldsymbol{\Sigma}_p,\bar{\mathbf{u}}_t,\bar{\mathbf{v}}_t\right)$ (recall that β is a regularization hyper-parameter which should not be optimized in-sample). We can simplify the problem by setting $\bar{\mathbf{v}}_t=0$ and $\bar{\mathbf{u}}_t=\bar{\mathbf{u}}$ (i.e. take a constant mean in the prior). In this case, the vector of model parameter to learn with IRL inference is $\theta=\left(\lambda,\eta,\rho,\boldsymbol{\Omega},\boldsymbol{\Sigma}_r,\boldsymbol{\Sigma}_p,\bar{\mathbf{u}}\right)$. A "proper" IRL setting would correspond to only learning parameters of the reward function $(\lambda,\eta,\rho,\boldsymbol{\Omega})$ while keeping parameters $\left(\boldsymbol{\Sigma}_r,\boldsymbol{\Sigma}_p,\bar{\mathbf{u}}\right)$ fixed (i.e. estimated outside of the IRL model). Optimization can be performed using available off-the-shelf software. In our implementation, we use the Adam optimization method within PyTorch to optimize the negative log-likelihood function.

GIRL performance on simulated data (Nelder-Mead):

lambda = 0.002, omega = 1.1, eta = 1.3, rho = 0.5



Thank You!