In this week, we will:

- Study batch-model Reinforcement Learning
- Introduce Q-Learning, a very famous algorithm of Reinforcement Learning
- Discuss Fitted Q Iteration, a practical version of Q-Learning algorithm

DP Approach to Option Model

Two step procedure:

- Estimate the model
- Simulate paths of stock price
- Find the optimal hedge and price by DP backward recursion

Need a pre-specified model of the world!
Can we do differently?

- RL solves the same problem as DP, i.e. it finds an optimal policy.
- Unlike DP, RL does not assume that transition probabilities and reward function are known.
- Instead, RL relies on samples to find an optimal policy.
- RL implements Vapnik's principle: it focuses on the target problem!

Batch-mode RL

Data: N trajectories, the information set

$$\mathcal{F}_t = \left\{ \mathcal{F}_t^{(n)} \right\}_{n=1}^{N}.$$

For each trajectory n, $\mathcal{F}_t^{(n)}$ contains:

- The stock price S_t (expressed as a function of X_t)
- The hedge position a_t
- Instantaneous reward R_t
- The next-time value X_{t+1}:

$$\mathcal{F}_t^{(n)} = \left\{ \left(X_t^{(n)}, a_t^{(n)}, R_t^{(n)}, X_{t+1}^{(n)} \right) \right\}_{t=0}^{T-1} \qquad \text{Eq: 1}$$

Approximating the Bellman

When the model is **unknown:**

We try to approximately solve the Bellman optimality equation

$$Q_t^\star(x,a) = \mathbb{E}_t\left[R_t + \gamma \max_{a\in\mathcal{A}} Q_{t+1}^\star\left(X_{t+1}, a\right) \middle| x, a\right]$$

by replacing expectations entering this equation by their empirical averages.

But what about on-line learning, when there are no fixed empirical averages?

Stochastic Approximations

The Robbins-Monro algorithm estimates the mean without directly summing the samples, but instead adding data points one by one, and iteratively updating the running estimation of the mean $\hat{x}_k$ (where k is the number of iteration, or the number of data points in a dataset):

$$\hat{x}_{k+1} = (1-\alpha_k)\hat{x}_k + \alpha_k x_k \qquad \text{Eq: 2}$$

where x_k is the k-th data point, and $\alpha_k < 1$ denotes the step size **(learning rate).**

Constraints on Learning Rate

Robbins and Monro showed that the iterative method of computing the mean (Eq: 2) converges to a true mean with probability one, provided that

$$\lim_{K\to\infty}\sum_{k=1}^{K}\alpha_k = \infty\,, \quad \lim_{K\to\infty}\sum_{k=1}^{K}\alpha_k^2 < \infty \qquad \text{Eq: 3}$$

- Can use simple parametric schedules, e.g. $\alpha_k = \frac{a}{k}$.
- The optimal choice of a learning rate α_k is not universal but specific to a problem.
- The Robbins-Monro algorithm (Eq: 2) is an on-line algorithm.
- Alternatively, stochastic approximations can be used to pick a chunk of data.

Q-Learning

- Assume a discrete-state/discrete-action version of the MDP model, as the Q-Learning in its original form (Watkins 1989) applies only for such setting
- The Q-function is represented in a tabulated form
- Q-Learning converges to the true optimal action-value function with probability one, given enough data (Watkins 1989)

The classical Q-Learning method implements a single-step update of the Q-function (Q-value) for each combination of a state and action met in data

Q-Learning: Example

Tabulated representation for Q-function:

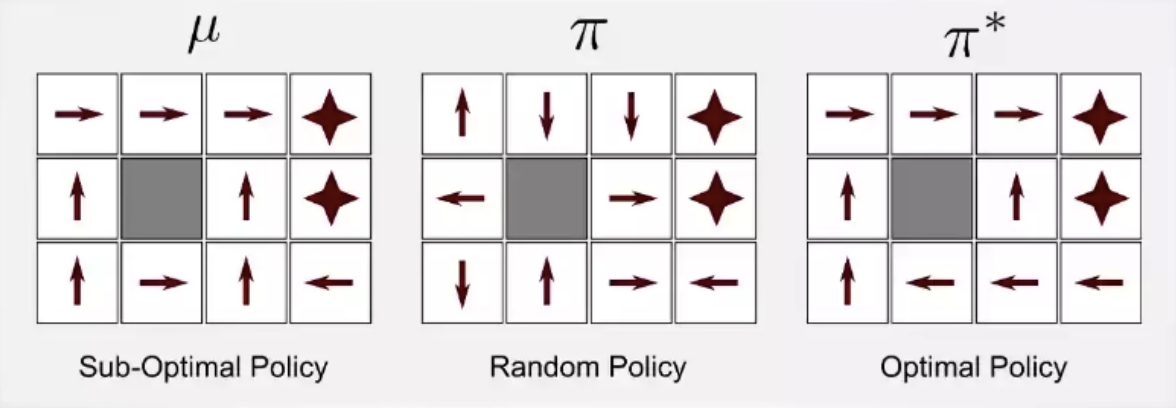


Figure: A maze problem. Source: https://mpatacchiola.github.io/blog/2017/01/29/dissecting-reinforcement-learning-3.html

Q-Learning: how it works

Q-Learning is obtained by using the Robbins-Monro stochastic approximation to estimate the unknown expectation in the Bellman optimality equation

$$Q_t^\star(x,a) = \mathbb{E}_t\left[R_t + \gamma \max_{a\in\mathcal{A}} Q_{t+1}^\star(X_{t+1},a)\,|\,x,a\right]$$

The Bellman optimality equation:

$$Q_t^\star(x,a) = \mathbb{E}_t\left[R_t + \gamma \max_{a\in\mathcal{A}} Q_{t+1}^\star(X_{t+1},a)\,|\,x,a\right]$$

Robbins-Monro:

$$\hat{x}_{k+1} = (1-\alpha_k)\hat{x}_k + \alpha_k x_k$$

The Bellman optimality equation:

$$Q_t^\star(x,a) = \mathbb{E}_t\left[R_t + \gamma \max_{a\in\mathcal{A}} Q_{t+1}^\star\left(X_{t+1}, a\right) \middle| x, a\right]$$

Robbins-Monro:

$$\hat{x}_{k+1} = (1-\alpha_k)\hat{x}_k + \alpha_k x_k$$

How to use Robbins-Monro to get Q-Learning :

$x_k \to R_t + \gamma \max_{a\in\mathcal{A}} Q_{t+1,k}^\star\left(X_{t+1}, a\right)$

$\hat{x}_k \to Q_{tk}^\star(x,a)$

This produces

$$Q_{t,k+1}^\star(X_t,a_t) = (1-\alpha_k)Q_{tk}^\star(X_t,a_t) +$$
$$\alpha_k\left[R_t\left(X_t,a_t,X_{t+1}\right) + \gamma \max_{a\in\mathcal{A}} Q_{t+1,k}^\star\left(X_{t+1},a\right)\right]$$

Fitted Q Iteration

- We need to look at all historical (or simulated) paths of the stock simultaneously to find optimal policy.
- The classical Q-Learning takes one observation at a time - can produce slow convergence with high noise level.
- Need an extension of Q-Learning for such batch-mode Reinforcement Learning.
- Enter Fitted Q Iteration (FQI).
- FQI: Ernest et. al. (2005) - time-independent case.
  Murphy (2005) - Q-Learning with a finite time horizon.
- FQI can work with either continuous

FQI: Basis Function Expansions

We use the same set of basis functions $\{\Phi_n(x)\}$ as we used in the DP solution.

As optimal Q-function $Q_t^\star(X_t, a_t)$ is a quadratic function of a_t, we can write it as an expansion in basis functions, with time-dependent coefficient matrix $\mathbf{W}_t$:

$$Q_t^\star(X_t, a_t) = \mathbf{A}_t^T \mathbf{W}_t \mathbf{\Phi}(X_t) \qquad \text{Eq: 4}$$

Here $\mathbf{A}_t = \left(1, a_t, \frac{1}{2}a_t^2\right)^T$,

$\mathbf{\Phi}(X_t) = \left(\Phi_1(X_t), \cdots, \Phi_M(X_t)\right)^T$, and

$$\mathbf{W}_t = \begin{pmatrix} W_{11}(t) & W_{12}(t) & \cdots & W_{1M}(t) \\ W_{21}(t) & W_{22}(t) & \cdots & W_{2M}(t) \\ W_{31}(t) & W_{32}(t) & \cdots & W_{3M}(t) \end{pmatrix} \qquad \text{Eq: 5}$$

Another Representation for $Q_t^\star$

We can also write the optimal Q-function differently:

$$Q_t^\star(X_t, a_t) = \mathbf{A}_t^T \mathbf{W}_t \mathbf{\Phi}(X_t) = \mathbf{A}_t^T \mathbf{U}_W(t, X_t)$$

Here the vector $\mathbf{U}_W(t, X_t)$ is defined as follows:

$$\mathbf{U}_W(t, X_t) = \mathbf{W}_t \mathbf{\Phi}(X_t)$$

Terminal conditions for components of vector $\mathbf{U}_W(t, X_t)$:

$$\mathbf{U}_W^{(0)}(T, X_T) = -\Pi_T(X_T) - \lambda\, Var\left[\Pi_T(X_T)\right]$$

$$\mathbf{U}_W^{(1)}(T, X_T) = 0\,, \quad \mathbf{U}_W^{(2)}(T, X_T) = 0$$

$a_t^\star$ and $Q_t^\star$ From $\mathbf{U}_W(t, X_t)$

We can compute the optimal action and optimal Q-function at this action in terms of elements of $\mathbf{U}_W(t, X_t)$:

$$a_t^\star(X_t) = -\frac{\mathbf{U}_W^{(1)}(t, X_t)}{\mathbf{U}_W^{(2)}(t, X_t)} \qquad \boxed{\text{Eq: 6}}$$

$$Q_t^\star(X_t, a_t^\star) = \mathbf{U}_W^{(0)}(t, X_t) - \frac{1}{2}(a_t^\star)^2\, \mathbf{U}_W^{(2)}(t, X_t)$$

If we know $a_t^\star$ and $Q_t^\star$ from observations, these equations can be viewed as two conditions on three components $\mathbf{U}_W^{(i)}$.

Dot Product Representation for $Q_t^\star$

We want to convert matrix $\mathbf{W}_t$ to a vector.

Re-arrange terms in Eq.(4) to convert it into a product of a parameter vector and a vector that depends on both the state and the action:

$$\begin{aligned} Q_t^\star(x,a) &= \mathbf{A}_t^T \mathbf{W}_t \mathbf{\Phi}(X) \\ &= \sum_{i=1}^{3}\sum_{j=1}^{M}\left(\mathbf{W}_t \odot \left(\mathbf{A}_t \otimes \mathbf{\Phi}^T(X)\right)\right)_{ij} \\ &= \vec{\mathbf{W}}_t \cdot vec\left(\mathbf{A}_t \otimes \mathbf{\Phi}^T(X)\right) \\ &\equiv \vec{\mathbf{W}}_t \vec{\mathbf{\Psi}}(X_t, a_t) \end{aligned} \qquad \text{Eq: 7}$$

Here $\odot$ stands for an element-wise (Hadamard) product. The vector $\vec{\mathbf{W}}_t$ is given by $\mathbf{W_t}$ with concatenating columns.

$\vec{\mathbf{\Psi}}(X_t, a_t) = vec\left(\mathbf{A}_t \otimes \mathbf{\Phi}^T(X)\right)$ is a vector obtained from the outer product $\mathbf{A}_t \otimes \mathbf{\Phi}^T(X)$

How Many Unknowns?

- For the RL setting:
  - The parameter vector $\vec{\mathbf{W}}_t$ and the state-action basis $\vec{\mathbf{\Psi}}_t$ both have $3M$ components
  - The number of data records for a RL problem per time step is $3N$ (X_t, a_t, R_t)
- For the DP setting:
  - $2M$ parameters defining the optimal policy and action value function
  - But only N values of X_t as input data
- The RL setting has more unknowns, but also more data!

Bellman Iteration as Regression

Coefficients $\mathbf{W}_t$ can now be computed recursively backward in time for $t = T-1, \ldots, 0$. To this end, the one-step Bellman optimality equation

$$Q_t^\star(x,a) = \mathbb{E}_t\left[R_t + \gamma \max_{a\in\mathcal{A}} Q_{t+1}^\star\left(X_{t+1}, a\right) \middle| x, a\right]$$

is interpreted as regression of the form

$$\begin{aligned} & R_t\left(X_t, a_t, X_{t+1}\right) + \gamma \max_{a\in\mathcal{A}} Q_{t+1}^\star\left(X_{t+1}, a\right) \\ & = \vec{\mathbf{W}}_t \vec{\boldsymbol{\Psi}}\left(X_t, a_t\right) + \varepsilon_t \end{aligned}$$

where ε_t is a random noise at time t with mean zero.

Bellman Regression: RL vs DP

- For the DP setting:
  - The Q-function is expanded in the state basis $\boldsymbol{\Phi}_t$
  - Applies only at the optimal Q-function at the optimal action $a_t^\star$
  - Rewards R_t are *computed*
- For the RL setting:
  - The Q-function is expanded in the state-action basis $\vec{\boldsymbol{\Psi}}_t$
  - Works for any action a_t, not just for the optimal action $a_t^\star$
  - Rewards R_t and actions a_t are *observed*

Least-square Optimization

Coefficients $\mathbf{W}_t$ are therefore found by solving the following least-square optimization problem:

$$\mathcal{L}_t\left(\mathbf{W}_t\right) = \sum_{k=1}^{N_{MC}} \left(R_t\left(X_t, a_t, X_{t+1}\right)\right. \qquad \boxed{\text{Eq: 8}}$$

$$\left. + \gamma \max_{a \in \mathcal{A}} Q^{\star}_{t+1}\left(X_{t+1}, a\right) - \vec{\mathbf{W}}_t \vec{\boldsymbol{\Psi}}\left(X_t, a_t\right)\right)^2$$

Note that this relation holds for a general *off-model, off-policy* setting of the Fitted Q Iteration method of RL.

Special Case: On-policy Learning

On-policy problem

$$\mathcal{L}_t^{(op)}\left(\mathbf{W}_t\right) = \sum_{k=1}^{N_{MC}} \left(R_t\left(X_t, a_t^\star, X_{t+1}\right)\right. \qquad \text{Eq: 9}$$

$$\left. +\gamma Q_{t+1}^\star\left(X_{t+1}, a_{t+1}^\star\right) - \vec{\mathbf{W}}_t \vec{\boldsymbol{\Psi}}\left(X_t, a_t^\star\right)\right)^2$$

Compare this with the optimization problem for DP:

$$F_t(\omega) = \sum_{k=1}^{N_{MC}} \left(R_t\left(X_t, a_t^\star, X_{t+1}\right)\right. \qquad \text{Eq: 10}$$

$$\left. +\gamma Q_{t+1}^\star\left(X_{t+1}, a_{t+1}^\star\right) - \sum_n^M \omega_{nt}\Phi_n\left(X_t^k\right)\right)^2$$

This implies

$$\lim_{N_{MC}\to\infty} Q_t^{\star, on-policy} = \lim_{N_{MC}\to\infty} Q_t^{\star, model} \qquad \text{Eq: 11}$$

Special Case: On-policy Learning

Because our optimization problem is *convex* and hence has a unique solution, and because Q-Learning is an *off-policy* algorithm, it means that a solution of Eq.(8) converges to the same limit given by Eq.(11).

For a large but finite number N_{MC}:

$$\mathcal{L}_t^{(DP)}\left(\mathbf{W}_t\right) = \sum_{k=1}^{N_{MC}} \left(\sum_n^M \omega_{nt}\Phi_n\left(X_t^k\right) - \vec{\mathbf{W}}_t \vec{\boldsymbol{\Psi}}\left(X_t, a_t\right)\right)^2$$

Solve this for the vector $\vec{\mathbf{W}}_t$

$$\vec{\mathbf{W}}_t = \left[\sum_{k=1}^{N_{MC}} \vec{\boldsymbol{\Psi}}\left(X_t^k, a_t^k\right) \vec{\boldsymbol{\Psi}}^T\left(X_t^k, a_t^k\right)\right]^{-1} \times$$

$$\sum_{k=1}^{N_{MC}} \vec{\boldsymbol{\Psi}}\left(X_t^k, a_t^k\right) \sum_n^M \omega_{nt}\Phi_n\left(X_t^k\right) \qquad \text{Eq: 12}$$

Off-policy FQI Solution

Introduce another pair of a matrix $\mathbf{S_t}$ and a vector $\mathbf{M_t}$:

$$
\begin{aligned}
S_{nm}^{(t)} &= \sum_{k=1}^{N_{MC}} \Psi_n\left(X_t^k, a_t^k\right) \Psi_m\left(X_t^k, a_t^k\right) \\
M_n^{(t)} &= \sum_{k=1}^{N_{MC}} \Psi_n\left(X_t^k, a_t^k\right) \left(R_t\left(X_t, a_t, X_{t+1}\right)\right. \\
&+ \left. \gamma \max_{a \in \mathcal{A}} Q_{t+1}^{\star}\left(X_{t+1}, a\right)\right)
\end{aligned}
$$

The optimal weights ω_t, the optimal Q-function at time t are then

$$
\mathbf{W}_t^{\star} = \mathbf{S}_t^{-1}\mathbf{M}_t \qquad \text{Eq: 13}
$$

Avoiding Optimistic Bias

$$Q^{\star}_{t+1}\left(X_{t+1}, a^{\star}_{t+1}\right) = \mathbf{U}^{(0)}_{W}\left(t+1, X_{t+1}\right)$$
$$+a^{\star}_{t+1}\mathbf{U}^{(1)}_{W}\left(t+1, X_{t+1}\right)$$
$$+\frac{\left(a^{\star}_{t+1}\right)^{2}}{2}\mathbf{U}^{(2)}_{W}\left(t+1, X_{t+1}\right) \qquad \text{Eq: 14}$$

To avoid an over-estimation bias, we should **not** maximize this expression w.r.t $a^{\star}_{t+1}$. Instead we should use here the analytic expression:

$$a^{\star}_{t}\left(X_{t}\right) = \frac{\mathbb{E}_{t}\left[\Delta\hat{S}_{t}\hat{\mathrm{I}}\mathrm{I}_{t+1} + \frac{1}{2\gamma\lambda}\Delta S_{t}\right]}{\mathbb{E}_{t}\left[\left(\Delta\hat{S}_{t}\right)^{2}\right]} \qquad \text{Eq: 15}$$

In this way, we avoid a potential overestimation problem of Q-Learning.

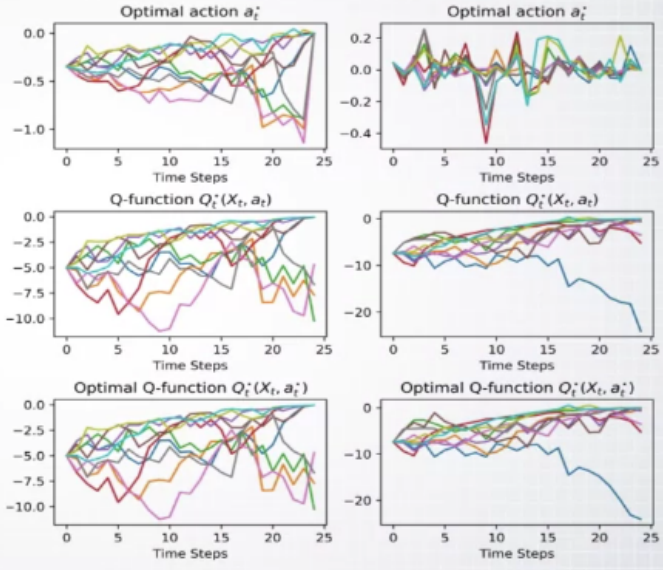
- The discrete-time Black-Scholes model as a MDP model
- Dynamic Programming solution when the model is *known*
- Reinforcement Learning FQI solution when the model is *unknown*
- As the RL solution relies on Q-Learning and FQI, it is a *model-free* and *off-policy* solution
- Can use the DP solution to benchmark RL solutions (FQI and possibly other RL methods)
- Simple semi-analytical solutions for both the DP and RL settings

- The MDP model is equivalent to a multi-period version of the Markowitz model for a simple portfolio of a stock and cash
- The Black-Scholes model is obtained in the continuous-time limit of such Markowitz model with log-normal dynamics of stock prices.
- The classical BS model and other Mathematical Finance model compute a "fair" "risk-neutral" option prices, ignore risk in options
- In our MDP model, residual risk in options is *priced.*

FQI Solution: On-policy Learning

The ATM European put option:

Parameter values: $K = 100$, $T = 1$, $S_0 = 100$, $\mu = 0.05$, $\sigma = 0.15$, $r = 0.03$, $\Delta t = 1/24$, $\lambda = 0.001$. Two sets of MC with $N_{MC} = 50,000$.



Produce randomized hedges from optimal DP hedges by multiplying by a random uniform number in the interval $[1-\eta, 1+\eta]$ where $0 < \eta < 1$. Take the values of $\eta = [0.15, 0.25, 0.35, 0.5]$ to test the noise tolerance of our algorithms. Results for $\eta = 0.5$:

Noise Tolerance for Off-policy Learning

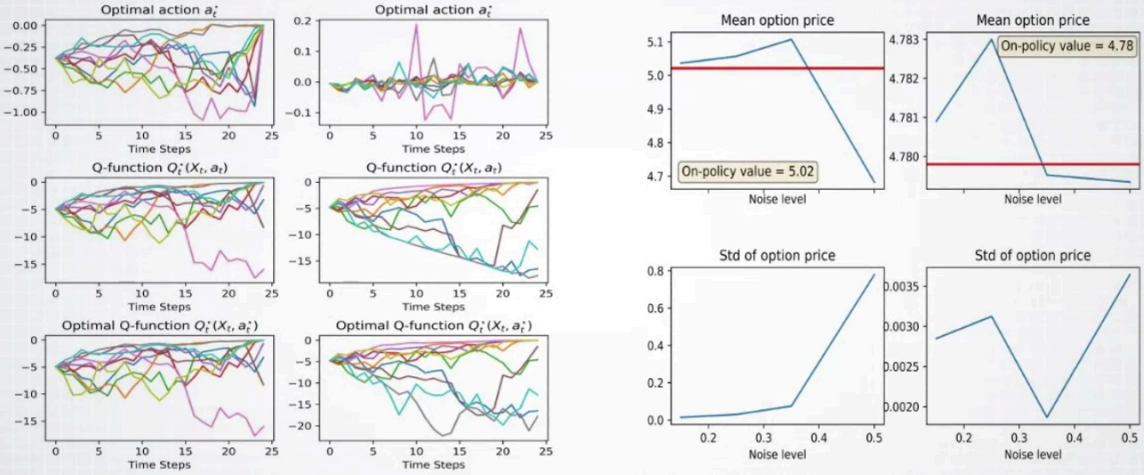


Figure: Means and standard deviations of option prices obtained with *off-policy* FQI learning with data obtained by randomization of DP optimal actions. Horizontal red lines show values obtained with *on-policy* learning corresponding to $\eta = 0$.