

# FINM 32000: Homework 7

Due Friday May 19, 2023 at 11:59pm

The code in the `ipynb` file should do Problem 1a if you set `algorithm = 'value'`.

It should do Problem 1b if you set `algorithm = 'policy'`

## Problem 1a

Complete the coding of the provided `ipynb` file which prices the Bermudan put option under GBM, with the same parameters as in the Excel worksheet from class (which has been posted on Canvas), using the Longstaff-Schwartz method.

Report an estimated price, based on 10000 paths.

At each exercise date, do the regression using only the paths that are in-the-money (at that specific date – so there may be different subsamples on different dates), not all of the paths.

## Problem 1b

The Longstaff-Schwartz method can be regarded as an example of a *Reinforcement Learning* (RL) algorithm. It selects actions (“exercise” vs. “continue”) to try to maximize an expected reward (option payoff) that depends on the transitions of a state variable (the underlying  $X$ ).

In particular, Longstaff-Schwartz takes a *Value*-function approach to solving the dynamic programming formulation of the Reinforcement Learning problem. It finds an estimate  $\hat{f}_n$  (same notation as L7) of the *value function* for the continuation action, by using OLS regression, of simulated continuation payoffs on the state variable. This estimated continuation value  $\hat{f}_n$  is compared against the value function for the exercise action, which is just the payoff function (for example  $\text{Payoff}(X) = K - X$  in the case of a put):

If  $\hat{f}_n(X_{t_n}) > \text{Payoff}(X_{t_n})$  then continue to hold at time  $t_n$

If  $\hat{f}_n(X_{t_n}) \leq \text{Payoff}(X_{t_n})$  then exercise at time  $t_n$

Here we will consider a different approach to RL.

In contrast to Value-function RL, another approach to Reinforcement Learning is the *Policy*-based approach. Rather than trying to estimate the value function (for the continuation action), it tries to more directly optimize the time- $t_n$  policy function, let's denote it  $\Phi$ , which maps each  $X$  to one of two outputs:  $\{0, 1\}$ , where 0 denotes continuing to hold, while 1 denotes stopping (exercising).

If  $\Phi(X_{t_n}) = 0$  then continue to hold at time  $t_n$

If  $\Phi(X_{t_n}) = 1$  then exercise at time  $t_n$

In the particular one-dimensional example of put pricing that we have been studying, we know what form the stopping policy function should take. In theory it should be an indicator function

$$\Phi_{c_n}(X) = \mathbf{1}_{X \leq c_n}$$

with a parameter  $c_n$  is a specific “critical” or “threshold” level of the stock price  $X$ . Below  $c_n$  you should exercise, and above  $c_n$  you should continue to hold the put. So, in principle, we could try to estimate the optimal threshold  $c_n$  by choosing it to maximize the average, across all simulated paths, of the simulated payout resulting from the policy  $\Phi_{c_n}$  at time  $t_n$ .

However, this optimization has some numerical difficulties, due to the discontinuity of this “hard stopping” decision function  $\Phi$  which only has two outputs  $\{0, 1\}$ . So suppose that we optimize a smoother function, a “*soft* stopping” decision function  $\varphi$  which produces outputs in the interval *between* 0 and 1. Let  $\varphi$  have two parameters  $a, b$  (which may depend on the time slice  $n$ ) and specifically let  $\varphi$  be<sup>1</sup> a *sigmoid* or *logistic* function of  $b(X - a)$ :

$$\varphi_{a,b}(X) = \frac{1}{1 + \exp(-b(X - a))}. \quad (*)$$

For large negative  $b$ , the  $\varphi_{a,b}$  will behave similarly to  $\Phi_a$ , in that it’s near 1 for  $X < a$  and near 0 for  $X > a$ . But unlike the hard stopping decision function, the soft decision function  $\varphi$  is more optimizer-friendly, because it varies continuously between 0 and 1. It can be interpreted as making the exercise decision randomly, with probability  $\varphi_{a,b}(X)$  of exercising, and probability  $1 - \varphi_{a,b}(X)$  of continuing to hold, conditional on  $X$ . At time  $t_n$  the optimizer should optimize

$$\max_{a,b} \left( \frac{1}{M} \sum_{m=1}^M \left( \varphi_{a,b}(X_{t_n}^m) \times (K - X_{t_n}^m) + (1 - \varphi_{a,b}(X_{t_n}^m)) \times (\text{Continuation payout on the } m\text{th path}) \right) \right)$$

where  $X^m$  denotes the  $m$ th simulated path. Then calculate payouts by converting this optimized soft stopping decision into a hard stopping decision by

$$\Phi(X_{t_n}) = \mathbf{1}_{\varphi_{\hat{a}, \hat{b}}(X_{t_n}) \geq 0.5} \times \mathbf{1}_{\text{Payoff}(X_{t_n}) > 0}$$

where  $\hat{a}$  and  $\hat{b}$  denote the optimized parameter values. Multiplying by  $\mathbf{1}_{\text{Payoff}(X_{t_n}) > 0}$  makes sure that you are not exercising OTM options. It should not be needed if your  $\varphi$  has been trained correctly, but we include it as a precaution.

Implement this policy optimization approach, by completing the code in the **ipynb** file. Most of the coding is already provided.

---

<sup>1</sup>On this problem, which is simple in the sense that the exercise region in  $X$ -space is just a one-dimensional interval, a single sigmoid function (\*) is sufficient to approximate the optimal stopping policy.

On harder problems, where the exercise region may be a complicated subset of a multidimensional  $X$ -space, the function (\*) can be upgraded to a *deep neural network*.

For instance see <http://jmlr.org/papers/volume20/18-232/18-232.pdf>

## Problem 2

Let  $R_1$  and  $R_2$  be random variables and define the function  $F : \mathbb{R}^2 \rightarrow \mathbb{C}$  by

$$F(u, v) := \mathbb{E}e^{iuR_1 + ivR_2}.$$

Then  $F$  is said to be the characteristic function (or joint characteristic function) of  $(R_1, R_2)$

Express the following in terms of  $F$ .

(a)  $\mathbb{E}(R_1^2 + R_2^2)$

(b)  $\varphi(w)$ , where  $\varphi$  denotes the characteristic function of  $4R_1 - 3R_2$ .

(c)  $G(x, y)$  where  $G$  denotes the joint characteristic function of  $(R_3, R_4)$ .

Assume that  $R_3$  and  $R_4$  are independent.

Assume that  $R_3$  has the same distribution as  $R_1$ , and  $R_4$  has the same distribution as  $R_2$ .