Solving high-dimensional optimal stopping problems using deep learning

Shaghayegh Dianat

February 5, 2025

Definition (Stopping Time)

Let $(\Omega, \mathcal{F}, (\mathcal{F}_t)_{t \geq 0}, \mathbb{P})$ be a filtered probability space. A **stopping time** is a random variable $\tau : \Omega \to [0, +\infty]$ such that for every $t \geq 0$:

$$\{\tau \leq t\} \in \mathcal{F}_t$$

i.e., the event "stopping before or at time t" is measurable with respect to the information available up to time t ($\mathcal{F}_t$).

Example (First Passage Time for Brownian Motion)

Consider a standard Brownian motion $(W_t)_{t \geq 0}$ and its natural filtration $\mathcal{F}_t = \sigma(W_s : 0 \leq s \leq t)$. For $a > 0$, define:

$$\tau_a = \inf\{t \geq 0 : W_t = a\}$$

τ_a is a stopping time.

Core Concepts in Optimal Stopping Theory

1. Snell Envelope

The Snell envelope $U(t)$ of a reward process (X_t) is:

$$U_t = \max\{g(t, X_t), \mathbb{E}[U_{t+1} \mid \mathcal{F}_t]\}$$

- Smallest supermartingale dominating $g(t, X_t)$,the reward received if you stop at time t
- Encodes optimal expected value at time t

2. Optimal Value

The **optimal value** V is:

$$V = \sup_{\tau \in \mathcal{T}} \mathbb{E}[g(X_\tau, \tau)]$$

where $\mathcal{T}$ is the set of all stopping times..

3. Continuation Value

The **continuation value** C_t at time t:

$$C_t = \mathbb{E}[U_{t+1} \mid \mathcal{F}_t]$$

Expected reward from not stopping at t

Stop if $g(t, X_t) \geq C_t$, continue otherwise

4. Optimal Stopping Time

The **optimal stopping time** τ^* satisfies:

$$\tau^* = \inf\{t \geq 0 : g(t, X_t) = U_t\}$$

and achieves $\mathbb{E}[g(\tau^*, X_{\tau^*})] = V$.

Key Relationship:

$$U(t) = \max\{g(t, X_t),\ C_t\} \quad \text{with} \quad C_t = \mathbb{E}[U_{t+1} \mid \mathcal{F}_t]$$

Key Option Pricing Concepts

1. Option & Option Pricing

An **option** grants the holder the right (but not obligation) to buy/sell an asset at a strike price K by expiration. Pricing involves determining its fair value via models like:

$$\text{European Call: } C = \mathbb{E}^{\mathbb{Q}}[e^{-rT}(S_T - K)^+]$$

where $\mathbb{Q}$ is the risk-neutral measure, S_T is the asset price at expiry T, and r is the risk-free rate.

2. American Option

An **American option** can be exercised *at any time* before expiration. Its price $V(t, S_t)$ solves:

$$V(t, S_t) = \sup_{\tau \in [t, T]} \mathbb{E}^{\mathbb{Q}}[e^{-r(\tau - t)}(S_\tau - K)^+ \mid \mathcal{F}_t]$$

where τ is a stopping time. Pricing requires dynamic programming or PDE

3. Bermudan Option

A **Bermudan option** allows exercise on predefined dates $\{t_1, t_2, \ldots, t_N\}$. Its value satisfies:

$$V(t, S_t) = \max\left\{(S_t - K)^+,\ \mathbb{E}^{\mathbb{Q}}[e^{-r\Delta t} V(t + \Delta t, S_{t+\Delta t}) \mid \mathcal{F}_t]\right\}$$

at each exercisable time t. Less flexible than American but more tractable numerically.

4. Max Call Bermudan Option (High-Dimensional)

A **Max Call Bermudan option** on d-assets has payoff:

$$\text{Payoff} = \left(\max_{1 \le i \le d} S_\tau^{(i)} - K\right)^+$$

where τ is a stopping time in the Bermudan schedule. Pricing in high dimensions ($d \gg 1$) suffers from the *curse of dimensionality*.

Optimal Stopping Approaches

Continuation Value Estimation (Tilley, 1993): Uses Monte Carlo methods to estimate continuation values, helping to derive stopping rules or approximate the Snell envelope.
Optimal Exercise Boundary Approximation (Andersen, 2000): Focuses on determining the stopping boundary, often modeled as a free boundary in PDEs.
Dual Approach (Rogers, 2002; Haugh & Kogan, 2004): Reformulates the problem using duality and martingale methods to derive alternative representations and bounds.

Key Challenge: While it is possible to solve optimal stopping problems theoretically, numerical solutions can be challenging due to the curse of dimensionality, where higher dimensions lead to increasingly difficult and expensive computations. To address this, we employ the methodology presented by Becker et al, which is a deep learning approach for approximating the optimal stopping time,

Training a Neural Network

The training data can consist of input and target pairs (x^m, z^m) or just inputs (x^m).

Training the network refers to setting the weight w_i and bias b_i terms using the training data.

Training is done by minimizing a loss function, C, w.r.t θ.

Example: Mean squared-error loss function

$$C(\theta) = \frac{1}{M} \sum_{m=1}^{M} ||z(\theta, x^m) - z^m||^2$$

Training a Neural Network (Continued)

The optimal parameters are found by minimizing the loss function via a gradient descent algorithm.

Example: Vanilla gradient descent with learning rate η

$$\theta_{t+1} = \theta_t - \eta \cdot \nabla C(\theta_t)$$

$\nabla C(\theta_t)$ is computed using backpropagation.

After training, we run the algorithm using a new dataset, this is the testing data.

Note: Minimizing a loss function via gradient descent is equivalent to maximizing a reward function via gradient ascent.

Expressing Stopping Times as a Series of 0-1 Decisions

Consider the representation

$$\tau = \sum_{n=1}^{N} n\, f_n(X_n) \prod_{j=0}^{n-1} \Big(1 - f_j(X_j)\Big), \tag{1}$$

where for each $n = 0, 1, \ldots, N$, the function

$$f_n : \mathbb{R}^d \to \{0, 1\}$$

is measurable and represents a binary stopping decision. The interpretation is as follows:

Binary Decision: At time n, the value $f_n(X_n)$ equals 1 if the process should stop and 0 if it should continue.

First-Stop Mechanism: The product term

$$\prod_{j=0}^{n-1} \Big(1 - f_j(X_j)\Big)$$

ensures that no stopping decision has been taken at any earlier time $j < n$. If any previous $f_j(X_j) = 1$, then the product becomes 0

Earliest Optimal Stopping: Consequently, the sum in (1) picks out the first time n at which $f_n(X_n) = 1$. For instance, if $f_{n_0}(X_{n_0}) = 1$ and $f_j(X_j) = 0$ for all $j < n_0$, then

$$\prod_{j=0}^{n_0-1} \Big(1 - f_j(X_j)\Big) = 1,$$

and all later terms vanish.

Optimality via Backward Induction: In practice, the functions $\{f_n\}_{n=0}^{N}$ are chosen recursively (backward in time) such that, at each stage, the decision to stop is optimal relative to the expected future reward (continuation value). This procedure guarantees that the overall stopping time τ is optimal.

Thus, Equation (1) encodes an optimal stopping time by combining sequential binary decisions into a single rule that stops at the first optimal time.

Theorem 1

This theorem addresses the recursive construction of an optimal stopping time for a discrete-time Markov process. Let

$$V_n = \sup_{\tau \in \mathcal{T}_n} E\big[g(\tau, X_\tau)\big]$$

be the optimal expected reward when stopping is allowed only at times $n, n+1, \ldots, N$. Assume that for time $n+1$, an optimal stopping time

$$\tau_{n+1} = \sum_{m=n+1}^{N} m\, f_m(X_m) \prod_{j=n+1}^{m-1} \Big(1 - f_j(X_j)\Big)$$

has been constructed via measurable functions $f_m : \mathbb{R}^d \to \{0, 1\}$ (with $f_N \equiv 1$). Then, exists a measurable function

$$f_n : \mathbb{R}^d \to \{0, 1\}$$

such that defining

$$\tau_n = n\, f_n(X_n) + \tau_{n+1}\left(1 - f_n(X_n)\right), \tag{2}$$

one obtains:

$$E\big[g(\tau_n, X_{\tau_n})\big] \geq V_n - \left(V_{n+1} - E\big[g(\tau_{n+1}, X_{\tau_{n+1}})\big]\right).$$

This inequality implies that if τ_{n+1} is optimal (or near-optimal) for times $n+1, \ldots, N$, then there is an optimal decision at time n that nearly preserves the optimality when combined with τ_{n+1}.

Recursive Optimality: The theorem formalizes the idea that optimal stopping decisions can be made recursively. Starting from the terminal time N (where stopping is trivial), one can determine the stopping decision at time $N-1$, then at $N-2$, and so on, back to time 0.

Role of the Continuation Value: The function

$$h_n(X_n) := E\left[g(\tau_{n+1}, X_{\tau_{n+1}}) \mid X_n\right]$$

represents the expected reward if the process continues. The decision function f_n is chosen based on a comparison between the immediate reward $g(n, X_n)$ and the continuation value $h_n(X_n)$.

Binary Decisions: The theorem shows that it suffices to decide at each time step whether to stop (i.e., $f_n(X_n) = 1$) or continue (i.e., $f_n(X_n) = 0$) based solely on the current state X_n. This is a key insight in many optimal stopping problems and underlies numerical methods (such as those using neural networks) to approximate the optimal stopping rule.

Sketch of Proof

The proof of Theorem 1 can be understood in the following key steps:

By the Doob–Dynkin lemma, there exists a measurable function $h_n : \mathbb{R}^d \to \mathbb{R}$ such that

$$h_n(X_n) = E\left[g(\tau_{n+1}, X_{\tau_{n+1}}) \mid X_n\right].$$

This function represents the *continuation value* if the decision is made to *continue* at time n.

Compare the immediate reward $g(n, X_n)$ with the continuation value $h_n(X_n)$.

Define the binary decision function:

$$f_n(x) = \begin{cases} 1, & \text{if } g(n,x) \geq h_n(x), \\ 0, & \text{if } g(n,x) < h_n(x). \end{cases}$$

Intuitively, if stopping immediately at time n is as good as or better than waiting, then $f_n(x) = 1$.

With the decision function f_n defined, the new stopping time is given by

$$\tau_n = n\, f_n(X_n) + \tau_{n+1} \Big(1 - f_n(X_n)\Big).$$

This construction guarantees that if $f_n(X_n) = 1$, then $\tau_n = n$, otherwise the process continues according to the previously defined τ_{n+1}.

One shows that for any stopping time $\tau \in \mathcal{T}_n$, the expected reward satisfies

$$E\big[g(\tau_n, X_{\tau_n})\big] \geq E\big[g(\tau, X_\tau)\big] - \epsilon,$$

where

$$\epsilon = V_{n+1} - E\big[g(\tau_{n+1}, X_{\tau_{n+1}})\big].$$

In the case where τ_{n+1} is optimal (i.e., $\epsilon = 0$), this implies $E[g(\tau_n, X_{\tau_n})] \geq V_n$, meaning that τ_n is optimal among stopping times in $\mathcal{T}_n$.

By applying the above steps recursively from $n = N - 1$ down to $n = 0$, one constructs a sequence of decision functions $\{f_n\}_{n=0}^{N}$. The final stopping time,

$$\tau = \sum_{n=1}^{N} n\, f_n(X_n) \prod_{j=0}^{n-1} \Big(1 - f_j(X_j)\Big),$$

is then optimal for the original problem.

Summary: The proof of Theorem 1 leverages the properties of conditional expectations and the Markov property of the process to show that a binary decision at each time—based solely on a comparison between the immediate reward and the continuation value—can be combined recursively to yield an optimal stopping rule and can be used to compute Vn.

Introducing a Neural Network

Using $f_N \equiv 1$ we construct a sequence of neural networks, $f^{\theta_n} : \mathbb{R}^d \to \{0, 1\}$, to approximate f_n for $n = N - 1, \dots, 0$.

We can then approximate τ_{n+1} via

$$\sum_{k=n}^{N} k \cdot f^{\theta_k}(X_k) \prod_{j=n}^{k-1} \left(1 - f^{\theta_j}(X_j)\right)$$

Problem:

f^{θ_n} produces binary output $\Longrightarrow$ it is not continuous w.r.t. θ.

Need continuous output to train θ_n via a gradient-based optimization algorithm.

Neural Network with Continuous Output

Introduce $F^\theta : \mathbb{R}^d \to (0,1)$, a multi-layer, feed-forward neural network of the form

$$F^\theta = \psi \circ a_I^\theta \circ \phi_{q_{I-1}} \circ a_{I-1}^\theta \circ \ldots \phi_{q_1} \circ a_1^\theta$$

where

q_I are the number of nodes in the hidden layers.
$a_1^\theta : \mathbb{R}^d \to \mathbb{R}^{q_1}, a_{I-1}^\theta : \mathbb{R}^{q_{I-2}} \to \mathbb{R}^{q_{I-1}}, \ldots, a_I^\theta : \mathbb{R}^{q_{I-1}} \to \mathbb{R}$ satisfy

$$a_i^\theta(x) = W_i x + b_i$$

$\phi_{q_I} : \mathbb{R}^{q_I} \to \mathbb{R}^{q_I}$ are ReLU activation function,
$\phi_{q_I}(x_1, \ldots, x_{q_I}) = \left(x_1^+, \ldots, x_{q_i}^+\right)$.
$\psi : \mathbb{R} \to \mathbb{R}$ is the logistic sigmoid function, $\psi(x) = 1/\left(1 + e^{-x}\right)$.

Return to Binary Decisions

The dimension of the parameter space is:

$$q = \begin{cases} d+1 & \text{if } I = 1 \\ 1 + q_1 + \cdots + q_{I-1} + dq_1 + \cdots + q_{I-2}q_{I-1} + q_{I-1} & \text{if } I \geq 2 \end{cases}$$

and for given $x \in \mathbb{R}^d$, $F^\theta(x)$ is continuous as well as almost everywhere smooth in θ. Our aim is to determine $\theta_n \in \mathbb{R}^q$ so that

$$\mathbb{E}\left[g(n, X_n)\, F^{\theta_n}(X_n) + g\left(\tau_{n+1}, X_{\tau_{n+1}}\right)\left(1 - F^{\theta_n}(X_n)\right)\right]$$

is close to the $\sup_{\theta \in \mathbb{R}^q} \mathbb{E}\left[g(n, X_n)\, F^\theta(X_n) + g\left(\tau_{n+1}, X_{\tau_{n+1}}\right)\left(1 - F^\theta(X_n)\right)\right]$. After that, we can define the function $f^{\theta_n} : \mathbb{R}^d \to \{0, 1\}$ by

$$f^{\theta_n} = 1_{[0,\infty)} \circ a_I^{\theta_n} \circ \varphi_{q_{I-1}} \circ a_{I-1}^{\theta_n} \circ \cdots \circ \varphi_{q_1} \circ a_1^{\theta_n} \qquad (3)$$

where 1 is the indicator function of $[0, \infty)$.

Selecting a Reward Function

We want to define a reward function that yields a stopping decision at time n that will maximize our expected future payoff.
If at time n we:

Stop $\implies f_n(X_n) = 1$ and we will receive payoff $g(n, X_n)$

Continue and after proceed optimally $\implies f_n(X_n) = 0$ and we will eventually receive payoff of $g\left(\tau_{n+1}, X_{\tau_{n+1}}\right)$.

So we want our reward function at time n to approximate

$$\sup_{f \in \mathcal{D}} \mathbb{E}\left[g(n, X_n) f(X_n) + g\left(\tau_{n+1}, X_{\tau_{n+1}}\right)(1 - f(X_n))\right] \qquad (4)$$

where $\mathcal{D}$ is the set of all $f : \mathbb{R}^d \to \{0, 1\}$ measurable.

...But can we even replace f in (4) with neural network f^θ ?

Theorem 2

Let $n \in \{0, \ldots, N-1\}$ and fix a stopping time $\tau_{n+1} \in \mathcal{T}_{n+1}$, then $\forall \epsilon > 0$,$\forall l >= 2$, there exists $q_1, ..., q_{l-1} \in \mathbb{N}^+$such that

$$\sup_{\theta \in \mathbb{R}^q} \mathbb{E}\left[g(n, X_n) f^\theta(X_n) + g\left(\tau_{n+1}, X_{\tau_{n+1}}\right)\left(1 - f^\theta(X_n)\right)\right]$$
$$\geq \sup_{f \in \mathcal{D}} \mathbb{E}\left[g(n, X_n) f(X_n) + g\left(\tau_{n+1}, X_{\tau_{n+1}}\right)(1 - f(X_n))\right] - \epsilon$$

where $\mathcal{D}$ is the set of all measurable functions $f \colon \mathbb{R}^d \to \{0, 1\}$. g that we can replace f with f^θ Therefor, for any depth$l >= 2$, a neural network of the form (3) is flexible enough to decide on close-to optimal stopping decisions if it has a sufficient number of nodes.

Corrallary 1

For any $\epsilon > 0$, there exists $q_1, ..., q_{I-1} \in \mathbb{N}^+$and neural network functions of the form (3) such that $f^{\theta_N} \equiv 1$ and the stopping time

$$\hat{\tau} = \sum_{n=1}^{N} n f^{\theta_n}(X_n) \prod_{k=0}^{n-1} \left(1 - f^{\theta_k}(X_k)\right) \tag{5}$$

satisfies $\mathbb{E}g(\hat{\tau}, X_{\hat{\tau}}) \geq \sup_{\tau \in \mathcal{T}} \mathbb{E}g(\tau, X_\tau) - \epsilon$.

This means we can approximate the sequence of optimal stopping decisions $\{f_n\}_{n=0}^{N}$ with the sequence of optimized neural networks $\{f^{\theta_n}\}_{n=0}^{N}$.

Computing Estimates via the Neural Network

Step 1. Simulate M independent paths of the Markov process $(X_n)_{n=0}^N$. These sample paths are denoted $(x_n^m)_{n=0}^N$ for $m = 1, \ldots, M$. These (x_n^m) will be our training data.
Set $f^{\theta_N}(x_N^m) \equiv 1, \forall m$, since we must stop at time N.
Step 2. Compute the rest of the $\{\theta_n\}_n$ recursively backwards for $n = N - 1, \ldots, 0$. We will do that using a loss function such that replicates (4) in the neural network, and so, at each epoch we will search for the best θ_n that maximizes (4) (or the one that minimizes the negative of (4) by gradient decent).
Step 3. Assume we are at time step n (and so, we have already computed $f^{\theta_{n+1}}, \ldots, f^{\theta_N}$). We can compute the $n+1$ optimal stopping time for each of the m paths:

$$\hat{\tau}_{n+1}^m = \sum_{k=n+1}^{N} k f^{\theta_k}(x_n^m) \prod_{j=n+1}^{k-1} \left(1 - f^{\theta_j}(x_j^m)\right) \tag{6}$$

Remark 1. Notice that, if we stop at time n along the path m with probability F^{θ_n} and afterwards behave optimally with the next $f^{\theta_{n+1}}, \dots, f^{\theta_N}$, obtaining eventually the payoff of $g\left(\hat{\tau}_{n+1}^m\right)$ the realized reward for the m-th simulated path is:

$$r_n^m(\theta_n) = g(n, x_n^m) F^{\theta_n}(x_n^m) + g\left(\hat{\tau}_{n+1}^m, x^m_{\hat{\tau}^m_{n+1}}\right) \cdot \left(1 - F^{\theta_n}(x_n^m)\right)$$

In other words, the virtual reward along the path m is the probability of stopping at the time we are times the reward of stopping at this time plus the reward of following the path until we get to the optimal time.
Remark 2.Observe that this reward, for sufficiently large M, the averages of r_n^m with respect to m replicate

$$\mathbb{E}\left[g(n, X_n) f^{\theta}(X_n) + \left(1 - f^{\theta}(X_n)\right) g\left(\tau_{n+1}, X_{\tau_{n+1}}\right)\right] \tag{7}$$

Step 4. We can thus approximate (7) with

$$r_n^{MC} = \frac{1}{M}\sum_{m=1}^{M} r_n^m(\theta_n) \tag{8}$$

Notice that this is the Monte Carlo (MC) estimator for $\mathbb{E}_m[r_n^m]$.
Also, it is a smooth function with respect to θ almost everywhere (by definition of (8)). This is our desirable function to have optimized via a gradient descent algorithm with respect to θ_n (our loss function). After obtaining the optimal θ_n, we will substitute F^{θ_n} with f^{θ_n}.
We repeat this step for $n = N - 1, \dots, 0$.
Step 5. Generate a new set of sample paths $(y_n^m)_{n=0}^N$ for $m = 1, \dots, M$, which will be our testing data.

Step 6. Using the $\{\theta_n\}_n$ that we found with our training data (that is, $\left((x_n^k)_{n=0}^N\right)$, $k = 1, 2, \ldots$, compute backwards recursively for $n = N-1, \ldots, 0$

$$\hat{\tau}_{n+1}^m = \sum_{k=n+1}^{N} k f^{\theta_k}(y_k^m) \prod_{j=n+1}^{k-1} \left(1 - f^{\theta_j}(y_j^m)\right)$$

along each of the M sample paths.

Step 7. At time $n = 0$, we finish by computing the current optimal stopping time

$$\hat{\tau}^m = \sum_{n=1}^{N} k f^{\theta_n}(y_n^m) \prod_{k=1}^{n-1} \left(1 - f^{\theta_k}(y_k^m)\right)$$

This is the estimator for the $\hat{\tau}$ of (5).

Step 8. The estimator of $\mathbb{E} g(\hat{\tau}, X_{\hat{\tau}})$ is given by:

$$\hat{V} = \frac{1}{M} \sum_{m=1}^{M} g\left(\hat{\tau}_m, y_{\hat{\tau}_m}^m\right) \tag{9}$$

Remark 3. Notice that $\hat{V}$ is the Monte Carlo estimator for $\mathbb{E}g(\hat{\tau}, X_{\hat{\tau}})$, and so, is unbiased. It is also consistent, thanks to the law of large numbers.
Remark 4. If the Markov process X starts from a deterministic initial value $x_0 \in \mathbb{R}^d$, the initial stopping decision is given by a constant $f_0 \in \{0, 1\}$. To learn f_0 from the simulated paths of X, we can compare the initial reward $g(0, x_0)$ to the Monte Carlo estimate $\hat{V}$ of $\mathbb{E}g(\tau_1, X_{\tau_1})$ where $\tau_1 \in \mathcal{T}_1$ is of the form

$$\tau_1 = \sum_{n=1}^{N} n f^{\theta_n}(X_n) \prod_{j=1}^{n-1} \left(1 - f^{\theta_j}(X_j)\right)$$

for $f^{\theta_N} \equiv 1$ and trained parameters $\theta_1, \ldots, \theta_{N-1} \in \mathbb{R}^q$. Then, stop immediately $(f_0 = 1)$ if $g(0, x_0) \geq \hat{V}$ and continue $(f_0 = 0)$ otherwise. Then, the stopping time is of the form

$$\tau^{\Theta} = \begin{cases} 0 & \text{if } f_0 = 1 \\ \tau_1 & \text{if } f_0 = 0 \end{cases}$$

Notice that τ_1 is equivalent to $\hat{\tau}^m$ specifying the drawing of the sample paths y_n^m of X.

Algorithm 1 Deep Optimal Stopping Time Algorithm (Part 1)

```
1: Simulate $M$ independent paths of $(X_n)_{n=0}^N : (x_n^m)_{n=0}^N$, training data.
2: Fix $f^{\theta_n} \equiv 1, \forall n$. Initialize an $N+1 \times M$ matrix $MF$:
```

$$MF[i][j] = \begin{cases} 0, & \forall i = 1, \dots, N-1, \forall j = 1, \dots, M \\ 1, & \forall j = 1, \dots, M \end{cases}$$

```
3: Initialize an $N+1 \times M$ matrix $M_T$:
```

$$M_T[i][j] = \begin{cases} 0, & \forall i = 1, \dots, N-1, \forall j = 1, \dots, M \\ N, & \forall j = 1, \dots, M \end{cases}$$

```
4: for $n = N-1$ to 0 do
5:     Initialize a Neural Network $F^{\theta_n}$ with a suitable optimizer.
6:     Define the loss function:
```

$$r(\theta_n) = -\frac{1}{M} \sum_{m=1}^{M} \Big(g(n, x_n^m) \cdot F^{\theta_n}(x_n^m) + g(\tau_{n+1}^m, x_{\tau_{n+1}}^m) \cdot (1 - F^{\theta_n}(x_n^m)) \Big)$$

Algorithm 2 Deep Optimal Stopping Time Algorithm (Part 2)

```
1: We switch from $F^{\theta_n}$ to $f^{\theta_n}$:
       $$f^{\theta_n}(x_n^m) = (F^{\theta_n}(x_n^m) > 0.5)$$
2: Update $MF[n][m] = (F^{\theta_n}(x_n^m) > 0.5)$ for all $m$.
3: Update $M_T[n][m]$ equal to the index of the maximum $MF[i][m]$ where
   $i = n, \dots, N$ for all $m$.
4: Prioritize $M_T[n][m]$ over $M_T[n+1][m]$ if both are $MF[i][m] = 1$.
5: Simulate $K_L$ independent paths $(X_n)_{n=0}^N : (y_n^k)_{n=0}^N$, testing data.
6: Compute $\hat{L} := \hat{V}$ with $(y_n^k)_{n=0}^N$ instead of $(x_n^m)_{n=0}^N$.
```

Confidence interval for optimal value

Note that $\hat{V}$ (9) is a consistent estimator for $\mathbb{E}g(\hat{\tau}, X_{\hat{\tau}})$ by the law of large numbers. In addition, since it is just the standard Monte Carlo estimate for $\mathbb{E}g(\hat{\tau}, X_{\hat{\tau}})$, it is also unbiased. Furthermore, by the central limit theorem, a $1-\alpha$, $\alpha \in (0,1)$ confidence interval for $\mathbb{E}g(\hat{\tau}, X_{\hat{\tau}})$ is:

$$\left[\hat{V} - z_{\alpha/2}\frac{\hat{\sigma}}{\sqrt{M}}, \hat{V} + z_{\alpha/2}\frac{\hat{\sigma}}{\sqrt{M}}\right]$$

where $z_{\alpha/2}$ is the $1-\alpha/2$ quantile of $\mathcal{N}(0,1)$ and $\hat{\sigma}$ is the corresponding sample standard deviation which is given by

$$\hat{\sigma} = \sqrt{\frac{1}{M-1}\sum_{m=1}^{M}\left(g(\hat{\tau}_m, y^m_{\hat{\tau}_m}) - \hat{V}\right)^2}.$$

Application: Bermudan Max Call Option

Bermudan max-call option expiring at time T with strike price K written on d assets, $X^1, \ldots, X^d$, with $N+1$ equidistant exercise times $t_n = nT/N, n = 0, 1, \ldots, N$.

The corresponding payoff function at time t is: $\left(\max_{i\in\{1,\ldots,d\}} X_t^i - K\right)^+$.

So its price at time t is $\sup_\tau \mathbb{E}\left[e^{-r\tau}\left(\max_{i\in\{1,\ldots,d\}} X_\tau^i - K\right)^+\right]$.

Frame this as an optimal stopping problem $\sup_{\tau\in\mathcal{T}} \mathbb{E} g(\tau, X_\tau)$ where

$$g(n,x) = e^{-rt_n}\left(\max_{i\in\{1,\ldots,d\}} x^i - K\right)^+$$

Multidimensional Black-Scholes underlying assets

We will apply the optimal stopping problem to the pricing of multi-dimensional Bermudan max-call options with the underlying asset being driven by a multi-dimensional Black-Scholes model, replicating the results obtained by Becker et al and comparing them with the literature.

Assume a Black-Scholes market model with d uncorrelated assets.

For $i = 1, \dots, d$ set:

$$x_0^i = 90/100/110, \quad K = 100, \quad \sigma_i = 0.2, \quad \delta_i = 0.1, \quad r = 0.05, \quad T = 3,$$

Asset price paths can be simulated via

$$x_{n,i}^m = x_{0,i} \cdot \exp\left\{ \sum_{k=0}^{n} \left(\left(r - \delta_i - \sigma_i^2/2 \right) \Delta t + \sigma_i \sqrt{\Delta t} \cdot Z_{k,i}^m \right) \right\} \qquad (10)$$

where $\Delta t = T/N$ and $Z_{k,i}^m \sim \mathcal{N}(0,1)$.

Results

d	S0	Actual	Estimate
2	100	13.902	13.778
3	90	11.29	11.176
5	110	36.710	36.673
10	90	26.208	25.494

Results computed using $8192 * (1500 + d)$ sample paths.

Actual estimates from [Becker et al., 2018] and [Hu, 2019].

References

Main Reference:

Becker, S., Cheridito, P., and Jentzen, A. (2018).
Deep optimal stopping.

Boix Torres, A. (2022-2023).
A Deep Learning Method for Optimal Stopping Problems.
Advisor: Argimiro Arratia Quesada. Department of Computer Science , BGSMath , IMTech.

Hijazi, S., Kumar, R., and Rowen, C. (2015).

Andersen, L., and Broadie, M. (2004).
Management Science, 50(9), 1222-1234.

Broadie, M., and Cao, M. (2008).
Quantitative Finance, 8(8), 845-861.