Pricing and hedging American-style options with deep learning

Sebastian Becker∗ Patrick Cheridito† Arnulf Jentzen‡

December 2019

Abstract

This paper describes a deep learning method for pricing and hedging American-style options. It first computes a candidate optimal stopping policy. From there it derives a lower bound for the price. Then it calculates an upper bound, a point estimate and confidence intervals. Finally, it constructs an approximate dynamic hedging strategy. We test the approach on different specifications of a Bermudan max-call option. In all cases it produces highly accurate prices and dynamic hedging strategies yielding small hedging errors.

Keywords: American option, Bermudan option, optimal stopping, lower bound, upper bound, hedging strategy, deep neural network

1 Introduction

Early exercise options are notoriously difficult to value. For up to three underlying risk factors, tree-based and classical PDE approximation methods usually yield good numerical results; see e.g. [19, 14, 27] and the references therein. To treat higher-dimensional problems, various simulation based methods have been developed; see e.g., [31, 4, 11, 26, 32, 15, 8, 24, 13, 9, 8, 21, 18, 23] have already used shallow neural networks to estimate continuation values. More recently, in [30] optimal stopping problems in continuous time have been solved by approximating the solutions of the corresponding free boundary PDEs with deep neural networks. In [31, 3] deep learning has been used to directly learn optimal stopping strategies. The main focus of these papers is to derive optimal stopping rules and accurate price estimates.

The goal of this article is to develop a deep learning method which learns the optimal exercise behavior, prices and hedging strategies from samples of the underlying risk factors. It first learns a candidate optimal stopping strategy by regressing continuation values on multilayer neural networks. Employing the learned stopping strategy on a new set of Monte Carlo samples gives a low-biased estimate of the price. Moreover, the candidate optimal stopping strategy can be used

∗Zenai AG, Zurich, Switzerland
†Department of Mathematics, ETH Zurich, Switzerland
‡Faculty of Mathematics and Computer Science, University of Münster, Germany

1That is, feedforward networks with a single hidden layer.
to construct an approximate solution to the dual martingale problem of [28] and [18], yielding a high-biased estimate and confidence intervals for the price. In the last step, our method learns a dynamic hedging strategy in the spirit of [17] and [10]. But here, the continuation value approximations learned during the construction of the optimal stopping strategy can be used to break the hedging problem down into a sequence of smaller problems that learn the hedging portfolio only from one possible exercise date to the next. Alternative ways of computing hedging strategies consist in calculating sensitivities of option prices (see e.g., [3, 7, 21]) or approximating a solution to the dual martingale problem (see [28, 29]).

Our work is related to the preprints [25] and [12]. [25] also uses neural network regression to estimate continuation values. But the networks are slightly different. While [25] works with leaky ReLU activation functions, we use tanh activation. Moreover, [25] provides a proof that the pricing algorithm converges if the number of simulations and the sizes of the networks go to infinity, whereas we calculate a posteriori guarantees for the prices and use the estimated continuation value functions to implement an efficient hedging algorithm. [12] proposes an alternative way of calculating prices and hedging strategies for American-style options by solving BSDEs.

The rest of the paper is organized as follows. In Section 2 we describe our neural network version of the Longstaff–Schwartz algorithm to estimate continuation values and construct a candidate optimal stopping strategy. In Section 3 the latter is used to derive lower and bounds as well as confidence intervals for the price. Section 4 discusses two different ways of computing dynamic hedging strategies. In Section 5 the results of the paper are applied to price and hedge a call option on the maximum of different underlying assets. Section 6 concludes.

2 Calculating a candidate optimal stopping strategy

We consider an American-style option that can be exercised at any one of finitely many times $0 = t_0 < t_1 < \cdots < t_N = T$. If exercised at time $t_n$, it yields a discounted payoff given by a square-integrable random variable $G_n$ defined on a filtered probability space $(\Omega, \mathcal{F}, \mathbb{G} = (\mathcal{G}_n)_{n=0}^N, \mathbb{P})$. We assume that $\mathcal{G}_n$ describes the information available at time $t_n$ and $G_n$ is of the form $g(n, X_n)$ for a measurable function $g: \{0,1,\ldots,N\} \times \mathbb{R}^d \to \mathbb{R}^+$ and a $d$-dimensional $\mathcal{G}$-Markov process $(X_n)_{n=0}^N$. We assume $X_0$ to be deterministic and $\mathbb{P}$ to be an equivalent martingale measure. So that the value of the option at time 0 is given by

$$V = \sup_{\tau \in \mathcal{T}} \mathbb{E}[G_\tau],$$

where $\mathcal{T}$ is the set of all $\mathcal{G}$-stopping times $\tau: \Omega \to \{0,1,\ldots,N\}$. If the option has not been exercised before time $t_n$, its discounted value at that time is

$$V_{t_n} = \text{ess sup}_{\tau \in \mathcal{T}_n} \mathbb{E}[G_\tau | \mathcal{F}_n],$$

2This covers Bermudan options. An American option that is continuously exercisable must be approximated by using a sufficiently fine time grid.

3That is, $X_n$ is $\mathcal{G}_n$-measurable, and $\mathbb{E}[f(X_{n+1}) | \mathcal{F}_n] = \mathbb{E}[f(X_{n+1}) | X_n]$ for all $n \leq N - 1$ and every measurable function $f: \mathbb{R}^d \to \mathbb{R}$ such that $f(X_{n+1})$ is integrable.
where $\mathcal{T}_n$ is the set of all $\mathcal{G}$-stopping times satisfying $n \leq \tau \leq N$.

Obviously, $\tau_N \equiv N$ is optimal for $V_T = G_N$. From there, one can recursively construct the stopping times

$$
\tau_n := \begin{cases} 
n & \text{if } G_n \geq \mathbb{E}[G_{\tau_{n+1}} \mid X_n] \\
n_{n+1} & \text{if } G_n < \mathbb{E}[G_{\tau_{n+1}} \mid X_n].
\end{cases}
$$

(2)

Clearly, $\tau_n$ belongs to $\mathcal{T}_n$, and it can be checked inductively that

$$
V_{t_n} = \mathbb{E}[G_{\tau_n} \mid \mathcal{G}_n] = G_n \lor \mathbb{E}[V_{t_{n+1}} \mid X_n] \quad \text{for all } n \leq N - 1.
$$

(3)

In particular, $\tau_n$ is an optimizer of (1).

Recursion (2) is the theoretical basis of the Longstaff–Schwartz method [26]. Its main computational challenge is the approximation of the conditional expectations $\mathbb{E}[G_{\tau_{n+1}} \mid X_n]$. It is well-known that $\mathbb{E}[G_{\tau_{n+1}} \mid X_n]$ minimizes the mean squared distance $\mathbb{E}[(G_{\tau_{n+1}} - Z)^2]$ over all $\sigma(X_n)$-measurable random variables $Z : \Omega \to \mathbb{R}$; see, e.g., [1]. The Longstaff–Schwartz algorithm approximates $\mathbb{E}[G_{\tau_{n+1}} \mid X_n]$ by projecting $G_{\tau_{n+1}}$ on the linear span of finitely many functions of $X_n$. But it is also possible to project on a different subset. If the subset is given by $c_\theta(X_n)$ for a function family $c_\theta : \mathbb{R}^d \to \mathbb{R}$ parametrized by $\theta$, one can apply the following variant of the Longstaff–Schwartz algorithm:

(i) Simulate paths $(x_n^k)_{n=0}^N, k = 1, \ldots, K$, of the underlying process $(X_n)^N$.

(ii) Set $s_{N}^k \equiv N$ for all $k$.

(iii) For $1 \leq n \leq N - 1$, approximate $\mathbb{E}[G_{\tau_{n+1}} \mid X_n]$ with $c_\theta(X_n)$ by minimizing the sum

$$
\sum_{k=1}^{K} \left( g(s_{n+1}^k, x_{n+1}^k) - c_\theta(x_n^k) \right)^2 \quad \text{over } \theta.
$$

(4)

(iv) Approximate the continuation value function by $C_\theta^n := c_\theta^n \lor 0$ and set

$$
s_n^k := \begin{cases} 
n & \text{if } g(n, x_n^k) \geq C_\theta^n(x_n^k) \\
_{s_{n+1}^k} & \text{otherwise.}
\end{cases}
$$

(5)

(v) Define $\theta_0 := \frac{1}{K} \sum_{k=1}^{K} g(s_1^k, x_1^k)$, and set $C_\theta^{0}$ constantly equal to $\theta_0$.

In this paper we specify $c_\theta$ as a feedforward neural network, which in general, is of the form

$$
a_1^\theta \circ \varphi_{q_1} \circ a_1^\theta \circ \cdots \circ \varphi_{q_{I-1}} \circ a_1^\theta,
$$

(6)

where

4The main difference between this algorithm and the one of Longstaff and Schwartz (2001) is the use of neural networks instead of basis functions. In addition, the sum in (3) is over all simulated paths, whereas Longstaff and Schwartz (2001) focus on the in-the-money paths to save computation effort. Moreover, since we know that continuation values cannot be negative we work with $C_\theta^n = c_\theta^n \lor 0$ instead of $c_\theta^n$. 

3
• \( I \geq 1 \) and \( q_0, q_1, \ldots, q_I \) denote the depth and numbers of nodes in the different layers.

• \( a_{\theta}^1: \mathbb{R}^{q_0} \to \mathbb{R}^{q_1}, \ldots, a_{\theta}^I: \mathbb{R}^{q_{I-1}} \to \mathbb{R}^{q_I} \) are affine functions,

• For \( j \in \mathbb{N} \), \( \varphi_j: \mathbb{R}^{j} \to \mathbb{R}^{j} \) is of the form \( \varphi_j(x_1, \ldots, x_j) = (\varphi(x_1), \ldots, \varphi(x_j)) \) for a given activation function \( \varphi: \mathbb{R} \to \mathbb{R} \).

The components of the parameter \( \theta \) consist of the entries of the matrices \( A_1, \ldots, A_I \) and vectors \( b_1, \ldots, b_I \) appearing in the representation of the affine functions \( a_{\theta}^i x = A_i x + b_i, i = 1, \ldots, I \). So, \( \theta \) lives in \( \mathbb{R}^{q} \) for \( q = \sum_{i=1}^{I} q_i(q_i - 1 + 1) \). To minimize (4) we use a stochastic gradient descent method.

3 Pricing

3.1 Lower bound

Once \( \theta_0, \theta_1, \ldots, \theta_{N-1} \) have been determined, we set \( \Theta = (\theta_0, \ldots, \theta_{N-1}) \) and define

\[
\tau^\Theta := \min \left\{ n \in \{0, 1, \ldots, N - 1\} : g(n, X_n) \geq C^\theta_n(X_n) \right\}, \quad \text{where } \min \emptyset \text{ is understood as } N.
\]

This defines a valid \( \mathcal{G} \)-stopping time. Therefore, \( L = \mathbb{E} g(\tau^\Theta, X_{\tau^\Theta}) \) is a lower bound for the optimal value \( V \). But typically, it is not possible to calculate the expectation exactly. Therefore, we generate simulations \( g^k \) of \( g(\tau^\Theta, X_{\tau^\Theta}) \) based on sample paths \( (x_n^k)^N_{n=0}, k = K+1, \ldots, K+K_L \), of \( (X_n)^N_{n=0} \) generated independently of \( (x_n^k)^N_{n=0}, k = 1, \ldots, K \), and approximate \( L \) with the Monte Carlo average

\[
\hat{L} = \frac{1}{K_L} \sum_{K+1}^{K+K_L} g^k.
\]

Denote by \( z_{\alpha/2} \) the \( 1 - \alpha/2 \) quantile of the standard normal distribution and consider the sample standard deviation

\[
\hat{\sigma}_L = \sqrt{\frac{1}{K_L - 1} \sum_{K+1}^{K+K_L} \left( g^k - \hat{L} \right)^2}.
\]

Then one obtains from the central limit theorem that

\[
\left[ \hat{L} - z_{\alpha/2} \frac{\hat{\sigma}_L}{\sqrt{K_L}}, \infty \right)
\]

is an asymptotically valid \( 1 - \alpha/2 \) confidence interval for \( L \).
3.2 Upper bound, point estimate and confidence intervals

From $\tau^\Theta$, we construct the $\mathbb{G}$-martingale $M_0^\Theta = 0, M_1^\Theta, \ldots, M_N^\Theta$ as in Subsection of [5]. We know from Proposition 7 of [5] that if $(\varepsilon_n)_{n=0}^N$ is a sequence of integrable random variables satisfying $E[\varepsilon_n | \mathcal{G}_n] = 0$ for all $n = 0, 1, \ldots, N$, then

$$U = E\left[\max_{0 \leq n \leq N} (g(n, X_n) - M_n^\Theta - \varepsilon_n)\right]$$

is an upper bound for $V$. As in [5], we use nested simulation to generate realizations $M^k$ of $M_n^\Theta + \varepsilon_n$ along simulated paths $(z^k_n)_{n=0}^N$, $k = 1, \ldots, K_U$, of $(X_n)_{n=0}^N$ and estimate $U$ as

$$\hat{U} = \frac{1}{K_U} \sum_{k=1}^{K_U} \max_{0 \leq n \leq N} (g(n, z^k_n) - M^k_n).$$

Our point estimate of $V$ is

$$\hat{V} = \hat{L} + \hat{U}.$$

The sample standard deviation of the estimator $\hat{U}$, given by

$$\hat{\sigma}_U = \sqrt{\frac{1}{K_U - 1} \sum_{k=1}^{K_U} \left( \max_{0 \leq n \leq N} (g(n, z^k_n) - M^k_n) - \hat{U} \right)^2},$$

can be used together with the one-sided confidence interval (8) to construct the asymptotically valid two-sided $1 - \alpha$ confidence interval

$$[\hat{L} - z_{\alpha/2} \hat{\sigma}_L / \sqrt{K_L}, \hat{L} + z_{\alpha/2} \hat{\sigma}_U / \sqrt{K_U}]$$

for the true value $V$; see [5].

4 Hedging

We now consider $e \in \mathbb{N}$ tradable financial instruments as hedging instruments and assume the candidate optimal exercise strategy $\tau^\Theta$ derived in Section 2 does not stop at time 0. We fix a positive integer $M$ and introduce a time grid $0 = u_1 < u_2 < \cdots < u_{NM}$ such that $u_{nM} = t_n$ for all $n = 0, 1, \ldots, N$. We suppose that the information available at time $u_m$ is described by $\mathcal{F}_m$, where $\mathbb{F} = (\mathcal{F}_m)_{m=0}^{MN}$ is a filtration satisfying $\mathcal{F}_{nM} = \mathcal{G}_n$ for all $n$. If the hedging instruments

---

5The use of nested simulation ensures that $M^k_n$ are unbiased estimates of $M_n^\Theta$, which is crucial for the validity of the upper bound. We do not use the approximate continuation value functions $C^{\tau_n}$ trained in Section 2 to compute upper bounds.
To do that we approximate the functions $h \approx (h_m)_{m=0}^{NM-1}$ such that $Y_{m+1}^M = X_m$ for all $n = 0, \ldots, N$. A hedging strategy consists of a sequence $h = (h_m)_{m=0}^{NM-1}$ of functions $h_m : \mathbb{R}^d \to \mathbb{R}_e$ specifying the holdings in $P^1, \ldots, P_e$ at time $u_m$. The resulting discounted gains at time $u_m$ are given by

$$(h \cdot P)_{u_m} := \sum_{j=0}^{m-1} h_j(Y_j) \cdot (p_{j+1}(Y_{j+1}) - p_j(Y_j)) := \sum_{j=0}^{m-1} \sum_{i=1}^e h_j^i(Y_j) \cdot (p_{i+1}^j(Y_{j+1}) - p_i^j(Y_j)).$$

### 4.1 Hedging until the first possible exercise date

For a typical Bermudan option, the time between two possible exercise dates $t_n - t_{n-1}$ might range between a week and several months. In case of an American option, we choose $t_n = n \Delta$ for a small amount of time $\Delta$ such as a day or half a day. If $\tau^\Theta$ does not stop at time 0, we only compute the hedge until time $t_1$. If the option is still alive at time $t_1$, the hedge can then be computed until time $t_2$ and so on. To construct a hedge from time 0 to $t_1$, we approximate the time-$t_1$ value of the option with $V_{t_1}^{\theta_1} = u^{\theta_1}(X_1)$ for the function $u^{\theta_1}(x) = g(1, x) \lor C^{\theta_1}(x)$, where $C^{\theta_1} : \mathbb{R}^d \to \mathbb{R}$ is the approximate time-$t_1$ continuation value function estimated in Section 2. Then we search for hedging positions $h_m, m = 0, 1, \ldots, M - 1$, that minimize the mean squared error

$$\mathbb{E} \left[ (\hat{V} + (h \cdot P)_{t_1} - V_{t_1}^{\theta_1})^2 \right].$$

To do that we approximate the functions $h_m$ with neural networks $h^\lambda : \mathbb{R}^d \to \mathbb{R}_e$ of the form (6) and try to find parameters $\lambda_0, \ldots, \lambda_{M-1}$ which minimize

$$\sum_{k=1}^{K_H} \left( \hat{V} \sum_{m=0}^{M-1} h^\lambda_m(y^k_m) \cdot p_{m+1}(y^k_{m+1}) - p_m(y^k_m) \cdot v^{\theta_1}(y^k_M) \right)^2$$

for simulations $(y^k_m)_{m=0}^M, k = 1, \ldots, K_H$ of $(Y_m^M)_{m=0}^M$. We train the networks $h^\lambda_0, \ldots, h^\lambda_{M-1}$ together, again using a stochastic gradient descent method.

Once $\lambda_0, \ldots, \lambda_{M-1}$ have been determined, we assess the quality of the hedge by simulating new realizations $(y^k_m)_{m=0}^M, k = K_H + 1, \ldots, K_H + K_E$ of $(Y_m^M)_{m=0}^M$ and calculating the empirical intermediate hedging shortfall

$$\frac{1}{K_E} \sum_{k=K_H+1}^{K_H+K_E} \left( \hat{V} + \sum_{m=0}^{M-1} h^\lambda_m(y^k_m) \cdot p_{m+1}(y^k_{m+1}) - p_m(y^k_m) \cdot v^{\theta_1}(y^k_M) \right)$$

for simulations $(y^k_m)_{m=0}^M, k = 1, \ldots, K_H$.

---

6 That is, $Y_m$ is $\mathcal{F}_m$-measurable and $\mathbb{E}[f(Y_{m+1}) \mid \mathcal{F}_m] = \mathbb{E}[f(Y_{m+1}) \mid Y_m]$ for all $m \leq NM - 1$ and every measurable function $f : \mathbb{R}^d \to \mathbb{R}$ such that $f(X_{m+1})$ is integrable.

7 We again choose networks with depth $I = 3$ and activation function $\varphi = \tanh$.

8 Independent of $(y^k_m)_{m=0}^M, k = 1, \ldots, K_H$.
4.2 Hedging until the exercise time

Alternatively, one can precompute the whole hedging strategy from time 0 to T and then use it until the option is exercised. In order to do that we introduce the functions
\[ v^{\theta_n}(x) := g(n, x) \lor C^{\theta_n}(x), \quad x \in \mathbb{R}^d, \]
and hedge the difference \( v^{\theta_n}(Y_{nM}) - C^{\theta_{n-1}}(Y_{(n-1)M}) \) on each of the time intervals \([t_{n-1}, t_n]\), \(n = 1, \ldots, N\), separately. This can be done like in Subsection 4.1 except that now we have to simulate complete paths \((y_{m}^k)_{m=0}^{N M}\) of \((Y_m^M)_{m=0}^{N M}\), \(k = 1, \ldots, K_H\), and then for all \(n = 1, \ldots, N\), find parameters \(\lambda_{(n-1)M}, \ldots, \lambda_{nM-1}\) that minimize
\[
\sum_{k=1}^{K_H} \left( C^{\theta_{n-1}}(y_{(n-1)M}^k) + \sum_{m=(n-1)M}^{nM-1} h^\lambda_m(y_{m}^k) \cdot \left( p_{m+1}(y_{m+1}^k) - p_m(y_{m}^k) \right) - v^{\theta_n}(y_{nM}^k) \right)^2.
\]

Once the hedging strategy has been trained, we simulate independent samples \((y_{m}^k)_{m=0}^{N M}, k = K_H + 1, \ldots, K_H + K_E\), of \((Y_m^M)_{m=0}^{N M}\) and denote the realization of \(\tau^\theta\) along each sample path \((y_{m}^k)_{m=0}^{N M}\) by \(\tau^k\). The corresponding empirical hedging shortfall is given by
\[
\frac{1}{K_H+K_E} \sum_{k=K_H+1}^{K_H+K_E} \left( \hat{V} + \sum_{m \leq \tau^k N} h^\lambda_m(y_{m}^k) \cdot \left( p_{m+1}(y_{m+1}^k) - p_m(y_{m}^k) \right) - g(\tau^k, X+\tau^k) \right). \tag{11}
\]

5 Example

In this section we study a Bermudan max-call option on \(d\) financial assets with risk-neutral price dynamics
\[
S_t^i = s_0^i \exp\left(\left[r - \delta_i - \sigma_i^2/2\right]t + \sigma_i W_t^i\right), \quad i = 1, 2, \ldots, d, \tag{12}
\]
for a risk-free interest rate \(r \in \mathbb{R}\), initial values \(s_0^i \in (0, \infty)\), dividend yields \(\delta_i \in [0, \infty)\), volatilities \(\sigma_i \in (0, \infty)\) and a \(d\)-dimensional Brownian motion \(W\) with constant instantaneous correlations \(\rho_{ij} \in \mathbb{R}\) between different components \(W^i\) and \(W^j\). The option has payoff \((\max_{1 \leq i \leq d} S_t^i - K)^+\) for a strike price \(K \in \mathbb{R}_+\) and can be exercised at one of finitely many times \(0 = t_0 < t_1 < \cdots < t_N = T\).

For notational simplicity, we assume in the following that \(t_n = nT/N\) for \(n = 0, 1, \ldots, N\), and all assets have the same\(^9\) characteristics; that is, \(s_0^i = s_0, \delta_i = \delta\) and \(\sigma_i = \sigma\) for all \(i = 1, \ldots, d\).

---

9 The computations were run on a NVIDIA GeForce GTX 1080 GPU with 1974 MHz core clock and 8 GB GDDR5X memory with 1809.5 MHz clock rate. As underlying system we used an Intel Core i7-6800K 3.4 GHz CPU with 64 GB DDR4-2133 memory running Tensorflow 1.11 on Ubuntu 16.04.

10 That is, \(E(W_t^i - W_s^i | W_s^j - W_s^i) = \rho_{ij}(t - s)\) for all \(i \neq j\) and \(s < t\).

11 Simulation based methods work for any price dynamics that can efficiently be simulated. Prices of max-call options on underlying assets with different price dynamics were calculated in \(^9\) and \(^5\).
Table 1: Price estimates for max-call options on 5 and 10 symmetric assets for parameter values of $r = 5\%$, $\delta = 10\%$, $\sigma = 20\%$, $\rho = 0$, $K = 100$, $T = 3$, $N = 9$. $t_L$ is the number of seconds it took to train $\tau^\Theta$ and compute $\hat{L}$. $t_U$ is the computation time for $\hat{U}$ in seconds. 95% CI is the 95% confidence interval (9). The last column lists the 95% confidence intervals computed in [5].

| $d$ | $s_0$ | $\hat{L}$ | $t_L$ | $\tilde{U}$ | $t_U$ | Point Est. | 95% CI | DOS 95% CI |
|-----|-------|--------|-----|---------|-----|-----------|-------|-----------|
| 5   | 90    | 16.634 | 542.3 | 16.641  | 9.6 | 16.638    | [16.618, 16.656] | [16.633, 16.648] |
| 5   | 100   | 26.157 | 541.5 | 26.162  | 9.7 | 26.160    | [26.138, 26.183] | [26.138, 26.174] |
| 5   | 110   | 36.787 | 541.9 | 36.802  | 9.8 | 36.795    | [36.766, 36.824] | [36.745, 36.789] |
| 10  | 90    | 26.279 | 569.4 | 26.287  | 10.5| 26.283    | [26.261, 26.305] | [26.189, 26.289] |
| 10  | 100   | 38.373 | 568.9 | 38.374  | 10.4| 38.373    | [38.352, 38.395] | [38.300, 38.367] |
| 10  | 110   | 50.907 | 569.1 | 50.928  | 10.2| 50.917    | [50.884, 50.951] | [50.834, 50.937] |

5.1 Pricing results

Let us denote $X_n = S_{t_n}$, $n = 0, 1, \ldots, N$. Then the price of the option is given by

$$
\sup_{\tau} \mathbb{E}\left[e^{-r\tau} \left(\max_{1 \leq i \leq d} X^i_{\tau} - K\right)\right],
$$

where the supremum is over all stopping times $\tau: \Omega \to \{0, 1, \ldots, N\}$ with respect to the filtration generated by $(X_n)_{n=0}^N$. The option payoff does not carry additional information. But the training of the continuation values worked more efficiently when we used it as an additional feature. So instead of $X_n$ we simulated the extended state process $\hat{X}_n = (X_1^n, \ldots, X_d^n, X_{d+1}^n)$ for $X_{d+1}^n = e^{-\gamma} \left(\max_{1 \leq i \leq d} X^i_n - K\right)\right)$

to train the continuation value functions $c^\theta_n$, $n = 1, \ldots, N - 1$. The network $c^\theta: \mathbb{R}^{d+1} \to \mathbb{R}$ was chosen of the form (6) with depth $I = 3$ (two hidden layers), $d + 50$ nodes in each hidden layer and activation function $\varphi = \tanh$.

Our results for $\hat{L}$, $\hat{U}$, $\hat{V}$ and 95% confidence intervals for different specifications of the model parameters are reported in Table 1. It can be seen that to achieve a pricing accuracy comparable to the more direct methods of [5] and [6], the networks used in the construction of the candidate optimal stopping strategy have to be trained for a longer time.

5.2 Hedging results

Suppose the hedging portfolio can be rebalanced at the times $u_m = mT/(NM)$, $m = 0, 1, \ldots, NM$, for a positive integer $M$. We assume dividends payed by shares of $S^i$ held in the hedging portfolio are continuously reinvested in $S^i$. This results in the adjusted discounted price processes

$$
P^i_{u_m} = s_0 \exp(\sigma W^i_{u_m} - \sigma^2 u_m/2), \quad m = 0, 1, \ldots, NM.
$$
We set $P_{im} = Y_{im}$. To learn the hedging strategy, we trained neural networks $h_m : \mathbb{R}^d \to \mathbb{R}^d$, $m = 0, \ldots, NM - 1$, of the form \eqref{eq:neural_network} with depth $I = 3$, $d + 50$ nodes in each of the hidden layers and activation function $\varphi = \tanh$. Again, we used mini-batch stochastic gradient descent with Xavier initialization, batch normalization and Adam updating.

Table 2 reports the intermediate hedging shortfall \eqref{eq:intermediate_shortfall} and the total hedging shortfall \eqref{eq:total_shortfall} for various numbers $M$ of rebalancing times between two different exercise dates $t_{n-1}$ and $t_n$.

6 Conclusion

In this article we used deep neural networks to price and hedge American-style options. In a first step our method uses a neural network version of the Longstaff–Schwartz algorithm to estimate continuation values and design a candidate optimal stopping rule. The learned stopping rule immediately gives a low-biased estimate of the price. In addition, it can be used to construct an approximate solution of the dual martingale problem of \cite{28} and \cite{18}. This gives a high-biased estimate and confidence intervals for the price. To achieve the same pricing accuracy as the more direct approaches of \cite{5} and \cite{6}, we had to train the neural network approximations of the continuation values for a longer time. But computing approximate continuation values has the advantage that they can be used to break the hedging problem into a sequence of subproblems that compute the hedge only from one possible exercise date to the next.

References

[1] B. Bru and H. Heinich. Meilleures approximations et médianes conditionnelles. Annales de l’I.H.P. Probabilités et Statistiques, 21(3):197–224, 1985.

[2] L. Andersen. A simple approach to the pricing of Bermudan swaptions in the multifactor LIBOR market model. The Journal of Computational Finance, 3(2):5–32, 2000.

[3] V. Bally, G. Pagès and J. Printems. A quantization tree method for pricing and hedging multidimensional American options. Mathematical Finance, 15(1):119–168, 2005.

[4] J. Barraquand and D. Martineau. Numerical valuation of high dimensional multivariate American securities. The Journal of Financial and Quantitative Analysis, 30(3):383–405, 1995.

[5] S. Becker, P. Cheridito und A. Jentzen. Deep optimal stopping. Journal of Machine Learning Research, 20(74), 2019.

[6] S. Becker, P. Cheridito, A. Jentzen und T. Welti. Solving high-dimensional optimal stopping problems using deep learning Arxiv Preprint, 2019.
| $d$ | $s_0$ | $M$ | IHS | $\frac{\text{IHS}}{\hat{V}}$ | CT1 | HS | $\frac{\text{HS}}{\hat{V}}$ | CT2 |
|-----|-----|-----|-----|-----------------|-----|----|-----------------|-----|
| 5   | 90  | 12  | 0.182 | 1.1% | 252 | 0.682 | 4.1% | 2,370 |
| 5   | 90  | 24  | 0.130 | 0.8% | 385 | 0.495 | 3.0% | 3,652 |
| 5   | 90  | 48  | 0.094 | 0.6% | 781 | 0.370 | 2.2% | 6,930 |
| 5   | 90  | 96  | 0.070 | 0.4% | 1534| 0.293 | 1.8% | 1,343 |
| 5   | 100 | 12  | 0.221 | 0.8% | 253 | 0.788 | 3.0% | 2,385 |
| 5   | 100 | 24  | 0.155 | 0.6% | 386 | 0.571 | 2.2% | 3,659 |
| 5   | 100 | 48  | 0.109 | 0.4% | 782 | 0.425 | 1.6% | 6,933 |
| 5   | 100 | 96  | 0.078 | 0.3% | 1,538| 0.337 | 1.3% | 1,351 |
| 5   | 110 | 12  | 0.266 | 0.5% | 254 | 0.886 | 2.4% | 2,338 |
| 5   | 110 | 24  | 0.188 | 0.4% | 388 | 0.642 | 1.7% | 3,609 |
| 5   | 110 | 48  | 0.137 | 0.3% | 784 | 0.477 | 1.0% | 6,867 |
| 5   | 110 | 96  | 0.097 | 0.3% | 1,536| 0.375 | 1.0% | 1,329 |
| 10  | 90  | 12  | 0.198 | 0.8% | 287 | 0.903 | 3.4% | 2,694 |
| 10  | 90  | 24  | 0.147 | 0.6% | 454 | 0.712 | 2.7% | 4,165 |
| 10  | 90  | 48  | 0.115 | 0.4% | 920 | 0.609 | 2.3% | 8,135 |
| 10  | 90  | 96  | 0.103 | 0.4% | 1,836| 0.584 | 2.2% | 1,600 |
| 10  | 100 | 12  | 0.231 | 0.6% | 288 | 1.109 | 2.7% | 2,712 |
| 10  | 100 | 24  | 0.172 | 0.4% | 448 | 0.795 | 2.1% | 4,189 |
| 10  | 100 | 48  | 0.134 | 0.3% | 908 | 0.684 | 1.8% | 7,943 |
| 10  | 100 | 96  | 0.116 | 0.3% | 1,819| 0.651 | 1.7% | 1,571 |
| 10  | 110 | 12  | 0.263 | 0.5% | 287 | 1.130 | 2.2% | 2,653 |
| 10  | 110 | 24  | 0.196 | 0.4% | 449 | 0.883 | 1.7% | 4,113 |
| 10  | 110 | 48  | 0.154 | 0.3% | 909 | 0.747 | 1.5% | 8,037 |
| 10  | 110 | 96  | 0.134 | 0.3% | 1,836| 0.699 | 1.4% | 15,822 |

Table 2: Empirical hedging shortfalls for 5 and 10 underlying assets and different numbers $M$ of rehedging times between consecutive possible exercise times $t_{n-1}$ and $t_n$. The values of the parameters $r$, $\delta$, $\sigma$, $\rho$, $K$, $T$ and $N$ were chosen as in Table 1. IHS is the intermediate hedging shortfall (10) and HS the total hedging shortfall (11). They were estimated using $K_E = 4,096,000$ sample paths. $\hat{V}$ is our point estimate of the price from Table 1. CT1 is the computation time in seconds for training the hedging strategy from time 0 to $t_1 = T/N$. CT2 is the number of seconds it took to train the complete hedging strategy from time 0 to $T$. 

10
[7] B. Bouchard and X. Warin. Monte-Carlo valuation of American options: facts and new algorithms to improve existing methods. In emphNumerical Methods in Finance Vol. 12 of Springer Proc. Math., 215–255, 2012.

[8] S.J. Berridge and J.M. Schumacher. An irregular grid approach for pricing high-dimensional American options. Journal of Computational and Applied Mathematics, 222(1):94–111, 2008.

[9] M. Broadie and M. Cao. Improved lower and upper bound algorithms for pricing American options by simulation. Quantitative Finance, 8(8):845–861, 2008.

[10] H. Buehler, L. Gonon, J. Teichmann and B. Wood. Deep hedging. Quantitative Finance, 19(8), 2019.

[11] J.F. Carriere. Valuation of the early-exercise price for options using simulations and non-parametric regression. Insurance: Mathematics and Economics, 19(1):19–30, 1996.

[12] Y. Chen and J.W.L. Wan. Deep neural network framework based on backward stochastic differential equations for pricing and hedging American options in high dimensions Arxiv Preprint, 2019.

[13] D. Egloff, M. Kohler and N. Todorovic. A dynamic look-ahead Monte Carlo algorithm for pricing Bermudan options. The Annals of Applied Probability, 17(4):1138–1171, 2007.

[14] P.A. Forsyth and K.R. Vetzal. Quadratic convergence for valuing American options using a penalty method. SIAM J. Sci. Comput., 23, 2095–2122, 2002.

[15] D. García. Convergence and biases of Monte Carlo estimates of American option prices using a parametric exercise rule. Journal of Economic Dynamics and Control, 27(10):1855–1879, 2003.

[16] X. Glorot and Y. Bengio. Understanding the difficulty of training deep feedforward neural networks. In Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics, PMLR, 9:249–256, 2010.

[17] J. Han, A. Jentzen and W. E. Solving high-dimensional partial differential equations using deep learning. PNAS, 115(34), 8505–8510, 2018.

[18] M.B. Haugh and L. Kogan. Pricing American options: a duality approach. Operations Research, 52(2):258–270, 2004.

[19] J.C. Hull. Options, Futures and Other Derivatives. Pearson/Prentice Hall, 2003.

[20] S. Ioffe and C. Szegedy. Batch normalization: accelerating deep network training by reducing internal covariate shift. In Proceedings of the 32nd International Conference on Machine Learning, PMLR, 37:448–456, 2015.
[21] S. Jain and C.W. Oosterlee. The stochastic grid bundling method: efficient pricing of Bermudan options and their Greeks. *Applied Mathematics and Computation*, 269:412–431, 2015.

[22] D.P. Kingma and J. Ba. Adam: A method for stochastic optimization. *International Conference on Learning Representations*, 2015.

[23] M. Kohler, A. Krzyżak and N. Todorovic. Pricing of high-dimensional American options by neural networks. *Mathematical Finance*, 20(3):383–410, 2010.

[24] A. Kolodko and J. Schoenmakers. Iterative construction of the optimal Bermudan stopping time. *Finance and Stochastics*, 10(1):27–49, 2006.

[25] B. Lapeyre and J. Lelong. Neural network regression for Bermudan option pricing *ArXiv Preprint*, 2019.

[26] F.A. Longstaff and E.S. Schwartz. Valuing American options by simulation: a simple least-squares approach. *The Review of Financial Studies*, 14(1):113–147, 2001.

[27] C. Reisinger and J.H. Witte. On the use of policy iteration as an easy way of pricing American options. *SIAM J. Financial Math.*, 3, 459–478, 2012.

[28] L.C.G. Rogers. Monte Carlo valuation of American options. *Mathematical Finance*, 12(3):271–286, 2002.

[29] L.C.G. Rogers. Dual valuation and hedging of Bermudan options. *SIAM Journal on Financial Mathematics*, 1(1):604–608, 2010.

[30] J. Sirignano and K. Spiliopoulos. DGM: A deep learning algorithm for solving partial differential equations. *Journal of Computational Physics*, 375:1339–1364, 2018.

[31] J.A. Tilley. Valuing American options in a path simulation model. *Transactions of the Society of Actuaries*, 45:83–104, 1993.

[32] J.N. Tsitsiklis and B. Van Roy. Regression methods for pricing complex American-style options. *IEEE Transactions on Neural Networks*, 12(4):694–703, 2001.