Deep Recurrent Optimal Stopping: Supplementary Material

We provide an outline of the supplement that complements our paper.

Appendix A: provides proofs of all Theorems in the paper.

Appendix B: considers formulations of stopping problems specified in terms of costs instead of rewards and shows how to transform these into a consistent reward formulation.

Appendix C: provides discussion and pseudo-code to implement the temporal loss used to learn fitted Q-iteration policies for DNN-FQI and RNN-FQI

Appendix D: includes further details and discussion of the experiments, including model hyper-parameter settings, numerical results with confidence intervals, details of compute environment, model sizes, train and inference times, etc. In Appendix E.3, we also include a new experiment that compares OSPG to state-of-the-art PDE benchmarks in the pricing of American options, which is a continuous-time optimal stopping problem.

Appendix E: includes a treatment of baseline subtraction in the context of optimal stopping policy gradients (OSPG).

A Appendix A: Proofs

A.1 Proof of Lemma 3.0.1

Proof. The Lemma follows from the Bayes net trajectory model of Figure 1a, and the special structure of OS action trajectories: $A_\tau = 1$ and $A_n = 0, \forall n < \tau$. For any OS state-action trajectory, we have:

$$\mathbb{P}(\mathbf{A}_\tau, \mathbf{S}_H) = \underbrace{\mathbb{P}(\mathbf{S}_0) \prod_{j=1}^{H} \mathbb{P}(\mathbf{S}_j | \mathbf{S}_{j-1})}_{\mathbb{P}(\mathbf{S}_H)} \prod_{n=0}^{\tau} \mathbb{P}(A_n | \mathbf{S}_n) = \mathbb{P}(\mathbf{S}_H) \prod_{n=0}^{\tau} \mathbb{P}(A_n | \mathbf{S}_n) \tag{15}$$

Therefore, conditioning on the state trajectory, we have $\mathbb{P}(\mathbf{A}_\tau | \mathbf{S}_H) = \prod_{n=0}^{\tau} \mathbb{P}(A_n | \mathbf{S}_n)$. Note that by the structure of finite-horizon OS trajectories $\mathbf{A}_\tau$ is a sequence of continue actions terminated by a stop action at τ. Thus there is a bijective mapping between stopping times and complete action trajectories given by:

$$\kappa : \{0, 1, 2, \cdots, H\} \mapsto \{1, 01, 001, \cdots \underbrace{00 \cdots 0}_{H-1 \;\; 0s} 1\} \tag{16}$$

So $\mathbb{P}(\mathbf{A}_\tau | \mathbf{S}_H) = \mathbb{P}(\kappa(\tau) | \mathbf{S}_H) = \mathbb{P}(\tau | \mathbf{S}_H)$. Consider a trajectory stopping at $\tau = j$ and recall that our stochastic stopping policy is defined as $\phi_j(\mathbf{S}_j) : \mathbb{P}(A_j = 1 | \mathbf{S}_j)$.

Thus, if $j = 0$:

$$\mathbb{P}(\tau = 0 | \mathbf{S}_H) = \mathbb{P}(\mathbf{A}_0 | \mathbf{S}_H) = \mathbb{P}(A_0 = 1 | \mathbf{S}_0) = \phi_0(\mathbf{S}_0) \tag{17}$$

If $0 < j < H$:

$$\begin{aligned} \mathbb{P}(\tau = j | \mathbf{S}_H) &= \mathbb{P}(A_0 = 0, \cdots, A_{j-1} = 0, A_j = 1 | \mathbf{S}_H) \\ &= \mathbb{P}(A_j = 1 | \mathbf{S}_H) \prod_{n=0}^{j-1} \mathbb{P}(A_n = 0 | \mathbf{S}_H) \\ &= \phi_j(\mathbf{S}_j) \prod_{n=0}^{j-1} (1 - \phi_n(\mathbf{S}_n)) \end{aligned} \tag{18}$$

Finally, if $j = H$:

$$\begin{aligned}
\mathbb{P}(\tau = H|\mathbf{S}_H) &= \mathbb{P}(A_0 = 0, \cdots, A_{H-1} = 0, A_H = 1|\mathbf{S}_H) \\
&= \mathbb{P}(A_H = 1|\mathbf{S}_H) \prod_{n=0}^{H-1} \mathbb{P}(A_n = 0|\mathbf{S}_H) \\
&= \phi_H(\mathbf{S}_H) \prod_{n=0}^{H-1} (1 - \phi_n(\mathbf{S}_n)) \\
&= \prod_{n=0}^{H-1} (1 - \phi_n(\mathbf{S}_n)) \qquad (19)
\end{aligned}$$

where we have used the fact that in the finite horizon setting $\phi_H(\mathbf{S}_H) := 1$ by definition.

We may verify $\sum_{j=0}^{H} \mathbb{P}(\tau = j|\mathbf{S}_H) = 1$ since:

$$\begin{aligned}
1 - \sum_{j=0}^{H-1} \mathbb{P}(\tau = j|\mathbf{S}_H) &= 1 - \phi_0(\mathbf{S}_0) - \sum_{j=0}^{H-1} \phi_j(\mathbf{S}_j) \prod_{n=0}^{j-1} (1 - \phi_n(\mathbf{S}_n)) \\
&= (1 - \phi_0(\mathbf{S}_0)) \left[1 - \phi_1(\mathbf{S}_1) - \sum_{j=1}^{H-1} \phi_j(\mathbf{S}_j) \prod_{n=1}^{j-1} (1 - \phi_n(\mathbf{S}_n)) \right] \\
&= (1 - \phi_0(\mathbf{S}_0))(1 - \phi_1(\mathbf{S}_1)) \times \\
&\quad \left[1 - \phi_2(\mathbf{S}_2) - \sum_{j=2}^{H-1} \phi_j(\mathbf{S}_j) \prod_{n=2}^{j-1} (1 - \phi_n(\mathbf{S}_n)) \right] \\
&\vdots \\
&= \prod_{n=0}^{H-1} (1 - \phi_n(\mathbf{S}_n) = \mathbb{P}(\tau = H|\mathbf{S}_H)
\end{aligned}$$

Further, since τ is a stopping time random variable we have:

$$\mathbb{P}(\tau = j|\mathbf{S}_H) = \mathbb{P}(\tau = j|\mathbf{S}_j) := \psi_j(\mathbf{S}_j) \qquad (20)$$

This is because stopping time random variables have the property that $\mathbb{1}(\tau = j)$ is a function of $\mathbf{S}_j$. So we can determine if $\tau = j$ or not by only considering $\mathbf{S}_j$ making the event $\{\tau = j\}$ conditionally independent of $S_{j+k}, \forall k > 0$ given $\mathbf{S}_j$[35]. This completes the proof. ☐

A.2 Proof of Theorem 3.1

Proof. We apply Jensen's inequality to the objective $J_{\text{WML}}(\boldsymbol{\theta})$ to obtain a lower bound:

$$J_{\text{WML}}(\boldsymbol{\theta}) = \sum_{i=1}^{N} \tilde{w}_i \log \sum_{j=0}^{H} \psi_j^{\boldsymbol{\theta}}(\mathbf{s}_{ij}) \tilde{r}_{ij} \geq \sum_{i=1}^{N} \sum_{j=0}^{H} \tilde{w}_i q_{ij} \log \left[\frac{\psi_j^{\boldsymbol{\theta}}(\mathbf{s}_{ij}) \tilde{r}_{ij}}{q_{ij}} \right] := J(\mathbf{Q}, \boldsymbol{\theta}) \qquad (21)$$

where $\mathbf{Q} = [q_{ij}]$ is any row-stochastic matrix satisfying $q_{ij} \geq 0, \forall i, j$ and $\sum_{j=0}^{H} q_{ij} = 1$. Starting with a given $\boldsymbol{\theta}^{(0)}$ the lower bound is maximized (Jensen's inequality becomes an equality) when $\mathbf{Q} = \mathbf{Q}^{(0)} := [q_{ij}^{(0)}]$ such that:

$$q_{ij}^{(0)} = \frac{\psi_j^{\boldsymbol{\theta}^{(0)}}(\mathbf{s}_{ij}) \tilde{r}_{ij}}{\sum_{j=0}^{H} \psi_j^{\boldsymbol{\theta}^{(0)}}(\mathbf{s}_{ij}) \tilde{r}_{ij}} \qquad (22)$$

This is the E-step in Theorem 3.1. Note that the E-step does not change the objective, so $J_{\text{WML}}(\boldsymbol{\theta}^{(0)}) = J(\mathbf{Q}^{(0)}, \boldsymbol{\theta})$. By ignoring terms that do not depend on $\boldsymbol{\theta}$, maximizing $J(\mathbf{Q}^{(0)}, \boldsymbol{\theta})$ w.r.t. $\boldsymbol{\theta}$ can be seen as equivalent to maximizing

$$J_M(\boldsymbol{\theta}) = \sum_{i=1}^{N} \sum_{j=0}^{H} \tilde{w}_i q_{ij} \log \psi_j^{\boldsymbol{\theta}}(\mathbf{s}_{ij}) \qquad (23)$$

This is the M-step. By the lower bound established in equation (21) and since the M-step maximizes $J(\mathbf{Q}^{(0)}, \boldsymbol{\theta})$ to obtain $\boldsymbol{\theta}^{(1)}$, we have $J_{\text{WML}}(\boldsymbol{\theta}^{(1)}) \geq J(\mathbf{Q}^{(0)}, \boldsymbol{\theta}^{(1)}) \geq J(\mathbf{Q}^{(0)}, \boldsymbol{\theta}^{(0)}) = J_{\text{WML}}(\boldsymbol{\theta}^{(0)})$. Therefore a round of E-M results in either an increase or no change in the objective. Since the WML objective is upper-bounded by $N \log H$, the monotone increasing sequence $J_{\text{WML}}(\boldsymbol{\theta}^{(k)})$ converges to a local maximum of the objective. $\square$

A.3 Proof of Corollary 3.1.1

Proof. It suffices to show that the W-step also increases the objective $J_{\text{WML}}(\tilde{\mathbf{w}}, \boldsymbol{\theta})$.

We maximize $J_{\text{WML}}(\tilde{\mathbf{w}}, \boldsymbol{\theta})$ w.r.t. $\tilde{\mathbf{w}}$, subject to the constraint $\sum_{i=1}^{N} \tilde{w}_i = 1$, resulting the following Lagrangian.

$$\mathcal{L}(\tilde{\mathbf{w}}, \lambda) = \sum_{i=1}^{N} \tilde{w}_i \log \sum_{j=0}^{H} \psi_j^{\boldsymbol{\theta}}(\mathbf{s}_{ij}) \tilde{r}_{ij} - \sum_{i=1}^{N} \tilde{w}_i \log \frac{\tilde{w}_i}{\tilde{r}_i} + \lambda \left(1 - \sum_{i=1}^{N} \tilde{w}_i\right) \tag{24}$$

Taking partial derivative w.r.t. $\tilde{w}_i$ and λ and setting to zero, we have:

$$\frac{\partial \mathcal{L}(\tilde{\mathbf{w}}, \lambda)}{\partial \tilde{w}_i} = \log \sum_{j=0}^{H} \psi_j^{\boldsymbol{\theta}}(\mathbf{s}_{ij}) \tilde{r}_{ij} + \log \tilde{r}_i - (1 + \log \tilde{w}_i) - \lambda = 0 \tag{25}$$

$$\frac{\partial \mathcal{L}(\tilde{\mathbf{w}}, \lambda)}{\partial \lambda} = \left(1 - \sum_{i=1}^{N} \tilde{w}_i\right) = 0 \tag{26}$$

Noting that $\tilde{r}_i := \frac{\sum_{j=0}^{H} r_{ij}}{\sum_{i=1}^{N} \sum_{j=0}^{H} r_{ij}}$ and $\tilde{r}_{ij} := \frac{r_{ij}}{\sum_{j=0}^{H} r_{ij}}$ and simplifying, we have:

$$1 + \lambda = \log \left[\frac{\sum_{j=0}^{H} \psi_j^{\boldsymbol{\theta}}(\mathbf{s}_{ij}) r_{ij}}{r \tilde{w}_i}\right] \tag{27}$$

where $r = \sum_{i=1}^{N} \sum_{j=0}^{H} r_{ij}$. Exponentiation of both sides and cross-multiplication results in:

$$[r \exp{(1 + \lambda)}]\, \tilde{w}_i = \sum_{j=0}^{H} \psi_j^{\boldsymbol{\theta}}(\mathbf{s}_{ij}) r_{ij} \tag{28}$$

Summing over i, we have:

$$r \exp{(1 + \lambda)} = \sum_{i=1}^{N} \sum_{j=0}^{H} \psi_j^{\boldsymbol{\theta}}(\mathbf{s}_{ij}) r_{ij} \tag{29}$$

Substituting back in equation (28), we finally have:

$$\tilde{w}_i^* = \frac{\sum_{j=0}^{H} \psi_j^{\boldsymbol{\theta}}(\mathbf{s}_{ij}) r_{ij}}{\sum_{i=1}^{N} \sum_{j=0}^{H} \psi_j^{\boldsymbol{\theta}}(\mathbf{s}_{ij}) r_{ij}} \tag{30}$$

Thus the maximizing $\tilde{\mathbf{w}}^*$ is exactly the W-step. Therefore, the form of the W-step ensures $J_{\text{WML}}(\tilde{\mathbf{w}}^{(k)}, \boldsymbol{\theta}^{(k)}) \geq J_{\text{WML}}(\tilde{\mathbf{w}}^{(k-1)}, \boldsymbol{\theta}^{(k)}) \geq J_{\text{WML}}(\tilde{\mathbf{w}}^{(k-1)}, \boldsymbol{\theta}^{(k-1)})$. Thus we have a monotone increasing and bounded sequence $J_{\text{WML}}(\tilde{\mathbf{w}}^{(k)}, \boldsymbol{\theta}^{(k)})$ converging to a local maximum of $J_{\text{WML}}(\tilde{\mathbf{w}}, \boldsymbol{\theta})$. $\square$

A.4 Proof of Theorem 4.1

Proof. We use the log-derivative trick [42]. $\nabla_{\boldsymbol{\theta}} \psi_j^{\boldsymbol{\theta}}(\mathbf{s}_j) = \psi_j^{\boldsymbol{\theta}}(\mathbf{s}_j) \nabla_{\boldsymbol{\theta}} \log \psi_j^{\boldsymbol{\theta}}(\mathbf{s}_j)$. So :

$$\nabla_{\boldsymbol{\theta}} J_{OS}(\boldsymbol{\theta}) = \mathbb{E}_{\mathbf{s}_H \sim \mathbb{P}(\mathbf{s}_H)} \left[\sum_{j=0}^{H} r_j \nabla_{\boldsymbol{\theta}} \psi_j^{\boldsymbol{\theta}}(\mathbf{s}_j)\right] = \mathbb{E}_{\mathbf{s}_H \sim \mathbb{P}(\mathbf{s}_H)} \left[\sum_{j=0}^{H} r_j \psi_j^{\boldsymbol{\theta}}(\mathbf{s}_j) \nabla_{\boldsymbol{\theta}} \log \psi_j^{\boldsymbol{\theta}}(\mathbf{s}_j)\right] \tag{31}$$

$\square$

A.5 Proof of Proposition 4.1

Proof. First, for a given $\boldsymbol{\theta}^{(k)}$, substituting for $q_{ij}^{(k)}$ from Theorem 3.1 and $\tilde{w}_i^{(k)}$ from the W-step yields:

$$\begin{aligned}
\tilde{w}_i^{(k)} q_{ij}^{(k)} &= \left[\frac{\sum_{n=0}^{H} \psi_n^{\boldsymbol{\theta}^{(k)}}(\mathbf{s}_{in}) r_{in}}{\sum_{m=1}^{N} \sum_{n=0}^{H} \psi_n^{\boldsymbol{\theta}^{(k)}}(\mathbf{s}_{mn}) r_{mn}}\right] \left[\frac{\psi_j^{\boldsymbol{\theta}^{(k)}}(\mathbf{s}_{ij}) \tilde{r}_{ij}}{\sum_{n=0}^{H} \psi_n^{\boldsymbol{\theta}^{(k)}}(\mathbf{s}_{in}) \tilde{r}_{in}}\right] && (32)\\
&= \left(\sum_{n=0}^{H} r_{in}\right) \psi_j^{\boldsymbol{\theta}^{(k)}}(\mathbf{s}_{ij}) \tilde{r}_{ij} \left[\frac{1}{\sum_{m=1}^{N} \sum_{n=0}^{H} \psi_n^{\boldsymbol{\theta}^{(k)}}(\mathbf{s}_{mn}) r_{mn}}\right] && (33)\\
&= \underbrace{\left[\psi_j^{\boldsymbol{\theta}^{(k)}}(\mathbf{s}_{ij}) r_{ij}\right]}_{v_{ij}^{(k)}} \underbrace{\left[\frac{1}{\sum_{m=1}^{N} \sum_{n=0}^{H} \psi_n^{\boldsymbol{\theta}^{(k)}}(\mathbf{s}_{mn}) r_{mn}}\right]}_{z^{(k)}} && (34)
\end{aligned}$$

Therefore, we may write the M-step objective as :

$$\begin{aligned}
J_M^{(k)}(\boldsymbol{\theta}) &= z^{(k)} \sum_{i=1}^{N} \sum_{j=0}^{H} v_{ij}^{(k)} \log \psi_j^{\boldsymbol{\theta}}(\mathbf{s}_{ij}) && (35)\\
&\propto \frac{1}{N} \sum_{i=1}^{N} \sum_{j=0}^{H} v_{ij}^{(k)} \log \psi_j^{\boldsymbol{\theta}}(\mathbf{s}_{ij}) = \bar{J}_M^{(k)}(\boldsymbol{\theta}) && (36)
\end{aligned}$$

Dropping the iteration index (k), by defining constants $v_{ij} := \psi_j^{\boldsymbol{\theta}}(\mathbf{s}_{ij}) r_{ij}$ calculated with the most recent value of $\boldsymbol{\theta}$, the M-step objective is equivalent to the following objective:

$$\bar{J}_M(\boldsymbol{\theta}) = \frac{1}{N} \sum_{i=1}^{N} \sum_{j=0}^{H} v_{ij} \log \psi_j^{\boldsymbol{\theta}}(\mathbf{s}_{ij}) \tag{37}$$

where $v_{ij} := \psi_j^{\boldsymbol{\theta}}(\mathbf{s}_{ij}) r_{ij}$ is calculated with the most recent value of $\boldsymbol{\theta}$ and held constant. Taking the gradient w.r.t $\boldsymbol{\theta}$ we have

$$\nabla_{\boldsymbol{\theta}} \bar{J}_M(\boldsymbol{\theta}) = \frac{1}{N} \sum_{i=1}^{N} \sum_{j=0}^{H} v_{ij} \nabla_{\boldsymbol{\theta}} \log \psi_j^{\boldsymbol{\theta}}(\mathbf{s}_{ij}) \tag{38}$$

Now, substituting for v_{ij}, we have:

$$\nabla_{\boldsymbol{\theta}} \bar{J}_M(\boldsymbol{\theta}) = \frac{1}{N} \sum_{i=1}^{N} \sum_{j=0}^{H} r_{ij} \psi_j^{\boldsymbol{\theta}}(\mathbf{s}_{ij}) \nabla_{\boldsymbol{\theta}} \log \psi_j^{\boldsymbol{\theta}}(\mathbf{s}_{ij}) \tag{39}$$

This can be expressed as an expectation over sample trajectories as:

$$\nabla_{\boldsymbol{\theta}} \bar{J}_M(\boldsymbol{\theta}) = \mathbb{E}_{\mathbf{s}_H \sim \mathbb{P}(\mathbf{s}_H)} \left[\sum_{j=0}^{H} r_j \psi_j^{\boldsymbol{\theta}}(\mathbf{s}_j) \nabla_{\boldsymbol{\theta}} \log \psi_j^{\boldsymbol{\theta}}(\mathbf{s}_j)\right] = \nabla_{\boldsymbol{\theta}} J_{OS}(\boldsymbol{\theta})$$

Thus the gradient of the transformed M-step objective is identical to the optimal stopping policy gradient (OSPG). Therefore, if we perform the E-step, W-step and a single gradient update, the sequence of policy parameters $\boldsymbol{\theta}_n$ will exactly correspond to the updated OSPG policy parameters. We may therefore appeal to literature on incremental partial M-step E-M algorithms [27] and gradient descent [21] to conclude that for small enough step-size, that increases $J_M(\boldsymbol{\theta})$, the policy updates converge to a local maximum of both $J_{WML}(\tilde{\mathbf{w}}, \boldsymbol{\theta})$ and $J_{OS}(\boldsymbol{\theta})$.

□

A.6 Proof of Corollary 4.1.1

Proof. From the proof of Proposition 4.1, the M-step objective is equivalent to the following objective:

$$\bar{J}_M(\boldsymbol{\theta}) = \frac{1}{N}\sum_{i=1}^{N}\sum_{j=0}^{H} v_{ij}\log\psi_j^{\boldsymbol{\theta}}(\mathbf{s}_{ij}) \tag{40}$$

Now, substituting for $\psi_j^{\boldsymbol{\theta}}(\mathbf{s}_{ij})$ in terms of the stopping policy using the trajectory reparameterization lemma (Lemma 3.0.1):

$$\begin{aligned}
J_M(\boldsymbol{\theta}) \propto \bar{J}_M(\boldsymbol{\theta}) \quad &= \quad \frac{1}{N}\sum_{i=1}^{N} v_{i0}\log\phi_0^{\boldsymbol{\theta}}(\mathbf{s}_{i0}) + \sum_{j=1}^{H-1} v_{ij}\left[\log\phi_j^{\boldsymbol{\theta}}(\mathbf{s}_{ij}) + \sum_{n=0}^{j-1}\log(1-\phi_n^{\boldsymbol{\theta}}(\mathbf{s}_{in}))\right] \\
&\quad\quad + v_{iH}\sum_{n=0}^{H-1}\log(1-\phi_n^{\boldsymbol{\theta}}(\mathbf{s}_{in})) \\
&= \quad \frac{1}{N}\sum_{i=1}^{N}\left[\sum_{j=0}^{H-1} v_{ij}\log\phi_j^{\boldsymbol{\theta}}(\mathbf{s}_{ij})\right] \\
&\quad\quad + v_{i1}\log(1-\phi_0^{\boldsymbol{\theta}}(\mathbf{s}_{i0})) \\
&\quad\quad + v_{i2}\left[\log(1-\phi_0^{\boldsymbol{\theta}}(\mathbf{s}_{i0})) + \log(1-\phi_1^{\boldsymbol{\theta}}(\mathbf{s}_{i1}))\right] \\
&\quad\quad + v_{i3}\left[\log(1-\phi_0^{\boldsymbol{\theta}}(\mathbf{s}_{i0})) + \log(1-\phi_1^{\boldsymbol{\theta}}(\mathbf{s}_{i1})) + \log(1-\phi_2^{\boldsymbol{\theta}}(\mathbf{s}_{i2}))\right]\cdots \\
&\quad\quad + v_{iH}\sum_{n=0}^{H-1}\log(1-\phi_n^{\boldsymbol{\theta}}(\mathbf{s}_{in})) \\
&= \quad \frac{1}{N}\sum_{i=1}^{N}\left[\sum_{j=0}^{H-1} v_{ij}\log\phi_j^{\boldsymbol{\theta}}(\mathbf{s}_{ij})\right] + \left[\sum_{j=0}^{H-1}\log(1-\phi_j^{\boldsymbol{\theta}}(\mathbf{s}_{ij}))\left\{\sum_{n=j+1}^{H} v_{in}\right\}\right] \\
&= \quad \frac{1}{N}\sum_{i=1}^{N}\sum_{j=0}^{H-1} v_{ij}\log\phi_j^{\boldsymbol{\theta}}(\mathbf{s}_{ij}) + \log(1-\phi_j^{\boldsymbol{\theta}}(\mathbf{s}_{ij}))\left[\sum_{n=j+1}^{H} v_{in}\right]
\end{aligned}$$

Since by Proposition 4.1, we have $\nabla_{\boldsymbol{\theta}} J_{OS}(\boldsymbol{\theta}) = \nabla_{\boldsymbol{\theta}} \bar{J}_M(\boldsymbol{\theta})$. Setting $k_{ij} := \left[\sum_{n=j+1}^{H} v_{in}\right]$ in the above expression for $\bar{J}_M(\boldsymbol{\theta})$ and taking the gradient, we have:

$$\nabla_{\boldsymbol{\theta}} J_{OS}(\boldsymbol{\theta}) = \frac{1}{N}\sum_{i=1}^{N}\sum_{j=0}^{H}\left[\frac{v_{ij}(1-\phi_j^{\boldsymbol{\theta}}(\mathbf{s}_{ij})) - k_{ij}\phi_j^{\boldsymbol{\theta}}(\mathbf{s}_{ij})}{\phi_j^{\boldsymbol{\theta}}(\mathbf{s}_{ij})(1-\phi_j^{\boldsymbol{\theta}}(\mathbf{s}_{ij}))}\right]\nabla_{\boldsymbol{\theta}}\phi_j^{\boldsymbol{\theta}}(\mathbf{s}_{ij}) \tag{41}$$

This completes the proof of Corollary 4.1.1

□

B Dealing with costs instead of rewards

Although one could have used the negative of cost as a reward, this is inconsistent with our Bayesian net model since we require positive rewards to define the reward augmented trajectory model. The following result addresses this issue by transforming a problem with costs to one with a suitable reward specification.

Proposition B.1. Given costs $c_{ij} \geq 0$ the following two problems are equivalent:

$$\arg\min_{\boldsymbol{\theta}} \frac{1}{N}\sum_{i=1}^{N}\sum_{j=0}^{H} c_{ij}\psi_j^{\boldsymbol{\theta}}(\mathbf{s}_{ij}) \equiv \arg\max_{\boldsymbol{\theta}} \frac{1}{N}\sum_{i=1}^{N}\sum_{j=0}^{H} \underbrace{\left(\sum_{n=0}^{H} c_{in}\right)\left(1-\frac{c_{ij}}{\sum_{n=0}^{H} c_{in}}\right)}_{r'_{ij}}\psi_j^{\boldsymbol{\theta}}(\mathbf{s}_{ij})$$

Proof. Starting with the cost minimization problem, we may write:

$$\arg\min_{\boldsymbol{\theta}} \frac{1}{N}\sum_{i=1}^{N}\sum_{j=0}^{H} c_{ij}\psi_j^{\boldsymbol{\theta}}(\mathbf{s}_{ij}) = \arg\min_{\boldsymbol{\theta}} \frac{1}{N}\sum_{i=1}^{N} c_i \sum_{j=0}^{H} \tilde{c}_{ij}\psi_j^{\boldsymbol{\theta}}(\mathbf{s}_{ij}) \tag{42}$$

where $c_i := \sum_{j=0}^{H} c_{ij}$ and $\tilde{c}_{ij} = \frac{c_{ij}}{c_i}$. Since $\sum_{j=0}^{H} \psi_j^{\boldsymbol{\theta}}(\mathbf{s}_{ij}) = 1$, we may subtract the constant $\frac{1}{N}\sum_{i=1}^{N} c_i \sum_{j=0}^{H} \psi_j^{\boldsymbol{\theta}}(\mathbf{s}_{ij})$ from the objective yielding:

$$\begin{aligned}
\arg\min_{\boldsymbol{\theta}} \frac{1}{N}\sum_{i=1}^{N}\sum_{j=0}^{H} c_{ij}\psi_j^{\boldsymbol{\theta}}(\mathbf{s}_{ij}) &= \arg\min_{\boldsymbol{\theta}} \frac{1}{N}\sum_{i=1}^{N} c_i \sum_{j=0}^{H} \left(\tilde{c}_{ij} - 1\right)\psi_j^{\boldsymbol{\theta}}(\mathbf{s}_{ij}) \\
&= \arg\max_{\boldsymbol{\theta}} \frac{1}{N}\sum_{i=1}^{N} c_i \sum_{j=0}^{H} \left(1 - \tilde{c}_{ij}\right)\psi_j^{\boldsymbol{\theta}}(\mathbf{s}_{ij}) \\
&= \arg\max_{\boldsymbol{\theta}} \frac{1}{N}\sum_{i=1}^{N}\sum_{j=0}^{H} \underbrace{\left(\sum_{n=0}^{H} c_{in}\right)\left(1 - \frac{c_{ij}}{\sum_{n=0}^{H} c_{in}}\right)}_{r'_{ij}} \psi_j^{\boldsymbol{\theta}}(\mathbf{s}_{ij})
\end{aligned}$$

where we have expanded c_i and $\tilde{c}_{ij}$. This completes the proof. $\square$

C Neural Fitted Q-iteration approaches: DNN-FQI and RNN-FQI

Fitted Q-iteration methods [39, 40] use the Wald-Bellman equation (WBE) as follows: First, given parameterized function approximations $K_j^{\boldsymbol{\theta}}(S_j)$, a single Wald-Bellman step is bootstrapped, for a batch of trajectories yielding $\hat{V}_j(s_{ij}) = \max\{r_{ij}, K_j^{\boldsymbol{\theta}}(s_{ij})\}, \forall j < H, 1 \leq i \leq N$. Next the parameters of the continuation function are fit: $\boldsymbol{\theta}^* = \arg\min \sum_i \sum_j (K_j^{\boldsymbol{\theta}}(s_{ij}) - \hat{V}_{j+1}(s_{ij+1}))^2$, and the process is iterated. To provide competent DNN/RNN baselines for fitted Q-iteration (FQI) methods that are missing in the literature, we introduce a temporal FQI loss (Algorithm 2).

Algorithm 2 Pseudo-code for mini-batch computation of the FQI loss

```
Input: $\mathbf{R} := [r_{ij}]$, $\mathbf{K} := \left[K_j^{\boldsymbol{\theta}}(\mathbf{s}_{ij})\right]$ {$1 \leq i \leq N_b, 0 \leq j \leq H$, $N_b$ is batch size}
$\mathbf{V}_{:,0:H-1}$ = stop-gradient(max{$\mathbf{R}_{:,0:H-1}, \mathbf{K}_{:,0:H-1}$}) {bootstrap WBE to get value targets}
$\mathbf{V}_{:,H} = \mathbf{R}_{:,H}$ {final target is reward for last step}
J = MSE($\mathbf{V}_{:,1:H}, \mathbf{K}_{0,0:H-1}$) {next step value is target for current continuation function}
```

D Further details of the experiments

All experiments were performed on a shared server configured with 2 × Intel Xeon Silver 12-core, 2.10 GHz CPUs with 256GB RAM and equipped with 6 NVIDIA 2080Ti GPUs. However, experiments were run on a single GPU at a time and no computation was distributed across GPUs.

D.1 Model hyper-parameter settings

Table 2 shows general hyper-parameter settings used for all experiments. We apply Batch normalization to the input and outputs of layer activation functions at all hidden layers. Due to the dense correlation structure between assets at each time step of the American option pricing experiment, we choose the hidden units to be greater than the input dimension d.

As with RL policy gradients, we may subtract a baseline value[29] to reduce variance. The OSPG algorithm uses baseline b that does not depend on time-index j, sufficient to guarantee an unbiased OSPG estimator (see Appendix E). In our experiments, we use:

$$b = \frac{1}{NH}\sum_{i=1}^{N}\sum_{j=0}^{H} r_{ij} \tag{43}$$

Table 2: model hyper-parameter settings

method	hyper-parameter	tuned range	value
DOS, DNN-FQI, DNN-OSPG	num hidden layers	n/a	2
RRLSM, RNN-FQI, RNN-OSPG	num hidden layers	n/a	1
All models	hidden layer units	n/a	20
All models (Am. Option pricing)	hidden layer units	n/a	20 + d
All models	batch size	n/a	64
All models (Am. Option pricing)	batch size	n/a	128
All models	learning rate	$\{0.01, 0.001, 0.0001\}$	0.001
All models	epochs	early stopping	100
All models	batches/epoch	n/a	200
All models	optimizer	n/a	Adam
RRLSM	Kernel noise std	n/a	0.0001
RRLSM	Recurrent noise std	n/a	0.3

D.2 Pricing Bermudan options

Table 3 shows model sizes (in trainable parameters), training, and inference times (per time-step). Model sizes grow with input dimension except for RRLSM, which uses an RNN with random, non-trainable weights to extract features from the input. The parameter size of DOS is about an order of magnitude higher than DNN-FQI and DNN-OSPG since parameters in backward induction methods like DOS grow linearly with the number of time steps (parameters are not shareable across time steps). Training and inference times are also high since individual models must be fit and inferred at each time step.

Table 3: model sizes and compute times for Bermudan max-call experiment

method	assets	model-size (params)	mean training time (seconds)	mean time/prediction (μ-seconds)
DOS	20	9,225	271	3
DNN-FQI	20	1,047	23	5
DNN-OSPG	20	1,047	29	7
RRLSM	20	198	2	14
RNN-FQI	20	2,807	32	8
RNN-OSPG	20	2,807	37	8
DOS	50	15,165	289	3
DNN-FQI	50	1,707	28	6
DNN-OSPG	50	1,707	29	7
RRLSM	50	198	2	14
RNN-FQI	50	4,667	32	8
RNN-OSPG	50	4,667	32	8
DOS	100	25,065	326	3
DNN-FQI	100	2,807	30	6
DNN-OSPG	100	2,807	28	6
RRLSM	100	198	2	14
RNN-FQI	100	7,767	33	8
RNN-OSPG	100	7,767	30	8
DOS	200	44,865	317	3
DNN-FQI	200	5,007	30	6
DNN-OSPG	200	5,007	27	6
RRLSM	200	198	2	15
RNN-FQI	200	13,967	35	8
RNN-OSPG	200	13,967	30	8

Table 4: American geometric-call option pricing: Results

			average return (error %)				
d	s_0	p^*	LS [23]	PDE-DGM [37]	PDE-BSDE [7]	DNN-FQI [34]	DNN-OSPG
7	90	5.9021	5.8440 (0.98%)	NA	**5.8822** (0.34%)	5.7977 (1.77%)	5.8704 (0.54%)
7	100	10.2591	10.1736 (0.83%)	NA	10.2286 (0.30%)	10.1022 (1.53%)	**10.2518** (0.07%)
7	110	15.9878	15.8991 (0.55%)	NA	**15.9738** (0.09%)	15.0487 (5.87%)	15.9699 (0.11%)
13	90	5.7684	5.5962 (3.00%)	NA	**5.7719** (0.06%)	5.7411 (0.47%)	5.7436 (0.43%)
13	100	10.0984	9.9336 (1.60%)	NA	**10.1148** (0.16%)	9.9673 (1.30%)	10.0691 (0.29%)
13	110	15.8200	15.6070 (1.40%)	NA	**15.8259** (0.04%)	14.7759 (6.60%)	15.8107 (0.06%)
20	90	5.7137	5.2023 (9.00%)	NA	**5.7105** (0.06%)	5.6607 (0.93%)	5.6983 (0.27%)
20	100	10.0326	9.5964 (4.40%)	**10.0296** (0.03%)	10.0180 (0.15%)	9.6372 (3.94%)	10.0100 (0.23%)
20	110	15.7513	15.2622 (3.10%)	NA	15.7425 (0.06%)	14.9345 (5.19%)	**15.7553** (0.03%)
100	90	5.6322	OOM	NA	5.6154 (0.30%)	5.3858 (4.38%)	**5.6211** (0.20%)
100	100	9.9345	OOM	**9.9236** (0.11%)	9.9187 (0.16%)	9.3954 (5.43%)	9.8954 (0.40%)
100	110	15.6491	OOM	NA	15.6219 (0.17%)	14.6335 (6.49%)	**15.6301** (0.12%)
200	100	9.9222	OOM	9.9004 (0.22%)	**9.9088** (0.14%)	9.3772 (5.49%)	9.8991 (0.23%)

D.3 Pricing American options

The scope of the paper is solving *discrete-time, finite-horizon, model-free* optimal stopping problems. Bermudan options that have discrete exercise opportunities are one example application. American options, which are more popular, are based on continuous-time asset price evolution and have a continuum of possible exercise times. One way to convert this to our discrete-time setting is to solve related Bermudan options. These options limit exercise opportunities to a fine discrete time grid.

State-of-the-art algorithms for pricing American options are based on Partial differential equation (PDE) methods. These methods are model-based since they start with a PDE (such as the multi-dimensional Black-Scholes Model) defining process evolution. For example, PDE methods often assume Markovian Black-Scholes Dynamics, and the PDEs to be solved require the Black-Scholes model parameters, such as covariance of the Brownian motion, volatility, risk-free interest rate, and dividend yield. In contrast, model-free methods, such as FQI and OSPG algorithms, do not use prior information on the evolution dynamics of the underlying stochastic process.

Nevertheless, we compare our model-free OSPG method against state-of-the-art PDE methods such as the Deep Galerkin Method (DGM) [37] and Backward Stochastic Differential Equations method (BSDE)[7] by suitable discretization of the original continuous time-problem. Note that the PDE methods also require discretization of the original PDE (ex, using the Euler-Maruyama scheme) or random sampling (as used in DGM) but do not end up directly solving a Bermudan option.

We consider multi-dimensional continuous-time American geometric-average call options with Black-Scholes dynamics considered in [37] and [7]. The payoff of these options depends on the price of d underlying assets with multi-dimensional Black-Scholes dynamics and the strike price K. The dynamics are Markovian, with payoff (reward) given by:

$$S_t^m = s_0^m \exp([r - \delta_m - \sigma_m^2/2]t + \sigma W_t^m, \quad R_t = \left(\left[\prod_{m=1}^{d} S_t^m \right]^{\frac{1}{d}} - K \right)^+ \tag{44}$$

for $m = 1, 2, \cdots d$. $r \in \mathbb{R}$ is the risk-free rate, $s_0^m \in (0, \infty)$ represents the initial price, $\delta_m \in [0, \infty)$ is the dividend yield, $\sigma_m \in (0, \infty)$ the volatility and W is a d-dimensional Brownian motion, with instantaneous correlation between its d components given by $\mathbb{E}[W_t^i W_t^j] = \rho_{ij} t$. The reward for exercise at time t is given by R_t. We discretize t into $H + 1 = 100$ possible exercise opportunities, using times $t_j = jT/H$ for $j = 0, 1, \cdots H$. T is the option expiration duration in years. This yields the stopping problem: $\sup_{0 \leq \tau \leq H} \mathbb{E}[R_\tau]$.

The specific option we consider is characterized by the following parameters: $K = 100$, $r = 0.0$, $\sigma_m = 0.25$, $\rho_{ij} = 0.75$ $\forall i \neq j$, $\delta_m = 0.02$, $T = 2$. The exact price of this option, p^*, can be determined semi-analytically for comparison [7]. We generate 10,000 batches (with a batch size of 128) for training and compute option prices on a 3,000-batch test sample. We compare vs. published results from state-of-the-art model-based PDE baselines, including the Deep Galerkin Method (PDE-DGM) [37], the Backward Stochastic Differential Equation (PDE-BSDE) method [7]. We also include published results [7] from the industry standard Longstaff-Schwartz (LS) option

Table 5: Stopping a fractional Brownian motion: Results

	average return (standard deviation)						
h	DOS	DNN-FQI	DNN-OSPG	DOS-ES	RRLSM	RNN-FQI	RNN-OSPG
0.05	0.70 (0.01)	0.87 (0.06)	1.14 (0.00)	1.18 (0.01)	1.16 (0.00)	1.24 (0.04)	**1.28** (0.01)
0.10	0.54 (0.01)	0.68 (0.04)	0.92 (0.01)	0.93 (0.01)	0.94 (0.00)	0.99 (0.03)	**1.03** (0.02)
0.20	0.34 (0.01)	0.42 (0.08)	0.58 (0.00)	0.55 (0.01)	0.57 (0.00)	0.59 (0.04)	**0.64** (0.01)
0.30	0.19 (0.00)	0.14 (0.07)	0.29 (0.10)	0.28 (0.01)	0.29 (0.00)	0.27 (0.08)	**0.36** (0.01)
0.40	0.10 (0.01)	0.02 (0.03)	0.13 (0.00)	0.09 (0.01)	0.10 (0.00)	0.04 (0.04)	**0.15** (0.00)
0.50	**0.00** (0.00)	**0.00** (0.00)	**0.00** (0.00)	**0.00** (0.00)	**0.00** (0.00)	**0.00** (0.00)	**0.00** (0.00)
0.60	0.08 (0.01)	0.03 (0.03)	**0.10** (0.00)	0.08 (0.01)	0.06 (0.00)	0.03 (0.03)	**0.10** (0.03)
0.70	0.16 (0.01)	0.08 (0.08)	0.17 (0.02)	0.18 (0.01)	0.19 (0.00)	0.10 (0.07)	**0.20** (0.00)
0.80	0.23 (0.01)	0.03 (0.05)	0.25 (0.00)	0.26 (0.01)	**0.27** (0.00)	0.18 (0.09)	**0.27** (0.01)
0.90	0.31 (0.01)	0.14 (0.12)	0.28 (0.10)	0.32 (0.01)	**0.33** (0.00)	0.21 (0.12)	**0.33** (0.00)
0.95	0.35 (0.01)	0.01 (0.05)	0.34 (0.01)	**0.36** (0.00)	**0.36** (0.00)	0.25 (0.08)	**0.36** (0.00)

pricing algorithm [23], which uses linear approximation with multi-dimensional basis functions, suitable for low dimensional settings.

Table 4 summarizes the experiment's results. Published results [7] for LS, DGM, and BSDE are included. NA denotes results not reported in the literature, while OOM indicates *out of memory*. Our model-free DNN-OSPG algorithm compares favorably with state-of-the-art model-based PDE-based option pricing methods that have prior knowledge of process evolution. Specifically, unlike LS, whose accuracy degrades with the number of correlated assets, and DNN-FQI, whose accuracy degrades up to 6.5%, DNN-OSPG retains excellent performance (< 0.55% error) closely approaching the model-based PDE methods (< 0.35% error). Of course, a key advantage of DNN-OSPG is that it can be used to price options for which no known PDE is available to describe process evolution.

D.4 Stopping a fractional Brownian motion

Table 5 provides numerical values corresponding to Figure 2. RNN-OSPG dominates competing methods across all values the Hurst parameter in this non-Markovian setting.

Table 6: Stopping a fractional Brownian motion: Model sizes and compute times

method	model-size (params)	mean training time (seconds)	mean time/prediction (μ-seconds)
DOS	62,900	2250	26.9
DNN-FQI	651	21	0.5
DNN-OSPG	651	29	0.5
DOS-ES	276,300	2228	25.9
RRLSM	2,200	9	5.0
RNN-FQI	1,691	14	1.1
RNN-OSPG	1,691	79	1.1

D.5 Early stopping a sequential multi-class classifier

We start with the 34 datasets, and corresponding values of α used in the [1] and remove binary classification datasets. This leaves 17 datasets. However, many of these have very few series or have a large number of classes relative to series size, which might make them unsuitable for training RNN classifiers. Nevertheless, we report results on all 17 datasets. Table 7 reports the experiment's mean and standard deviation (over ten random splits) results over 17 UCR time-series datasets. Figure 2 provides a graphical visualization of the same results. N denotes the number of trajectories, H denotes the length of the series, and K denotes the number of classes. α trades-off classification accuracy and earliness. RNN-OSPG achieves the best performance on 15 of the 17 datasets.

Table 7: Early stopping a sequential multi-class classifier: Results

		average cost (standard deviation)		
Dataset	N/H/K/α	RRLSM	RNN-FQI	RNN-OSPG
CBF	930/128/3/0.8	0.192 (0.017)	0.308 (0.138)	**0.186** (0.011)
ChlorineConcentration	4,307/166/3/0.4	0.468 (0.011)	0.523 (0.073)	**0.455** (0.003)
Crop	24,000/46/24/0.06	0.428 (0.005)	0.407 (0.010)	**0.389** (0.008)
ECG5000	5000/140/5/0.5	0.110 (0.005)	0.217 (0.059)	**0.099** (0.006)
ElectricDevices	16,637/96/7/0.1	0.257 (0.015)	0.260 (0.012)	**0.228** (0.009)
FaceAll	2,250/131/14/0.01	0.108 (0.025)	0.100 (0.010)	**0.092** (0.017)
FaceUCR	2,250/131/14/0.5	0.358 (0.035)	0.465 (0.049)	**0.328** (0.028)
FiftyWords	905/270/50/0.5	0.667 (0.027)	0.812 (0.041)	**0.634** (0.029)
InsectWingbeatSound	2,200/256/11/1	0.666 (0.064)	0.897 (0.110)	**0.646** (0.080)
MedicalImages	1,141/99/10/0.07	0.296 (0.033)	0.309 (0.031)	**0.263** (0.038)
MelbournePedestrian	3,633/24/10/0.8	0.591 (0.065)	0.597 (0.068)	**0.487** (0.044)
MixedShapesRegularTrain	2,925/1024/5/0.1	0.194 (0.024)	0.194 (0.008)	**0.191** (0.037)
NonInvasiveFetalECGThorax2	3,765/750/42/0.04	0.418 (0.094)	0.343 (0.008)	**0.295** (0.064)
StarLightCurves	9,236/1024/3/0.3	**0.120** (0.043)	0.315 (0.067)	0.165 (0.035)
Symbols	1,020/398/6/0.2	**0.103** (0.013)	0.208 (0.030)	0.239 (0.070)
UWaveGestureLibraryX	4,478/315/8/0.5	0.523 (0.023)	0.658 (0.026)	**0.498** (0.040)
WordSynonyms	905/270/25/0.6	0.760 (0.024)	0.913 (0.043)	**0.727** (0.041)

E Baseline subtraction for variance reduction

As with RL policy gradients, we may subtract a baseline value[29] to reduce variance. However, unlike the general RL case, due to the stopping-time-based formulation of the OSPG, the OSPG baseline should not be time-varying.

Proposition E.1 (baseline subtraction for OSPG)**.** The optimal stopping policy gradient (OSPG) of Theorem 4.1 is invariant to the subtraction of a constant (w.r.t. trajectory) baseline from every reward in the trajectory. Thus:

$$\begin{aligned}\nabla_{\boldsymbol\theta} J_{OS}(\boldsymbol\theta) &= \mathbb{E}_{\mathbf{s}_H\sim\mathbb{P}(\mathbf{s}_H)}\left[\sum_{j=0}^{H} r_j \psi_j^{\boldsymbol\theta}(\mathbf{s}_j)\nabla_{\boldsymbol\theta}\log\psi_j^{\boldsymbol\theta}(\mathbf{s}_j)\right]\\ &= \mathbb{E}_{\mathbf{s}_H\sim\mathbb{P}(\mathbf{s}_H)}\left[\sum_{j=0}^{H}(r_j-b)\psi_j^{\boldsymbol\theta}(\mathbf{s}_j)\nabla_{\boldsymbol\theta}\log\psi_j^{\boldsymbol\theta}(\mathbf{s}_j)\right]\end{aligned}$$

Proof. It suffices to show $\mathbb{E}_{\mathbf{s}_H\sim\mathbb{P}(\mathbf{s}_H)}\left[\sum_{j=0}^{H} b\psi_j^{\boldsymbol\theta}(\mathbf{s}_j)\nabla_{\boldsymbol\theta}\log\psi_j^{\boldsymbol\theta}(\mathbf{s}_j)\right]=0$. We proceed as follows:

$$\begin{aligned}E_{\mathbf{s}_H\sim\mathbb{P}(\mathbf{s}_H)}\left[\sum_{j=0}^{H} b\psi_j^{\boldsymbol\theta}(\mathbf{s}_j)\nabla_{\boldsymbol\theta}\log\psi_j^{\boldsymbol\theta}(\mathbf{s}_j)\right] &= E_{\mathbf{s}_H\sim\mathbb{P}(\mathbf{s}_H)}\left[\sum_{j=0}^{H} b\psi_j^{\boldsymbol\theta}(\mathbf{s}_j)\frac{\nabla_{\boldsymbol\theta}\psi_j^{\boldsymbol\theta}(\mathbf{s}_j)}{\psi_j^{\boldsymbol\theta}(\mathbf{s}_j)}\right]\\ &= E_{\mathbf{s}_H\sim\mathbb{P}(\mathbf{s}_H)}\left[\nabla_{\boldsymbol\theta}\sum_{j=0}^{H} b\psi_j^{\boldsymbol\theta}(\mathbf{s}_j)\right]\\ &= E_{\mathbf{s}_H\sim\mathbb{P}(\mathbf{s}_H)}\left[\nabla_{\boldsymbol\theta} b\underbrace{\sum_{j=0}^{H}\psi_j^{\boldsymbol\theta}(\mathbf{s}_j)}_{=1}\right]\\ &= 0 \qquad (45)\end{aligned}$$

□