ディープ・オプティマル・ストッピングに関するディスカッション

最終プロジェクトSTA 4246

ヴァネッサ・ピザンテ

トロント大学学生ID1004
874022
v.pizante@mail.utoronto.ca
2019年4月12日開催

概要

本プロジェクトでは、主に[Becker et al., 2018]が提案するフィードフォワード多層ニューラルネットワークを採用し、最適停止問題を解くことに焦点を当てる。また、最適停止問題に対する関連するニューラルネットワークや機械学習アプローチについても議論します。このような近似手法の主要な動機が、従来の数値手法が高次元で遭遇する問題を改善することであることを強調します。我々は、多数の資産に書かれたバミューダンマックスコールオプションの価格設定と、問題の次元を上げることで分数ブラウン運動をマルコフ過程として再フレーミングすることによって、どのように最適に停止できるかを検証することによって、これを説明する。

1 はじめに

最適停止問題とは、あらかじめ定義された報酬の概念を最大化する（あるいは、損失を最小化する）ために、逐次観測される確率変数を用いて行動を起こす時期を選択する問題である。 本報告では、このような乱数列を用いた問題を扱う。

は $X = (X_n)_{n=0}^{N}$ と表し、フィルタリングされた確率空間 $(\Omega, F, (\mathrm{F})^N, P)$ 上の R^d -valued discrete time Markov process とする。

この文脈で、数学的にτは以下の場合、X*停止*時間であるという。

$\{\tau = n\} \in F, \forall n \in \{0, 1, ..., N\}$*とする。*我々が最大化しようとする報酬の概念は次式で与えられる。

$$V = \sup_{\tau \in T} \mathrm{E}g(\tau, X_\tau) \tag{1}$$

ここで、T はすべての X *停止*時刻の集合、$g : \{0, 1, ..., N\}$ *は計測可能な可積分関数である。*$\times \mathrm{R}^d \to \mathrm{R}$ は測定可能で可積分な関数である。

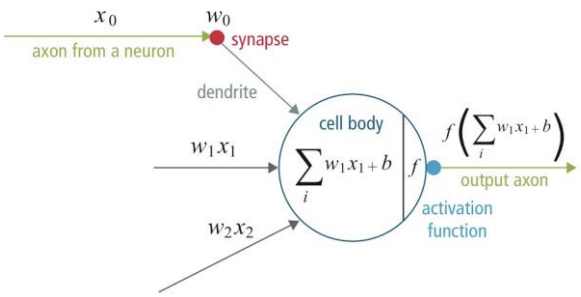
このような有限個の停止時間を持つ最適停止問題は理論的には正確に解くことができ、最適V*は*
スネルの包絡線によって与えられ、対応する最適停止時間は停止による報酬が継続による期待報酬（継続値）を初めて上回ったときとなります。数値的には、これらの問題はしばしば動的計画法に基づくアプローチで解かれる。しかし、ツリーベースの手法などのより伝統的な数値的アプローチは、マルコフ過程X *の*
次元が3より大きい場合、パフォーマンスが低下します[Becker et al., 2018]。これは、解くことができる最適な問題の種類を制限します。例えば、より高い次元で作業できることで、多くの可能性を持つ何百もの原資産に書かれたオプションの価格を決定することができます。
の運動時間である。さらに、任意のプロセス$X_{n=0}$ は、必要なものを含めることでマルコフに
$= (X_n)_{n=0}^{N}$ することができます。
の情報は、現在の状態でのプロセスの履歴に関するものである。しかし、これは明らかに問題の次元を増加させる。

計算機数理ファイナンスの新たな研究分野として、最適停止問題への機械学習アプローチの適用が目指されている。多くの機械学習アルゴリズムは、非常に高い次元に強く、幅広い問題に適用できる柔軟性を持っています。本報告では、[Becker et al., 2018]による最適停止問題へのフィードフォワード多層ニューラルネットワークの適用を中心に紹介する。彼らのアプローチを掘り下げる前に、まずニューラルネットワークの予備知識を確認する。

1.1 ニューラルネット

ニューラルネットワークは、人間の脳を大まかにモデル化した計算システムである。生物学的な詳細は省くが、基本的な考え方は、あるニューロンが他のニューロンや外部ソースから入力を受け、それを使って出力を生成するというものである。この出力は、入力として別のニューロンに渡されるか、システム全体の出力として扱われます。計算ニューラルネットワークでは、これらのニューロンはしばしばノードと呼ばれる。

一つのノードが入力を受け取り、それを出力に変換する方法を図1に示す。x_1 , x_2 は入力であり、このノードが第1層にある場合はネットワーク全体の入力、または前の層の出力である。各x_i は対応するw_i を持ち、これはその重みと呼ばれ、当該入力の相対的重要度を定量化する。そして、図 1 の「セル本体」に見られるように、これらの入力と重みは、バイアス項を追加して合計される。この和は、活性化関数fを介して変換された後、そのノードの出力となる。活性化関数の出力は、このノードの相対的な発火率として考えることができる。



出典はこちら[ヒジャジら、2015年］。

図1: ニューラルネットワークの1つのノード。

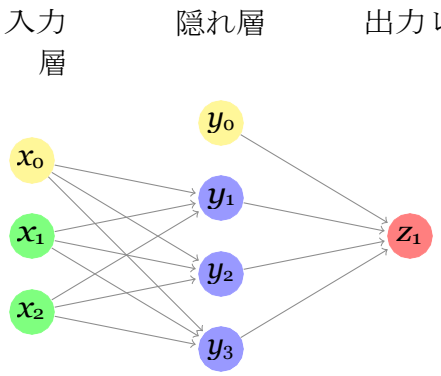


図2：フィードフォワードニューラルネットワークの例。

図2は、密な単層フィードフォワードニューラルネットワークアーキテクチャにおいて、ノードがどのようにリンクされるかを示している。密な」という用語は、ネットワークが完全に接続されていることを意味し、与えられたノードが前の層のすべてのノードから入力を取り、その出力を次の層のすべてのノードに送ることを意味し、単層は単にネットワークで特徴付けられる単一の隠れ層を意味する。

図2のニューラルネットワークが何を行っているかを数学的に理解するために、$x = (x_1, x_2)$ と $y = (y_1, y_2, y_3)$ のような合成関数 $f(x) = z_1$ として表現することができる。

$$f^{\theta} = a_1 \circ \varphi_1 \circ a_2 \circ \varphi_2$$

のところです。

- $a_1 : \mathrm{R}^2 \to \mathrm{R}^3$ および $a_2 : \mathrm{R}^3 \to \mathrm{R}^2$ は、$a_1(x) = W_1 x + x_0$ という形式の一次関数であり $a_2(x) = W_y y + y_0$ ここで、$W_1 \in \mathrm{R}^{3\times 2}, W_2 \in \mathrm{R}^{1\times 3}, x_0 \in \mathrm{R}^3, y_0 \in \mathrm{R}$ です。
- $\varphi_1 : \mathrm{R}^3 \to \mathrm{R}^3$, $\varphi_2 : \mathrm{R} \to \mathrm{R}$ は、隠れ層と出力層に対応する活性化関数である。ReLU活性化関数やロジスティックシグモイドが一般的である。
- f^{θ} のθ部分は、モデルの重みとバイアスの項をすべて含むモデルのパラメータ、すなわち

$\theta = \{W_1, x_0, W_2, y_0\}$ を指す。

モデルのアーキテクチャが確立されたら、次はモデルを訓練する必要がある。

モデルの学習は
モデルとは、入力とターゲットのペアである (x^m, z^m) または
からなる学習データを使用することである。$_M$、単に
の入力、$(x^m_{m=1}$ を学習し、最適なθを求める。この意味での最適とは、ある損失を最小にするこ$)^M$ とである。
このような関数の例として、対の学習データに対する平均二乗誤差損失関数があります。

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

θに関するCの最小化は、通常、ある種の勾配降下アルゴリズムによって行われる。最も基本的な例はバニラ勾配降下法であり、これは更新ステップを持つ。

$$\theta_{t+1} \leftarrow \theta^t - \eta \cdot \nabla_\theta C(\theta^t)$$

ここで、ηは学習率である。$_i$バックプロパゲーションと呼ばれるグラフィカルな差分化手法を用いて、各反復で ∇_θ $C(\theta^t$)を近似する。これは、ニューラルネットワークグラフを効率的に逆行して、連鎖則により各 $\partial C/\partial$ を計算する手法に過ぎない。学習が完了するのは、勾配降下アルゴリズムが完了したときであり、これは$_\theta C(\theta^t) < E$ を意味し、E はある種の
これは $C(\theta^t$) がある種の極値に達したことを意味するので、事前に選択された閾値は 0 に近い。ネットワークの学習が完了したら、次にテストデータと呼ばれる新しいデータセットでアルゴリズムを実行する。

さて、最適停止問題とニューラルネットワークの基礎の紹介が終わったので、[Becker et al., 2018]の手法に移る前に、最適停止問題に適用されている他の機械学習技術について簡単なサーベイを行う。

2 文献レビュー

先に述べたように、最適停止問題に機械学習アプローチを適用する主な動機は、以前の方法が直面していた次元の呪いを克服する能力である。これは、彼らの多層フィードフォワードニューラルネットワークアプローチの触媒として[Becker et al., 2018]によって強調されているが、この同じ問題に取り組むために関連する機械学習手法を適用する他のいくつかの論文がある。以下のセクションでは、これらの類似の動機付けを持つメカニズムの主要なアイデアを簡単に見ていきます。

2.1 最小二乗モンテカルロ法へのニューラルネットワークの応用

最適停止問題に対する初期のアプローチに機械学習技 術を適用した強力な例は、Longstaff-Schwartz アルゴリズムに ニューラルネットワークを適用した[Kohler et al.Longstaff-Schwartz アルゴリズム [Longstaff and Schwartz, 2001] は、最小二乗モンテカルロ法 (LSMC) を用いて、最適停止問題のベンチマークであるアメリカン・オプション
の価格を近似的に計算するもので、Longstaff-Schwartz アルゴリズム は、LSMC を用いて、アメリカン・オプションの価格を近似的に計 算する。

価格過程 (X_t)、権利行使価格 K、有効期限 T を持つ資産に対して書かれたアメリカン・コール・オプションの価格は、単純に (1) の V を計算することになります。ここで $g(t, X_t) = (X_t - K)^+$ と $T = \tau \mid 0 \leq t \leq T$ とします。

Longstaff and Schwartz, 2001] と [Kohler et al., 2010] の両者において、時間は $\{t_n\}_{n=0}^N$ で離散化される。

$t_n = nT/N$, 表記を簡略化するために $\{0, 1, \ldots, N\}$ と表記する。そして、M個の資産価格経路 $(x_n^m)_{n=0}^N$ の形のシミュレーションを行い、時間Nの後にオプション価格$V_n^m = g(n, x_n^m)$を計算する。

を各経路に沿わせる。両アルゴリズムとも、その後、$n = N - 1, \ldots, 0$ に対して、近似的には、各タイムステップで継続値 q_n を模倣し、$q_n(x_n^m) < g(n, x_n^m)$ であれば運動する、というものである。

設定に変換されます

$$V_n^m = \begin{cases} g(n, x_n^m) & \text{if } q_n(x_n^m) < g(n, x_n^m) \text{である。} \\ e^{-rtn} V_{n+1} & \text{でないと} \end{cases}$$

ここで、rは市場リスクフリーリターンレートである。

両者のアプローチの違いは、$q_n(x)$を計算する方法が重要である。Longstaff-Schwartz アルゴリズムは、最小二乗モンテカルロ法を用いて継続値を計算する。

これには、q_n を $\hat{q}_n(x) = \sum_{k=0}^{K} \beta_k \varphi_k(x)$ で近似することが必要である。 ここで、$\varphi_k(x)$ は何らかの活性化関数である。

は典型的な多項式（すなわち、$\varphi_k(x) = x^k$ ）であり，$\beta = (\beta_k)$ は最小二乗誤差関数を最小化する。

$$C = \sum_{m=1}^{M} (e^{-rtn} V_{n+1} - q_n(x_n^m)) \tag{2}$$

先に述べたように、この継続値の計算方法は、多数の資産を対象にしたオプションの場合、信頼性が低くなります。Kohler et al., 2010]が提案する代替案は、このような高次元の問題に対処するために、最小二乗損失関数を持つ単層のニューラルネットワークを構築することである。具体的には、彼らは次のようなニューラルネットワークを提案している。

$$q^{\theta_n} = a_2^{\theta_n} \circ \sigma \circ a_1^{\theta_n}$$

どこ

- $a_1 : \mathrm{R}^d \to \mathrm{R}^K$ は、$W_1 x + b_1$ の形の線形関数で、ここで $W_1 \in \mathrm{R}^{K \times d}$, $b_1 \in \mathrm{R}^d$
- $\sigma : \mathrm{R}^k \to [0, 1]^K$ はシグモイド活性化関数で、k個のエントリそれぞれについて、$\sigma_k(y_k) = \frac{1}{1+e^{-y_k}}$
- $a_2 : \mathrm{R}^K \to \mathrm{R}$ は、$W_2 x + b_2$ という形の一次関数で、$W_2 \in \mathrm{R}^K$, $b_2 \in \mathrm{R}$, $W_2 =$ となる。 $(w_{2,1}, ..., w_{2,K})$ は、$\sum_{k=0}^{K}$ を満たす。 $|w_{2,k}| \le C_n$

このアルゴリズムで重要なことは、学習用のペア、(x_n^m, V_{n+1}^m) に対して、バックワード前のステップの出力に依存する(2)の形の損失関数を採用することによって再帰的に後方、V^m.Kohler et al., 2010]の結果は、より複雑なオプションに適用した場合、以下のようになります。

(高次元ではLongstaff-Schwartzアルゴリズムより優れているが、低次元ではより単純なオプション(American put)でほぼ同じ結果である。これは、最適停止問題を解く古い方法が、より複雑な問題を解くために、ニューラルネットワークを使ってどのように修正されるかを示している。次に、別の機械学習技術を用いた最適停止問題へのアプローチについて見ていきます。Q-ラーニングである。

2.2 Q-Learningの考え方

最適停止問題に適用されている機械学習アルゴリズムの異なる分

野に、強化学習がある。一般に、これらのアルゴリズムはマルコフ決定過程の機械学習的アプローチである。強化学習アルゴリズムが従来のマルコフ決定過程のアプローチと大きく異なる点は、強化学習アプローチがモデルフリーであること、つまり、マルコフ過程の遷移確率に分布を課さないことである。ここで検討する具体的なアプローチは、[Yu and Bertsekas, 2007]の Q-Learning アルゴリズムである。この著者らは、このアプローチを適用する主な動機として、より大きな状態空間における動的計画法の性能の低さを挙げていることに注目すべきである。

Q-learningのアプローチを最適停止問題に適用すると、この問題を(X_n
)によって支配されるマルコフ決定過程と見なすことができ、ここで、最適停止問題の期待値を最小化することを目指す。
で停止するか継続するかを最適な順序で決定することで、合計α割引コスト（$\alpha \in (0,1)$）を算出する。
を各ステップごとに計算する。全体のコスト関数を最小化しようとするのではなく、各状態からの各判断のコストを個別に調べる方が計算効率が良い。この各状態での各判断のコスト関数をQ関数と呼ぶ。

したがって、状態X_n にいるとすると、停止してコストf (X_n) を発生させるか、継続してコストh (X_n , X_{n+1}) を発生させるかの2つの可能な決定がある（これらの関数の値は状態X_n
からわかっていると仮定している）。つまり、Q関数は状態X_n
が与えられたときの任意の行動に関連するコストに過ぎないので、$Q(X_n, \text{stop}) = f(X_n)$ と $Q(X_n, \text{cont.}) = h(X_n, X_{n+1})$ を意味します。従って、状態の最適なQ関数は
X_n は、Q^* (X_n) = mina∈{stop,cont.}と定義される。}$Q(X_n$, $a)$となる。このことから、最適な停止方針は
は、X_n が以下の集合に入ると同時に停止すると再定義される。

$$D = \{X_n \mid f(X_n) \le Q(X_n)\}。$$

このQ^* 関数を求めるために、Q-Learningを採用する。これは、対応するベルマン誤差を最小化するように、Q-関数を近づけるための単純な反復アルゴリズムである。具体的には、ある停止していない状態プロセス(x_n)をシミュレートした後、Q-Learningアルゴリズムの更新条件を近似的に求めます。
imate Q^* です。

$$Q^*(x_n) \leftarrow Q^*(x_n) + \gamma(g(x_n, x_{n+1}) + \alpha \min\{c(x_{n+1}), Q^*(x_{n+1})\} - Q^*$$

$(x_n))$ ここで$\gamma \in (0, 1)$ は学習率である。

アルゴリズムの計算効率をさらに高めるために，[Yu and Bertsekas, 2007] は，X_n の変換の線形射影を使用する Q^* の近似を考察している．

$$Q^*(x_n) \equiv \varphi(x_n)^t \beta_n$$

そして、対応する投影ベルマン誤差は、以下のようにQ-learningアルゴリズムに基づく最小二乗近似によって、β_n に関して最小化される。

$$\beta_{n+1} = \beta_n + \alpha(\beta\hat{}n+1 - \beta_n)$$

どこ

$$\beta\hat{}n = \arg\min_{\beta} \sum_{k=o}^{n} \left(\varphi(x_k)^t \beta - h(x_k, x_{k+1}) - \gamma \min\{f(x_{k+1}), \varphi(x_k)^t \beta_k\}\right)^2$$

これはQ-learningアルゴリズムの非常に小さな修正であることに注意してください。重要な違いは、お

そらく複雑なQ関数を最小化する代わりに、最小化する定数β_n
のベクトルを見つけるだけでよいということです。

Yu and Bertsekas, 2007]では数値的な例が提示されていないため、このアルゴリズムの性能を実際に評価することは困難であることは、注目に値します。この論文で見つけたもう一つの問題は，継続値関数 $h(X_n , X_{n+1})$をどのように得るのが最善かについて議論していないことである．しかし、我々の目的のためにこれを検討することは、ニューラルネットワークとは別の機械学習の一分野が最適停止問題を解くためにどのように使われるかという興味深い考察を提供することになる。

2.3 拡張機能より複雑なニューラルネットワーク

最後に、[Becker et al, 2018]の最近の拡張を見ることで、最適停止問題へのニューラルネットワークの適用に関する議論に戻ります。我々が調べる作品は[Hu, 2019]で、彼らは最適停止問題を画像分類問題として再キャストすることを検討しています。この再キャスティングは、各入力xにクラスラベルを割り当てるプロセスを参照するランク応答曲面を用いて行われます。この方法で問題を変換する主な動機は、[Becker et al., 2018]と[Kohler et al., 2010]で採用された単純なフィードフォワードネットワークのアーキテクチャを、より良いパフォーマンスをもたらすことができるコンボリューションニューラルネットワークで置き換えることができることである。

前述したように、この方法を適用するための最初のステップは、問題をサー
面ランキング問題である。これは、各入力x に 対 し て
最小曲面のインデックスを抽出し、それをクラスラベルとして扱うものである。したがって
、我々は入力空間X を クラスに分割していることになる。時間 n
における最適停止問題では、停止と継続の2つのラベルしか持たないので、最適な選択は次のようになる。
は、次のように表面ランキング問題として再構成される。

$$C(n, X_n) = \arg\max_{j \in \{stop,\, cont.\}} \mu_j(n, X_n)$$

ここで、$\mu_{stop}(n, X_n) = g(n, X_n)$ は停止に関連する報酬、$\mu_{cont.}(n, X_n)$ は最適経路が時間$n+1, \ldots, N$ で取られると仮定して時間n で継続に関連する報酬である。$\mu_{cont.}(n, X_n)$ は閉じた形を持たないが、シミュレーションを使って計算できることに注意。このプロセスは、実は、本論文の後の節で詳しく検討する「**?**」の仕事と非常によく似ている。重要な違いは、これらのランク応答面を採用することで、[Becker et al., 2018]が提案する高密度フィードフォワードニューラルネットワークに代わって、コンボルーションニューラルネットワークを採用することができる点である。

CNNのようなより繊細なネットワーク・アーキテクチャを採用することで、計算効率を向上させることができる。このことは、[Hu, 2019]によって、彼らのバミューダ・オプション価格シミュレーションが[Becker et al., 2018]よりも迅速かつ効率的に計算されたことが実証されています。しかし、彼らは、完全畳み込みニューラルネットワークの数学的収束理論が欠如しているため、彼らの方法が合理的な近似を提供することを数学的に確認することができなかったことに注意を促しています。一方、[Becker et al., 2018]は、密なフィードフォワードニューラルネットワーク（例えば[Leshno et al., 1993]）について確立された収束理論を採用し、彼らの近似の収束を確認することができます。

3 数学的枠組み

本節では、(1)のような最適停止問題を解くために、多層フィードフォワードニューラルネットワークを実装するのに必要な数学的枠組みに焦点を当てる。このために二つの重要

な定理を用いる。1つ目は、最適停止時間τが一連の0-
1停止判断の関数として表現できることを示す。もう1つは、この0-
1停止判定をフィードフォワード多層ニューラルネットワークで近似できることを示す。
これらの重要な結果は、最適停止問題を解くためにニューラルネットワークがどのように
利用できるかを数学的に立証している。

3.1 停止時間を一連の停止判断として表現すること

最初の課題は、マルコフ過程(X_n $_{n=o}$ に従って作ることができる。
)の停止を決定することを示すことである。^{N}
を関数の列$\{f_n(X_n)\}^N$,f_n :$\mathrm{R}^d \to \{0,1\}$とする。このために、まず、全体の
(1)の最適停止問題を停止問題群に変換する。具体的には、時間

*N*ストップ問題

$$V_n = \sup_{\tau \in T_n} \mathrm{E}g(\tau, X_\tau) \tag{3}$$

ここで、T_n は、$n \leqq \tau \leqq N$
となる全ての*X停止*時刻の集合である。明らかに、処理は時間Nで停止しなければならないので、$T_N = \{N\}$、したがって、時間N最適停止時間は$\tau_N = N$である。さらに
は0-1停止判定列なので、$f_n(X_n)\}_N$　　は0-1停止判定列なので、$f_N \equiv 1$と書き、したがっては、$\tau_N = Nf_N(X_N)$ と書く。

さて、時間Nの最適停止時間を0-1の停止判定の関数として書きましたが
$f_N(X_N), 0 \leq n \leq N-1$ に対して同じことをすることを目指す。意味するところは、時間 n 停止を書くことを目指す。
時間, τ_n , $\{f_k(X_k)\}_N$の関数とし$_{k=n}$て。
.我々は、以下の式を用いてそれを行うことを提案する。

$$\tau_n = \sum_{k=n}^{N} k f_k(X_k) \prod_{j=n}^{k-1} (1 - f_j(X_j)) \tag{4}$$

ここで、$f_N \equiv 1$.これはT_n
の停止時間を定義するが、この停止時間が最適かどうかはまだ確認されていない。次の定理は、τ_n が実際には時間 n の最適停止時間であることを示す。

定理1.*$n \in \{0, \ldots, N-1\}$に対して、$\tau_{n+1} \in T_{n+1}$ は、次のような形式であるとする。*

$$\tau_{n+1} = \sum_{k=n+1}^{N} k f_k(X_k) \prod_{j=n+1}^{k-1} (1 - f_j(X_j)) \tag{5}$$

すると、測定可能な関数f_n が存在する。$\mathrm{R}^d \to \{0, 1\}$ $\tau_n \in T_n$
を満たすような測定可能な関数が存在する。

$$\mathrm{E}g(\tau_n, X_{\tau_N}) \geqq V_n - (V_{n+1} - \mathrm{E}g(\tau_{n+1}, X_{\tau_{n+1}})) \tag{6}$$

ここで、V_n と V_{n+1} は(3)を*満たす。*

*証明する。*まず、任意の停止時間$\tau \in T_n$ を設定し、$E = V_{n+1} - \mathrm{E}g(\tau_{n+1}, X_{\tau_{n+1}})$
とすることから始める。
Doob-Dynkinにより、gは可積分なので、$\exists \mathrm{h}_n$ measurableで次のようになる。

$$h_n(X_n) = \mathrm{E}[g(\tau_{n+1}, X_{\tau_{n+1}}) | X_n]\text{である。}$$

さて、(4)により、以下も$X_{n+1}, \ldots, X_N$の可測関数である。

$$g(\tau_{n+1}, X_{\tau_{n+1}}) = \sum_{k=n+1}^{N} g(k, X_k) I_{\tau_{n+1}=k} = \sum_{k=n+1}^{N} g(k, Xk)_{If_k(X_k) \prod_{j=n}^{k-1} (1-f_j(X_j))=1}$$

これは、$g(\tau_{n+1}, X_{\tau_{n+1}})$が$X_{n+1}, \ldots, X_N$
において測定可能であることを示している。なぜなら、それは$X_{n+1}, \ldots, X_N$

の測定可能関数の和として書き直せるからである。したがって、$g(\tau_{n+1}, X_{\tau_{n+1}})$は$X_{n+1}, \ldots, X_N$において測定可能であるから、次が成り立つ。
というのは、$(X_n)_{n=0}^{N}$がマルコフ型であることから

$$h_n(X_n) = \mathrm{E}[g(\tau_{n+1}, X_{\tau_{n+1}}) | X_n] = \mathrm{E}[g(\tau_{n+1}, X_{\tau_{n+1}}) | F_n]$$である。

ここで、以下の集合がF_nにあることを立証することができる。

$D = \{g(n, X_n) \geqq h_n(X_n)\}$となる。　であり　$E = \{\tau = n\}$とする。

さらに、イベント $\tau_n = nI_D + \tau_{n+1} I_{D^c}$ は T_n にあり、$\tau\sim = \tau_{n+1} I_E + \tau I_{E^c}$ は T_{n+1} にある。したがって、以下のことがわかる。

$$\begin{aligned} \mathrm{Eg}(\tau, X_{\tau_{n+1}}) &= V_{n+1} - E \\ &= \sup_{\tau \in Tn} \mathrm{Eg}(\tau, X_\tau) - E \qquad \text{の定義により、} V_n \\ &\geqq \mathrm{Eg}(\tau\tilde{}, X_{\tau\tilde{}}) - E \end{aligned}$$

以下の通りです。

$$\mathrm{E}[g(\tau, X_{\tau_{n+1}}) I_{E^c}] \geqq \mathrm{E}[g(\tau\tilde{}, X_{\tau\tilde{}}) I_{E^c}] - E = \mathrm{E}[g(\tau, X_\tau) I_{E^c}] - E \qquad (7)$$

$I_{E^c} = 1$ は $\tau\sim = \tau$ を意味するので、これが成り立つことに注意。最後に、以下のようになる。

$\mathrm{Eg}(\tau_n, X_{\tau_n}) = \mathrm{E}[g(n, X_n) I_D + g(\tau_{n+1}, X_{\tau_{n+1}}) I_{D^c}]$である。　τの定義により$_{n+1}$

$= \mathrm{E}[g(n, X_n) I_D + h_n(X_n) I_{D^c}]$である。　$h_n(X_n)$の定義による。

$\geqq \mathrm{E}[g(n, X_n) I_E + h_n(X_n) I_{E^c}]$となる。　$g(\mathrm{n}, X_n) \geqq h_n(X_n)$ ならば $I_D = 1$ であるから。

$= \mathrm{E}[g(n, X_n) I_E + g(\tau_{n+1}, X_{\tau_{n+1}}) I_{E^c}]$である。　h の定義により$_n(X_n)$

$\geqq \mathrm{E}[g(n, X_n) I_E + g(\tau, X_\tau) IEC] - E$　(7)により

$= \mathrm{Eg}(\tau, X_\tau) - E$

これは任意のτについて成り立つので（任意に固定したため）、$\mathrm{Eg}(\tau_n, X_{\tau_n}) \geqq V_n - E$となり、式6が成立する。

Lastly, we must show that the τ_n we found here is the same as that in 4. Defining $f_n : \mathrm{R}^d \to \{0, 1\}$ by

$$_n f(x) \begin{matrix} 1 & \text{if } g(n, x) \geq h_n(x) \\ 0 & \text{if } g(n, x) < h_n(x) \end{matrix}$$

は、$I_D = f_n(X_n)$ となる。したがって

$\tau_n = nf_n(X_n) + \tau_{n+1}(1 - f_n(X_n))$　τの定義により$_n$

$= \sum_{k=n}^{N} kf_k(X_k) \prod_{j=n}^{k-1} (1 - f_j(X_j))$　定理文のτ_{n+1} の定義により

必要に応じて

□

*備考*定理1は、式(4)で与えられる停止時間を用いて、V_n を計算できることを示している。すなわち、不等式(6)は明らかに次のように等価である。

$$\text{スーパー}_{\tau \in T_{n+1}} \mathrm{Eg}(\tau, X_\tau) - \mathrm{Eg}(\tau_{n+1}, X_{\tau_{n+1}}) \geq \sup_{\tau \in Tn} \mathrm{Eg}(\tau, X_\tau) - g(\tau_n, X_{\tau_N})$$

したがって、$\tau_N = Nf(X_N)$ がわかっているので、逆帰納法により、τ_n が十分な最適停止時間を提供することがわかる。

*備考*定理1により、$V = \sup_{\tau \in T} Eg(\tau, X_\tau)$
に対応する全体最適停止時間は次式で与えられることが分かる。

$$\tau = \bigsqcup_{n=1}^{N} n f_n(X_n) \prod_{k=0}^{n-1} (1 - f_k(X_k))$$

3.2 ニューラルネットワークの導入

の系列の関数として最適な停止時間が計算できることを示しました。

0-1停止判定$\{f_n\}_{n=0}^{N}$に対して、その関数を近似的に求める方法が必要である。の方法は Becker et al., 2018]が提案するこれを行うには、ニューラルネットワークを採用することである。具体的には、$f^{\theta_n}: \mathrm{R}^d \to \{0, 1\}$という形式のニューラルネットワークの列を、パラメータ$\theta_n \in \mathrm{R}^q$で構築し、$f_n$を近似する。そして、$\tau_{n+1}$を単純に以下のようにして近似することが可能である。

$$\prod_{k=n}^{N} f^{\theta_k}(X_k) \prod_{j=n}^{k-1} (1 - f^{\theta_j}(X_j)) \tag{8}$$

なお、時刻 n の継続値を計算するためには τ_{n+1} が必要なので、ステップ n では τ_n ではなく τ_{n+1} を参照している、$V_{n+1} = \mathrm{E}[g(\tau_{n+1}, X_{\tau_{n+1}})]$ とする。後述するように、V_{n+1}はニューラルネットワークの訓練に用いる報酬関数の重要な部分である。

最初に取り組むべき問題は、f^{θ_n} は $\{0, 1\}$ の値のみを出力し、その結果 θ_n に関して連続的ではないことである。これは、
ニューラルネットワークのパラメータ学習において、通常、
ネットワークの何らかの報酬関数（または何らかの損失関数）を
最大化するために勾配ベースの最適化アルゴリズムを適用する
ことに起因する問題である。この問題に対処するため、我々は、パラメータを学習する際に多層のフィードフォワードニューラルネットワークで、区間(0, 1)の確率を出力します。そして、学習後、結果を0-1の停止判断に変換することができる。すなわち、$\theta \in \{\theta_0, \ldots, \theta_N\}$に対して、$F^\theta : \mathrm{R}^d \to (0, 1)$という形のニューラルネットワークを導入する。

$$F^\theta = \psi \circ a_3^\theta \circ \varphi_{q_2} \circ a_2^\theta \circ \varphi_{q_1} \circ a_1^\theta \tag{9}$$

どこ

- q_1 と q_2 は隠れ層のノード数である。
- $a_1^\theta : \mathrm{R}^d \to \mathrm{R}^{q_1}, a_2^\theta : \mathrm{R}^{q_1} \to \mathrm{R}^{q_2}, a_3^\theta : \mathrm{R}^{q_2} \to \mathrm{R}$ は以下の形式の線形関数である。

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

  ここで、$A_1 \in \mathrm{R}^{q_1 \times d}, A_2 \in \mathrm{R}^{q_2 \times q_1}, A_1 \in \mathrm{R}^{1 \times q_2}$は行列、$b_1 \in \mathrm{R}^{q_1}, b_2 \in \mathrm{R}^{q_2}, b_3 \in \mathrm{R}^{q_3}$はベクターである。

- $\varphi_{q_i} : \mathrm{R}^{q_i} \to \mathrm{R}^{q_i}$ はReLU活性化関数で、各コンポジットの最大値を取るだけです。nentと0であることから、$\varphi_q(x_1, \ldots, x_q) = (x_1^+, \ldots, x_q^+)$を意味する。

- $\psi : \mathrm{R} \to \mathrm{R}$ はロジスティックシグモイド関数で、$\psi(x) = 1/(1 + e^{-x})$ となる。

なお、このネットワークで調整すべきパラメータは、$\theta = \{A_1, A_2, A_3, b_1, b_2, b_3\} \in \mathrm{R}^q$, ここで $q = q_1(d + q_2 + 1) + 2q_2 + 1$ である。(9)の学習により最適なθ_nを求めたら、次にネットワーク f^θ を定義する。$\mathrm{R}^d \to \{0, 1\}$ を以下のように定義する。

$$f\theta = I_{[0,\infty]} \circ a\theta \circ \varphi_{q_2} \circ a_2^\theta \circ \varphi_{q_1} \circ a_1^\theta \tag{10}$$

なお、[Becker et al., 2018]の論文では、この置換が有効であることを示す正式な証明は含まれていない。しかし、$F^{\theta}{}^{n}$を時間nで停止することが最適である確率と考えると、直感的に理解できるのです。

停止判定を滑らかにし、0-1停止判定に戻す方法を定義したので、損失関数または報酬関数を定義しなければならないが、その順序は次のとおりである。

を使ってパラメータ θ_n
を調整する。各時間ステップで、我々の目的は我々の期待される将来のリワードを最大化することである。時間nにおいて、もし我々が停止すれば$(f_n(X_n) = 1)$ $g(n, X_n)$ の報酬を受け取り、もし我々が継続すれば $(f_n(X_n) = 0)$ そして最適な振る舞いを続ければ、定理1により最終的に$g(\tau_{n+1}, X_{\tau_{n+1}})$ の再ワードを受け取ることを知っています。したがって、各時間ステップで、我々は関数f:を最大化する$\mathrm{R}^d \to \{0, 1\}$。

$$\mathrm{E}[g(n, X_n)f(X_n) + g(\tau_{n+1}, X_{\tau_{n+1}})(1 - f(X_n))]\text{である。} \tag{11}$$

次の定理により、(11)のfをf^θ
に置き換えることができる。これにより、以下を最大化することができる。
(11) を$\theta \in \mathrm{R}^q$ に関して、最適な関数fを求める必要はない。$\mathrm{R}^d \to \{0, 1\}$を求める。

定理2. *$n \in \{0, \ldots, N-1\}$とし、停止時間$\theta_{n+1} \in T_{n+1}$を固定すると、$\forall E > 0$、存在する。$q_1, q_2 \in \mathrm{N}^+$のようなものである。*

$$\sup_{\theta \in Rq} \mathrm{E}[g(n, X_n)f^\theta(X_n) + g(\tau_{n+1}, X_{n+1})(1 - f^\theta(X_n))] \text{ となります。}$$
$$X)_\tau$$
$$\geq \sup_{f \in D} \mathrm{E}[g(n, X_n)f(X_n) + g(\tau_{n+1}, X_{\tau_{n+1}})(1 - f(X_n))] \quad E \tag{12}$$

ここで、Dはすべての測定可能な関数fの集合である。$\mathrm{R}^d \to \{0, 1\}$である。

*証明する。*gは可積分と仮定しているので、$\mathrm{E}|g(n, X_n)| < \infty$が成り立つことを思い出してください。
$\forall n \in \{0, \ldots, N\}$であり、したがって$f\sim$を見つけることができる。$\mathrm{R}^d \to \{0, 1\}$は以下のように測定可能である。

$$\mathrm{E}[g(n, X_n)f\tilde{}\theta(X_n) + g(\tau_{n+1}, X_{n+1})(1 - f\sim\theta(X_n))] \text{ とする。}$$
$$, X)_\tau$$
$$\geq \sup_{f \in D} \mathrm{E}[g(n, X_n)f(X_n) + g(\tau_{n+1}, X_{\tau_{n+1}})(1 - f(X_n))] - E/4 \tag{13}$$

次に、$A = \{x \in \mathrm{R}^d \mid f\tilde{} = 1\}$に対して$f\sim = I_A$
とし、Aはボレル集合であることに注意する。gの可積劈条件により、以下の有限ボレル測度を定義することもできる。 :

$B \to \mathrm{E}[|g(n, X_n)|I_B(X_n)]$である。 であり $B \to \mathrm{E}[|g(\tau_{n+1}, X_{\tau_{n+1}})|I_B(X_n)]$である。

R^d

上の有限ボレル測度の稠密性から、以下のような$K \subseteq A$コンパクトが存在することがわかる

。

$$\mathrm{E}[g(n, X_n)I_K(X_n) + g(\tau_{n+1}, X_{\tau_{n+1}})(1 - I_K(X_n))]\text{である。}$$
$$\geqq E[g(n, X_n)f\tilde{}\theta(X_n) + g(\tau_{n+1}, X_{n+1})(1 - f\sim\theta(X_n))] \text{ とする。} - E/4 \tag{14}$$
$$, X)_\tau$$

ここで、距離関数$\rho_K : \mathrm{R}^d \to [0, \infty]$ を $\rho_K(x) = \inf_{y \in K} /x - y/_2$ で定義する。次に、連続関数$k_j : \mathrm{R}^d \to [-1, 1], j \in \mathrm{N}$ を次式で定義することができる。

$$k_j(x) = \max\{1 - j - \rho_K(x), -1\}.$$

にポイントワイズで収束するもので，$I_K - I_{K^c}$
。ルベスクの支配的収束定理により、以下のようになる。

$$\mathrm{E}[g(n, X_n)_{Ikj(Xn)\geqq 0} + g(\tau_{n+1}, X_{\tau_{n+1}})(1 - {}_{Ikj(Xn)\geqq 0})\,]\text{である。}$$

$$\geqq E[g(n, X_n) I_K(X_n) + g(\tau_{n+1}, X_{\tau_{n+1}})(1 - I_K(X_n))] - E/4 \tag{15}$$

Leshno et al., 1993]により、k_j
は測度0の点でのみ不連続性を持ち、有界であるため、コンパクト集合上で一様に関数 h : $\mathrm{R}^d \rightarrow \mathrm{R}$ で近似することが可能であり、ここで

$$h(x) = \sum_{i=1}^{r} (v_i^T x + c_i)^+ - \sum_{i=1}^{s} (w_i^T x + d_i)^+$$

(16)である。

は、r, s∈N, v_1 , ..., v_r , w_1 , ..., w_s ∈R^d , c_1 , ..., c_r , d_1 , ..., d_s ∈R. に対して、次式が成り立つ。

$$\mathrm{E}[g(n, X_n)Ih_{(Xn)\geq 0} + g(\tau_{n+1}, X_{\tau_{n+1}})(1 - Ih_{(Xn)\geq 0})\ ]\text{である。}$$

$$\geqq E[g(n, X_n)_{Ikj\,(Xn)\geqq 0} + g(\tau_{n+1}, X_{\tau_{n+1}})(1 - {}_{Ikj\,(Xn)\geqq 0})] - E/4 \tag{17}$$

$f^{\theta} = I_{[0,\infty]} \circ a^{\theta} \circ \varphi_q \circ a^{\theta}$ ⅜ $\varphi_{q\ 2} \circ a^{\theta}_2$を想起し₁てください。したがって、$I_{[0,\infty]} \circ h$
を以下のニューラルネットワークとして表現することができるのです。
は、十分に大きな q_1 , q_2 に対して f^{θ} という形になり、(17)
が意味するところは、次のようになる。

$$\mathrm{E}[g(n, X_n)f^{\theta}(X_n) + g(\tau_{n+1}, {}_{n+1})(1 - f^{\theta}(X_n))] \text{となります。}$$
$$X)_{\tau}$$

$$\geqq E[g(n, X_n)_{Ikj\,(Xn)\geqq 0} + g(\tau_{n+1}, X_{\tau_{n+1}})(1 - {}_{Ikj\,(Xn)\geqq 0})] - E/4 \tag{18}$$

したがって、式（13）、式（14）、式（15）、式（18）を組み合わせると、次のようになる。

$$\sup_{\theta\in Rq} \mathrm{E}[g(n, X_n)f^{\theta}(X_n) + g(\tau_{n+1}, {}_{n+1})(1 - f^{\theta}(X_n))] \text{となります。}$$
$$X)_{\tau}$$

$$\geq \sup_{f\in D} \mathrm{E}[g(n, X_n)f(X_n) + g(\tau_{n+1}, X_{\tau_{n+1}})(1) \quad f(X_n))] \quad E - (19) -$$

必要に応じて

□

以下の結果は、定理1、2から直接得られるものである。

定理1. *任意の* $E > 0$ *と停止問題* $\sup_{\tau\in T} \mathrm{E}g(\tau, X_{\tau})$ *に対して、* $q_1, q_2 \in \mathrm{N}^+$ *と、* $f^{\theta_N} \equiv 1$ *と停止時間* (10) *の形のニューラルネット関数が存在する。*

$$\tau\hat{} = \prod_{n=1}^{N} \prod_{k=0}^{n-1} \mathrm{rr}\ \eta f^{\theta_n}(X_n)\ (1 - f^{\theta_k}(X_k)) \tag{20}$$

は、$\mathrm{E}g(\tau\hat{}, X_{\tau\hat{}}) \geq \sup_{\tau\in T} \mathrm{E}g(\tau, X_{\tau})$ -
E*を満たす。*

ここまでで、(1)の形の問題の解をニューラルネットワークの列{f^{θ}
$_n$}を用いて近似できることがわかったが、ここでf^{theta}
nは(10)の形である。次節では、このニューラルネットワークを実際にどのように計算するのかについて詳しく説明する。

3.3 インプリメンテーション

ここでは、実際に(1)のVを
どのように近似するかについて説明する。まず、シミュレーションとして
マルコフ過程のM*個の*独立な経路 $(X_n)^N$. これらのサンプル経路を (x^m) とする。^{N} にとって
$_{n=0}$ $_{n\ n=0}$
$m = 1, ..., M$.次に、時間N で停止しなければならないので、$f^{\theta_N}(x^m) \equiv 1, \ \forall m$
とする。ここで、θを計算することができる。$_n$
は、$n = N - 1, ..., 0$について再帰的に後退する。

時間ステップnにいると仮定する。つまり、$f^{\theta_{n+1}}, ..., f^{\theta_N}$をすでに計算したことになる。

このとき、

m 個の経路のそれぞれについて、時間 $n+1$ の最適停止時間を次のように計算する。

$$\tau_{n+1}^{m} = \sum_{k=n+1}^{N} k f^{\theta_k}(x_n^m) \prod_{j=n+1}^{k-1} (1 - f^{\theta_j}(x_j^m))$$

パスmに沿った時間nにおいて、確率F^{θ_n}で停止し、その後停止判断$f^{\theta_{n+1}}, \ldots, f^{\theta_N}$を守る場合、実現された報酬は次のようになることに注意。

$$r_n^m(\theta_n) = g(n, x_n^m) F^{\theta_n}(x_n^m) + g(\tau_{n+1}^m, x_{\tau_{n+1}^m}^m)$$

十分大きなMに対して、$\mathrm{E}[g(n, X_n)f^{\theta}(X_n) + g(\tau_{n+1}, X_{\tau_{n+1}})(1 - f^{\theta}(X_n))]$ は、approximately
とすることができる。
はたと

$$\frac{1}{M}\sum_{m=1}^{M} r_n^m(\theta_n) \tag{21}$$

であり、構成上θに関して滑らかな関数である。したがって、(21)は、θ_n
に関して勾配降下アルゴリズムによって最適化したい関数として働く。後ほど詳述する計算例では、このAdam法を採用する。そして、最適なθ_n
を得た後、前節で述べたように、得られたF^{θ_n}を0-1停止判定、f^{θ_n}に変換する。

これを $n = N - 1, \ldots, 0$ について繰り返した後、今度は新しいサンプルパスの集合 $(y^m)_{n=0}^{N}$
について生成します。
$m = 1, \ldots, M$.これがテストデータである。ここで、$n = N - 1, \ldots, 0$
について、再帰的に、以下を使用する。
学習データを用いて求めたパラメータ $\{\theta_n\}$ を計算する。

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

m個のサンプルパスのそれぞれに沿って
時間ゼロになったら、現在の最適な停止時間を計算して終了します

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

であり、(20)式から$\hat{\tau}$の推定値である。これに対応する$\mathrm{Eg}(\hat{\tau}, X_{\hat{\tau}})$の推定値は次式で与えられる。

$$\hat{V} = \frac{1}{M}\sum_{m=1}^{M} g(\tilde{\tau}_m, y_{\tilde{\tau}_m}^m) \tag{22}$$

なお、$\hat{V}$は大数の法則により$\mathrm{Eg}(\hat{\tau}, X_{\hat{\tau}})$に対して整合的な推定量である。また、$\mathrm{Eg}(\hat{\tau}, X_{\hat{\tau}})$の標準的なモンテカルロ推定値に過ぎないので、これも不偏である。さらに、大数の法則により
中心極限定理により、$\mathrm{Eg}(\hat{\tau}, X_{\hat{\tau}})$の$1-\alpha, \alpha \in (0, 1)$の信頼区間は次のようになる。

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

ここで、$z_{\alpha/2}$は$N(0, 1)$の$1-\alpha/2$分位であり、σ^は対応する標本標準偏差で、次式で与えられます。

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

*備考*Becker et al., 2018] において、$\hat{V}$が
は、$\mathrm{Eg}(\hat{\tau}, X_{\hat{\tau}})$

)の不偏推定値であり、それは
はsup$\tau \in T$ Eg(τ, X_τ)の不偏推定値ではない。しかし、彼らはまた、自分たちの例では
を見たところ、V^*が良い結果でした。

4 事例紹介

以下のセクションでは、最適停止問題を解くためにこの手順を適用したいくつかの例を見ていきます。最初に見るのはBermudan max-call optionで、[Becker et al., 2018]で議論された手順を用いて我々自身がPythonで実装したものである。我々が見る2つ目の例は、分数ブラウン運動を最適に停止する問題である。なお、この方法は私たち自身が実装したものではないので、[Becker et al., 2018]の計算結果を用いて議論することになります。

4.1 バミューダンマックスコールオプション

時刻Tに満期を迎えるバミューダンマックスコールオプションは、d個の資産$\{X_t^i\}_{i=1}^d$に対して書かれたオプションで、次のようになります。
は、権利行使価格Kでd資産の1つを購入する権利はあるが、義務はない。
時間格子上の点 $0 = t_0 < t_1 < \ldots < t_N = T$.

ブラック・ショールズ市場モデルであるため、資産価格は幾何学的なブラウン運動に従うと仮定します。

$$X_t^i = (r - \delta_i)dt + \sigma_i dW_t^i \tag{23}$$

どこ

- x_0^i は時刻0の株価
- $r \in [0, \infty]$ はマーカリスクフリーリターンレート
- $\delta_i \in [0, \infty]$ は配当利回り
- $\sigma_i \in [0, \infty]$ は資産ボラティリティ
- W_t^i は d 次元ブラウン運動 W の i^{th} 成分である。ここでは瞬間的な資産間の相関は$\rho_{ij} = 0$である。

時刻tにおける$\text{オプションの対応するペイオフ関数は、}(\max_{i\in\{1,\ldots,d\}} X_t^i - K)^+$
の形となる。
であることから、その価格は次式で与えられる。

$$\text{スーパーエ}_{\tau}\left[e^{-r\tau}\left(\text{マックス}_{i\in\{1,\ldots,d\}} X_\tau^i - K\right)^+\right] \tag{24}$$

とする。

ここで、いくつかの簡単な表記法の変換を適用して、このオプション価格問題を(1)の形の最適停止問題として書くことができる。オプションは $t_0, t_1, \ldots, t_N$ でしか行使できないので、(24) は $\sup_{\tau\in T} Eg(\tau, X_\tau)$ と書くことができる、ここで

$$g(n, x) = e^{-rt_n}\left(\text{マックス}_{i\in\{1,\ldots,d\}} x^i - K\right)^+ \tag{25}$$

とする。

また、等距離の時間グリッドを仮定しているため、$t_n = n \cdot T/N$ となり、$X_n \equiv X_{t_n}$ for $n = 0, \ldots, N$ … となる。

この例では、$i = 1, \ldots, d$ に対して、以下のようになります。

$x_0^i = 90,$ $K = 100,$ $\sigma_i = 20\%$。 $\delta_i = 10\%$。 $r = 5\%,$ $T = 3,$ $N = 9$

そこで、(23)に従って資産価格を次のようにシミュレートする。

$$x_{n,i}^m = x_{o,i} \cdot \exp\left(\sum_{k=o}^{n} \left((r - \delta_i - \sigma^2/2)\Delta t + \sigma_i \sqrt{\Delta t} \cdot Z_{k,i}^m \right) \right)。 \quad (26)$$

ここで、$\Delta t = T/N$ および $Z^m_{k,i} \sim N(0, 1)$.

この例を実装するために使用したコードの全体は、付録で入手可能であることに注意してください。このセクションでは、この例がどのように実装されているかを適切に説明するために、いくつかの重要なステップを含むコードのブロックにハイライトを当てるだけです。Pythonでニューラルネットワークを構築するために、Pytorchパッケージを使用します。これにより、以下の式に対応するニューラルネットワークを構築することができます。
(9)を次のとおりとします。

```
class NeuralNet(torch.nn.Module):
    def init (self, d, q1, q2):
        super(NeuralNet, self).init()
        self.a1 = nn.Linear(d, q1)
        self.relu = nn.ReLU()
        self.a2 = nn.Linear(q1, q2)
        self.a3 = nn.Linear(q2, 1)
        self.sigmoid = nn.シグモイド()

    def forward(self, x):
        out = self.a1(x)
        out = self.relu(out)
        out = self.a2(out)
        out = self.relu(out)
        out = self.a3(out)
        out = self.シグモイド(out)
        引揚げる out
```

なお、[Becker et al., 2018]のレコメンデーションに従って、$q_1 = q_2 = d + 40$とした。Pytorchでニューラルネットワークを構築することで、最小化したい損失関数（我々の場合は(21)の負）をカスタマイズすることも可能であることに留意してください。これはPythonで以下のように定義されています。

```
def loss(y_pred, s, x, n, tau):
    r_n = torch.zeros((s.M))
    for m in range(0, s.M):
        r_n[m] = -s.g(n,m, x)* y_pred[m] - s.g(tau[m],m, x)*(1 - y_pred[m])
    return(r_n.mean())
```

なお、上記のコードにおけるs.gは、単純に式$g(n, x)$で与えられる関数である。
e^{-rt_n} (最大 x^i K n とな る

また、時間 n のニューラルネット f^{θ} n に対して、最適なパラメータ θ_n を求める関数 NNtime を python で定義している。この関数では、最適な重みを選択するためのフィードフォワード、バックプロパゲーションアルゴリズムが図示されています。各エポック、つまりアルゴリズムに学習データセット全体を提示するごとに、まず、前のエポックで設定したθ_n を使って F^{θ}_n を計算する。次に、バックプロパゲーションによって対応する損失関数の勾配を計算し、Adam最適化アルゴリズムに従ってθ_n を更新します。これは以下のPythonで実行される。

```
def NN(n,x,s, tau_n_plus_1 ):
    epochs= 50
    model= NeuralNet( s. d, s. d+ 40 , s. d+ 40 )
    オプティマイザ = torch .optim.Adam ( model. parameters (), lr = 0 .0001 )

    for epoch in range ( epochs):F
        = model.forward ( X[
        n]) optimizer.zero_grad ()
        criterion = loss(F,S,X,n, tau_n_plus_1 )
        criterion .backward ()
        オプティマイザーのステップ()

    return F, model
```

先に言及したように、今回採用する最適化アルゴリズムは、[Kingma and Ba, 2014]に詳述されているように、Adam Methodと呼ばれるものである。更新方式そのものは、以下のアルゴリズム1に詳述されています。

アルゴリズム1：アダム法

入力：α（ステップサイズ）、$r(\theta)$（目的関数）、θ^0（初期パラメータ）

1 $\beta_1 = 0.9; \beta_2 = 0.99; E = 10^{-8}$ (アルゴリズムのデフォルト)を設定する。
2 $m_0 = v_0 = 0$
3 $t = 0$
4 **while** $r(\theta_t)$ *not minimized* **do**
5 $t = t + 1$
6 $G_t = \nabla_\theta r(\theta_{t-1})$ (バックプロップにより得られる)
7 $m_t = \beta_1 \cdot m_{t-1} + (1 - \beta_1) \cdot G_t$ (1st モーメント推定値)
8 $v_t = \beta_2 \cdot v_{t-1} + (1 - \beta_2) \cdot (G_t)^2$ (2nd 瞬間推定値)
9 $\hat{m} = m_t / (1 - (\beta_1)^t)$ (バイアス補正)
10 $\hat{v}_t = v_t / (1 - (\beta_2)^t)$ (バイアス補正)
11 $\theta_t = \theta_{t-1} - \alpha \cdot \hat{m} / (\sqrt{\hat{v}_t + E})$

出力：θ^t (更新されたパラメータ)

AdamはAdaptive Moment estimationの略語である。この名前のAdaptiveは、頻繁に現れる特徴に対してはより小さな更新を行い（つまり学習率が低くなる）、あまり現れない特徴に対してはより大きな更新を行うアルゴリズムであることを意味している。これは、過去に計算されたすべての勾配に基づいて、各反復で学習率をスケーリングすることによって行われる。これは、アルゴリズム1の11行目の更新条件の形で示されている。m_t と v_t は勾配の第1および第2モーメントの推定値とみなすことができるため、名前のモーメント部分はアルゴリズム1の7行目と8行目で明らかである。

なお、時間ステップnで最適なθ_nを計算した後、これを用いて、次のように計算を進めます。で詳述したように、次のステップで対応する最適な停止時刻を求める。各タイムステップのパラメータ $\{\theta_n\}$ を求めたら、次に新しいセット の幾何学的ブラウン運動をテストデータとし、前記$\{\theta_n\}$ を用いて (22) の $V\hat{}$ を計算する。

このアルゴリズムを、$M = 5,000$ サンプルパスを用いて、$d = 2, 5, 10$ の原資産について実行しました。我々の推定値は表1に示されています。

[1]実際の値は、$d = 2$については[Hu, 2019]から、$d = 5, 10$については[Becker et al, 2018]から、それぞれ$M = 160,000$サンプルパスで計算された推定値である。

d	実績[1]	見積もり	標準誤差	95%信頼区間
2	8.04	7.50	0.23	(7.05, 7.95)
5	16.64	16.80	0.32	(16.18,17.43)
10	26.20	23.83	0.29	(23.25, 24.40)

表 1: V の推定値を表にしています。結果は、$M = 5,000$ サンプルパスを用いて計算された。

なお、表1の我々の推定値は、私自身の限られたコーディング能力と私のノートパソコンの限られた計算能力の両方により、我々のコードが実行時間の点で完全に最適化されていないため、$M = 5,000$サンプル経路のみを使用して計算されました。そこで参考までに、[Hu, 2019] (for $d = 2$) と [Becker et al., 2018] (for $d = 5$ and $d = 10$) の結果は、$M = 160,000$ sample pathsを使って計算されたので、含まれています。しかし、我々は$M = 5,000$のサンプル資産価格経路しか使用していないことを考えると、我々の結果はまだ[Hu, 2019]と[Becker et al., 2018]の結果にかなり近く、我々の実装が正しいことを示唆するものであった。

また、[Becker et al., 2018]は$d = 500$の資産に書き込まれた最大コールオプションの推定を含み、それでもわずか533.5秒の計算時間を誇っていることに注目すべきです[2]。これは、最適停止問題に対してニューラルネットワークのアプローチを採用した場合に実現可能となる、計算効率とそれに対応する合理的な推定値の生成を強調しています。

4.2 フラクショナルブラウン運動の最適な停止方法

最後の例では、[Becker et al., 2018]の仕事について、分数ブラウン運動を最適に停止するために私たちのニューラルネットワークアルゴリズムを採用することを説明します。我々は、当該プロセスの次元を増加させることによって、非マルコフ型プロセスをマルコフ型にすることができる方法を紹介します。従来の数値計算手法では、この次元の増加により、この問題は計算不可能になっただろう。しかし、我々のニューラルネットワークのアプローチを用いることで、このより複雑なプロセスの最適停止値を近似的に求めることができる。

まず、分数ブラウン運動は、標準ブラウン運動の一般化であることを想起する。具体的には、ハーストパラメーターHを持つ分数ブラウン運動は、平均0、共分散構造

$$\mathrm{E}[W_t^H W_s^H] = \frac{1}{2}\left(t^{2H} + s^{2H} - |t-s|^{2H}\right) \tag{27}$$

なお、標準ブラウン運動は、単純にハーストパラメータH=1/2のフラクショナルブラウン運動である。

時刻0と時刻1の間で最適に停止した分数ブラウン運動の値を推定したい。

≤τ≤1 イーダブリュー$_H^\tau$ (28)

まず最初に、任意停止定理を思い出してください。これは、$X = (X_t)_{t\geq 0}$ をマルチンゲールとすると、次のようになります。τはX停止時間であり、τまたは$X_{\tau\wedge t}$のいずれかが有界であれば、$EX_\tau = EX_0$ となる。したがって、標準的な

0≤τ≤1 τ t

ブラウン運動はマルチンゲールであることから、sup EW$^{1/2}$ = 0. ただし、$H \neq$ 1/2の場合、W^H は

[2]参考までに、$d = 5$の場合の彼らの推定値を計算するのに、[Becker et al., 2018]は42.3秒しかかかっていない。

はマルチンゲールではないので、任意の停止定理は適用されません。

そこで、(28)を近似するために、まず、時間間隔[0, 1]を100の時間ステップに離散化する、$t_n = \frac{n}{100}$, $n = 0, ..., 100$. そして、100次元のマルコフ過程、$(X_n)_{n=0}^{100}$ を記述します。
を作ることができる。$(W^H_{t_n})_{n=0}^{100}$ を以下のとおりとします。

$$x_0 = (0, 0, ..., 0)$$
$$x_1 = (W^H_{t_1}, 0, ..., 0)$$
$$x_2 = (W^H_{t_1}, W^H_{t_2}, ..., 0)$$
$$x_{100} = (W^H_{t_{100}}, W^H_{t_{99}}, ..., W^H_{t_1})$$

g とする。$\mathrm{R}^{100} \to \mathrm{R}$ を $g(x_1, ..., x_{100}) = x_1$ とすると、次のように計算することを目指します。

$$\sup_{\tau \in T} \mathrm{E} g(X_\tau) \tag{29}$$

ここで、τは全てのX停止時間の集合であり、(28)を近似するためである。

(29) を近似するために、[Becker et al., 2018] は $(X_n)_{n=0}^{100}$ Davies Harte法を介して、その
をシミュレーションした。
は、$Y^H_n = W^H_n - W^H_{n-1}$ は、各 $n = 1, ..., N$ に対して、定常ガウス過程を形成しています。
を定義しています。
自己共分散のある

$$|k + 1|2H - |k|2H + |k - 1|2H$$

$$\mathrm{E}[Y_n Y_{n+k}] = \frac{}{2(100^{2H})}$$

次に，(X_n)のサンプルパスを $W^H = \sum_{k=1}^{n} Y^H_k$ で計算する．
によって計算される。そして、ニューラルネットワークを学習させ
を、前のセクションで説明したように、$d = 100$、$q_1 = 110$、$q_2 = 55$の形で（10）し、モンテカルロ近似によって（28）近似した。Becker et al., 2018]によって計算された近似された最適停止解を図3にプロットしています。

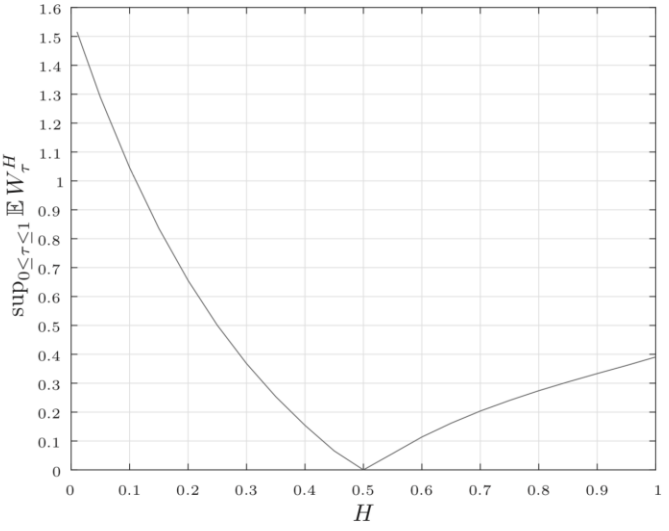


図3：[Becker et al.,
2018]の結果。最適に停止した分数ブラウン運動の期待値w.r.t.そのハーストパラメータH.

なお、H＜1/2の場合はH＞1/2の場合よりも最適に停止したW^H
の値が大きくなると予想されるので、図3の形状も直感的に理解できる。その理由を理解するために、極端な話

$H = 1$ の場合、W^H の増分は自己共分散が 1 に等しく、時刻 t で W^H が減少すると、t 以降のすべての時刻でその値が減少することを意味する。したがって、$H = 1$ の最適停止戦略は、W^H の減少が見られた場合にのみ停止することになる。 逆に、H が 0 に近い場合は強い負の自己相関を意味するので、市場はより大きく変動する傾向があると示唆する。この場合、W^H の値が時刻 t に低下しても、ある $s > 0$ の場合には時刻 $t + s$ に回復する可能性が高いので、$H = 1$ の場合よりも最適な戦略の選択肢が増える。

さらに、分数ブラウン運動を最適に停止するこの方法の有効性を示すために、[Becker et al., 2018]は、(28)を近似するために従来の数値離散化アプローチを採用している[Kulikov and Gusyatnikov, 2016]と結果を比較しています。ニューラルネットワークアプローチでは、$H < 1/2$ では [Kulikov and Gusyatnikov, 2016] の最大 5 倍、$H > 1/2$ では最大 3 倍の結果が得られています。

5 まとめ

本報告では、ニューラルネットワークを中心とした機械学習技術を用いることで、最適停止問題がどのように解決されるかに踏み込んでみた。これまで繰り返し述べてきたように、最適停止問題にこのような手法を適用する最大の動機は、いわゆる次元の呪いに対する有効な治療法であることである。高次元のマルコフ過程を含む最適停止問題に恐れを抱かなくなることは、様々な意味で解放されることになる。このレポートで見てきたように、多くの資産に対して書かれたオプションの最適な行使時間を求めることができ、また、分数ブラウン運動を高次元Markov過程に変換することによって最適停止させることができた。この意味で、我々が設定した枠組みは、離散化マルコフ過程に基づくあらゆる有限最適停止問題に適用できるほど一般的であるため、この方法を他の種類の最適停止問題の解決に採用できるかを見ることが、このプロジェクトの興味ある拡張部分である。

私たちの文献レビューは、最適停止問題へのニューラルネットワークのアプリケーションのトピックが同様にさらに探求され得る方法について、いくつかの興味深いアイデアをもたらす。Becker et al., 2018]で採用されたフィードフォワードネットワーク構造は、最もシンプルで最も古いニューラルネットワークアーキテクチャの一つである。したがって、[Hu, 2019]で提案された畳み込みニューラルネットワークのアプローチは、より計算効率の高いネットワークアーキテクチャを我々の同じ問題に適用することを提案しており、興味をそそられる。しかし、我々が以前に議論したように、これらのアーキテクチャに対して開発された収束理論が不足しており、それゆえ彼らは、我々が定理2で行ったように、彼らのニューラルネットワークが最適停止解の正確な近似を提供することを正式に証明することができなかった。したがって、今後の研究として、畳み込みニューラルネットワークの近似の収束特性を研究することが考えられる。特に最適停止問題で必要となる収束理論は、2の証明で採用した[Leshno et al., 1993]によるフィードフォワードネットワークの定理に相当する畳み込みニューラルネットワークであることに注意されたい。

また、[Becker et al., 2018]は、どの例においても隠れ層のノード数q_1 と q_2

をどのように選択するかについてあまり詳しく触れていない。したがって、さらなる研究には、この最適停止問題の文脈でq_1 と q_2 を最適に選択するための数学的枠組みの開発を検討することが含まれるかもしれません。また、再び[Becker et al., 2018]が2つの隠れ層ネットワークの選択についてほとんど洞察を提供しなかったように、採用された層の数が選択されるプロセスを見る価値があるでしょう。

最後に、このレポートに取り組んでいる間に浮かんだ疑問は、他のどのような数理ファイナンスの問題が、おそらくニューラルネットワークのアプローチを採用することで高次元で近似できるのかということです。最適停止問題は古典的な数理ファイナンスの問題で、これまでにも多くの研究がなされてきましたが、それに対する最も単純なニューラルネットワーク構造の適用に焦点を当てた論文が、2018年に発表されました。これは、高次元のマスキュラーファイナンス問題の解をニューラルネットワークなどの機械学習で近似するという、今後の研究テーマの豊富さを示しているのかもしれません。

6 付録

6.1 Pythonコード

```
numpyをnpとしてインポ
ートする インポート
torch
import torch . nn as nn
np. random . seed (
234198 )

インポート scipy .stats

種類株式:
    def init ( self , T, K, sigma , delta , So , r, N, M, d): self.T =
        T
        の自己紹介をします。  K=K
        self. sigma = sigma * np.
        ones( d) self. delta = delta
        self.So=So* np. ones( d)
        self. r=r
        self.N=
        N
        self.M=
        M
        self.d=d

    def GBM ( self):

        dt= self.  T/ self.  N So_vec=
        self.So* np. ones (( 1 , S. M, S. d))

        Z=np. random . standard_normal ( ( self. N, self. M, self. d)) s= self.So* np.
        exp ( np. cumsum (( self. r- self. delta - 0 . 5 * self. sigma ** 2 )* dt+ self.
                                                    シグマ * np. sqrt( dt)*Z, axis= 0 )).


        s=np. append ( So_vec , s, axis=
        0 ) return s


    def g ( self,n , m, X ): max1 = torch .max( X[ int(
        n),m,:]. float ()- self.K)

        return np. exp (- self. r*( self. T/ self. N)* n)* torch .max( max1 , torch .tensor([ 0
        . 0 ])).


#%%
class NeuralNet( torch . nn. Module ):
    def init ( self , d, q1 , q2 ):
        super( NeuralNet , self). init ()
        self. a1 = nn.Linear(d, q1 ) self.
        relu = nn.ReLU ()
        self. a2 = nn.Linear( q1 , q2 )
        self. a3 = nn.Linear( q2 , 1 )
        self. sigmoid =nn.シグモイド()

    def forward ( self , x): out
        = self.a1 ( x )
        out = self. relu (
        out) out = self. a2 (
```

```
out) out = self.
relu (out) out =
self. a3 (out)
out = self. シグモイド(out)
```

```
        引揚げる

def loss( y_pred ,s, x, n, tau ):
    r_n = torch .zeros ((s. M))
     for m in range (0 , s. M):

         r_n [ m] =- s. g(n,m, x)* y_pred [ m] - s. g( tau [ m],m, x)*( 1 - y_pred [ m])

     戻り値( r_n . 平均値())

#%%

S= stock (3 , 100 , 0 . 2, 0 . 1, 90 , 0 . 05 , 9, 5000 , 10 )

X= torch .from_numpy ( S. GBM ()). float ()
#%%

def NN(n,x,s, tau_n_plus_1
    ): epochs= 50
    model= NeuralNet( s. d, s. d+ 40 , s. d+ 40 )
    オプティマイザ = torch .optim .Adam ( model. parameters (), lr = 0 .0001 )

    for epoch in range ( epochs):F
        = model.forward ( X[
        n]) optimizer.zero_grad ()
        criterion = loss(F,S,X,n, tau_n_plus_1 )
        criterion .backward ()
        オプティマイザーのステップ()

    return F, model

mods=[ None ]* S. N
tau_mat=np. zeros (( S. N+1
, S. M)) tau_mat[ S. N,:] =S.
N. (S. N,:)
tau_mat=np.N

f_mat=np. zeros (( S. N+1 ,
S. M)) f_mat[ S. N,:] = 1

#%%
for n in range ( S. N-1 ,- 1 ,-1 ):
    probs , mod_temp =NN(n, X, S, torch . from_numpy ( tau_mat[ n+ 1 ]). float
    ()) mods[ n] = mod_temp
    np_probs= probs. detach (). numpy (). reshape ( S. M)
    print(n, ":", np. min ( np_probs),", ", np. max( np_probs))

    f_mat[n,:] =( np_probs > 0 . 5 )* 1 .0

    tau_mat[n,:] =np. argmax ( f_mat , axis= 0 ).

#%%
Y= torch .from_numpy ( S. GBM ()). float ()

tau_mat_test=np. zeros (( S. N+1 , S. M))
tau_mat_test[ S. N,:] =S.N

f_mat_test=np. zeros (( S. N+1 , S.
M)) f_mat_test[ S. N,:] = 1

V_mat_test=np. zeros (( S. N+1 , S. M))V_est_test=np.
zeros( S. N+ 1 )
```

```
for m in range (0 , S. M):V_mat_test[ S. N,
    m] =S. g( S. N,m, Y)

V_est_test[ S. N] =np. mean (V_mat_test[ S. N,:])



for n in range (S. N-1 ,-1 ,-1 ):
    mod_curr= mods[ n]
    probs= mod_curr( Y[
    n])
    np_probs= probs. detach (). numpy (). reshape (S. M)

    f_mat_test[n,:] =( np_probs > 0 . 5 )* 1 .0

    tau_mat_test [n,:]=np. argmax (f_mat_test ,

    axis= 0 )


    for m in range (0 , S. M):
        V_mat_test[n, m] =np. exp ((n- tau_mat_test [n, m])*(- S. r* S. T/ S. N))* S. g(
        tau_mat_test)
                                                    [n, m],m, X)
        # np. exp(( n- tau_mat_test[n, m])*(- S. r* S. T/ S. N))* S. g(
        tau_mat_test[n, m],m, X)

#%%
V_est_test=np. mean (V_mat_test ,axis= 1 )
V_std_test=np. std (V_mat_test ,axis= 1 )
V_se_test= V_std_test/( np. sqrt( S. M )
).

z= scipy . stats. norm . ppf( 0 . 975 )
lower= V_est_test[ 0 ] - z* V_se_test[ 0
] upper= V_est_test[ 0 ] + z* V_se_test[ 0 ]
.....1.

print( V_est_test[
0 ]) print(
V_se_test[ 0 ])
print( 下位 )
print( 上位 )
```

参考文献

[Becker et al., 2018] Becker, S., Cheridito, P., and Jentzen, A. (2018).Deep optimal stopping.Technical report.

[Hijazi et al., 2015] Hijazi, S., Kumar, R., and Rowen, C. (2015).画像認識のための畳み込みニューラルネットワークの使用。

[Hu, 2019] Hu, R. (2019).最適停止問題への応用を伴う応答曲面のランキングのための深層学習*.arXiv e-prints*.

[Kingma and Ba, 2014] Kingma, D. P. and Ba, J. (2014).Adam: A method for stochastic optimization.*CoRR*.

[Kohler et al., 2010] Kohler, M., Krzyak, A., and Todorovic, N. (2010).高次元のアメリカン・オプションのニューラルネットワークによるプライシング。*数理ファイナンス*, 20(3):383-410.

[Kulikov and Gusyatnikov, 2016] Kulikov, A. V. and Gusyatnikov, P. P. (2016).分数的ブラウニアンモーションの停止時間。In *Computational Management Science*, pages 195-200.Springer International Publishing.

[Leshno et al., 1993] Leshno, M., Lin, V. Y., Pinkus, A., and Schocken, S. (1993).非多項式活性化関数を持つ多層フィードフォワードネットワークは、あらゆる関数を近似することができる。*ニューラルネット*, 6:861-867.

[Longstaff and Schwartz, 2001] Longstaff, F. A. and Schwartz, E. S. (2001).シミュレーションによるアメリカのオプションの価値評価。によるアメリカン・オプションの評価：単純な最小二乗法によるアプローチ。このような場合、「金融*研究」（Review of Financial Studies*）、113-147ページを参照してください。

[Yu and Bertsekas, 2007] Yu, H. and Bertsekas, D. P. (2007).Q-learning algorithms for optimal stopping based on least squares.*2007 European Control Conference (ECC)*, pages 2368-2375.