2019 IEEE 58th Conference on Decision and Control (CDC)
Palais des Congrès et des Expositions Nice Acropolis
Nice, France, December 11-13, 2019

Deep Forward-Backward SDEs for Min-max Control

Ziyi Wang[1], Keuntaek Lee[1], Marcus A. Pereira[1], Ioannis Exarchos[2] and Evangelos A. Theodorou[1]

Abstract— **This paper presents a novel approach to numerically solve stochastic differential games for nonlinear systems. The proposed approach relies on the nonlinear Feynman-Kac theorem that establishes a connection between parabolic deterministic partial differential equations and forward-backward stochastic differential equations. Using this theorem the Hamilton-Jacobi-Isaacs partial differential equation associated with differential games is represented by a system of forward-backward stochastic differential equations. Numerical solution of the aforementioned system of stochastic differential equations is performed using importance sampling and a neural network with Long Short-Term Memory and Fully Connected layers. The resulting algorithm is tested on two example systems in simulation and compared against the standard risk neutral stochastic optimal control formulations.**

I. INTRODUCTION

Stochastic optimal control is a mature discipline of control theory with a plethora of applications to autonomy, robotics, aerospace systems, computational neuroscience, and finance. From a methodological stand point, stochastic dynamic programming is the pillar of stochastic optimal control theory. Application of the stochastic dynamic programming results in the so-called Hamilton-Jacobi-Bellman (HJB) Partial Differential Equation (PDE). Algorithms for stochastic control can be classified into different categories depending on the way of how they are dealing with the curse of dimensionality in solving the HJB PDE for systems with many degrees of freedom and/or states.

Game-theoretic, or min-max, extension to optimal control was first investigated by Isaacs [1]. He associated the solution of a differential game with the solution to a HJB-like equation, namely its min-max extension, also known as the Hamilton-Jacobi-Isaacs (HJI) equation. The HJI equation was derived heuristically under the assumptions of Lipschitz continuity of the cost and the dynamics, in addition to the assumption that both of them are separable in terms of the maximizing and minimizing controls. Despite extensive results in the theory of differential games, algorithmic development has seen less growth, due to the involved difficulties in addressing such problems. Prior work, including the Markov Chain approximation method [2], largely suffers by the curse of dimensionality. In addition, a specific class of min-max control trajectory optimization methods have been derived recently, relying on the foundations of *differential dynamic programming* (DDP) [3]–[5], which requires linear and/or quadratic approximation of the dynamics and value function.

[1]: Georgia Institute of Technology, Atlanta, GA, USA. [2] :Emory University, Atlanta, GA, USA. Email: zwang450@gatech.edu

Due to the inherent difficulties of solving stochastic differential games, most of the effort in optimal control theory was focused on the HJB PDE. Addressing the solution of the HJB equation, a number of algorithms for stochastic optimal control have been proposed that rely on the probabilistic representation of solutions of linear and nonlinear backward PDEs. Starting from the path integral control framework [6], the HJB equation is transformed into a linear backward PDE under certain conditions related to control authority and variance of noise. The probabilistic representation of the solution of this PDE is provided by the linear Feynman-Kac theorem [7]–[9]. The nonlinear Feynman-Kac theorem avoids the assumption required in the path integral control framework at the cost, however, of representing the solution of the HJB equation with a system of Forward-Backward Stochastic Differential Equations (FBSDEs) [10], [11]. Previous work by our group aimed at improving sampling efficiency and reducing computational complexity, and in [12]–[14] an importance sampling scheme was proposed and employed to develop iterative stochastic control algorithms using the FBSDE formulation. This work lead to algorithms for L^2, L^1, risk-sensitive stochastic optimal control, as well as stochastic differential games [15]–[17].

In [18] the authors incorporate deep learning algorithms, such as Deep Feed-Forward Neural Networks, within the FBSDE formulation and demonstrated the applicability the resulting algorithms to solving PDEs. While the approach in [18] offers an efficient method to represent the value function and its gradient, it has been only applied to PDEs that correspond to simple dynamics. Motivated by the limitations of the existing work on FBSDEs and Deep Learning (DL), the work in reference [19] utilizes importance sampling together with the benefits of recurrent neural networks in order to capture the temporal dependencies of the value function and to scale the deep FBSDE algorithm to high dimensional nonlinear stochastic systems.

In this work, we demonstrate that the FBSDEs associated with stochastic differential games can be solved with the deep FBSDE framework. We focus on the case of min-max stochastic control that corresponds to risk sensitive control. Using the Long Short-Term Memory (LSTM) network architecture [20], we introduce a scalable deep min-max FBSDE controller that results in trajectories with reduced variance. We demonstrate the variance reduction benefit of this algorithm against the standard risk neutral stochastic optimal control formulation of the deep FBSDE framework on a pendulum and a quadcopter in simulation.

The rest of this paper is organized as follows: in Section II we introduce the min-max stochastic control problem,

978-1-7281-1398-2/19/$31.00 ©2019 IEEE

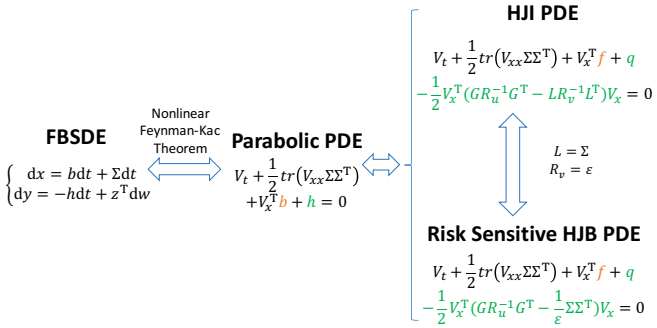

Fig. 1: A schematic diagram showing the relationship between PDEs and FBSDE. Terms in orange denote drift in FSDE, and terms in green denote drift in BSDE.

demonstrate its connection to risk sensitive control, and reformulate the problem with a system of FBSDEs. We present the min-max FBSDE controller in Section III. In Section IV, we compare the controller introduced in this work against the deep FBSDE algorithm for standard stochastic optimal control, and we explore the variance reduction benefit of our controller as a function of risk sensitivity. Finally, we conclude the paper in Section V.

II. FBSDE for Differential Games

A. Min-Max Stochastic Control

Consider a system with control affine dynamics in a differential game setting as follows:

$$\begin{aligned} \mathrm{d}x = f(x(t),t)\mathrm{d}t + G(x(t),t)u(t)\mathrm{d}t + L(x(t),t)v(t)\mathrm{d}t \\ + \Sigma(x(t),t)\mathrm{d}w(t) \quad t \in [\tau, T]. \end{aligned} \tag{1}$$

where $\tau \in [0,T]$, T is the task horizon, $x \in \mathbb{R}^n$ is the state, $u \in \mathbb{R}^p$ is the minimizing control, $v \in \mathbb{R}^q$ is the adversarial control, $w(t)$ is a standard m dimensional Brownian motion, $f : \mathbb{R}^n \times [\tau, T] \to \mathbb{R}^n$ represents the drift, $G : \mathbb{R}^n \times [\tau, T] \to \mathbb{R}^{n\times p}$ represents the actuator dynamics, $L : \mathbb{R}^n \times [\tau, T] \to \mathbb{R}^{n\times q}$ represents the adversarial control dynamics and $\Sigma : \mathbb{R}^n \times [\tau, T] \to \mathbb{R}^{n\times m}$ represents the diffusion. For this system we can define the following cost function:

$$\begin{aligned} &J(\tau, x_\tau; u(\cdot), v(\cdot)) = \\ &\mathbb{E}\left[g(x(T)) + \int_\tau^T \left(q(x(t),t) + \frac{1}{2}u^{\mathrm{T}} R_u u - \frac{1}{2} v^{\mathrm{T}} R_v v \right) \mathrm{d}t \right], \end{aligned} \tag{2}$$

where $g : \mathbb{R}^n \to \mathbb{R}^+$ is the terminal state cost, $q : \mathbb{R}^n \to \mathbb{R}^+$ is the running state cost, and $R_u \in \mathbb{R}^{p\times p}$ and $R_v \in \mathbb{R}^{q\times q}$ are positive definite matrices.

The min-max stochastic control problem is formulated as follows:

$$V(x_\tau, \tau) = \inf_{u\in\mathscr{U}} \sup_{v\in\mathscr{V}} J(\tau, x_\tau; u(\cdot), v(\cdot)), \tag{3}$$

where the minimizing controller's goal is reducing the cost under all admissible strategies $\mathscr{U}$, while the adversarial controller maximizes the cost under all admissible non-anticipating (meaning that at each instant of time, no future values of the opponent's control are known to each agent; see [21, equation 4] for more details) strategies $\mathscr{V}$.

The HJI equation for this problem is:

$$\begin{cases} V_t + \inf_{u\in\mathscr{U}} \sup_{v\in\mathscr{V}} \left\{ \frac{1}{2}\mathrm{tr}\left(V_{xx}\Sigma\Sigma^{\mathrm{T}}\right) + V_x^{\mathrm{T}}(f + Gu + Lv) \right. \\ \left. + q + \frac{1}{2}u^{\mathrm{T}}R_u u - \frac{1}{2}v^{\mathrm{T}}R_v v \right\} = 0, \quad (t,x) \in [\tau, T) \times \mathbb{R}^n, \\ V(x,T) = g(x), \quad x \in \mathbb{R}^n. \end{cases} \tag{4}$$

The terms inside the infimum and supremum operations are collectively called the Hamiltonian. The optimal minimizing and adversarial controls u and v are those for which the gradient of the Hamiltonian vanishes, which take the following form:

$$\begin{aligned} u(x(t),t) &= -R_u^{-1} G^{\mathrm{T}} V_x, \\ v(x(t),t) &= R_v^{-1} L^{\mathrm{T}} V_x. \end{aligned} \tag{5}$$

Substitution of the expressions above into the HJI equation results in:

$$\begin{cases} V_t + \frac{1}{2}\mathrm{tr}\left(V_{xx}\Sigma\Sigma^{\mathrm{T}}\right) - \frac{1}{2}V_x^{\mathrm{T}}\left(GR_u^{-1}G^{\mathrm{T}} - LR_v^{-1}L^{\mathrm{T}}\right)V_x \\ + V_x^{\mathrm{T}} f + q = 0, \quad (t,x) \in [\tau, T) \times \mathbb{R}^n, \\ V(x,T) = g(x), \quad x \in \mathbb{R}^n. \end{cases} \tag{6}$$

Note that we will drop functional dependence in all PDEs for notational compactness. In the following section we show the equivalence of a certain case of min-max control to risk sensitive control.

B. Risk Sensitive Stochastic Optimal Control

Risk sensitive stochastic optimal control [22] is essential in cases where decision has to be made in a manner that is robust to the stochasticity of the environment. Let us consider the following performance index:

$$\begin{aligned} &J(\tau, x_\tau; u(\cdot)) = \\ &\varepsilon \ln \mathbb{E}\left[\exp \frac{1}{\varepsilon}\left(g(x(T)) + \int_\tau^T q(x(t),t) + \frac{1}{2}u(t)^{\mathrm{T}} R u(t) \mathrm{d}t \right) \right], \end{aligned} \tag{7}$$

where $\varepsilon \in \mathbb{R}^+$ is the risk sensitivity. The risk sensitive stochastic optimal control problem is formulated with the following value function:

$$V(x_\tau, \tau) = \inf_{u\in\mathscr{U}} J(\tau, x_\tau; u(\cdot)), \tag{8}$$

subject to the dynamics:

$$\mathrm{d}x(t) = f(x(t),t)\mathrm{d}t + G(x(t),t)u(t)\mathrm{d}t + \sqrt{\frac{\varepsilon}{2\gamma^2}}\tilde{\Sigma}(x(t),t)\mathrm{d}w(t), \tag{9}$$

where $\gamma \in \mathbb{R}^+$ is a small constant, and $\tilde{\Sigma}$ represents diffusion [23].

The HJB equation for this stochastic optimal control problem is formulated as follows:

$$\begin{cases} V_t + \inf_{u\in\mathscr{U}} \left\{ \dfrac{\varepsilon}{4\gamma^2}\mathrm{tr}\left(V_{xx}\tilde{\Sigma}\tilde{\Sigma}^{\mathrm{T}}\right) + V_x^{\mathrm{T}}(f+Gu) + q \right. \\ \left. + \dfrac{1}{2}u^{\mathrm{T}}Ru + \dfrac{1}{4\gamma^2}V_x^{\mathrm{T}}\tilde{\Sigma}\tilde{\Sigma}^{\mathrm{T}}V_x \right\} = 0, \quad (t,x)\in[\tau,T)\times\mathbb{R}^n, \\ V(x,T) = g(x), \quad x\in\mathbb{R}^n. \end{cases} \tag{10}$$

The optimal control can be obtained by finding the control where the gradient of the terms inside the infimum vanishes and has the form $u(x(t),t) = -R^{-1}G^{\mathrm{T}}V_x$. By substituting in the optimal control and setting $\Sigma = \sqrt{\dfrac{\varepsilon}{2\gamma^2}}\tilde{\Sigma}$ in (10), we get the following final form of the HJB PDE:

$$\begin{cases} V_t + \dfrac{1}{2}\mathrm{tr}\left(V_{xx}\Sigma\Sigma^{\mathrm{T}}\right) - \dfrac{1}{2}V_x^{\mathrm{T}}\left(GR^{-1}G^{\mathrm{T}} - \dfrac{1}{\varepsilon}\Sigma\Sigma^{\mathrm{T}}\right)V_x \\ + V_x^{\mathrm{T}}f + q = 0, \quad (t,x)\in[\tau,T)\times\mathbb{R}^n, \\ V(x,T) = g(x), \quad x\in\mathbb{R}^n. \end{cases} \tag{11}$$

Note that the above PDE is a special case of the HJI PDE (6) when $L = \Sigma$ and $R_v = \varepsilon I$ (Fig. 1). Intuitively, this means that min-max control collapses to risk sensitive control when the adversarial control comes in the same channels as noise, and the control authority of this adversary is proportional to the risk sensitivity. In this paper, we focus on the special case of risk sensitive min-max control, although the framework is applicable to any general min-max control problem.

C. FBSDE Reformulation

We now reformulate the min-max control PDE (6) in the risk sensitive case to a set of FBSDEs. Here we restate the nonlinear Feynman-Kac lemma from [13] for convenience of the reader. For the derivation we refer the reader to [24, Proposition 4.3]:

Theorem 1 (Nonlinear Feynman-Kac)**.** *Consider the following Cauchy problem:*

$$\begin{cases} V_t + \dfrac{1}{2}\mathrm{tr}(V_{xx}\Sigma\Sigma^{\mathrm{T}}) + V_x^{\mathrm{T}}b + h = 0, \quad (t,x)\in[\tau,T)\times\mathbb{R}^n, \\ V(T,x) = g(x), \quad x\in\mathbb{R}^n, \end{cases} \tag{12}$$

wherein the functions Σ, $b(t,x)$, $h(t,x,V,\Sigma^{\mathrm{T}}V_x)$, and $g(x)$ satisfy mild regularity conditions. Then (12) *admits a unique viscosity solution $V:[\tau,T]\times\mathbb{R}^n\to\mathbb{R}$, which has the following probabilistic representation:*

$$\left(V(t,x), \Sigma^{\mathrm{T}}V_x(t,x)\right) = \left(y(t,x), z(t,x)\right), \ \forall (t,x)\in[\tau,T]\times\mathbb{R}^n, \tag{13}$$

wherein $\left(x(\cdot),y(\cdot),z(\cdot)\right)$ is the unique adapted solution of the FBSDEs given by:

$$\begin{cases} \mathrm{d}x(t) = b(x(t),t)\mathrm{d}t + \Sigma(x(t),t)\mathrm{d}w(t), \quad t\in[\tau,T] \\ x(\tau) = \xi \end{cases} \tag{14}$$

and

$$\begin{cases} \mathrm{d}y(t) = -h(t,x(t),y(t),z(t))\mathrm{d}t + z(t)^{\mathrm{T}}\mathrm{d}w(t), \quad t\in[\tau,T] \\ y(T) = g(x(T)) \end{cases} \tag{15}$$

In order to apply the Nonlinear Feynman-Kac theorem to (6), we assume that there exist matrix-valued functions $\Gamma_u:[\tau,T]\times\mathbb{R}^n\to\mathbb{R}^{m\times p}$ and $\Gamma_v:[\tau,T]\times\mathbb{R}^n\to\mathbb{R}^{m\times q}$ such that $G(x(t),t) = \Sigma(x(t),t)\Gamma_u(x(t),t)$ and $L(x(t),t) = \Sigma(x(t),t)\Gamma_v(x(t),t)$ for all $(t,x)\in[\tau,T]\times\mathbb{R}^n$, satisfying the same regularity conditions. This assumption suggests that there cannot be a channel containing control input but no noise. In the risk sensitive case of min-max control, this assumption is already satisfied with $L(x(t),t) = \Sigma(x(t),t)$ and $\Gamma_v(x(t),t) = I$, where I is a $m\times m$ identity matrix because adversarial control enters through the noise channels. Under this assumption, Theorem 1 can be applied to the risk sensitive case of HJI equation (6) with

$$\begin{aligned} b(x(t),t) &= f(x(t),t) \\ h(x(t),y(t),z(t),t) &= q(x(t)) \\ &\quad - \frac{1}{2}V_x^{\mathrm{T}}\left(\Sigma\Gamma_u R^{-1}\Gamma_u^{\mathrm{T}}\Sigma^{\mathrm{T}} - \frac{1}{\varepsilon}\Sigma\Sigma^{\mathrm{T}}\right)V_x. \end{aligned} \tag{16}$$

The relationship between FBSDE (14), (15), HJI PDE (6), HJB PDE (11), and the parabolic PDE (12) is summarized in Fig. 1.

D. Importance Sampling

The system of FBSDEs in (14) and (15) corresponds to a system whose dynamics are uncontrolled. In many cases, especially for unstable systems, it is hard or impossible to reach the target state with uncontrolled dynamics. We can address this problem by modifying the drift term in the dynamics (forward SDE) with an additional control term. Through Girsanov's theorem (see [25, Chapter 3], [26, Chapter 5]) of change of measure, the drift term in the forward SDE (14) and backward SDE (15) can be changed such that the process $\tilde{w}(t) = w(t) + \int_\tau^t K(s)\,ds$, is a brownian motion under the new probability measure. This procedure is called importance sampling as we end up sampling from a different distribution of trajectories due to a modified drift term instead of the original distribution of uncontrolled trajectories. This results in a new FBSDE system given by

$$\begin{cases} \mathrm{d}\tilde{x}(t) &= [b(\tilde{x}(t),t) + \Sigma(\tilde{x},t)K(t)]\mathrm{d}t \\ &\quad + \Sigma(\tilde{x}(t),t)\mathrm{d}w(t), \quad t\in[\tau,T] \\ \tilde{x}(\tau) &= \xi \end{cases} \tag{17}$$

and

$$\begin{cases} \mathrm{d}\tilde{y}(t) &= [-h(t,\tilde{x}(t),\tilde{y}(t),\tilde{z}(t)) + \tilde{z}^{\mathrm{T}}K(t)]\mathrm{d}t \\ &\quad + \tilde{z}(t)^{\mathrm{T}}\mathrm{d}w(t), \quad t\in[\tau,T] \\ \tilde{y}(T) &= g(\tilde{x}(T)) \end{cases}, \tag{18}$$

for any measurable, bounded and adapted process $K:[\tau,T]\to\mathbb{R}^n$. It is easy to verify that the PDE associated with the new system is the same as the original one (12). For the full derivation of change of measure for FBSDEs, we refer

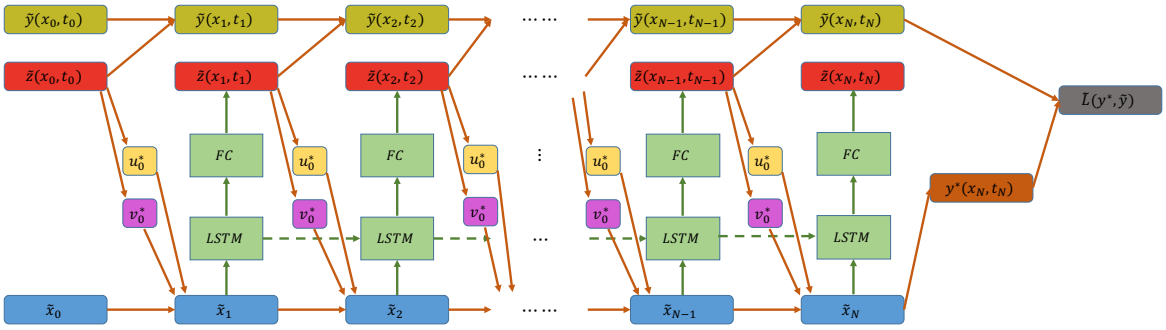

Fig. 2: Deep min-max FBSDE network architecture. The LSTM block represents multi-layered long-short term memory layers with non-linear activations, and the FC block represents one fully connected output layer with linear activations. The red arrows are the dynamics propagation, and the green arrows represent forward pass through the neural network. The same LSTM and FC layers are used in all time steps with shared weights. The figure above is therefore an unrolled representation of the recurrent neural network.

readers to proof of Theorem 1 in [14]. We can conveniently set $K = \Gamma_u(\tilde{x}(t),t)\bar{u} + \Gamma_v(\tilde{x}(t),t)\bar{v}$ for min-max control. Note that the nominal controls $\bar{u}$ and $\bar{v}$ can be any open or closed loop control or control from a previous iteration.

E. Forward Sampling of BSDE

The compensated BSDE (18) needs to satisfy a terminal condition, meaning its solution needs to be propagated backward in time; however, as it is a stochastic process for which noise is also integrated, a simple backward integration of the process would imply that at each point in time, its value depends on particular future values of the noise, violating the assumption that future noise values are always unknown and random. This poses a challenge for sampling based methods to solve the system of FBSDEs. One solution is to approximate the conditional probability of the process and back-propagate the expected value. This approach lacks scalability due to inevitable compounding of approximation errors that are accumulated at every time step during regression.

This problem can be alleviated with DL. Using a deep recurrent network, we can initialize the value function and its gradient at $t = \tau$ and treat the initializations as trainable network parameters. This allows for the BSDE to be propagated forward in time along with the FSDE. At the final time, the terminal condition can be compared against the propagated value in the loss function to update the initialzations as well as the network parameters. Compared to the conditional probability approximation scheme, the DL approach has the additional advantage of not accumulating errors at every time step since the recurrent network at each time step contributes to a common goal of predicting the target terminal condition and thus prediction errors are jointly minimized.

III. Deep Min-max FBSDE Controller

With eqs. (17) and (18), we have a system of FBSDEs that we can sample from around a nominal control trajectory. Inspired by the network architecture developed in [19], we propose a deep min-max FBSDE algorithm that solves the risk sensitive formulation of the min-max control.

A. Numerics and Network Architecture

The time horizon $\tau < t < T$ can be discretized as $n = \{0,1,\cdots,N\}$ with a time discretization of $\Delta t = (T-\tau)/N$. With this we can approximate the continuous variables as step functions and obtain their discretization as $\tilde{x}_n, \tilde{y}_n, \tilde{z}_n, u_n = \tilde{x}(t), \tilde{y}(t), \tilde{z}(t), u(t)$ if $\tau + n\Delta t \leq t < \tau + (n+1)\Delta t$. Both the dynamics and the value function are propagated forward using the Euler-Maruyama scheme.

The network architecture used in this paper is shown in Fig. 2, which is based on the LSTM network in [19] with min-max objective and value function dynamics incorporated. LSTM is a natural choice of network here since it is designed to effectively deal with the vanishing gradient problem in recurrent prediction of long time series [20]. We use two LSTM layers and one fully connected layer in the network with tanh activation and Xavier initialization [27]. At every time step, the network predicts the value function gradient using the current state as input. The optimal minimizing and adversarial control are then calculated with

$$u_n^* = -R_u^{-1}\Gamma_{u,n}\tilde{z}_n \tag{19}$$

$$v_n^* = \frac{1}{\varepsilon}\tilde{z}_n \tag{20}$$

and fed back to the dynamics for importance sampling. Note that the adversarial control is only present during training. After the network is trained, only the optimal minimizing control is used at test time. By exposing the minimizing controller to an adversary that behaves in an optimal fashion, it becomes more robust resulting in trajectories with smaller variances.

B. Algorithm

The Deep Min-max FBSDE algorithm can be found in Algorithm 1. It solves a finite time horizon control problem by approximating the gradient of the value function $\tilde{z}_n^i$ (the superscript i denotes the batch index, and the batch-wise computation can be done in parallel) at every time step with a LSTM, which is parameterized by θ, and propagating the

Algorithm 1: Deep Min-max FBSDE Controller

Given:

$\tilde{x}_0 = \xi$, f, G, Σ: Initial state and system dynamics;
g, q, R_u, $\varepsilon(= R_v)$: Cost function parameters;
N: Task horizon; K: Number of iterations;
M: Batch size; Δt: Time discretization;
λ: weight-decay parameter;
γ: Loss function parameter;

Parameters:

$\tilde{y}_0 = V(\tilde{x}_0, \tau; \psi)$: Value function at $t = \tau$;
$\tilde{z}_0 = \Sigma^{\mathrm{T}} \nabla_{\tilde{x}} V$: Gradient of value function at $t = \tau$;
θ: Weights and biases of all neural network layers;

Initialize:

$\{\tilde{x}_0^i\}_{i=1}^M$, $\tilde{x}_0^i = \xi$
$\{\tilde{y}_0^i\}_{i=1}^M$, $\tilde{y}_0^i = V(\tilde{x}_0^i, 0; \psi)$
$\{\tilde{z}_0^i\}_{i=1}^M$, $\tilde{z}_0^i = \Sigma^{\mathrm{T}} \nabla_{\tilde{x}} V(\tilde{x}_0^i, 0; \psi)$

for $k = 1$ **to** K **do**
  for $i = 1$ **to** M **do**
    for $n = 1$ **to** $N-1$ **do**
      Compute gamma matrix: $\Gamma_{u,n}^i = \Gamma_u(\tilde{x}_n^i)$;
      $u_n^{i*} = -R_u^{-1} \Gamma_{u,n}^{i\mathrm{T}} \tilde{z}_n^i$;
      $v_n^{i*} = \frac{1}{\varepsilon} \tilde{z}_n^i$;
      Sample Brownian noise: $\Delta w_n^i \sim \mathcal{N}(0, 1)$
      Update value function:
      $\tilde{y}_{n+1}^i = \tilde{y}_n^i - \left(\tilde{h}(\tilde{x}_n^i, \tilde{y}_n^i, \tilde{z}_n^i) + \tilde{z}_n^i (\Gamma_{u,n}^i u_n^{i*} + v_n^{i*})\right)\Delta t$
      $+ \tilde{z}_n^{i\mathrm{T}} \Delta w_n^i \sqrt{\Delta t}$
      Update system state:
      $\tilde{x}_{n+1}^i =$
      $\tilde{x}_n^i + f(\tilde{x}_n^i)\Delta t + \Sigma\left((\Gamma_{u,n}^i u_n^{i*} + v_n^{i*})\Delta t + \Delta w_n^i \sqrt{\Delta t}\right)$
      Predict gradient of value function:
      $\tilde{z}_{n+1}^i = f_{LSTM}(\tilde{x}_{n+1}^i; \theta_k)$
    end for
    Compute target terminal value: $y_N^{*i} = g(\tilde{x}_N^i)$
  end for
  Compute mini-batch loss:
  $\mathscr{L} = \frac{1}{M} \sum_{i=1}^{M} \left(\gamma \|y_N^{*i} - \tilde{y}_N^i\|^2 + (1-\gamma)\|y_N^{*i}\|^2\right) + \lambda \|\theta_k^2\|^2$
  $\theta_{k+1} \leftarrow$ Adam.step$(\mathscr{L}, \theta_k)$; $\psi_{k+1} \leftarrow$ Adam.step$(\mathscr{L}, \psi_k)$
end for
return θ_K, ψ_K

FBSDE associated with the control problem. For a given initial state condition ξ, the algorithm randomly initializes the value function and its gradient at $n = 0$. The initial values are trainable and are parameterized by ψ. At every time step of each training iteration, control inputs are sampled around the calculated optimal minimizing and adversarial controls (19)(20) and applied to the system. Both SDEs (17)(18) are then forward propagated to the next time step. At the final time step $n = N$, a modified L^2 loss with regularization is computed which compares the propagated value function $\tilde{y}_N^i$ against the true value function y_N^{*i} calculated using the final state ($y_N^{*i} = g(\tilde{x}_N^i)$). For training our network, we propose a

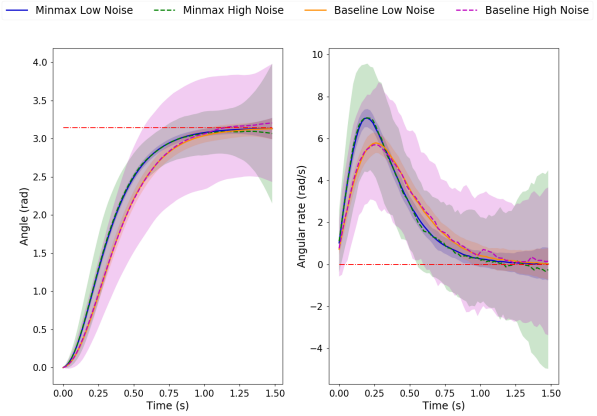

Fig. 3: Pendulum states. *Left*: Pendulum Angle; *Right*: Pendulum Angular Rate.

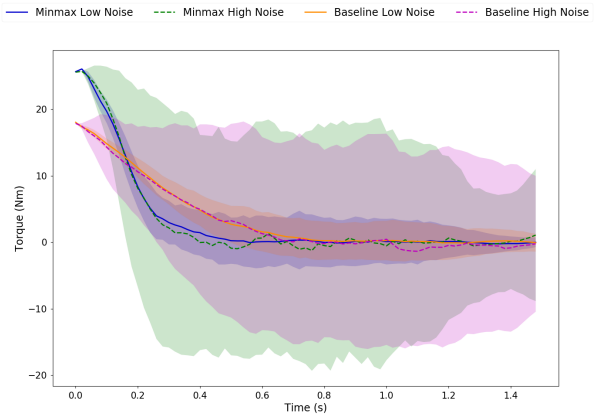

Fig. 4: Pendulum controls.

new regularized loss function, which is a convex combination of a) the difference between the target and the predicted value function, and b) the target value function itself:

$$\mathscr{L} = \frac{1}{M} \sum_{i=1}^{M} \left(\beta \|y_N^{*i} - \tilde{y}_N^i\|^2 + (1-\beta)\|y_N^{*i}\|^2\right) + \mathrm{Reg}(\theta_k), \quad (21)$$

since we want the prediction to be close to the target and at the same time, the target value function to converge to zero for the sake of the optimality. Notice that this additional component in the loss function is possible only due to importance sampling. The modified drift is implemented as a connection in the computational graph between the LSTM output and input to forward SDE at the next timestep. This allows the network parameters to influence the next state and hence the final state. The network can be trained by Stochastic Gradient Descent (SGD) type optimizer and in our experiments, we used the Adam [28] optimizer.

IV. Experiments

The algorithm is implemented on a pendulum and quadcopter system in simulation. The control objective for the two systems is to reach a target state in finite time. The trained networks are tested on 128 trajectories. The time

TABLE I: Comparison of total state variance between deep min-max controller and baseline deep FBSDE controller.

	Pendulum		QuadCopter	
	Low Noise	High Noise	Low Noise	High Noise
Baseline	5.3	149.5	3.2	78.6
RS	3.9	134.2	2.7	69.6
Variance Reduction (%)	26	10	16	11

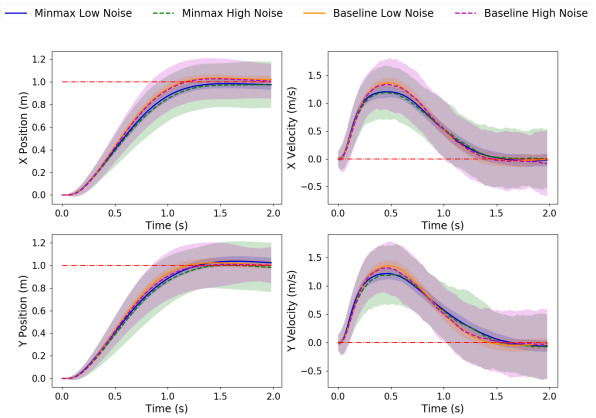

Fig. 5: Quadcopter states. *Top Left*: X Position; *Top Right*: X Velocity; *Bottom Left*: Y Position; *Bottom Right*: Y Velocity.

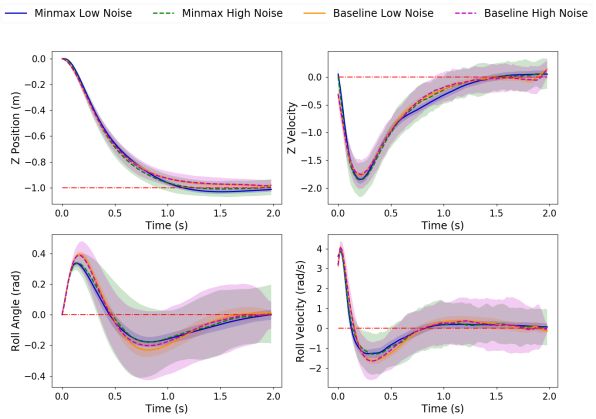

Fig. 6: Quadcopter states. *Top Left*: Z Position; *Top Right*: Z Velocity; *Bottom Left*: Roll Angle; *Bottom Right*: Roll Velocity.

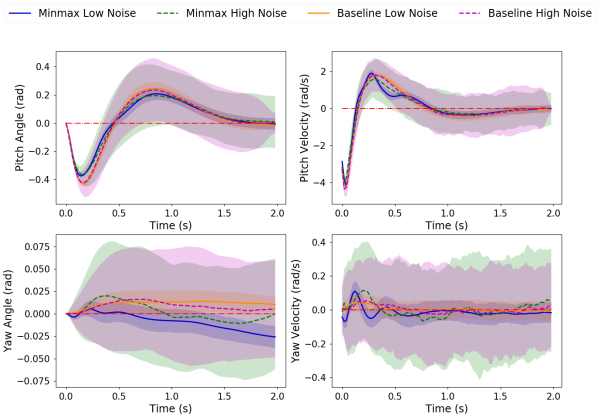

Fig. 7: Quadcopter states. *Top Left*: Pitch Angle; *Top Right*: Pitch Velocity; *Bottom Left*: Yaw Angle; *Bottom Right*: Yaw Velocity.

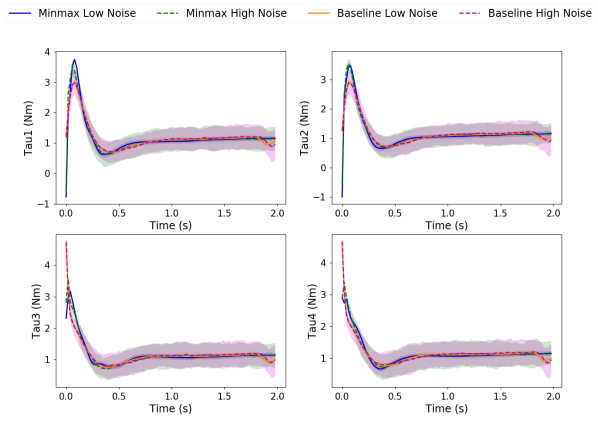

Fig. 8: Quadcopter controls.

discretization is 0.02 seconds across all cases. We compare the algorithm proposed in this paper with the one in [19], where the standard optimal control problem is considered, in two different noise conditions. We will use "Minmax" to denote the algorithm in this work and "Baseline" for the algorithm that we are comparing against. All experiments were done in TensorFlow [29] on an Intel i7-4820k CPU Processor.

In all trajectory plots, the solid line denotes the mean trajectory in low noise condition, the dashed line denotes the mean trajectory in high noise condition, and the red dashed line denotes the target state. In addition, the 4 conditions are denoted by different colors, with blue for Minmax in low noise condition, green for Minmax in high noise condition, orange for Baseline in low noise condition, and magenta for Baseline with high noise. The shaded region of each color denotes the 95% confidence region.

A. Pendulum

For the pendulum system, the algorithm was implemented to complete a swing-up with a time horizon of 1.5 seconds. The two system states are the pendulum angle $[rad]$ and the pendulum angular rate $[rad/s]$. Fig. 3 plots the pendulum states in all 4 cases (Minmax with low and high noise and Baseline with low and high noise). The control applied to the system is the torque $[N \cdot m]$ (Fig. 4).

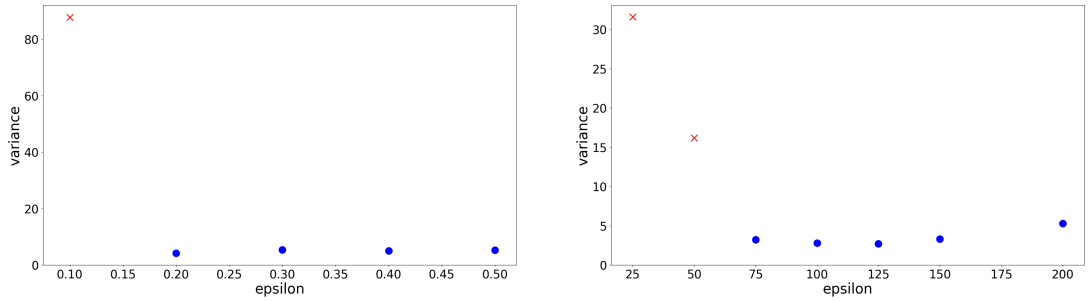

Fig. 9: Total state variance vs. ε for both systems. *Left: Pendulum; Right: QuadCopter.*

B. Quadcopter

The algorithm was implemented on a quadcopter system to reach a final target state from an initial position with a time horizon of 2 seconds. The initial condition is 0 across all states. The target is 1 $[m]$ upward, forward and to the right from the initial position with zero velocities and attitudes. The quadcopter dynamics used can be found in [30]. The 12 system states are composed of the position $[m]$, angles $[rad]$, linear velocities $[m/s]$, and angular velocities $[rad/s]$. The control inputs to the system are 4 torques $[N \cdot m]$, which control the rotors (Fig. 8).

C. Reduced Variance with Deep Min-max FBSDE Controller

The trajectory plots (Fig. 3, 5, 6, and 7) compare the Deep Min-max FBSDE controller against the risk neutral Deep FBSDE controller in a low noise setting and a high noise setting for both systems. From the plots we can observe that the min-max controller proposed in this work accomplishes the tasks with similar level of performance compared to the baseline controller. Numerical comparisons of the total state variance (sum of variance in all states over the entire trajectory) of all test cases can be found in Table I. The results demonstrate at least 10% reduction in total state variance across all cases. It is worth noting that the high noise setting results in less variance reduction benefits. By examining the substitution of $\Sigma = \sqrt{\frac{\varepsilon}{2\gamma^2}}\tilde{\Sigma}$ from (10) to (11) in risk sensitive control derivation, we can see that increasing noise level is in some sense equivalent to increasing ε. This naturally reduces the effect of the risk sensitive controller, as shown in the next section.

D. Variance vs. Risk Sensitivity

We also investigated the relationship between total state variance and risk sensitivity (adversarial control cost) in the two systems. Fig. 9 plots the total state variance for different risk sensitivity ε (R_v) values while also keeping track of whether the control objective is met. In the variance versus ε scatter plots, blue circles are used to denote runs where the control objective is met, whereas red cross denotes runs where the controller fails to meet the control objective. Since the risk sensitivity parameter ε is inversely proportional to the adversarial control authority, we expect the risk sensitive min-max controller to behave like the standard optimal controller as ε increases to infinity. On the other hand, as ε gets smaller, the adversarial control will eventually dominate the minimizing control and cause controller failure. This is reflected in the plots as we can observe that the minimizing controller starts to fail when ε is too low. It is worth noting that the failure threshold increases as the system gets more complex and higher dimensional.

V. Conclusions

In this paper, we proposed the Deep Min-max FBSDE Control algorithm, based on the risk sensitive case of stochastic game-theoretic optimal control theory. Utilizing prior work on importance sampling of FBSDEs and efficiency of the LSTM network to predict long time series, the algorithm is capable of solving stochastic game-theoretic control problems for nonlinear systems with control-affine dynamics. Comparison of this algorithm against the standard stochastic optimal control formulation suggests that by considering an adversarial control in the form of noise-related risk, the controller outputs trajectories with lower variance. Our algorithm scales in terms of the number of system states and system complexity for the min-max control problem, while the previous works did not. For future works, we would like to explore different network architectures to reduce the training time.

Acknowledgments

This research was supported by the NSF CMMI award #1662523 and Amazon AWS.

References

[1] R. Isaacs. *Differential Games: A Mathematical Theory with Applications to Warfare and Pursuit, Control and Optimization*. New York: Willey, 1965.

[2] H. Kushner. Numerical approximations for stochastic differential games. *SIAM J. Control Optim.*, 41:457–486, 2002.

[3] J. Morimoto, G. Zeglin, and C. Atkeson. Minimax differential dynamic programming: Application to a biped walking robot. *IEEE/RSJ International Conference on Intelligent Robots and Systems, Las Vegas, NV*, pages 1927–1932, October 27-31, 2003.

[4] J. Morimoto and C. Atkeson. Minimax differential dynamic programming: An application to robust biped walking. *Advances in Neural Information Processing Systems (NIPS), Vancouver, British Columbia, Canada*, December 9-14, 2002.

[5] W. Sun, E. A. Theodorou, and P. Tsiotras. Game-theoretic continuous time differential dynamic programming. *American Control Conference, Chicago, IL*, pages 5593–5598, July 1–3, 2015.

[6] H. J. Kappen. Path integrals and symmetry breaking for optimal control theory. *Journal of Statistical Mechanics: Theory and Experiment*, 11:P11011, 2005.

[7] W.H. Fleming. Exit probabilities and optimal stochastic control. *Applied Math. Optim*, 9:329–346, 1971.

[8] W. H. Fleming and H. M. Soner. *Controlled Markov processes and viscosity solutions*. Applications of mathematics. Springer, New York, 2nd edition, 2006.

[9] I. Karatzas and S. E. Shreve. *Brownian Motion and Stochastic Calculus (Graduate Texts in Mathematics)*. Springer, 2nd edition, August 1991.

[10] J. Yong and X.Y. Zhou. *Stochastic Controls: Hamiltonian Systems and HJB Equations*. Stochastic Modelling and Applied Probability. Springer New York, 1999.

[11] Etienne Pardoux and Aurel Rascanu. *Stochastic Differential Equations, Backward SDEs, Partial Differential Equations*, volume 69. 07 2014.

[12] I. Exarchos. *Stochastic Optimal Control-A Forward and Backward Sampling Approach*. PhD thesis, Georgia Institute of Technology, 2017.

[13] I. Exarchos and E. A. Theodorou. Learning optimal control via forward and backward stochastic differential equations. In *American Control Conference (ACC), 2016*, pages 2155–2161. IEEE, 2016.

[14] I. Exarchos and E. A. Theodorou. Stochastic optimal control via forward and backward stochastic differential equations and importance sampling. *Automatica*, 87:159–165, 2018.

[15] I. Exarchos, E. A. Theodorou, and P. Tsiotras. Stochastic L^1-optimal control via forward and backward sampling. *Systems & Control Letters*, 118:101–108, 2018.

[16] Ioannis Exarchos, Evangelos A Theodorou, and Panagiotis Tsiotras. Game-theoretic and risk-sensitive stochastic optimal control via forward and backward stochastic differential equations. In *2016 IEEE 55th Conference on Decision and Control (CDC)*, pages 6154–6160. IEEE, 2016.

[17] I. Exarchos, E. A. Theodorou, and P. Tsiotras. Stochastic Differential Games – A Sampling Approach via FBSDEs. *Dynamic Games and Applications*, pages 1–20, 2018.

[18] Jiequn Han et al. Deep Learning Approximation for Stochastic Control Problems. *arXiv preprint arXiv:1611.07422*, 2016.

[19] Marcus A Pereira, Ziyi Wang, Ioannis Exarchos, and Evangelos A Theodorou. Learning deep stochastic optimal control policies using forward-backward sdes. In *Robotics: science and systems*, 2019.

[20] Sepp Hochreiter and Jürgen Schmidhuber. LSTM can solve hard long time lag problems. In *Advances in Neural Information Processing Systems*, pages 473–479, 1997.

[21] Somil Bansal, Mo Chen, Sylvia Herbert, and Claire J Tomlin. Hamilton-jacobi reachability: A brief overview and recent advances. In *2017 IEEE 56th Annual Conference on Decision and Control (CDC)*, pages 2242–2253. IEEE, 2017.

[22] T. Basar and P. Berhard. *H-infinity Optimal Control and Related Minimax Design*. Birkhauser, Boston, 1995.

[23] Wendell H. Fleming and William M. McEneaney. Risk-sensitive control on an infinite time horizon. *SIAM Journal of Control Optimization*, 33(6):1881–1915., 1995.

[24] Nicole El Karoui, Shige Peng, and Marie Claire Quenez. Backward stochastic differential equations in finance. *Mathematical finance*, 7(1):1–71, 1997.

[25] Ioannis Karatzas and Steven E Shreve. Brownian motion. In *Brownian Motion and Stochastic Calculus*, pages 47–127. Springer, 1998.

[26] Steven E Shreve. *Stochastic calculus for finance II: Continuous-time models*, volume 11. Springer Science & Business Media, 2004.

[27] Xavier Glorot and Yoshua Bengio. Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, pages 249–256, 2010.

[28] Diederik P. Kingma and Jimmy Ba. Adam: A Method for Stochastic Optimization. *Proceedings of the 3rd International Conference on Learning Representations (ICLR)*, abs/1412.6980, 2014.

[29] Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dan Mané, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. Software available from tensorflow.org.

[30] Maki K Habib, Wahied Gharieb Ali Abdelaal, Mohamed Shawky Saad, et al. Dynamic modeling and control of a quadrotor using linear and nonlinear approaches. Master's thesis, The American University in Cairo, 2014.