Deep Learning for solving Hamilton-Jacobi-Bellman equations in finance

Shuaiqiang Liu and Kees Vuik

Introduction:

Partial differential equations (PDE), in particular Hamilton-Jacobi-Bellman (HJB) equations, are widely used for stochastic optimal control in finance, for example, hedging, pension funds management and high frequency trading. The solution of HJB gives optimality of a control with respect to an objective function. Their analytical solution is often unavailable, and numerical methods are required to solve such PDEs. However, classical numerical methods (e.g., finite difference/element method) become inefficient when solving high dimensional HJBs due to the curse of dimensionality, that is, the grid-based PDE discretization methods become exponentially expensive with the increasing dimension. Think of hedging a basket option in finance, where multiple risky underlying assets are involved. Alternatively, HJB PDEs can be transformed to their associated stochastic differential equations (SDE) by Feynman-Kac theorem, and Monte-Carlo simulation-based algorithms [3] can be used in the case of multiple dimensional problems (mainly for medium-size), which may become time-consuming in very high dimensions.

Recently deep learning methods have made tremendous achievements as an advanced numerical technique to solve high dimensional PDEs, for instance, Physics-informed neural networks to solve PDEs [4], deep backward dynamic programming [2], deep backward stochastic differential equation [1]. The latter two methods rely on the following two facts: the connection between PDE and associated SDE, a neural network as a function approximator. There are still some challenging problems, for example, how to deal with HJBs in the case of stochastic control with jump-diffusion models (e.g., stock price jumps [5]), best practice to set up deep learning for HJBs (e.g., fully connected deep neural networks are found unstable when solving high dimensional PDEs in [1]).

This Msc thesis will be focused on developing deep learning-based numerical techniques to solve HJB equations and its applications in finance.

Objectives:

1. Derive HJB equations for stochastic optimal control problems (dynamic programming, Ito lemma, etc).
2. Study deep learning algorithms (neural networks, stochastic gradient descent, etc)
3. Review/implement recent developments in deep learning for solving HJB, e.g., deep backward dynamic programming, deep backward stochastic differential equation, by applying Feynman-Kac theorem and deep neural networks, etc.
4. Develop neural networks-based numerical methods for high dimensional HJB for stochastic optimal control without or with jumps (numerical analysis, program codes, best practice).
5. Perform numerical experiments for some applications in finance.

References:

[1] J. Han, A. Jentzen, and W. E. Solving high-dimensional partial differential equations using deep learning. PNAS, 2018

[2] C. Hure, H. Pham, and X. Warin. Deep backward schemes for high-dimensional nonlinear

PDEs. Mathematics of Computation, 2020.
[3] F. Cong and C.W. Oosterlee. Multi-period mean–variance portfolio optimization based on Monte-Carlo simulation. Journal of Economic Dynamics and Control, 2018.
[4] M. Raissia, P. Perdikaris and G.E. Karniadakis. Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. Journal of Computational Physics, 2019.
[5] R.C. Merton. Option pricing when underlying stock returns are discontinuous. Journal of Financial Economics, 1976.

Milestone:

6. Derive HJB equations for stochastic optimal control problems (dynamic programming, Ito lemma, etc).
7. Study deep learning algorithms (neural networks, stochastic gradient descent)
8. Review recent developments in deep learning-based algorithms for solving HJB, e.g., deep backward dynamic programming, deep backward stochastic differential equation, based on Feynman-Kac theorem, deep neural networks, etc.
9. Develop numerical methods for high dimensional HJB (stochastic optimal control without or with jumps, numerical analysis, program codes)
10. Perform implementation for applications in finance.