Real-world Derivatives Hedging with Deep Reinforcement Learning

Ashwin Rao

ICME, Stanford University

November 15, 2019

Classical Pricing and Hedging of Derivatives

- Classical Pricing/Hedging Theory assumes frictionless markets
- Frictionless := continuous trading, any volume, no transaction costs
- Assumptions of arbitrage-free and completeness lead to (dynamic and exact) replication of derivatives with *basic* market securities
- Replication strategy gives us the pricing and hedging solutions
- This is the foundation of the famous Black-Scholes formulas
- However, the real-world has many frictions $\Rightarrow$ *Incomplete Market*
- ... where Derivatives cannot be exactly replicated

Pricing and Hedging in an Incomplete Market

- In an incomplete market, we have multiple risk-neutral measures
- So, multiple derivative prices (each consistent with no-arbitrage)
- The market/trader "chooses" a risk-neutral measure (hence, price)
- Based on a specified preference for trading risk versus return
- This preference is equivalent to specifying a Utility function
- Maximizing "risk-adjusted return" of the derivative plus hedges
- Reminiscent of the Portfolio Optimization problem
- Likewise, we can set this up as a stochastic control (MDP) problem
- Where the decision at each time step is: *Trades in the hedges*
- So what's the best way to solve this MDP?

Deep Reinforcement Learning (DRL)

- Dynamic Programming not suitable in practice due to:
  - Curse of Dimensionality
  - Curse of Modeling
- So we solve the MDP with *Deep Reinforcement Learning* (DRL)
- The idea is to use real market data and real market frictions
- Developing realistic simulations to derive the optimal policy
- The optimal policy gives us the (practical) hedging strategy
- The optimal value function gives us the price (valuation)
- Formulation based on Deep Hedging paper by J.P.Morgan researchers
- More details in the prior paper by some of the same authors

Problem Setup

- We will simplify the problem setup a bit for ease of exposition
- This model works for more complex, more frictionful markets too
- Assume time is in discrete (finite) steps $t = 0, 1, \ldots, T$
- Assume we have a position (portfolio) D in m derivatives
- Assume each of these m derivatives expires in time $\leq T$
- Portfolio-aggregated *Contingent Cashflows* at time t denoted $X_t \in \mathbb{R}$
- Assume we have n basic market securities as potential hedges
- Hedge positions (units held) at time t denoted $\alpha_t \in \mathbb{R}^n$
- Cashflows per unit of hedges held at time t denoted $Y_t \in \mathbb{R}^n$
- Prices per unit of hedges at time t denoted $P_t \in \mathbb{R}^n$
- PnL position at time t is denoted as $\beta_t \in \mathbb{R}$

States and Actions

- Denote state space at time t as $\mathcal{S}_t$, state at time t as $s_t \in \mathcal{S}_t$
- Among other things, s_t contains $t, \alpha_t, P_t, \beta_t, D$
- s_t will include any market information relevant to trading actions
- For simplicity, we assume s_t is just the tuple $(t, \alpha_t, P_t, \beta_t, D)$
- Denote action space at time t as $\mathcal{A}_t$, action at time t as $a_t \in \mathcal{A}_t$
- a_t represents units of hedges traded (positive for buy, negative for sell)
- Trading restrictions (eg: no short-selling) define $\mathcal{A}_t$ as a function of s_t
- State transitions $P_{t+1}|P_t$ available from a *simulator*, whose internals are estimated from real market data and realistic assumptions

Sequence of events at each time step $t = 0, \ldots, T$

1. Observe state $s_t = (t, \alpha_t, P_t, \beta_t, D)$
2. Realize cashflows (from holding positions) $= X_t + \alpha_t \cdot Y_t$
3. Perform action (trades) a_t to produce trading PnL $= -a_t \cdot P_t$
4. Trading transaction costs, example $= -\gamma P_t \cdot |a_t|$ for some $\gamma > 0$
5. Update α_t as: $\alpha_{t+1} = \alpha_t + a_t$ for $t = 0, \ldots, T$
   (force-liquidation at termination means $a_T = \alpha_T$)
6. Update PnL β_t as:

$$\beta_{t+1} = \beta_t + X_t + \alpha_t \cdot Y_t - a_t \cdot P_t - \gamma P_t \cdot |a_t| \text{ for } t = 0, \ldots, T$$

7. Reward $r_t = 0$ for all $t = 0, \ldots, T-1$ and $r_T = U(\beta_{T+1})$ for an appropriate concave Utility function U (based on risk-aversion)
8. Simulator evolves hedge prices from P_t to P_{t+1}

Pricing and Hedging

- Assume we now want to enter into an incremental position (portfolio) D' in m' derivatives (denote the combined position as $D \cup D'$)
- We want to determine the *Price* of the incremental position D', as well as the hedging strategy for D'
- Denote the Optimal Value Function at time t as $V_t^* : \mathcal{S}_t \rightarrow \mathbb{R}$
- Pricing of D' is based on the principle that introducing the incremental position of D' together with a calibrated cashflow at $t = 0$ should leave the Optimal Value (at $t = 0$) unchanged
- Precisely, Price of D' is the value x such that

$$V_0^*((0, \alpha_0, P_0, \beta_0 - x, D \cup D')) = V_0^*((0, \alpha_0, P_0, \beta_0, D))$$

- The hedging strategy at time t is given by the Optimal Policy $\pi_t^* : \mathcal{S}_t \rightarrow \mathcal{A}_t$

DRL Approach a Breakthrough for Practical Trading?

- The industry practice/tradition has been to start with *Complete Market* assumption, and then layer ad-hoc/unsatisfactory adjustments
- There is some past work on pricing/hedging in incomplete markets
- But it's theoretical and not usable in real trading (eg: Superhedging)
- My view: This DRL approach is a breakthrough for practical trading
- Key advantages of this DRL approach:
  - Algorithm for pricing/hedging independent of market dynamics
  - Computational cost scales efficiently with size m of derivatives portfolio
  - Enables one to faithfully capture practical trading situations/constraints
  - Deep Neural Networks provide great function approximation for RL