MCTS for Trading and Hedging

November 2021

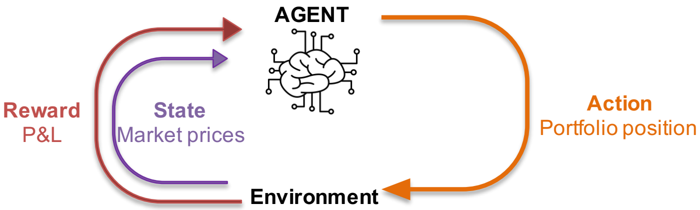
Edoardo Vittori, Amarildo Likmeta, Marcello Restelli

POLITECNICO
MILANO 1863

Outline

Introduction

Introduction - Trading and Hedging as MDPs

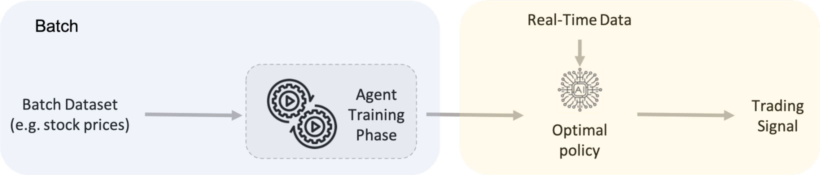


Trading	Option Hedging
• $a_t \in \{-1, 0, 1\}$: current portfolio	• $a_t \in [0, 1]$: current hedge portfolio
• $s_t = ([p_{t-w}, ..., p_t], a_{t-1})$	• $s_t = [S_t, C_t, \frac{\partial C_t}{\partial S_t}, a_{t-1}]$
• $r_t = a_t \cdot (p_t - p_{t-1}) - c(a_t - a_{t-1})$	• $r_t = C_t(S_t) - C_t(S_{t-1}) - a_t \cdot (S_t - S_{t-1}) - m \cdot \lvert a_t - a_{t-1} \rvert$

Batch RL vs MCTS

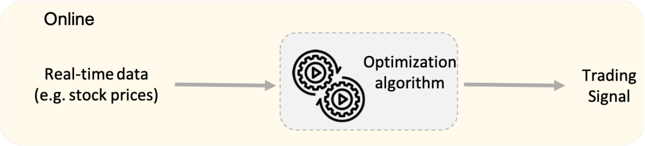
Batch RL

- General Policy
- Model Free
- Real Time Testing
- Stationary Policy

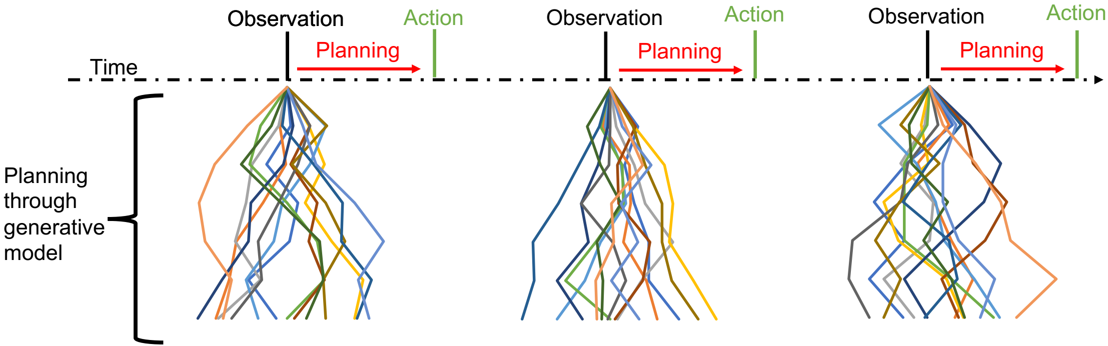


MCTS

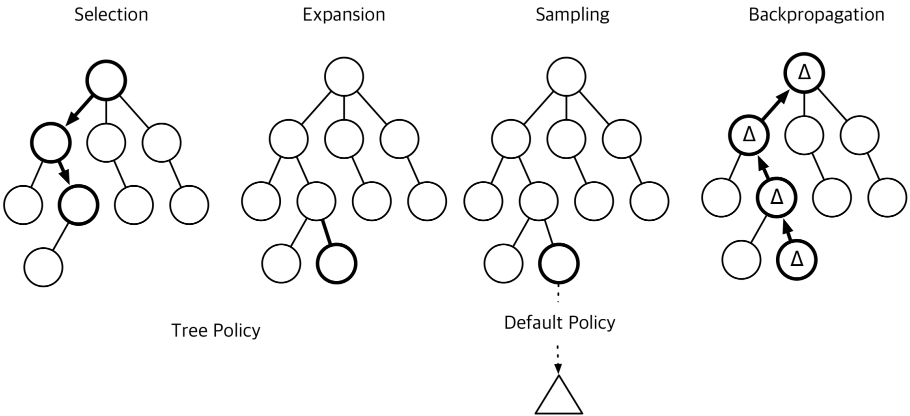
- Local Policy
- Model Based
- Some Delay
- Non Stationary Policy



Monte Carlo Tree Search

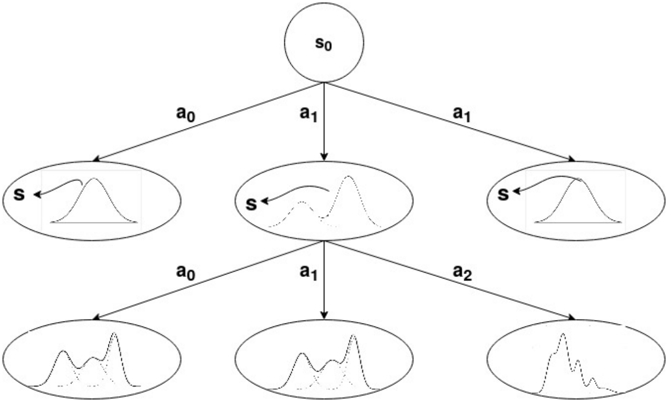


Upper Confidence Tree [Kocsis and Szepesvári, 2006]



- Selection using UCB$_1$ $a_n = \arg\max_{i=1..K} \overline{X}_{i,T_i(n-1)} + C\sqrt{\frac{2\log n}{T_i(n-1)}}$
- Necessary to tune the parameter C, larger C increases exploration
- Convergence to the optimal solution in the limit.

Open loop planning



Open-loop value of τ, starting from state s:

$$V_{OL}(s,\tau) = \mathbb{E}\left[\sum_{t=1}^{m} \gamma^t r_t \middle| s_0 = s, a_t \in \tau\right]$$

Open-loop value of a node $\mathcal{N}_{d,i}$:

$$\mathcal{V}\left(\mathcal{N}_{d,i}\right) = \mathop{\mathbb{E}}_{s \sim \mathcal{P}(\cdot|s_0, \tau_{d,i})} \left[V^*_{OL}(s)\right],$$

where
$V^*_{OL}(s) = \max_{\tau \in \mathcal{A}^m} V_{OL}(s,\tau)$

Progressive Widening

- continuous action spaces
- stochastic transition models
- convergence to optimal solution
- necessary to tune the PW parameter α, smaller values create more actions/states

Algorithm 1 Polynomial Upper Confidence Tree (PUCT) Extract [Couetoux, 2013]

if $\lfloor n(x)^{\alpha} \rfloor > \lfloor (n(x)-1)^{\alpha} \rfloor$ **then**
  $a \leftarrow s(x)$
  Children$(a) \leftarrow$Children$(a) \cup (\omega)$
else
  $a \leftarrow \arg\max_{a \in C(z)} \hat{V}(x,a) + \sqrt{\frac{n(x)^{e(d)}}{n(x,a)}}$
end if
if $\lfloor n(w)^{\alpha} \rfloor = \lfloor (n(w)-1)^{\alpha} \rfloor$ **then**
  select the child of ω least visited during the simulation
else
  $[x', r] \leftarrow M(x,a)$
  Children$(x,a) \leftarrow$ Children$(x,a) \cup (x')$
end if

TD Backpropagation

Standard Backpropation:

$$\mathcal{Q}_t\left(\mathcal{N}_{d,i}, a\right) = (1 - \frac{1}{N})\mathcal{Q}_t\left(\mathcal{N}_{d,i}, a\right) + \frac{1}{N}(r_t + \gamma \mathcal{V}_t\left(\mathcal{N}_{d+1,j}\right))$$

Temporal Difference Backpropagation, based on the Q-Learning update rule [Watkins, 1989], as follows:

$$\mathcal{Q}_t\left(\mathcal{N}_{d,i}, a\right) = (1 - \beta)\mathcal{Q}_t\left(\mathcal{N}_{d,i}, a\right) + \beta\left(r_t + \gamma \max_{a'} \mathcal{Q}_t\left(\mathcal{N}_{d+1,j}, a'\right)\right)$$

Necessary to tune learning parameter β

Our Approach: Open Loop Q-Learning UCT

Our algorithm consists of the following parts and parameters:

- UCT $\rightarrow$ parameter C
- Open loop for stochastic states
- Progressive widening for continuous actions $\rightarrow$ parameter α
- Q-Learning backpropagation $\rightarrow$ parameter β

For the search, it is necessary to choose:

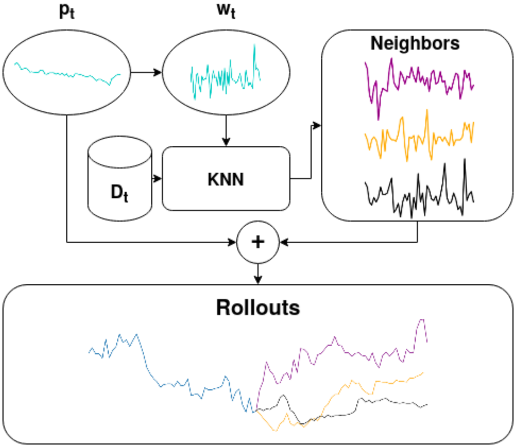
- Budget
- Search depth $\in [2, \text{full episode}]$
- Generative model (which can potentially have further parameters)

Lots of parameters to optimize in order to achieve the optimal solution!

Trading with MCTS

Clustering generative model

- Start from the current price window $w_t = (p_{t-M}, \dots, p_{t-1})$.
- Extract window of returns $\delta_t = \frac{p_t - p_{t-1}}{p_{t-1}}$, $\boldsymbol{\delta}_t = (\delta_{t-M}, \dots, \delta_{t-1})$.
- Find the K nearest neighbors of $\boldsymbol{\delta}_t$ in the historical dataset D.
- Use the neighbors to project future asset prices.



Trading with MCTS

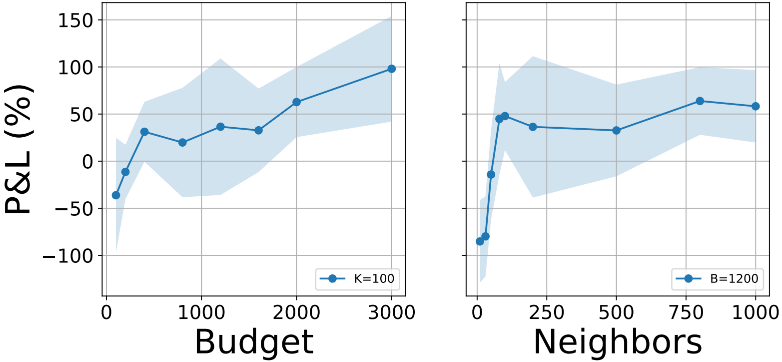
- a_t: current portfolio $\{-1, 0, 1\}$
- $s_t = ([p_{t-w}, ..., p_t], a_{t-1})$
- $r_t = a_t \cdot (p_t - p_{t-1}) - c(a_t - a_{t-1})$, where $c(a_t - a_{t-1}) = \frac{bid-ask}{2} \cdot |a_t - a_{t-1}|$

Algorithm 2 Trading with MCTS

Require: prices $p_{-n}, ..., p_0$, window size M, number of neighbors K

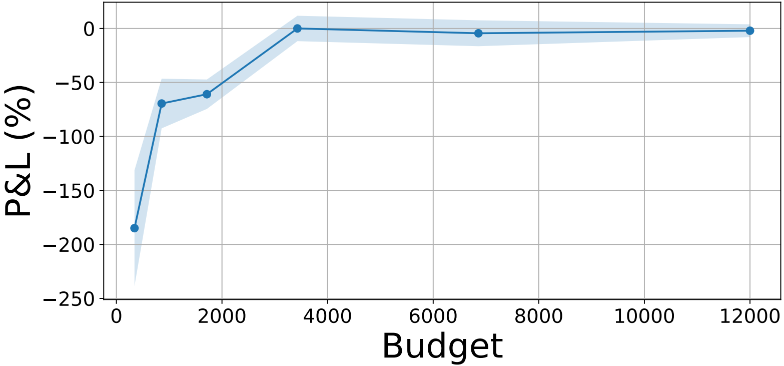
for $t \in \{1, \ldots, T\}$ **do**
    Find K MC trajectories from current return window $\delta_{t-M}, ..., \delta_{t-1}$
    Plan with QL-OL UCT, sampling the rollouts from the K neighbors.
    Select action $a_t = \max_a \mathcal{Q}\left(\mathcal{N}_{0,0}, a\right)$
    Observe p_t from the market
end for

Results of EURUSD FX data - no costs



Annualized average P&L with no transaction costs, as a function of the search budget and the numbers of neighbors. Average over 50 runs, 95% confidence intervals.

Results of EURUSD FX data with costs

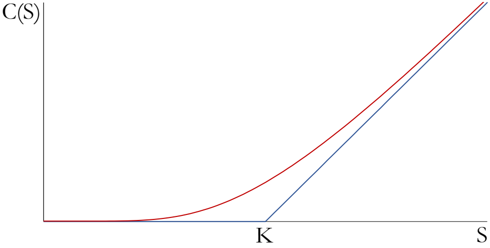


Annualized average P&L with transaction costs (10^{-5}) as a function of the search budget, $K = 100$. Average over 50 runs, 95% confidence intervals.

Option Hedging with MCTS

Option hedging

Vanilla options: contracts that offer the buyer the right to buy or sell a certain amount of the *underlying asset* at a predefined price at a certain future time.



Black & Scholes assumptions:

- continuous time
- continuous "lot size"
- no transaction costs

Black & Scholes pricing: trade the underlying continously so to match the option delta

$$\delta = \frac{\partial C(S)}{\partial S}.$$

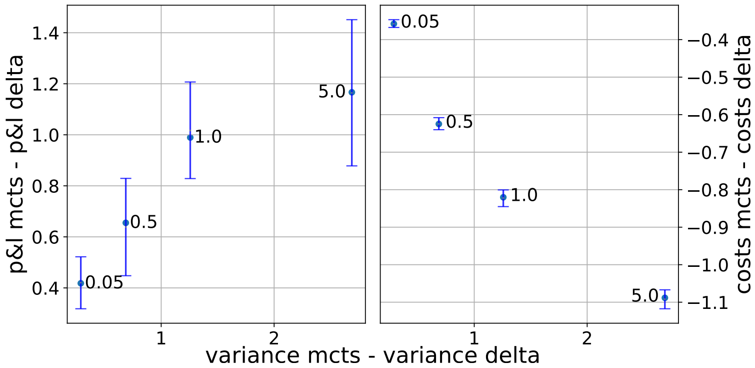
Option Hedging with MCTS

- a_t: current hedge portfolio
- $s_t = [S_t, C_t, \frac{\partial C_t}{\partial S_t}, a_{t-1}]$
- $r_t = C_t(S_t) - C_t(S_{t-1}) - a_t \cdot (S_{t+t} - S_{t-1}) - m \cdot |a_t - a_{t-1}|$

Algorithm 3 Option hedging with MCTS

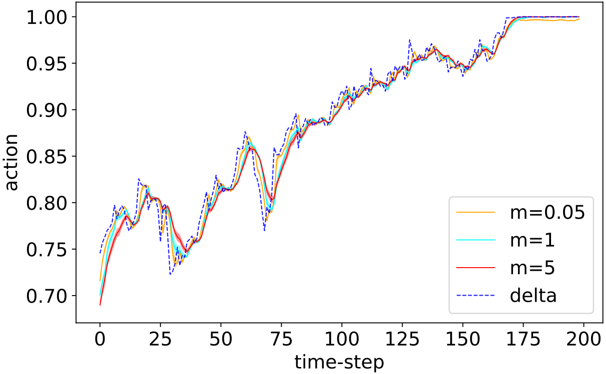
Require: Observe underlying price p_0, option price o_0
  repeat
    Calculate implied Black volatility σ_t and delta d_t from o_t, p_t
    Plan with QL-OL UCT using GBM(p_t, σ_t) with rollouts as delta hedge
    Select action $a_t = \max_a \mathcal{Q}\left(\mathcal{N}_{0,0}, a\right)$
    Observe p_{t+1}, o_{t+1} from the market
  until Option expiry

Results on simulated data



P&l of the MCTS agent w.r.t. the delta hedge (left) and trading costs generated by MCTS agent w.r.t delta hedge (right). Average of 2000 simulations, 95% CI. Results in EUR, annualized and for a single option.

Results on real data



Action on SX7E, single option with strike 90 and expiry 17/06/2021, starting 25 working days before expiry.

Conclusions and Outlook

Conclusions and Outlook

Future works

- Improve the generative model for trading
- Improve the generative model for option hedging
- Extend alphazero [Silver et al., 2017] to stochastic states and continuous actions

Contacts

- Edoardo Vittori - edoardo.vittori@polimi.it
- Amarildo Likmeta - amarildo.likmeta@polimi.it

References

[Bisi et al., 2020] Bisi, L., Sabbioni, L., Vittori, E., Papini, M., and Restelli, M. (2020).
Risk-averse trust region optimization for reward-volatility reduction.
In *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence*.

[Couetoux, 2013] Couetoux, A. (2013).
Monte Carlo tree search for continuous and stochastic sequential decision making problems.
PhD thesis.

[Kocsis and Szepesvári, 2006] Kocsis, L. and Szepesvári, C. (2006).
Bandit based monte-carlo planning.
In *European conference on machine learning*. Springer.

[Silver et al., 2017] Silver, D., Schrittwieser, J., Simonyan, K., Antonoglou, I., Huang, A., Guez, A., Hubert, T., Baker, L., Lai, M., Bolton, A., et al. (2017).
Mastering the game of go without human knowledge.
nature, 550(7676):354–359.

[Watkins, 1989] Watkins, C. J. C. H. (1989).
Learning from delayed rewards.
PhD thesis, King's College, Cambridge.

Experimental setup

- **Financial environment:**
  - both simulated and **real** data
  - a single underlying S;
  - a vanilla call option, unitary notional;
- **Financial parameters:**
  - around 20-day long paths, ending at the option's maturity;
  - 7 time steps per day;
  - σ calibrated from data
- **Search setup:**
  - budget 10,000
  - search depth 4
  - lognormally-generated market data for S: $dS_t = \sigma S_t dW_t$

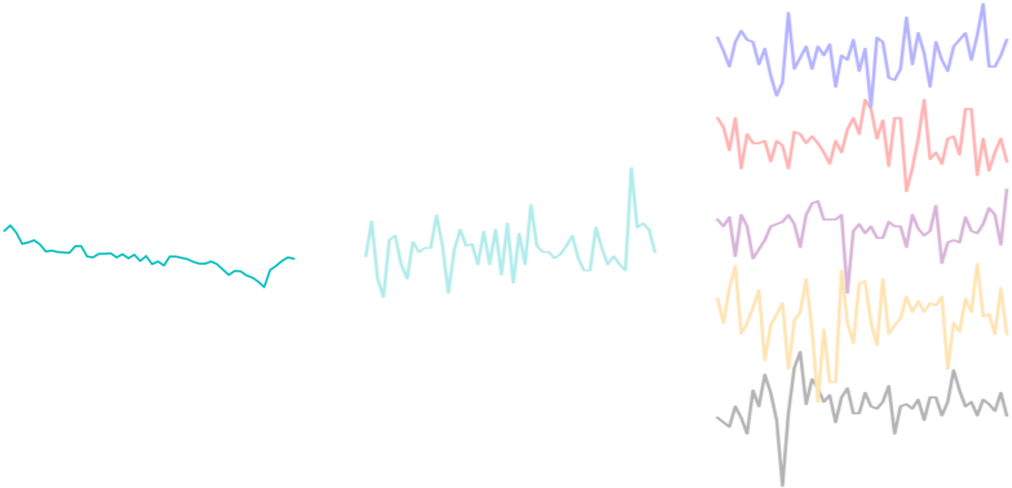
Risk aversion

Considering $\mathcal{R}(s_t, a_t) = f(\rho_t)$, with $\rho_t = C_t(S_t) - C_t(S_{t-1}) - a_t \times (S_{t+k} - S_{t-1}) - c(n)$
We experimented with various forms of f:

- $f(x) = x - \lambda x^2$. It is an approximation of the reward volatility term as defined in [Bisi et al., 2020]. It is possible to vary λ in order to obtain different results balancing the trade-off between risk and reward.
- $f(x) = -x^2$ in this case, we are trying to be completely risk-averse.
- $f(x) = -|abs(x)|$ as before, risk aversion is the only objective.
- $f(x) = x_{MCTS} - x_{delta}$ here we are trying to replicate the delta hedge.
- one last possibility is to use the relationship that higher transaction costs lead to a less risk averse behavior and vice-versa. In other words, making ticksize a parameter to be optimized, where lower ticksize leads to a higher risk aversion.

Clustering generative model

Current Price Window | Current Return Window | Neighbor Returns

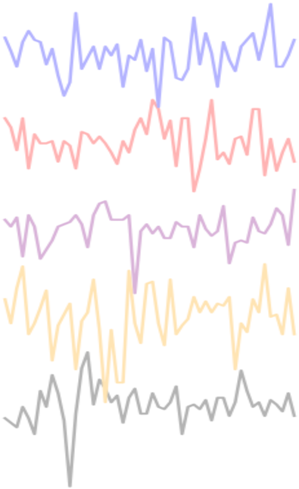


Clustering generative model

Current Price Window

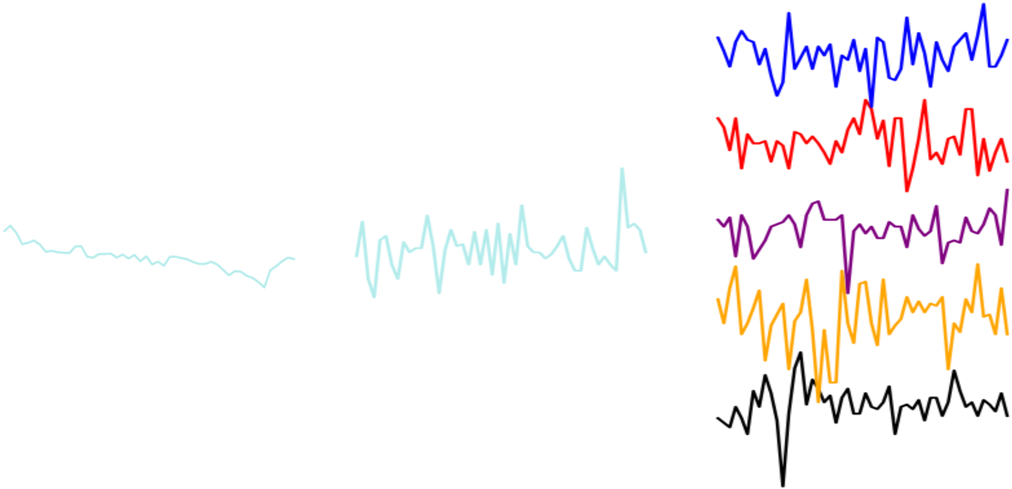
Current Return Window

Neighbor Returns



Clustering generative model

Current Price Window | Current Return Window | Neighbor Returns



Clustering generative model