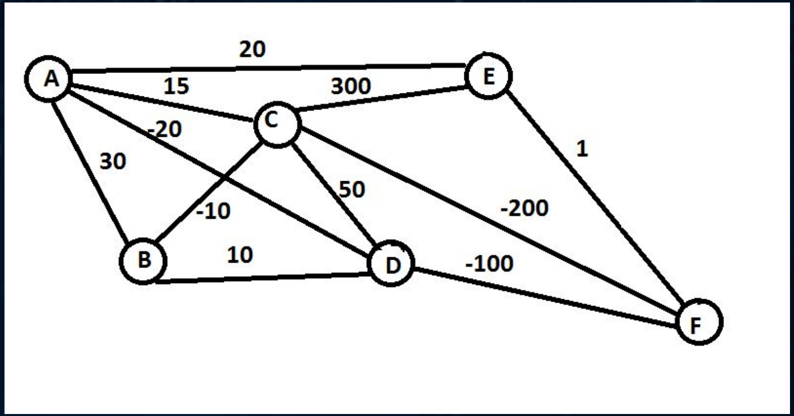
Reinforcement Learning for Trading

Dr Tom Starke
tom@aaaquants.com

QUANTS

What Is Reinforcement Learning?

- Markov decision process
- Take action (A)
- Transition to state (S)
- Get reward (R)
- A policy (π) defines the set of actions
  - Probability of taking an action from a particular state
- The reward we get defines our value (V)

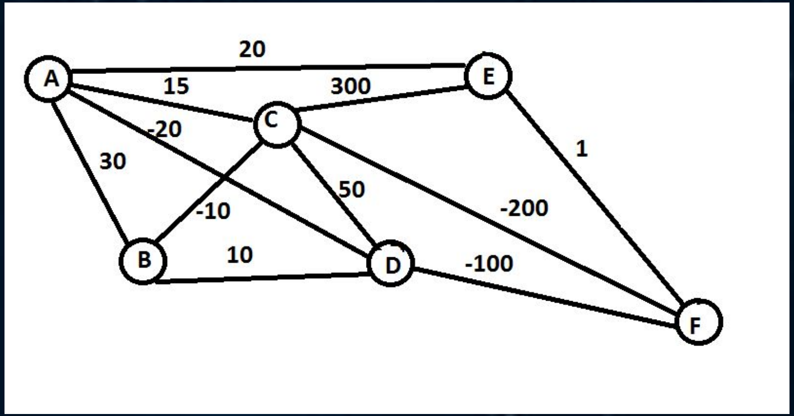


Maximise: $E\,(r_t \mid \pi_t\, s_t)$

Dr Tom Starke
tom@aaaquants.com

Choosing a Policy

- Pick the lowest value at each node
  - Epsilon-greedy
- Not the optimal policy
- Exploration vs exploitation



With some exploration we might achieve better Value!

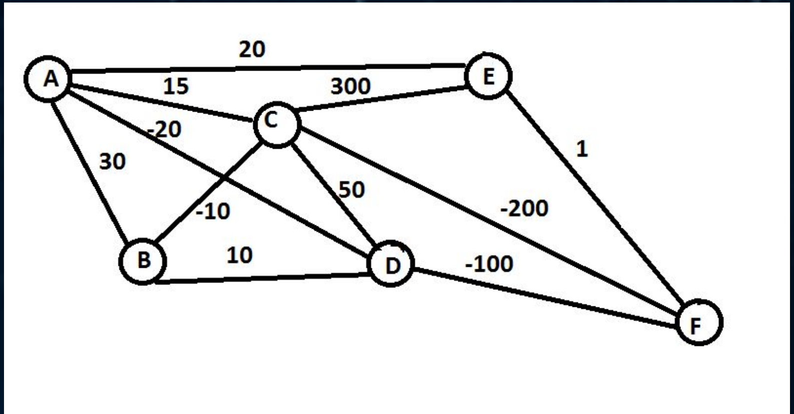
The actions we took **before** we got into a situation with high reward deserves some credit too (not quite as much but some).

Dr Tom Starke
tom@aaaquants.com

QUANTS

Calculating the Value of an Action

- We retroactively apply rewards up a chain of memories
- Certain actions are preferable even if they don't lead to reward
- We define an "action value function" (Q)
- Q defines the value of action a in state s
- Q is traditionally calculated with the Bellman Equation
- We now use deep learning to estimate Q



Dr Tom Starke
tom@aaaquants.com

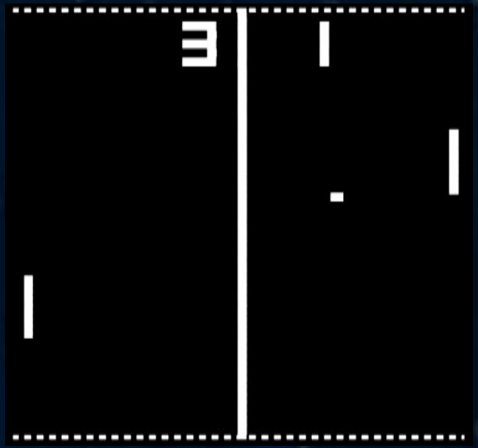
Practical Consideration

1. **"Gamification"** of trading
2. How is the system trained (each game independent)?
3. Reward-function engineering
4. What features do we use for the neural network?
5. How to test the system?
6. What type of ANN should be used?

Dr Tom Starke
tom@aaaquants.com

1) Gamification

- Computer games in their simplest form have:
  - State
  - Cursor movement
  - Reward
- For a trading game this would be equivalent to:
  - Historical and current prices, technical data and alternative sources
  - Buy/Sell/Do Nothing
  - PnL



Dr Tom Starke
tom@aaaquants.com

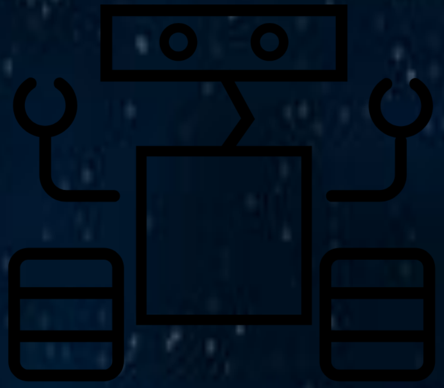
2) How To Train The System?

- Each entry and exit is an individual game
- Run through the price series sequentially or randomly
- Make the whole price series one single game
- Train on each instrument separately or on all with the same learner

Dr Tom Starke
tom@aaaquants.com

3) Reward Function Design

- Pure PnL on exit, otherwise zero
- PnL from start of trade to every time step t
- PnL per tick
- Punishment for long hold times
- Alternatives to PnL:
  - Recognition of trading direction
  - Recognition of correct regime



Dr Tom Starke
tom@aaaquants.com

4) What Features To Use

- OHLCV
- Technical indicators
- Time of day, day of week, time of year
- Different time granularity
- Other instruments
- Alternative data

Dr Tom Starke
tom@aaaquants.com

5) How To Test the System

- Sine waves
- Trend curves
- Random walks
- Different types of autocorrelation
- Adding noise to “clean” test curves
- Recurring patterns

Dr Tom Starke
tom@aaaquants.com

QUANTS

6) What Type of Algorithm?

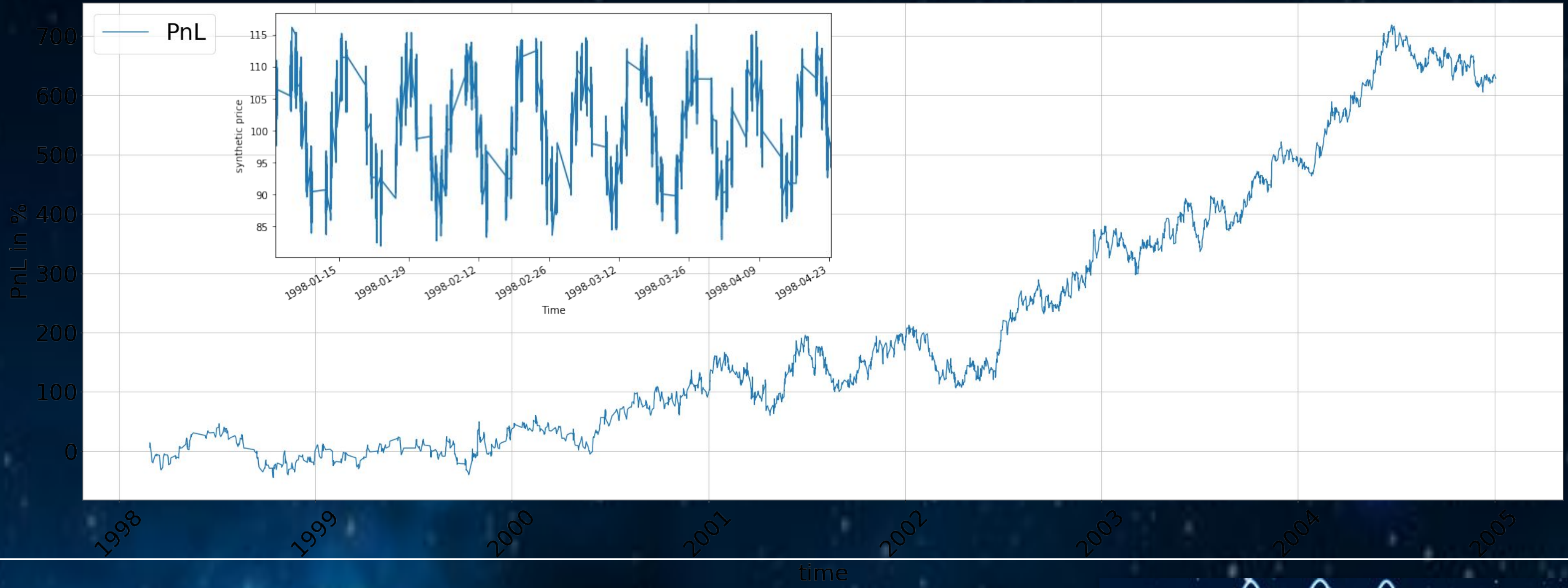
- Standard neural network
- Convolutional NN
- SVM or decision tree
- LSTM

Dr Tom Starke
tom@aaaquants.com

Sine Wave - Trend - No Noise

Dr Tom Starke
tom@aaaquants.com

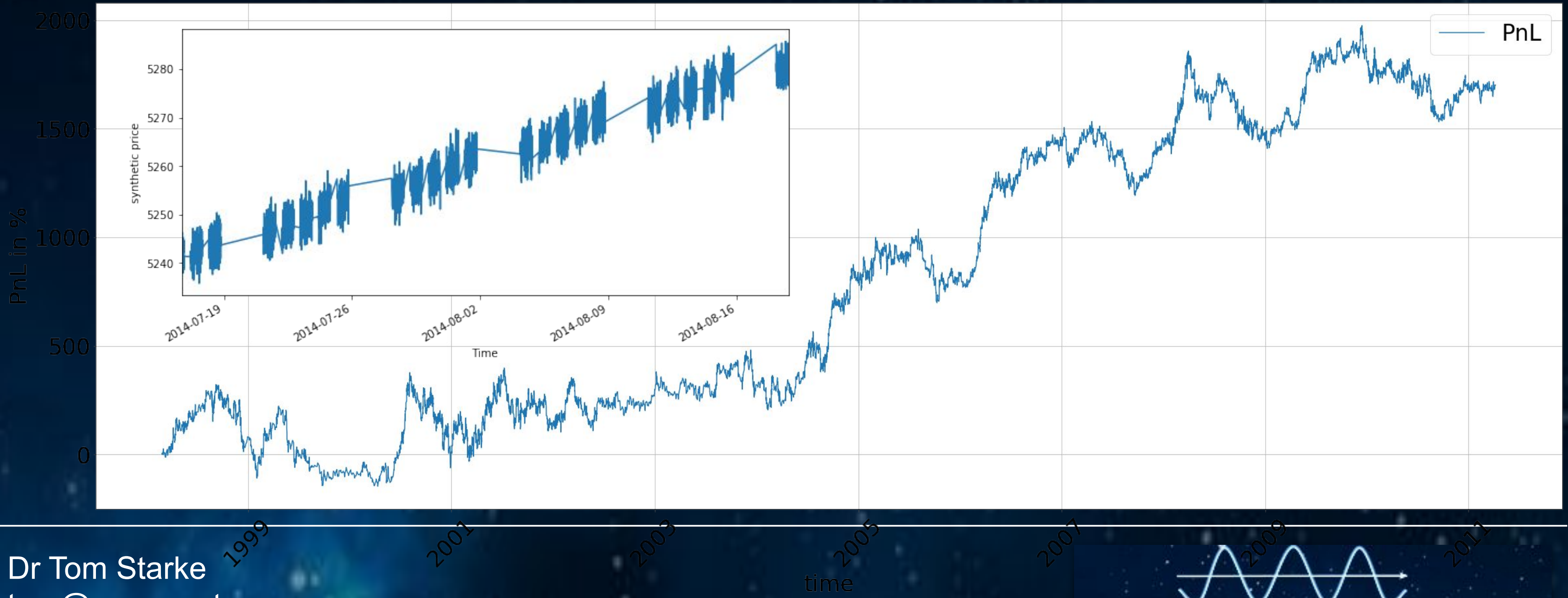
Synthetic Price - Since Wave With Noise



Dr Tom Starke
tom@aaaquants.com

QUANTS

Trend With Noise



Dr Tom Starke
tom@aaaquants.com

Lessons To Be Learned

Ref: https://www.alexirpan.com/2018/02/14/rl-hard.html

- RL can be very sample inefficient
  - Even for a simple Atari game RL needs 70 million frames to achieve human performance
    - Distributional DQN (Bellemare et al, 2017)
- Reward function design is hard
- Rewards in trading are sparse
- Local optima are hard to escape
- RL could just be overfitting peculiar chart patterns
- Results are unstable and hard to reproduce

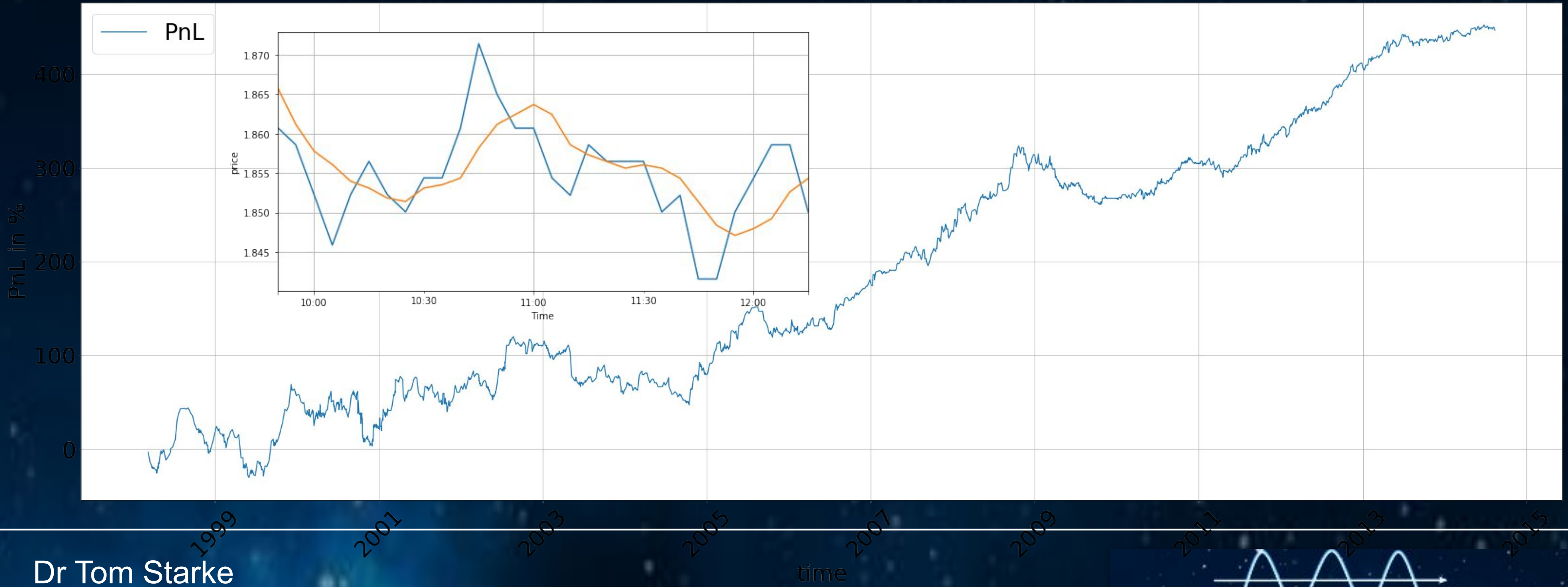
Dr Tom Starke
tom@aaaquants.com

What Makes RL So Hard?

- Financial time series are very noisy
- Financial systems are dynamic - rules keep changing
- Rules evolve by the very act of understanding them
- Computing power is still limited
- New algorithms are yet to be discovered

Dr Tom Starke
tom@aaaquants.com

Performance with 5-period SMA smoothed price curve



Dr Tom Starke
tom@aaaquants.com