EXPERIMENTATION - SHORT DESCRIPTION:

Theoretical usage of Reinforcement learning model in FX market. No transaction costs.

At this point this draft model is just an experiment showing initial theoretical results without transaction costs to explore the model's ability to capture patterns in the original time series. As transaction costs are a critical part of high-frequency trading significantly impacting the trading results, applying transactions cost and retraining the model accordingly is the key planned next step. Until then this experimental version is not applicable to any real-world scenarios.

LLM Coding Assistance: Throughout the development of this project, various LLMs such as GPT-4o, o1, Claude 3.5 Sonnet, Gemini 2.0 Flash were used as coding assistants for tasks such as code generation and debugging.

VERSION CONTROL

Description document			
Version	File name	Date of upload	Type
V01	README_Experimentation - model_FX_RL_PPO_v01 - Doc_v01.docx	January 2025	Initial draft

Code & model			
Version	File name	Date of upload	Type
V01	FX_RL_PPO_v01 .ipynb	January 2025	Initial draft

CONTENTS

CONCEPT

Transaction costs: no transaction costs are assumed in this initial experiment with an aim to first just explore the theoretical capabilities of the model to find patterns in the time series and later further develop the model by applying real-world transaction cost parameters. This is a fundamental limitation of the model, and the current results are therefore highly theoretical and not applicable to real world scenarios.

High level description: a machine learning model for high frequency FX trading using historical hourly data alongside technical and some economic indicators as features.

Goal: to predict profitable actions (Buy, Sell, Hold) on a risk adjusted basis with no transaction cost assumption.

Asset class: EUR/USD. It typically carries more favorable characteristics for financial modeling compared to other asset classes where non-stationarity with a trend typically being involved; it’s less of a concern with FX due to its inherent mean reversion tendency.

Features: wealth of technical indicators and a few economic variables (EUR-USD interest rate differentials).

Model type: reinforcement learning. Learning from trial and error of various actions with a neural network.

Evaluation: various accuracy and financial metrics on both training and testing datasets.

CURRENT STATE

This is an initial draft version, which aims to provide an infrastructure to experiment with various hyperparameters and set of features in a relatively simple environment with evaluation capabilities. As a next step the model can be enhanced based on the lessons learned and major observations of the results of the current version. One key area is introducing real-world transaction costs to the environment and rerun the training potentially with different sets of parameters that would work better for the environment including transaction costs. Until then these results remain highly theoretical.

KEY RESULTS

Please note that the results shown below assume no transaction costs. This is therefore just a theoretical result at this point to see the model’s ability to capture patterns in the original series, and in its current form not applicable to any real-world scenarios. Applying transaction cost and slippage rate is the key planned next step.

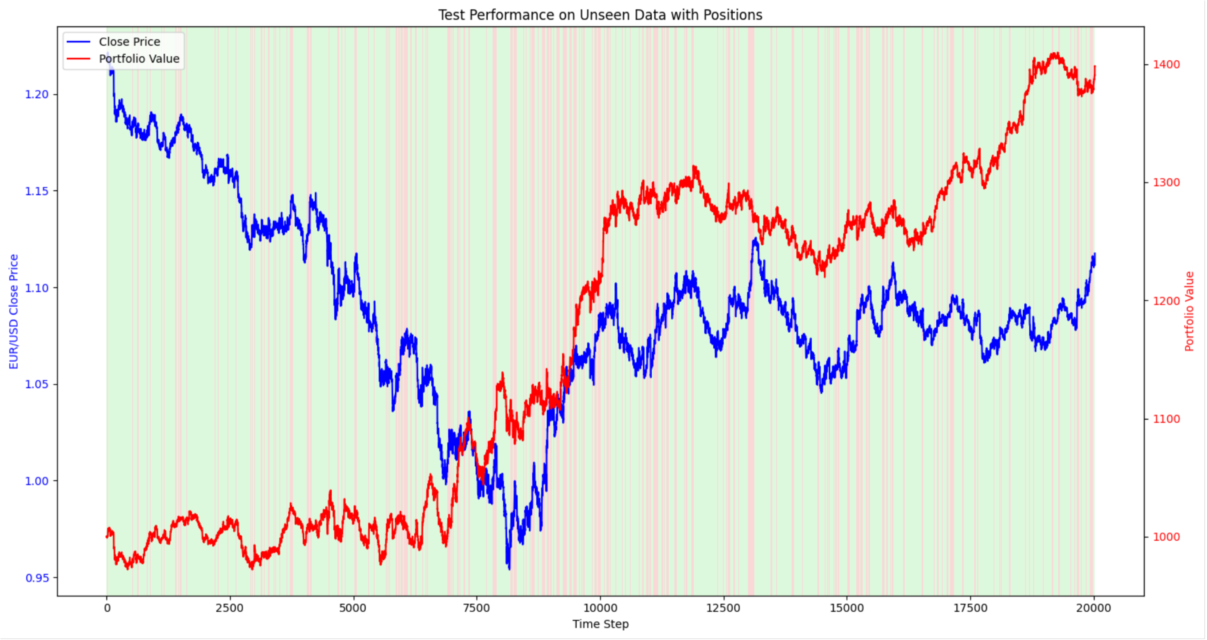
The metrics and charts presented here are the early results of the initial draft version of the current model infrastructure.

Testing results on unseen data: some of the key metrics can be seen below (the full list can be found in the Appendix).

- Initial balance: EUR 1,000
- Testing time window: 2021-2024
- **Compound Annual Growth Rate: 10.2%**
- **Sharpe Ratio: 0.96** (assuming a 2% risk free rate)
- Maximum Drawdown: 7.2%
- Annual Volatility: 8.5%
- Total Holding Period Return: 38.2%
- F1 score (Long positions, open + closed): 0.5813
- **F1 score (Short positions, open + closed): 0.2810**
- **Closed trades' win rate: 55.25%**

The model shows theoretical potential for positive risk-adjusted returns **in a zero-transaction cost environment** primarily leveraging on its ability to capture up price moves relatively well (the general accuracy in financial trade modelling is not expected to be as high as in other domains due to the high noise-to-signal ratio being typically present in financial time series). On the other hand, the model struggles with accurately capturing down price moves, which is a key area of further investigation. Depending on the price regime in the trading window, it can be a significant limitation, which could potentially be counterbalanced by longer trading periods.

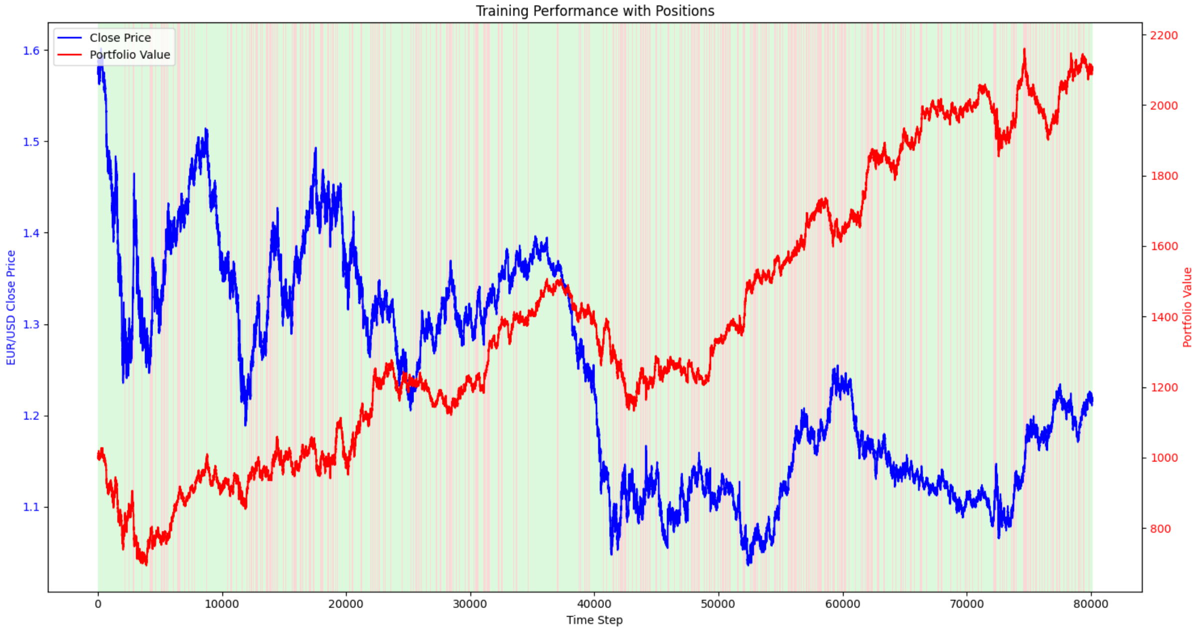
Applying real-world transaction cost alters performance to a significant extent. Including cost elements will make it necessary to retrain the model and experiment with different hyperparameter settings.



Training results: some of the key metrics can be seen below (the full list can be found in the Appendix).

- Initial balance: EUR 1,000
- Testing time window: 2008-2021
- **Compound Annual Growth Rate: 5. 7%**
- **Sharpe Ratio: 0.3** (assuming a 2% risk free rate)
- Maximum Drawdown: 32.5%
- Annual Volatility: 12.4%
- Total Holding Period Return: 108.8%
- F1 score (Long positions, open + closed): 0.5689
- **F1 score (Short positions, open + closed): 0.3251**
- **Closed trades' win rate: 55.27%**

Generally similar pattern can be observed for the training results as well with lower ability for the model to accurately capture down price moves. The Sharpe ratio is materially lower than in the testing window with lower CAGR. This profile could also be impacted by the higher price volatility especially in the first half of the training time window.



METHODOLOGY OUTLINE INCLUDING DATA PREPARATION

Infrastructure: Stable-Baseline 3, PyTorch.

Please see more information about the applied Stable-Baseline3 infrastructure here: https://stable-baselines3.readthedocs.io/en/master/

Model type: Proximal Policy Optimization (PPO), an on-policy model with a discrete action space.

Reward function (target): risk adjusted return over a specified forward time window; more specifically it is the 5-hour forward average return divided by the 5-hour forward volatility. The idea is to 1) reduce noise by taking an average and to 2) include risk associated with the given action to incentivize the model to take those actions where the likely reward exceeds the associated risk.

Features: Several technical indicators including among others volatility, momentum, directional movement, moving averages, mean-reversion, price acceleration, trend strengths, volume related metrics. These technical indicators are enriched with fundamental economic variables of FED and ECB rates alongside some rate differential related indicators. The full list of features can be seen in the code.

Data preparation:

- The main source is IC Markets hourly EUR/USD close price spanning the time frame between 2008 and 2024. The number of data points is about 100k (80% train and 20% test). The dataset includes basic price information (high, low, open, close, volume, spread).

- The economic indicators are sourced from the ECB and FED official websites. The economic indicators resampled to an hourly frequency to adapt to the trading frequency. NaN values are forward filled to avoid lookahead bias.
- The technical indicators are calculated using the library Talib (some of the metrics are manually constructed).
- The up and down hourly price moves appear to be consistent with a normal distribution, and the range of the price level historically is within a relatively narrow band (unlike in the case of stocks and other asset classes, where significant trend is usually present), rendering the classic financial modelling issue of non-stationarity less of a concern for EUR/USD asset. Log-return of the close price nonetheless included as a feature to further mitigating any potential negative impact.
- To handle the high noise-to-signal ratio typically attributed to financial datasets, the 10-hour rolling average of the feature values (where meaningful) have been applied supporting the model to better pick up the signals as opposed to react on noise.

Hyperparameters: The aim was to use values that are within the industry standard range with some experiments and adjustments to adapt the given infrastructure and feature profile.

Evaluation: The model is evaluated on both the training and testing time window. The evaluation includes three main metric types:

- Accuracy statistics such as precision, recall, F1 score.
- Distribution scanning metrics like the ratio of long and short positions.
- Trading metrics such as portfolio value evolution, Sharpe ratio, maximum drawdown, annual volatility, Compound Annual Growth Rate (CAGR).

The ideal situation is having a model that produces high risk-adjusted return on both the training and testing datasets capturing general trends appropriately. The assessment is based on the totality of the metric types.

LIMITATIONS & WEAKNESSES

*No transaction costs: T*ransaction cost and slippage rate assumptions are critical determinants of trading performance. The current results assuming no transaction costs are therefore theoretical and not applicable to any real world scenarios.

Representativeness: It could be that due to some market regime shifts and change in trading behaviour/profile over time, the trained model is not sufficiently representative to the testing and potentially live trading environment. Training still uses data up until around 2020 and the model requires vast amounts of data to learn properly. Hence, this is a fine balance to strike between data availability and representativeness.

Over/underfitting potential: No regularization technique like dropout is currently applied during training, which could lead to overfitting to the training data. No validation dataset is currently being implemented for training, which would also support the detection of potential overfitting. However, based on the accuracy statistics overfitting does not appear to be the case with this particular model training; in fact, underfitting is more likely to be present in the current setup based on the statistics. This is also evident in the very low F1 score for short positions.

Limited learning during training based on some metrics: based on the metrics it seems that there is room for material further accuracy improvements for the training data. The model actually works better on the testing set. The evolution of metric values like loss and value loss also indicates that the model may not learn efficiently and sufficiently during training.

Features: The list currently does not contain sentiment indicators of economic news, and forecasts for economic variables, which could all improve the model accuracy even to a significant extent.

Hyperparameter tuning: Given the nature of neural network-based models, finding the optimal set of hyperparameters is crucial. The currently set parameters may not be close enough to the optimal ones; however, PPO model can be fundamentally more stable than some other reinforcement learning models due to relatively smaller policy updates.

High noise to signal ratio: This is an inherent issue with almost every asset in financial modelling. To mitigate the high noise to signal ratio, 10-hour rolling average of the feature values have been applied at the expense of potentially reducing the signalling power of the features.

Compute resources: Due to compute resource related limitations the hyperparameters like the batch size and net dimension was set at levels that might be suboptimal.

Set of evaluation metrics: the trading results are assessed based on the portfolio value, Sharpe ratio and similar metrics. No 'baseline' metrics are currently available to compare the model results against. For stock trading such a baseline metric would be the total return with a simple 'Buy&Hold' strategy. Such a metric is less meaningful in FX trading.

Discrete action space: There are three possible actions (Buy, Sell, Hold) in the currently applied model infrastructure. This can already be sufficient, but models providing a continuous action space could further enhance the model performance primarily via better position sizing capabilities (i.e. using the action probabilities for position sizing).

Number of data point: Although the current number of datapoints of about 100k is considered to be sufficient, neural network-based models typically work better on large datasets due to better generalization capabilities.

Feature scaling: It was performed based on the entire dataset, with no separate scaling of the training and testing data, which might introduce some lower-level data leakage from testing into training. It's not deemed as a material issue in this particular case. Nonetheless, an area of future improvement.

FUTURE IMPROVEMENT DIRECTIONS

Real-world transaction cost: One of the key areas to improve the current setting is to apply real-world transaction costs in the environment. That will likely make it necessary to experiment with a different parameter setting that would work for the environment that includes transaction costs as well.

Hyperparameters: The results are sensitive to the set of hyperparameters. One direction of improvement could be to use Search Grid to automate the combination of hyperparameters to train the model on to find the optimal group of parameters. On the other hand, it should be carefully designed because such a function would entail significant increase in compute resource and training time.

Model types: Another main direction can be to try and test it with other reinforcement learning models (e.g. Soft Actor-Critic) and infrastructures (e.g. ElegantRL). This particular asset type could benefit from different model learning algorithms, which might yield higher accuracy and trading results on both training and testing data.

Features: additional improvement potential can be to enrich the set of features with more economic data and ideally with sentiment indicators of social and economic news. Additionally, some feature selection mechanism could also be implemented to reduce the highly correlated ones, however, this is less of a concern with neural network-based models compared to 'classic' models like linear regressions.

Infrastructure: enhance the evaluation of the trained models by adding further functionalities, metrics and statistics (e.g. a formal confusion matrix) to better assess the effectiveness of the models. Adding regularization techniques like dropout and early stopping can also improve the model performance.

Training dataset: to increase the number of data points, a data history with a higher frequency (e.g. minute data) could be considered. Alternatively additional FX pairs could be added to the dataset that are expected to behave in a similar fashion to the EUR/USD. The idea behind that would be for the model to find some general relationships that are present for all the major FX pairs. It would bring the benefit of more data points and reduced possibility of overfitting at the expense of likely lower representativeness with respect to individual FX pairs. Finding a careful balance between the two factors can be a sensible, possible step ahead.

Additionally, a more robust data preparation process could be added to the framework with formal outlier handling and stationarity testing functions.

CONFIGURATION

Applied files:

These csv files are used in the code to source historical price data as well as creating technical and economic features.

- ECB policy rate: file ECB Data Portal_20241228020814.
  - Source: https://data.ecb.europa.eu/data/data-categories/ecbeurosystem-policy-and-exchange-rates/official-interest-rates
- FED funds rate: file FEDFUNDS
  - Source: https://fred.stlouisfed.org/series/FEDFUNDS
- EURUSD hourly rate: file EURUSD_H1_200806301600_202408232300date_format
  - Source: IC Markets

How to run it:

- Have the underlying csv files stored in a single folder.
- In the code 'FX_RL_PPO_v01 .ipynb' replace the file path references with yours in the 'if __name__ == "__main__":'.
  - DATA_FOLDER. Add here your path where your csv files are stored.
  - MODEL_FOLDER. Add here your path where you want to trained model to be saved.
- Set the 'TradingConfig' parameters:
  - Hyperparameters (e.g. learning rate)
- Set the Target variable parameter: 'self.forward_window', i.e. set how many hours should be included when making the target variable.
- Run the Python code. It will save the relevant training and testing files in the reference folder.

APPENDICES

TESTING - WITH NO LEVERAGE

```
=== Test Set Metrics ===

CAGR: 0.1025

Sharpe Ratio: 0.9675

Maximum Drawdown: 0.0720

Annual Volatility: 0.0853

Start Value: 1000.0000

End Value: 1381.5136

Total Holding Period Return: 0.3815

Number of Trades (open + closed positions): 20030.0000

Number of Long Trades (open + closed positions): 15474.0000

Number of Short Trades (open + closed positions): 4556.0000

Long-Short Ratio (open + closed positions): 0.7725

Precision (Overall, open + closed positions): 0.4695

Precision (Long, open + closed positions): 0.4751

Precision (Short, open + closed positions): 0.4506

Recall (Long, open + closed positions): 0.7485

Recall (Short, open + closed positions): 0.2042

True_positives: 7352.0000

False_positives: 8002.0000

True_negatives: 2053.0000

False_negatives: 2470.0000

F1 score (Long): 0.5813

F1 score (Short): 0.2810

Number_of_closed_trades: 3591.0000

Closed_trades_win_rate: 0.5525
```

Mean pnl - profitable trades: 1.2887

Mean pnl - unprofitable trades: -1.3609

Median pnl - profitable trades: 0.6888

Median pnl - unprofitabletrades: -0.5558

TRAINING - WITH NO LEVERAGE

=== Training Set Metrics ===

CAGR: 0.0571

Sharpe Ratio: 0.2999

Maximum Drawdown: 0.3250

Annual Volatility: 0.1238

Start Value: 1000.0000

End Value: 2087.8551

Total Holding Period Return: 1.0879

Number of Trades (open + closed positions): 80133.0000

Number of Long Trades (open + closed positions): 57705.0000

Number of Short Trades (open + closed positions): 22428.0000

Long-Short Ratio (open + closed positions): 0.7201

Precision (Overall, open + closed positions): 0.4726

Precision (Long, open + closed positions): 0.4817

Precision (Short, open + closed positions): 0.4492

Recall (Long, open + closed positions): 0.6946

Recall (Short, open + closed positions): 0.2548

True_positives: 27799.0000

False_positives: 29470.0000

True_negatives: 10074.0000

False_negatives: 12221.0000

F1 score (Long): 0.5689

F1 score (Short): 0.3251

Number_of_closed_trades: 15850.0000

Closed_trades_win_rate: 0.5527

Mean pnl - profitable trades: 1.8097

Mean pnl - unprofitable trades: -2.1050

Median pnl - profitable trades: 1.0442

Median pnl - unprofitabletrades: -0.8545