OLPS: A Toolbox for On-Line Portfolio Selection

Bin Li
Economics and Management School, Wuhan University
Wuhan, P.R. China 430072
Generating Name

Doyen Sahoo
School of Information Systems, Singapore Management University
Singapore 178902
Generating Name

Steven C. H. Hoi
School of Information Systems, Singapore Management University
Singapore 178902
Generating Name

Abstract

On-line portfolio selection is a practical financial engineering problem, which aims to sequentially allocate capital among a set of assets in order to maximize long-term return. In recent years, a variety of machine learning algorithms have been proposed to address this challenging problem, but no comprehensive open-source toolbox has been released for various reasons. This article presents the first open-source toolbox for “On-Line Portfolio Selection” (OLPS), which implements a collection of classical and state-of-the-art strategies powered by machine learning algorithms. We hope that OLPS will facilitate the development of new learning methods and enable the performance benchmarking and comparisons of different strategies. OLPS is an open-source project released under Apache License (version 2.0), which is available at [https://github.com/OLPS/](https://github.com/OLPS/) More info and documentations can be found in our project website [http://OLPS.stevenhoi.org](http://OLPS.stevenhoi.org). Keywords: On-line portfolio selection, online learning, trading platform, simulation. ©2014 Bin Li, Doyen Sahoo and Steven Hoi.
Contents

1 Introduction ............................................. 4
  1.1 Target Task ........................................ 4
  1.2 Installation ........................................ 6
     1.2.1 Supported Platforms ............................ 6
     1.2.2 Installation Instructions ....................... 6
     1.2.3 Folders and Paths ............................... 6
  1.3 Implemented Strategies ............................. 6
  1.4 Included datasets ................................... 6
  1.5 Quick Start ......................................... 6

2 Framework and Interfaces ............................. 8
  2.1 Graphical User Interface ........................... 8
     2.1.1 Getting Started ................................ 8
     2.1.2 Algorithm Analyser .............................. 9
     2.1.3 Experimenter .................................... 10
     2.1.4 Results Manager ................................ 10
     2.1.5 Configuration Manager ......................... 13
  2.2 Pseudo Graphical User Interface ................. 13
     2.2.1 Getting Started ................................ 13
     2.2.2 Algorithm Analyser .............................. 14
     2.2.3 Experimenter .................................... 18
     2.2.4 Configuration Manager ......................... 24
  2.3 Command Line Interface ......................... 24
     2.3.1 Trading Manager ................................. 24
     2.3.2 Examples ........................................ 27

3 Strategies ............................................... 29
  3.1 Benchmarks .......................................... 29
     3.1.1 Uniform Buy And Hold ......................... 29
     3.1.2 Best Stock ....................................... 30
     3.1.3 Uniform Constant Rebalanced Portfolios .... 31
     3.1.4 Best Constant Rebalanced Portfolios ....... 31
  3.2 Follow the Winner .................................. 32
     3.2.1 Universal Portfolios ............................ 32
     3.2.2 Exponential Gradient ............................ 33
     3.2.3 Online Newton Step ............................. 33
  3.3 Follow the Loser ..................................... 34
     3.3.1 Anti Correlation .................................. 34
     3.3.2 Passive Aggressive Mean Reversion .......... 35
     3.3.3 Confidence Weighted Mean Reversion ......... 36
     3.3.4 Online Moving Average Reversion ............ 37
  3.4 Pattern Matching based Approaches ............... 38
     3.4.1 Nonparametric Kernel-based Log-optimal Strategy .. 38
1. Introduction

1.1 Target Task

In this section, we first motivate the portfolio selection problem (Gy"{o}rfi et al., 2012; Li and Hoi, 2014) via a real-life example and then formulate the on-line portfolio selection model, which will be used in our model.

Before formulating the problem, let us consider a representative real life problem faced by everybody. John is a 30 year old young man, who has a capital of $10,000. He wants to increase the capital to $1,000,000 by the time he retires at the age of 60, in order to maintain his current living standards. He has no other sources of income, and only relies on this initial capital. He aims to achieve this target by investing in financial markets, which consists of three assets, that is, Microsoft (stock, symbol: “MSFT”), Goldman Sachs (stock, symbol: “GS”), and Treasury bill. All historical records on the three assets, mainly price quotes, are publicly available. Then, every month, John gets new information about the assets and faces one crucial problem, that is, “How to allocate (rebalance) his capital every month such that his allocation can increase his money in the long run?”

Now let us formally formulate the above task. Suppose we have a finite number of $m \geq 2$ investment assets, over which an investor can invest for a finite number of periods. An investment in the market for the $t^{th}$ period is specified by a portfolio strategy $b_t = (b_{t,1}, \ldots, b_{t,m})$, where $b_{t,i}, i = 1, \ldots, m$ represents the proportion of wealth invested in asset $i$ at the beginning of $t^{th}$ period. Typically, portfolio is self-financed and no margin/short sale is allowed, therefore each entry of a portfolio is non-negative and adds up to one, that is, $b_t \in \Delta_m$, where $\Delta_m = \{\mathbf{b}_t : \mathbf{b}_t \succeq 0, \sum_{i=1}^{m} b_{t,i} = 1\}$. The investment procedure is represented by a portfolio strategy, that is, $b_1 = (\frac{1}{m}, \ldots, \frac{1}{m})$ and the following sequence of mappings,

$$b_t : \mathbb{R}_+^{m(t-1)} \rightarrow \Delta_m, t = 2, 3, \ldots,$$

where $b_t = b_t (x_1^{t-1})$ is the portfolio determined at the beginning of $t^{th}$ period upon observing past market behaviors. We denote by $b^n_t = \{b_1, \ldots, b_n\}$ the strategy for $n$ periods, which is the output of an on-line portfolio selection strategy.

On the $t^{th}$ period, a portfolio $b_t$ produces a portfolio period return $s_t$, that is, the wealth changes by a factor of $s_t = b_t \top x_t = \sum_{i=1}^{m} b_{t,i} x_{t,i}$. Since we reinvest and adopt relative prices, the wealth would change multiplicatively. Thus, after $n$ periods, a portfolio strategy $b^n_t$ will produce a portfolio cumulative wealth of $S_n$, which changes initial wealth by a factor of $\prod_{t=1}^{n} b_t \top x_t$,

$$S_n (b^n_1, x^n) = S_0 \prod_{t=1}^{n} b_t \top x_t,$$

where $S_0$ denotes initial wealth, and is set to $1$ for convenience.

1. Here, “month” represents a period, which can be one day, one week, or one month, etc.
2. $\succeq 0$ denotes that each element of the vector is non-negative.

**Input:** $x_1^n$: Historical market price relative sequence  
**Output:** $S_n$: Final cumulative wealth

1. Initialize $S_0 = 1$, $b_1 = (\frac{1}{m}, \ldots, \frac{1}{m})$;
2. for $t = 1, 2, \ldots, n$ do
   3. Portfolio manager learns a portfolio $b_t$;
   4. Market reveals a price relative vector $x_t$;
   5. Portfolio incurs period return $s_t = b_t^\top x_t$ and updates cumulative return $S_t = S_{t-1} \times (b_t^\top x_t)$;
   6. Portfolio manager updates his/her decision rules;
3. end

We present the framework of the above task in Protocol 1. In this task, a portfolio manager’s goal is to produce a portfolio strategy $(b_1^n)$ upon the market price relatives $(x_1^n)$, aiming to achieve certain targets. He/she computes the portfolios in a sequential manner. On each period $t$, the manager has access to the sequence of past price relative vectors $x_t^{t-1}$. He/she then computes a new portfolio $b_t$ for next price relative vector $x_t$, where the decision criterion varies among different managers. Then the manager will rebalance to the new portfolio via buying and selling the underlying stocks. At the end of a trading period, the market will reveal $x_t$. The resulting portfolio $b_t$ is scored based on portfolio period return $s_t$. This procedure is repeated until the end, and the portfolio strategy is finally scored by the portfolio cumulative wealth $S_n$.

Note that we have made several general and common nontrivial assumptions in the above model:

1. Transaction cost: no explicit or implicit transaction costs exist;
2. Market liquidity: one can buy and sell required amount, even fractional, at last close price of any given trading period;
3. Market impact: any portfolio selection strategy shall not influence the market, or any other stocks’ prices.

Finally, as we are going to design intelligent learning algorithms that fit in the above model, let us fix the objective of proposed learning algorithms. For a portfolio selection task, one can choose to maximize risk-adjusted return (Markowitz, 1952; Sharpe, 1964) or to maximize cumulative return (Kelly, 1956; Thorp, 1971) at the end of a period. While the model is online, which contains multiple periods, we choose to maximize the cumulative return (Hakansson, 1971), which is also the objective of almost all existing algorithmic studies.

All the implemented strategies follow the same architecture in Protocol 1 and they are called at Line 3.

---

3. Explicit costs include commissions, taxes, stamp duties, and fees, etc. Implicit costs include the bid-ask spread, opportunity costs, and slippage costs, etc.
1.2 Installation

1.2.1 Supported Platforms

OLPS is based on Matlab (both 32-bit and 64-bit) and Octave (except the GUI Part), thus is supported on 32-bit and 64-bit version of Linux, Mac OS, and Windows. The first version of OLPS is developed and tested on Matlab 2009a, while the latest version of OLPS is tested on Matlab 2014b.

1.2.2 Installation Instructions

Installation of the toolbox is quite straightforward. We can download or clone the latest version of OLPS from the project website at [https://github.com/OLPS/](https://github.com/OLPS/). Then the toolbox is available in the folder. Note that the structure of the toolbox is predefined, which decides the running datasets and logs.

1.2.3 Folders and Paths

The toolbox consists of four folders in relative path: “/Strategy”, “/Data”, “/GUI”, “/Log”, “/Documentation”. The folder “/Strategy” consists of the core strategies for on-line portfolio selection, which will be introduced in Section 3. The folder also consists of the commands used in the Command Line Interface, which will be introduced in Section 2.3. The folder “/Data” includes some popular datasets in forms of .mat, which will be detailed in Section 1.4. The folder “/GUI” includes the files to run the Graphical User Interface, which will be detailed in Section 2.1. The folder “/Log” stores the experimental details of a strategy on a dataset, which will be generated after the simulation process. The folder “/Documentation” contains some documentations of the toolbox, including one summary paper and one comprehensive documentation of the toolbox.

1.3 Implemented Strategies

Table 1 illustrates all implemented strategies in the toolbox.

1.4 Included datasets

As shown in Table 2, six main datasets are widely used for the on-line portfolio selection task. We do not include the high frequency datasets [Li et al., 2013] as they are private. For other variants, such as the reversed datasets [Borodin et al., 2004] and margin datasets [Helmbold et al., 1998], users may generate themselves, which will not be provided in the toolbox.

1.5 Quick Start

To quick start the OLPS toolbox, we provide three types of calling method. The first type is a GUI mode, allowing users to select algorithms and datasets to run. The second type is a batch mode, calling algorithm in the command line. The first GUI mode contains two entries. One is “OLPS_gui.m”, which starts the GUI. Note that Octave does not provide a full-fledged GUI functions, the toolbox only provides GUI in Matlab. The other is “OLPS_pgui.m”, that starts a pseudo GUI in the command line, works in both Octave and Matlab and contains all functionalities and features offered by the GUI.

The third type of calling method runs algorithms by calling the manager function in the command line. For example, we can call “OLPS_cli.m”, which iterates over all strategies on a specified
<table>
<thead>
<tr>
<th>Categories</th>
<th>Strategies</th>
<th>Sections</th>
<th>Strategy Names</th>
</tr>
</thead>
<tbody>
<tr>
<td>Benchmarks</td>
<td>Uniform Buy And Hold</td>
<td>3.1.1</td>
<td>ubah</td>
</tr>
<tr>
<td></td>
<td>Best Stock</td>
<td>3.1.2</td>
<td>best</td>
</tr>
<tr>
<td></td>
<td>Uniform Constant Rebalanced Portfolios</td>
<td>3.1.3</td>
<td>ucrp</td>
</tr>
<tr>
<td></td>
<td>Best Constant Rebalanced Portfolios</td>
<td>3.1.4</td>
<td>bcrp</td>
</tr>
<tr>
<td>Follow the Winner</td>
<td>Universal Portfolios</td>
<td>3.2.1</td>
<td>up</td>
</tr>
<tr>
<td></td>
<td>Exponential Gradient</td>
<td>3.2.2</td>
<td>eg</td>
</tr>
<tr>
<td></td>
<td>Online Newton Step</td>
<td>3.2.3</td>
<td>ons</td>
</tr>
<tr>
<td>Follow the Loser</td>
<td>Anti Correlation</td>
<td>3.3.1</td>
<td>anticor/anticor_anticor</td>
</tr>
<tr>
<td></td>
<td>Passive Aggressive Mean Reversion</td>
<td>3.3.2</td>
<td>pamr/pamr_1/pamr_2</td>
</tr>
<tr>
<td></td>
<td>Confidence Weighted Mean Reversion</td>
<td>3.3.3</td>
<td>cwmr_var/cwmr_stdev</td>
</tr>
<tr>
<td></td>
<td>On-Line Moving Average Reversion</td>
<td>3.3.4</td>
<td>olmar1/olmar2</td>
</tr>
<tr>
<td>Pattern Matching</td>
<td>Nonparametric Kernel-based Log-optimal</td>
<td>3.4.1</td>
<td>bk</td>
</tr>
<tr>
<td></td>
<td>Nonparametric Nearest Neighbor Log-optimal</td>
<td>3.4.2</td>
<td>bnn</td>
</tr>
<tr>
<td></td>
<td>Correlation-driven Nonparametric Learning</td>
<td>3.4.3</td>
<td>corn/cornu/cornk</td>
</tr>
<tr>
<td>Others</td>
<td>M0</td>
<td></td>
<td>m0</td>
</tr>
<tr>
<td></td>
<td>T0</td>
<td></td>
<td>t0</td>
</tr>
</tbody>
</table>

Table 1: All implemented strategies in the toolbox.

<table>
<thead>
<tr>
<th>File Names (.mat)</th>
<th>Dataset</th>
<th>Region</th>
<th>Time Frame</th>
<th># Periods</th>
<th># Assets</th>
</tr>
</thead>
<tbody>
<tr>
<td>nyse-o</td>
<td>NYSE (O)</td>
<td>US</td>
<td>07/03/1962 - 12/31/1984</td>
<td>5651</td>
<td>36</td>
</tr>
<tr>
<td>nyse-n</td>
<td>NYSE (N)</td>
<td>US</td>
<td>01/01/1985 - 06/30/2010</td>
<td>6431</td>
<td>23</td>
</tr>
<tr>
<td>tse</td>
<td>TSE</td>
<td>CA</td>
<td>01/04/1994 - 12/31/1998</td>
<td>1259</td>
<td>88</td>
</tr>
<tr>
<td>sp500</td>
<td>SP500</td>
<td>US</td>
<td>01/02/1998 - 01/31/2003</td>
<td>1276</td>
<td>25</td>
</tr>
<tr>
<td>msci</td>
<td>MSCI</td>
<td>Global</td>
<td>04/01/2006 - 03/31/2010</td>
<td>1043</td>
<td>24</td>
</tr>
<tr>
<td>djia</td>
<td>DJIA</td>
<td>US</td>
<td>01/14/2001 - 01/14/2003</td>
<td>507</td>
<td>30</td>
</tr>
</tbody>
</table>

Table 2: All included datasets in the toolbox.
dataset in the command line. All the parameters used in the file are set according to their original studies, respectively. To iterate over all algorithms and datasets, we can run the following codes.

```matlab
>> OLPS_cli('djia');
>> OLPS_cli('msci');
>> OLPS_cli('nyse-n');
>> OLPS_cli('nyse-o');
>> OLPS_cli('sp500');
>> OLPS_cli('tse');
```

2. Framework and Interfaces

In this toolbox, we provide three interfaces to call the implemented strategies, i.e., Graphical User Interface (GUI), Pseudo Graphical User Interface (PGUI) and Command Line Interface (CLI). The framework can be easily extended to include new algorithms and datasets.

2.1 Graphical User Interface

In the Graphical User Interface (GUI), users will call the implemented algorithms via interaction with the GUI. We provide a menu driven interface for the user to select datasets and algorithms, and input the desired arguments. After providing the inputs and hitting the start button, the algorithm(s) execute. Upon completion, the results and relevant graphs are displayed.

2.1.1 Getting Started

![Figure 1: Starting the Trading Manager.](image)
To start the GUI, we type the following command in the MATLAB.

```matlab
>> OLPS_gui
```

After executing the above command the Trading Manager starts. As shown in Figure 1, the opening window has five buttons. The About and Exit buttons are self-explanatory. The other three are the main functional buttons. The Algorithm Analyser button will start a new window, in which, the user can run a single algorithm and analyse its performance relative to the basic benchmarks. The Experimenter is used for selecting multiple algorithms and comparing their performances. The Configuration button is used to add or delete algorithms and datasets that can be used by the toolbox.

### 2.1.2 Algorithm Analyser

![Algorithm Analyser](image)

**Figure 2** Various components of the Algorithm Analyser.

On pressing the Algorithm Analyser button, a new window opens which will be used for running and analysing an algorithm. Figure 2 depicts the Algorithm Analyser running Online Moving Average Reversion on the S&P500 dataset. There are drop down menus for selecting the algorithm and the dataset. The input parameter fields will dynamically change depending on the inputs the algorithm requires (default parameters have been provided). When a particular dataset is selected, some
preliminary performance details of the algorithm are displayed. There are three types of preliminary results displayed. *Basic Benchmarks* displays the cumulative returns for four simple algorithms - *Uniform Buy And Hold*, *Uniform Constant Rebalanced Portfolio*, *Best Stock in hindsight*, and *Best Constant Rebalanced Portfolio* (BCRP). For more details on these algorithms, please refer to Section 3. The *Returns Distribution* shows the annualized mean return and standard deviation of each asset in the dataset. The *All Assets* option shows the performance graph of cumulative returns of all the assets in the dataset.

### 2.1.3 Experimenter

![Figure 3: Various components of the Experimenter.](image)

When devising trading strategies, we usually want to compare the performance of these strategies relative to each other. For this purpose, we provide the *Experimenter*. On pressing the Experimenter button, a new window opens which will offer us the platform for comparing different strategies. First, the dataset is selected. From the list of algorithms, a subset can be selected to be executed. Among the selected algorithms, the input parameters have to be provided and saved (default values are already there). Figure 3 gives an example of comparing five strategies on the *MSCI World Index* dataset. The five algorithms being compared are *Uniform Buy & Hold*, *Uniform Constant Rebalanced Portfolio*, *Best Constant Rebalanced Portfolio*, *Passive Aggressive Mean Reversion*, *Confidence Weighted Mean Reversion* and *Online Moving Average Reversion*.

### 2.1.4 Results Manager

After hitting the *Start* button in the *Algorithm Analyser* or *Experimenter*, the execution starts. In the Algorithm Analyser, a progress bar indicates the execution of a single algorithm. In the case of Experimenter, there are two progress bars. One indicates the number of algorithms executed (along
with which algorithm is being executed currently), and the other shows the status of completion of that individual algorithm.

When the execution is over, the Results Manager shows all the basic performance metrics of the algorithms. Since we have two different managers - one for analysing a single algorithm, and one for comparing multiple algorithms, we made two different result managers.

![Figure 4: Results Manager for Algorithm Analyser.](image)

**Result Manager 1** The first result manager for Algorithm Analyser is shown in Figure 4. The table in the window quantifies the results of the algorithm as compared to the basic benchmarks. The numbers from this table can directly be copied and pasted. There is large graph space which displays the information on a particular attribute selected in the left column.

**Returns**

It contains information about the daily performance of the algorithm. The user can choose to view the cumulative returns and the daily returns. The option of a log (base 10) plot is provided for easier visualization when the difference in performance of the algorithm and the benchmarks is significantly high.
Risk Analysis

There are six metrics to evaluate the risk, and risk adjusted returns of the algorithm. They are **Sharpe Ratio**, **Calmar Ratio**, **Sortino Ratio**, **Value at Risk**, and **Maximum Draw Down**. An input box called **Window** is provided next to each metric. The purpose of the window is to analyse the consistency of the algorithm, instead of just the final result. For example, entering 252 in the Sharpe Ratio Window, will plot a graph of the Sharpe Ratio of the Algorithm for time period $t - 252$ to $t$, for all $t$. When the window size is large such that $t$ is less than the window size, then the computation starts from $t = 1$. The risk metrics are assumed to be zero for the first 50 time periods. This has been done to avoid extreme values due to lack of data in the initial periods.

Portfolio Analysis

The **Portfolio Allocation** shows the distribution of wealth allocated to each asset by the algorithm. The **Step by Step** helps us look at the portfolio allocation for any particular given day. Lastly we have a portfolio animation which accepts an input called window. Visualizing portfolio changes based on daily frequency can be overwhelming, and difficult to interpret, especially when the daily portfolio changes are significant. Instead we allow the user to choose a moving average portfolio of the last window number of days This results in a smoother change of the portfolio allocation.

![Figure 5: Results Manager for Experimenter.](image-url)
Result Manager 2  The second results manager is very similar to the first manager, except that it
is designed for Experimenter. The table in the window quantifies the performance of the algorithms
relative to each other. Like the first manager, this manager also has three sections. A preview of this
manager can be seen in Figure 5.

Returns  
The daily returns across the entire time period of the dataset for all the algorithms can be over-
whelming to view. A time period can be selected, and the daily performance of the algorithms is
displayed for only that time period.

Risk Analysis  
This section is almost identical to the first results manager. The only difference is that here, the
metrics are evaluated for every algorithm and displayed together.

Portfolio Analysis  
This shows the distribution of portfolio allocation for all the algorithms.

2.1.5 Configuration Manager

Configuration Manager is responsible to adding a newly developed strategy to the toolbox, such
that we can call the algorithm and compare with existing implemented algorithms. We leave the
detail description of this part to the developer manual in Section 4.

2.2 Pseudo Graphical User Interface

The Pseudo Graphical User Interface (PGUI) is basically the GUI in the command line, which is
compatible with both Matlab and Octave. The PGUI mode contains all the same functionalities
as the GUI mode, and users will call the implemented algorithms via interaction with a GUI in the
command line. We provide a menu driven interface for the user to select algorithms and datasets,
and input the desired arguments. After providing the inputs and selecting the execution option, the
algorithm(s) execute. Upon completion, the results and relevant graphs are displayed.

2.2.1 Getting Started

To start the PGUI mode, we type the following command in the MATLAB.

>> OLPS_pgui

After executing the above command, the root menu starts as follows:

>> OLPS_pgui

**************************************************************************
** OLPS: Online Portfolio Selection via Machine Learning **
**************************************************************************
1. Algorithm Analyser
2. Experimenter
3. Configuration
4. About

4. GUI mode in Section 2.1 works only in Matlab, which is proprietary.
5. Exit

Please enter your choice (1-5):

The root menu has five choices. The menu item 4 (About) and 5 (Exit) choices are self-explanatory. The other three are the main functional buttons. The menu item 1 (Algorithm Analyser) will enter Algorithm Analyser, in which, the user can run a single algorithm and analyse its performance relative to the basic benchmarks. The menu item 2 (Experimenter) is used for selecting multiple algorithms and comparing their performances. The menu item 3 (Configuration) is used to add or delete algorithms and datasets that can be used by the toolbox.

2.2.2 Algorithm Analyser

On choosing the menu item 1 (Algorithm Analyser) and Enter, new choices in Algorithm Analyser open which will be used for running and analysing an algorithm.

Please enter your choice (1-7):

Pressing the menu item 1 (Select Algorithm) and Enter will list all available algorithms as follows, and we can select an algorithm to analyze.

Please enter your choice (1-7):

|-- ALGORITHM ANALYSER: SELECT ALGORITHM --|

1. Uniform Buy & Hold (Benchmark)
2. Best Stock (Benchmark)
3. Uniform Constant Rebalanced Portfolio (Benchmark)
4. Best Constant Rebalanced Portfolio (Benchmark)
5. Universal Portfolio
6. Exponential Gradient
7. Online Newton Step
8. Switching Portfolio
9. MO
10. Anticor-1
11. Anticor-2
12. Nonparametric kernel based log optimal strategy
13. Nonparametric nearest neighbour based log optimal strategy
14. Correlation driven non parametric Uniform (CORN-U)
15. Correlation driven non parametric Top K (CORN-K)
16. Passive Aggressive Mean Reversion - BASIC
17. Passive Aggressive Mean Reversion - PAMR-1
18. Passive Aggressive Mean Reversion - PAMR-2
19. Confidence Weighted Mean Reversion (Variance)
20. Confidence Weighted Mean Reversion (Standard Deviation)
21. Online Moving Average Reversion (Simple MA)
22. Online Moving Average Reversion (Exponential MA)

Please enter your choice of algorithm (1-22): 6

You have selected: Exponential Gradient
Returning back to Algorithm Analyser...

After selecting an algorithm by entering its number (eg, 6 for EG algorithm) and Enter, the PGUI will return the the last menu of Algorithm Analyzer.

Then, we can set its parameters by pressing 2 (Set Parameters) and Enter. It will guide the user to set the parameters one by one, as we set for the EG algorithm below. After the selection, it will return to the Algorithm Analyzer.

******************************************************************************
** OLPS: ALGORITHM ANALYSER **
******************************************************************************
1. Select Algorithm
2. Set Parameters
3. Select Dataset
4. Preliminary Visualizations
5. View Current Job
6. START EXECUTION
7. Back
******************************************************************************

Please enter your choice (1-7): 2

Please Enter the new values of parameters for Algorithm:
Algorithm: Exponential Gradient
1) Learning Rate (current value = 0.05): 0.05
2) Transaction Cost (current value = 0): 0
We can further select the dataset by pressing 3 (Select Dataset) and Enter. Below we select the MSCI dataset for the analysis.

Please enter your choice (1-7): 3

--- ALGORITHM ANALYSER: SELECT DATASET ---

1. DJIA (14-Jan-01 to 14-Jan-03)  
2. MSCI (01-Apr-06 to 31-Mar-10)  
3. NYSE (O) (03-Jul-62 to 31-Dec-84)  
4. NYSE (N) (01-Jan-85 to 30-Jun-10)  
5. SP500 (02-Jan-98 to 31-Jan-03)  
6. TSE (04-Jan-94 to 31-Dec-98)

Please enter your choice of Dataset (1-6): 2

You have selected: MSCI (01-Apr-06 to 31-Mar-10)
Returning back to Algorithm Analyser...

We can visualize the assets in the chosen dataset by pressing 4 (Preliminary Visualizations) and Enter. It will pop up three figures, illustrating the wealth curves of the four benchmark strategies (i.e., Uniform BAH, Uniform CRP, Best Stock, and Best CRP), and the wealth curves for all individual assets, and the return distribution of each asset (i.e., the annualized mean return and standard deviation of each asset). Figure 6 shows the example of MSCI dataset.

Till now, we have selected EG algorithm and MSCI dataset, and set its parameters. Our selections can be viewed by pressing 5 (View Current Job) and Enter. Take our case for example,

Please enter your choice (1-7): 5

--- CURRENT JOB ---
Algorithm: Exponential Gradient  
Dataset: MSCI (01-Apr-06 to 31-Mar-10)  
Parameters -  
Learning Rate = 0.05  
Transaction Cost = 0

Returning back to Algorithm Analyser...

Finally, we can analyze the algorithm by pressing the menu item 6 (START EXECUTION) and Enter to execute the choice. The execution will automatically output the following contents, including a comparison with 4 benchmarks in 10 performance metrics.

***************************
Performance of the Algorithm compared to baselines based on several metrics  
"Final Value" [ 0.8986] [ 0.9268] [ 1.5040] [ 1.5057] [ 0.9260]
**AN ON-LINE PORTFOLIO SELECTION TOOLBOX**

![Basic Benchmarks](image1)

![Individual Asset Return Curve](image2)

![Returns Distribution of Each Stock](image3)

**Figure 6:** Preliminary Visualizations of the MSCI dataset.

<table>
<thead>
<tr>
<th></th>
<th>[1x28 char]</th>
<th>[1.7961e-05]</th>
<th>[5.3001e-05]</th>
<th>[4.7016e-04]</th>
<th>[4.6781e-04]</th>
<th>[5.1829e-05]</th>
</tr>
</thead>
<tbody>
<tr>
<td>‘Annualised Return’</td>
<td>[ -0.0255]</td>
<td>[ -0.0182]</td>
<td>[ 0.1036]</td>
<td>[ 0.1039]</td>
<td>[ -0.0184]</td>
<td></td>
</tr>
<tr>
<td>‘Standard Deviation’</td>
<td>[ 0.0155]</td>
<td>[ 0.0158]</td>
<td>[ 0.0125]</td>
<td>[ 0.0122]</td>
<td>[ 0.0158]</td>
<td></td>
</tr>
<tr>
<td>‘Sharpe Ratio’</td>
<td>[ -0.2662]</td>
<td>[ -0.2313]</td>
<td>[ 0.3201]</td>
<td>[ 0.3288]</td>
<td>[ -0.2324]</td>
<td></td>
</tr>
<tr>
<td>‘Calmar Ratio’</td>
<td>[ -0.0394]</td>
<td>[ -0.0312]</td>
<td>[ 0.2585]</td>
<td>[ 0.2498]</td>
<td>[ -0.0315]</td>
<td></td>
</tr>
<tr>
<td>‘Sortino Ratio’</td>
<td>[ 0.0240]</td>
<td>[ 0.0694]</td>
<td>[ 0.7368]</td>
<td>[ 0.7463]</td>
<td>[ 0.0681]</td>
<td></td>
</tr>
<tr>
<td>‘Value at Risk’</td>
<td>[ 0.0236]</td>
<td>[ 0.0245]</td>
<td>[ 0.0210]</td>
<td>[ 0.0206]</td>
<td>[ 0.0245]</td>
<td></td>
</tr>
<tr>
<td>‘Maximum Draw Down’</td>
<td>[ 0.6475]</td>
<td>[ 0.6436]</td>
<td>[ 0.3935]</td>
<td>[ 0.4080]</td>
<td>[ 0.6438]</td>
<td></td>
</tr>
</tbody>
</table>

********************************************
**OLPS: RESULTS MANAGER**
********************************************

1. View Table of Results
2. View Returns Graph
3. View Risk Analysis Plots
4. Portfolio Allocation Analysis
5. Save Results
6. Back

Please enter your choice (1-6):

After execution, the system will automatically enter the result manager, which will show various forms of results. Choosing 1 (View Tables of Results) will compare the results in a table. Choosing 2 (View Returns Graph) will output the the wealth curves (original and logarithmic) of the selected algorithm and the benchmarks, for example, Figure [2]
Choosing the menu item **3** (View Risk Analysis Plots) will output the risk analysis figures of the selected algorithm. For example, Figure 8 shows the Sharpe ratio, Calmar Ratio, Sortino Ratio, Value at Risk, and Maximum Drawdown of the algorithm, respectively.

Choosing the menu item **4** (Portfolio Allocation Analysis) will output the figures on portfolio allocation, i.e., Figure 9 outputs the average allocation of each asset and its standard deviation.

Choosing the menu item **5** (Save Results) will save the table of results to \"Log/Results\". Note that the typed file name needs to be embraced by ‘ and ’.

---

**OLPS: RESULTS MANAGER**

1. View Table of Results
2. View Returns Graph
3. View Risk Analysis Plots
4. Portfolio Allocation Analysis
5. Save Results
6. Back

Please enter your choice (1-6): 5
Enter Name of File to save results: ‘20150622.mat’
Results saved in /Log/Result

Choosing the menu item **6** (Back) will return to Algorithm Analyser.

**2.2.3 EXPERIMENTER**

When devising trading strategies, we usually want to compare the performance of these strategies relative to each other. For this purpose, we provide the **Experimenter**. We can enter the Experimenter by choosing the menu item **2** (Experimenter) in the root menu.

---

**OLPS: Online Portfolio Selection via Machine Learning**

---

18
An On-Line Portfolio Selection Toolbox

Figure 8: View Risk Analysis Plots.

1. Algorithm Analyser
2. Experimenter
3. Configuration
4. About
5. Exit

Please enter your choice (1-5): 2
Figure 9: Portfolio Allocation Analysis.

--- ALGORITHM ANALYSER: SELECT ALGORITHM ---

1. Uniform Buy & Hold (Benchmark)
2. Best Stock (Benchmark)
3. Uniform Constant Rebalanced Portfolio (Benchmark)
4. Best Constant Rebalanced Portfolio (Benchmark)
5. Universal Portfolio
6. Exponential Gradient
7. Online Newton Step
8. Switching Portfolio
9. MO
10. Anticor-1
11. Anticor-2
12. Nonparametric kernel based log optimal strategy
13. Nonparametric nearest neighbour based log optimal strategy
14. Correlation driven non parametric Uniform (CORN-U)
15. Correlation driven non parametric Top K (CORN-K)
16. Passive Aggressive Mean Reversion - BASIC
17. Passive Aggressive Mean Reversion - PAMR-1
18. Passive Aggressive Mean Reversion - PAMR-2
19. Confidence Weighted Mean Reversion (Variance)
20. Confidence Weighted Mean Reversion (Standard Deviation)
21. Online Moving Average Reversion (Simple MA)
22. Online Moving Average Reversion (Exponential MA)

Currently Selected Algorithms:
1

Please Enter the Strategy IDs that you want to compare (1:22)
Eg. To compare Strategies 1, 3, 4, 5, type: [1, 3, 4, 5]
[6 16]

Then, we can set algorithms’ parameters by pressing 2 (Set Parameters). At first, it will show
current job and the selected algorithms’ default parameters. Then, we can choose the algorithms to
be updated. For example, we can choose 2 (Passive Aggressive Mean Reversion - BASIC) and set
its parameters.

Please enter your choice (1-5): 2
The current job is:

CURRENT JOB
Algorithms:

1. Exponential Gradient
   Learning Rate = 0.05
   Transaction Cost = 0

2. Passive Aggressive Mean Reversion - BASIC
   Epsilon = 0.5
   Transaction Cost = 0
Dataset: DJIA (14-Jan-01 to 14-Jan-03)

Which Algorithm Parameters do you want to change: 2

Please Enter the new values of parameters for Algorithm:
Algorithm: Passive Aggressive Mean Reversion - BASIC
1) Epsilon (current value = 0.5): 0.5
2) Transaction Cost (current value = 0): 0

In the Experimenter, the selection of dataset is the same to that in Algorithm Analyser, by pressing 3 (Select Dataset).

After these steps, we can view the current job by pressing 4 (View Current Job), which will show the algorithms and their parameters, and the target dataset.

Please enter your choice (1-5): 4

CURRENT JOB
Algorithms:

1. Exponential Gradient
   Learning Rate = 0.05
   Transaction Cost = 0

2. Passive Aggressive Mean Reversion - BASIC
   Epsilon = 0.5
   Transaction Cost = 0

Dataset: MSCI (01-Apr-06 to 31-Mar-10)

Pressing the menu item 5 (START EXECUTION) will execute all the algorithms on the target dataset. After execution, the system will enter into the result manager, which will automatically show a table comparing the selected algorithms in ten performance metrics.

Performance of the Algorithm compared to baselines based on several metrics

<table>
<thead>
<tr>
<th>Metric</th>
<th>'eg'</th>
<th>'pamr'</th>
</tr>
</thead>
<tbody>
<tr>
<td>Final Value</td>
<td>[0.9260]</td>
<td>[15.2320]</td>
</tr>
<tr>
<td>Mean Return for every period</td>
<td>[5.1829e-05]</td>
<td>[0.0029]</td>
</tr>
<tr>
<td>Annualised Return</td>
<td>[-0.0184]</td>
<td>[0.9309]</td>
</tr>
<tr>
<td>Standard Deviation</td>
<td>[0.0158]</td>
<td>[0.0230]</td>
</tr>
<tr>
<td>Annualised Standard Deviation</td>
<td>[0.2512]</td>
<td>[0.3648]</td>
</tr>
<tr>
<td>Sharpe Ratio</td>
<td>[-0.2324]</td>
<td>[2.4424]</td>
</tr>
<tr>
<td>Calmar Ratio</td>
<td>[-0.0315]</td>
<td>[1.6773]</td>
</tr>
</tbody>
</table>
The Result Manager 2 can show the table of results as above, or we can press 1 (View Table of Results).

Choosing the menu item 2 (View Returns Graph) will show two wealth curves illustrating the selected algorithms, as shown in Figure 10.

Choosing 3 (View Daily Returns) will show the daily returns. Sometimes we have a long period, thus we need to choose the start and end of time period, as follows:

Enter start of time period: 1
Enter end of time period: 200

And a figure similar to Figure 11 illustrating the daily returns will pop up.

Choosing 4 (View Risk Analysis Plots) will pop up five figures for risk analysis, which are similar to Figure 12.
Choosing 5 (Portfolio Allocation Analysis) will output the figures on portfolio allocation, i.e., Figure 13 outputs the average allocation of each asset and its standard deviation.

Choosing 6 (Save Results) will save the table of results to “\Log\Results\”.

Choosing 7 (Back) will return to Algorithm Analyser.

2.2.4 CONFIGURATION MANAGER

2.3 Command Line Interface

In the Command Line Interface (CLI), users can run algorithms by calling the commands. This mode is designed for researchers, who are interested in playing the detail codes. In particular, we provide a meta function named manager, which is responsible for preprocessing (such as initializing datasets and variables, etc.), calling specified strategies, and postprocessing (such as analyzing and outputting the results, etc.).

The root directory contains a sample of calling all algorithms, named “OLPScli.m”, this will help users understand the usage of the CLI mode.

2.3.1 TRADING MANAGER

The Trading Manager, as shown in Algorithm 2, controls the whole simulation of on-line portfolio selection. At the start (Line 2), it loads market data from the specified dataset. Note that this can be easily extended to load data from real brokers. Then, Line 3 and 8 open and close two logging files, one for text and one for .MAT format. Line 4 and 6 measure the computational time of the execution of a specified strategy. Measuring the time in the trading manager ensures a fair comparison of the computational time among different strategies. Line 5 is the core component, which calls the specified strategy with specified parameters. Section 3 will illustrate all included strategies and their usages. Line 7 analyzes the executed results of the strategy, which will be introduced later. The “manager.m” usage is shown as follows.

Usage

```matlab
function [cum_ret, cumprod_ret, daily_ret, ra_ret, run_time]...
    = manager(strategy name, dataset name, varargins, opts);
```

(a) Daily Returns of the Algorithms

Figure 11: View Daily Returns.
• cum_ret: cumulative return;
• cumprod_ret: a vector of cumulative returns at the end of every trading day;
• daily_ret: a vector of daily returns at the end of every trading day;
• ra_ret: analyzed result;
• run_time: computational time of the core strategy (excluding the manager routine);
• strategy_name: the name of strategy. All implemented strategies’ names are listed in the fourth column of Table 1.
Algorithm 2: Trading manager for on-line portfolio selection.

Input: strategy name: A string of the specified strategy;
       dataset name: A string of the specified dataset;
       varargs: A variable-length input argument list for the specified strategy;
       opts: A variable for options controlling the trading environment.

Output: cumulative ret: Final cumulative wealth;
        Cumprod ret: Cumulative wealth at the end of each period;
        daily ret: daily return for each period;
        ra ret: analyzed results, including risk adjusted returns;
        run time: Time for the strategy (in sec).

begin
  Initialize market data from dataset;
  Open the log file and mat file;
  Start the time variables;
  Call strategy with parameters in varargs;
  Terminate the time variables;
  Analyze the results;
  Close the log file and mat file;
end

- dataset name: the name of dataset;
- varargs: variable-length input argument list;
- opts: options for behavioral control.

**Example**  This example calls the ubah (Uniform Buy And Hold, or commonly referred as the market strategy) strategy on the “NYSE (O)” dataset.

```matlab
[cum_ret, cumprod_ret, daily_ret, ra_ret, run_time]...
    = manager('ubah', 'nyse-o', {0}, opts);
```

To facilitate the debugging of trading strategies, we also use controlling variables to control the trading environment. In particular, the last parameter `opts` in the above example contains the controlling variables. As shown in Table 3, it consists of five controlling variables.

<table>
<thead>
<tr>
<th>Variables</th>
<th>Descriptions</th>
<th>Possible Values</th>
<th>Explanation for values</th>
</tr>
</thead>
<tbody>
<tr>
<td>opts.quiet_mode</td>
<td>display debug info?</td>
<td>0 or 1</td>
<td>No or Yes</td>
</tr>
<tr>
<td>opts.display_interval</td>
<td>display info time interval?</td>
<td>Any number (e.g., 500)</td>
<td>Display every 500 periods.</td>
</tr>
<tr>
<td>opts.log_record</td>
<td>record the .log file?</td>
<td>0 or 1</td>
<td>No or Yes</td>
</tr>
<tr>
<td>opts.mat_record</td>
<td>record the .mat file?</td>
<td>0 or 1</td>
<td>No or Yes</td>
</tr>
<tr>
<td>opts.analyze_mode</td>
<td>analyze the algorithm?</td>
<td>0 or 1</td>
<td>No or Yes</td>
</tr>
<tr>
<td>opts.progress</td>
<td>show the progress bar?</td>
<td>0 or 1</td>
<td>No or Yes</td>
</tr>
</tbody>
</table>

Table 3: Controlling variables.

The **Result Manager** analyzes the results and returns an array containing the basic statistics, Sharpe ratio and Calmar ratio and their related statistics. Details about the returned statistics are described in Table 4.

**Usage**

```matlab
function [ra_ret] ...
    = ra_result_analyze(fid, data, cum_ret, cumprod_ret, daily_ret, opts);
```

**2.3.2 Examples**

**Example 1**  Calling BCRP strategy on the SP500 dataset, mute verbosed outputs:

```matlab
>> opts.quiet_mode = 1; opts.display_interval = 500;
 opts.log_mode = 1; opts.mat_mode = 1;
 opts.analyze_mode = 1; opts.progress = 0;
>> manager('bcrp', 'sp500', {0}, opts);
```

Then the algorithm outputs are listed below:

```matlab
>> manager('bcrp', 'sp500', {0}, opts);
----Begin bcrp on sp500-----
-------------------------------------
BCRP(tc=0.0000), Final return: 4.07
-------------------------------------
----End bcrp on sp500-----
```

>>
<table>
<thead>
<tr>
<th>Index</th>
<th>Descriptions</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Number of periods</td>
</tr>
<tr>
<td>2</td>
<td>Strategy’s average period return</td>
</tr>
<tr>
<td>3</td>
<td>Market’s average period return</td>
</tr>
<tr>
<td>4</td>
<td>Strategy’s winning ratio over the market</td>
</tr>
<tr>
<td>5</td>
<td>Alpha ($\alpha$)</td>
</tr>
<tr>
<td>6</td>
<td>Beta ($\beta$)</td>
</tr>
<tr>
<td>7</td>
<td>$t$-statistics</td>
</tr>
<tr>
<td>8</td>
<td>$p$-value</td>
</tr>
<tr>
<td>9</td>
<td>Annualized percentage yield</td>
</tr>
<tr>
<td>10</td>
<td>Annualized standard deviation</td>
</tr>
<tr>
<td>11</td>
<td>Sharpe ratio</td>
</tr>
<tr>
<td>12</td>
<td>Drawdown at the end</td>
</tr>
<tr>
<td>13</td>
<td>Maximum drawdown during the periods</td>
</tr>
<tr>
<td>14</td>
<td>Calmar ratio</td>
</tr>
</tbody>
</table>

Table 4: Vector of the analyzed results.

**Example 2**  Calling BCRP strategy on the SP500 dataset, display verbose outputs:

```matlab
>> opts.quiet_mode = 0; opts.display_interval = 200;
opts.log_mode = 1; opts.mat_mode = 1;
opts.analyze_mode = 1; opts.progress = 0;
>> manager('bcrp', 'sp500', {0}, opts);
```

Then the algorithm outputs are listed below:

```matlab
>> manager('bcrp', 'sp500', {0}, opts);
Running strategy bcrp on dataset sp500
Loading dataset sp500.
Finish loading dataset sp500
The size of the dataset is 1276x25.
----Begin bcrp on sp500----
-------------------------------------
Parameters [tc:0.000000]
day Daily Return Total return
500 1.055339 4.634783
1000 1.018404 4.560191
BCRP(tc=0.0000), Final return: 4.07
-------------------------------------
----End bcrp on sp500----
Stop Time: 2013-0721-13-22-08-144.
Elapse time(s): 2.486262.
Result Analysis
```

28
3. Strategies

This section focuses on describing the implemented strategies in the toolbox. We describe the four implemented categories of algorithms, i.e., Benchmarks, Follow the Winner, Follow the Loser, and Pattern Matching based approaches.

3.1 Benchmarks

In the financial markets, there exist various benchmarks (such as indices, etc.). In this section, we introduce four benchmarks, that is, Uniform Buy And Hold, Best Stock, Uniform Constant Rebalanced Portfolios, and Best Constant Rebalanced Portfolios.

3.1.1 Uniform Buy And Hold

**Description**  “Buy And Hold” (BAH) strategy buys the set of assets at the beginning and holds the allocation of assets till the end of trading periods. BAH with initial uniform portfolio is termed “Uniform Buy And Hold” (UBAH), which is often a market strategy in the related literatures. The final cumulative wealth achieved by a BAH strategy is initial portfolio weighted average of individual stocks’ final wealth,

\[ S_n(BAH(b_1)) = b_1 \cdot \left( \sum_{t=1}^{n} x_t \right), \]
where \( b_1 \) denotes the initial portfolio. In case of UBAH, \( b_1 = (\frac{1}{m}, \ldots, \frac{1}{m}) \). To see its update clearly, BAH’s explicit portfolio update can also be written as,

\[
b_{t+1} = \frac{b_t \odot x_t}{b_t x_t},
\]

(1)

where \( \odot \) denotes the operation of element-wise product.

**Usage**

\[
\text{ubah}(\text{fid}, \text{data}, \{\lambda\}, \text{opts});
\]

- \( \text{fid} \): file handle for writing log file;
- \( \text{data} \): market sequence matrix;
- \( \lambda \in [0, 1) \): proportional transaction cost rate;
- \( \text{opts} \): options for behavioral control.

**Example**  Call market (uniform BAH) strategy on the “NYSE (O)” dataset with a transaction cost rate of 0.

1: >> manager(’market’, ’nyse-o’, \{0\}, opts);
Example Call Best Stock strategy on the “NYSE (O)” dataset with a transaction cost rate of 0.

```
1: >> manager('best', 'nyse-o', (0), opts);
```

3.1.3 Uniform Constant Rebalanced Portfolios

**Description** “Constant Rebalanced Portfolios” (CRP) is a fixed proportion strategy, which rebalances to a preset portfolio at the beginning of every period. In particular, the portfolio strategy can be represented as \( b^n = \{b, b, \ldots \} \). The final cumulative portfolio wealth achieved by a CRP strategy after \( n \) periods is defined as,

\[
S_n (CRP (b)) = \prod_{t=1}^{n} b^\top x_t.
\]

In particular, “Uniform CRP” (UCRP) chooses a uniform portfolio as the preset portfolio, that is, \( b = (\frac{1}{m}, \ldots, \frac{1}{m}) \).

**Usage**

```
ucrp(fid, data, {\lambda}, opts);
```

- fid: file handle for writing log file;
- data: market sequence matrix;
- \( \lambda \in [0, 1) \): transaction costs rate;
- opts: options for behavioral control.

Example Call UCRP strategy on the “NYSE (O)” dataset with a transaction cost rate of 0.

```
1: >> manager('ucrp', 'nyse-o', (0), opts);
```

3.1.4 Best Constant Rebalanced Portfolios

**Description** “Best Constant Rebalanced Portfolios” (BCRP) is a special CRP strategy that sets the portfolio as the portfolio that maximizes the terminal wealth in hindsight. BCRP achieves a final cumulative portfolio wealth as follows,

\[
S_n (BCRP) = \max_{b \in \Delta_m} S_n (CRP (b)) = S_n (CRP (b^*)) ,
\]

and its portfolio is calculated in hindsight as,

\[
b^* = \arg \max_{b^* \in \Delta_m} \log S_n (CRP (b^*)) = \arg \max_{b \in \Delta_m} \sum_{t=1}^{n} \log \left( b^\top x_t \right).
\]
Usage

\[
\text{bcrp}(\text{fid, data, } \{\lambda\}, \text{opts});
\]

- \text{fid}: file handle for writing log file;
- \text{data}: market sequence matrix;
- \(\lambda \in [0, 1)\): Transaction costs rate;
- \text{opts}: options for behavioral control.

Example

Call BCRP strategy on the “NYSE (O)” dataset with a transaction cost rate of 0.

1: >> manager('bcrp', 'nyse-o', {0}, opts);

3.2 Follow the Winner

Follow the Winner approach is characterized by transferring portfolio weights from the underperforming assets (experts) to the outperforming ones.

3.2.1 Universal Portfolios

Description

Cover’s “Universal Portfolios” (UP) \cite{Cover1991} uniformly buys and holds the whole set of CRP experts within the simplex domain. Its cumulative wealth is calculated as,

\[
S_n(UP) = \int_{\Delta_m} S_n(b) \, d\mu(b).
\]

Moreover, we adopt a implementation \footnote{\cite{??}}, which is based on non-uniform random walks that are rapidly mixing and requires a polynomial time.

Usage

\[
\text{up}(\text{fid, data, } \{\lambda\}, \text{opts});
\]

- \text{fid}: file handle for writing log file;
- \text{data}: market sequence matrix;
- \(\lambda \in [0, 1)\): transaction costs rate;
- \text{opts}: options for behavioral control.

Example

Call Cover’s Universal Portfolios on the “NYSE (O)” dataset with default parameters and a transaction cost rate of 0.

1: >> manager('up', 'nyse-o', {0}, opts);
3.2.2 Exponential Gradient

**Description**  “Exponential Gradient” (EG) (Agarwal et al., 2006) tracks the best stock and adopts regularization term to constrain the deviation from previous portfolio, i.e., EG’s formulation is,

\[ \mathbf{b}_{t+1} = \arg \max_{\mathbf{b} \in \Delta_m} \eta \log \mathbf{b} \cdot \mathbf{x}_t - R(\mathbf{b}, \mathbf{b}_t), \]

where \( \eta \) refers to the learning rate and \( R(\mathbf{b}, \mathbf{b}_t) \) denotes relative entropy, or \( R(\mathbf{b}, \mathbf{b}_t) = \sum_{i=1}^{m} b_i \log \frac{b_i}{b_{t,i}} \).

Solving the optimization, we can obtain EG’s portfolio explicit update,

\[ b_{t+1,i} = b_{t,i} \exp \left( \frac{\eta x_{t,i}}{b_t \cdot x_t} \right) / Z, \quad i = 1, \ldots, m, \]

where \( Z \) denotes the normalization term such that the portfolio element sums to 1.

**Usage**

\[
\text{eg(fid, data, \{\eta, \lambda\}, opts);} \]

- \( \text{fid: file handle for writing log file;} \)
- \( \text{data: market sequence matrix;} \)
- \( \eta: \text{Learning rate;} \)
- \( \lambda: \text{Transaction costs rate;} \)
- \( \text{opts: options for behavioral control.} \)

**Example**  Call Exponential Gradient on the “NYSE (O)” dataset with a learning rate of 0.05 and a transaction cost rate of 0.

```matlab
1: >> manager('eg', 'nyse-o', {0.05, 0}, opts);
```

3.2.3 Online Newton Step

**Description**  “Online Newton Step” (ONS) (Agarwal et al., 2006) tracks the best CRP to date and adopts an L2-norm regularization to constrain portfolio’s variability. In particular, its formulation is,

\[ \mathbf{b}_{t+1} = \arg \max_{\mathbf{b} \in \Delta_m} \sum_{\tau=1}^{t} \log (\mathbf{b} \cdot \mathbf{x}_\tau) - \frac{\beta}{2} \| \mathbf{b} \|. \]

Solving the optimization, we can obtain ONS’ explicit portfolio update,

\[ \mathbf{b}_1 = \left( \frac{1}{m}, \ldots, \frac{1}{m} \right), \quad \mathbf{b}_{t+1} = \Pi_{\Delta_m}^A (\delta A_t^{-1} \mathbf{p}_t), \]

with

\[ A_t = \sum_{\tau=1}^{t} \left( \frac{\mathbf{x}_\tau \mathbf{x}_\tau^\top}{(\mathbf{b}_\tau \cdot \mathbf{x}_\tau)^2} \right) + I_m, \quad \mathbf{p}_t = \left( 1 + \frac{1}{\beta} \right) \sum_{\tau=1}^{t} \frac{\mathbf{x}_\tau}{\mathbf{b}_\tau \cdot \mathbf{x}_\tau}, \]

where \( \beta \) is the trade-off parameter, \( \delta \) is a scaling term, and \( \Pi_{\Delta_m}^A (\cdot) \) is an exact projection to the simplex domain.
Usage

ons(fid, data, {η, β, δ, λ}, opts)

- fid: file handle for writing log file;
- data: market sequence matrix;
- η: mixture parameter;
- β: trade off parameter;
- δ: heuristic tuning parameter.
- λ: transaction costs rate
- opts: options for behavioral control.

Example  Call Online Newton Step on the “NYSE (O)” dataset with a transaction cost rate of 0.

1: >> manager('ons', 'nyse-o', {0, 1, 1/8, 0}, opts);

3.3 Follow the Loser

The Follow the Loser approaches assume that the underperforming assets will revert and outperform others in the subsequent periods. Thus, their common behavior is to move portfolio weights from the outperforming assets to the underperforming assets.

3.3.1 Anti Correlation

Description  “Anti Correlation” (Anticor) (Borodin et al., 2004) transfers the wealth from the outperforming stocks to the underperforming stocks via their cross-correlation and auto-correlation. Anticor adopts logarithmic price relatives in two specific market windows, that is, $y_1 = \log (x_{t-w}^t - w)$ and $y_2 = \log (x_{t-w+1}^t)$. It then calculates the cross-correlation matrix between $y_1$ and $y_2$,

$$M_{cov}(i, j) = \frac{1}{w - 1} (y_{1,i} - \bar{y}_1)^\top (y_{2,j} - \bar{y}_2)$$

$$M_{cor}(i, j) = \left\{ \begin{array}{ll}
M_{cov}(i, j) / \sigma_1(i) \cdot \sigma_2(j) & \sigma_1(i), \sigma_2(j) \neq 0 \\
0 & \text{otherwise}
\end{array} \right.$$  

Then following the cross-correlation matrix, Anticor moves the proportions from the stocks increased more to the stocks increased less, in which the corresponding amounts are adjusted according to the cross-correlation matrix. In particular, if asset $i$ increases more than asset $j$ and their sequences in the window are positively correlated, Anticor claims a transfer from asset $i$ to $j$ with the amount equals the cross correlation value ($M_{cor}(i, j)$) minus their negative auto correlation values ($\min \{0, M_{cor}(i, i)\}$ and $\min \{0, M_{cor}(j, j)\}$). These transfer claims are finally normalized to keep the portfolio in the simplex domain.

Usage  We implemented two Anticor algorithms, i.e., $\text{BAH}_{W}(\text{Anticor})$ and $\text{BAH}_{W}(\text{Anticor(Anticor))}$. Their usages are listed below.
ANTICOR(FID, DATA, \{W, \lambda\}, OPTS);

• fid: file handle for writing log file;
• data: market sequence matrix;
• W: Maximal window size;
• \lambda: Transaction cost rates;
• opts: options for behavioral control.

Example Call both Anticor algorithms on the “NYSE (O)” dataset with a maximal window size of 30 and a transaction cost rate of 0.

1: >> manager(‘anticor’, ‘nyse-o’, {30, 0}, opts);
2: >> manager(‘anticor_anticor’, ‘nyse-o’, {30, 0}, opts);

3.3.2 PASSIVE AGGRESSIVE MEAN REVERSION

Description Rather than tracking the best stock, “Passive Aggressive Mean Reversion” (PAMR) (Li et al., 2012) explicitly tracks the worst stocks, while adopting regularization techniques to constrain the deviation from last portfolio. In particular, PAMR’s formulation is,

\[
b_{t+1} = \arg \min_{b \in \Delta_m} \frac{1}{2} \|b - b_t\|^2 \quad \text{s.t.} \quad \ell_{\epsilon}(b; x_t) = 0,
\]

where \(\ell_{\epsilon}(b; x_t)\) denotes a predefined loss function to capture the mean reversion property,

\[
\ell_{\epsilon}(b; x_t) = \begin{cases} 
0 & \text{if } b \cdot x_t \leq \epsilon \\
(b \cdot x_t - \epsilon) & \text{otherwise}
\end{cases}.
\]

Solving the optimization, we can obtain PAMR’s portfolio update,

\[
b_{t+1} = b_t - \tau_t (x_t - \bar{x}_t 1), \quad \tau_t = \max \left\{ 0, \frac{b_t \cdot x_t - \epsilon}{\|x_t - \bar{x}_t 1\|^2} \right\}.
\]

Usage We implemented three PAMR algorithms, i.e., PAMR, PAMR-I and PAMR-II. Their usages are listed below.

\[
pamr(fid, data, \{\epsilon, \lambda\}, opts);
pamr_1(fid, data, \{\epsilon, C, \lambda\}, opts);
pamr_2(fid, data, \{\epsilon, C, \lambda\}, opts);
\]

• fid: file handle for writing log file;
• data: market sequence matrix;
• \epsilon: mean reversion threshold;
• C: aggressive parameter;
• \lambda: transaction cost rates;
• opts: options for behavioral control.
Example Call the three PAMR algorithms on the “NYSE (O)” dataset with mean reversion threshold of 0.5, aggressive parameter of 30 and a transaction cost rate of 0.

1: >> manager('pamr', 'nyse-o', {0.5, 0}, opts);
2: >> manager('pamr_1', 'nyse-o', {0.5, 500, 0}, opts);
3: >> manager('pamr_2', 'nyse-o', {0.5, 500, 0}, opts);

3.3.3 Confidence Weighted Mean Reversion

Description “Confidence Weighted Mean Reversion” (CWMR) [Li et al., 2013] models the portfolio vector a Gaussian distribution, and explicitly updates the distribution following the mean reversion principle. In particular, CWMR’s formulation is,

\[
(\mu_{t+1}, \Sigma_{t+1}) = \arg\min_{\mu \in \Delta_m} \sum \text{KL}(\mathcal{N}(\mu, \Sigma) || \mathcal{N}(\mu_t, \Sigma_t))
\]

s.t. \( \Pr[\mu \cdot x_t \leq \epsilon] \geq \theta \).

Expanding the constraint, the resulting optimization problem is not convex. The authors provided two methods to solve the optimization, i.e., CWMR-Var and CWMR-Stdev. CWMR-Var involves linearizing the constraint and solving the resulting optimization, one can obtain the closed form update scheme as,

\[
\mu_{t+1} = \mu_t - \lambda_{t+1} \Sigma_t (x_t - \bar{x}_t 1), \quad \Sigma_t^{-1} = \Sigma_t^{-1} + 2\lambda_{t+1} \phi x_t x_t^\top
\]

where \( \lambda_{t+1} \) corresponds to the Lagrangian multiplier calculated by Eq. (11) in [Li et al., 2013] and \( \bar{x}_t = \frac{1}{\Sigma_t 1} \Sigma_t x_t \) denotes the confidence weighted price relative average. CWMR-Stdev involves the decomposition of the covariance matrix and can also releases similar portfolio update formulas.

Usage We implemented two CWMR algorithms, i.e., CWMR-Var and CWMR-Stdev. Their usages are listed below.

\[
cwmr_var(fid, data, \{\phi, \epsilon, \lambda\}, opts);
cwmr_stdev(fid, data, \{\phi, \epsilon, \lambda\}, opts);
\]

- fid: file handle for writing log file;
- data: market sequence matrix;
- \( \phi \): confidence parameter;
- \( \epsilon \): mean reversion threshold;
- \( \lambda \): transaction cost rates;
- opts: options for behavioral control.

Example Call the two CWMR algorithms on the “NYSE (O)” dataset with confidence parameter of 2, mean reversion parameter of 0.5 and a transaction cost rate of 0.

1: >> manager('cwmr_var', 'nyse-o', {2, 0.5, 0}, opts);
2: >> manager('cwmr_stdev', 'nyse-o', {2, 0.5, 0}, opts);
3.3.4 **Online Moving Average Reversion**

**Description** “Online Moving Average Reversion” (OLMAR) (7) explicitly predicts next price relatives following the mean reversion idea, i.e., MAR-1 borrows simple moving average,

$$\bar{x}_{t+1} = \frac{1}{w} \left( 1 + \frac{1}{x_t} + \cdots + \frac{1}{\bigodot_{i=0}^{w-2} x_{t-i}} \right),$$

where \( w \) is the window size and \( \bigodot \) denotes element-wise product, and MAR-2 borrows exponential moving average,

$$\bar{x}_{t+1}(\alpha) = \alpha \bar{x}_t + (1 - \alpha) \bar{x}_{t+1},$$

where \( \alpha \in (0, 1) \) denotes the decaying factor and the operations are all element-wise. Then, OLMAR’s formulation is,

$$b_{t+1} = \arg \min_{b \in \Delta_m} \frac{1}{2} \| b - b_t \|^2 \text{ s.t. } b \cdot \bar{x}_{t+1} \geq \epsilon.$$

Solving the optimization, we can obtain its portfolio update,

$$b_{t+1} = b_t + \lambda_{t+1} (\bar{x}_{t+1} - \bar{x}_{t+1} 1),$$

where \( \bar{x}_{t+1} = \frac{1}{m} (1 \cdot \bar{x}_{t+1}) \) denotes the average predicted price relative and \( \lambda_{t+1} \) is the Lagrangian multiplier calculated as,

$$\lambda_{t+1} = \max \left\{ 0, \frac{\epsilon - b_t \cdot \bar{x}_{t+1}}{\| \bar{x}_{t+1} - \bar{x}_{t+1} 1 \|^2} \right\}.$$

**Usage** We implemented two OLMAR algorithms, i.e., OLMAR-I and OLMAR-II. Their usages are listed below.

```matlab
olmar1(fid, data, {\epsilon, W, \lambda}, opts);
olmar2(fid, data, {\epsilon, \alpha, \lambda}, opts);
```

- \( \text{fid: file handle for writing log file;} \)
- \( \text{data: market sequence matrix;} \)
- \( \text{\epsilon: mean reversion threshold;} \)
- \( \text{W: window size for simple moving average;} \)
- \( \text{\alpha \in [0, 1]: decaying factor to calculate exponential moving average;} \)
- \( \text{\lambda \in [0, 1]: transaction cost rates;} \)
- \( \text{opts: options for behavioral control.} \)

**Example** Call the two OLMAR algorithms on the “NYSE (O)” dataset with mean reversion threshold of 10, window size of 5, decaying factor of 0.5, and a transaction cost rate of 0.

```matlab
1: >> manager('olmar1', 'nyse-o', {10, 5, 0}, opts);
2: >> manager('olmar2', 'nyse-o', {10, 0.5, 0}, opts);
```
Algorithm 3: Sample selection framework ($C(\mathbf{x}_1^t, w)$).

**Input:** $\mathbf{x}_1^t$: Historical market sequence; $w$: window size; 
**Output:** $C$: Index set of similar price relatives.

1. Initialize $C = \emptyset$;
2. if $t \leq w + 1$ then
   return;
3. end
4. for $i = w + 1, w + 2, \ldots, t$ do
5. if $\mathbf{x}_{t-w}^{i-1}$ is similar to $\mathbf{x}_{t-w+1}^t$ then
6. $C = C \cup \{i\}$;
7. end
8. end

3.4 Pattern Matching based Approaches

The Pattern Matching based approaches are based on the assumption that market sequences with similar preceding market appearances tend to re-appear. Thus, the common behavior of these approaches is to firstly identify similar market sequences that are deemed similar to the coming sequence, and then obtain a portfolio that maximizes the expected return based on these similar sequences. Algorithm 3 illustrates the first step, or the sample selection procedure. The second step, or the portfolio optimization procedure, often follows the following optimization,

$$b_{t+1} = \arg\max_{b \in \Delta_m} \prod_{i \in C(\mathbf{x}_1^t)} b \cdot \mathbf{x}_i. \quad (2)$$

3.4.1 Nonparametric Kernel-based Log-optimal Strategy

**Description** “Nonparametric kernel-based sample selection” (B$^K$K) identifies the similarity set by comparing two market windows via Euclidean distance,

$$C_K(\mathbf{x}_1^t, w) = \left\{ w < i < t + 1 : \| \mathbf{x}_{t-w+1}^t - \mathbf{x}_{t-w}^{i-1} \| \leq \frac{c}{\ell} \right\},$$

where $c$ and $\ell$ are the thresholds used to control the number of similar samples. Then, it obtains an optimal portfolio via solving Eq. (2).

**Usage**

```matlab
bk_run(fid, data, {K, L, c, \lambda}, opts);
```

- fid: file handle for writing log file;
- data: market sequence matrix;
- K: maximal window size;
- L: used to split the parameter space of each k;
• c: similarity threshold;
• $\lambda \in [0, 1)$: transaction cost rates;
• opts: options for behavioral control.

**Example**  Call the $B^K$ algorithm on the “NYSE (O)” dataset with default parameters and a transaction cost rate of 0.

```matlab
1: >> manager('bk', 'nyse-o', {5, 10, 1, 0}, opts);
```

3.4.2 Nonparametric Nearest Neighbor Log-optimal Strategy

**Description**  “Nonparametric nearest neighbor-based sample selection” ($B^{NN}$) (Győrfi et al., 2008) searches the price relatives whose preceding market windows are within the $\ell$ nearest neighbor of latest market window in terms of Euclidean distance,

$$C_N(x^t_1, w) = \{ w < i < t + 1 : x^{i-1}_{i-w} \text{ is among the } \ell \text{ NNs of } x^t_{t-w+1} \},$$

where $\ell$ is a threshold parameter. Then, the strategy obtains an optimal portfolio via solving Eq. (2).

**Usage**

```matlab
bnn(fid, data, {K, L, $\lambda$}, opts)
```

• fid: file handle for writing log file;
• data: market sequence matrix;
• K: maximal window size;
• L: parameter to split the parameter space of each k;
• $\lambda \in [0, 1)$: transaction cost rates;
• opts: options for behavioral control.

**Example**  Call the $B^{NN}$ algorithm on the “NYSE (O)” dataset with default parameters and a transaction cost rate of 0.

```matlab
1: >> manager('bnn', 'nyse-o', {5, 10, 0}, opts);
```

3.4.3 Correlation-driven Nonparametric Learning Strategy

“Correlation-driven nonparametric sample selection” (CORN) (Li et al., 2011) identifies the similarity among two market windows via correlation coefficient,

$$C_C(x^t_1, w) = \left\{ w < i < t + 1 : \frac{\text{cov}(x^{i-1}_{i-w}, x^t_{t-w+1})}{\text{std}(x^{i-1}_{i-w}) \text{ std}(x^t_{t-w+1})} \geq \rho \right\},$$

where $\rho$ is a pre-defined threshold. Then, it obtains an optimal portfolio via solving Eq. (2).
Usage

corn(fid, data, {w, c, λ}, opts);
cornu(fid, data, {K, L, c, λ}, opts);
cornk_run(fid, data, {K, L, pc, λ}, opts)

- fid: file handle for writing log file;
- data: market sequence matrix;
- w: window size;
- K: maximal window size;
- L: used to split the parameter space of each k;
- c: correlation threshold;
- pc: percentage of experts to be selected;
- $\lambda \in [0, 1)$: transaction cost rates;
- opts: options for behavioral control.

Example  Below we call three CORN algorithms with their default parameters.

1: >> manager('corn', 'nyse-o', {5, 0.1, 0}, opts);
2: >> manager('cornu', 'nyse-o', {5, 1, 0.1, 0}, opts);
3: >> manager('cornk', 'nyse-o', {5, 10, 0.1, 0}, opts);

4. Developer Manual

One important feature of the toolbox is to allow other researchers to add/delete new algorithms. This section will describe the development of new strategies, including developing new strategies and adding the developed strategies into the toolbox such that users can compare the strategy with the state of the art.

4.1 Developing New Strategies

Developing new strategies requires additional efforts on research. Suppose that users have designed a new strategy, i.e., technically, the user has a function that outputs portfolios. Note that in the toolbox we only consider the portfolio selection methods, but it may be further extended to handle single stock trading. To add the strategy to our toolbox, we can reuse the “template.m” in the Strategy folder. The user’s portfolio selection function can be easily called in Line 40, the parameters can be passed through the parameter “varargins”, which is a varying length variables for parameters. Renaming the file name, we can obtain a strategy.

1 function [ cum_ret, cumprod_ret, daily_ret, daily_portfolio] = template( fid, data, varargins, opts )
2 % This is a template for writing a portfolio selection algorithm
3 %
4 % [ cum_ret, cumprod_ret, daily_ret, daily_portfolio] = template( fid, data, varargins, opts )
5 %
6 % Please put the description of your algorithm here
7 % Name of Strategy:
8 % Author:
9 % Description:
%% Make changes to this section to construct your algorithm

%% Read Parameters
p1 = varargin{1};

%% Initialize variables
[r, c] = size(data);
b = ones(c, 1) / c;
returns = zeros(r, 1);
portfolio = ones(r, c) / c;

%% Static
progress = waitbar(0, 'Executing Algorithm...');

%% The looping over the entire dataset for backtesting
% The algorithm looping over r time periods
for t = 1:1:r

%% TO be filled by users %%

%% Static
% compute x of the optimization problem i.e. todaysRelative
todaysRelative = data(t,:);
% Compute the returns of algorithm
portfolio(t, :) = b;
returns(t) = b’ * (todaysRelative - 1);

%% Change this section to describe your strategy portfolio selection method
% Use solver/algorithm to find new portfolio vector at end of time period t

%% Static
% Update Progress
if mod(t, 50) == 0
waitbar(t / r);
end

end

%% Static
% Compute additional statistics (for quick individual run without GUI)
% Can be deleted
stats.finalValue = prod(returns + 1);
Y = r / 252;
stats.sharpe = ((stats.finalValue)^(1/Y) - 1.04) / (std(returns) * sqrt(252));
stats.averageInTopStock = mean(max(portfolio'));
stats.averageInTop2Stocks = flipud(sort(portfolio'));
Moreover, we need to specify its parameters, such that our toolbox can recognize the new strategy. Users can create a configuration file, such as “template_config.m” for template strategy. The config files are straightforward, as follows:

% This is an example of how to write the config file
% Line 1 - name of the strategy
% Line 2 - number of Parameters
% Line 3 - Name of Parameter 1
% Line 4 - Default Value of Parameter 1
% ...(do not leave an empty line)
Template Algorithm
3
p1
0
p2
1
p3
10

4.2 Incorporating New Strategies

After developing the required functions for a new strategy, we can call the new strategy in the command line. Alternatively, we can add the strategy to the toolbox such that users can compare the strategy with the state of the art. The following two sections describe the addition/deletion in the GUI and PGUI modes, respectively.

4.2.1 In the GUI Mode

Here, we describe how to add or delete new algorithms and datasets via Configuration Manager, as shown in Figure 14.

New Strategy A template (“template.m”) has been provided in the Strategy folder which is based on the general framework for online portfolio selection (as described in Protocol 1). The user should enter his code to learn the new portfolio within the specified region of the loop. Without any changes to the code, the template will behave as a Uniform Constant Rebalanced Portfolio strategy, owing to the fact that we start with a uniform portfolio, and never update it. All new strategies coded must remain in the Strategy folder. Once the files are created in the folders, the configuration should
be changed using the Configuration Manager GUI, which controls the loading of algorithms and datasets into the Trading Manager.
Configuration The configuration determines the algorithms and datasets being used in the toolbox. Within the _config_ folder, there is a file called _config_. This is the active configuration, which means the toolbox uses this file to determine which algorithms and datasets would be preloaded. There is another file _config_default_ which is the configuration provided by the toolbox. Initially the content of the default and the active configuration are the same. A new configuration can be created by clicking on the _Configuration_ button in the start window. It automatically loads the active configuration, to which the user can add or delete new algorithms/datasets.

4.2.2 In the PGUI mode

Here, we describe how to add or delete new algorithms and datasets via _Configuration Manager_, by choosing 3. (Configuration).

```
** OLPS: Online Portfolio Selection via Machine Learning **
1. Algorithm Analyser
2. Experimenter
3. Configuration
4. About
5. Exit
```

Please enter your choice (1-5):3

```
** OLPS: CONFIG MANAGER **
1. Add Strategy
2. Remove Strategy
3. Add Dataset
4. Remove Dataset
5. Set Active Configuration
6. Back
```

Please enter your choice (1-6):

New Strategy A template (“template.m”) has been provided in the Strategy folder which is based on the general framework for online portfolio selection (as described in Protocol [1]). The user should enter his code to learn the new portfolio within the specified region of the loop. Without any changes to the code, the template will behave as a _Uniform Constant Rebalanced Portfolio_ strategy, owing to the fact that we start with a uniform portfolio, and never update it. All new strategies coded must remain in the Strategy folder. Once the files are created in the folders, the configuration should be changed using the Configuration Manager GUI, which controls the loading of algorithms and datasets into the Trading Manager.
1. Add Strategy
2. Remove Strategy
3. Add Dataset
4. Remove Dataset
5. Set Active Configuration
6. Back

Please enter your choice (1-6): 1

Instructions:
1. Create new strategy, and save it in folder /Strategy
2. If file name is ‘newStrategy.m’, create and appropriately populate ‘newStrategy_config.m’ in the same folder
3. An example of this type of files is shown in ‘template.m’ and ‘template_config.m’

Enter name of file that you want to add (e.g. ‘newStrategy’): ‘template’

Strategy Added to config. It can now be read by the toolbox.

Set it as the active configuration and restart the toolbox.

**************************
** OLPS: CONFIG MANAGER **
**************************

Please enter your choice (1-6): 2

Name of config file do you want to update (e.g. ‘config_myconfig’): ‘config’

Strategies in current config:
1. Uniform Buy & Hold (Benchmark)
2. Best Stock (Benchmark)
3. Uniform Constant Rebalanced Portfolio (Benchmark)
4. Best Constant Rebalanced Portfolio (Benchmark)
5. Universal Portfolio
6. Exponential Gradient
7. Online Newton Step
8. Switching Portfolio
9. NO
10. Anticor-1
11. Anticor-2
12. Nonparametric kernel based log optimal strategy
13. Nonparametric nearest neighbour based log optimal strategy
14. Correlation driven non parametric Uniform (CORN-U)
15. Correlation driven non parametric Top K (CORN-K)
16. Passive Aggressive Mean Reversion - BASIC
17. Passive Aggressive Mean Reversion - PAMR-1
18. Passive Aggressive Mean Reversion - PAMR-2
19. Confidence Weighted Mean Reversion (Variance)
20. Confidence Weighted Mean Reversion (Standard Deviation)
21. Online Moving Average Reversion (Simple MA)
22. Online Moving Average Reversion (Exponential MA)
23. Template Algorithm

Enter ID of Strategy to be removed: 23
Strategy removed from config. Set it as the active configuration and restart the toolbox.

Configuration

The configuration determines the algorithms and datasets being used in the toolbox. By maintaining different configuration files, users can switch from their config files to others. Thus, they can use different set of algorithms and datasets. Within the config folder, there is a file called config. This is the active configuration, which means the toolbox uses this file to determine which algorithms and datasets would be preloaded. There is another file config_default which is the configuration provided by the toolbox. Initially the content of the default and the active configuration are the same. A new configuration can be created by clicking on the Configuration button in
the start window. It automatically loads the active configuration, to which the user can add or delete new algorithms/datasets.

***************************************************
** OLPS: Online Portfolio Selection via Machine Learning **
***************************************************
1. Algorithm Analyser
2. Experimenter
3. Configuration
4. About
5. Exit

Please enter your choice (1-5):3

***************************************************
** OLPS: CONFIG MANAGER **
***************************************************
1. Add Strategy
2. Remove Strategy
3. Add Dataset
4. Remove Dataset
5. Set Active Configuration
6. Back

Please enter your choice (1-6):5

Name of config file do you want to make active (e.g. ‘config_myconfig’):’config_default’

Update Configuration. Restart toolbox to read the updated configuration

***************************************************
** OLPS: CONFIG MANAGER **
***************************************************
1. Add Strategy
2. Remove Strategy
3. Add Dataset
4. Remove Dataset
5. Set Active Configuration
6. Back

Please enter your choice (1-6):

4.3 Adding New Datasets

4.3.1 In the GUI mode

A dataset is in the form of price relative vectors of various assets. The $t^{th}$ row represents the price relative of all the assets at time $t$. The user just has to save the new price relative matrix in the Data folder. Data of different frequencies can be used as well. All the datasets provided in the toolbox are of daily frequency. Once the files are created in the folders, the configuration should be changed using the Configuration Manager GUI, which controls the loading of algorithms and datasets into the Trading Manager.

4.3.2 In the PGUI mode

A dataset is in the form of price relative vectors of various assets. The $t^{th}$ row represents the price relative of all the assets at time $t$. The user just has to save the new price relative matrix in the Data folder. Data of different frequencies can be used as well. All the datasets provided in the toolbox...
are of daily frequency. Once the files are created in the folders, the configuration should be changed using the Configuration Manager, which controls the loading of algorithms and datasets into the Trading Manager.

**************************
**OLPS: CONFIG MANAGER **
**************************
1. Add Strategy
2. Remove Strategy
3. Add Dataset
4. Remove Dataset
5. Set Active Configuration
6. Back

Please enter your choice (1-6): 4
Name of config file do you want to update (e.g. 'config_myconfig'): 'config'
Datasets in current config:
1. DJIA (14-Jan-01 to 14-Jan-03)
2. MSCI (01-Apr-06 to 31-Mar-10)
3. NYSE (O) (03-Jul-62 to 31-Dec-84)
4. NYSE (N) (01-Jan-85 to 30-Jun-10)
5. SP500 (02-Jan-98 to 31-Jan-03)
6. TSE (04-Jan-94 to 31-Dec-98)
Enter ID of Data to be removed: 6
Dataset removed from config. Set it as the active configuration and restart the toolbox.

**************************
**OLPS: CONFIG MANAGER **
**************************
1. Add Strategy
2. Remove Strategy
3. Add Dataset
4. Remove Dataset
5. Set Active Configuration
6. Back

Please enter your choice (1-6): 3
First, save new dataset in folder /Data, in the correct format.
Enter name of file (e.g. 'newData'): 'TSE'
Enter Data Description (e.g. 'New Data - 1st Jan, 2014 to 1st Jan, 2015'): 'New TSE data'
Enter frequency (e.g. 1 for daily data): 1
Enter name of config file do you want to update (e.g. 'config_myconfig'): 'config'

Added new Dataset.

**************************
**OLPS: CONFIG MANAGER **
**************************
1. Add Strategy
2. Remove Strategy
3. Add Dataset
4. Remove Dataset
5. Set Active Configuration
6. Back

Please enter your choice (1-6): 4
Name of config file do you want to update (e.g. 'config_myconfig'): 'config'
Datasets in current config:
1. DJIA (14-Jan-01 to 14-Jan-03)
5. Conclusion

In this manual, we describe the On-Line Portfolio Selection (OLPS) toolbox in detail. OLPS is the first toolbox for the research of on-line portfolio selection problem. It is easy to use and can be extended to include new algorithms and datasets. We hope that this toolbox can facilitate the further research in this topic.
References


