Journal of Machine Learning Research 1 (2014) 1-30 Submitted 4/00; Published 10/00

OLPS: A Toolbox for On-Line Portfolio Selection

Bin Li BINLI.WHU@WHU.EDU.CN
Economics and Management School, Wuhan University
Wuhan, P.R. China 430072
Doyen Sahoo DOYENSAHOO.2014@SMU.EDU.SG
School of Information Systems, Singapore Management University
Singapore 178902
Steven C. H. Hoi CHHOI@SMU.EDU.SG
School of Information Systems, Singapore Management University
Singapore 178902

Editor:

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 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/`. More info and documentations can be found in our project website:`http://OLPS.stevenhoi.org`.
Keywords: On-line portfolio selection, online learning, trading platform, simulation.

Contents

1. Introduction

1.1 Target Task

In this section, we first motivate the portfolio selection problem (Györfi 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 formally 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[1], 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 a investor can invest for a finite number of $n \geq 1$ periods.

On the t^{th} period, $t = 1, \ldots, n$, the asset (close) prices are represented by a vector $\mathbf{p}_t \in \mathbb{R}_+^m$, and each element $p_{t,i}, i = 1, \ldots, m$ represents the close price of asset i. Their price changes are represented by a *price relative vector* $\mathbf{x}_t \in \mathbb{R}_+^m$, each component of which denotes the ratio of t^{th} close price to last close price, that is, $x_{t,i} = \frac{p_{t,i}}{p_{t-1,i}}$. Thus, an investment in asset i throughout period t changes by a factor of $x_{t,i}$. Let us denote by $\mathbf{x}_1^n = \{\mathbf{x}_1, \ldots, \mathbf{x}_n\}$ a sequence of price relative vectors for n periods, and $\mathbf{x}_s^e = \{\mathbf{x}_s, \ldots, \mathbf{x}_e\}, 1 \leq s < e \leq n$ as a market window.

An investment in the market for the t^{th} period is specified by *portfolio vector* $\mathbf{b}_t = (b_{t,1}, \ldots, b_{t,m})$, 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, $\mathbf{b}_t \in \Delta_m$, where $\Delta_m = \{\mathbf{b}_t : \mathbf{b}_t \succeq 0, \sum_{i=1}^m b_{t,i} = 1\}$[2]. The investment procedure is represented by a *portfolio strategy*, that is, $\mathbf{b}_1 = \left(\frac{1}{m}, \ldots, \frac{1}{m}\right)$ and the following sequence of mappings,

$$\mathbf{b}_t : \mathbb{R}_+^{m(t-1)} \to \Delta_m, t = 2, 3, \ldots,$$

where $\mathbf{b}_t = \mathbf{b}_t\left(\mathbf{x}_1^{t-1}\right)$ is the portfolio determined at the beginning of t^{th} period upon observing past market behaviors. We denote by $\mathbf{b}_1^n = \{\mathbf{b}_1, \ldots, \mathbf{b}_n\}$ the strategy for n periods, which is the output of an on-line portfolio selection strategy.

On the t^{th} period, a portfolio $\mathbf{b}_t$ produces a *portfolio period return* s_t, that is, the wealth changes by a factor of $s_t = \mathbf{b}_t^\top \mathbf{x}_t = \sum_{i=1}^m b_{t,i} x_{t,i}$. Since we reinvest and adopt relative prices, the wealth would change multiplicatively. Thus, after n periods, a portfolio strategy $\mathbf{b}_1^n$ will produce a *portfolio cumulative wealth* of S_n, which changes initial wealth by a factor of $\prod_{t=1}^n \mathbf{b}_t^\top \mathbf{x}_t$,

$$S_n\left(\mathbf{b}_1^n, \mathbf{x}_1^n\right) = S_0 \prod_{t=1}^n \mathbf{b}_t^\top \mathbf{x}_t,$$

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.

Protocol 1: On-line portfolio selection.

Input: $\mathbf{x}_1^n$: Historical market price relative sequence
Output: S_n: Final cumulative wealth

1 Initialize $S_0 = 1, \mathbf{b}_1 = \left(\frac{1}{m}, \ldots, \frac{1}{m}\right)$;
2 **for** $t = 1, 2, \ldots, n$ **do**
3 Portfolio manager learns a portfolio $\mathbf{b}_t$;
4 Market reveals a price relative vector $\mathbf{x}_t$;
5 Portfolio incurs period return $s_t = \mathbf{b}_t^\top \mathbf{x}_t$ and updates cumulative return $S_t = S_{t-1} \times \left(\mathbf{b}_t^\top \mathbf{x}_t\right)$;
6 Portfolio manager updates his/her decision rules;
7 **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 ($\mathbf{b}_1^n$) upon the market price relatives ($\mathbf{x}_1^n$), 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 $\mathbf{x}_1^{t-1}$. He/she then computes a new portfolio $\mathbf{b}_t$ for next price relative vector $\mathbf{x}_t$, 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 $\mathbf{x}_t$. The resulting portfolio $\mathbf{b}_t$ 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[3] 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/`. 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, which allows users to select algorithms and datasets to run. The second type is a batch mode, which calls 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", which starts a pseudo GUI in the command line, which 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

Categories	Strategies	Sections	Strategy Names
Benchmarks	Uniform Buy And Hold	3.1.1	`ubah`
	Best Stock	3.1.2	`best`
	Uniform Constant Rebalanced Portfolios	3.1.3	`ucrp`
	Best Constant Rebalanced Portfolios	3.1.4	`bcrp`
Follow the Winner	Universal Portfolios	3.2.1	`up`
	Exponential Gradient	3.2.2	`eg`
	Online Newton Step	3.2.3	`ons`
Follow the Loser	Anti Correlation	3.3.1	`anticor`/`anticor_anticor`
	Passive Aggressive Mean Reversion	3.3.2	`pamr`/`pamr_1`/`pamr_2`
	Confidence Weighted Mean Reversion	3.3.3	`cwmr_var`/`cwmr_stdev`
	On-Line Moving Average Reversion	3.3.4	`olmar1`/`olmar2`
Pattern Matching	Nonparametric Kernel-based Log-optimal	3.4.1	`bk`
	Nonparametric Nearest Neighbor Log-optimal	3.4.2	`bnn`
	Correlation-driven Nonparametric Learning	3.4.3	`corn`/`cornu`/`cornk`
Others	M0		`m0`
	T0		`t0`

Table 1: All implemented strategies in the toolbox.

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

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.

```
>> 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

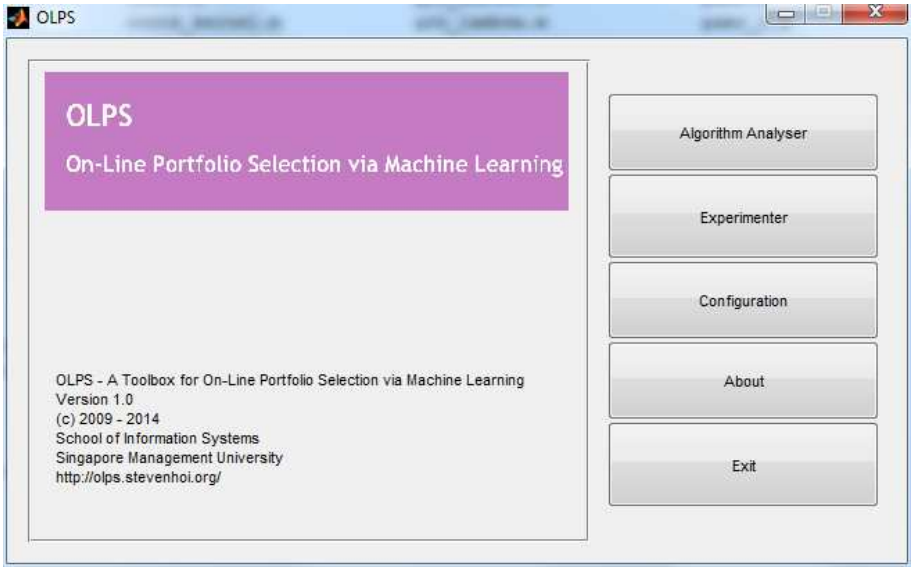


Figure 1: Starting the Trading Manager.

To start the GUI, we type the following command in the 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

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

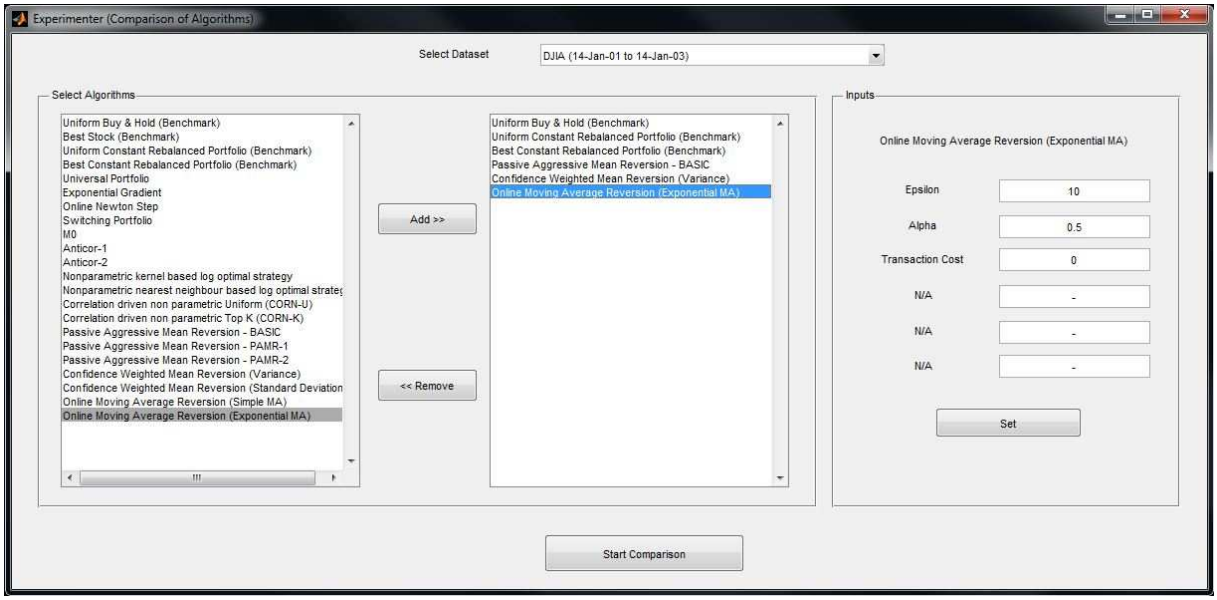


Figure 3: Various components of the Experimenter.

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.

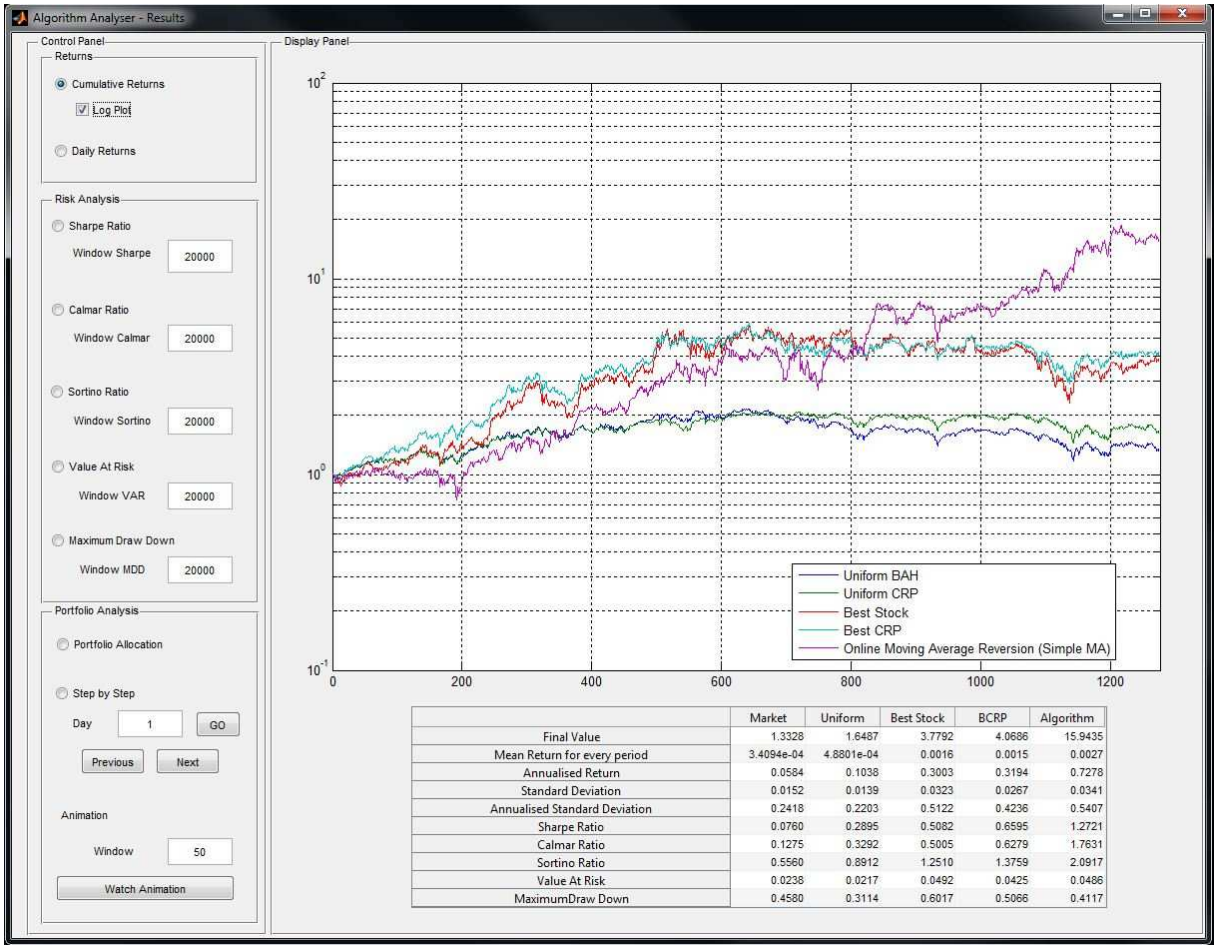


Figure 4: Results Manager for Algorithm Analyser.

Result Manager 1 The first result manager for *Algorithm Analyzer* 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.

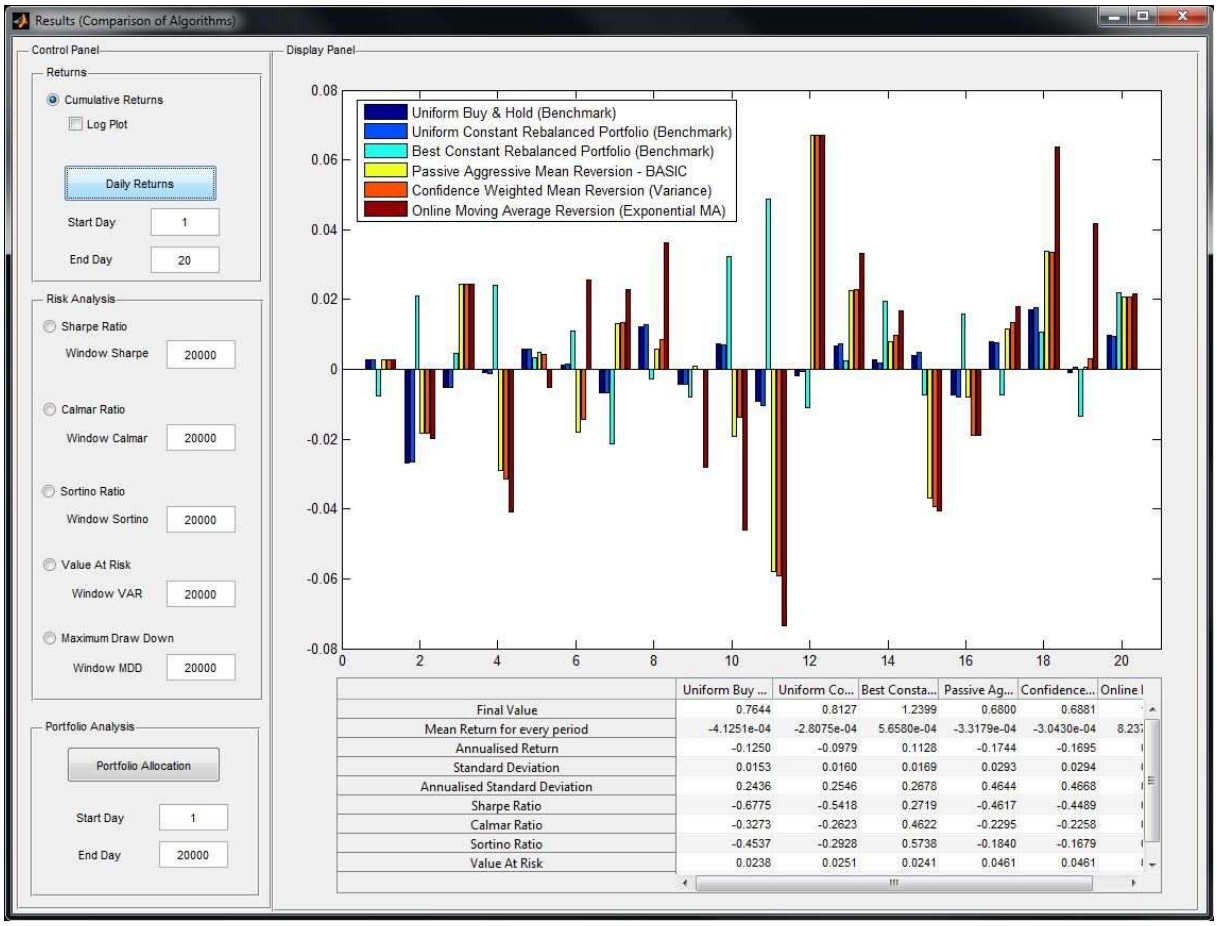


Figure 5: Results Manager for Experimenter.

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 overwhelming 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

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

New dataset 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.

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.

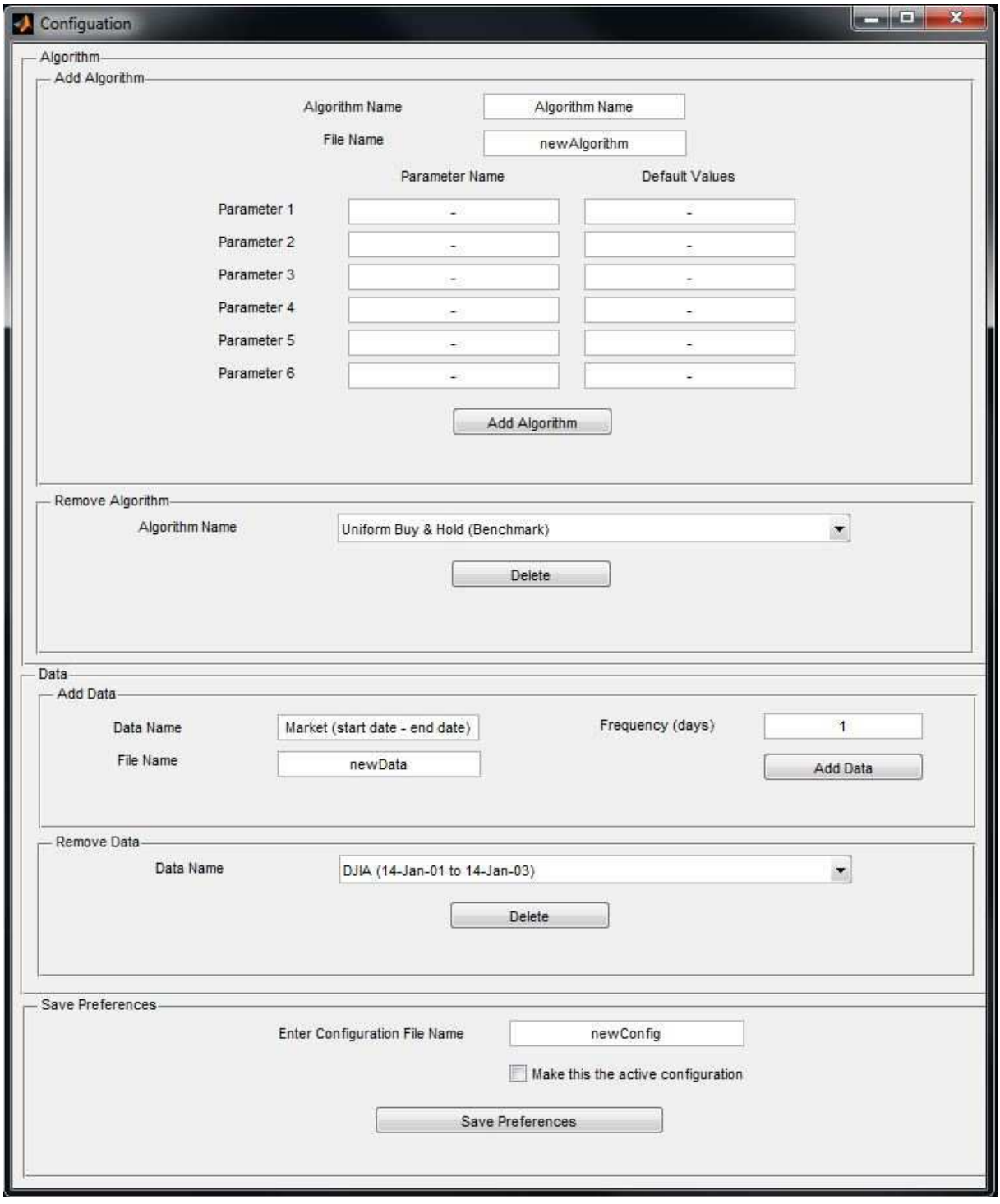


Figure 6: Configuration Manager.

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[4]. 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
5. Exit
************************************************************

Please enter your choice (1-5):
```

The root menu has five choices. The **4**(. About) and **5**(. Exit) choices are self-explanatory. The other three are the main functional buttons. The **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 **2**(. Experimenter) is used for selecting multiple algorithms and comparing their performances. The **3**(. Configuration) is used to add or delete algorithms and datasets that can be used by the toolbox.

2.2.2 ALGORITHM ANALYSER

On pressing **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 (1-5):1

*****************************
** OLPS: ALGORITHM ANALYSER **
```

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

```
*****************************
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):
```

Pressing **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):1
_____________________________________________
-- 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.M0
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 (1-23): 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.

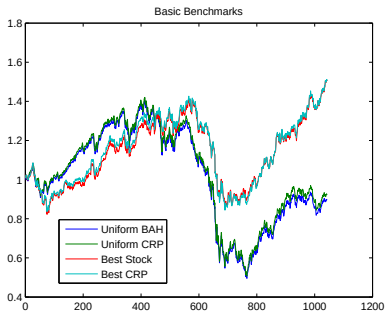
```
*****************************
** OLPS: ALGORITHM ANALYSER **
*****************************
1. Select Algorithm
2. Set Parameters
3. Select Dataset
4. Preliminary Visualizations
5. View Current Job
6. START EXECUTION
7. Back
*****************************

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.

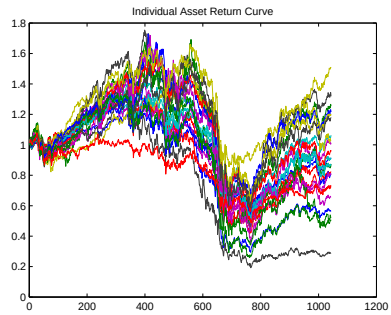
```
Please enter your choice (1-7):3


_____________________________________________
-- 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

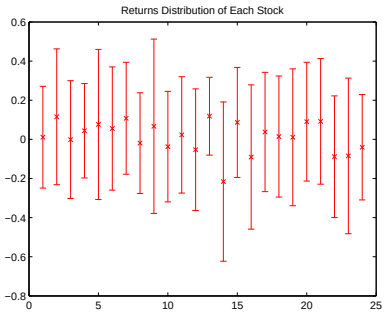
(i.e., Uniform BAH, Uniform CRP, Best Stock, and Best CRP.), and the wealth curves for all individual assets, and the return distribution of each asset (i.e., the annualized mean return and standard deviation of each asset). Figure 7 shows the example of MSCI dataset.



(a) Basic Benchmarks



(b) Individual Asset Return Curve



(c) Returns Distribution of Each Stock

Figure 7: Preliminary Visualizations of the MSCI dataset.

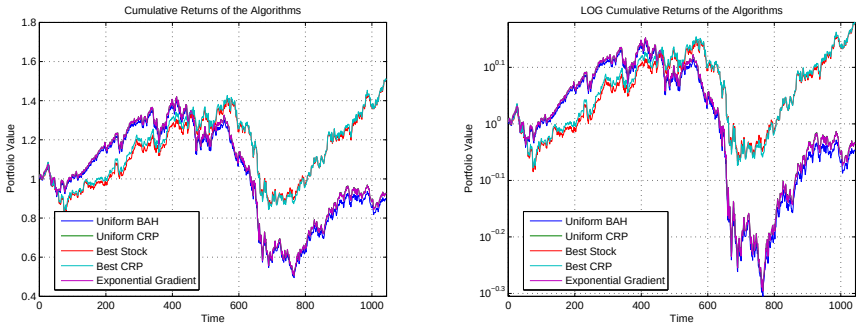
Till now, we have selected EG algorithm and MSCI dataset, and set its parameters. Our selections can be viewed by pressing **5**(. View Current Job) and **Enter**. Take our case for example,

```
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 **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.

```
**************************
Please enter your choice (1-6):1
Performance of the Algorithm compared to baselines based on several metrics
    ''                      'Market'        'Uniform'       'BestStock'     '
    'Final Value'           [    0.8986]    [    0.9268]    [    1.5040]    [
            [1x28 char]     [1.7961e-05]    [5.3001e-05]    [4.7016e-04]    [
    'Annualised Return'     [   -0.0255]    [   -0.0182]    [    0.1036]    [
    'Standard Deviation'    [    0.0155]    [    0.0158]    [    0.0125]    [
            [1x29 char]     [    0.2461]    [    0.2516]    [    0.1988]    [
    'Sharpe Ratio'          [   -0.2662]    [   -0.2313]    [    0.3201]    [
    'Calmar Ratio'          [   -0.0394]    [   -0.0312]    [    0.2585]    [
    'Sortino Ratio'         [    0.0240]    [    0.0694]    [    0.7368]    [
    'Value at Risk'         [    0.0236]    [    0.0245]    [    0.0210]    [
    'Maximum Draw Down'     [    0.6475]    [    0.6436]    [    0.3935]    [


**************************
** 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):
```

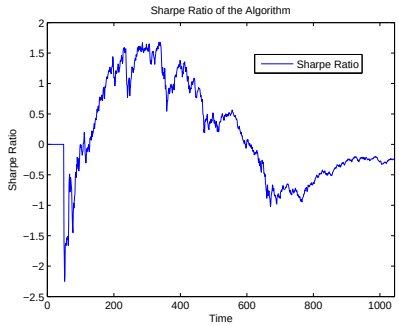
After execution, the system will automatically enter the result manager, which will show various forms of results. Choosing **1**(. View Tables of Results) will compare the results in a table. Choosing **2**(. View Returns Graph) will output the the wealth curves (original and logarithmic) of the selected algorithm and the benchmarks, for example, Figure 8.



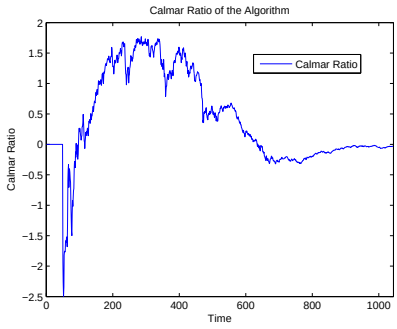
(a) Cumulative Returns of the Algorithms (b) LOG Cumulative Returns of the Algorithms

Figure 8: View Returns Graph.

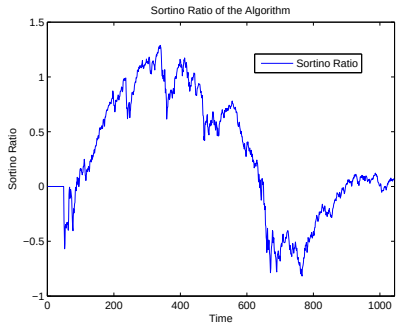
Choosing **3**(. View Risk Analysis Plots) will output the the risk analysis figures of the selected algorithm. For example, Figure 9 shows the Sharpe ratio, Calmar Ratio, Sortino Ratio, Value at Risk, and Maximum Drawdown of the algorithm, respectively.



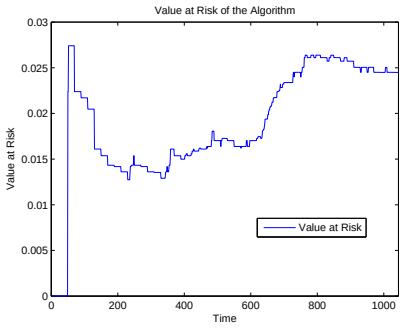
(a) Sharpe Ratio of the Algorithm



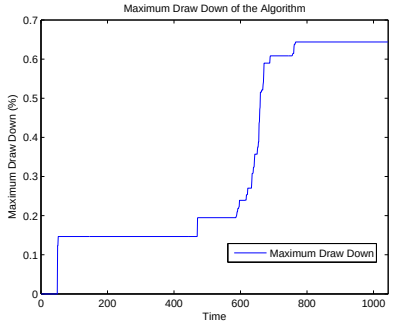
(b) Calmar Ratio of the Algorithm



(c) Sortino Ratio of the Algorithm



(d) Value at Risk of the Algorithm



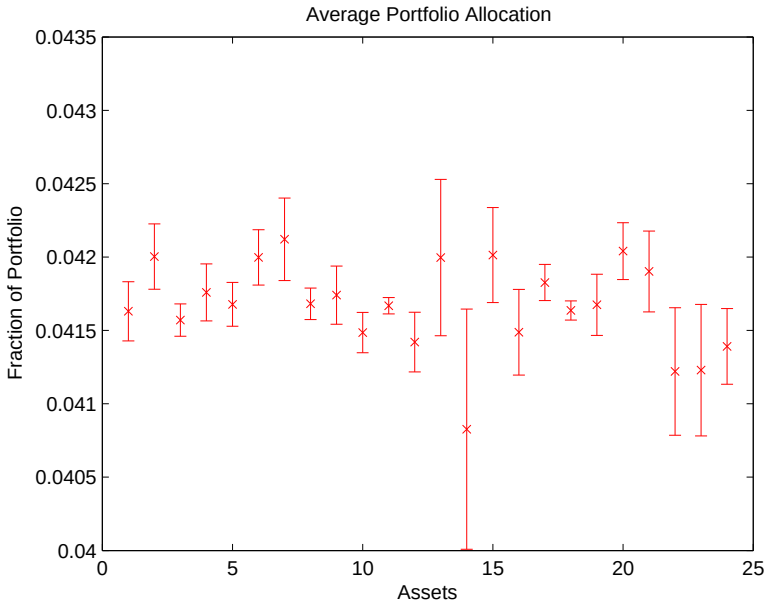
(e) Maximum Drawdown of the Algorithm

Figure 9: View Risk Analysis Plots.

Choosing **4**(. Portfolio Allocation Analysis) will output the figures on portfolio allocation, i.e., Figure 10 outputs the average allocation of each asset and its standard deviation.

Choosing **5**(. Save Results) will save the table of results to "\Log\Results\".

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



(a) Average Portfolio Allocation

Figure 10: Portfolio Allocation Analysis.

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 **2**(. Experimenter) in the root menu.

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

Please enter your choice (1-5):2

************************
** OLPS: EXPERIMENTER **
************************
1. Select Algorithms
2. Set Parameters
3. Select Dataset
```

```
4. View Current Job
5. START EXECUTION
6. Back
************************

Please enter your choice (1-5):
```

By selecting **1**(. Select Algorithms), the system will list existing algorithms and we can choose algorithms to compare. For the following example, EG (NO. 6) and PAMR (NO. 16) algorithms are selected.

```
_____________________________________________
-- 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.M0
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 **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
    ''                                 'eg'            'pamr'
    'Final Value'                      [    0.9260]    [15.2320]
    'Mean Return for every period'     [5.1829e-05]    [ 0.0029]
    'Annualised Return'                [   -0.0184]    [ 0.9309]
    'Standard Deviation'               [    0.0158]    [ 0.0230]
    'Annualised Standard Deviation'    [    0.2512]    [ 0.3648]
    'Sharpe Ratio'                     [   -0.2324]    [ 2.4424]
    'Calmar Ratio'                     [   -0.0315]    [ 1.6773]
    'Sortino Ratio'                    [    0.0681]    [ 2.5379]
    'Value at Risk'                    [    0.0245]    [ 0.0352]
    'Maximum Draw Down'                [    0.6438]    [ 0.5528]
```

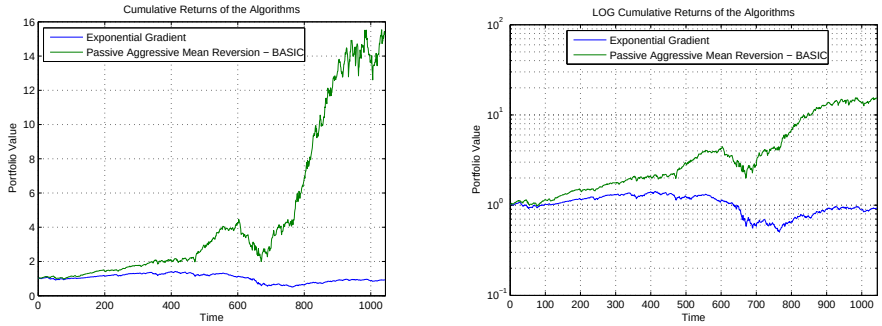
The *Result Manager 2* can show the table of results as above, or we can press **1**(. View Table of Results).

```
*****************************
** OLPS: RESULTS MANAGER 2 **
*****************************
1. View Table of Results
2. View Returns Graph
3. View Daily Returns
4. View Risk Analysis Plots
5. Portfolio Allocation Analysis
6. Save Results
7. Back
*****************************
Please enter your choice (1-7):
```

Choosing **2**(. View Returns Graph) will show two wealth curves illustrating the selected algorithms, as shown in Figure 11.

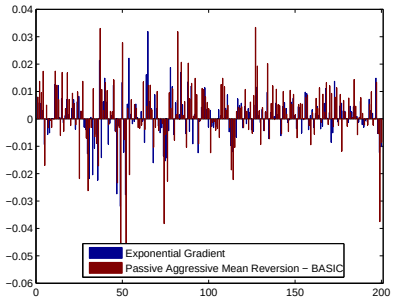
Choosing **3**(. View Daily Returns) will shows the daily returns. Sometimes we have a long period, thus we need to choose the start and end of time period, as follows:

```
Please enter your choice (1-7):3
Enter start of time period: 1
Enter end of time period: 200
```



(a) Cumulative Returns of the Algorithms (b) LOG Cumulative Returns of the Algorithms

Figure 11: View Returns Graph.



(a) Daily Returns of the Algorithms

Figure 12: View Daily Returns.

And a figure similar to Figure 12 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 13.

Choosing **5**(. Portfolio Allocation Analysis) will output the figures on portfolio allocation, i.e., Figure 14 outputs the average allocation of each asset and its standard deviation.

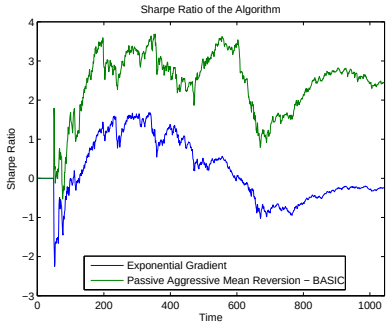
Choosing **6**(. Save Results) will save the table of results to "\Log\Results\".

Choosing **7**(. Back) will return to Algorithm Analyser.

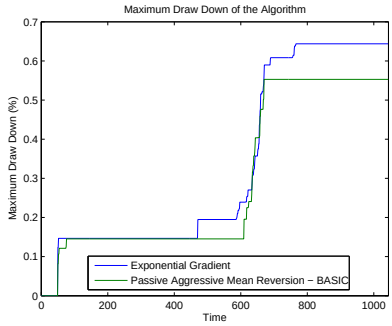
2.2.4 CONFIGURATION MANAGER

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

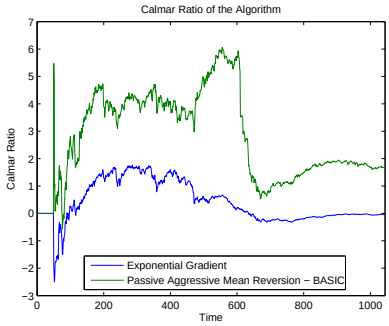
```
************************************************************
** OLPS: Online Portfolio Selection via Machine Learning **
************************************************************
1. Algorithm Analyser
2. Experimenter
3. Configuration
```



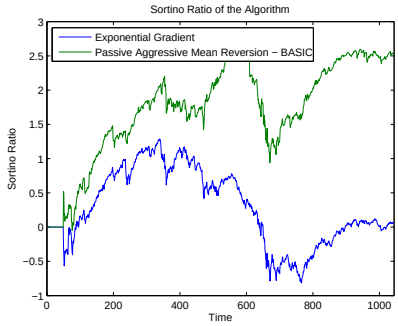
(a) Sharpe Ratio of the Algorithm



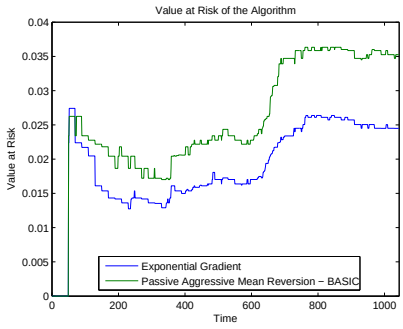
(b) Calmar Ratio of the Algorithm



(c) Sortino Ratio of the Algorithm



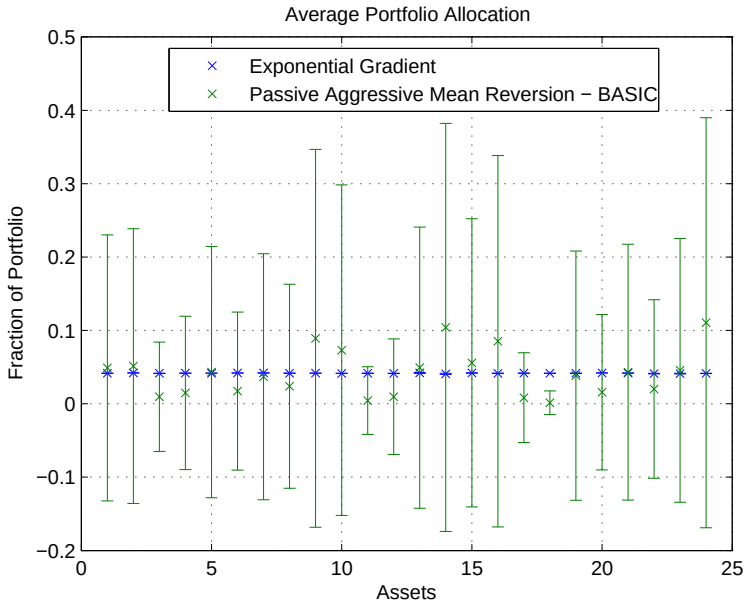
(d) Value at Risk of the Algorithm



(e) Maximum Drawdown of the Algorithm

Figure 13: View Risk Analysis Plots.

```
4. About
5. Exit
************************************************************

Please enter your choice (1-5):3
**************************
** OLPS: CONFIG MANAGER **
**************************
1. Add Strategy
```



(a) Average Portfolio Allocation

Figure 14: Portfolio Allocation Analysis.

```
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.

```
************************
** 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):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'
Enter name of configuration you want to change (e.g. 'config_myconfig'):'config'
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 **
*************************
1. Add Strategy
2. Remove Strategy
3. Add Dataset
4. Remove Dataset
5. Set Active Configuration
6. Back
*************************
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.M0
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.
```

New dataset 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 frquency (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)
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.New TSE data
Enter ID of Data to be removed:
```

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):
```

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 "OLPS_cli.m", this will help users understand the usage of the CLI mode.

2.3.1 TRADING MANAGER

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;
varargins: 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_et: analyzed results, including risk adjusted returns;
run_time: Time for the strategy (in sec).

```
1 begin
2     Initialize market data from dataset;
3     Open the log file and mat file;
4     Start the time variables;
5     Call strategy with parameters in varargins;
6     Terminate the time variables;
7     Analyze the results;
8     Close the log file and mat file;
9 end
```

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

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

- 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.

- dataset_name: the name of dataset;
- varargins: 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.

```
[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.

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

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

```
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:

```
>> 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:

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

Index	Descriptions
1	Number of periods
2	Strategy's average period return
3	Market's average period return
4	Strategy's winning ratio over the market
5	Alpha (α)
6	Beta (β)
7	*t*-statistics
8	*p*-value
9	Annualized percentage yield
10	Annualized standard deviation
11	Sharpe ratio
12	Drawdown at the end
13	Maximum drawdown during the periods
14	Calmar ratio

Table 4: Vector of the analyzed results.

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

```
>> 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:

```
>> 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.
Start Time: 2013-0721-13-22-05-664.
----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
-------------------------------------
```

```
Statistical Test
Size: 1276
MER(Strategy): 0.0015
MER(Market):0.0003
WinRatio:0.5063
Alpha:0.0010
Beta:1.3216
t-statistics:2.1408
p-Value:0.0162
-------------------------------------
Risk Adjusted Return
Volatility Risk analysis
APY: 0.3240
Volatility Risk: 0.4236
Sharpe Ratio: 0.6705
Drawdown analysis
APY: 0.3240
DD: 0.3103
MDD: 0.5066
CR: 0.6395
-------------------------------------
>>
```

2.3.3 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:
10% The sections labelled as "Static" of the file need not be changed
11
12%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
13% This file is part of OLPS: http://OLPS.stevenhoi.org/
14% Original authors:
15% Contributors:
16% Change log:
17%
18%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
19
```

```
20    %% Make changes to this section to construct your algorithm
21
22    %% Read Parameters
23    p1 = varargins{1};
24
25    %% Initialize variables
26    [r c] = size(data);
27    b = ones(c,1)/c;
28    returns = zeros(r,1);
29    portfolio = ones(r,c)/c;
30
31    %% Static
32    progress = waitbar(0,'Executing Algorithm...');
33
34
35    %% The looping over the entire dataset for backtesting
36    % The algorithm looping over r time periods
37    for t = 1:1:r
38
39
40 %%TO be filled by users %%
41
42
43        %% Static
44        % compute x ofthe optimization problem i.e. todaysRelative
45        todaysRelative = data(t,:)';
46        % Compute the returns of algorithm
47        portfolio(t, :) = b;
48        returns(t) = b'*(todaysRelative-1);
49
50        %% Change this section to describe your strategy portfolio selection method
51
52        % Use solver/algorithm to find new portfolio vector at end of time
53        % period t
54
55
56        %%  Static
57        % Update Progress
58        if mod(t, 50) == 0
59            waitbar((t/r));
60        end
61
62    end
63
64
65    %% Static
66
67    % Compute additional statistics (for quick individual run without GUI)
68    % Can be deleted
69    stats.finalValue = prod(returns+1);
70    Y =r/252;
71    stats.sharpe = ((stats.finalValue)^(1/Y) - 1.04)  / (std(returns)*sqrt(252));
72    stats.averageInTopStock = mean(max(portfolio') );
73    stats.averageInTop2Stocks = flipud(sort(portfolio'));
74    stats.averageInTop2Stocks = mean(sum(stats.averageInTop2Stocks(1:2, :)));
75    stats.variance = (std(returns)*sqrt(252));
76
77
78    % Conversion of results to algorithm format
79    % cum_ret, cumprod_ret, daily_ret, daily_portfolio
80    daily_portfolio = portfolio;
81    daily_ret = returns+ 1;
82    cumprod_ret = cumprod(daily_ret);
83    cum_ret = cumprod_ret(end);
```

```
84
85    close(progress);
86end
```

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
```

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,

$$S_n\left(BAH\left(\mathbf{b}_1\right)\right) = \mathbf{b}_1 \cdot \left(\bigodot_{t=1}^{n} \mathbf{x}_t\right),$$

where $\mathbf{b}_1$ denotes the initial portfolio. In case of UBAH, $\mathbf{b}_1 = \left(\frac{1}{m}, \ldots, \frac{1}{m}\right)$. To see its update clearly, BAH's explicit portfolio update can also be written as,

$$\mathbf{b}_{t+1} = \frac{\mathbf{b}_t \bigodot \mathbf{x}_t}{\mathbf{b}_t^\top \mathbf{x}_t}, \tag{1}$$

where $\bigodot$ denotes the operation of element-wise product.

Usage

```
ubah(fid, data, {λ}, opts);
```

- fid: file handle for writing log file;
- data: market sequence matrix;
- $\lambda \in [0, 1)$: proportional transaction cost rate;
- 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);
```

3.1.2 BEST STOCK

Description "Best Stock" (Best) is a special BAH strategy that buys the best stock in hindsight. The final cumulative wealth achieved by the Best strategy can be calculated as,

$$S_n\left(Best\right) = \max_{\mathbf{b} \in \Delta_m} \mathbf{b} \cdot \left(\bigodot_{t=1}^{n} \mathbf{x}_t \right) = S_n\left(BAH\left(\mathbf{b}^\circ\right)\right),$$

where the initial portfolio $\mathbf{b}^\circ$ can be calculated as,

$$\mathbf{b}^\circ = \arg\max_{\mathbf{b} \in \Delta_m} \mathbf{b} \cdot \left(\bigodot_{t=1}^{n} \mathbf{x}_t \right).$$

Its portfolio update can also be explicitly written as the same as Eq. (1), except that the initial portfolio equals $\mathbf{b}^\circ$.

Usage

```
best(fid, data, {λ}, 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 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 $\mathbf{b}_1^n = \{\mathbf{b}, \mathbf{b}, \dots\}$. The final cumulative portfolio wealth achieved by a CRP strategy after n periods is defined as,

$$S_n\left(CRP\left(\mathbf{b}\right)\right) = \prod_{t=1}^{n} \mathbf{b}^\top \mathbf{x}_t.$$

In particular, "Uniform CRP" (UCRP) chooses a uniform portfolio as the preset portfolio, that is, $\mathbf{b} = \left(\frac{1}{m}, \dots, \frac{1}{m}\right)$.

Usage

```
ucrp(fid, data, {λ}, 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,

$$S_n\left(BCRP\right) = \max_{\mathbf{b} \in \Delta_m} S_n\left(CRP\left(\mathbf{b}\right)\right) = S_n\left(CRP\left(\mathbf{b}^\star\right)\right),$$

and its portfolio is calculated in hindsight as,

$$\mathbf{b}^\star = \arg\max_{\mathbf{b}^n \in \Delta_m} \log S_n\left(CRP\left(\mathbf{b}\right)\right) = \arg\max_{\mathbf{b} \in \Delta_m} \sum_{t=1}^{n} \log\left(\mathbf{b}^\top \mathbf{x}_t\right).$$

Usage

```
bcrp(fid, data, {λ}, 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 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) (Cover, 1991) uniformly buys and holds the whole set of CRP experts within the simplex domain. Its cumulative wealth is calculated as,

$$S_n(UP) = \int_{\Delta_m} S_n(\mathbf{b})\, d\mu(\mathbf{b}).$$

Moreover, we adopt a implementation (**?**), which is based on non-uniform random walks that are rapidly mixing and requires a polynomial time.

Usage

```
up(fid, data, {λ}, 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 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) (**?**) tracks the best stock and adopts regularization term to constrain the deviation from previous portfolio, i.e., EG's formulation is,

$$\mathbf{b}_{t+1} = \underset{\mathbf{b}\in\Delta_m}{\arg\max} \quad \eta \log \mathbf{b}\cdot\mathbf{x}_t - R\left(\mathbf{b},\mathbf{b}_t\right),$$

where η refers to the learning rate and $R\left(\mathbf{b},\mathbf{b}_t\right)$ denotes relative entropy, or $R\left(\mathbf{b},\mathbf{b}_t\right) = \sum_{i=1}^{m} b_i \log \frac{b_i}{b_{t,i}}$. Solving the optimization, we can obtain EG's portfolio explicit update,

$$b_{t+1,i} = b_{t,i} \exp\left(\eta \frac{x_{t,i}}{\mathbf{b}_t\cdot\mathbf{x}_t}\right)/Z, \quad i = 1,\ldots,m,$$

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

Usage

```
eg(fid, data, {η, λ}, opts);
```

- fid: file handle for writing log file;
- data: market sequence matrix;
- η: Learning rate;
- λ: Transaction costs rate;
- 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.

```
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 a L2-norm regularization to constrain portfolio's variability. In particular, its formulation is,

$$\mathbf{b}_{t+1} = \underset{\mathbf{b}\in\Delta_m}{\arg\max} \sum_{\tau=1}^{t} \log\left(\mathbf{b}\cdot\mathbf{x}_\tau\right) - \frac{\beta}{2}\left\|\mathbf{b}\right\|.$$

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

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

with

$$\mathbf{A}_t = \sum_{\tau=1}^{t}\left(\frac{\mathbf{x}_\tau\mathbf{x}_\tau^\top}{\left(\mathbf{b}_\tau\cdot\mathbf{x}_\tau\right)^2}\right) + \mathbf{I}_m, \quad \mathbf{p}_t = \left(1+\frac{1}{\beta}\right)\sum_{\tau=1}^{t}\frac{\mathbf{x}_\tau}{\mathbf{b}_\tau\cdot\mathbf{x}_\tau},$$

where β is the trade-off parameter, δ is a scaling term, and $\Pi_{\Delta_m}^{\mathbf{A}_t}\left(\cdot\right)$ 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, $\mathbf{y}_1 = \log\left(\mathbf{x}_{t-2w+1}^{t-w}\right)$ and $\mathbf{y}_2 = \log\left(\mathbf{x}_{t-w+1}^{t}\right)$. It then calculates the cross-correlation matrix between $\mathbf{y}_1$ and $\mathbf{y}_2$,

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

$$M_{cor}\left(i,j\right) = \begin{cases} \frac{M_{cov}(i,j)}{\sigma_1(i) * \sigma_2(j)} & \sigma_1\left(i\right), \sigma_2\left(j\right) \neq 0 \\ 0 & \text{otherwise} \end{cases}$$

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 ($M_{cor}\left(i,j\right)$) minus their negative auto correlation values ($\min\left\{0, M_{cor}\left(i,i\right)\right\}$ and $\min\left\{0, M_{cor}\left(j,j\right)\right\}$). These transfer claims are finally normalized to keep the portfolio in the simplex domain.

Usage We implemented two Anticor algorithms, i.e., BAH_W(Anticor) and BAH_W(Anticor(Anticor)). Their usages are listed below.

```
anticor(fid, data, {W, λ}, opts);
anticor_anticor(fid, data, {W, λ}, opts);
```

- fid: file handle for writing log file;
- data: market sequence matrix;
- W: Maximal window size;
- λ: 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,

$$\mathbf{b}_{t+1} = \arg\min_{\mathbf{b}\in\Delta_m} \frac{1}{2}\left\|\mathbf{b}-\mathbf{b}_t\right\|^2 \quad \text{s.t.} \quad \ell_\epsilon\left(\mathbf{b};\mathbf{x}_t\right)=0,$$

where $\ell_\epsilon\left(\mathbf{b};\mathbf{x}_t\right)$ denotes a predefined loss function to capture the mean reversion property,

$$\ell_\epsilon\left(\mathbf{b};\mathbf{x}_t\right) = \begin{cases} 0 & \mathbf{b}\cdot\mathbf{x}_t \le \epsilon \\ \mathbf{b}\cdot\mathbf{x}_t - \epsilon & \text{otherwise} \end{cases}.$$

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

$$\mathbf{b}_{t+1} = \mathbf{b}_t - \tau_t\left(\mathbf{x}_t - \bar{x}_t\mathbf{1}\right), \quad \tau_t = \max\left\{0, \frac{\mathbf{b}_t\cdot\mathbf{x}_t-\epsilon}{\left\|\mathbf{x}_t-\bar{x}_t\mathbf{1}\right\|^2}\right\}.$$

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

```
pamr(fid, data, {ϵ, λ}, opts);
pamr_1(fid, data, {ϵ, C, λ}, opts);
pamr_2(fid, data, {ϵ, C, λ}, opts);
```

- fid: file handle for writing log file;
- data: market sequence matrix;
- ϵ: mean reversion threshold;
- C: aggressive parameter;
- λ: 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,

$$\begin{aligned}(\mu_{t+1}, \mathbf{\Sigma}_{t+1}) = \underset{\mu \in \Delta_m, \mathbf{\Sigma}}{\arg\min} \quad & \mathrm{D}_{\mathrm{KL}}\left(\mathcal{N}\left(\mu, \mathbf{\Sigma}\right) \| \mathcal{N}\left(\mu_t, \mathbf{\Sigma}_t\right)\right) \\ \text{s.t.} \quad & \Pr\left[\mu \cdot \mathbf{x}_t \leq \epsilon\right] \geq \theta.\end{aligned}$$

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,

$$\mu_{t+1} = \mu_t - \lambda_{t+1} \mathbf{\Sigma}_t \left(\mathbf{x}_t - \bar{x}_t \mathbf{1}\right), \quad \mathbf{\Sigma}_{t+1}^{-1} = \mathbf{\Sigma}_t^{-1} + 2\lambda_{t+1}\phi \mathbf{x}_t \mathbf{x}_t^\top,$$

where λ_{t+1} corresponds to the Lagrangian multiplier calculated by Eq. (11) in Li et al. (2013) and $\bar{x}_t = \frac{\mathbf{1}^\top \mathbf{\Sigma}_t \mathbf{x}_t}{\mathbf{1}^\top \mathbf{\Sigma}_t \mathbf{1}}$ 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, {φ, ε, λ}, opts);
cwmr_stdev(fid, data, {φ, ε, λ}, opts);
```

- fid: file handle for writing log file;
- data: market sequence matrix;
- ϕ: confidence parameter;
- ϵ: mean reversion threshold;
- λ: 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) (**?**) explicitly predicts next price relatives following the mean reversion idea, i.e., MAR-1 borrows simple moving average,

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

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

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

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

$$\mathbf{b}_{t+1} = \underset{\mathbf{b}\in\Delta_m}{\arg\min} \quad \frac{1}{2}\|\mathbf{b}-\mathbf{b}_t\|^2 \text{ s.t. } \mathbf{b}\cdot\tilde{\mathbf{x}}_{t+1} \geq \epsilon.$$

Solving the optimization, we can obtain its portfolio update,

$$\mathbf{b}_{t+1} = \mathbf{b}_t + \lambda_{t+1}\left(\tilde{\mathbf{x}}_{t+1} - \bar{x}_{t+1}\mathbf{1}\right),$$

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

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

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

```
olmar1(fid, data, {ε, W, λ}, opts);
olmar2(fid, data, {ε, α, λ}, opts);
```

- fid: file handle for writing log file;
- data: market sequence matrix;
- ϵ: mean reversion threshold;
- W: window size for simple moving average;
- $\alpha \in [0,1]$: decaying factor to calculate exponential moving average;
- $\lambda \in [0,1)$: transaction cost rates;
- 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.

```
1: >> manager('olmar1', 'nyse-o', {10, 5, 0}, opts);
2: >> manager('olmar2', 'nyse-o', {10, 0.5, 0}, opts);
```

Algorithm 3: Sample selection framework ($C\left(\mathbf{x}_1^t, w\right)$).

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

```
Initialize C = ∅;
if t ≤ w + 1 then
    return;
end
for i = w + 1, w + 2, ..., t do
    if x_{i-w}^{i-1} is similar to x_{t-w+1}^{t} then
        C = C ∪ {i};
    end
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,

$$\mathbf{b}_{t+1} = \underset{\mathbf{b}\in\Delta_m}{\arg\max} \prod_{i\in C\left(\mathbf{x}_1^t\right)} \mathbf{b}\cdot\mathbf{x}_i. \tag{2}$$

3.4.1 NONPARAMETRIC KERNEL-BASED LOG-OPTIMAL STRATEGY

Description "Nonparametric *kernel-based* sample selection" ($\mathrm{B^K}$) (Györfi et al., 2006) identifies the similarity set by comparing two market windows via Euclidean distance,

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

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

Usage

```
bk_run(fid, data, {K, L, c, λ}, 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 $\mathrm{B^K}$ algorithm on the "NYSE (O)" dataset with default parameters and a transaction cost rate of 0.

```
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" ($\mathrm{B^{NN}}$) (Györfi et al., 2008) searches the price relatives whose preceding market windows are within the ℓ nearest neighbor of latest market window in terms of Euclidean distance,

$$C_N\left(\mathbf{x}_1^t, w\right) = \left\{w < i < t+1 : \mathbf{x}_{i-w}^{i-1} \text{ is among the } \ell \text{ NNs of } \mathbf{x}_{t-w+1}^t\right\},$$

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

Usage

```
bnn(fid, data, {K, L, λ}, 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 $\mathrm{B^{NN}}$ algorithm on the "NYSE (O)" dataset with default parameters and a transaction cost rate of 0.

```
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,

$$C_C\left(\mathbf{x}_1^t, w\right) = \left\{w < i < t+1 : \frac{cov\left(\mathbf{x}_{i-w}^{i-1}, \mathbf{x}_{t-w+1}^t\right)}{std\left(\mathbf{x}_{i-w}^{i-1}\right) std\left(\mathbf{x}_{t-w+1}^t\right)} \geq \rho\right\},$$

where ρ 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. 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 this toolbox can facilitate the further research in this topic.

References

Amit Agarwal, Elad Hazan, Satyen Kale, and Robert E. Schapire. Algorithms for portfolio management based on the newton method. In *Proceedings of International Conference on Machine Learning*, 2006.

Allan Borodin, Ran El-Yaniv, and Vincent Gogan. Can we learn to beat the best stock. *Journal of Artificial Intelligence Research*, 21:579–594, 2004.

Thomas M. Cover. Universal portfolios. *Mathematical Finance*, 1(1):1–29, 1991.

László Györfi, Gábor Lugosi, and Frederic Udina. Nonparametric kernel-based sequential investment strategies. *Mathematical Finance*, 16(2):337–357, 2006.

László Györfi, Frederic Udina, and Harro Walk. Nonparametric nearest neighbor based empirical portfolio selection strategies. *Statistics and Decisions*, 26(2):145–157, 2008.

László Györfi, Gy. Ottucsák, and Harro Walk. *Machine Learning for Financial Engineering*. Singapore : World Scientific, 2012.

Nils H. Hakansson. Capital growth and the mean-variance approach to portfolio selection. *The Journal of Financial and Quantitative Analysis*, 6(1):517–557, 1971.

David P. Helmbold, Robert E. Schapire, Yoram Singer, and Manfred K. Warmuth. On-line portfolio selection using multiplicative updates. *Mathematical Finance*, 8(4):325–347, 1998.

Jr. Kelly, J. A new interpretation of information rate. *Bell Systems Technical Journal*, 35:917–926, 1956.

Bin Li and Steven C. H. Hoi. Online portfolio selection: A survey. *ACM Computing Surveys*, 46 (3):35, 2014.

Bin Li, Steven C.H. Hoi, and Vivekanand Gopalkrishnan. Corn: Correlation-driven nonparametric learning approach for portfolio selection. *ACM Transactions on Intelligent Systems and Technology*, 2(3):21:1–21:29, 2011.

Bin Li, Peilin Zhao, Steven Hoi, and Vivekanand Gopalkrishnan. Pamr: Passive aggressive mean reversion strategy for portfolio selection. *Machine Learning*, 87(2):221–258, 2012.

Bin Li, Steven C.H. Hoi, Peilin Zhao, and Viveknand Gopalkrishnan. Confidence weighted mean reversion strategy for online portfolio selection. *ACM Transactions on Knowledge Discovery from Data*, 7(1):4:1 – 4:38, 2013.

Harry Markowitz. Portfolio selection. *The Journal of Finance*, 7(1):77–91, 1952.

William F. Sharpe. Capital asset prices: A theory of market equilibrium under conditions of risk. *The Journal of Finance*, 19(3):425–442, 1964.

E. O. Thorp. Portfolio choice and the kelly criterion. In *Business and Economics Section of the American Statistical Association*, 1971.