MIE377: Financial Optimization Models

Project 1 (Winter 2024)

Due Date: 18–Mar–2024 by 11:59 PM

Please use MATLAB or Python to solve this project. This is a group project, with groups of 2 or 3 students per group. Each group is expected to submit their own original work.

You are given:

- A dataset consisting of adjusted closing prices for 20 U.S. stocks [1], and factor returns corresponding to 8 different factors [2].
- A project template written in MATLAB containing
  - A main program template 'MIE377_Project1_Main.m' (do not modify this template)
  - A MATLAB function template 'Project1_Function.m' (use this template to code your project)
  - Two MATLAB functions ('MVO.m' and 'OLS.m') with some examples of functions you can use to supplement your 'Project1_Function.m' deliverable.
- A second project template written in Python containing
  - A main program template 'MIE377_Project1_Main.ipynb' (do not modify this template)
  - A Python file 'services\project_function.py' (use this template to code your project)
  - Two Python files ('services\optimization.py' and 'services\estimators.py') with some examples of functions you can use to supplement your 'services\project_function.py' deliverable.
  - It is recommended that the Anaconda distribution for Python be used.
    - An environment.yml file is provided that can be installed via the command `conda env create -f environment.yml` . The environment is set-up to work on Windows. You can create your own environment on Mac/Linux but please ensure your final code runs error-free in the provided environment.
    - The environment includes the python package `cvxpy` a general purpose package used to solve convex optimization problems
    - BEFORE running the environment set up command, please follow the `cvxpy` instructions for your OS to ensure your machine is ready for `cvxpy` to be installed.
    - Useful packages such as `sklearn` (machine learning estimators) and `Gurobi` (mixed integer optimization solver) are also included in the environment.
    - DO NOT modify the environment unless the TAs approve the modification. The TAs will not install new packages to run your code unless they are approved.
- Note: You are allowed to create your own additional functions if needed.
- (Optional) A LaTeXtemplate for you to write your report. Anyone interested in learning how to use LaTeXis welcome to use this template. While no penalties or rewards will be given for using LaTeX, it is a valuable skill that makes mathematical communication easier. This youtube provides a great introduction to LaTeX.

You should hand in:

- A formal model development report.
- The MATLAB/Python program and functions you wrote to solve this project.

⇒ Please submit all your files inside a compressed folder, including your report and MATLAB/Python code. The name of the submitted file should be **FirstnameLastname.zip**. Only one submission per team is required. Please submit this file electronically through Quercus.

- Each member in your team must submit a team assessment form (see Section 4 for details)

---

1 Introduction

The purpose of this project is to design an automated asset management system, i.e., an algorithmic trading system. In particular, this algorithm should leverage some of the concepts, methods and optimization models seen throughout the course.

As part of the project, you are given a dataset of asset prices and factor returns that you may use to train, validate and test your algorithm. You should develop your own investment strategy based on a model or ensemble of models that you believe will have the best and most consistent performance under different market environments (e.g., bull and bear market environments).

This project has two main deliverables: a formal report detailing your model development, and a MATLAB/Python program (and functions) containing your trading algorithm. As previously mentioned, your algorithm can be trained, validated and tested using the sample data provided. After submission, your algorithms will be tested on two unseen historical datasets to test their performance. The algorithms will be assessed on their out-of-sample financial performance and their computational runtime. From a financial perspective, they will be assessed by their out-of-sample Sharpe ratio and their average turnover rate. This is discussed in greater detail in Section 3.

2 Investment competition rules

This project has some rules to narrow down its scope. These rules pertain to the input data frequency and type, the in-sample calibration and out-of-sample investment windows, and the required output. The rules are the following.

- The training data will consist of 20 U.S. stocks and 8 factors (see Tables 1 and 2). The frequency of observations is monthly, ranging from Jan–2002 to Dec–2016. The factor data also includes a 9th feature corresponding to the risk-free rate.
- Your trading algorithm will be assessed using two previously unseen datasets. However, these datasets will be similar in format to the original dataset provided. The data will consist of monthly asset prices and factor returns, and the factors will be the same as those of the original dataset.
- For all datasets, the first five years are reserved for calibration and estimation, and will not be used for the out-of-sample analysis. In other words, the initial 5-year period in the dataset will not be used to assess financial performance.
  - This means your first in-sample calibration period can be at most five years long. However, any subsequent calibration period is allowed to use any new available data.
  - Alternatively, note that you do not need to use the full five-year period for parameters estimation if you do not wish to (e.g., you could choose to use the most recent 2-year period for estimation instead of the full five years).
- For all datasets, the asset will consist of stocks and exchange-traded funds (ETFs). The unseen datasets could consist of anywhere between 15 to 40 assets each.
- Portfolios will be rebalanced every six months.
- Finally, you are allowed to modify the 'Project1_Function.m' ('project_function.py') file as much as you wish. Additionally, you can prepare and submit any other functions to accompany your 'Project1_Function.m' ('project_function.py') file. For example, you have been given the 'MVO.m' ('optimization.py') and 'OLS.m' ('estimation.py') files (which you can also modify) and you can create any additional functions.
  - However, the input and output of 'Project1_Function.m' (project_function in 'project_function.py') must remain consistent.
  - Moreover, you are not allowed to modify the 'MIE377_Project1_Main.m (.pynb)' program since this template is designed to assess all teams.

– Note: Your program should not print output to the console. In other words, the program should not print hundreds of lines to the console for no reason. This is to make the already-coded output easy to read. The only exception will be warning messages intrinsic to MATLAB/Python that you are unable to suppress.

Table 1: List of assets by ticker

F	CAT	DIS	MCD	KO	PEP	WMT	C	WFC	JPM
AAPL	IBM	PFE	JNJ	XOM	MRO	ED	T	VZ	NEM

Table 2: List of factors

Market ('Mkt_RF')	Size ('SMB')	Value ('HML')	Short-term reversal ('ST_Rev')
Profitability ('RMW')	Investment ('CMA')	Momentum ('Mom')	Long-term reversal ('LT_Rev')

3 Assessment criteria

3.1 Competition

The competition will test your trading algorithms using two previously unseen data sets. These new datasets will have the same format as the original dataset provided.

The assessment over these two datasets will be referred to as 'Trial 1' and 'Trial 2'. The trials may vary in number of assets and investment period, but will consistently rebalance the portfolios every six months. The assessment criteria is shown below in order of importance.

1. Ex-post Sharpe ratio over the entire investment horizon (higher is better). The Sharpe ratio is worth 80% of the score per trial.

2. Average turnover rate (lower is better). The average turnover rate is worth 20% of the score per trial.

   - Note: the 'turnover rate' of the initial portfolio is not included in the score. In other words, creating your initial portfolio is 'free'.

3. Computational runtime (lower is better). The runtime will only become an issue if your trading algorithm takes more than (approximately) 5 minutes to run during each trial. For example, an algorithm that takes 30 minutes to run will be penalized. However, algorithms that run sufficiently fast will not be penalized (e.g., from an evaluation perspective, there is no difference between an algorithm that runs in 10 seconds versus 2 minutes).

3.2 Grading

The main objective of this project is the model development process, which must be presented as a formal report that details all of your research and development.

Your model development process should incorporate at least some elements of the material seen in class (e.g., multi-factor models, portfolio optimization). It is likely that many of these elements will not make it into the final version of your trading algorithm (e.g., you may test different factor models but only choose the one that you deem most beneficial to your trading algorithm).

However, even if some of these elements do not make it into the final version of your algorithm, you should still document them and show your analysis. In other words, you must justify your design choices

(e.g., you must discuss why a specific optimization model was selected, or why was a competing factor model discarded). These types of details will be the most important during grading.

Finally, please note that the majority of your grade is not dependent on your algorithm's performance during the competition. Instead, the majority of your grade depends on your submitted report detailing your model development and testing, as well as the structure and readability of your code. Thus, you can expect to do well in the project if you make a good effort during your model development. On the other hand, simply having a 'winning' algorithm without justification through a proper model development report does not guarantee you will earn a good grade.

4 Deliverables

4.1 Report (70%)

Prepare a formal report, including an introduction, methodology, model selection process and testing, and a conclusion outlining the strengths and weaknesses of your algorithm and why you chose it. The report should demonstrate your understanding of finance and optimization theory, and reflect your knowledge of the material we have seen in class.

The report is worth 70% of the total. The distribution is the following

- Formal report structure and presentation: 10%
- Methodology: 20%
- Analysis from training, validation and testing: 25%
- Discussion and conclusion: 15%

4.2 MATLAB/Python program (30%)

Prepare a MATLAB/Python program and functions to perform the computational experiment. Use the templates provided. You are allowed modify these templates as much as you see fit. In addition, you are allowed (and encouraged) to create your own functions if needed. However, please do not modify the 'MIE377_Project1_Main.m (.ipynb)' program.

Be sure to properly comment on your code to briefly explain what you are doing. Your code should be easy to read and the TA should be able to run it. The TA will not debug your code and should not have to search for the results within the code.

The MATLAB/Python program is worth 30% of the total, and it includes the assessment of the performance of your algorithm. The distribution is the following

- Properly structured and commented code: 10%
- Score from 'Trial 1': 10%
- Score from 'Trial 2': 10%

4.3 Team assessment

Finally, each member in your team must submit a team assessment form. These forms should be submitted via Quercus for each member. The form should provide a very brief assessment of the contributions of each team member, as well as a score from 0 (poor) to 10 (excellent) for each member.

References

[1] Quandl.com. *Wiki – Various End-Of-Day Stock Prices*. https://www.quandl.com/databases/WIKIP/usage/export. [Online; accessed 07-Nov-2017]. 2017. (Visited on 11/07/2017).

[2] K. R. French. *Data Library*. http://mba.tuck.dartmouth.edu/pages/faculty/ken.french/data_library.html. [Online; accessed 01-Feb-2020]. 2020. (Visited on 2020).