Paragon Global Investments
2024 Summer Assignment

To everyone who has completed the Education pod, I hope you have familiarized yourself with many of the ideas present in the lectures. Over the course of this summer, we have prepared an assignment for both Education pod members and new PMs hoping to join a university chapter. These projects are constructed to test the various qualities of successful quantitative researchers, such as abilities in research, coding, presentation, as well as dedication to pursuing the Quant industry. These projects are meant to be time-consuming, albeit fairly simple if you take full advantage of the resources provided to you.

We have provided two choices for your projects. Please choose the one you find most interesting. The project will be **due August 20, 2024 at 11:59pm EST** and will be evaluated by your respective Quant Education heads to determine candidacy for the Systematic Fund. Don't stress out too much about it, though, as evaluation primarily focuses on time and effort put into the project as opposed to results and accuracy. For both projects, office hours are available upon request. Please direct questions and requests to hw2810@columbia.edu (Harrison) for project 1 and nicholas_klatsky@brown.edu for project 2, but address your emails to both of us. In fact, it is encouraged that you reach out for office hours at any point where you are unsure about the project specifications, and definitely don't think of needing help on this project as a bad thing. This project is intended to be instructive as well as evaluative!

Please email your submission both hw2810@columbia.edu and nicholas_klatsky@brown.edu in the following format:

- Email header: **PGI Summer Project Submission**
- To be submitted as a zip-file
  - **Project writeup**
  - **Jupyter notebooks**
  - **Any external datasets you used (in .csv format)**

Important tips for both projects:

1. Organization of code is very important. Try to modularize your code, i.e. write individual functions to do individual things. The modules are outlined fairly well above, but feel free to further separate the code.
2. Keep in mind that as a quantitative analyst, your goal is to conduct research. This is one of the few fields where you are not attempting to complete a bunch of tasks with a clear end goal, but rather you must declare goals for yourself and summarize/answer the results to these questions to your mentors/bosses. Be sure to summarize and interpret all results, rather than leaving the numbers for your evaluators to interpret themselves.
3. These projects would be extremely beneficial to add to your resume, as they demonstrate a thorough understanding of the process, beginning-to-end, of either constructing a trading strategy or identifying signals in the market. In fact, this material is in all Master's in Finance/Financial Engineering algorithmic trading courses. You can use these projects as a point to jump off of in order to create profitable strategies in the future (if you can find a profitable strategy, many of the top quant firms would likely hire you).
4. Email hw2810@columbia.edu and nicholas_klatsky@brown.ed with questions!
5. Best of luck!

Project 1: Momentum Signals

The goal of this project is to create a momentum-based trading strategy for the universe of stocks in the DJI (30 blue chip stocks), and go through the entire process from signal generation to backtesting and hyperparameter optimization to portfolio evaluation. Note that the expectation of this assignment is not to create a strategy that can profit, but rather to complete a beginning-to-end alpha development project so that when you become Systematic Fund members, you can apply the process to new signals which will actually be profitable. Given that the signals are so simple, it is also very unlikely that you will find anything implementable today. The presentation which you are essentially trying to replicate is given here. Email hw2810@columbia.edu in case you can't access the link.

Project steps (steps in bold are open ended - these parts will be evaluated for time and effort):

6. Read about momentum
    a. There are tons of resources available to learn about momentum, but perhaps the best one is the book *Empirical Asset Pricing: The Cross Section of Stock Returns* by Bali, Engle, and Scott. Please read through the Momentum chapter, as our work here is very similar to the book's implementation. If you have trouble finding the book, please email hw2810@columbia.edu for access.
7. Get data
    a. Our focus is on long-term momentum as opposed to intra-day momentum (we want to make a strategy that undergoes rebalancing monthly, rather than daily or minutely).
    b. Get the daily price data for all stocks in the DJIA from an API of choice (or whatever method you choose), stretching back as far as you can find, preferably since pre-2005.
        i. There might be a stock that only IPO'ed recently. Feel free to either remove this stock or include it as you see fit
        ii. It is best if this data also includes market cap of each stock. Market cap is the total value of all outstanding shares of a company (so you can also find the # of shares and multiply it by the price every day).
    c. You should complete this project in Python. **Take full advantage of this time to learn as much as you can about pandas, numpy, and data analysis in Python. These skills will serve you well outside of this club. Also, it is probably best to use Jupyter notebooks for this assignment, as you can easily display plots.**
8. Do data preprocessing and exploratory data analysis
    a. **This part is fairly open ended.** Your goal is to understand the data. Report some summary statistics, fill in NA values, and locate patterns in the data. Think about what you would want to see as an investor.
    b. Make sure to calculate returns (price today / price yesterday - 1), see presentation.
9. Calculate the momentum signals that you would like to test
    a. See presentation, but these are 3 month returns, 6 month returns, 12 month returns, 11 month returns from month t-12 to month t-1, etc.
    b. Conduct exploratory analysis on these values as well. Some ideas, beyond the basics, include running regressions of future returns on these signals, grouping the stocks into 3 groups by each signal (top 3rd, middle 3rd, bottom 3rd) and reporting results from each portfolio (or returns), etc. Office hours are available if you are unsure what this means, or if you have any ideas you would like to run by me.
10. Construct a long-only momentum trading strategy
    a. Develop a class/function that, each month, calculates the returns from buying the stocks which have the highest momentum signal (assuming we group them into 3 groups, buy the longest 3 groups).

   b. Please work with returns as opposed to prices. Note that returns are additive. Say you invest 30% of some capital amount in stock 1, 50% in stock 2 and 20% in stock 3. Then, returns can be calculated by the following formula (assuming r_i is the returns on stock i)
      i. $0.3r_1 + 0.5r_2 + 0.2r_3$
      ii. For long-only portfolios, just take the positive bits
      iii. Convince yourself that this linear combination is correct
   c. The result from this step is, for each strategy, a return stream (a list containing a consecutive stream of returns spanning the beginning to the end of the backtest date)
   d. Notice, in this step, that a couple key parameters are not specified, see step 6.
11. Construct a function that takes in a return stream and returns the following performance statistics
   a. Plot an index:
      i. Consider if I had 1000$ and invested in this return stream, what would the index look like?
      ii. Do 1000 * df['return'].cumprod() and plot this series
   b. Mean return (annualized, if you're not sure what this means, feel free to shoot me an email)
   c. Volatility (annualized)
   d. Sharpe ratio (forget the risk-free rate, just do mean return / volatility)
   e. Hit rate (what percentage of all months are winning months)
   f. Max drawdown
      i. This is how much of our capital we can lose in one drawdown
      ii. Calculate drawdown as the cumulative product divided by the cumulative max of the cumulative product
         1. df['return'].cumprod() / df['return'].cumprod().cummax()
      iii. Take the maximum of the drawdowns to get max drawdown
   g. Highest monthly gain (annualized)
   h. Worst monthly loss (annualized)
   i. Turnover
      i. This one is hard, but incredibly important. We skip this for now, but if you would like to implement turnover, reach out to hw2810@columbia.edu for details.
   j. Any other ones you have heard of and would like to implement.
      i. No need to go bonkers here, the above ones are the main statistics for evaluating portfolios
      ii. Return skewness, return kurtosis
12. Backtest the momentum strategy for different parameter settings
   a. In step 4, we created a general strategy but there are a few settings which are not defined. Feel free to vary a couple (or all) of the following parameters and create a bunch of different portfolios to determine which one is best. The ones required are noted
   b. Portfolio Allocation (required)
      i. We can choose between equal weight (each long position gets 1/n allocation, and same for short positions), rank-based weighting (the best long position gets more allocation than the worst long position), value-weighted (weight the positions based on market cap of the stock, not required)
      ii. For this part, ensure that your weights for the long positions sum to 1
         1. Ex: If the market cap is 25 billion, 30 billion, 45 billion, then the weights should be
            a. 25/(25 + 30 + 45) = 0.25, 30/(25 + 30 + 45) = 0.3, 45/(25 + 30 + 45) = 0.45, respectively
   c. Momentum signals (required)

        i. Recall we created a few momentum signals. Run the backtest on a few or all of the signals (3 month momentum, 9 month momentum, 11 month-skip 1 month momentum, etc)

    d. Long only vs long-short (encouraged)
        i. Create a long only portfolio that buys the worst group (bottom 3rd) and calculates the return stream
        ii. Then, for each month, subtract the returns of the worst 3rd portfolio from the returns of the best 3rd portfolio (this is the same as a long-short portfolio).
        iii. Make sure that the other parameters (allocation and signal) is the same for the long and short portfolio
    e. Rebalancing frequency (optional)
        i. Try to create a strategy that rebalances every week, or every 3 months, or every 6 months, or every year
    f. Signal overlay (optional)
        i. You may notice that the strategies lose money during times of high volatility. Try liquidating all positions when VIX (need to get VIX data) is high (define "high" for yourself), and see if this improves your strategy.
    g. Asset universe (very optional)
        i. This one is hard, since it requires more data, but feel free to replicate the results on other asset classes, such as bonds, currencies, commodities, etc.
        ii. Only do this if you have a ton of time on your hands
    h. Anything else you can think of

13. Writeup (No page limit/requirement, but it should review all of your results). It must include all of the following.
    a. 1 paragraph discussion on the Efficient Market Hypothesis. Why do all signals eventually become useless? Why is it so hard to find profitable signals today? Extra points if you incorporate this into your analysis of the results.
    b. Discuss any and all assumptions you make in each step, from gathering data, to calculating performance metrics. Specify any methodology which is not specified here (or just write, "implemented exactly as described").
    c. Analyze the different portfolios, which one performs best? (How are you measuring "best"?)
    d. For the best portfolio, take a look at the index (the plot of returns in step 5). Do you notice any points when the portfolio does poorly? Why do you think it performed poorly?

Project 2: Factor Models

The goal of this project is to create factor models for the monthly returns of an asset class of your choice. If you haven't worked with factor models before, all you need to know is that they are essentially linear regression models which represent the return on any security as a linear function of multiple 'factors'. Therefore, they achieve dimensionality reduction by representing the return on 'n' assets using only 'k' underlying factors, with 'k' < 'n'. For example, if we apply the classic famous Fama-French 3-factor model to the S&P 500, we are representing the return on 500 assets via projection onto a 3-dimensional subspace. In this project we will be using Explicit Factor models rather than Implicit Factor models. See section 5.2 here for more detail, but the fundamental reason is that Explicit Factor models are economically interpretable whereas Implicit Factor models are not. However, we certainly recommend using statistical techniques such as PCA to help guide your search for explicit factors.

All factor models face the same tradeoff: A model with fewer factors explains less variance in the data, but reduces the dimensionality of the data further. Therefore, a successful factor model tries to explain as much variance as possible using as few factors as possible. The upper bound on the % of variance that can be explained by any 'k' factors can be found via the Explained Variance Ratio of the first 'k' principal components.

- For background on traditional equity factor models, you can refer to the below papers:
  - An extension of the Fama-French model to 5 factors is given here.
  - An example of the construction of a new equity factor is given here (note that the underlying data is accessible through this link as well).

We have prepared several datasets below, and you can choose which one you would like to work with. Below are project guidelines for each dataset. There are many papers not linked below which contain descriptions of factor models for different asset classes. We encourage you to read such papers in order to inform your models, but please cite your sources where appropriate. We also encourage you to model monthly returns rather than daily returns, as you are more likely to find economically interpretable factors by doing so. Here is the link to the data.

- Commodities
  - Data on historical futures curves for 9 of the most liquid commodities.
    - Here is an explainer on commodity futures. All you need to know is that the futures curves represent the next 'n' contracts due to expire. For instance, if we are considering the copper market, then because there is a contract for every month of the year, the futures curve as of May 2024 would contain the price of the copper futures contracts expiring in June, July, August, etc.
    - A paper containing a description of a factor model for commodity returns can be found here.
    - If you choose this option, we encourage you to achieve some form of improvement over the model given here by introducing a new factor or altering the existing factors to explain more variance. It is fine if you opt to model only a subset of the futures curve, such as the returns to the front-month contract (meaning, at any point in time, the active contract with the closest expiry date).

- Currencies
  - Data on historical exchange rates vs. USD for 33 currencies.
    - A description of a factor model for currencies can be found here.

        - If you choose this option, we encourage you to achieve some form of improvement over the model given here by introducing a new factor or altering the existing factors to explain more variance.
        - For reference, a list of currency codes can be found here.

- Global Yield Curves
    - If you choose this dataset, your task is to develop a factor model for global yield curves, which are simply a collection of interest rates on government bonds at different maturities. See here for an introduction to yield curves.
    - Since this is a challenging modeling problem, it is fine if you elect to model only a subset of the yield curve (such as a factor model for only the 10-year yield across different countries).
        - You may also model any feature of the yield curve that you wish, whether that be the period-over-period change in yield for a given maturity (which is a proxy for bond returns given that yield = coupon / price), the term premium at each maturity (in which case you are not modeling returns but rather the shape of the yield curve), etc.
    - You can find an example of a global yield curve factor model here.
        - Note that this paper treats the factors as latent variables, as does much of the literature on yield curve factor models. Therefore, it is fine if you model implicit rather than explicit factors for this dataset.

- Global Equities
    - You may construct a factor model for any non-US equity market, but you will have to source your own data. If you choose this option, you should first assess how some of the well-documented factor models for US equities perform when applied to your country's equity market and then modify the model to improve performance.

- US Equities
    - If you wish to use US Equities, you must introduce at least one new factor not found in a published paper and analyze the extent to which it improves explained variance or leads to more successful applications than existing factor models. You can use this dataset or any other dataset of US stocks

Project steps (steps in bold are open ended - these parts will be evaluated for time and effort):

1. Clone the repo
    a. Please do all development for your chosen project inside the corresponding folder. For example, if you choose the commodities project, please create all of your Jupyter notebooks inside of the 'Commodities' folder.
    b. In addition, please always use relative file paths so that we can easily run your code. Lastly, please keep all data (including external datasets you introduce) inside of the '!Data' folder.
    c. Note: We have made the repo read-only, so you can't push any commits. Instead, please simply include all .ipynb files in your final submission as instructed at the top of this document.

2. **Build your factor model.**

   a. This is intentionally left open-ended, so please don’t worry if you feel you are deviating from the approach taken in any of the above papers. As long as you develop a model which achieves the fundamental goal of representing a high-dimensional dataset of asset returns using a lower-dimensional set of factors, we are interested to see what you come up with.

3. Assess the proportion of total variance in the dataset explained by your factor model.

4. Examine the residuals of your model. Can you find any structure in the residuals?
   a. **Discuss next steps you might take**, if you were given time to extend your model, to reduce the structure in the residuals such that they become closer to white noise.
      i. Discuss the pros and cons of extending your model in this way.

5. **Backtest of a trading strategy**

   FIRST:

   a. Use your factor model to construct any trading strategy of your choice

   OR

   b. Construct any trading strategy of your choice (which does not have to incorporate your factor model)

   THEN:

   c. Construct a function that takes in a return stream and returns the following performance statistics
      i. Plot an index:
         1. Consider if I had 1000$ and invested in this return stream, what would the index look like?
         2. Do 1000 * df[‘return’].cumprod() and plot this series
      ii. Mean return (annualized, if you’re not sure what this means, feel free to shoot me an email)
      iii. Volatility (annualized)
      iv. Sharpe ratio (forget the risk-free rate, just do mean return / volatility)
      v. Hit rate (what percentage of all months are winning months)
      vi. Max drawdown
         1. This is how much of our capital we can lose in one drawdown
         2. Calculate drawdown as the cumulative product divided by the cumulative max of the cumulative product
            a. df[‘return’].cumprod() / df[‘return’].cumprod().cummax()
         3. Take the maximum of the drawdowns to get max drawdown
      vii. Highest monthly gain (annualized)
      viii. Worst monthly loss (annualized)
      ix. Any other statistics you would like to report
   d. Pass the returns of your trading strategy into the above function.

        i. Compare it to common indices such as the S&P 500 and any other indices or return streams you deem relevant to the asset class you chose (if applicable).

    e. Examine the ‘factor loadings’ of your strategy (aka regress the return stream of the strategy you constructed on the factors in your model and present the coefficients).

6. Writeup (No page limit/requirement, but it should review all of your results). It must include all of the following.
    a. Describe your model. Explain how each of your factors is constructed and why you chose them.
    b. Discuss any data pre-processing you performed, as well as any external datasets you utilized.
    c. Please export all Jupyter notebooks as PDFs and attach them to your writeup.
    d. Discuss any and all assumptions you make in each step, from gathering data, to calculating performance metrics. Specify any methodology which is not specified here (or just write, “implemented exactly as described”).
    e. Discuss any papers you read which were helpful to you during the project. Explain any important ways in which your work differs from that of the authors.
    f. Discuss at least one potential application of your factor model beyond what has been asked of you here. This does not have to be confined to alpha-seeking trade strategies but can include risk management, analytical tools, etc.
    g. Reflect on the success of your model. What tradeoffs does it make? What characteristics of the dataset contributed to the success (or failure) of your model?