

2023_final_project

May 10, 2023

1 2023 Spring ORF307 Final Project

Instructor: B. Stellato

As: I. Wang, V. Ranjan, S. Clarke, J. Ge

1.1 Instructions

Exam files: the exam files are available to download at [this link](#).

Date and time: from May 12, 2023 at 00:01am to May 17, 2023 at 11:59pm.

- Total time after download: 24 hours. No late submissions allowed.

1.2 Exam rules

- You are allowed to use all course materials on the midterm (lecture notes, books, precept materials, code, and homeworks). But you cannot use internet to search for answers.
 - You have to justify all your answers. If you use code from the course materials, you have to explain what each step means.
 - You cannot communicate with anyone during the exam.
 - No late submissions allowed. Make sure your submission goes through on time. You can resubmit as many times as you like until your time expires.
 - The exam is to be submitted electronically on Gradescope before 11:59pm on the final day. Please follow the [Gradescope guide](#) upload the PDF version of your exam.
-

2 Final Exam

2.1 Portfolio Optimization

You are a portfolio manager who is given $n = 11$ stocks to invest in:

– AMD, AMZN, BAC, GOOG, GS, K, KO, META, NKE, SHELL, SPY

Specifically, your goal is to decide a vector of weights $w \in \mathbf{R}^n$, where w_i is the fraction of your portfolio that you allocate to stock i .

To decide these weights, you are given historical data for these 11 stocks for every trading day from 02/01/2023 to 04/28/2023 ($T = 61$ total trading days). The data can be found in the `returns.csv` file, which we import as the returns matrix $R \in \mathbf{R}^{T \times n}$. The data file contains percentages, so we divide by 100 when forming the matrix R . An entry R_{ti} then contains the (decimal) return on trading day t for the price of stock i (negative values correspond to a loss). We define the vector of daily average returns per asset as $\mu = (R^T \mathbf{1})/T$.

```
[ ]: import numpy as np
      np.set_printoptions(precision=4) # Print few decimal places
      np.set_printoptions(suppress=True) # Suppress scientific notation
      import cvxpy as cp
      import pandas as pd
      from numpy.linalg import cholesky as llt
      import matplotlib.pyplot as plt

[ ]: # read the data
      returns = pd.read_csv('returns.csv')
      returns

[ ]: tickers = list(returns)
      tickers

[ ]: R = returns.to_numpy() / 100 # convert percentages to decimals when forming R
      T, n = R.shape
      R.shape

[ ]: mu = R.T @ np.ones(T) / T
      mu
```

2.1.1 Part A: Least Squares (25 points)

For a portfolio with constant weight w over the entire historical time period, the returns are given by $r = Rw$. In this scenario, we aim to choose a w that would have worked well in the past with the hope that it will work well in the future. Over the time series of length T , the average return is given by $\mathbf{avg}(r) = \mathbf{1}^T r / T$ and the risk, or standard derivation, is given by $\mathbf{std}(r) = \|r - \mathbf{avg}(r)\mathbf{1}\|/\sqrt{T}$.

1. (6 points) Our goal is to find portfolio normalized weights w that minimize the risk given a target daily mean return ρ . Formulate such a problem using constrained least squares.
2. (6 points) You believe, due to new economic developments, that it would be wise to specifically set aside some amount of your portfolio to the S&P 500 Index (stock 11, ticker SPY, in the data). So, you decide that you will always set aside 10% of your portfolio to buy stock in SPY. Add this constraint to the constrained least squares problem.
3. (6 points) Write the KKT linear system for the optimality conditions for the constrained least squares problem that you created in question 2.
4. (7 points) Say we want to observe a certain return η over the entire 3-month period. Since ρ represents target average daily return, the two are related by $\rho = \eta/T$. For $\eta = 5\%, 10\%, 20\%$,

solve the constrained linear system that you created in question 2 to find the optimal weights. For each weight vector w , plot the historical (average) returns vs time.

2.1.2 Part B: Linear programming (25 points)

For this part, rather than setting a target mean return, we aim to minimize the risk-return tradeoff with risk-aversion parameter $\lambda > 0$. In particular, we will consider $\|Rw - (\mu^T w)\mathbf{1}\|_1/T$ as a proxy for the risk, instead of using the risk definition from part A. Also, we will only consider long portfolios, i.e. without short-selling.

5. (5 points) Formulate the above problem of finding the optimal normalized weights as a piecewise linear optimization problem (you can keep the norm expressions explicit for now)
6. (5 points) Formulate this in CVXPY. For 100 values of λ , logarithmically spaced in the range $[10^{-1}, 10^2]$, solve the problem. For each optimal w , plot the average daily historical return vs risk for varying λ .

Note: If you choose to use a `cp.Parameter()` for λ , you will need to pass the flag `nonneg=True` to enforce it to be nonnegative. Also, you can use a `np.logspace()` to create the logarithmically spaced λ values.

7. (5 points) Write this problem as an LP.
8. (5 points) Take the dual of the LP you wrote in question 7.

Hint: you can use the identity $(\mu^T w)\mathbf{1} = Sw$, where

$$S = \begin{bmatrix} - & \mu^T & - \\ & \vdots & \\ - & \mu^T & - \end{bmatrix}.$$

9. (5 points) Formulate the dual problem with $\lambda = 1$ and solve it in CVXPY. Print the value of the dual variable that corresponds to the primal constraint $w \geq 0$. What does it tell you about the optimal primal variable w and the corresponding stock weights in the portfolio?

2.1.3 Part C: Integer programming for sparse allocations (25 points)

Consider, again, the risk-return trade off minimization problem with the 1-norm defined in Part B. Now, suppose that we are prohibited from holding more than k stocks in our portfolio (i.e. $k < n$).

10. (6 points) Formulate the new constraint in terms of the *cardinality* of w . Then, formulate such constraints using a big- M formulation and add them to the problem. What is the smallest value of M that you can use to encode the cardinality constraints?
11. (7 points) Solve the new integer program with $\lambda = 10$ and $k = 5$. Print out the optimal objective value. For the optimal solution, which stocks have nonzero weight?
12. (6 points) Relax the integrality constraints from your formulation in question 11 and solve the corresponding problem in CVXPY. Print out the optimal objective value. For the optimal solution, which stocks have nonzero weight? To account for numerical issues, we will say that a component w_i is nonzero if it is larger than 10^{-8} .
13. (6 points) A common way to relax cardinality constraints is to replace it directly with the 1-norm. Perform this relaxation and solve the corresponding problem in CVXPY. Print out

the optimal objective value and solution. Did either of these relaxations (this one or the one in question 12) give a better bound on the objective value? Explain your answer.

2.1.4 Part D: Integer programming for sparse trades (25 points)

Consider, again, the risk-return trade off minimization problem from part B. In this part, suppose that we have a current portfolio allocation $w^{\text{cur}} = (1/n)\mathbf{1}$, i.e., a uniform portfolio with equal weight in all stocks. With the historical data, we want to minimize the risk-return tradeoff, while, at the same time, penalizing trades. In fact, trades can turn out quite expensive if performed too often.

This means we would like to penalize difference between the new portfolio weights and the current ones, i.e., $w - w^{\text{cur}}$. In particular, we would like to penalize the number of nonzeros in that difference, which correspond to individual trades.

14. (6 points) Adjust the problem to enforce a maximum of 5 deviations from w^{cur} and formulate it as a mixed-integer LP.
15. (6 points) Solve the new problem with CVXPY with $\lambda = 1$, print the average return and risk, and verify that $w - w^{\text{cur}}$ has at most 5 nonzero values.
16. (6 points) Solving the last problem can take a long time in large dimensions. We would like to solve a similar version that induces sparsity. Replace the hard constraints in question 14 with a 1-norm penalization in the objective, weighted by coefficient β . Formulate the problem as a LP.
17. (7 points) Formulate the problem in CVXPY with parameters λ, β . Fix $\lambda = .1$, and solve the problem for 100 logarithmically spaced values of β in the range $[10^{-3}, 10^{-2}]$. For each β and corresponding optimal solution w , plot the number of nonzero values of $w - w^{\text{cur}}$ vs. β , with β on a logarithmic x axis. Again, to account for numerical issues, we will say that a component $(w - w^{\text{cur}})_i$ is nonzero if $|(w - w^{\text{cur}})_i| \geq 10^{-7}$. What do you notice?