ERDŐS QUANT FINANCE BOOT CAMP MINI-PROJECTS SUMMARY

NICHOLAS SWITALA

1. Introduction

Below I give brief descriptions of my work on the four assigned mini-projects in the Erdős Institute's summer 2025 quantitative finance boot camp, as well as a non-assigned "side project" that I worked out as well. All code was written in Python, using Jupyter notebooks; specific Python packages used in the projects are mentioned by name.

2. Mini-Projects 1 and 2

In the open-ended mini-project 1 we constructed examples of a high-risk portfolio and a low-risk portfolio. I looked at the Global Industry Classification Standard's division of the market into eleven sectors. Portfolio 1, which was to be "high-risk", consisted of the top ten stocks in the *information technology* sector by market cap: Microsoft, Nvidia, Apple, Broadcom, Oracle, Palantir, Salesforce, Cisco, IBM, and Intuit. (While this list includes several well-known "tech stocks", other companies commonly viewed as "tech stocks" are categorized in other sectors. For example, Amazon is in the consumer discretionary sector and Google/Alphabet is in communication services.) For Portfolio 2, I took the top five stocks in each sector by market cap and chose, from among those five, the one with the lowest volatility (annualized standard deviation of log returns) during the first quarter of 2025. The resulting portfolio contained Microsoft, Johnson & Johnson, Visa, TJX Companies, Coca-Cola, Chevron, Welltower, Linde, Duke Energy, AT&T, and Honeywell. (In the Jupyter Notebook I also built a third "control" portfolio out of the S&P 500 index and Berkshire Hathaway, but I will ignore this third portfolio for the purposes of this summary.) Both Portfolio 1 and Portfolio 2 give equal weight to all of their constituent stocks.

Using the `yfinance` package which downloads stock data as a `pandas` DataFrame, I checked to see what would have happened to $10000 invested in each of those two portfolios over the course of the first quarter of 2025. I then repeated this simulation for the full year 2024; in both cases I also computed the (annualized) volatility of the portfolio.

- $10000 invested in Portfolio 1 at the beginning of 2025 would yield $9323 (a loss!) at the end of the first (calendar) quarter, with annualized volatility of 30.7%.
- $10000 invested in Portfolio 2 at the beginning of 2025 would yield $10957 at the end of the first (calendar) quarter, with annualized volatility of 10.7% (lower than the volatility of the S&P 500 overall).

- $10000 invested in Portfolio 1 at the beginning of 2024 would yield $17606 at the end of the year, with volatility of 21.6%.
- $10000 invested in Portfolio 2 at the beginning of 2024 would yield $11649 at the end of the year, with volatility of 8.8% (again lower than the overall S&P's volatility, but with lower returns as well).

A portfolio consisting of only eleven stocks doesn't seem all that diversified, and yet we see that it is less volatile than the full S&P, because the full index is cap-weighted and therefore skews heavily toward its top ten or twenty constituents, some of which are highly correlated with one another.

Mini-project 2 asked us to use a statistical normality test to check whether stock/portfolio returns of our choosing actually follow a lognormal distribution. I chose to look at the portfolios from the previous project. Since 2025 has been a particularly volatile year in the markets so far, I looked only at 2024 data (one quarter at a time as well as over the whole year). From the `scipy.stats` package, I used the Shapiro-Wilk test with a p-value threshold of 0.05. Using this test, I found that Portfolio 1's daily returns appeared to follow a lognormal distribution during 2024 as a whole, as well as during each (calendar) quarter of 2024 individually. Portfolio 2's daily returns appeared lognormal during quarters 1, 2, and 4 of 2024, but not during quarter 3 and, interestingly, not during the year as a whole.

3. Mini-Project 3

In mini-project 3, we were asked to explore the sensitivity of the price of a call option to changes in the underlying stock price and to the passage of time. These are two examples of partial derivatives of the option price; the collection of all such partial derivatives is known as *the Greeks* (each of them is denoted by a Greek letter), with the two specific examples mentioned being Delta (price-sensitivity) and Theta (time-sensitivity). In the project, I gave the details of the calculation of an explicit formula for Delta. I also wrote code to approximate the Greeks using Monte Carlo and the finite difference method under Black-Scholes assumptions. However, as the Black-Scholes Greeks are well-understood, little independent research was needed to carry out these calculations. The primary value of mini-project 3 was in visualization. I created `matplotlib` plots of Delta, Gamma (Delta's derivative with respect to spot price), and Theta in several cases (in-the-money, out-of-the-money, and at-the-money options) as well as a `seaborn` heatmap from which one can see the time-sensitivity along one axis and the price-sensitivity along the other.

4. Mini-Project "3.5"

This was not an officially assigned mini-project, but was based on some of my own independent exploration using my background in linear algebra. In the boot camp lectures, we considered the effect of periodic Delta-hedging on an option position. In order to simulate a periodically hedged option, we need to generate random *paths* followed by the underlying stock price, not just its values at the option's expiration. If we need to simulate the values of the stock at times $t_1 < t_2 < \cdots < t_n$, one approach (easily done in `numpy` in a vectorized way) is incremental path generation: draw the first value S_{t_1}, then draw S_{t_2} *conditional on the already-drawn* S_{t_1}, and so on. An alternative is to draw the entire path at once, which can be done

since we know what the covariance matrix C of $S_{t_1}, \ldots, S_{t_n}$ must be.

The necessary construction is a *pseudo square root* of C, that is, a matrix A such that $AA^T = C$. Such an A is not unique; there is a unique lower triangular choice of A (AA^T in this case is known as a *Cholesky decomposition*), which matches what is done in the incremental path generation method, but there are other possibilities. For example, if $C = PDP^T$ is an orthogonal diagonalization of the symmetric matrix C, then $A = PD^{1/2}$ is another pseudo square root of C that can be used to generate stock price paths. One advantage of this method is that if the diagonal entries of D (which are the eigenvalues of C) are listed in decreasing order, then the transform formulas used to generate the stock price paths take the form of summations where the terms at the end are weighted less heavily. "Forgetting" a few of these terms (truncating the sum) results in a simulation of a lower-dimensional problem that converges more quickly.

In the mini-project, I performed Monte Carlo simulations to price an *Asian option* whose payoff at expiration is based on the average of the stock price at several different times using three methods: incremental path generation, orthogonal diagonalization, and the truncated variant. I also compared the run times of these simulations as well as their rates of convergence.

5. Mini-Project 4

The last mini-project asked us to revisit Delta-hedging assuming that volatility is not constant. The Black-Scholes model assumes that the stock's volatility σ remains constant during the life of the option, which is manifestly false in real-world markets. The boot camp lectures introduced the Heston model for stochastic volatility. In my mini-project, I looked at an alternative model for volatility, GARCH(1,1). I used maximum likelihood estimation to calibrate the parameters of a GARCH(1,1) model using daily S&P 500 returns from the previous year, and then simulated a periodically Delta-hedged call option on the S&P assuming the index's volatility follows this GARCH process. I also computed the value at risk (VaR) and expected shortfall of this hedged option position, and plotted histograms showing the distribution of profits under 10000 simulations with a varying hedge frequency (one hedge, two hedges, monthly hedges, biweekly hedges, and daily hedges).