Financial Risk Management and Regulation Project – Risk Calculation System

Group Member:
Runkun Xie
Duo Xu
Yifei Zhao

Contests

1. Executive Summary

This is a review for the risk calculation system for portfolio contains stocks and options.

The system contains several python scripts (*database.py*, *Input.py*, *Utilities.py* and *RiskCal.py*), setup files (*stock.csv*, *option.csv*) that users specified the stocks and options position, data files (*stocks.sqlite*, *iv.csv*) to store stocks and options data, and backup data files (*stock_price.csv*, *dates.csv*, *rf.csv*, and *underlying.csv*) for the test plans in case some computers do not have SQLite database.

The intention of this system is to calculate parametric, historical, and Monte Carlo VaR and ES. All these methods are implemented under GBM assumption or Normal assumption. In GBM assumption, the system assumes the portfolio price following a Geometric Brownian Motion; the Normal assumption assumes that the component stocks in the portfolio follow GBM process, while the value of the portfolio, which is the weighted sum of component stocks, follow a normal distribution.

The main strength of our system is that it is able to calculate various VaR and ES under different assumptions. Based on the implementation in class, we extend our system so that it can calculate portfolio with both stocks and options, portfolio with more than two stocks, and portfolio with long and short positions.

The weaknesses of our system are the discrepancies between calculated VaR and ES, and real VaR and ES in the market. The discrepancies arise from the several assumptions we made for the computation, which we will discuss in the assumption and conclusion part.

2. Model Documentation

2.1 Model Introduction

This is a risk calculation system for the portfolio consists of stock and option. The system is used to calculate Value at Risk (VaR) and Expected Shortfall (ES). It calculate VaR and ES using parametric, historical, and Monte Carlo method.

The algorithms in the system are based on the lectures and homework of Columbia Math GR 5320: Financial Risk Management and Regulation by Harvey J. Stein.

		Parametric (GBM)		Parametric (Norm)		Historical (abs)		Historical (rel)		Monte Carlo (GBM)		Monte Carlo (Norm)	
		VaR	ES	VaR	ES	VaR	ES	VaR	ES	VaR	ES	VaR	ES
Stock	Long	√	√	√	√	√	√	√	√	√	√	√	√
	Short	√	√	√	√	√ √	√ √	√ √	√ √	√ √	√ √	√ √	√ √
	L&S	√ √	√ √	√ √	√ √	√ √	√ √	√ √	√ √	√ √	√ √	√ √	√ √
with Op	Long	√ √	√ √	×	×	√ √	√ √	√ √	√ √	√	√	×	×
	Short	√ √	√ √	×	×	√ √	√ √	√ √	√ √	√ √	√ √	×	×
	L&S	√ √	√ √	×	×	√ √	√ √	√ √	√ √	√ √	√ √	×	×

The above graph summarizes the functionalities of our system. In the picture, the light green area is the algorithms we implement in the course, the dark green area along with the light green area is the risk statistics the system can calculate, and the light yellow area is the part where the system cannot calculate.

2.2 Model assumption

- Customers provide data given in the sample input correctly as stated in software design.
- Input list and data contain the same options and in the same sequence.
- Stocks and underlying stocks of the options should be within the list of S&P 500 companies. Otherwise, users should provide the required historical data by themselves.
- Options are at the money European options with the strike price equalling to the stock price.
- The stock prices follow Geometric Brownian Motion (GBM) with constant drift and constant volatility.
- Risk-free rate stays constant over the life of an option.
- Stocks pay no dividend.
- No transaction costs or taxes.
- No arbitrage opportunities exist.

2.3 Mathematical description

Risk models provided are value at risk (VaR) and expected shortfall (ES). Pricing models are GBM and Normal for Parametric and Monte Carlo.

- p level of VaR: p th quantile of the loss function
- Expected shortfall: expected loss conditioned on loss > VaR

- GBM: assume stock prices follow geometric brownian motion with drift rate μ and volatility σ

  $$GBM : S_T = S_0 e^{(\mu - \frac{\sigma^2}{2})T + \sigma W_T}$$

- Parametric (GBM): Fit GBM parameters for the portfolio; calculate the VaR and ES for the portfolio based on the drifts and volatilities using the formula

  $$ParametricGBM :$$
  $$VaR(Long) = V_0 - V_0 e^{(\mu - \frac{\sigma^2}{2})t + \sigma\sqrt{t}N(1-p)}$$
  $$VaR(Short) = V_0 e^{(\mu - \frac{\sigma^2}{2})t + \sigma\sqrt{t}N(p)} - V_0$$
  $$ES(Long) = V_0(1 - \frac{e^{\mu T}\Phi[\Phi^{-1}(1-p) - \sigma\sqrt{T}]}{1-p})$$
  $$ES(Short) = V_0(\frac{e^{\mu T}(1 - \Phi[\Phi^{-1}(p) - \sigma\sqrt{T}])}{1-p} - 1)$$

- Parametric (Norm): Calculate the VaR and ES for the portfolio based on assuming each stock and option follow GBM. Compute the VaR assuming normally distributed future values.

  $$ParametricNormal :$$
  $$E(V_t) = \sum w_i E(S_{i,t}) = V_0 e^{\overline{u}t}$$
  $$Var(V_t) = (V_0 e^{\overline{u}_t t})^T \cdot \sum (V_0 e^{\vec{u_t} t})$$
  $$VaR = V_0 - (EV_t - (1-p)\sigma(V_t))$$
  $$ES(Long) = V_0 - (\mu_t - \frac{e^{-\Phi^{-1}(1-p)^2/2}}{(1-p)\sqrt{2\overline{\mu}}}\sigma_t)$$
  $$ES(Short) = (\mu_t + \frac{e^{-\Phi^{-1}(1-p)^2/2}}{(1-p)\sqrt{2\overline{\mu}}}\sigma_t) - V_0$$

- Black-Scholes formula for call and option pricing

  $$Black - Scholes :$$
  $$d_1 = \frac{ln(S/K) + (r + \sigma^2/2)\tau}{\sigma\sqrt{\tau}}$$
  $$d_2 = d_1 - \sigma\sqrt{\tau}$$
  $$c = SN(d_1) - Ke^{-r\tau}N(d_2)$$
  $$p = Ke^{-r\tau}N(-d_2) - SN(-d_1)$$

2.4 Model calibration

For each date, parameters μ, σ, VaR and ES are calculated using a 5-year window or equivalent 5-year exponential weight (approximately 0.9989).

3. Software design documentation

The system is developed based on *Python 3.6*, *DB Browser for SQLite 3*, and *Microsoft Excel 2013*. Other versions may work fine too.

The components of the system are as follows:

- The whole system has 4 python scripts, which are ***database.py***, ***Input.py***, ***Utilities.py*** and ***RiskCal.py***. Run *RiskCal.py* and all the results would be presented.
- There are two data files we used to save data, first is an SQLite database we build, which is ***stocks.sqlite*** that stores the stock price data. The other is ***iv.csv***, which contains the implied volatility of all the underlying stocks of the options. Users should also renew the data of *iv.csv* if they want to use specific options.
- There are also two input files that users have to specify, which are ***stock.csv*** and ***option.csv*.**
- In case some users have no access to SQLite database on their computer, we also prepare 4 csv files with the necessary data for our test cases. They are stock_price.csv, dates.csv, rf.csv and ***underlying.csv***.

3.1 Input files

In *stock.csv*, users should specify what stocks they want to long/short and the corresponding positions. Users should give the Symbol (eg. XOM). The positions should be given as a weight. Users can change the elements in this file to personalize their stock portfolio.

In *option.csv*, users should specify the underlying stock in column 'Ticker', the position for each option in column 'Position', the type of each option in column 'Type', and the time to maturity for each option in column 'Maturity'. The sum of stock positions and option positions should be equal to 1 or -1. Notice that users should enter 1 for call options and 0 for put options. The time to maturity is measured in year. Users can change the elements in this file to personalize their option portfolio.

3.2 Data files

In *iv.csv*, the users should give the implied volatility of all the underlying stocks. Each column is for one underlying stock.

The database, *stocks.sqlite*, created by file *database.py* contains three tables related to each other to streamline the data download and input process.

In table *symbol*, we have all the stock names that we have enough data for. When our system gets portfolio requirements from the users, it automatically searches in *symbol* to check whether we already have the data. If not, it will download data from Yahoo Finance and store it into the database for future use. The variable id in *symbol* serves as the primary key and is linked to the variable symbol_id in the table *price*. For example, symbol_id = 1 indicates the price data for id = 1 in *symbol*, which is XOM. Table *price* has all the price dates and adjusted closing prices for each stock.

In table *rf*, we downloaded the 13-week Treasury Bill discount rate to construct the risk-free rates. The data is downloaded from Yahoo Finance with the ticker '^IRX'. However, the ^IRX data Yahoo Finance provides are the (annualized) discount rates for the 13-week treasury bill. That is, if the value of the data ^IRX at month t is x, then it means you pay 100 - 0.25x dollars at month t, and the Treasury will pay you 100 dollars at month t+3. So we transformed the original values of ^IRX to the risk-free rates with the following formula:

$$\left(\frac{1}{1-\frac{x}{100}\frac{1}{4}}\right)^{1/3}-1.$$

Just in case some users may not have access to SQLite database on their computer, we also provide 4 csv files containing the data need for our 4 test cases, to show how the system works. All data are in *stock_price.csv*, *dates.csv*, *rf.csv* and *underlying.csv*. Users can read the files into Python with the code in RiskCal.py. The code for this part is currently annotated.

3.3 Python Scripts

- ***Input.py***

The file *Input.py* reads the portfolio information from the input files, extracts stock prices and risk-free rates from the SQLite database, and process the data. It has three classes:

(a) The class *StocksRead* reads the stock positions and get stock data.
Member functions:
obtain: Check in the table *symbol* for each stock to see whether we have data for it
merge_data: Merge the data for all stocks together into a DataFrame, which can be output later.
get_data: Output all the stock prices, the stock positions, the dates, and the risk free rates. If we already have the data for the stock, it will extract the data from SQLite database and put them into a DataFrame. If not, it will automatically download the data from Yahoo Finance, put them into a DataFrame, and store them into SQLite database as well.

(b) The class *OptionsRead* reads the option positions and get underlying stock data. It works in a quite similar logic as StocksRead.
Member functions:
obtain: Check in the table symbol for each underlying stock to see whether we have data for it
merge_data: Merge the data for all underlying stocks together into a DataFrame, which can be output later.
get_data: Output all the underlying stock prices, the option positions, the option types, the maturities and the implied volatilities.

(c) The class *ProcessData* makes some process to the data to make them have equal length.
Member functions:

process: Find the shortest length among the stock price data, underlying price data, and the implied volatility data. Truncate the longer data to make all the data have equal length.

- ***database.py***

This is the file we used to generate *stocks.sqlite*, the data file that contains stocks price, interest rate, and so on.

- ***Utilities.py***

The file *Utilities.py* define two functions for use in *RiskCal.py*. The function *EuropeanOptions* calculates the European option prices within the Black-Scholes framework. The function *EW_all* performs exponential weighting to the price data.

- ***RiskCal.py***

This is the main class we used to calculate VaR and ES. The member functions and their functionalities are as follows:

(a) Main Member Functions:
cal_portfolio_price: Calculate portfolio price
cal_parametric_risk_GBM: Calculate parametric VaR and ES, assume portfolio follows GBM
cal_parametric_risk_Normal: Calculate parametric VaR and ES, assume stocks follows GBM, portfolio follows the normal distribution
cal_historical_risk_relative:Calculate historical VaR and ES of portfolio using relative return
cal_historical_risk_absolute: Calculate historical VaR and ES of portfolio using absolute return
cal_MonteCarlo_risk_GBM: Calculate Monte Carlo VaR and ES, assume portfolio follows GBM
cal_MonteCarlo_risk_Normal: Calculate Monte Carlo VaR and ES, assume stocks follows GBM, portfolio follows the normal distribution
backtest: Backtest all VaR to the history, and plot the results
print_results: Print all the results

(b) Other Member Function:
_cal_option_price: Calculate option price of option in the portfolio
_cal_portfolio_price_no_option: Calculate portfolio price without option
_cal_portfolio_price_with_option: Calculate portfolio price with option
_cal_logrtn: Calculate log return of price series
_cal_paras_window: Calculate GBM parameters using window method
_cal_paras_EW: Calculate GBM parameters using Exponential Weighting method

The standard procedure for using the system: **read data**, **create an example of RiskCal class**, **run print_results function**. The test case 1 to 4 illustrate how users should use our system.

The detail in RiskCal class goes like this: first, the class takes data and setups/parameters as input. Then, the class calculates the portfolio price. After that, the class use portfolio price series to estimate drift rate μ and volatility σ. Finally, the class do all the calculations and presents the results. Here are some implementation details in RiskCal:

(a) Portfolio with or without options. Based on user settings, the calculation of portfolio price with or without options can be different, so we separate these two calculations. If the portfolio does not contain option, we calculate the portfolio price directly, if not, we first calculate the option price series based on Black Scholes formula, then combine stock and option price to calculate portfolio price.

(b) GBM or normal assumptions. Under GBM assumption, we can use portfolio price and do all the calculation. However, under the normal assumption, each stock in portfolio follow GBM process. Therefore, we calibrate our system so that it can calculate N by 1 portfolio price series, or N by M stock price series. For example, the *_cal_paras_EW* function is capable of estimate N by M by M correlation matrix of the stock price series in each day, and also able to estimate N by 1 drift rate μ and volatility σ of the portfolio price.

(c) Long or short position. For the six VaR and ES functions, we treat long and short portfolio differently, by using different quantile of the normal distribution and different formulas. Also, since we assume the portfolio price is under GBM, so the price series need always to be positive or negative, so we can classify it to long or short position and use the corresponding risk calculation functions.

4. Validation Methodology and scope

4.1 Scope of validation

The validation is performed by testing several test cases which covered the main functionalities of the system. In the test process, we exam the test data, the completeness of test plans, and test results.

4.2 Test plan

In order to fully test the functionalities of our system, we prepared four test cases, which covered most of the situation the system may be faced. In detail, the first two cases would use XOM and INTC, in long or short position, to calculate various VaR and ES, and the results would be identical or similar to the homework, depends on the position setups. Then, we would test more complicated situation in case 3 and 4, which include the calculation of VaR and ES calculation of N stocks, and portfolio consists call or put options.

The test plans are as follows:

- ***Pure stock position***
  The portfolio consists of 2 stocks (XOM and INTC), equally weighted invest in the start day (+50%, +50%), and using Exponential Weighting method. The results would be identical to the homework.
- ***Pure stock position, portfolio contains long and short stocks***
  The portfolio contains 3 stocks (XOM, INTC, and 3M) and using Window method. We short the first two stocks and long the third stock (+51%, +51%, -2%). Since we only add a small long position of third stock, the result would be similar to homework, but the VaR and ES would be larger due to the larger leverage ratio.
- ***Portfolio with options: long N stocks and hedge by long put options***
  The portfolio long 3 stocks (XOM, INTC, and 3M) and long 2 put options (IBM and AAPL) and using Exponential Weighting method. The stocks and options are equally weighted invest, with stock position of (32%, 32%, 32%) and option position of (2%, 2%).
- ***Portfolio with options: short N stocks and hedge by long call options***
  The portfolio short 3 stocks (XOM, INTC, and 3M) and long 2 call options (IBM and AAPL) and using Window method. The stocks and options are equally weighted invest, with stock position of (-34%, -34%, -34%) and option position of (1%, 1%).

Major difference and summary of test cases:

	Asset Type	Position	Method	T	Window	Notional
Test Case 1	Stock (XOM, INTC)	(0.5, 0.5)	Exponential Weighting	20	5	$10000

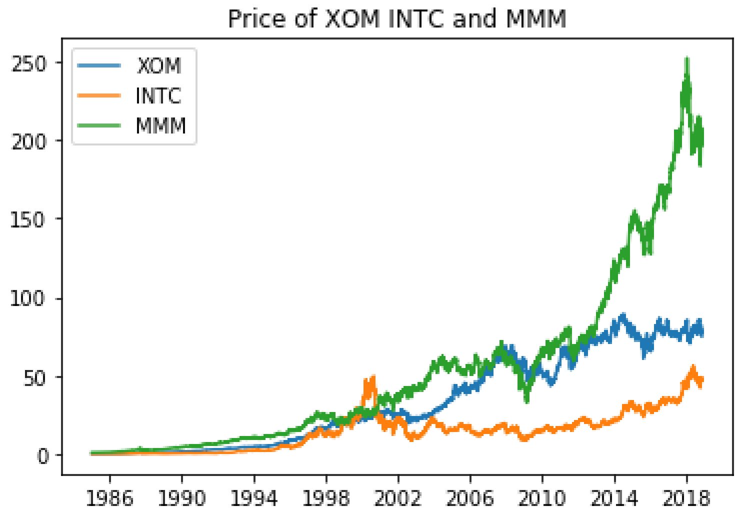
Test Case 2	Stock (XOM, INTC, 3M)	(0.51, 0.51, -0.02)	Window	20	5	$10000
Test Case 3	Stock (XOM, INTC, 3M)	(0.32, 0.32, 0.32)	Exponential Weighting	8	5	$10000
	Option (IBM and AAPL)	(0.02, 0.02)				
Test Case 4	Stock (XOM, INTC, 3M)	(-0.34, -0.34, -0.34)	Window	8	5	$10000
	Option (IBM and AAPL)	(0.1, 0.1)				

The test plans roughly contain all the situation the system may encounter. We test different method (Exponential Weighting and Window) used to calculate GBM parameters, add multiple call and put options in the test portfolios, use long and short position in different test cases, and try the calculation of VaR and ES of more than 2 stocks.

5. Validation Results

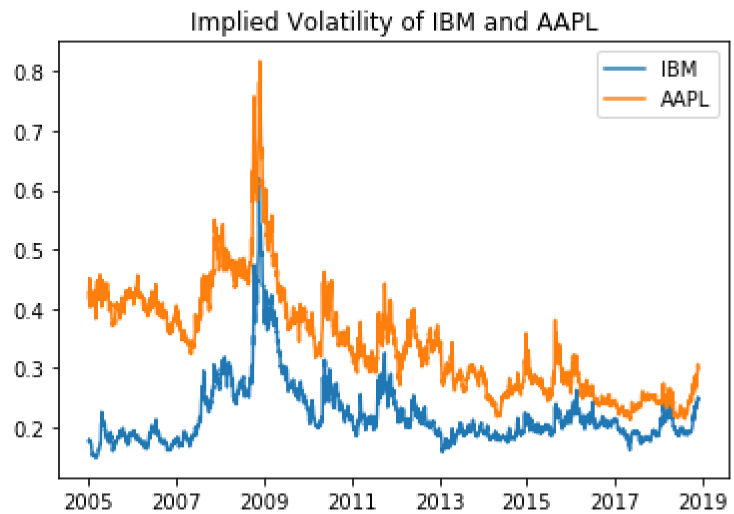
5.1 Data Analysis

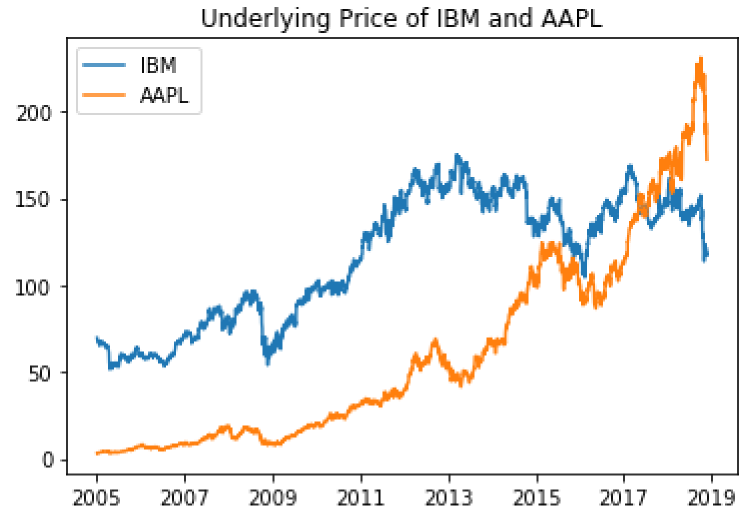
As mentioned, we build a SQLite database *stocks.sqlite* to store stock data and use *iv.csv* to contain our sample option implied volatility data. Here, we exam the data we used in testing samples:



The price series of stocks in test cases XOM, INTC and MMM). The price series is reasonable, they are similar to the data we have in the homework, and they do not contain jump point, which means the price is adjusted.

We have more than 30 years of data. So in the test cases, we would use a 5-year window (or 5-year equivalent beta for exponential weighting method), and a 20-year investment horizon, as we did in the homework.





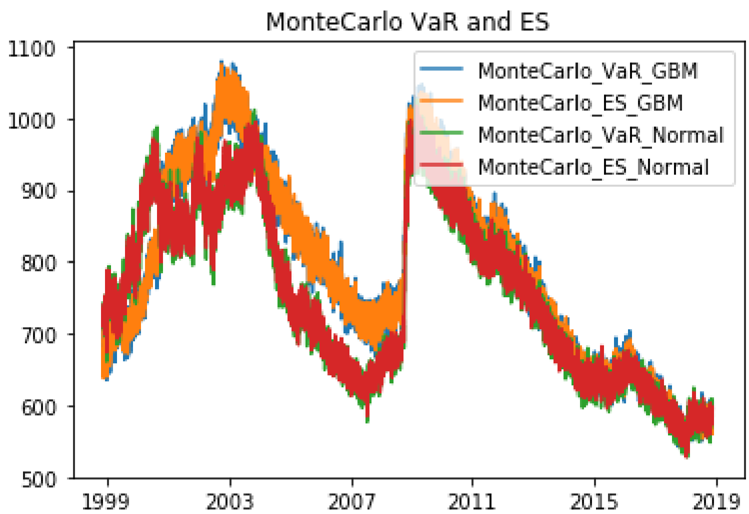
The implied volatility and underlying price of options in test cases (IBM and AAPL). As can be seen from the picture, the implied volatilities and the underlying prices are matched, the volatilities reach it peaks when the price series have a sudden drop.

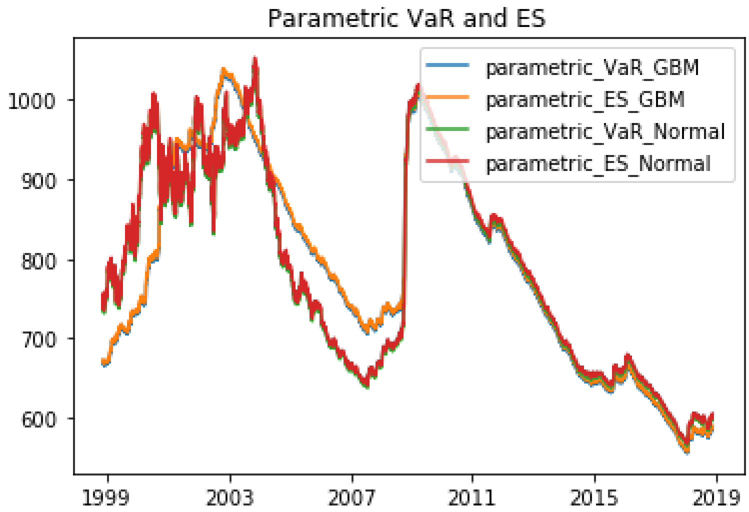
For options, we have approximate 13 years of data. So that in the test cases, we would use a 5 year window (or 5 year equivalent beta for exponential weighting method), and an 8-year investment horizon.

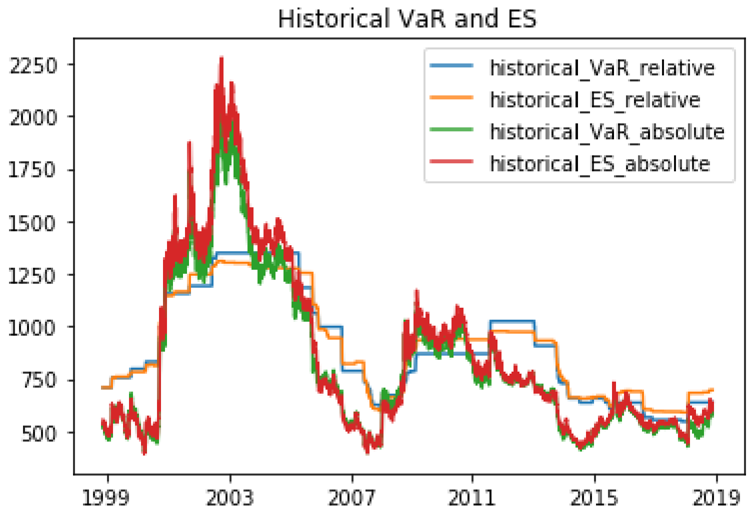
5.2 Test Results

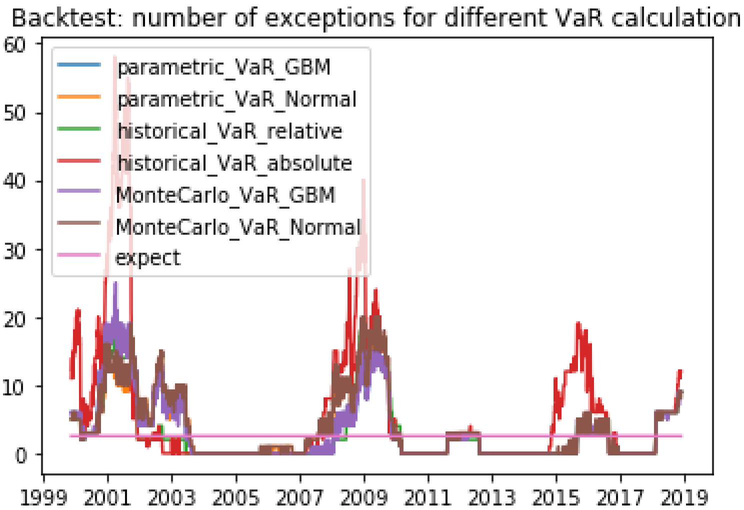
- **Test Case 1: Pure stock position**

	Asset Type	Position	Method	T	Window	Notional
Test Case 1	Stock (XOM, INTC)	(0.5, 0.5)	Exponential Weighting	20	5	$10000







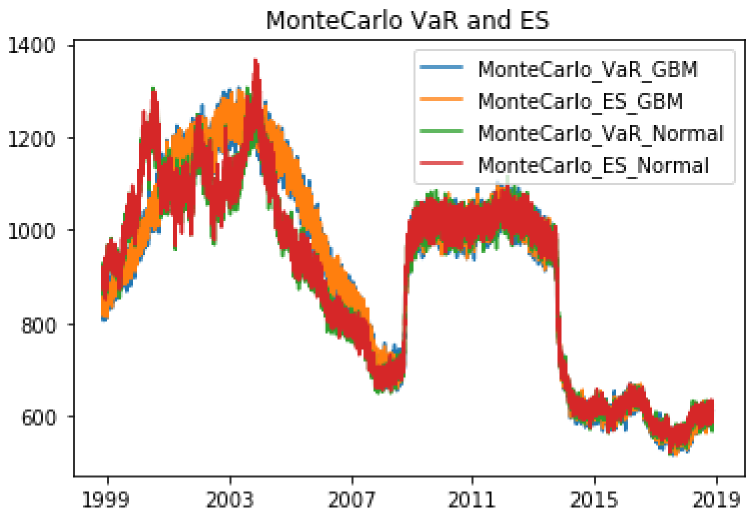


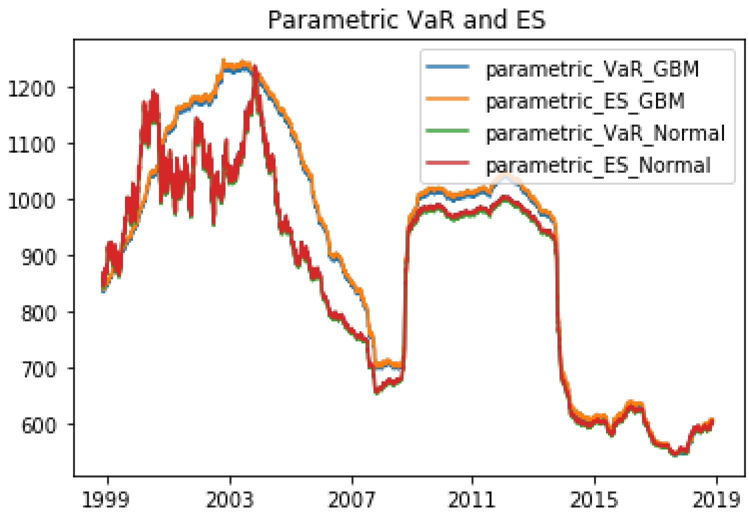
The first case consists of 2 stocks and using Exponential Weighting method, the results of Monte Carlo, parametric, and historical VaR and ES are matched to the results we obtained in homework.

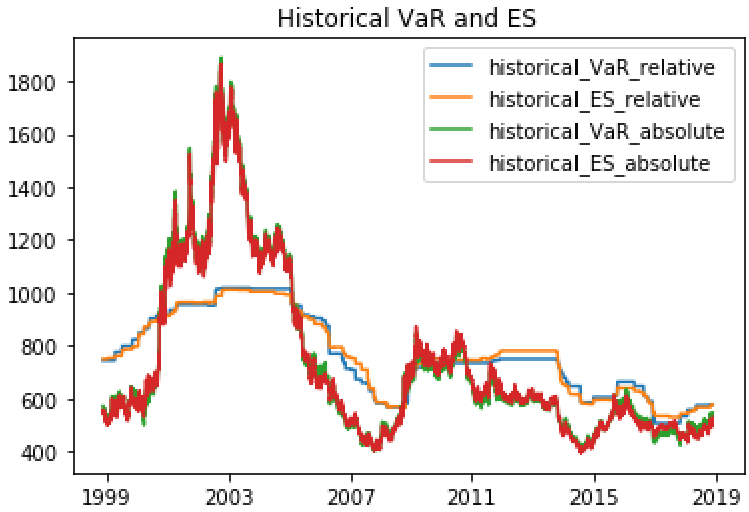
- **Test Case 2: Pure stock position containing long and short stocks**

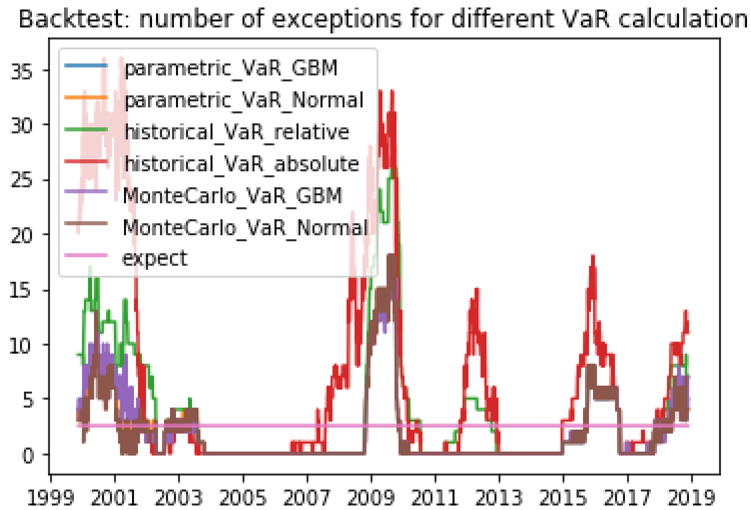
	Asset Type	Position	Method	T	Window	Notional
Test Case 2	Stock (XOM, INTC, 3M)	(0.51, 0.51, -0.02)	Window	20	5	$10000

The portfolio contains 3 stocks (XOM, INTC, and 3M) and using Window method. It is also similar to the homework in that it short XOM and INTC with -$5000, and the difference is that we add a third stock, 3M, which has a very small long position. The intention of this test case is to test the system's ability to handle multiple stocks, i.e., more than two stocks.









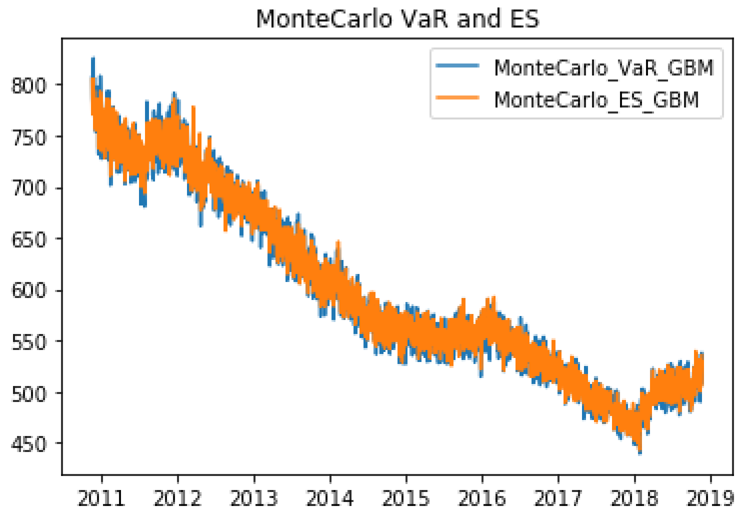
As expected, the short stock position has higher MC and parametric VaR, and lower historical VaR compared to long position in test case 1. It means that the return distribution has a fatter tail at the positive side than the negative side.

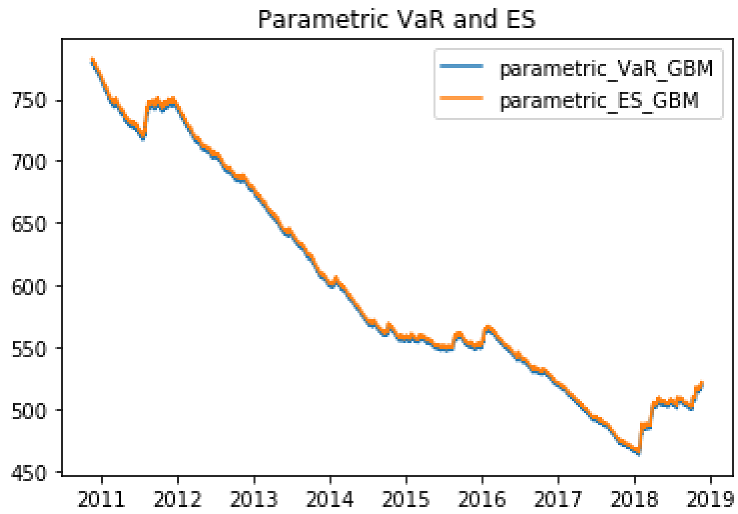
When compared to the short portfolio in HW 10, the test case 2 have higher VaR and ES. The reason is that when we add the +0.02 long position of 3M, we also increase the short position of XOM and INTC from -0.50 to -0.51, which increase the portfolio's leverage ratio.

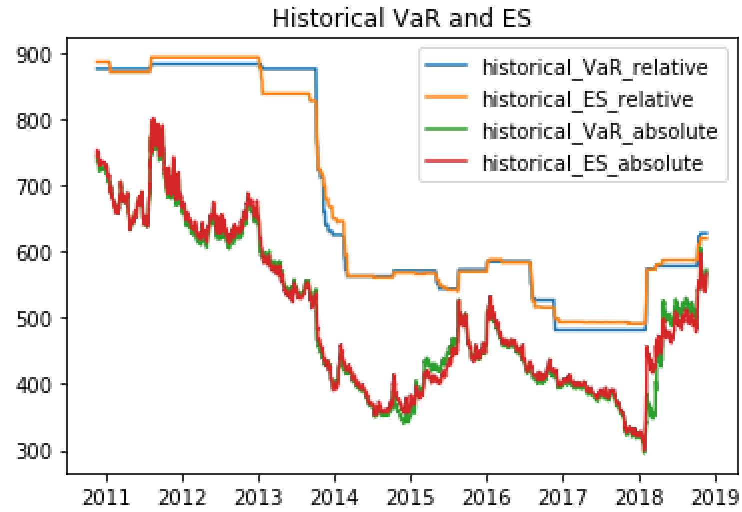
- **Test Case 3: Portfolio with option, long N stocks and hedge by long put options**

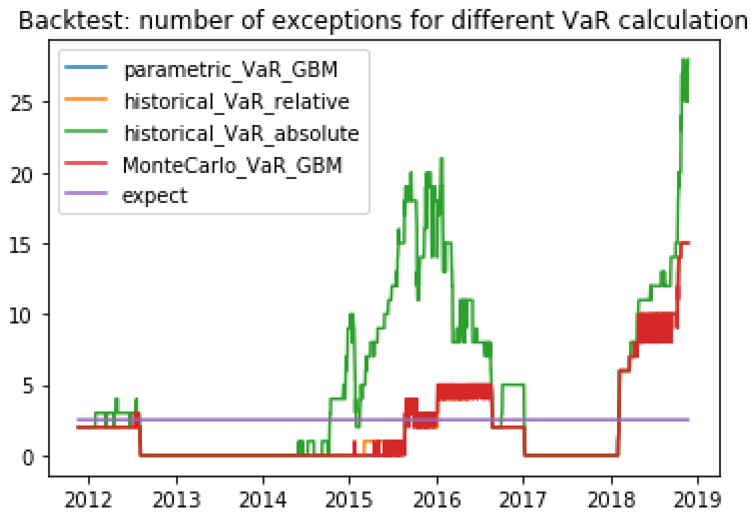
	Asset Type	Position	Method	T	Window	Notional
Test Case 3	Stock (XOM, INTC, 3M)	(0.32, 0.32, 0.32)	Exponential Weighting	8	5	$10000
	Option (IBM and AAPL)	(0.02, 0.02)				

In this case, the portfolio long 3 stocks (XOM, INTC, and 3M) and long 2 put options (IBM and AAPL) and using Exponential Weighting method. The stocks and options are equally weighted invest, with stock position of (32%, 32%, 32%) and option position of (2%, 2%).







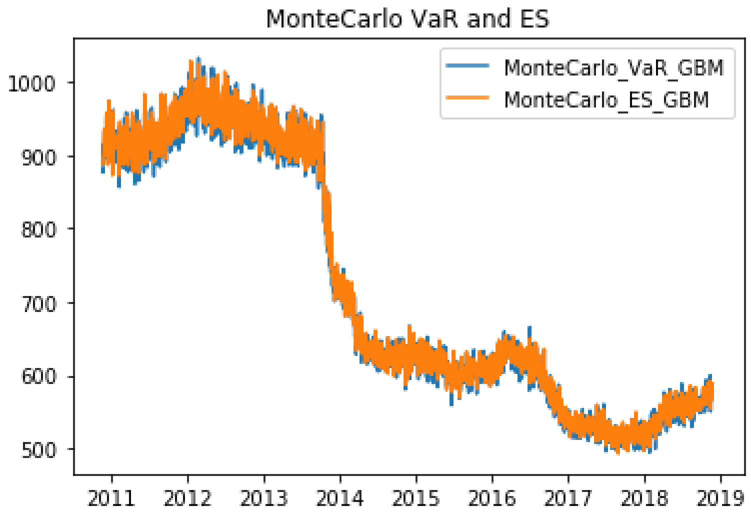


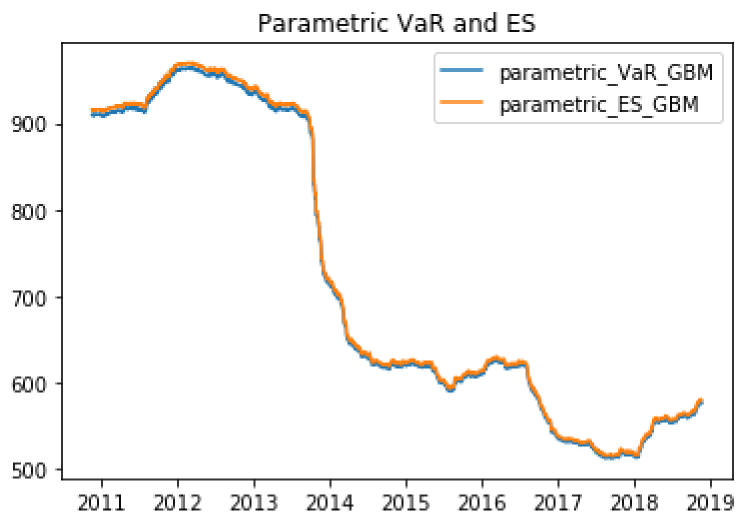
Although the stocks in the test case are new, the results are also reasonable. Compared to the previous two test case, this portfolio has VaR and ES that are roughly smaller than half of the VaR and ES in the previous two cases. If we assume the stocks in each test cases have similar drift and volatility, a portfolio of $10000 would have around $1000 99%VaR and 97.5%ES, as shown in the previous test cases. This test case has a smaller risk because it uses 4% of its portfolio value to long put options, and the hedging effect of the put option reduce the risk significantly.

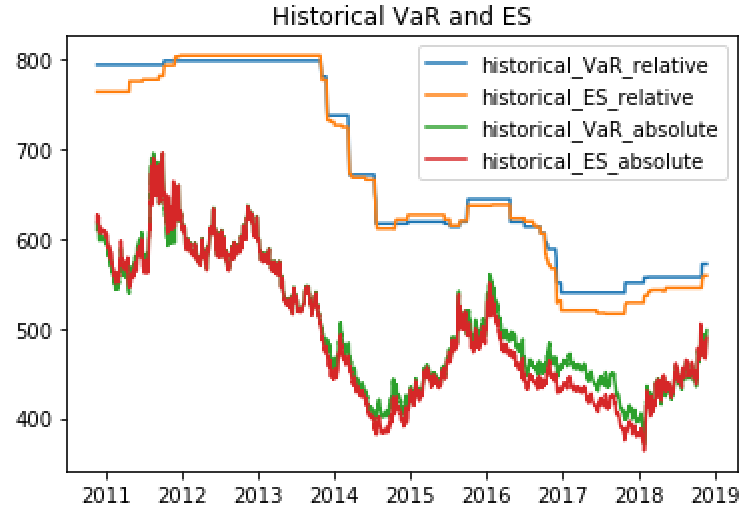
- **Test Case 4: Portfolio with option, short N stocks and hedge by long call options**

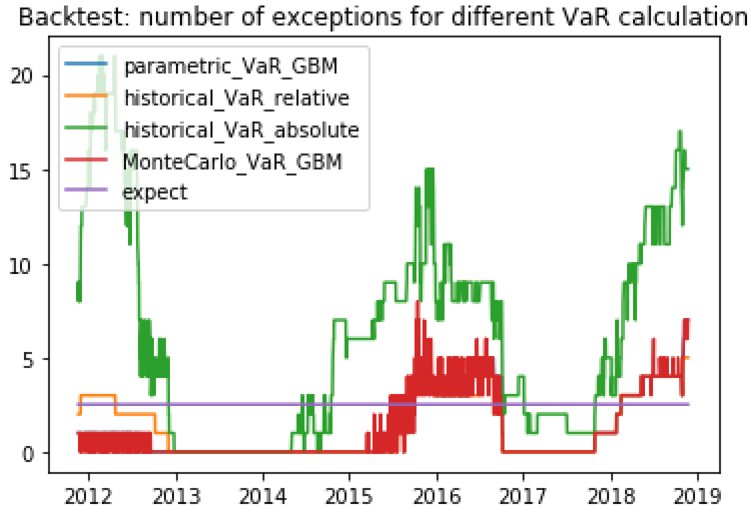
	Asset Type	Position	Method	T	Window	Notional
Test Case 4	Stock (XOM, INTC, 3M)	(-0.34, -0.34, -0.34)	Window	8	5	$10000
	Option (IBM and AAPL)	(0.01, 0.01)				

The last case is the opposite of case three. In this case, we short 3 stocks (XOM, INTC, and 3M) and long 2 call options (IBM and AAPL) and using Window method. The stocks and options are also equally weighted invest, with stock position of (-34%, -34%, -34%) and option position of (1%, 1%).









Under window method, the results are also reasonable, since they are similar to the test case three, and have jumps in VaR and ES due to the window estimation method.

6. Conclusion

We conclude this model is adequate and comprehensive for VaR and ES calculation. The system is capable of calculating the VaR and ES of portfolio contains stocks and options with long or short position, and backtest the results to the history. It extends the calculation in the class and meets the requirements of the projects.

The system, however, has certain weaknesses. One of the weakness is not considering the risk free return on cash position. Since the money market account have volatility equals 0, it only has a slight effect on risk analysis. The second weakness is that calculations for Parametric and Monte Carlo VaR and ES of portfolios with option under Normal assumption are unavailable. These could be improved by making the following changes:

- Using the US 3 month T bill rate as the drift rate for cash position and add the change of cash position into the calculations for VaR and ES
- Using existed functions in the system, calculate the correlation of stocks position and underlying stocks of the options position, simulate their price changes by GBM assumption and BS formula, and calculate MC VaR and ES
- Adding formula for Parametric VaR and ES for portfolio with option under Normal assumption

7. Bibliography

[1] Chen, R. (2018, 01). *Global Risk Management: A Quantitative Guide*. Retrieved 12, 2018, from http://faculty.fordham.edu/rchen/grm.pdf

[2] J.Michael, S. (2001). *Stochastic Calculus and Financial Applications* (Vol. 45). *Stochastic Modelling and Applied Probability*. Springer-Verlag.

[3] Harvey J., Stein (2018). *Lecture 4: Market Risk* [PowerPoint slides]. *MATH GR 5320 Financial Risk Management and Regulation*

[4] Harvey J., Stein (2018). *Lecture 5: Model Risk Management* [PowerPoint slides]. *MATH GR 5320 Financial Risk Management and Regulation*