Project Report

Applied Mathematics
at Lahore University of Management Sciences

Affan Zafar, Ayesha Moqeet, Fahad Hassan

Superviser - Dr.Zahra Lakdawala

25^{th} October 2021

Contents

1 — Acknowledgments

We would like to thank J. Nathan Kutz, Steven L. Brunton, Bingni W. Brunton and Joshua L. Proctor, authors of the book Dynamic Mode Decomposition for providing us with the algorithm and code on which we based the foundation for our research.

2 — Black Scholes

2.1 The Black-Scholes Equation

The Black–Scholes equation is a linear parabolic partial differential equation. In fact, almost all partial differential equations in finance are of a similar form. They are almost always linear, meaning that if you have two solutions of the equation then the sum of these is itself also a solution. Or at least they tended to be linear until recently. Financial equations are also usually parabolic, meaning that they are related to the heat or diffusion equation of mechanics. One of the good things about this is that such equations are relatively easy to solve numerically.

The Black-Scholes equation describes the price of an option over time. The derivation of this equation is complex and exceeds the scope of this paper, so we simply provide the equation.

$$\frac{\partial C}{\partial t} + \frac{1}{2}\sigma^2 S^2 \frac{\partial^2 C}{\partial S^2} + rS\frac{\partial C}{\partial S} = rC \tag{2.1}$$

C = Call option price

S = Current stock price

K = Strike price of the option

r = risk-free interest rate (a number between 0 and 1)

σ = volatility of the stocks return (a number between 0 and 1)

t = time to option maturity (in years)

N = normal cumulative distribution function

2.2 The Black-Scholes Assumptions

Lognormal distribution: The Black-Scholes-Merton model assumes that stock prices follow a lognormal distribution based on the principle that asset prices cannot take a negative value; they are bounded by zero.

No dividends: The BSM model assumes that the stocks do not pay any dividends or returns.

Expiration date: The model assumes that the options can only be exercised on its expiration or maturity date. Hence, it does not accurately price American options. It is extensively used in the European options market.

Random walk: The stock market is a highly volatile one, and hence, a state of random walk is assumed as the market direction can never truly be predicted.

Frictionless market: No transaction costs, including commission and brokerage, is assumed in the BSM model.

Risk-free interest rate: The interest rates are assumed to be constant, hence making the underlying asset a risk-free one.

Normal distribution: Stock returns are normally distributed. It implies that the volatility of the market is constant over time.

No arbitrage: There is no arbitrage. It avoids the opportunity of making a riskless profit.

2.3 Boundary Condtions

The Black–Scholes equation knows nothing about what kind of option we are valuing, whether it is a call or a put, nor what is the strike and the expiry. These points are dealt with by the Boundary conditions. We must specify the option value V as a function of the underlying at the expiry date T . That is, we must prescribe V(S, T), the payoff.

For example, if we have a **call** option then we know that

V(S, T) = max(S-E, 0).

For a **put** we have

V(S, T) = max(E-S, 0)

2.4 Black-Scholes formula for European option price

The Black-Scholes formula allows us the calculate the price of European call and put options.

$$C(S,t) = N(d_1)S - N(d_2)Ke^{-rt} \tag{2.2}$$

$$d_1 = \frac{1}{\sigma\sqrt{t}}\left[\ln\left(\frac{S}{K}\right) + t\left(r + \frac{\sigma^2}{2}\right)\right] \tag{2.3}$$

$$d_2 = \frac{1}{\sigma\sqrt{t}}\left[\ln\left(\frac{S}{K}\right) + t\left(r - \frac{\sigma^2}{2}\right)\right] \tag{2.4}$$

$$N(x) = \frac{1}{\sqrt{2\pi}}\int_{-\infty}^{x} e^{-\frac{1}{2}z^2}dz \tag{2.5}$$

2.5 Example

You want to buy an IBM European call option with a strike price of $210. The stock is currently trading at a price of $208.99 . You calculate the volatility of the stock to be 17%. The rate at which you can borrow and lend money is 5% (this is the risk-free interest rate). The time to maturity of the option is 77 days. What price should you pay (per share) for the option contract?

$$d_1 = \frac{1}{0.17\sqrt{0.21095}}\left[\ln\left(\frac{208.99}{210}\right) + 0.21095\left(0.05 + \frac{0.17^2}{2}\right)\right] = 0.1123799$$

$$d_2 = \frac{1}{0.17\sqrt{0.21095}}\left[\ln\left(\frac{208.99}{210}\right) + 0.21095\left(0.05 - \frac{0.17^2}{2}\right)\right] = 0.0343001$$

Now that we've calculated d_1 and d_2 we will calculate $N(d_1)$ and $N(d_2)$. It should be noted that it's not possible to evaluate equation by normal means so you must use a table or use computer software to evaluate the following integrals:

$$N(d_1) = N(0.1123799) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{0.1123799} e^{-\frac{1}{2}z^2} dz = 0.3516077$$

$$N(d_2) = N(0.0343001) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{0.0343001} e^{-\frac{1}{2}z^2} dz = 0.3135993$$

Now we plug these in equation to calculate the price of the call option (per share).

$$C(208.99, 0.21095) = (0.3516077)(208.99) - (0.3135993)(210)e^{-(0.05)(0.21095)} \approx 2.3454$$

You should pay around \$2.35 (per share) for the options contract. As a side note, options contracts are sold in lots of 100 shares, so this particular contract would sell for roughly \$235.

2.6 Methodology

Initially we took 10 years worth of stock prices and discretized by 126, as there are 252 trading days and our data to expiry was 6 months. Furthermore we discretized the per day stock price by 100. We generated 2500 manifolds (one for each day) holding everything constant except the price of underlying. The problem with this was that we weren't changing any hyparameters.

2.6.1 Finding "strike price" parameter

The fixed price of an option at which the owner of an option can buy or sell the underlying security is the strike price [9]. Strike price is one of the most important factors when it comes to valuing an option because without the strike price it is not possible to determine whether the option is valuable or worthless. We varied the strike price by 4 standard deviations on each side and maintained a difference

of one standard deviation between each strike price value. In total we ended up with 27 different strike prices.

$$K \pm 4\sigma \tag{2.6}$$

2.6.2 Finding "Time" parameter

Time of expiration is one of the most important key factors, which has a significant effect in the stock price in BSOPM. With the passing of time, the price of options also decreases if the price of the stock does not change with it. We have considered our time of expiration to be 10 years. Furthermore in order to solve the equation using FDM we discretized the time by 10.

2.6.3 Parameters

where S is the current stock price, K is the option strike, T is option maturity, r risk free rates, respectively. Finally, is the annualized volatility, i.e., the standard deviation of the return on the stock. we simulated a range of call option prices using a range of parameters shown in Table 1

Table 2.1: The range of parameters used to simulate 32400 call option prices.

Parameter	Range
Stock Price((S)	\$20 - \$120
Strike Price(K)	\$5.17 - \$210
Maturity(T)	10 years
Risk Free rate(r)	13%
Option Price(U)	0 - \$108

3 — DMD applied to BlackScholes

The origin of the DMD algorithm was in the fluid dynamics community as a method that could be used to extract the spatiotemporal patterns from fluids data that was high-dimensional in nature. Subsequent connection that could be established between the Koopman operator's eigenvectors and DMD modes made the technique a highly promising approach to analyzing nonlinear dynamical systems, with the most famous one being the Navier–Stokes equations. For our purposes we used the Black Scholes equation, a nonlinear partial differential equation like the Navier Stokes equation. Applying the DMD method on Black Scholes equation allowed us to the reduce the dimensionality of the high dimensional solutions as it incorporates SVD and identifies linear subspace that most accurately spans the data. Not only that its usefulness lies it in providing eigen values that determine a low dimensional system in order to capture behavior as it evolves in time.

We used a snapshot-based method to prepare the data. Firstly, we flattened the two-dimensional option price data at time tk for a single strike price k into a single tall column vector. From that in order to see if our dmd algorithm was working correctly, we reconstructed a single snapshot from the modes for both linear and nonlinear behavior of black Scholes model.

4 — DMD applied to Financial Data

Dynamic Mode Decomposition is built on the mathematical techniques Koopman theory which relies on mapping finite-dimensional nonlinear dynamical system to an infinite-dimensional linear system. It is a rigorous data driven and equation free modeling strategy. (i) providing a rigorous mathematical connection with dynamical systems theory, and (ii) adaptively modeling and controlling complex, nonlinear processes

4.1 Stabillity

Financial data is non linear process, while the stock market is assumed to be a complex, dynamical system. DMD decomposes stock portfolio data into low-rank features that behave with a prescribed temporal dynamics. In the process, the least-square fit linear dynamical system allows one to predict short-time future states of the system[1] We will be applying the following constraints and investigating how it impacts conditioning of the system

- Diversifying the Portfolio
- Varying the time window of the sample data

[1] Taken from Kurtz Dynamic Decomposition mode

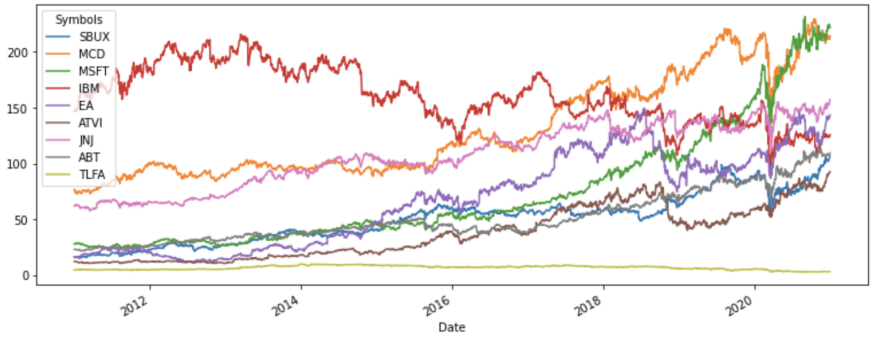


Figure 4.1: Time Series for nine stocks from the past decade
Time series demonstrates that Microsoft stock price has had the most exponential increase in past decade while the price of the leather stock has remained relatively the same.

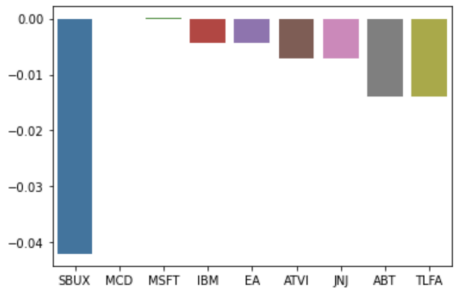


Figure 4.2: Eigenvalues

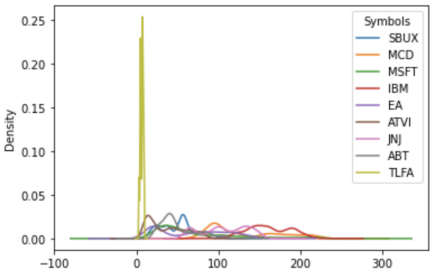


Figure 4.3: KDE

Above Figure shows the risk associated with Starbucks stock is the highest while the lowest is with McDonalds

4.1.1 Diversifying Portfolio

To diversify our portfolio we choose stocks from five industries including Gaming, leather,Technology, Healthcare and Food. In general the diversification led our matrix becoming highly dense and as a result ill - conditioned.

Industry Variation

After the diversification we tried grouping stocks from different sectors to see the impact on condition number of our matrix. On average after normalization the addition of two stocks regardless of the industry causes an increment of 25 to the

	Condition Number	Normalized Condition Number
Food, Tech	37.64	25.32
Food,Tech & Game	74.31	50.88
Food,Tech,Game,Health	656.31	80.88

Table 4.1: Impact on Condition Number on diversifying portfolio.

condition number. Non normalization causes an upward spike in the condition numbers

4.1.2 Varying the Time Window

Below are the results from a nine stock portfolio from five industries

	Condition Number	Normalized Condition Number
Two years	1026.312	119.737
Five years	665.246	85.050
Ten years	378.069	65.060

Table 4.2: Impact on Condition Number on increasing time frame.

As we increase the time window of the sample data, our problem becomes better conditioned and the matrix structure that is being built improves significantly and more effectively captures the structure for each stock

4.2 Code Listings

```
start = datetime.datetime(2011, 7, 30)
end = datetime.datetime(2021, 7, 30)
df_SBUX = web.DataReader("SBUX", 'yahoo', start, end)
```

Listing 4.1: Fetching the Data

Out[3]:

Date	High	Low	Open	Close	Volume	Adj Close
2011-07-29	20.445000	19.620001	19.945000	20.045000	31177400.0	16.906878
2011-08-01	20.385000	19.650000	20.219999	19.900000	16245600.0	16.784586
2011-08-02	19.830000	19.290001	19.705000	19.305000	16162600.0	16.282726
2011-08-03	19.660000	18.985001	19.305000	19.639999	14852600.0	16.565285
2011-08-04	19.580000	18.445000	19.375000	18.450001	20348400.0	15.561584

Figure 2.1

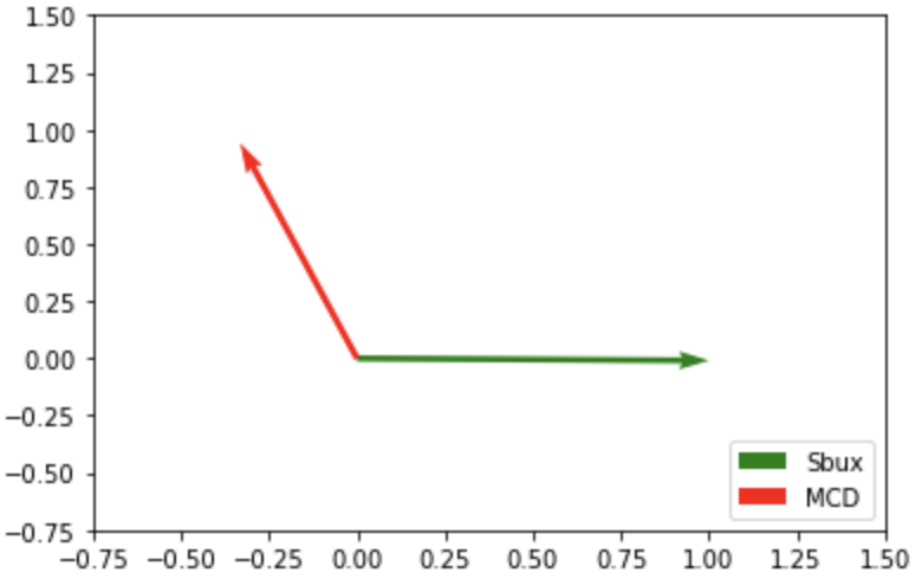
```
def cond_numb(start_date):
    df = pdr.DataReader(Stocks, data_source='yahoo', start =
        start_date, end='2021-01-01')['Close']
    X = df.T.values
    X_normalized = preprocessing.normalize(X, norm='l2')
    return "X:", la.cond(X) ,"Normalized X:",la.cond(X_normalized)
```

Listing 4.2: Condition Number

A — Appendix

A.0.1 Figures

Figure A.1: Normalized Eigenvectors multiplied by Eigenvalues of Two stocks



A.0.2 Flow Diagram

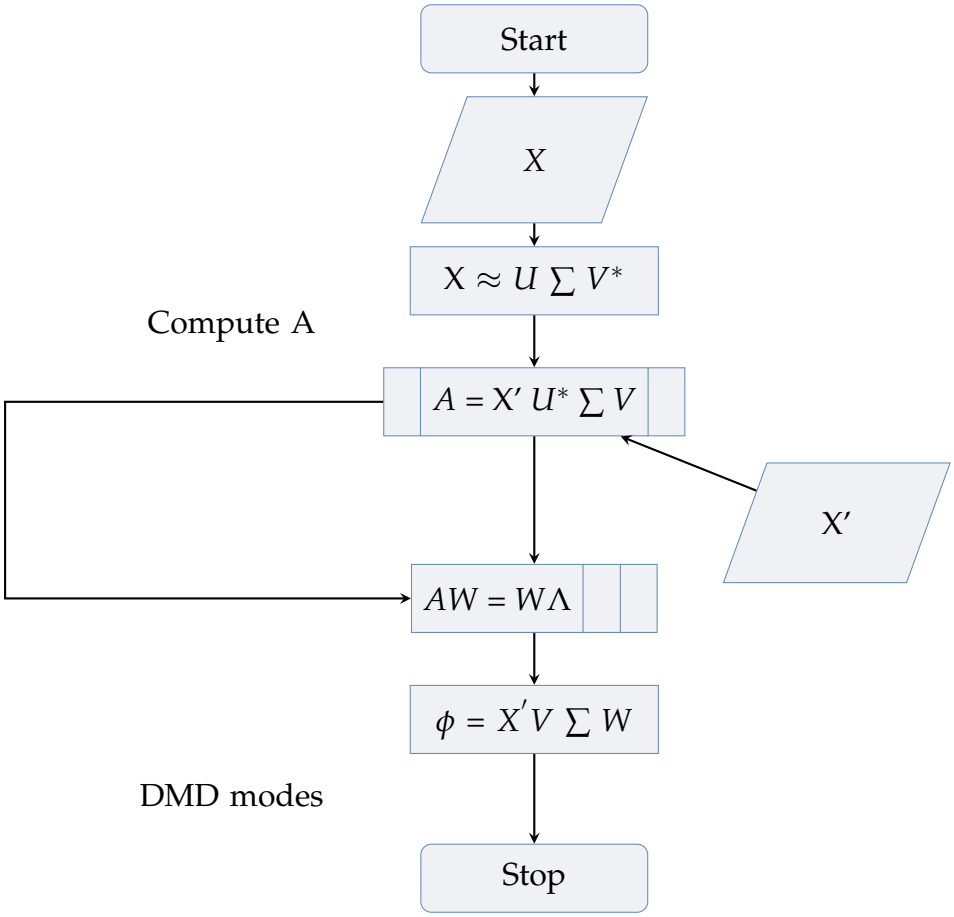


Figure A.2: Flow Diagram

*Source: [***mus:16***]*

A.0.3 Algorithims

Listing A.1: computing DMD modes

```
X1 = np.delete(X, -1, 1)
X2 = np.delete(X, 0, 1)

#Normalizing X1 and X2
X1_normalized = preprocessing.normalize(X1, norm='l2')
X2_normalized = preprocessing.normalize(X2, norm='l2')

#Singular Value Decomposition
U, s, vh = np.linalg.svd(X1_normalized, full_matrices=True)
S = np.diag(s)

#Reshaping
UT = U.T
V = vh.T
arr = np.zeros((2,2515))
ST = np.concatenate((la.inv(S),arr), axis =1)

#Computing rank reduced A
Atilde = UT @ X2_normalized @ V @ ST.T

#Eigen Decomposition
eig_vals, eig_vecs = la.eig(Atilde)

#DMD modes
phi = X2_normalized @ V @ ST.T @ eig_vecs
```