**Yield Curve & Optimal Fly**

A research funded by the Institute for Advanced Teaching and Learning at the University of Warwick

Student researcher: Bogdan-Remus Pintilie

Supervisor: Prof. Zeynep Kurter

Repository: https://github.com/BogdanRemusPintilie/Yield\_Curve-and-Optimal\_Fly

September 2024

© Bogdan-Remus Pintilie, 2024

**Abstract**

The yield curve, also known as the term structure of interest rates, and its behaviour are two of the most monitored aspects that economists, investors, traders, and other fixed income enthusiasts pay attention to. Predicting the yield curve with extreme accuracy is like finding the Holy Grail of fixed income trading and investing and a lot of effort and capital are put behind such an endeavour. This report covers my efforts to model the term structure of the US treasuries yield curve, forecast its evolution, and choose a profit-maximizing options fly that would reward me for my work. There are several techniques to model and forecast the yield curve. This report will cover the Principal Component Analysis (PCA) method using Singular Value Decomposition (SVD), the AutoRegressive Integrated Moving Average time series model (ARIMA), the Long Short-Term Memory machine learning model (LSTM), and the Lasso Regression to predict the yield curve. The report covers the steps I took to identify the optimal (i.e., profit-maximizing) fly without getting into the above code. A fly or butterfly is a delta-neutral limited risk options strategy preferred for its fixed and known downside and upside. The research shows that the first two principal components, PC1 (shift) and PC2 (slope), account for the most variance in the yield curve's evolution from October 2022 to June 2024. The research has also shown that a rolling ARIMA has a better than a vanilla LSTM for modelling predictor variables used to model the PCs. The conclusion is that the term structure of the yield curve is predicted to be below the current levels for all maturities. The optimal butterfly indicated by the research is the 1Mo/1Yr/10Yr butterfly, but the result highly depends on the data used and the quality of the models created.

**Yields**

Now, the yield that we refer to here is also known as the yield to maturity (YTM), and I will use these words interchangeably. The YTM is the annualised return anticipated on a bond if held until it matures. We also assume the coupon will be reinvested at the same rate (i.e., compounding). The YTM is the internal rate of return (IRR) on a bond (i.e., the rate for which the bond’s NPV equals 0). Because YTM is an annualised value, it allows for meaningful comparison between treasuries/bonds, etc. Two bonds with the same price could have very different characteristics (e.g., different coupon rates and maturities), which would affect the overall return to the investor. You will most likely get yields and not prices when searching for the current values of a bond, and now you know why. To explain what annualised return means, suppose the 2-year treasury has a 2% yield. If you were to hold that treasury for its entire duration, you’d get 2% on your investment each year. So, your total return would be 2% in the first year and 2% in the second year, which means an overall return of 4% when the treasury/bond matures.

The Fed controls the federal funds rate, displayed as an annualised value and applied to overnight lending/borrowing between financial institutions. This rate has no direct relationship with any yield on the curve, but it does influence them.

**Yield curve**

The yield curve is a graphical representation of the yields recorded/offered for each maturity plus some interpolation (see Figure 1). You need to interpolate to fill in the gaps since the US Treasury will only issue yields for a handful of maturities: 1 Mo, 2 Mo, 3 Mo, 4 Mo, 6 Mo, 1 Yr, 2 Yr, 3 Yr, 5 Yr, 7 Yr, 10 Yr, 20 Yr, 30 Yr. To interpolate, I used the CubicSpline method in Python to create polynomials for each interval (2 consecutive observed data points). The method solves the equations for you and derives the parameters of the polynomials. To evaluate the yield for maturities not given, the method np.linspace generates equally distanced points following the polynomial of each interval. In a normal upward curve, the short-term yield is below the long-term one. In a flat yield curve, yields are equal across maturities. In an inverted yield curve, the short-term yield is above the long-term one, as shown in the below picture.

A graph with a line graph and numbers

Description automatically generated

Figure 1. US Treasury Yield Curve (2024-06-01), Data Source: U.S. Department of the Treasury

**PCA**

PCA helps reduce the data's dimensionality while capturing the most significant patterns. When I say dimensionality, I mean the number of features/columns (maturities in our case) on which we have data. PCA will tell which features are most valuable for clustering the data. To give you an intuition of what the PCA does, suppose our data frame contains observations of two maturities over each month from October 2022 to June 2024. We want to plot the data points on a 2D graph. Say that 3 Mo is the x-axis (spans one dimension of the graph) and the 1 Yr is the y-axis (spans the other dimension). Figure 2 plots all the points representing the combination of yields of the two maturities for all given months.

A graph with blue dots

Description automatically generated

Figure 2. 3 Mo and 1 Yr PCA

This is a 2D plot, but imagine adding another maturity. You would have a 3D plot. In our case, we have 13 maturities, so you would get a “13D plot”. 3D is the most we can capture by a visualization. PCA is useful because it helps reduce dimensions while limiting errors (coming from overlooking dimensions) so that we can visualize the data. PCA also determines which dimension is the most valuable for clustering data points. If I calculate the average yields of the two maturities, I can centre the points (see Figure 3).

A graph with blue dots and red lines

Description automatically generated

Figure 3. 3 Mo and 1 Yr centred points

Then, you have to fit a line through the points, and there are several ways to look at it. The PCA will take the projection of the points on a random line that is first drawn (intersecting the origin). It then minimises the sum of the squared distance of the projections to the origin and rotates the line to achieve that. As an aside, the eigenvalue is the average of the sum of squared distances and tells you a vector’s magnitude. The eigenvector tells you the direction of the divergence. The fit line is called the Principal Component 1 (PC1). Once you have the line, you can then get the slope of PC1.

Now, you want to get the unit vector/eigenvector of PC1, which is done by dividing by the length of the vector given by the slope. This is to standardise values. The reason for this standardisation is to derive meaningful info because numbers between 0 and 1 can represent percentages, but without standardisation, it’s hard to derive information. The slope tells you the linear combination of the two axes. Standardising tells you how much of each axis contributes to the dispersion of the points.

PC2 is the perpendicular to the PC1 at the origin. This gives you the slope of PC2, and you derive the eigenvector of PC2. Then, projecting the points onto PC2, you will get the eigenvalue for PC2. By rotating the PCs to become the axes of a graph and knowing the coordinates of the points, you replot the points. This time around, they are standardised, so you will see the variation on each axis better. This step is not necessary, but it helps you visualize the variation more easily.

Eigenvalues are measures of variation. You know how much a maturity accounts for the total variation in the data (total variation = sum of eigenvalues across PCs). So, the eigenvalue of PC1 divided by the sum of eigenvalues gives you a percentage of the total dispersion of PC1. The goal is to select the PCs with the highest contribution (%) to total variation. In a 3D graph, if 2 PCs account for 99% of the variation, then plotting a 2D using those 2 PCs is a pretty good representation of the variation in the data points. Suppose you work with 3PCs. By the way, PC3 is perpendicular to PC1 as PC2 since you have three dimensions. If PC3 does not account for much of the variation, then erase it, project the points on the remaining PCs, and continue working with 2 PCs.

The PCA tells us what we must focus on to account for the variation of data points as best as possible, given a set constraint on the dimensions we can visualize. We try to simplify a problem while not creating large errors. It is a tradeoff between simplicity and accuracy, and in my research, I found that the first 3 PCs explain 98.66% of the total variation in the yield curve (see Figure 4). It is important to note that PC1 is the shift in the yield curve, PC2 is the slope of the yield curve, and PC3 is the curvature of the yield curve.

A graph with blue squares

Description automatically generated

Figure 4. Variation explained by the top 3 PCs

**Modelling the PCs**

I decided to model the three PCs using exogenous variables. This is because any model that uses past observations of the PCs to predict their future values contradicts the Efficient Market Hypothesis (EMH), which asserts that all information is factored into the pricing of products (in this case, the yields of the treasuries). This means there is no predictability because only new information can bring changes. I used a multitude of exogenous variables in my regression analysis. Some of them include: Inflation, The Federal Funds Rate, M2, VIX, 10-Year Breakeven Inflation Rate, and more. Please check the “Exogenous variables” Excel file for the full list. The frequency of the variables I gathered was not identical, and I decided to work with monthly data. This is the main reason I worked with monthly yield values: to create a data frame without dealing with indexing issues. Did this lead to imperfections? Perhaps, but the same would have happened with keeping the lowest frequency (daily) and interpolating daily values for the exogenous variables missing this daily frequency, which could raise stationarity issues. I care about the stationarity of these exogenous variables because I want to predict their values, and stationarity is a critical requirement for working with time series models.

Having finished the data collection and transformation for the exogenous variables I accounted for (not a complete list), I merged them with the values of the PCs. I then ran a Lasso regression (alpha=0.1) to identify which exogenous variables are helping model the PCs. The results of the Lasso regressions are:

PC1 (=0.914) = -0.1644 + (-0.5608) \* Federal Funds Rate + (-0.2959) \* 5-Year, 5-Year Forward Inflation Expectation Rate + (-0.1902) \* Moody's Seasoned Aaa Corporate Bond Yield + ***(-0.1833) \* Liabilities and Capital: Liabilities: Deposits with F.R. Banks, Other Than Reserve Balances: U.S. Treasury, General Account: Week Average*** + (-0.0292) \* 30-Year Fixed Rate Mortgage Average in the US + (0.0148) \* Inflation (the highlighted exogenous variable was eliminated due to its high predicted value which would make PC1 unreasonably high)

PC2 (=0.690) = -0.0746 + (0.1901) \* 30-year Breakeven Inflation Rate (%) + (-0.1895) \* Personal Saving Rate + (0.1300) \* 7-year Breakeven Inflation Rate (%)

Unfortunately, no combination of the exogenous variables I used resulted in a satisfactory (barely positive, if not negative, scores). As a result, I decided to abandon PC3, which only accounted for 2.78% of the total variation.

**Modelling and predicting the relevant exogenous variables**

Once I knew the exogenous variables that could be used for modelling, I needed to predict their value in six months. For this, I used an ARIMA model or an LSTM model. The ARIMA model used a rolling prediction method. The LSTM model uses the Adam optimization algorithm to assign weights to the inputs, hidden states, and biases such that the loss function (i.e., Mean Squared Error) is minimized during training. The hyperparameters of the ARIMA models (the AR and MA arguments) were optimized using the Bayesian Information Criterion (i.e., the optimal AR and MA combination minimizes the BIC value). Some hyperparameters of the LSTM models (the batch size and number of epochs) were optimised using KFold Cross Validation. Additionally, I set the seeds for Python’s built-in random module, NumPy, and TensorFlow, ensuring that operations involving randomness are reproducible. This includes the initialization of weights in the neural network. This happened because there were significant differences in output with each code run. The only exogenous variable for which I haven’t used a model to predict its value was the Federal Funds Rate because I calculated the expected/predicted value given probabilities of intervals for my horizon taken from CME FedWatch. I calculated the expected lower and upper bounds and then took the interval average. As for the other relevant exogenous variables, I modelled each using an ARIMA and an LSTM to observe how well each model predicted. Here, I am referring to predicting values for months that I already had data on, but the models weren’t trained on that data, so it is unseen to them. I split my data into training and testing and kept the same 6-month horizon in my test data for both ARIMA and LSTM. I then got the predictions and calculated each model's Root Mean Squared Error (RMSE). If an exogenous variable was not stationary, I did the first difference. If the first difference was not stationary, I used an LSTM. If the RMSE of the LSTM was better than ARIMA’s, I used an LSTM. Otherwise, I used an ARIMA model and returned to the original values if I took the first difference.

**The predicted Yield Curve**

Once I obtained the predicted values of the exogenous variables for the two formulas, I input them into their corresponding formulas and calculated the predicted values of the first 2 PCs in 6 months. I then used those two values to revert the PCA process and return to yields. I applied the loadings of the PCs and added back the mean yield for each maturity to cancel the standardisation. I then interpolated to get a smooth curve. Maturities were converted in years to allow for interpolation. This was the result:

A graph with a line and a line

Description automatically generated with medium confidence

Figure 5. The predicted Yield Curve

I then wanted to compare this curve to the latest one to interpret the result (see Figure 6).

A graph of a number of yield curves

Description automatically generated

Figure 6. The current vs. the predicted yield curves

What I can draw from this picture is that the predicted yield curve sits below the most recent one. This implies a downward shift in the curve. I also see that both curves are inverted. What could cause such a downward shift? Probably Federal Funds Rate cuts. When this plot was created, investors and traders weighed in on Fed cuts, with an overwhelming consensus of at least a 25 bps cut in the next meeting on September 18th, 2024. Then, there would be two more meetings on November 6th-7th, 2024, and December 17th-18th, 2024. Chances are that the Fed will start cutting the Federal Funds Rate, and by year’s end, the last 3 cuts could amount to 100 bps. Overall, I can see why the yield curve should sit below the current values, but I can’t be certain of the magnitude of the shift.

**Finding the optimal fly**

A fly consists of a long position in a short-term maturity, a double short position in a middle-term maturity, and a long position in a long-term maturity. The short, middle, and long-term meanings must not be taken at face value. I used them to explain the order of the positions. For example, 1Mo/1Yr/10Yr is a valid butterfly, and so is the 1Mo/10Yr/30Yr. To get to the best combination, I needed to solve an optimisation problem, saying that I wanted to maximise the relative change given the structure of the butterfly. So, first, I had to find all valid combinations of maturities that could be formed and then find which one of those valid combinations results in the greatest relative change. I mention relative because the yield curve has shifted downward for all maturities. So, it is inevitable that if I have a long and short position on different maturities, one will make me money, and the other will lose me money. So, I calculated the percentage change in the yield for each maturity and used that information as input to solve the optimisation problem. It is also important to notice that yield and price have an inverse relationship. So, if the change in yield is negative, this means that the yield has fallen, and this means that the price has increased. In this case, I cared about the price of the treasury. Once I solved the optimisation problem, I received the optimal position to build now for a holding period of 6 months, and that is the 1Mo/1Yr/10Yr fly. This result is very specific to the data used. Everything has an influence, from the initial timeframe used for the yields to the number of exogenous variables used to the choice of models in the forecasting stage to the length of the holding horizon and even the quality of the work that I have done. I want to make it abundantly clear that this result does not constitute investment or trading advice. You are fully liable for any losses or damages resulting from actions taken based on the information provided. The main goal of this research was to explore one way of tackling the interesting yet complex task of modelling and forecasting the yield curve.

**Further considerations**

Given more time, I would give more thought to a reasonable penalty in the Lasso regression, account for more exogenous variables, reduce stationarity concerns caused by outliers, and find a way to model PC3. Additionally, some research papers use the Nelson-Siegel-Svensson model (NSS) to model the yield curve, and I would follow a similar approach to contrast the results.