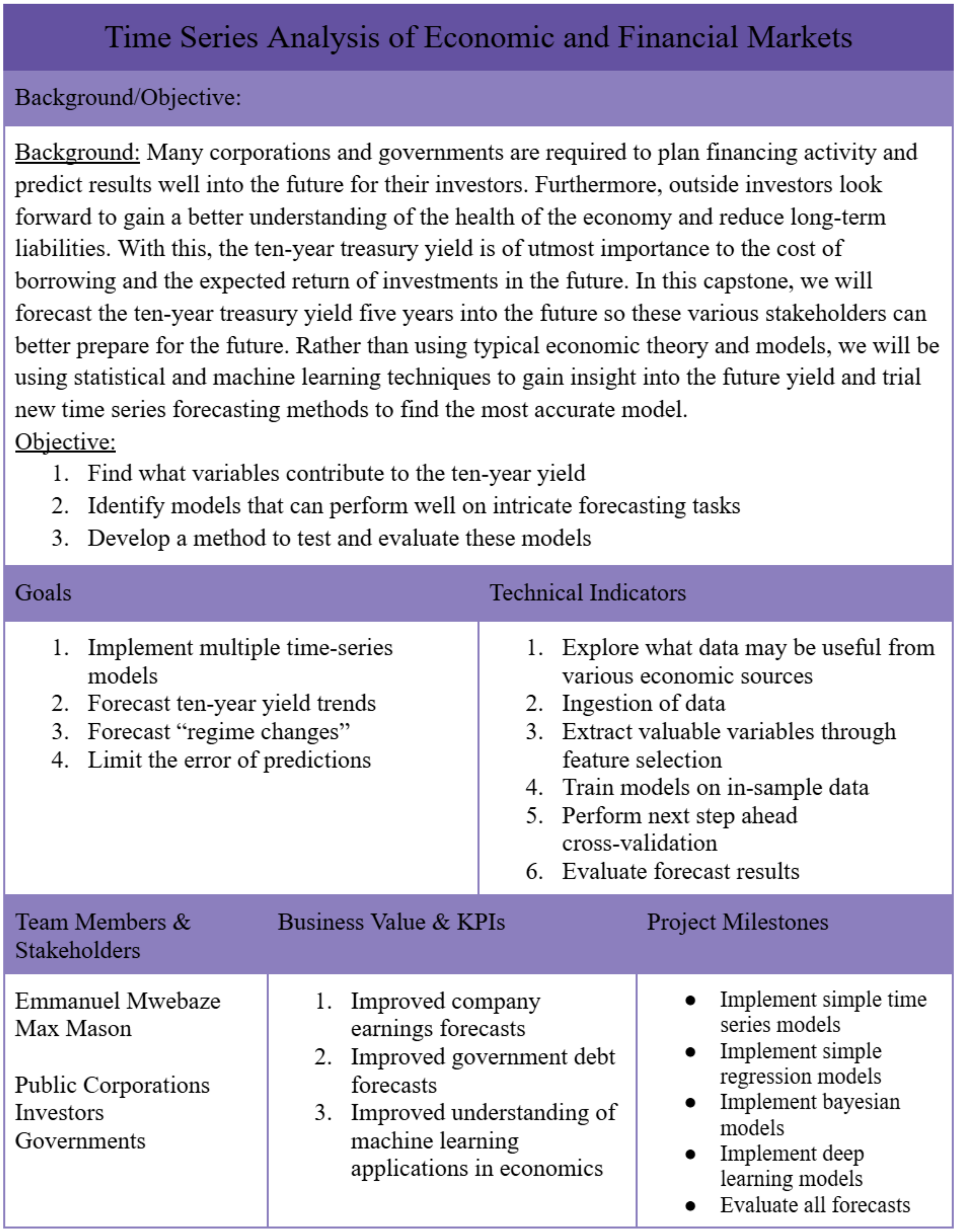
Final Report - Time Series Analysis of Economic and Financial Markets

Max Mason (mmm2491), Emmanuel Mwebaze (emm2293)

December 19th, 2024

1 Executive Summary Page



Time Series Analysis of Economic and Financial Markets

Background/Objective:

Background: Many corporations and governments are required to plan financing activity and predict results well into the future for their investors. Furthermore, outside investors look forward to gain a better understanding of the health of the economy and reduce long-term liabilities. With this, the ten-year treasury yield is of utmost importance to the cost of borrowing and the expected return of investments in the future. In this capstone, we will forecast the ten-year treasury yield five years into the future so these various stakeholders can better prepare for the future. Rather than using typical economic theory and models, we will be using statistical and machine learning techniques to gain insight into the future yield and trial new time series forecasting methods to find the most accurate model.

Objective:

1. Find what variables contribute to the ten-year yield
2. Identify models that can perform well on intricate forecasting tasks
3. Develop a method to test and evaluate these models

Goals	Technical Indicators
1. Implement multiple time-series models 2. Forecast ten-year yield trends 3. Forecast “regime changes” 4. Limit the error of predictions	1. Explore what data may be useful from various economic sources 2. Ingestion of data 3. Extract valuable variables through feature selection 4. Train models on in-sample data 5. Perform next step ahead cross-validation 6. Evaluate forecast results

Team Members & Stakeholders	Business Value & KPIs	Project Milestones
Emmanuel Mwebaze Max Mason Public Corporations Investors Governments	1. Improved company earnings forecasts 2. Improved government debt forecasts 3. Improved understanding of machine learning applications in economics	● Implement simple time series models ● Implement simple regression models ● Implement bayesian models ● Implement deep learning models ● Evaluate all forecasts

2 Introduction

2.1 Motivation

Various entities such as insurance companies, pension funds, and governmental borrowers manage long-term liabilities and finance activities over extended periods. It is therefore of utmost importance for them to accurately predict future financial trends over long horizons. Additionally because of the extended period of investments that they typically engage in, one of the major metrics by which they budget capital and make decisions is the yield of the longer maturity U.S. Treasury Securities. This project therefore focuses on the time series analysis of economic and financial data to predict trends in the U.S. Treasury Yield, emphasizing the 10-year maturity US Treasury yield. This yield is critical because it impacts a wide range of market participants.

2.2 Literature Review

Many studies on forecasting treasury bond yields focus on making predictions based solely on the evolution of the term structure of interest rates. One example of this is Diebold and Li (2006) who produced a three-factor auto-regressive model that displays robust out-of-sample performance. Modelling based on the whole term-structure provides additional benefits especially in contrast to modelling single tenors independently e.g. developing separate models for the 10 year note and 2 year note, especially since the latter can result in predictions that significantly diverge from each other and term structures that do not conform to economic intuition and real market movements. However, given the scope and resources of our academic project, we have strategically chosen to focus on a single 10-year maturity tenor over a suitable horizon of 5 years to deliver precise and actionable insights, rather than a complete yield curve model.

These constraints similarly steered us away from the autoregressive approach of Diebold and Li whose model only goes out to a 12-month prediction window, which is insufficient for our task which requires long prediction horizons. Additionally, as we will see later, the time series of the 10 year U.S. Treasury only shows a short lag of statistically significant (partial) autocorrelation, necessitating us to look for more predictors and methods outside of purely autoregressive time-series based models.

In this regard, we borrow from the robust scholarship that exists tackling this forecasting problem by incorporating a time-series analysis of macroeconomic features. For example, Fletcher and Gulley's (1996) review of Mishkin (1984) highlights the limitations of Mishkin's ARMA model predicting real interest rates by stating that "forecasters can only use information that is available to them at the time of the forecast". Their study, which leveraged a model that leverages macroeconomic variables (like lagged inflation and fuel shocks) from 1959 to 1992, utilized the rolling window methodology over a one-month window, highlighting its significance for modeling real interest rates.

Lastly, in expanding our model search space, we were inspired by Shu and Chu (2021)'s methodology that effectively utilized deep learning techniques for predicting the 10-year US Treasury yield, achieving a low mean squared error (MSE) of 0.0063. This was accomplished by using a Long Short-Term Memory (LSTM) network, and using yields of various US Treasury securities of different maturities as inputs. However, this study focused on leveraging the term structure by using multiple Treasury maturities with a short time horizon. Our study is inspired to also explore deep learning techniques, but will differ from this approach by emphasizing macroeconomic variables due to our long-term forecasting horizon, and the potential issues of relying only on term structure data for long-term predictions

3 Methods

3.1 Data Collection

Given the insights above, our project leverages time-series data obtained from the Federal Reserve Economic Database (FRED), the United States of America Congressional Budget Office (CBO) and Department of the Treasury for our task. An overview of the variables included for analysis are included in Table 1.

The time series data spans from 1986 to 2024, and for some variables that appear after 1986, we create a separate dataset from 2003 onward. These years were chosen due to the significant economic events impacting macroeconomic variables, for example significant shifts in inflation, nominal yields, oil prices, etc, providing a robust framework for analyzing shifts in market conditions. The choice of a wide variety of macroeconomic variables is noted because we would like to include all information known to forecasters at the time of forecasting, while allowing the model to pick significant variables at prediction time. We opted for monthly data frequency as it strikes an optimal balance between detail and manageability; most economic indicators relevant to our analysis are published on a monthly or quarterly basis; however, monthly data allows for more granular trend detection and robust long-term projections. Therefore, for more granular data than monthly, e.g. the dependent variable, we use the arithmetic mean for the month, and for less granular variables, we impute the previous estimate as the monthly value. Data was accessed through the Federal Reserve Economic Data (FRED) API using the 'fredr' package in R, which allowed for efficient downloading and updating of economic data series. The CBO budgetary data and its projections were sourced from its GitHub repository (U.S. Congressional Budget Office) and Treasury data from its quarterly refunding document website (U.S. Department of the Treasury).

3.2 Analysis and Preparation

This section delves into the comprehensive analysis and preparation of the dataset, with a specific focus on the 10-year Treasury yield, lagged by 60 months, which serves as our primary dependent variable. To understand its dynamics and suitability for time series forecasting, we first visualize its trends, followed by a detailed examination of its statistical properties.

Figure 1 shows the time series of the 10-year Treasury yield by month to capture its overall movement and identify any apparent trends or cyclicality. We note the overall decrease in 10-year yields as the decades progress from 1980, hitting particular lows in the 2010 decade and early 2020 before seeing a subsequent rise up to the current level of 4.2%. From the sample visualization, we note that the trend does not exhibit significant cyclical trends per year.

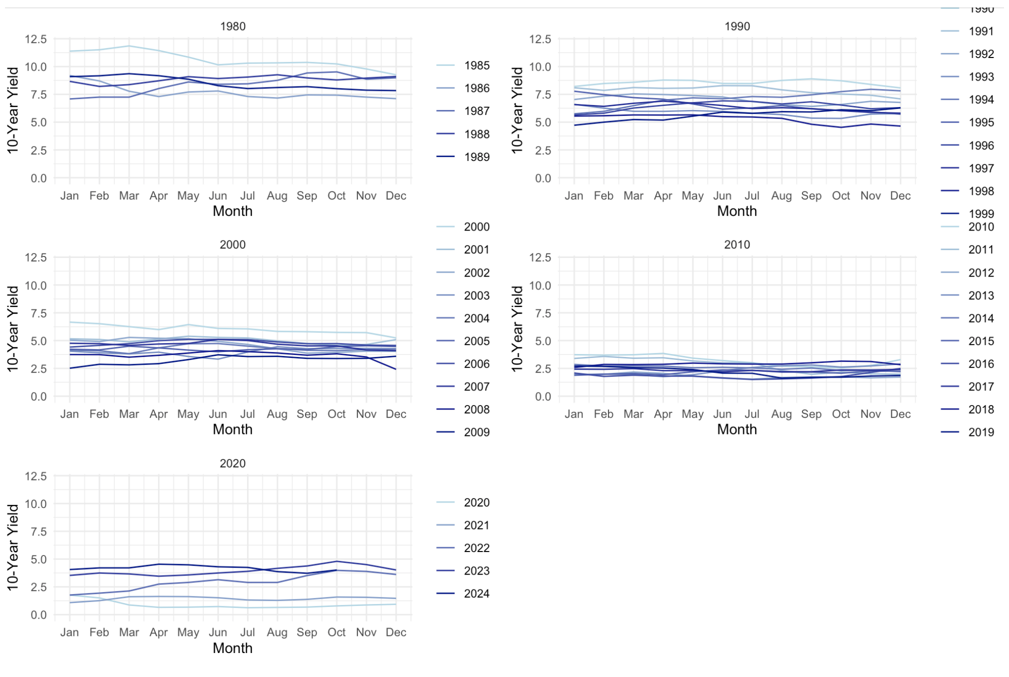


Figure 1: Time Series Plot of the 10-Year Treasury Yield

To further our understanding of the time-dependent structure inherent in the 10-year yield data, we analyze both its autocorrelation function (ACF) and partial autocorrelation function (PACF).

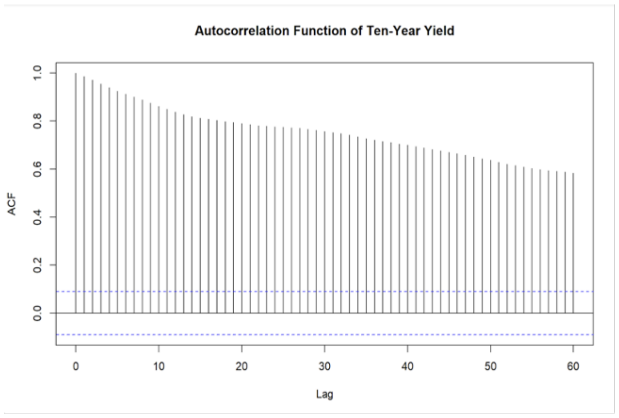


Figure 2: ACF Plot of the 10-Year Treasury Yield up to 60 lags

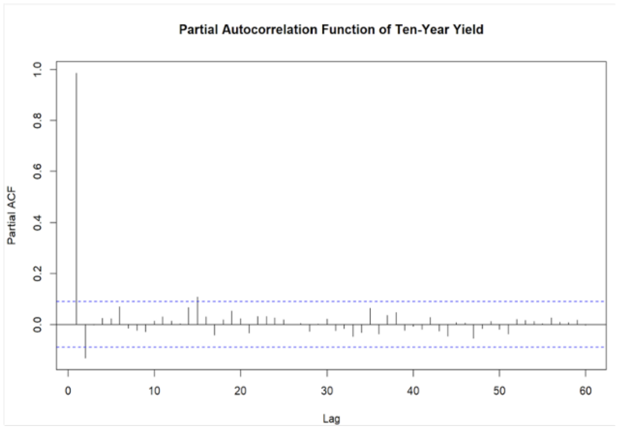


Figure 3: PACF Plot of the 10-Year Treasury Yield up to 60 lags

The ACF (Figure 2) shows significant autocorrelation, maintaining a level above 0.5 for up to 60 lags, suggesting a strong, persistent temporal dependency. In contrast, the PACF (Figure 3) is confined within a narrow range of 0.1, and only the first two lags are notably significant, which do not persist up to the 5-year horizon. This distinction suggests that while the yield exhibits strong autocorrelation and that the US Treasury yield is a good predictor to itself, much of this dependence diminishes when conditioned on its recent past values, highlighting the limited predictive power of past yields alone especially on a very long horizon. Finally, an Augmented Dickey-Fuller (ADF) test showed the presence of stationarity at the 5% significance level at all 60 lags, indicating no need for co-integration of this variable.

3.3 Feature Engineering

Due to the nature of our data, we were required to work with features with multiple updating time periods. A majority of the data that was collected was monthly but we also had daily data, quarterly data, as well as some yearly data. To combine these uneven time frames while preserving as much of the temporal dependencies as possible, we elected to transform all data to a monthly period. Daily data was aggregated by calculating the monthly averages while for quarterly and yearly data a back filling approach was used to maintain our data being point-in-time.

This approach inevitably led to some loss of granularity in our target variable, the transformation was still able to preserve a majority of its variability, as demonstrated in Figure 4 below. This plot overlays the original daily data with the aggregated monthly data to visualize this loss of granularity but to also demonstrate the retention of the trend and variability.

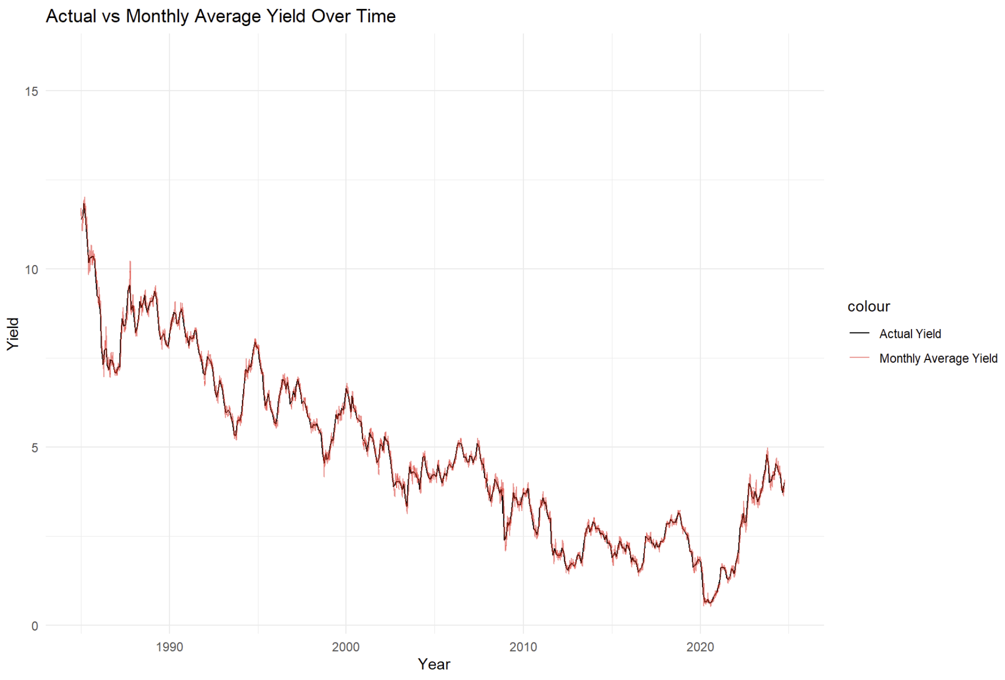


Figure 4: Comparison of Daily and Monthly 10-Year Treasury Yield Data

While some information was lost in the aggregation process, this approach was justified given our objective forecast time horizon of five years, which does not necessarily require such granular data and helps reduce the risk overfitting. Furthermore, this method enabled us to engineer additional features we could include in our models such as the monthly standard deviation, start-of-the-month value, end-of-the-month value, maximum value, and minimum value.

Lastly, for the independent variables, we plot a correlation plot to identify variables and remove features with a high correlation. Initially, we select features with a correlation threshold of 0.7 and select the features displayed above, and in the correlation matrix in Figure 5. Our correlation analysis led us to remove highly collinear variables such as GDP, PCE Price Index, Unemployment Rate, M2 money stock variables as well as medium and long term yields and inflation expectations due to their potential for multi-collinearity.

4 Modeling Approach

4.1 Sampling Techniques

Due to the nature of our data, we cannot take a simple random train-test split, but rather, are required to maintain the temporal ordering of our data to ensure that at a given point in time for which we are testing, our model does not contain any future information. To accomplish this task various sampling methods were used.

4.2 Sampling Techniques

Due to the nature of our data, we cannot take a simple random train-test split, but rather, are required to maintain the temporal ordering of our data to ensure that at a given point in time for which we are testing, our model does not contain any future information. To accomplish this task various sampling methods were used. We began with a naive train-test split, where we divided the dataset into two portions: the training set on the first 75% and the testing set on the remaining 25%, as seen in Figure 6.

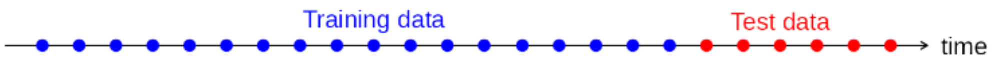


Figure 5: Diagram of Naive Train-Test Split

While this approach is straightforward and respects the chronological order of the data, it doesn't allow the model to update to new information as it becomes available similar to what may occur to a forecasting model in production. This leads to severe overfitting to the training data's trends and variations, resulting in poor performance on the test data due to this lack of adaptability.

Expanding Window Train-Test Split[1]

Next, we utilized an expanding window approach. This method mimics real-world forecasting scenarios, starting with an initial training set and progressively extending the training window by one point at a time, making a single prediction at each step on the out-of-sample data, as seen in Figure 7.

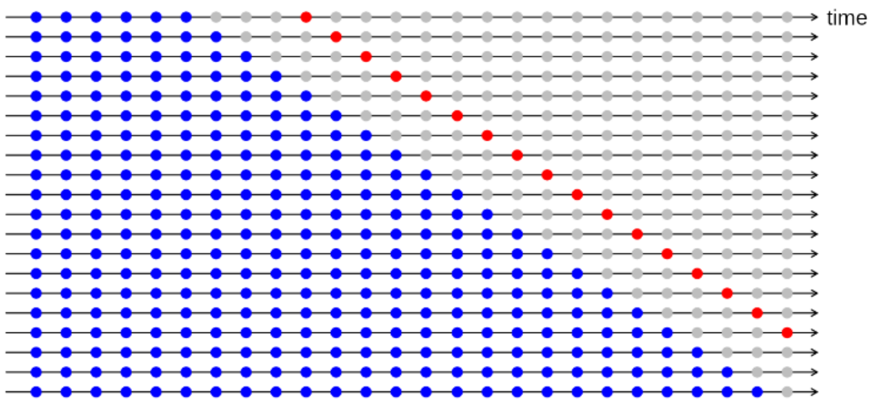


Figure 6: Diagram of Expanding Window Train-Test Split

Note the gap between the training data and the testing point, this occurs when the forecast is multiple time steps in the future. This is similar to our use case, but instead of three time steps shown here, our prediction is 60-steps ahead. While this technique allows the model to adapt, once the training set is large enough, it may overfit and struggle to react to changes such as those seen in economic data during the COVID-19 pandemic and other recessions.

Sliding Window Train-Test Split[2]

We also implemented a sliding window approach. Similar to the expanding window, it starts with an initial training set but maintains this fixed training set size by omitting the oldest data points as new ones are added and making a prediction on an out-of-sample data point as seen in Figure 7.

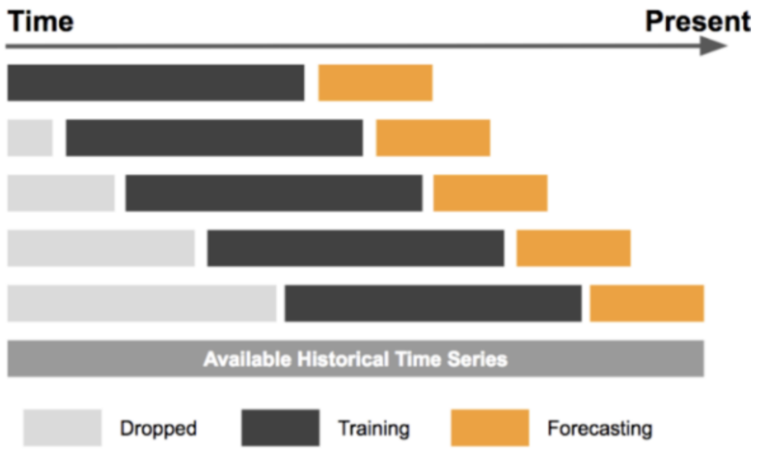


Figure 7: Diagram of Sliding Window Train-Test Split

This method strikes a balance by capturing recent data patterns and being more responsive to changes, although it may struggle with long-term trends due to older data points being discarded.

For these types of rolling train-test splits, we make one prediction at each step which is then saved in a list of predictions. We can then calculate the total test error comparing this list of predictions to the actual values.

Nested Sliding Window Train-Test Split

The last method we used was the nested sliding window approach. This method builds upon the sliding window approach, which we found outperformed the expanding window approach. The initial sliding window technique involves creating overlapping segments (windows) of fixed length for the dataset, where each window was used to train the model. In contrast, the nested sliding window approach introduces an additional layer of granularity through a second inner sliding window.

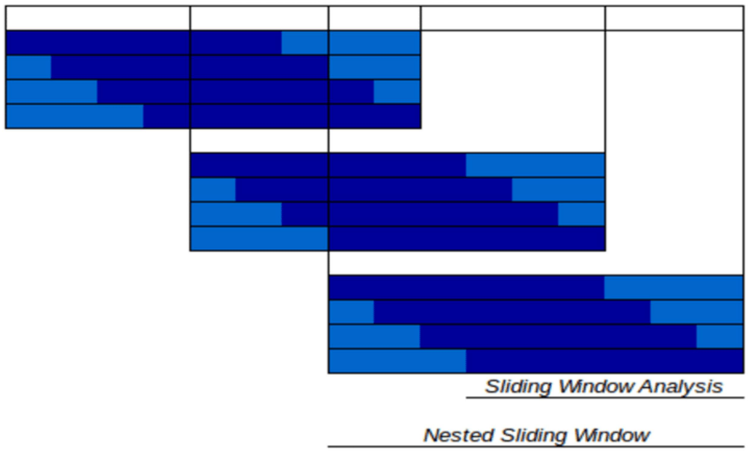


Figure 8: Diagram of Nested Sliding Window Train-Test Split

This method is comprised of an outer large window and an inner smaller window. The larger outer window defines the overall training period, while the small inner window slides within this outer window to create multiple training sets for this one large window time step. Testing is then done on the point immediately after the large window (in our case 60 time steps). Then the large window steps forward and the process is repeated.

This technique helps our models capture longer-term trends with the outer window, while the small windows enable adaptation to more recent trends and changes. Furthermore, this method is particularly effective for our use case, as we have a very limited amount of data and it increases the number of training samples we have for each large window.

4.3 Models Used

LASSO

LASSO (Least Absolute Shrinkage and Selection Operator) regression is a regularization technique that enhances model interpretability by shrinking coefficients and setting some to zero, effectively performing variable selection. It minimizes the residual sum of squares while adding a penalty proportional to the absolute values of the coefficients:

$$\min_{\beta} \left\{ \sum_{i=1}^{n} \left(y_i - \beta_0 - \sum_{j=1}^{p} \beta_j x_{ij} \right)^2 + \lambda \sum_{j=1}^{p} |\beta_j| \right\},$$

where λ is the regularization parameter controlling the trade-off between the model's fit to the data and the penalty for larger coefficients. In our analysis, we used LASSO to identify significant predictors with the highest explanatory power for the target variable. This is useful because of the large number of predictors initially which would diminish the performance of downstream models. Additionally, given its ease of interpretability, this would dbe necessary to confirm whether the yields would conform to economic intuition. At the same time, LASSO as a linear model would struggle with identifying weak but important features, interaction effects, and be prone to outliers especially given the small sample size in each regression.

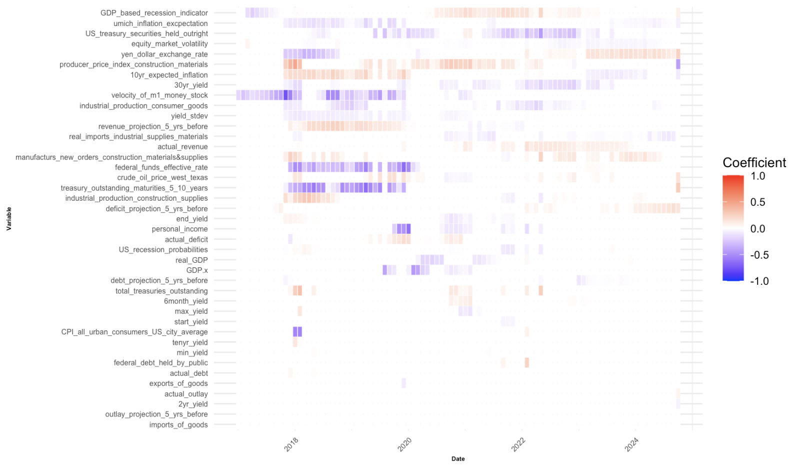


Figure 9: Coefficients for the LASSO Regression using 2003 data and sliding window

Figure 9 shows us a heatmap of the macroeconomic features selected by the LASSO model, ordered by its frequency of occurrence over the test period. We see that the current 10-year yield is among the least important factors, confirming the results from the PACF plot in Section 2. Additionally, some of the top indicators include a GDP-based recession indicator, showing the importance of tracking recession probabilities in predicting future yields, signalling its potential in assisting the model react to short-term regime shifts. We also see inflation (expressed in producer prices) and consumer inflation expectations playing a large role in the future 10 year yield, as well as the balance sheet of the Federal Reserve playing a more prominent role. Lastly, another useful feature for our modelling task is the length of the time periods at which features persist in their importance. For example, the GDP based recession indicator shows a negative correlation for around two years from 2020 to 2022, which is a similar time length to the inflation expectations feature.

Bayes Structural Time Series

The Bayesian Structural Time Series (BSTS) model is a probabilistic framework designed for time series analysis and forecasting. It decomposes the time series into interpretable components such as trend, seasonality, and regression effects while explicitly quantifying uncertainty. The model is expressed as:

$$y_t = Z_t \alpha_t + \epsilon_t, \quad \epsilon_t \sim \mathcal{N}(0, \sigma^2),$$

$$\alpha_{t+1} = T_t \alpha_t + R_t \eta_t, \quad \eta_t \sim \mathcal{N}(0, Q_t),$$

where y_t represents observed data, α_t is the latent state vector capturing dynamic components, and Z_t, T_t, R_t, and Q_t define the system's structure. Bayesian inference enables robust estimation of parameters and credible intervals for predictions.

Given our task's limited dataset (60 observations), BSTS is particularly suitable because it incorporates prior information and naturally handles uncertainty in parameter estimation. Additionally, its ability to decompose time series into components helps mitigate overfitting, making it an ideal choice for small-sample predictive modeling.

Gated Recurrent Unit

Introduced by Junyoung Chung et al.[9], the Gated Recurrent Unit (GRU) was originally built for Natural Language Processing (NLP). However, due to its recurrent nature and the similarities between Natural Language Generation (NLG) and forecasting, this model has become a mainstay in forecasting. The GRU addresses the issues of vanishing gradients and the difficulty of learning long-term dependencies that traditional RNNs faced. It introduces a gating mechanism to regulate the flow of information: the update gate controls how much of the prior hidden state is passed on, and the reset gate determines how much past information to forget.

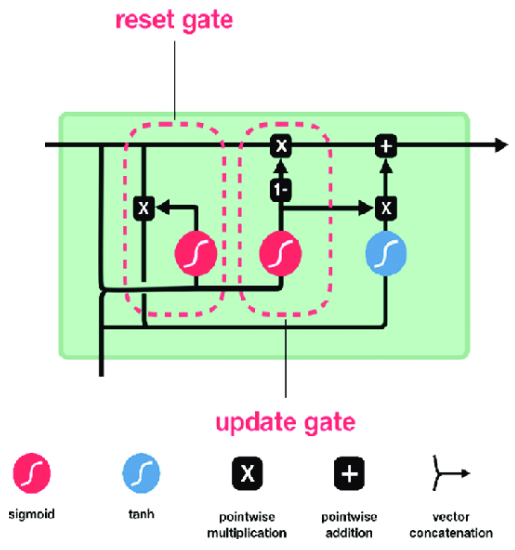


Figure 10: Diagram of a Gated Recurrent Network

This model's features that make it particularly effective for our task is its simplicity and size. Unlike other common NLG models used in forecasting, such as the LSTM and Transformer, the GRU combines its two gates and the hidden layer into a single state vector, making it computationally efficient with fewer parameters. In contrast, the LSTM has three gates separate from the hidden layer, resulting in a much higher parameter count, and the Transformer often requires millions of parameters. Both of which are too large for the roughly 300 data points we have for training.

Feed-Forward Network to CDF

This methodology was proposed to increase the amount of training data, allowing us to effectively train a feed-forward neural network without over-fitting. Instead of a regression task where we try to directly predict the target value, we transformed our problem into a classification task by applying $n = 40$ thresholds to our data. The task now becomes determining whether the target value is above (1) or below (0) each threshold. To achieve this, we add a column of thresholds that are evenly distributed on either side of the true value (20 above and 20 below in this case), resulting in each date (single observation) having $n = 40$ rows. This approach effectively multiplies the amount of training data by 40x, providing enough data to train the feed-forward network effectively.

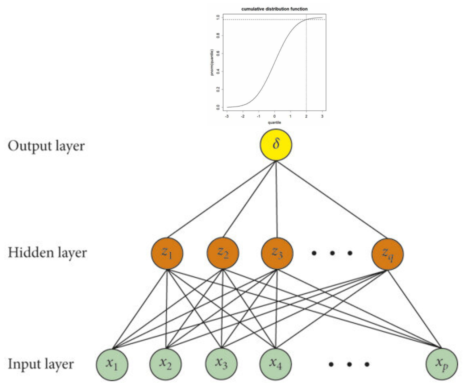


Figure 11: Simple Example Diagram of Feed-Forward Network into a CDF

What makes this model architecture particularly effective is its simplicity and the clever use of the sigmoid function applied to the model's output. By doing so, we are essentially producing the probability that the target value is greater than each threshold point. Consequently, the full n output of predictions for each date represents the Cumulative Distribution Function (CDF) of the target variable. This characteristic allows us to obtain our expected output by sampling N times from the CDF and averaging the results.

4.4 Evaluation Metrics

To assess the performance and accuracy of our models forecasts, we employed two widely recognized metrics: Root Mean Square Error (RMSE) and Mean Absolute Percent Error (MAPE). RMSE quantifies the difference between predicted and observed values, key for understanding the magnitude of prediction errors and how far our predictions deviate from the actual. MAPE is very similar to RMSE, however, it offers a more intuitive interpretation of the magnitude of prediction errors as a percentage which can help the understanding of our forecast accuracy for less technical stakeholders. RMSE is the most important metric as it gives the results in terms of the y-variable, meaning if our RMSE is 1 we know on average that our model is incorrect about the 10-year yield by 1% making it highly informative.

4.5 Results

We saw the best performance utilizing a GRU with a nested window technique. This lead to a new best RMSE of 1.265 and fixed our prior error of our model not producing auto correlated results.

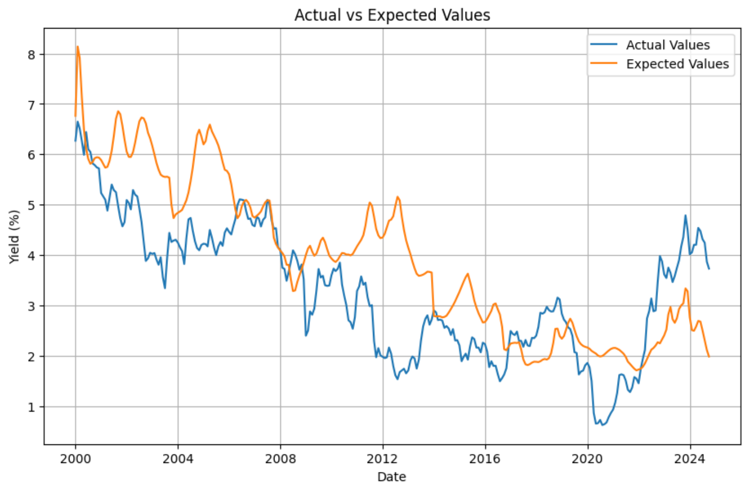


Figure 12: Testing Results of Nested Window sampled GRU

However, after observing the graph and the losses this model appears to just be mimicking the past. This can be seen later and later on as it looks more and more like the actual yield five years prior. After noticing this, further inspection resulted in the losses being near 0 in the last half of training. To combat this we tried something different where at the end of each large window in the nested loop we would discard our model and retrain a new one. This did lead to lower autocorrelation but the best results we achieved on both metrics.

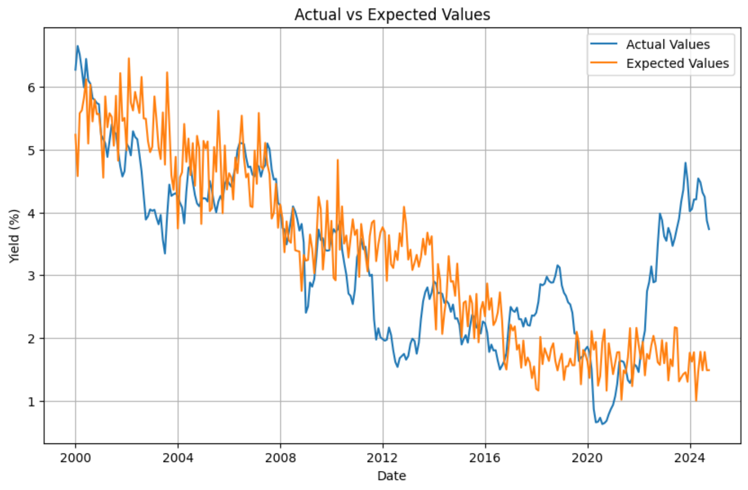


Figure 13: Testing Results of Nested Window sampled GRU with model discarding

Here we can see the model is clearly picking up overall trends but the predictions don't show high autocorrelation.

Metrics	**GRU (low-autocorrelation)**	**GRU (autocorrelated)**	**FFN to CDF**	**BSTS**	**LASSO**
Window	**Nested**	**Sliding**	**Sliding**	**Sliding**	**Sliding**
RMSE	1.084	1.265	1.589	1.42	4.29
MAPE	31.53/%	40.53%	66.45%	65.95%	129.7%

Table 1: Model Results on both Metrics

5 Discussion

This project explored various time series models to forecast the 10-year US Treasury yield using macroeconomic variables. Our analysis revealed that while the current 10-year yield exhibits autocorrelation, it is not a strong predictor of future yields, as indicated by the PACF analysis. This is further supported by our LASSO regression results that showed the current 10-year yield as among the least important factors. Conversely, our results indicated the significance of other macroeconomic indicators, such as a recession indicator, producer price inflation, consumer inflation expectations, and the Federal Reserve's balance sheet. Based on these factors, we proceeded to use some more predicitve time series models. Different sampling techniques were tested, with the nested sliding window approach showing promise in capturing both short-term changes and long-term trends, and in improving the amount of data learned by the GRU. We also observed variations in model accuracy across different economic regimes, with models exhibiting better performance post-COVID, suggesting the need for models that can adapt to regime shifts and avoid overfitting to old trends.

In conclusion, we have identified that macroeconomic features are critical for long-term treasury yield prediction, and these features can provide insight into future yields that a model based on the term structure alone cannot. The results of the LASSO regression highlighted the specific macroeconomic indicators of greatest importance, that were used to improve on the predictive results of simple ARIMA models. We also found that the choice of sampling technique is crucial, with the sliding and nested sliding window methods outperforming the expanding window and naive train-test split. For future work, we would explore the inclusion of futures pricing data to incorporate market based forecasts and test more advanced models, such as the Multivariate Bayesian Time Series Model, as well potentially transformer models. We could also use more economic

Further work is required to reduce the over-fitting of the GRU model as by discarding and retraining our model each time is not only computationally expensive, but we lose any semblance of continuity between predictions.

6 References

1 - Hyndman, Rob J., and George Athanasopoulos. "Evaluating Forecast Accuracy." Forecasting: Principles and Practice, 2nd ed., 2018, https://otexts.com/fpp2/accuracy.html (also figure 6 and 7) https://otexts.com/fpp2/accuracy.html

2 - Uber. "An Introduction to Forecasting." Uber Blog, https://www.uber.com/en-JP/blog/forecasting-introduction/. (also figure 8)

3 - IBM. "Lasso Regression." IBM Topics, https://www.ibm.com/topics/lasso-regression.

4 - DeepAI. "Lasso Regression." DeepAI Machine Learning Glossary and Terms, https://deepai.org/machine-learning-glossary-and-terms/lasso-regression.

5 - Capital One. "Understanding ARIMA Models." Capital One Tech, https://www.capitalone.com/tech/machine-learning/understanding-arima-models/.

6 - Qiu, J., Jammalamadaka, S.R. & Ning, N. Multivariate time series analysis from a Bayesian machine learning perspective. Ann Math Artif Intell 88, 1061–1082 (2020). https://doi.org/10.1007/s10472-020-09710-6

7 - Zhang, Y., Wu, R., Dascalu, S.M. et al. A novel extreme adaptive GRU for multivariate time series forecasting. Sci Rep 14, 2991 (2024). https://doi.org/10.1038/s41598-024-53460-y

8 - Wen, Qingsong, et al. "Transformers in Time Series: A Survey." 32nd International Joint Conference on Artificial Intelligence (IJCAI 2023), 2023, https://doi.org/10.48550/arXiv.2202.07125.

9 - Chung, Junyoung, et al. "Empirical Evaluation of Gated Recurrent Neural Networks on Sequence Modeling." arXiv.Org, 11 Dec. 2014, doi.org/10.48550/arXiv.1412.3555.

Other references for data and prior reading:

Diebold, Francis X., and Canlin Li. "Forecasting the term structure of government bond yields." Journal of Econometrics, vol. 130, no. 2, Feb. 2006, pp. 337–364, https://doi.org/10.1016/j.jeconom.2005.03.005.

"Federal Reserve Economic Data." FRED, Federal Reserve Bank of St. Louis, fred.stlouisfed.org/. Accessed 26 Oct. 2024.

Fletcher, Donna J., and O.David Gulley. "Forecasting the real interest rate." The North American Journal of Economics and Finance, vol. 7, no. 1, Mar. 1996, pp. 55–76, https://doi.org/10.1016/s1062-9408(96)90022-4.

Mishkin, Frederic. "The Real Interest Rate: A Multi-Country Empirical Study", Dec. 1982, https://doi.org/10.3386/w1047.

“Most Recent Quarterly Refunding Documents.” U.S. Department of the Treasury, 11 Oct. 2024, home.treasury.gov/policy-issues/financing-the-government/quarterly-refunding/most-recent-quarterly-refunding-documents.

L. Shu and J. -K. Chou, "Using Deep Learning Techniques to Predict 10-Year US Treasury Yield," 2021 11th International Conference on Information Science and Technology (ICIST), Chengdu, China, 2021, pp. 545-552, doi: 10.1109/ICIST52614.2021.9440560.

U.S. Congressional Budget Office. "eval-projections" GitHub, accessed October 26, 2023, https://github.com/us-cbo/eval-projections.