Finn 6211 - Final Project

Davis Vaughan

mvaugh15@uncc.edu

2018-04-23

Contents

1 Introduction **3**

2 Data **4**
- 2.1 Getting the data . . . 4
- 2.2 Cleaning . . . 4
- 2.3 Monthly and Ascending . . . 5

3 Fixed Income Features Calculations **6**
- 3.1 Spot Rates . . . 6
- 3.2 Zero Coupon Bond Prices . . . 7
- 3.3 One Month Returns . . . 8
- 3.4 Excess Returns . . . 8
- 3.5 Yield Curve Factors . . . 9

4 Question 1 **10**
- 4.1 Summary Statistics . . . 10
- 4.2 Autocorrelations . . . 11
- 4.3 Correlations . . . 12
- 4.4 Time Series Visualizations . . . 12

5 Question 2 **14**
- 5.1 Regression . . . 14
- 5.2 One Year Spot Rate . . . 15
- 5.3 Five Year Spot Rate . . . 16
- 5.4 Ten Year Spot Rate . . . 16
- 5.5 Decomposing the Spot Curve . . . 16
- 5.6 Coefficient Stability . . . 17

6 Question 3 **19**
- 6.1 Modified / Macaulay Duration . . . 19
- 6.2 Simple regression based hedging . . . 20
- 6.3 Multiplicative regression hedging . . . 20
- 6.4 Error calculation . . . 20
- 6.5 Model selection interface . . . 20
- 6.6 Model application . . . 21
- 6.7 Model results . . . 21

7 Conclusion **25**

Chapter 1

Introduction

This is the final project of the Finn 6211 class, with the intention of getting comfortable with spot rate data, their relation to yield curve factors, and various hedging strategies.

An R package has been created to accompany the report. It contains a number of helper functions for cleaning data, manipulating the time series, and creating the hedging strategies. The package is named `ratekit` and can be found on Github at https://github.com/DavisVaughan/ratekit.

The structure of the report is as follows. Chapter 2 is dedicated to retrieving and cleaning the data. Chapter 3 is focused on the calculation of features used in the questions. Chapters 4, 5, and 6 answer the three questions required in the report.

The code used to create the analysis is split into two places. In the `R/` folder of the attached zip file are the files used for downloading the data, cleaning it, and creating the fixed income features. The actual analysis and answering of the questions is done inside the `.Rmd` files found in the top level of the zip file.

This report was written with bookdown, a book authoring package for R.

Chapter 2

Data

2.1 Getting the data

The data is retrieved from the Federal Reserve website, under the discussion series: *The U.S. Treasury Yield Curve: 1961 to the Present*. The link for that site is here. The specific data set downloaded was the XLS file included on that site. `ratekit` provides the `download_rates_xls()` helper function for this.

The data was immediately opened in Excel, and resaved as an `xlsx` file. The format of the raw data is not a true `xls` file, rather, it is some flavor of `xml` file. This does not play nicely with R's packages for importing Excel data, so a resave was necessary and is done manually.

2.2 Cleaning

Data is brought in using the `readxl` package and the `ratekit` helper, `read_rates()`. This function reads the rectangle of rates data only, and sets any `-999.99` values to `NA`. These are often found through the dataset, and it is assumed that they represent missing values.

The column names in the data correspond to different types and lengths of rates used in the paper. The key for understanding the column names is below:

Table 2.1: Rates data: Column key

Series	Compounding Convention	Key
Zero-coupon yield	Continuously Compounded	SVENYXX
Par yield	Coupon-Equivalent	SVENPYXX
Instantaneous forward rate	Continuously Compounded	SVENFXX
One-year forward rate	Coupon-Equivalent	SVEN1FXX
Parameters	NA	BETA0 to TAU2

Most of these columns are not important for this analysis. Only the parameter columns and the date column are kept. To further examine the missing values, the `skimr` package was used, producing the following report.

Skim summary statistics
n obs: 14163
n variables: 7

Variable type: Date

variable	missing	complete	n	min	max	median	n_unique
date	0	14163	14163	1961-06-14	2018-03-29	1989-11-10	14163

Variable type: numeric

variable	missing	complete	n	mean	sd	p0	p25	p50	p75	p100
BETA0	0	14163	14163	5.88	4.63	0	3.03	5.01	7.92	25
BETA1	0	14163	14163	-0.82	5.05	-39.73	-3.07	-1.02	1.41	97.18
BETA2	0	14163	14163	-341.06	5684.56	-340683.77	-9.02	-0.99	1.93	94.87
BETA3	0	14163	14163	343.72	5684.31	-104.03	0	3.72	18.81	340681.5
TAU1	0	14163	14163	2.39	3.37	0.1	0.63	1.47	2.65	30
TAU2	4620	9543	14163	8.97	7.64	0.1	3.52	8.94	13.06	180.86

The `TAU2` column has a number of missing values. All of them occur before 1980, and were removed from the data set. After that removal, no missing values remain.

2.3 Monthly and Ascending

Monthly data is required for the report, but daily data is provided from the Fed. The data is converted to monthly using the `tibbletime` package. This leaves 459 rows of data for the project, spanning 1980-01-31 to 2018-03-29.

Chapter 3

Fixed Income Features Calculations

In this chapter, the construction of spot rates, zero coupon bond prices, excess returns, and yield factors are discussed. These features are used in hedging strategies and for general exploration of the data.

3.1 Spot Rates

Spot rates series can be constructed from the 6 parameters in the cleaned dataset. The following formula is used to construct the spot rates. It is integrated form of the Svensson extension of the Nelson and Siegal approach to calculating instantaneous forward rates. Svensson added a second hump term to the model that Nelson and Siegal created. Integrating the instantaneous forward rates gives us the spot rates.

$$
\begin{aligned}
y_t(n) = \beta_{0,t} & \\
& + \beta_{1,t} \frac{1 - \exp(-\frac{n}{\tau_{1,t}})}{\frac{n}{\tau_{1,t}}} \\
& + \beta_{2,t} [\frac{1 - \exp(-\frac{n}{\tau_{1,t}})}{\frac{n}{\tau_{1,t}}} - \exp(-\frac{n}{\tau_{1,t}})] \\
& + \beta_{3,t} [\frac{1 - \exp(-\frac{n}{\tau_{2,t}})}{\frac{n}{\tau_{2,t}}} - \exp(-\frac{n}{\tau_{2,t}})]
\end{aligned}
$$

To construct this, a programming concept known as a *function factory* was used. This is a specialized function that returns a function. This extends naturally to this use case because the outer function can accept the time series of the 6 parameters, and the inner function that is returned is parameterized by `n`, corresponding to the n-year spot rate at time t. This `spot_rate_factory()` function lives in `ratekit`.

Spot rates are examined in detail in Question 1, but the following graph provides a quick look at the major ones. As expected, longer maturity spot rates are consistently higher than short term spot rates. The 1980's were a time of incredibly high interest rates, and in recent years rates have been incredibly low.

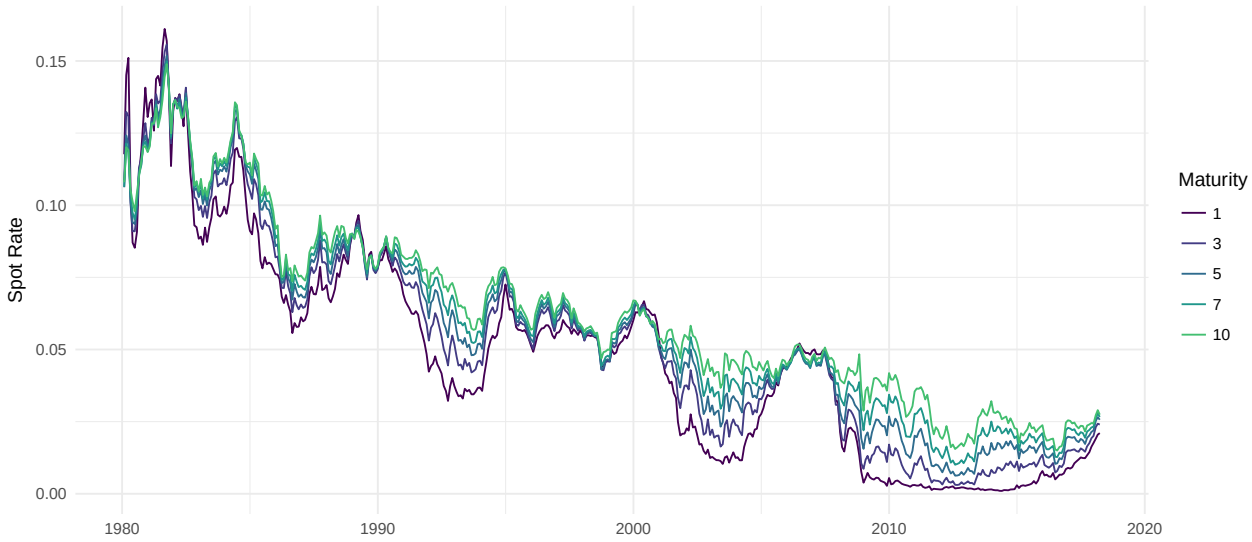

Figure 3.1: Spot rates at various maturities from 1980 onward.

3.2 Zero Coupon Bond Prices

N-year zero coupon bond prices can be calculated easily from their corresponding spot rates. The following formula is used to represent the relationship between the two.

$$P_t(n) = \exp(-y_t(n) \times n)$$

Similar to spot rates, a function factory named `zero_bond_price_factory()` was constructed that accepted the spot rate function, and returned a function that calculates a vector of bond prices parameterized by `n`.

Zero prices naturally exhibit an inverse relationship to spot rates. The impact of the high interest rates in the 1980's can be seen clearly here, with a large price spread between the short and long term maturity zeroes.

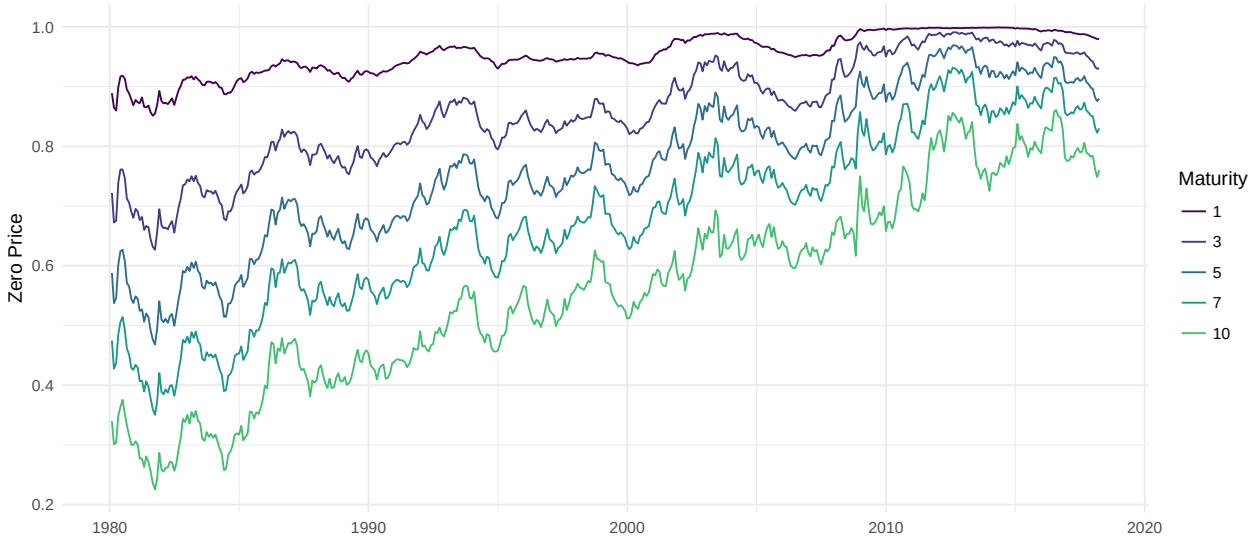

Figure 3.2: Zero prices at various maturities from 1980 onward.

3.3 One Month Returns

The time $t + \Delta$ return on an n-year bond is:

$$RET_{t+\Delta}(n) = \frac{P_{t+\Delta}(n - \Delta)}{P_t(n)} - 1$$

Using the zero bond prices from the previous section, it is straightforward to calculate returns. Care must be taken to align the price from next month's $n - \Delta$ maturity bond with today's n maturity bond, but otherwise the procedure is simple.

The distribution of returns is shown in Figure 3.3. As seen in both the figure and Table 3.1, higher maturity zeros have both larger average returns and more variance.

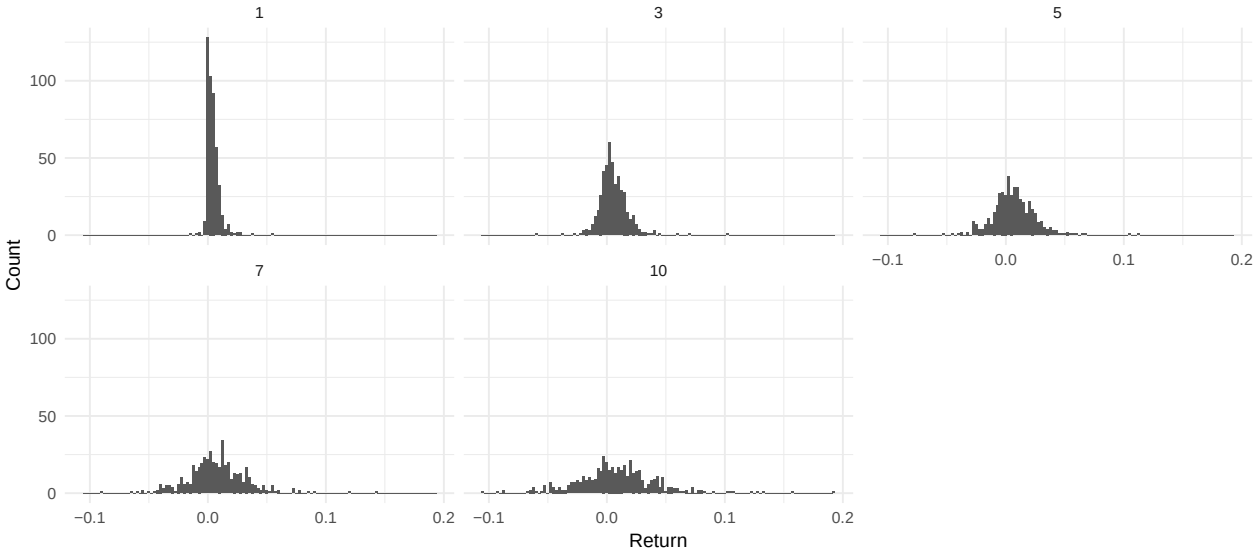

Figure 3.3: Return distributions for various maturities.

Table 3.1: Summary statistics of zero coupon bond returns since the 1980's.

Maturity	Average Return	Standard Deviation
1	0.0044510	0.0056046
3	0.0055643	0.0125839
5	0.0064662	0.0188464
7	0.0072532	0.0250410
10	0.0083056	0.0343082

3.4 Excess Returns

Excess returns are calculated over the 1 month treasury, specifically:

$$ER_{t+\Delta}(n) = RET_{t+\Delta}(n) - RET_{t+\Delta}(\Delta)$$

with $\Delta = 1/12$. Excess returns are only calculated for $n = 1, 3, 5, 7,$ and 10 year zeros.

The implementation of this is easy since returns for all maturities have already been calculated. The only potentially tricky thing to remember when calculating $RET_{t+\Delta}(\Delta)$ is that $P_{t+\Delta}(n-\Delta)=1$.

Excess returns show a similar distribution as returns, just shifted downwards by the 1 month treasury return.

3.5 Yield Curve Factors

Finally, the yield curve factors: level, slope, and curvature are calculated as:

$$\begin{aligned}\text{Level} &= y_t(1/4)\\ \text{Slope} &= y_t(8) - y_t(1/4)\\ \text{Curvature} &= [y_t(8) - y_t(2)] - [y_t(2) - y_t(1/4)]\end{aligned}$$

As seen in Figure 3.4, the raw values of the yield curve factors do not offer much insight on their own, but they will be useful later in decomposing the spot rate and in multiplicative regression hedging.

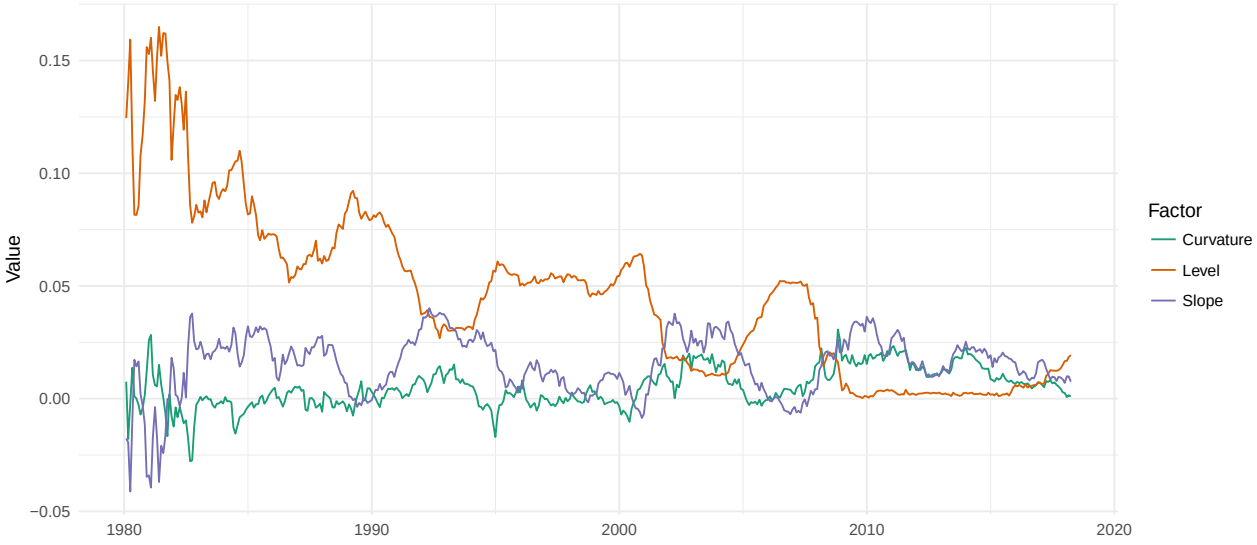

Figure 3.4: Yield Curve Factors over time

Chapter 4

Question 1

Question:

What are the time-series properties of the spot rates $y_t(1)$, $y_t(5)$, and $y_t(10)$? Report their summary statistics, including mean, standard deviation, skewness, kurtosis, and the first four autocorrelation coefficients, and the correlation matrix of the spot rates. Comment on your results. Also plot them and comment on the time series patterns.

In this question, data for the $y_t(1)$, $y_t(5)$, and $y_t(10)$ spot rates will be used. This has been calculated in Section 3.1, and the results from there are used here.

4.1 Summary Statistics

Reported in Table 4.1 are summary statistics on the three spot rate series.

Table 4.1: Summary statistics for 1, 5, and 10 year spot rates.

Maturity	Mean	Standard Deviation	Kurtosis	Skewness
1	0.0480538	0.0375029	2.913858	0.6505396
5	0.0566592	0.0345430	2.567191	0.5659543
10	0.0627654	0.0314939	2.586681	0.5793584

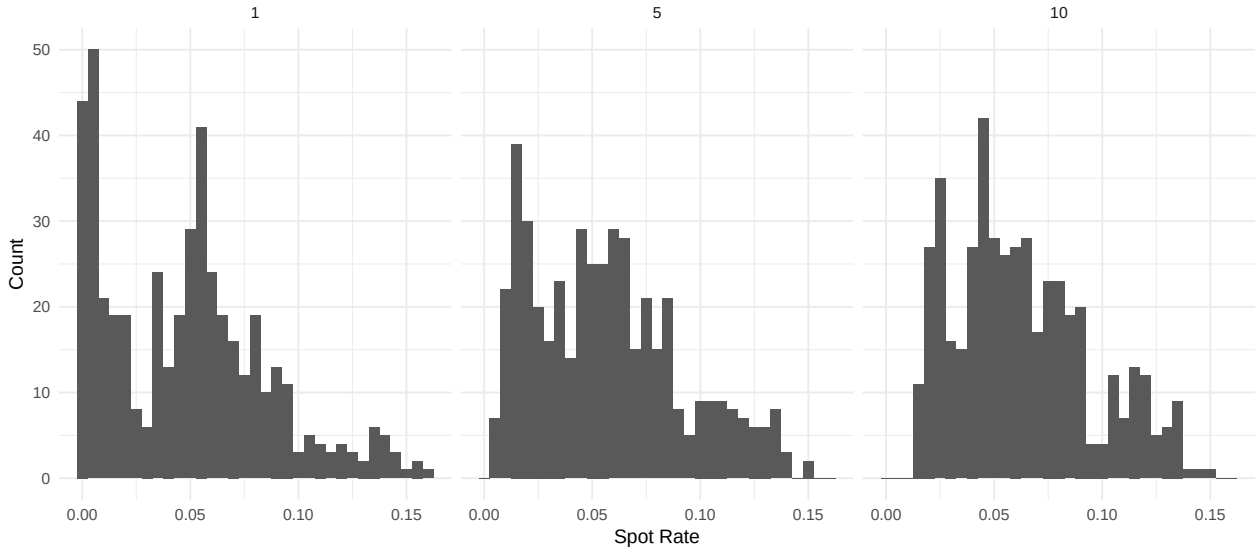

Figure 4.1: Spot rate distributions

Unsurprisingly, the average spot rate increases with the maturity, but interestingly, the shorter maturity spot rates have higher volatility. The kurtosis of all three are less than that of a normal distribution, but the 1 year maturity is very close. All three are right-skewed, with longer right tails, which makes sense considering extremely high interest rate periods do happen, but are rare. The histograms in Figure 4.1 are a nice visual confirmation of the results in the table. The skewness is clear, and the higher standard deviation for lower maturities might be attributed to the higher density at the extreme tails.

4.2 Autocorrelations

The first four autocorrelation correlation coefficients of the 3 series are reported in Table 4.2, along with a plot of the ACF for the series in Figure 4.2. Each of the series are highly autocorrelated.

Table 4.2: Autocorrelation coefficients for 1, 5, and 10 year spot rates.

Maturity	Lag 1	Lag 2	Lag 3	Lag 4
1	0.9878979	0.9697433	0.9527499	0.9411647
5	0.9911143	0.9791933	0.9686252	0.9599084
10	0.9906205	0.9793246	0.9691974	0.9606667

By looking at the entire ACF, we can see that the amount of autocorrelation increases in maturity. At farther out lags, 1 is less autocorrelated than 5 and 5 less than 10.

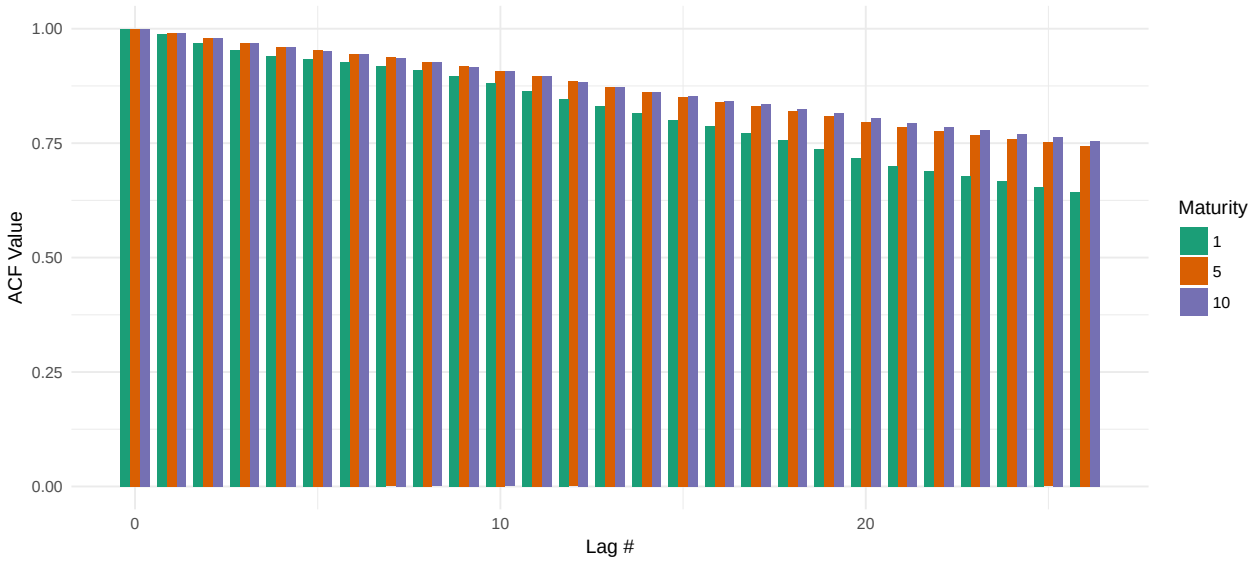

Figure 4.2: ACF for the 1, 5, and 10 year spot rates

4.3 Correlations

Moving on to correlations, it is clear that the the series are *highly* correlated. This should not be surprising whatsoever. Intuitively, the 1 year is more correlated with the 5 year than with the 10 year.

Table 4.3: Autocorrelation coefficients for 1, 5, and 10 year spot rates.

Maturity	1	5	10
1	1.0000000	0.9783620	0.9539364
5	0.9783620	1.0000000	0.9932312
10	0.9539364	0.9932312	1.0000000

4.4 Time Series Visualizations

A look at the time series of the three series confirms the highly autocorrelated and correlated nature of the three series. 1 year spot rates are almost always below the longer maturity rates, as one would expect. Since 2010, 1 year spot rates have been incredibly low, but have started to pick back up in the last few years.

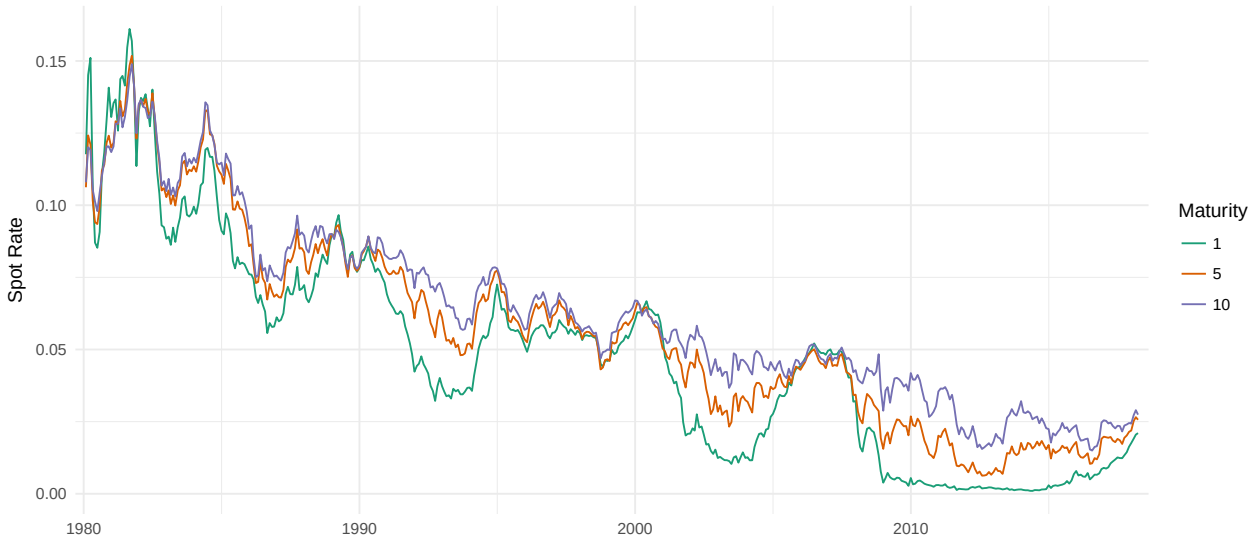

Figure 4.3: A look at the 1, 5, and 10 year spot rates over time

Chapter 5

Question 2

Question:

Can the three yield curve factors explain the time-series variation in spot rates? Regress $y_t(1)$ on a constant and X_t and comment on the regression statistics. Perform the same analysis for $y_t(5)$ and $y_t(10)$.

In this question, data for the $y_t(1)$, $y_t(5)$, and $y_t(10)$ spot rates will be used along with the yield curve factors, X_t. These have been calculated in Section 3.1 and Section 3.5, and the results from there are used here.

5.1 Regression

The following regression was run on every spot rate series:

$$y_t(n) = \alpha(n) + \beta_1(n)\text{Level}_t + \beta_2(n)\text{Slope}_t + \beta_3(n)\text{Curvature}_t + \epsilon_t(n)$$

Using the concept of multiple models from the book, `R 4 Data Science`, implementing these regressions in R is incredibly straightforward. The general concept involves two steps:

1) Split the data set of all 14 maturities into 14 groups, stored in a nested data frame.
2) For each group, run the linear model above and store the result.

The final result is a compact single data frame that contains the 14 resulting models along with the original data, indexed by the maturity.

```
## # A tibble: 14 x 3
##    maturity data               model
##       <dbl> <list>             <list>
##  1   0.0833 <tibble [459 x 5]> <S3: lm>
##  2   0.25   <tibble [459 x 5]> <S3: lm>
##  3   0.917  <tibble [459 x 5]> <S3: lm>
##  4   1      <tibble [459 x 5]> <S3: lm>
##  5   2      <tibble [459 x 5]> <S3: lm>
##  6   2.92   <tibble [459 x 5]> <S3: lm>
##  7   3      <tibble [459 x 5]> <S3: lm>
##  8   4.92   <tibble [459 x 5]> <S3: lm>
##  9   5      <tibble [459 x 5]> <S3: lm>
## 10   6.92   <tibble [459 x 5]> <S3: lm>
## 11   7      <tibble [459 x 5]> <S3: lm>
## 12   8      <tibble [459 x 5]> <S3: lm>
```

```
## 13   9.92   <tibble [459 x 5]> <S3: lm>
## 14  10      <tibble [459 x 5]> <S3: lm>
```

5.2 One Year Spot Rate

As seen in Table 5.1 and Table 5.2, all estimates for the 1 year spot rate model are highly significant, and the Adjusted R^2 is nearing 100%, suggesting that the model can explain essentially all of the variation in the spot rate.

Table 5.1: Regression results: One year spot

Term	Estimate	Standard Error	Statistic	P-Value
Intercept	0.001014021	0.0001280165	7.921022	1.8e-14
Level	0.989941953	0.0015226534	650.142657	0.0e+00
Slope	0.264774017	0.0034323179	77.141460	0.0e+00
Curvature	-0.428350107	0.0057618892	-74.341954	0.0e+00

Table 5.2: Regression R^2: One year spot

R Squared	R Squared Adj	Residual Std Error
0.9995224	0.9995193	0.0008223

A chart of the realized VS predicted time series for each model confirms how well the variation is explained. It is important to remember that this model is not predicting future rates, and is simply used to gather intuition about past rates. Nevertheless, it is interesting to see how well the yield curve factors explain the variation.

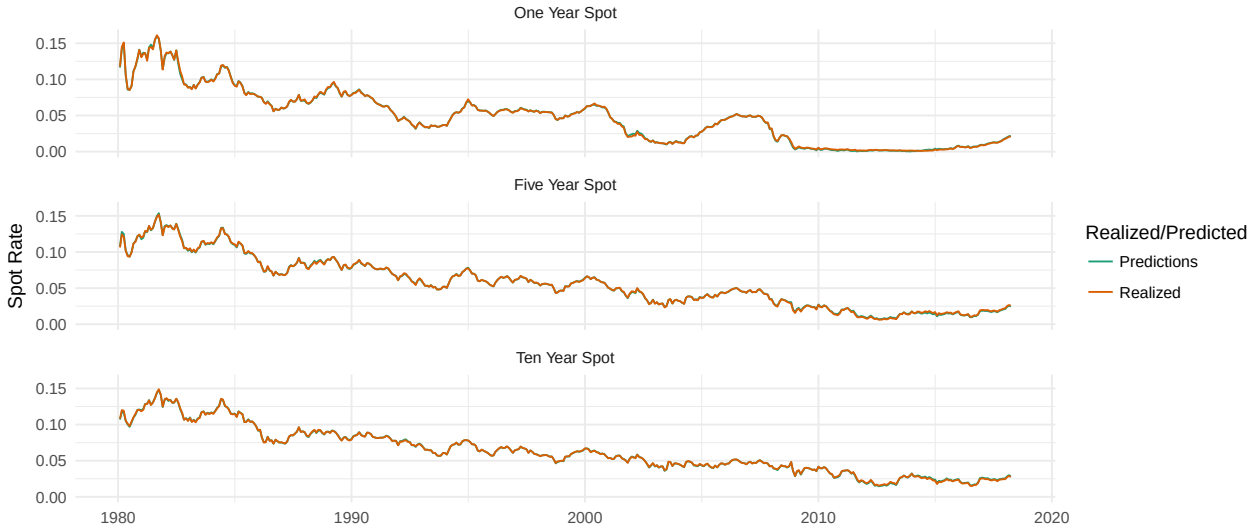

Figure 5.1: Regression in-sample realized VS predicted.

5.3 Five Year Spot Rate

The model for the 5 year rate is similar to the 1 year rate in terms of explanatory power.

Table 5.3: Regression results: Five year spot

Term	Estimate	Standard Error	Statistic	P-Value
Intercept	-0.001326488	0.0001178948	-11.25146	0
Level	1.010756975	0.0014022642	720.80351	0
Slope	0.862622600	0.0031609403	272.90063	0
Curvature	-0.237885069	0.0053063231	-44.83049	0

Table 5.4: Regression R^2: Five year spot

R Squared	R Squared Adj	Residual Std Error
0.9995226	0.9995194	0.0007572

5.4 Ten Year Spot Rate

And again, the 10 year model performs well too.

Table 5.5: Regression results: Ten year spot

Term	Estimate	Standard Error	Statistic	P-Value
Intercept	0.001285415	9.632731e-05	13.34424	0
Level	0.990119282	1.145736e-03	864.17722	0
Slope	1.041806029	2.582683e-03	403.38126	0
Curvature	0.101703126	4.335593e-03	23.45772	0

Table 5.6: Regression R^2: Ten year spot

R Squared	R Squared Adj	Residual Std Error
0.9996166	0.9996141	0.0006187

5.5 Decomposing the Spot Curve

Although not specifically asked for, it might be interesting to decompose and plot the spot curve at a few particular points in time. The procedure for this involved the following manipulations:

1) For month `m`, extract the spot rate at every available maturity for that month.
2) For month `m`, filter the yield curve factors down to that month, and multiply each maturity's regression coefficients by the corresponding yield curve factor. This gives the contribution of each factor for that month and each maturity.

3) Join the two data sets and chart them to view the decomposed spot rate for any month.

This procedure was streamlined into a single function, parameterized by the month to allow for easy plotting and comparison of multiple months. For example, the decomposed spot rate for January 2012 and January 1981 side by side is shown in Figure 5.2. January of 1981 was definitely an interesting time period! The spot rate is essentially inverted, with lower maturity bonds having higher spot rates than longer maturity bonds.

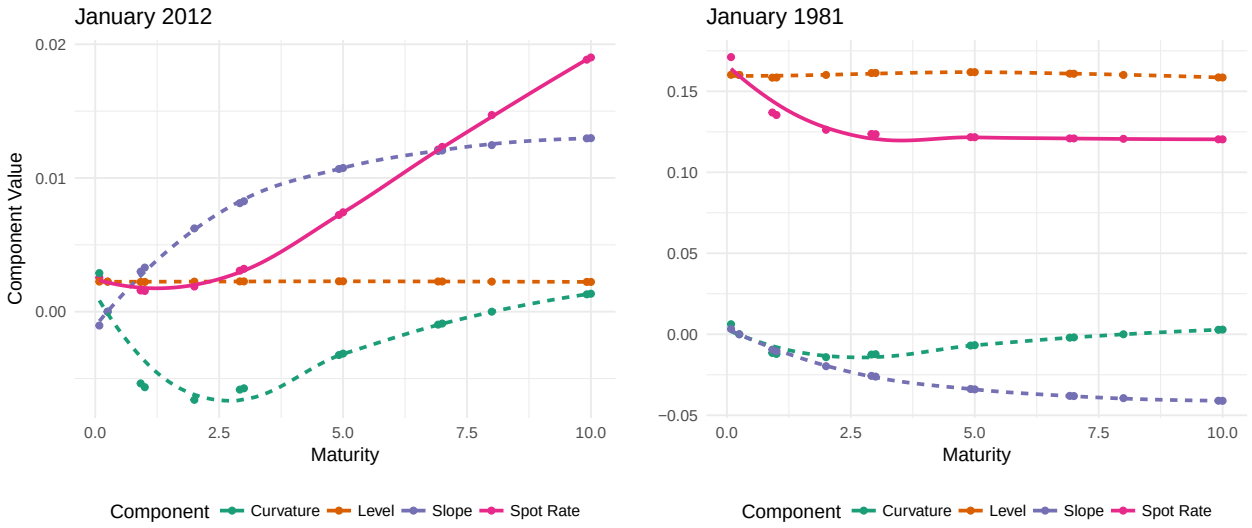

Figure 5.2: Decomposed Spot Rate for January of 2012 and 1981

5.6 Coefficient Stability

Another question worth asking is how stable the coefficients are throughout time. We can test this by running the same regression as before, but with a *rolling window*. This works by calculating the regression `spot_rate ~ level + slope + curvature` for the first 100 days, then shift forward 1 day and drop the last day, calculate the regression again, and repeat this for the length of the series. The `rsample` package provides a number of helpers for doing analysis exactly like this. The procedure for this is:

1) Split the data into 14 nested groups by maturity, as done in Section 5.1.
2) Further split each of the 14 groups into 359 rolling subsets using `rolling_origin()` from `rsample`.
3) For each maturity, and for each rolling split, run the regression from 5.1. This results in 5026 regressions, which on this computer can be run in parallel in ~8 seconds.

When the procedure has been run, a natural next step is to pick some of the 14 maturities and look at the stability of each coefficient over time. For example, the 1, 5, and 10 year coefficients over time are shown side by side in Figure 5.3. Most of the coefficients are fairly stable over time, with the exception of curvature. The curvature of the 1 year has begun to rise up from `-0.5` to around `-0.25` in recent years. This change over time is not refected in the `-0.428` curvature estimate we get from running the model over the full time period, and might offer other interesting insights. The 10 year curvature coefficient follows a similar pattern, but the 5 year curvature follows the opposite trend, decreasing in recent years.

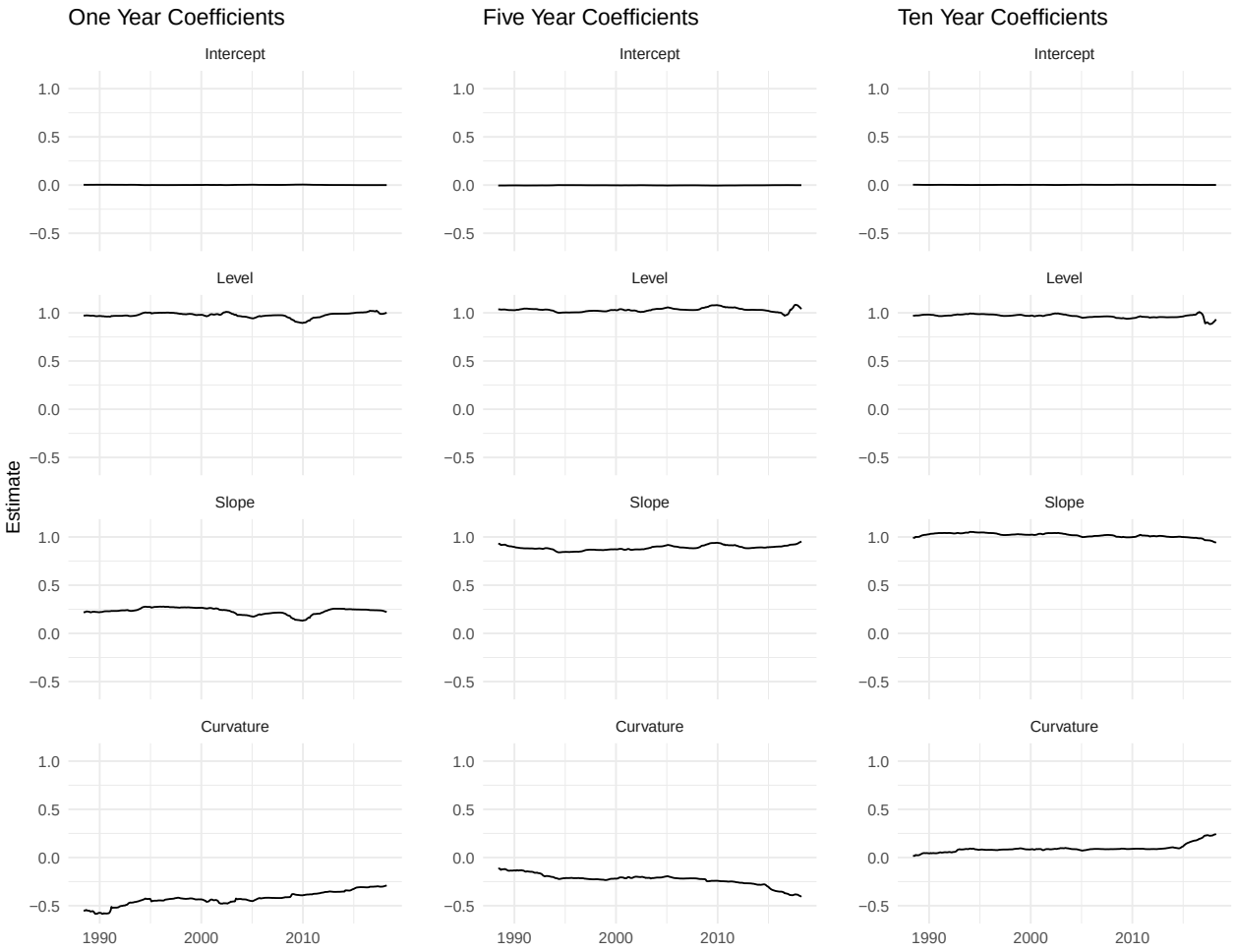

Figure 5.3: Coefficient stability for 1, 5, and 10 year spot rates. 100 day rolling window.

Chapter 6

Question 3

Question:

Perform the "out-of-sample" hedging exercises where $n_h = 3$. Split the sample into two halves, such that $T_0 = 2T$. Begin with the T-th month, calculate $w_T(1)$, $w_T(5)$, and $w_T(10)$ for each hedging strategy, and save $\epsilon_{T+\Delta}(n_h)$. Move forward and repeat the process each month, and calculate the root mean squared hedging error (RMSHE) for each strategy:

$$RMSHE = [\frac{1}{T}\sum \epsilon_{t+\Delta}(n_h)]^{0.5}$$

Report your results and evaluate the performance of the hedging strategies.

Implementation details

This chapter is broken into:

- 3 sections devoted to creating the modeling functions
- 1 section for creating a hedging error calculation function
- 1 section for creating an "interface" to the models and error calculations
- 1 section for running the models through the interface
- 1 section for reviewing model performance

The data for the $y_t(1)$, $y_t(3)$, $y_t(5)$, $y_t(7)$, and $y_t(10)$ spot rates is required to calculate the durations for the zero coupon bonds used in the hedging exercises. Also required are the yield curve factors and excess returns. These have all been calculated in 3, and the results from there are used.

The models developed below will be structured in a way so that they can all be fed with data from the same data set. To accomplish the rolling out-of-sample technique, the `rsample` package was used. Specifically, the `rolling_origin()` function was utilized which allows easy splitting of data into rolling sets of 230 months, which is the `T` value corresponding to half of the data.

6.1 Modified / Macaulay Duration

The first two strategies involve creating duration matched barbells using 1-year and 10-year zero coupon bonds to match the duration of a 3 year or 7 year bullet. Since the strategies are essentially the same, one modeling function is used for both, `model_duration()`, with an argument to select whether to use modified or macaulay duration. The duration is calculated using the `duration()` function that was created in the `ratekit` package and the barbell weights are calculated with the `barbell_weights()` function.

The function is created in such a way that it accepts a single split from the total data frame, and calculates the weights for just that one split. The weight for a 5 year zero is included to be consistent with the other models, but is set to `0`.

6.2 Simple regression based hedging

The regression based hedging method involves the regression:

$$ER_t(n_h) = w_t(1)ER_t(1) + w_t(5)ER_t(5) + w_t(10)ER_t(10) + u_t(n_h)$$

The R function `lm()` is used to run the regression, and the weights are extracted and returned in the same format as the duration models. This is all wrapped into a modeling function, `model_regression()`.

6.3 Multiplicative regression hedging

The multiplicative regression follows the model:

$$ER_t(n_h) = \theta_t(1; X_{t-\Delta})ER_t(1) + \theta_t(5; X_{t-\Delta})ER_t(5) + \theta_t(10; X_{t-\Delta})ER_t(10) + u_t(n_h)$$

Where $\theta_t(n; X_{t-\Delta})$ depends on the yield curve factors as:

$$\theta_t(n; X_{t-\Delta}) = a_t(n) + b_t(n)\text{Level}_{t-\Delta} + c_t(n)\text{Slope}_{t-\Delta} + d_t(n)\text{Curvature}_{t-\Delta}$$

Once the models are fit, the weights at time `t` can be calculated as θ_t values using the yield curve factors at `t`:

$$w_t(1) = \theta_t(1; X_t)w_t(5) = \theta_t(5; X_t)w_t(10) = \theta_t(10; X_t)$$

The entire procedure is wrapped into `model_multiplicative_regression()`, which, consistent with the other `model_*()` functions developed so far, accepts a single rolling split, along with the type of bullet (3 or 7 year), and returns the weights.

6.4 Error calculation

Hedging error for the weights set at time t are calculated at time $t + \Delta$ as:

$$\epsilon_{t+\Delta} = w_t(1)ER_{t+\Delta}(1) + w_t(5)ER_{t+\Delta}(5) + w_t(10)ER_{t+\Delta}(10)$$

These weights are then aggregated using RMSHE to determine overall model performance.

6.5 Model selection interface

A practical way to call all of the above models would be through an interface function that take as parameters: the data, the model to run, and the bullet to hedge against. Such a function was developed, and returns a data frame of the model type, the bullet used, the date of the error calculation, the weights, and the hedging errors.

6.6 Model application

Finally, a data frame is set up to easily iterate over all of the models, and the modeling function is run on each set of parameters. This data frame has a very compact form:

```
invokable
```

```
## # A tibble: 8 x 2
##   f           params
##   <chr>       <list>
## 1 apply_model <list [3]>
## 2 apply_model <list [3]>
## 3 apply_model <list [3]>
## 4 apply_model <list [3]>
## 5 apply_model <list [3]>
## 6 apply_model <list [3]>
## 7 apply_model <list [3]>
## 8 apply_model <list [3]>
```

Each element of the `params` column contains the data to be used in the model, the model type, and the bullet to hedge against.

```
invokable$params[[1]]
```

```
## $.data
## # Rolling origin forecast resampling
## # A tibble: 229 x 2
##   splits       id
##   <list>       <chr>
## 1 <S3: rsplit> Slice001
## 2 <S3: rsplit> Slice002
## 3 <S3: rsplit> Slice003
## 4 <S3: rsplit> Slice004
## 5 <S3: rsplit> Slice005
## # ... with 224 more rows
##
## $model
## [1] "modified_duration"
##
## $bullet
## [1] "3"
```

6.7 Model results

Table 6.1 displays the RMSHE results from each model. The regression methods significantly outperformed the duration based models, with much lower RMSHE. Among the duration models, Macaulay duration did marginally better, but the statistical significance is likely negligible. Among the regression models, the multiplicative regression did slightly better than the simple regression with the 3 year bullet and slightly worse with the 7 year bullet. The results are so close, however, that I am inclined to conclude that the simpler regression model is the best and most parsimonious model of the 4. There is little to indicate that longer maturity bullets are harder to hedge than shorter maturity bullets. In fact, with the regression models, the opposite seems to be true.

Table 6.1: Hedging Model Results

Model	Bullet	RMSHE
Macaulay	3	0.0038994
Macaulay	7	0.0039961
Modified	3	0.0039390
Modified	7	0.0040058
Multipl. Regression	3	0.0008060
Multipl. Regression	7	0.0007763
Regression	3	0.0008178
Regression	7	0.0007663

Because a rolling model was fit, a similar analysis can be performed as Section 5.6 where the stability of the coefficients over time was analyzed. In this case, the assignment of weights by the models over time can be analyzed rather than just the coefficients. Some interesting insights can be gathered Figure 6.1. Keeping in mind that duration models are *only* allowed to assign weight to a 1 year and 10 year zero, it is interesting to see how stagnant the weighting is between the 1 year and 10 year. The fact that the model essentially does not have the flexibility to vary the weights much over time likely contributes to its poor performance. The duration model also never shorts, which likely helps the regression models. The weightings in the duration models do make sense, with more than 50% being assigned to the 1 year bond for the 3 year bullet, and more than 50% being assigned to the 10 year for the 7 year bullet. In the regression models, there is an implicit risk free instrument that weight can be assigned to, so a 4th row in the figure is included to incorporate that. The simple regression model only shifts weights around in periods of high stress, 2001 and the tick around 2008 being two examples, but even so, the weightings are surprisingly stable over time. This is especially apparent when contrasted with the multiplicative regression. Weights are highly varied over time, but the performance gain for this is limited as seen in the RMSHE results.

Figure 6.1: Weight assignment over time

Another way to view the weights over time is by stacking them. This is done in Figure 6.2 and provides another unique view into the change in the total weight distribution over time. This view confirms the stagnant weights in the duration models, but also offers some other new insights. In the regression models, hedging the 3 year bullet requires shorting the 10 year and the risk free, while hedging the 7 year bullet only required hedging the 1 year bond. This view also further demonstrates the wild swings in the multiplicative regression, for questionable performance gains.

Figure 6.2: Stacked weight assignment over time

Chapter 7

Conclusion

In this project, spot rates from the Fed were analyzed along with their corresponding zero prices and returns over time. In addition, multiple hedging models were implemented and analyzed. The simple regression hedging model outperformed the two duration models, and was on par with the much more complicated multiplicative regression model.

I was particularly impressed with how well the yield curve factors explained the variation in the rates for other maturities. Apparently, using linear combinations of cetain maturity spot rates creates powerful explanatory variables for other maturities.

Further research could be done by including transactions costs in hedging performance calculations. This would likely penalize the multiplicative regression even further. Additionally, one could include thresholds that had to be hit before a certain recommended change in the weights was actually made. This could combat the transaction costs and would benefit the simple regression method along with the duration methods.