Forecasting CAD/USD Exchange Rates Under Pandemic Conditions

Junyoung Kang, Mei Han Lee, Nicole Yen Li Tham, Rajeev Parmasar

STAT 443: Forecasting (Winter 2022)

Team Contributions

Data

Sourcing the dataset was done by Nicole and Janice (Mei Han).

Regression Modelling

Regression modelling was done mainly by Nicole and Rajeev.

Smoothing Methods

Applying smoothing methods to the dataset was done primarily by Janice.

Box-Jenkins Modelling

Box-Jenkins Modelling was done primarily by Junyoung and Rajeev.

Conclusions

The conclusions were thoroughly discussed and done collectively by all group members.

Contents

List of Tables

List of Figures

1 Introduction

1.1 Motivation

Since the exchange rate is considered an indicator of the state of a country's economy, we wanted to study the impact of COVID-19 on the CAD/USD exchange rate.

By fitting a model to the data during this period, **we can attempt to predict how the currency and hence economy will be in the next week considering the effects of COVID-19.**

1.2 Data

We collected the daily exchange rate from the 1st of January 2020 until the 2nd of March 2022 from the **Bank of Canada's website** *(Sourced from: bankofcanada.ca/rates/exchange/daily-exchange-rates/).*

1.2.1 Time Series Plot

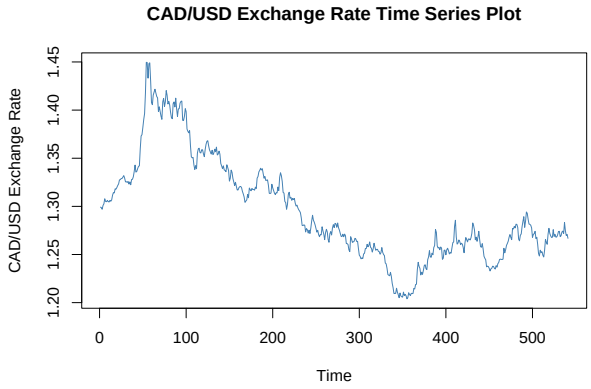

Figure 1: CAD/USD Exchange Rate Time Series Plot

1.2.2 Sample Auto-Correlation Function Plot

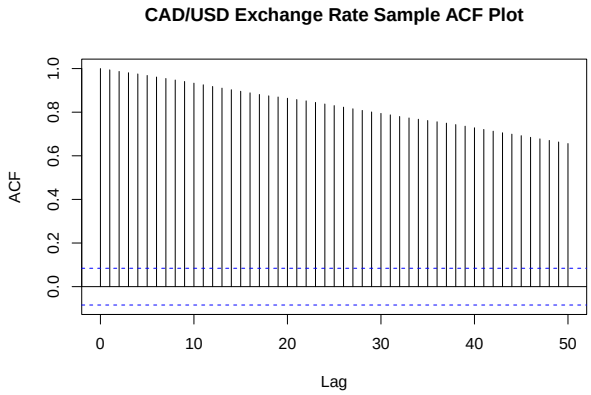

Figure 2: CAD/USD Exchange Rate Sample ACF Plot

1.2.3 Additive Classical Decomposition

We looked at our dataset using a weekly frequency and monthly frequency to confirm if we have any seasonality present.

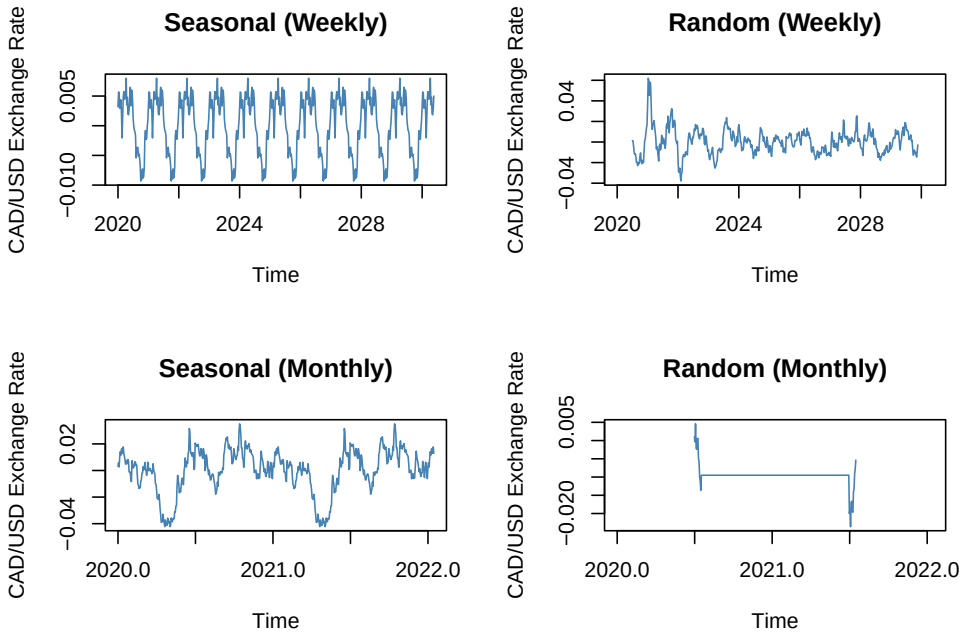

Figure 3: Additive Classical Decomposition Plots

Looking at **the weekly seasonal component**, we can see that **the periodic term is not significant** as its range is 30 times as small as that of the random component. Likewise, **the monthly seasonal component is also not significant** as its range is 10 times as small as that of the random component.

1.2.4 Key Observations

1.2.4.1 Trend There is a clear trend in the data (Figure 1.2) and the ACF plot (Figure 1.3) shows a slow linear decay. Hence the mean is not constant over time, **so the data is not stationary**. There is high frequency activity on top of the trend which seems to be noise. The variability seems to be non-constant over time as well.

1.2.4.2 Seasonality Based on the time series plot (Figure 1.2), **there is no seasonality apparent**. This is supported by the absence of a seasonal pattern in the ACF plot (Figure 1.3) as well as the vertical variation of the seasonal component in the decompose plot (Figure 1.4) being so low as to be insignificant.

1.2.4.3 Outliers There does not seem to have any obvious outliers present.

1.2.4.4 Missing Data The data only consists of business days since the exchange rates are not recorded on the weekends and holidays due to the close of the stock markets on these days.

1.2.4.5 Possible Solutions We find that since there is no seasonality, our dataset does not need any pre-processing before we can begin the model fitting process.

2 Modelling

We employ a variety of methods to select the optimum model which has the highest predictive power. This is equivalent the model with the **lowest** prediction mean squared error (MSE) defined as,

$$p\text{MSE} = \sum_{y \in \text{ test set}} \frac{(y_i - \hat{y}_i)^2}{\text{length(test set)}}$$

2.1 Training and Test Datasets

We will use the last month of our data as our test set (running from 1st Feb 2022 - 2nd Mar 2022) and train on the remainder of the data.

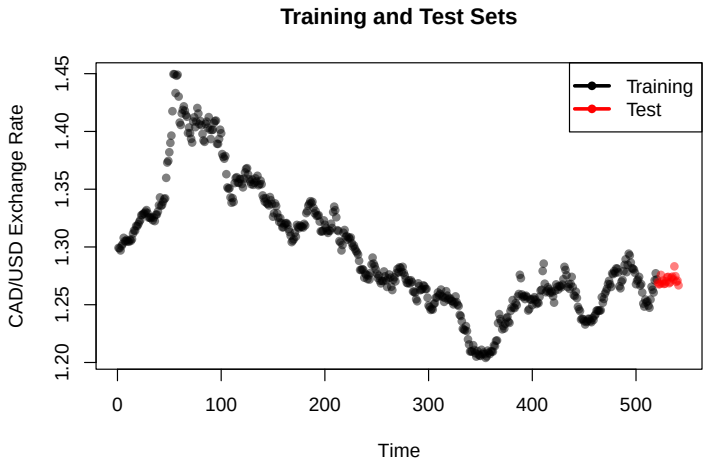

Figure 4: Dataset Train-Test Decomposition

2.2 Regression

We first propose a simpler unregularized polynomial regression method and then consider more complex shrinkage-based methods such as Ridge regression, LASSO regression and ElasticNet regression.

2.2.1 Unregularized Polynomial Regression

We propose the following model for our data.

$$y_i = \beta_0 + \beta_1 t_i + \beta_2 t_i^2 + ... + \beta_K t_i^K \quad , \; i = 0, 1, 2, ...540$$

where y_i represents the CAD/USD Exchange rate at time t_i.

We then fit polynomial regression models of degree $K = 1, 2, ..., 10$ and select the model with the lowest $pMSE$ as the optimal model.

Model Results Based on Figure 5, **our optimal unregularized model is a polynomial of degree 5** with a pMSE=$3.108e-05$.

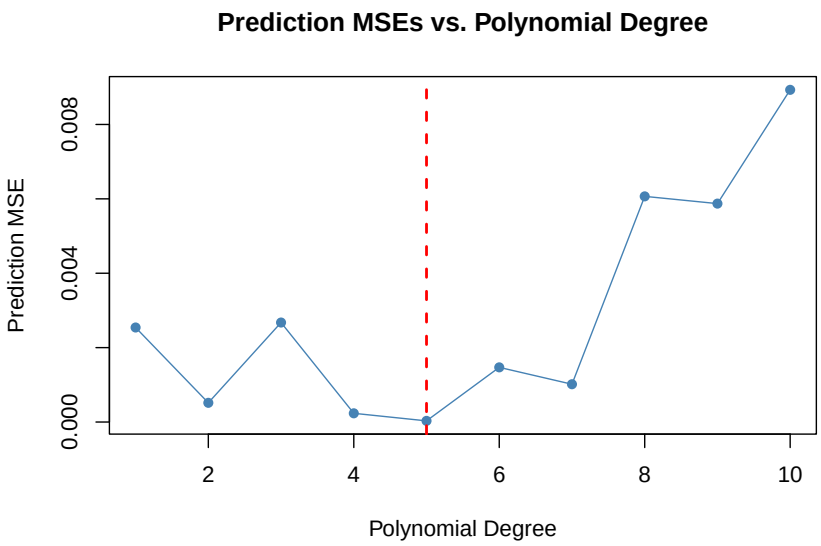

Figure 5: Prediction MSEs of Polynomial Regression Candidate Models

2.2.2 Elastic Net Regression

We propose the Elastic Net Regression method of regularization to model our data. The estimates of this linear model is given by,

$$\hat{\boldsymbol{\beta}} := \arg\min_{\beta} \left\{ \text{RSS } + \lambda \sum_{j=1}^{p} \left[(1-\alpha)\beta_j^2 + \alpha|\beta_j| \right] \right\}, \text{RSS } = \sum_{i=1}^{n} \left(y_i - \beta_0 - \sum_{j=1}^{p} x_{ij}\beta_j \right)^2$$

where RSS is the residual sum of squares from Ordinary Least Squares (OLS) regression.

Using the libraries `caret` and `glmnet`, we can find the optimal tuning hyperparameters ($\hat{\lambda}$,$\hat{\alpha}$). This means **our model selection will consider both Ridge regression** ($\alpha = 0$) **and also LASSO regression** ($\alpha = 1$) and we will also have variable selection occurring as we try to find the optimal model.

We find that **the optimal hyperparameters are $\lambda =$ 2.376e-04 and $\alpha =$ 0.1** , with a pMSE 8.648e-03.

2.2.3 Best Regression Model

We have summarized the pMSEs for both models below.

Table 1: Prediction MSEs for Regression Models

	Polynomial Regression	Elastic Net Regression
pMSE	3.108e-05	8.648e-03

Since **the polynomial regression model has a lower pMSE**, we select that model as our optimal regression model.

Here is the 5 Day forecast using our optimal regression model.

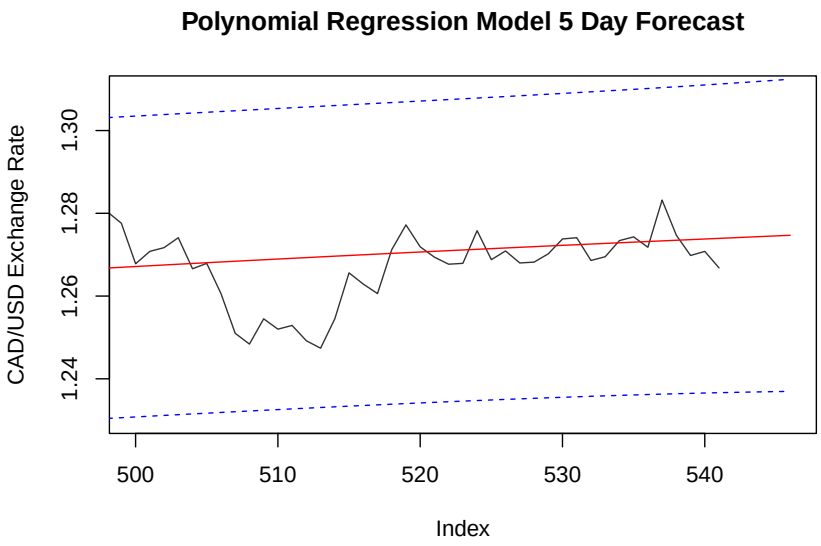

Figure 6: Polynomial Regression Model 5 Day Forecast

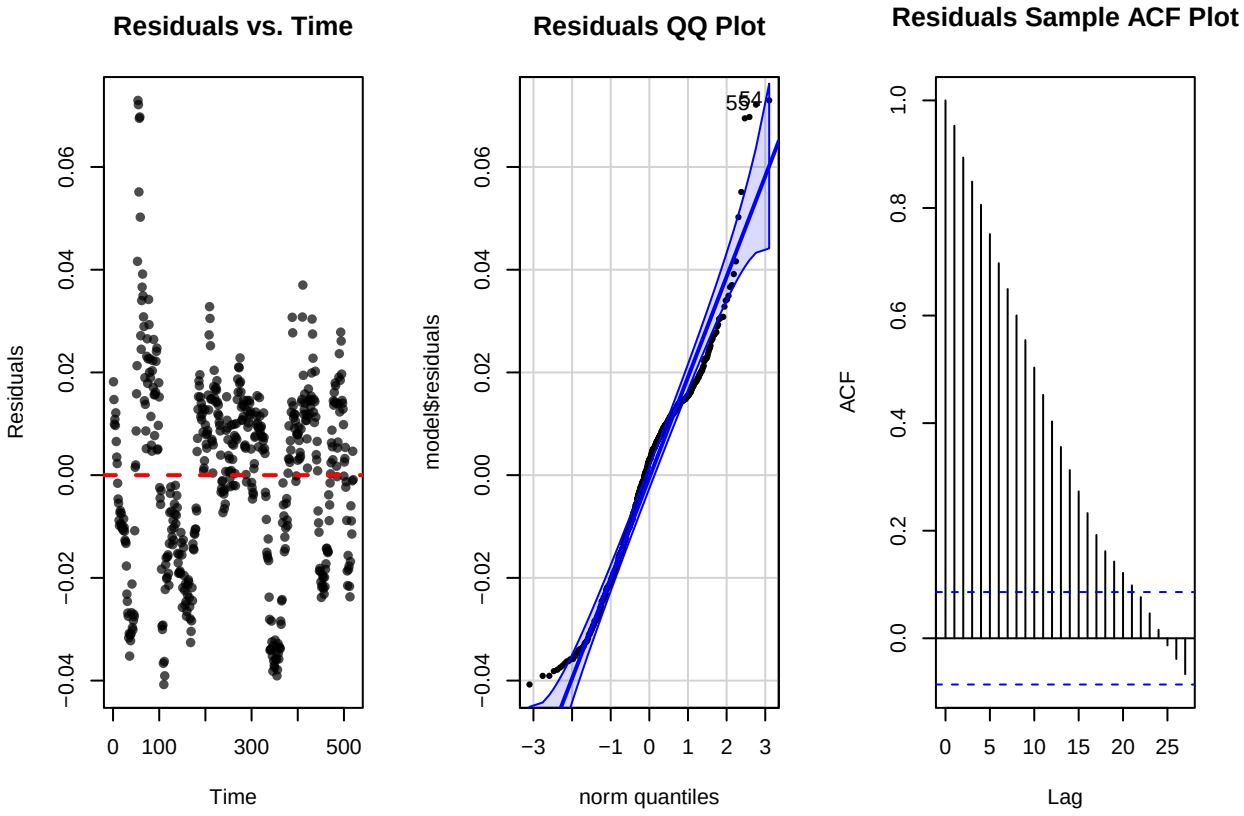

Figure 7: Polynomial Regression Residual Analysis

Model Diagnostics Here, our residuals have a mean of around 0 however, the variance is definitely not constant. Moreover, our residuals aren't quite normally distributed based on the QQ plot. The ACF also looks like there is a linear decay so our residuals may be correlated for different lags.

2.3 Smoothing

We use two types of smoothing to propose models for our dataset.

2.3.1 Simple Exponential Smoothing (SES)

From the Holt-Winters algorithm, we first try smoothing with $\beta = \gamma = 0$ so that our level L_t is given by

$$L_t = \alpha X_t + (1 - \alpha) L_{t-1}$$

Model Results We obtain the following results after fitting our SES model.

	Paramter Value
α	0.9999261

	Coefficient Value
a	1.2719

Table 2: SES Parameters and Coefficients

We have that pMSE = 1.418e-05.

2.3.2 Double Exponential Smoothing (DES)

Now, we have $\gamma = 0$ so that our equations become,

$$L_t = \alpha(X_t - I_{t-p}) + (1 - \alpha)(L_{t-1} + T_{t-1})$$
$$T_t = \beta(L_t - L_{t-1}) + (1 - \beta)T_{t-1}$$

	Parameter Value
α	1.0000000
β	0.0011174

	Coefficient Value
a	1.2719000
b	-0.0002496

Table 3: DES Parameters and Coefficients

Model Results We have that pMSE = 2.401e-05.

2.3.3 Best Smoothing Model

We have summarized the pMSEs of all the methods in the table below.

Table 4: Smoothing Models Prediction MSEs

	Simple Exponential Smooothing	Double Exponential Smoothing
pMSE	1.42e-05	2.4e-05

Comparing the pMSEs of the above methods, **the pMSE of the exponential smoothing is the smallest**. Therefore, it is chosen as the best model among the two models.

Using the exponential smoothing model to predict the coming five days data, **they will all be equal to CAD/USD 1.2668**.

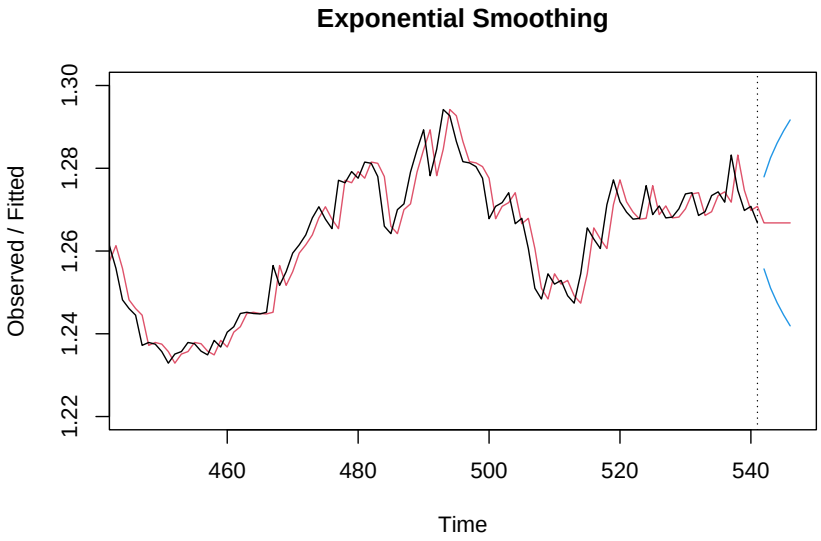

Figure 8: SES Model 5 Day Forecast

2.4 Box-Jenkins Modelling

Now, we will try to find a suitable Box-Jenkins model but first we have some questions to answer such as:

1. How can we deal with the trend in our data?
2. What models should we try?
3. How can we select the best model from our candidate models?

2.4.1 De-Trendizing Our Data

Recall from our original ACF plot (Figure 1.3), we noticed a linear decay in the lag spikes implying our data is non-stationary. As such, we need a method to make our data stationary and one such method is differencing. Rather than estimating our trend, we can remove it altogether.

We found that **first-order** differencing is sufficient for eliminating our trend and higher orders offer no significant benefit. Thus our process is given by

$$\nabla X_t = X_t - X_{t-1} \quad , \; t = 0, 1, ..., 540$$

First-Order Differenced Time Series Plot Here are plots comparing the effect of our differencing approach.

First-Order Differenced ACF and PACF Plots Here are the ACF and PACF plots for our differenced data.

Some key takeaways here,

1. **We no longer have any type of linear decay in our ACF plot** and most of the spikes are within our 95% confidence bands. As such, our differencing did a good job in making the data stationary.

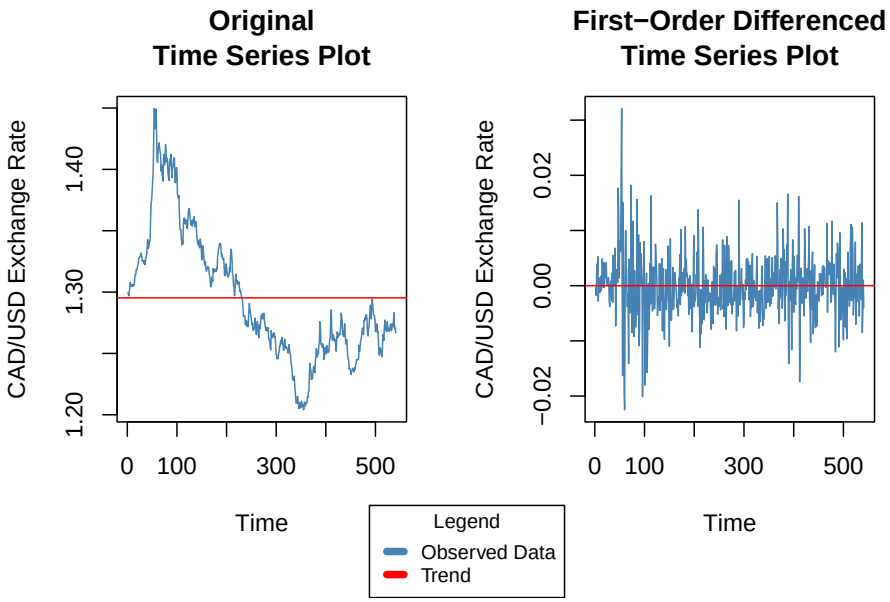

Figure 9: Time Series Plots Comparing Original Data and Differenced Data

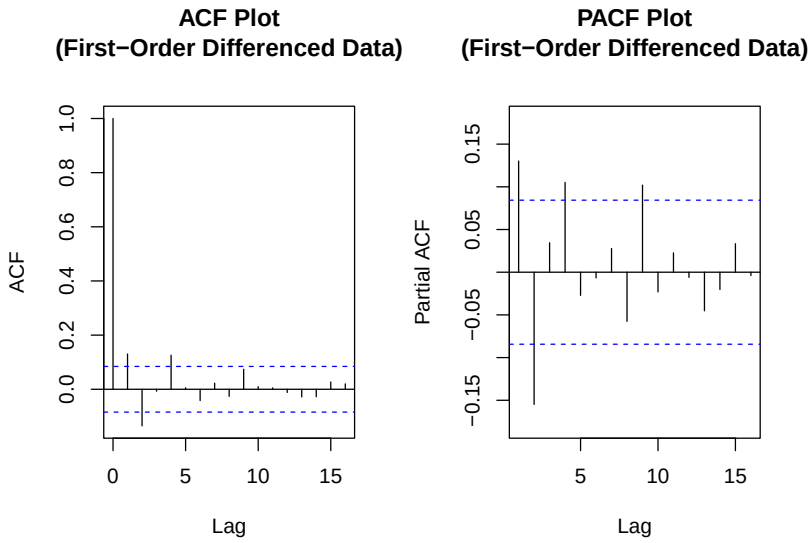

Figure 10: ACF and PACF Plots for Differenced Data

2. **Both our ACF and PACF plots seem to have a dampened sinusoidal pattern** which gives us some clues as to some candidate models.

2.4.2 Candidate Models

Recall our data is not seasonal, so interest rests on using the Box-Jenkins **ARMA(p,q) model** (or ARIMA with $d = 1$) defined as,

$$\Phi(B)X_t = \Theta(B)Z_t \quad , \; Z_t \sim \text{WN}(0, \sigma^2)$$

and the polynomials Φ and Θ defined as,

$$\Phi(z) = 1 - \phi_1 z - ... - \phi_p z^p$$

$$\Theta(z) = 1 + \theta_1 z + ... + \theta_q z^q$$

Now, our candidate **p** (**q**) parameters are selected as the spikes in the **PACF** (**ACF**) plot after which the **PACF** (**ACF**) becomes 0 for any lag greater than **p** (**q**). So we have,

$$p = \{0, 1, 2, 4, 9\}$$
$$q = \{0, 1, 2, 4\}$$

Now, we will fit models with all combinations of these parameters (20 models) and choose the model with the **lowest** prediction MSE.

Below is the candidates' model predictive performance on our test set.

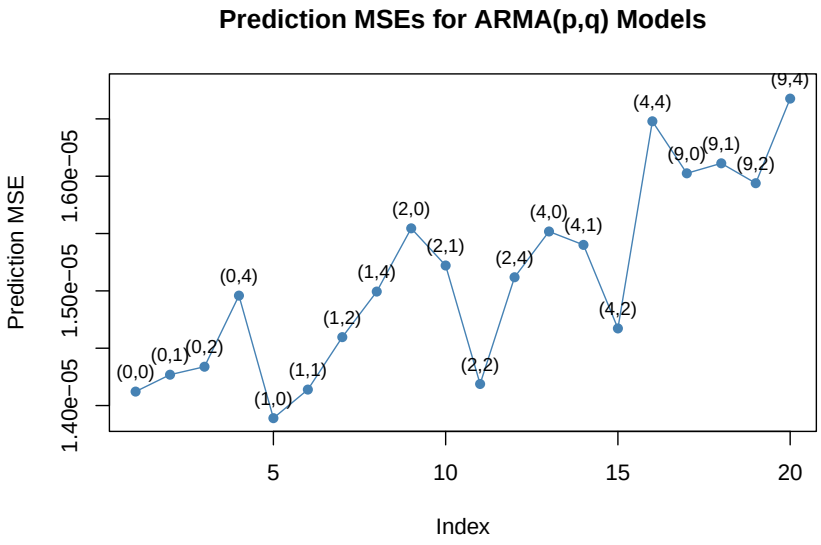

Figure 11: Candidate Models Prediction MSEs

2.4.3 Best Box-Jenkins Model

Based on the prediction MSEs (Figure 8), the model with the lowest pMSE is an **ARMA(1,0)** model with pMSE = 1.389e-05 however, we observe that **ARMA(2,2)** has a pMSE = 1.419e-05 which is not much higher than **ARMA(1,0)**.

As a result, after residual analysis on **ARMA(0,0)**, **ARMA(1,0)**, **ARMA(1,1)** and **ARMA(2,2)**, we opt for **ARMA(2,2)** as our best Box-Jenkins model since the difference in pMSEs are small but **this model is the only one with uncorrelated residuals**. Thus, our model is given by,

$$X_t - \phi_1 X_{t-1} - \phi_2 X_{t-2} = Z_t + \theta_1 Z_{t-1} + \theta_2 Z_{t-2} \quad , \; Z_t \sim \text{WN}(0, \sigma^2)$$

and after fitting this model on our entire dataset, our **parameter estimates** (with 95% Confidence Intervals) are given in the table below.

Table 5: Best Box Jenkins Model Coefficients

	Estimate	Lower Bound	Upper Bound
ϕ_1	-0.7192	-0.985956	-0.452444
ϕ_2	-0.6482	-0.879676	-0.416724
θ_1	0.8913	0.629052	1.153548
θ_2	0.6693	0.434884	0.903716

Model Diagnosis Now, we take a look at how well our model fits our assumptions i.e.

- Residuals have **constant zero mean** and **constant variance**. (1)
- Residuals are **Normally distributed**. (2)
- Residuals' **ACF is 0 for all lags**. (3)

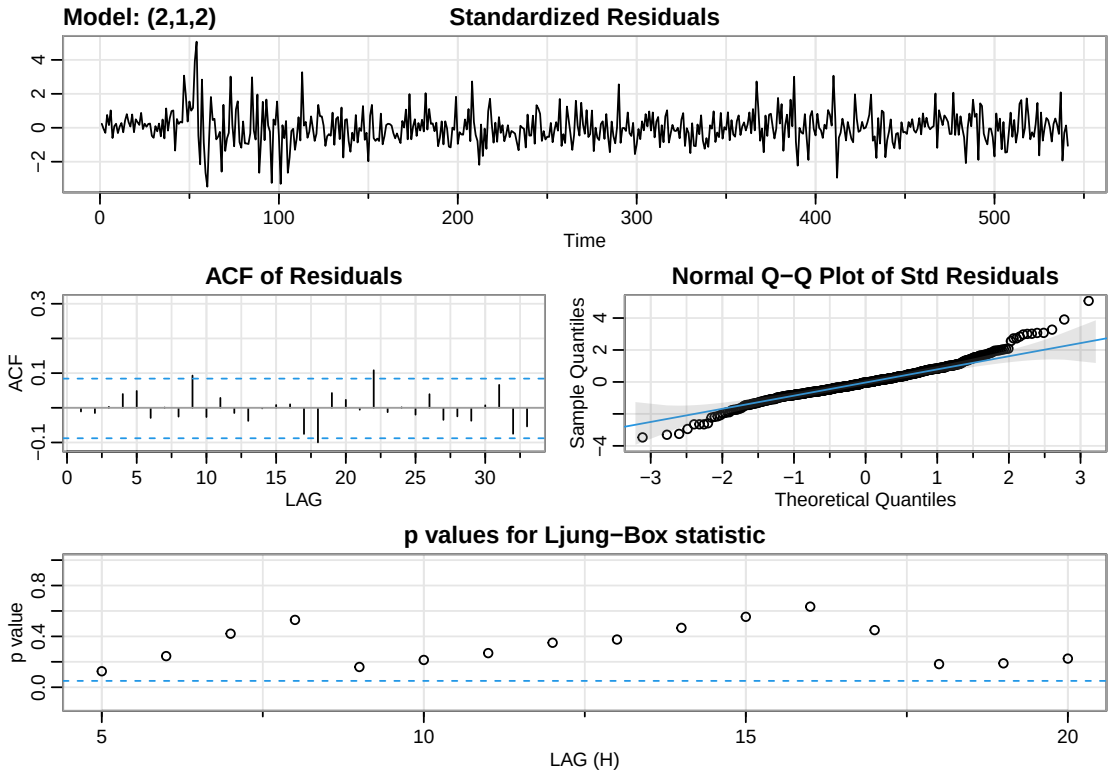

Figure 12: Optimal Model Residual Diagnostic Plots

Diagnostic Plots

Model Goodness of Fit From our diagnostic plots (Figure 10), in particular the standardized residuals plot, we see that **our residuals have a constant mean of around 0 however the variance seems to vary with time**. Moreover, from the QQ Plot, **the residuals seem to be a little more heavy tailed** than we would like but there does not seem to be any skewness.

However, the Residuals ACF plot looks good as **almost all of the spikes are within the 95% confidence bands**. Furthermore, the plot of the p-values shows that **the p values are mostly larger than 0.1 so** $H_0 : \rho_r(h) = 0\ ,\ h = 0, 1, 2, ...$ is most likely **not rejected**.

Overall, **we find that (3) holds well but (1) and (2) do not hold up so well**.

3 Best Overall Model

Here are the pMSEs for the best model from each of the previous strategy.

Table 6: Prediction MSEs for Best Candidate Models

	Polynomial Regression	Simple Exponential Smoothing	ARIMA(2,1,2)
pMSE	3.108e-05	1.418e-05	1.419e-05

Results from our analysis show that **the Box-Jenkins model ARIMA(2,1,2) is the best model to forecast our data**. Since we are interested in forecasting, we choose the ARIMA(2,1,2) model over the simple exponential smoothing model although it has a higher prediction MSE because **the simple exponential smoothing model does not take the trend of the data into account** and **the prediction MSEs are relatively close**.

3.1 5-Day Forecast

Here is a plot of our forecast for the next 5 days (3rd March 2022 to 9th March 2022 excluding weekdays) and the prediction intervals.

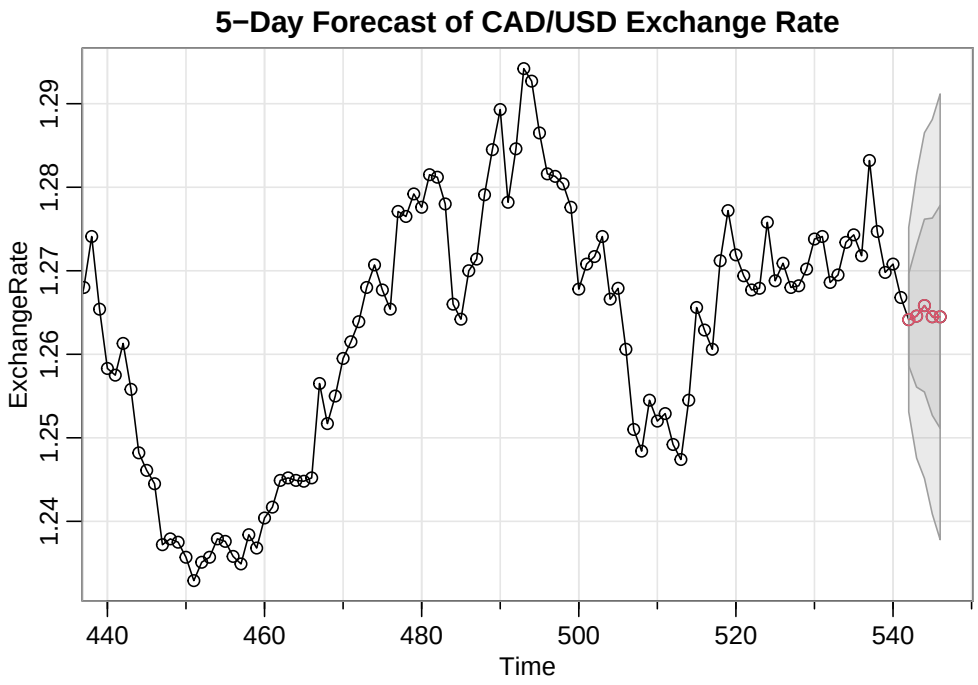

Figure 13: 5-Day Forecast of CAD/USD Exchange Rates

Table 7: 5-Day Forecast of CAD/USD Exchange Rate

	Forecast	Standard Error	95% Prediction Interval Lower Bound	95% Prediction Interval Upper Bound
542	1.264147	0.0055167	1.253335	1.274960
543	1.264573	0.0084998	1.247913	1.281232
544	1.265832	0.0103465	1.245553	1.286111
545	1.264496	0.0118087	1.241351	1.287641
546	1.264486	0.0133413	1.238337	1.290635

4 Conclusion

Based on our modelling results, **we expect that the exchange rate will be fluctuating between [1.238337, 1.2906349] for the next five days**.

In general, the market is expected to be bearish.

Appendix A

This appendix contains all code chunks used to generate this report from an `RMarkdown` file.

Helper Functions

```
# Fitting ARIMA models (Helper Fns) Calculates prediction
# MSE on test set
predictionMSE <- function(train_set, test_set, p, d, q) {
    forecast.obj <- astsa::sarima.for(train_set, n.ahead = length(test_set),
        p, d, q, plot = FALSE)
    MSE <- mean((test_set - forecast.obj$pred[1:length(forecast.obj$preds)])^2)

    return(MSE)
}

## Fits ARIMA model with (p,d,q) using data, and returns
## the model object
fitARIMA <- function(data, p, d, q) {
    mdl <- astsa::sarima(data, p, d, q)
    return(mdl)
}
```

Exploratory Data Analysis

```
# Reading in data
df <- read.csv("Data_Group28.csv")

# Convert the ExchangeRate into a time-series
ER.ts <- ts(df$ExchangeRate, start = c(2020, 1), frequency = 1)

# Time Series Plot
plot(ER.ts, ylab = "CAD/USD Exchange Rate", col = "steelblue",
    main = "CAD/USD Exchange Rate Time Series Plot")

# Sample Auto-Correlation Function Plot
acf(ER.ts, main = "CAD/USD Exchange Rate Sample ACF Plot", lag.max = 50)

# Trying different seasonalities
ER.ts.weekly <- ts(df$ExchangeRate, start = c(2020, 1), frequency = 52)
ER.ts.monthly <- ts(df$ExchangeRate, start = c(2020, 1), frequency = 265)
decomposed_ER.weekly <- decompose(ER.ts.weekly)
decomposed_ER.monthly <- decompose(ER.ts.monthly)
par(mfrow = c(2, 2))

plot(decomposed_ER.weekly$seasonal, main = "Seasonal (Weekly)",
    col = "steelblue", ylab = "CAD/USD Exchange Rate")
plot(decomposed_ER.weekly$random, main = "Random (Weekly)", col = "steelblue",
    ylab = "CAD/USD Exchange Rate")
plot(decomposed_ER.monthly$seasonal, main = "Seasonal (Monthly)",
    col = "steelblue", ylab = "CAD/USD Exchange Rate")
plot(decomposed_ER.monthly$random, main = "Random (Monthly)",
    col = "steelblue", ylab = "CAD/USD Exchange Rate")
```

Modelling

Training and Test Datasets

```
test.index <- 521:541
ER.Test <- data.frame(time = time(ER.ts)[test.index], ExchangeRate = ER.ts[test.index])
ER.Train <- data.frame(time = time(ER.ts)[-test.index], ExchangeRate = ER.ts[-test.index])
plot(ER.Train$time, ER.Train$ExchangeRate, pch = 16, col = adjustcolor("black",
    0.5), xlab = "Time", ylab = "CAD/USD Exchange Rate", main = "Training and Test Sets",
    xlim = c(ER.Train$time[1], ER.Train$time[1] + 541))
points(ER.Test$time, ER.Test$ExchangeRate, pch = 16, col = adjustcolor("red",
    0.5))
legend("topright", legend = c("Training", "Test"), lwd = 3, col = c("black",
    "red"), pch = 16)
```

Regression

```
#### Unregularized Polynomial Regression
rate <- ts(df$ExchangeRate)  # convert data to time series

# Split the data into test and train set
test.indx <- c(seq(521, 541))
day <- as.vector(time(rate))
rates <- as.vector(rate)
train <- rates[-test.indx]  # training set rates
day.train <- day[-test.indx]  # training set days
test <- rates[test.indx]
day.test <- day[test.indx]

K = 10

MSE <- c()
for (p in 1:K) {
    model = lm(train ~ poly(day.train, p))
    pred.rates = predict(model, new = data.frame(day.train = day[test.indx]),
        interval = "prediction")[, 1]

    MSE[p] = mean((rates[test.indx] - pred.rates)^2)
}
```

```
#### Unregularized Regression - Model Results plot of MSE
par(mfrow = c(1, 1))
plot(MSE, xlab = "Polynomial Degree", ylab = "Prediction MSE",
    pch = 16, main = "Prediction MSEs vs. Polynomial Degree",
    type = "o", col = "steelblue")
abline(v = which.min(MSE), col = "red", lwd = 2, lty = 2)
```

```
#### Elastic Net Regression

# build the model using the training set
set.seed(443)
train.set <- data.frame(rates = train, dates = day.train)
test.set <- data.frame(rates = test, dates = day.test)
elastic <- caret::train(rates ~ poly(dates, 10), data = train.set,
```

```
    method = "glmnet", trControl = caret::trainControl("cv",
        number = 10), tuneLength = 10)

# best tuning parameter
elastic$bestTune

# Coeffficient of the final model
coef(elastic$finalModel, elastic$bestTune$lambda)

# Make predictions on the test data
x.test <- model.matrix(rates ~ poly(dates, 10), data = test.set)[,
    -1]
elastic_pred <- predict(elastic, newdata = test.set)

# Prediction MSE
mean((test.set$rates - elastic_pred)^2)
```

```
#### Best Regression Model
mse.table <- t(data.frame(c(format(MSE[which.min(MSE)], scientific = TRUE,
    digits = 4), format(mean((test - elastic_pred)^2), scientific = TRUE,
    digits = 4))))
rownames(mse.table) <- c("pMSE")
colnames(mse.table) <- c("Polynomial Regression", "Elastic Net Regression")
knitr::kable(mse.table, caption = "Prediction MSEs for Regression Models")

### Generating Forecast Plot
p <- which.min(MSE)  # Should be 5
ER.df <- data.frame(rates = df$ExchangeRate, times = 1:541)
poly.mdl <- lm(rates ~ poly(times, degree = p), data = ER.df)
next5 <- data.frame(times = 542:546)

preds <- predict(poly.mdl, newdata = next5, interval = "prediction")
plot(ER.df$rates, xlim = c(500, 546), pch = 16, col = adjustcolor("black",
    0.8), ylim = c(1.23, 1.31), ylab = "CAD/USD Exchange Rate",
    main = "Polynomial Regression Model 5 Day Forecast", type = "l")

last100days <- 447:546
plot.preds <- predict(poly.mdl, newdata = data.frame(times = last100days),
    interval = "prediction")
points(last100days, plot.preds[, 1], col = adjustcolor("red",
    1), type = "l")
points(last100days, plot.preds[, 2], col = adjustcolor("blue",
    1), type = "l", lty = "dashed")
points(last100days, plot.preds[, 3], col = adjustcolor("blue",
    1), type = "l", lty = "dashed")
```

```
#### Model Diagnostics

# Graphical diagnostics and the residuals for the 6
# polynomial models.
p <- 5
model <- lm(train ~ poly(day.train, p))
par(mfrow = c(1, 3))  # Dividing the plotting page into 4 panels
plot(model$residuals, pch = 16, col = adjustcolor("black", 0.7),
    main = "Residuals vs. Time", ylab = "Residuals", xlab = "Time")  # # plotting the residuals vs time
abline(h = 0, lty = 2, lwd = 2, col = "red")  # plotting a horizontal line at 0
```

```r
car::qqPlot(model$residuals, pch = 16, main = "Residuals QQ Plot")

acf(model$residuals, main = "Residuals Sample ACF Plot")  #sample acf plot of residuals
```

Smoothing

```r
# Smoothing method
rate <- ts(df$ExchangeRate, frequency = 1)  #weekly cycle minusing the weedends
test.index <- 521:541
training.rate <- ts(df$ExchangeRate[-test.index], frequency = 1)

es <- HoltWinters(training.rate, gamma = FALSE, beta = FALSE)  # simple exponential smoothing
# es
pred = predict(es, n.ahead = 21)
APE.es = sum((pred - rate[test.index])^2)/length(test.index)

hw <- HoltWinters(training.rate, gamma = F)  #double exponential smoothing
# hw
HW.predict = predict(hw, n.ahead = 21)
APE.HW = sum((HW.predict - rate[test.index])^2)/length(test.index)
```

```r
#### SES Table
alpha.table <- c(`$\\alpha$` = es$alpha)

t1 <- knitr::kable(alpha.table, col.names = "Paramter Value",
    valign = "t", format = "latex", booktabs = TRUE, escape = FALSE)
t2 <- knitr::kable(es$coefficients, col.names = "Coefficient Value",
    valign = "t", format = "latex", booktabs = TRUE)

cat(c("\\begin{table}[h] \\centering ", t1, "\\hspace{1cm} \\centering ",
    t2, "\\caption{SES Parameters and Coefficients} \\end{table}"))
```

```r
#### DES Table
param.table <- c(`$\\alpha$` = hw$alpha[[1]], `$\\beta$` = hw$beta[[1]])

t1 <- knitr::kable(param.table, col.names = "Parameter Value",
    booktabs = TRUE, escape = FALSE, format = "latex", valign = "t")
t2 <- knitr::kable(hw$coefficients, col.names = "Coefficient Value",
    booktabs = TRUE, format = "latex", valign = "t")
cat(c("\\begin{table}[H] \\centering ", t1, "\\hspace{1cm} \\centering ",
    t2, "\\caption{DES Parameters and Coefficients} \\end{table}"))
```

```r
#### Best Smoothing Model
pmse.table <- data.frame(c(APE.es, APE.HW))
rownames(pmse.table) <- c("Simple Exponential Smooothing", "Double Exponential Smoothing")
colnames(pmse.table) <- "$p\\text{MSE}$"
knitr::kable(t(pmse.table), caption = "Smoothing Models Prediction MSEs",
    format = "latex", escape = FALSE, booktabs = TRUE) %>%
    kableExtra::kable_styling(full_width = F, latex_options = "HOLD_position")

#### Forecast
es.rate <- HoltWinters(rate, gamma = FALSE, beta = FALSE)
pred.es = predict(es.rate, prediction.interval = TRUE, n.ahead = 5)
plot(es.rate, pred.es, main = "Exponential Smoothing", xlim = c(446,
    546), ylim = c(1.22, 1.3))
```

Box-Jenkins Modelling

```
#### Time Series Plots Comparing Original Data and
#### Differenced Data

FO_diff.ts <- diff(ER.ts)
par(mfrow = c(1, 2), oma = c(2, 0, 0, 0))
plot(ER.ts, ylab = "CAD/USD Exchange Rate", col = "steelblue",
    main = "Original\nTime Series Plot")
abline(a = mean(ER.ts), b = 0, col = "red")
plot(FO_diff.ts, ylab = "CAD/USD Exchange Rate", col = "steelblue",
    main = "First-Order Differenced\nTime Series Plot")
abline(h = 0, col = "red")

par(fig = c(0, 1, 0, 1), oma = c(1, 0, 3, 0), mar = c(0, 0, 6,
    0), new = TRUE)
plot(0, 0, type = "l", bty = "n", xaxt = "n", yaxt = "n")
legend("bottom", legend = c("Observed Data", "Trend"), col = c("steelblue",
    "red"), lwd = 5, xpd = TRUE, cex = 0.8, seg.len = 1, title = "Legend")
# xpd = TRUE makes the legend plot to the figure
```

```
#### ACF and PACF Plots for Differenced Data

par(mfrow = c(1, 2))
acf(FO_diff.ts, main = "ACF Plot\n(First-Order Differenced Data)",
    lag.max = 16)  # Also looks good, maybe MA(1), MA(2), MA(4), MA(17), MA(18), MA(22)
pacf(FO_diff.ts, main = "PACF Plot\n(First-Order Differenced Data)",
    ylim = c(-0.18, 0.18), lag.max = 16)  # AR(1) or AR(2) or AR(4) or AR(9) or AR(22)
```

```
p_candidates <- c(0, 1, 2, 4, 9)
q_candidates <- c(0, 1, 2, 4)

p.table <- t(data.frame(p_candidates))
rownames(p.table) <- "p Candidates"
q.table <- t(data.frame(c("", q_candidates)))
rownames(q.table) <- "q Candidates"

# Finding best (p,q) combination in terms of pMSE
ARIMA_MSEs <- c()
for (p in p_candidates) {
    for (q in q_candidates) {
        current_mse <- predictionMSE(ER.Train$ExchangeRate, ER.Test$ExchangeRate,
            p, d = 1, q)
        current_model_name <- stringr::str_interp("(${p},${q})")
        names(current_mse) <- c(current_model_name)

        ARIMA_MSEs <- append(ARIMA_MSEs, current_mse)
    }
}
```

```
#### Candidate Models Prediction MSEs Plot
minMSE <- ARIMA_MSEs[which.min(ARIMA_MSEs)]
maxMSE <- ARIMA_MSEs[which.max(ARIMA_MSEs)]
plot(ARIMA_MSEs, type = "o", pch = 16, col = "steelblue", ylab = "Prediction MSE",
    main = "Prediction MSEs for ARMA(p,q) Models", ylim = c(minMSE,
        maxMSE + 1e-07))
```

```
text(ARIMA_MSEs, labels = names(ARIMA_MSEs), cex = 0.85, pos = c(rep(3,
    length(ARIMA_MSEs))), col = adjustcolor("black", 1.5))
```

```
#### Best Box-Jenkins Model

bestBoxJenkins.mdl <- fitARIMA(df$ExchangeRate, 2, 1, 2)
coef.table <- bestBoxJenkins.mdl$ttable[-5, ]
wn.sigma2 <- bestBoxJenkins.mdl$fit$sigma2

CI.table_lower <- coef.table[, 1] + (-1) * 1.96 * coef.table[,
    2]
CI.table_upper <- coef.table[, 1] + (1) * 1.96 * coef.table[,
    2]
CI.table <- cbind(coef.table[, 1], CI.table_lower, CI.table_upper)
rownames(CI.table) <- c("$\\phi_1$", "$\\phi_2$", "$\\theta_1$",
    "$\\theta_2$")
colnames(CI.table) <- c("Estimate", "Lower Bound", "Upper Bound")

knitr::kable(CI.table, booktabs = TRUE, caption = "Best Box Jenkins Model Coefficients",
    escape = FALSE)
```

```
#### Optimal Model Residual Diagnostic Plots
bestBoxJenkins.mdl <- fitARIMA(df$ExchangeRate, 2, 1, 2)
```

Best Overall Model

```
#### Best Overall Model Table
reg.mse <- format(MSE[which.min(MSE)], scientific = TRUE, digits = 4)
smooth.mse <- format(APE.es, scientific = TRUE, digits = 4)
bj.mse <- format(ARIMA_MSEs[11], scientific = TRUE, digits = 4)

final_mse.table <- t(data.frame(c(reg.mse, smooth.mse, bj.mse)))
colnames(final_mse.table) <- c("Polynomial Regression", "Simple Exponential Smoothing",
    "ARIMA(2,1,2)")
rownames(final_mse.table) <- c("pMSE")

knitr::kable(final_mse.table, booktabs = TRUE, caption = "Prediction MSEs for Best Candidate Models")
```

```
#### 5-Day Forecast of CAD/USD Exchange Rates using Best
#### Model
ExchangeRate <- df$ExchangeRate
preds <- astsa::sarima.for(ExchangeRate, n.ahead = 5, p = 2,
    d = 1, q = 2, main = "5-Day Forecast of CAD/USD Exchange Rate")
ses <- preds$se[1:length(preds$se)]
ests <- preds$pred[1:length(preds$pred)]
pis_lower <- ests - 1.96 * ses
pis_upper <- ests + 1.96 * ses
```

```
forecast.table <- data.frame(ests = ests, ses = ses, pi_lower = pis_lower,
    pi_upper = pis_upper)

rownames(forecast.table) <- c(542:546)
colnames(forecast.table) <- c("Forecast", "Standard Error", "95% Prediction Interval Lower Bound",
```

```
    "95% Prediction Interval Upper Bound")
knitr::kable(forecast.table, booktabs = TRUE, caption = "5-Day Forecast of CAD/USD Exchange Rate")
```

Appendix B

This appendix contains the code from the individual files of each team member.

Elastic Net.R

```
##-------------------------DATA-------------------------##

# Describe the data
data <- read.csv("~/Downloads/Data_Group28.csv")  # load the data in
rate <- ts(data$ExchangeRate)  # convert data to time series
ts.plot(rate, main = "CAD/USD Exchange Rate")  # time series plot
acf(rate)  # acf plot

# Split the data into test and train set
test.indx <- c(seq(521, 541))
day <- as.vector(time(rate))
rates <- as.vector(rate)
train <- rates[-test.indx]  # training set rates
day.train <- day[-test.indx]  # training set days
test <- rates[test.indx]
day.test <- day[test.indx]

##-------------------------------POLYNOMIAL UNREGULARIZED-------------------------------##


MSE = c()
# Change K from 1 to 10 to get polynomial fits up to degree
# K polynomial
par(mfrow = c(1, 1))
plot(day, rates, pch = 16, cex = 1.2, ylim = c(1.2, 1.5), type = "l",
    main = "World Pulp Prices and Shipment")
K = 10
for (p in 1:K) {
    model = lm(train ~ poly(day.train, p))
    pred.rates = predict(model, new = data.frame(day.train = day.test),
        interval = "prediction")[, 1]

    # Graphical diagnostics and the residuals for the 6
    # polynomial models.
    summary(model)  # gives the parameter estimates, R^2, p-values, etc.

    par(mfrow = c(3, 2))  # Dividing the plotting page into 6 panels
    plot(model$fitted, model$residuals, pch = 16, col = adjustcolor("black",
        0.7))  # plot of fitted values vs residuals
    title(main = paste0("Polynomial Model - p=", as.character(p)))
    abline(h = 0, lty = 2, lwd = 2, col = "red")  # plotting a horizontal line at 0
    car::qqPlot(model$residuals, pch = 16)
    title(main = paste0("Polynomial Model - p=", as.character(p)))
    plot(model$residuals, pch = 16, col = adjustcolor("black",
        0.7))  # # plotting the residuals vs time
    title(main = paste0("Polynomial Model - p=", as.character(p)))
    abline(h = 0, lty = 2, lwd = 2, col = "red")  # plotting a horizontal line at 0
    acf(model$residuals)  #sample acf plot of residuals
```

```
    # data with the fitted model
    plot(day, rates, pch = 16)
    title(main = paste0("Polynomial Model - p=", as.character(p)))
    lines(day, c(model$fitted.values, pred.rates), col = "blue",
        lwd = 2)

    # plot of y vs. y.hat
    plot(train ~ model$fitted.values, ylab = "Observed y (rate)",
        xlab = "Fitted value", xlim = c(1.2, 1.5), ylim = c(1.2,
            1.5), pch = 16, col = adjustcolor("black", 0.7))
    abline(a = 0, b = 1, lty = 2, lwd = 2, col = "red")
    title(main = paste0("Pulp Prices y vs. y.hat - p=", as.character(p)))

    MSE[p] = mean((test - pred.rates)^2)
    # plot of MSE
    par(mfrow = c(1, 1))
    plot(MSE, xlab = "Polynomial Degree", ylab = "Prediction MSE",
        pch = 16)
    abline(v = which.min(MSE), col = "red", lwd = 2, lty = 2)
}




##---------------------------------ELASTIC NET-------------------------------------##

library(tidyverse)
library(caret)
library(glmnet)


# build the model using the training set
elastic <- train(train ~ poly(day.train, 10), data = data.frame(day.train,
    train), method = "glmnet", trControl = trainControl("cv",
    number = 10), tuneLength = 10)

# best tuning parameter
elastic$bestTune

# Coeffficient of the final model
coef(elastic$finalModel, elastic$bestTune$lambda)

# Make predictions on the test data
x.test <- model.matrix(train ~ poly(day.train, 10), data = data.frame(day.test,
    test))[, -1]
elastic_pred <- elastic %>%
    predict(x.test) %>%
    as.vector()

# Prediction MSE
mean((test - elastic_pred)^2)
```

shrinkage-regression-script.R

```
########################### STAT 443 FINAL PROJECT -
########################### Shrinkage Methods

# loading in data
df <- read.csv("Data_Group28.csv")
df.ts <- ts(df)
df.times <- time(df.ts)[1:length(df$ExchangeRate)]

# RIDGE Search Grid
Log.Lambda.Seq <- c(c(-10, -5, -2, -1, -0.5), seq(0, 10, by = 0.1))
Lambda.Seq <- exp(Log.Lambda.Seq)

## Pre-allocated for performance
CV.ests <- rep(NA, length(1:19))
Lambda.ests <- rep(NA, length(1:19))
for (p in 2:20) {
    # We want to run 10-fold CV for each degree (p) and
    # keep track of the optimal lambda and CV estimate
    # Defaults to 10-fold and MSE as loss fn and intercept
    # as TRUE
    currentMdl <- glmnet::cv.glmnet(poly(df.times, degree = p),
        df$ExchangeRate, intercept = TRUE, alpha = 0, standardize = TRUE)
    Lambda.ests[p - 1] <- currentMdl$lambda.min
    CV.ests[p - 1] <- currentMdl$cvm[currentMdl$index[1]]
}

plot(2:20, Lambda.ests, ylab = expression(lambda), xlab = "Polynomial Degree",
    main = "Lambda vs. Polynomial Degree", pch = 16, col = "blue",
    type = "o")

plot(2:20, CV.ests, ylab = "sMSE", xlab = "Polynomial Degree",
    main = "sMSE vs. Polynomial Degree", pch = 16, col = "purple",
    type = "o")

# Given we want best predictive power, we'll chose the
# lowest sMSE and use the lowest lambda in the training We
# use the intercept here
chosen_degree <- which.min(CV.ests)
chosenMdl <- glmnet::glmnet(poly(df.times, degree = chosen_degree +
    1), df$ExchangeRate, lambda = Lambda.ests[which.min(Lambda.ests)],
    intercept = TRUE, family = "gaussian", standardize = TRUE,
    alpha = 0)

chosenMdl.fitted.vals <- predict(chosenMdl, newx = poly(df.times,
    degree = chosen_degree + 1), type = "response")

plot(df.times, df$ExchangeRate, col = adjustcolor("black", 0.5),
    xlab = "Time", ylab = "Daily Sales", main = "Daily Sales over Time",
    pch = 16)
points(df.times, chosenMdl.fitted.vals, col = adjustcolor("red",
    0.5), pch = 16)

# LASSO HELPER: Chooses index of best lambda based on
# number of params
select_index <- function(cv.object) {
```

```
    cv.non_zero_arr <- cv.object$nzero
    cv.index <- cv.object$index  # lambda.min is 1, lambda.1se is 2

    var_count_min <- cv.non_zero_arr[cv.index[1]]
    var_count_1se <- cv.non_zero_arr[cv.index[2]]

    return(ifelse(var_count_1se < var_count_min, cv.index[2],
        cv.index[1]))
}


## Pre-allocated for performance
CV.ests_LASSO <- rep(NA, length(1:19))
Lambda.ests_LASSO <- rep(NA, length(1:19))
for (p in 2:20) {
    # We want to run 10-fold CV for each degree (p) and
    # keep track of the optimal lambda and CV estimate
    # Defaults to 10-fold and MSE as loss fn and intercept
    # as TRUE
    currentMdl <- glmnet::cv.glmnet(poly(df.times, degree = p),
        df$ExchangeRate, intercept = TRUE, alpha = 1, standardize = TRUE)
    chosen_index <- select_index(currentMdl)
    Lambda.ests_LASSO[p - 1] <- currentMdl$lambda[chosen_index]
    CV.ests_LASSO[p - 1] <- currentMdl$cvm[chosen_index]
}

plot(2:20, Lambda.ests_LASSO, ylab = expression(lambda), xlab = "Polynomial Degree",
    main = "Lambda vs. Polynomial Degree", pch = 16, col = "blue",
    type = "o")

plot(2:20, CV.ests_LASSO, ylab = "sMSE", xlab = "Polynomial Degree",
    main = "sMSE vs. Polynomial Degree", pch = 16, col = "purple",
    type = "o")

chosen_degree_LASSO <- which.min(CV.ests_LASSO)
chosenMdl_LASSO <- glmnet::glmnet(poly(df.times, degree = chosen_degree_LASSO +
    1), df$ExchangeRate, lambda = Lambda.ests_LASSO[which.min(Lambda.ests_LASSO)],
    intercept = TRUE, family = "gaussian", standardize = TRUE,
    alpha = 1)

chosenMdl.fitted.vals_LASSO <- predict(chosenMdl_LASSO, newx = poly(df.times,
    degree = chosen_degree_LASSO + 1), type = "response")

plot(df.times, df$ExchangeRate, col = adjustcolor("black", 0.5),
    xlab = "Time", ylab = "Daily Sales", main = "Daily Sales over Time",
    pch = 16)
points(df.times, chosenMdl.fitted.vals_LASSO, col = adjustcolor("red",
    0.5), pch = 16)
points(df.times, chosenMdl.fitted.vals, col = adjustcolor("blue",
    0.5), pch = 16)
```

Smoothing.R

```
# Smoothing method
test.index = 521:541
```

```r
training.rate = ts(data$ExchangeRate[-test.index], frequency = 1)

ma <- function(data, q) {
    # data: dataset in time series format and q: the order
    # of the MA filter
    n <- length(data)
    out <- NULL
    out$ma <- rep(0, length = n - 2 * q)
    out$time <- time(data)[(q + 1):(n - q)]
    for (i in 1:length(out$ma)) {
        lis <- i + (-q:q)
        out$ma[i] <- sum(data[q + lis])/(2 * q + 1)
    }
    out$ma <- c(rep(NA, q), out$ma, rep(NA, q))
    out$time <- c(rep(NA, q), out$time, rep(NA, q))
    out
}

# Fitting and plotting moving average filter with
# q=1,2,...,6
for (q in 10:20) {
    plot(rate, main = "Exchange Rate", ylab = "Rate", xlab = "Week",
        pch = 16, col = adjustcolor("black", 0.7))
    temp.fit <- ma(rate, q)
    lines(temp.fit$time, temp.fit$ma, col = q)
    text(5, 650, paste0("q = ", q), col = q, cex = 1.5)
}

es <- HoltWinters(training.rate, gamma = FALSE, beta = FALSE)  # simple exponential smoothing
es
pred = predict(es, n.ahead = 21)
APE.es = sum((pred - rate[test.index])^2)/length(test.index)

hw <- HoltWinters(training.rate, gamma = F)  #double exponential smoothing
hw
HW.predict = predict(hw, n.ahead = 21)
APE.HW = sum((HW.predict - rate[test.index])^2)/length(test.index)


c(APE.es, APE.HW)

es.rate <- HoltWinters(rate, gamma = FALSE, beta = FALSE)
pred.es = predict(es.rate, prediction.interval = TRUE, n.ahead = 5)
pred.es
plot(es.rate, pred.es, main = "Exponential Smoothing")
```

ARMA model.R

```
# Read the data
data <- read.csv(file = "C:/Waterloo/STAT 443/CADUSDExchangeRate.csv")

# Turn the data into a time series.
data.ts <- ts(data$ExchangeRate)
plot(data.ts)
acf(data.ts)
# Steady, slow decrease in ACF, thus not stationary. Shows
# a trend, but no seasonality.

# Use First Order Differencing to detrendize the data
diff.data.ts <- diff(data.ts)
plot(diff.data.ts, main = "First Order Differencing", ylab = "Residual")
acf(diff.data.ts, main = "ACF of First Order Differencing")
# Now the detrendized data is stationary.

# AR and MA Possible Model Selection using ACF and PACF.
par(mfrow = c(1, 2))
acf(diff.data.ts, lag.max = 16, main = "ACF of First Order Differencing")
pacf(diff.data.ts, lag.max = 16, main = "PACF of First Order Differencing")

# 1. ACF seems to be cut off after lag 1 and PACF has a
# damped sign wave.  MA(1).

# 2. ACF seems to be cut off after lag 2 and PACF has a
# damped sign wave.  MA(2).

# 3. ACF seems to be cut off after lag 4 and PACF has a
# damped sign wave.  MA(4)

# 4. ACF has a damped sign wave and PACF seems to be cut
# off after lag 1.  AR(1).

# 5. ACF has a damped sign wave and PACF seems to be cut
# off after lag 2.  AR(2).

# 6. ACF has a damped sign wave and PACF seems to be cut
# off after lag 4.  AR(4).

# 7. ACF has a damped sign wave and PACF seems to be cut
# off after lag 9.  AR(9).

# Training set -> 1:511 (511 data) Test set -> 512:541 (30
# data)
par(mfrow = c(1, 1))
Test.indx = 512:541
Data.vec = as.vector(data.ts)
Test = Data.vec[Test.indx]
Training = Data.vec[-Test.indx]

library(forecast)
# Predict 30 data sets and calculate the MSE 1.
diff_MA1 <- Arima(Training, order = c(0, 1, 1)) %>%
    forecast(h = 30)
MSE_MA1 <- mean((Test - diff_MA1$mean)^2)
```

```r
# 2.
diff_MA2 <- Arima(Training, order = c(0, 1, 2)) %>%
    forecast(h = 30)
MSE_MA2 <- mean((Test - diff_MA2$mean)^2)

# 3.
diff_MA4 <- Arima(Training, order = c(0, 1, 4)) %>%
    forecast(h = 30)
MSE_MA4 <- mean((Test - diff_MA4$mean)^2)

# 4.
diff_AR1 <- Arima(Training, order = c(1, 1, 0)) %>%
    forecast(h = 30)
MSE_AR1 <- mean((Test - diff_AR1$mean)^2)

# 5.
diff_AR2 <- Arima(Training, order = c(2, 1, 0)) %>%
    forecast(h = 30)
MSE_AR2 <- mean((Test - diff_AR2$mean)^2)

# 6.
diff_AR4 <- Arima(Training, order = c(4, 1, 0)) %>%
    forecast(h = 30)
MSE_AR4 <- mean((Test - diff_AR4$mean)^2)

# 7.
diff_AR9 <- Arima(Training, order = c(9, 1, 0)) %>%
    forecast(h = 30)
MSE_AR9 <- mean((Test - diff_AR9$mean)^2)

# 8.
diff_ <- Arima(Training, order = c(2, 1, 2)) %>%
    forecast(h = 30)
MSE_ARMA22 <- mean((Test - diff_$mean)^2)
MSE_ARMA22

# Compare each model and find the lowest MSE
rbind(MSE_MA1, MSE_MA2, MSE_MA4, MSE_AR1, MSE_AR2, MSE_AR4, MSE_AR9,
    MSE_ARMA22)

# Best model is ARMA(2,2).

# Forecast future 5 days
model <- Arima(data.ts, c(2, 1, 2))
model.fc <- predict(model, n.ahead = 5)
model

# Plot the original data with the predicted data
ts.plot(data.ts, model.fc$pred)
Upper = model.fc$pred + model.fc$se
Lower = model.fc$pred - model.fc$se
x = c(time(Upper), rev(time(Upper)))
y = c(Lower, rev(Upper))
polygon(x, y, border = 8, col = gray(0.6, alpha = 0.2))
lines(model.fc$pred, col = "red", lwd = 0.01)
```

```r
# Plot only the predicted data
plot(model.fc$pred, type = "o", ylab = "Exchange Rate", main = "Predicted 5 Days of Exchange Rate",
    ylim = c(1.25, 1.28))
grid()
polygon(x, y, border = 8, col = gray(0.6, alpha = 0.2))
text(model.fc$pred ~ time(model.fc$pred), labels = round(pred,
    4), data = model.fc, cex = 0.4, font = 2, pos = 1)

plot(residuals(model), ylab = "Residuals", main = "Standardized Residuals")
acf(residuals(model), lag.max = 16, main = "ACF of Residuals")
qqnorm(residuals(model))
qqline(residuals(model))
library(FitAR)
LBQPlot(residuals(model), lag.max = 16)
```

box-jenkins-script.R

```r
## STAT 443 FINAL

# Reading in data
df <- read.csv("Data_Group28.csv")

# Preview Dataframe
head(df)

# Convert the ExchangeRate into a time-series
ER.ts <- ts(df$ExchangeRate)

# Diagnostic Plots
plot(ER.ts)
acf(ER.ts)  # Linear Decay; No Pattern so no seasonality

# First Order Differencing
FO_diff.ts <- diff(ER.ts)
plot(FO_diff.ts)  # Now it looks like constant mean
acf(FO_diff.ts)  # Also looks good, maybe MA(1), MA(2), MA(4), MA(17), MA(18), MA(22)
pacf(FO_diff.ts, ylim = c(-0.5, 0.5))  # AR(1) or AR(2) or AR(4) or AR(9) or AR(22)

# Splitting Dataset
Test.indx <- 521:541
Test.indx <- (length(ER.ts) - 30):length(ER.ts)
ER.Test <- data.frame(time = time(ER.ts[Test.indx]), ExchangeRate = ER.ts[Test.indx])

ER.Train <- data.frame(time = time(ER.ts[-Test.indx]), ExchangeRate = ER.ts[-Test.indx])

# Fitting ARIMA models (Helper Fns) Calculates prediction
# MSE on test set
predictionMSE <- function(train_set, test_set, p, d, q) {
    forecast.obj <- astsa::sarima.for(train_set, n.ahead = length(test_set),
        p, d, q)
    MSE <- mean((test_set - forecast.obj$pred[1:length(forecast.obj$preds)])^2)

    return(MSE)
}
```

```r
## Fits ARIMA model with (p,d,q) using data, and returns
## the model object
fitARIMA <- function(data, p, d, q) {
    mdl <- astsa::sarima(data, p, d, q)
    return(mdl)
}

# STRATEGY 1) Find MSEs of all trial models 2) Select the
# lowest 3 3) Model Analysis for these 3 4) Choose Best
# Model (maybe from AIC?)

# MODELS TO TRY ARIMA(1,1,1) ARIMA(1,1,2) ARIMA(1,1,4)
# ARIMA(2,1,1) ARIMA(2,1,2) ARIMA(2,1,4) ARIMA(4,1,1)
# ARIMA(4,1,2) ARIMA(4,1,4)

p_candidates <- c(0, 1, 2, 4, 9)
q_candidates <- c(0, 1, 2, 4)

ARIMA_MSEs <- c()
for (p in p_candidates) {
    for (q in q_candidates) {
        current_mse <- predictionMSE(ER.Train$ExchangeRate, ER.Test$ExchangeRate,
            p, d = 1, q)
        current_model_name <- stringr::str_interp("(${p},1,${q})")
        names(current_mse) <- c(current_model_name)

        ARIMA_MSEs <- append(ARIMA_MSEs, current_mse)
    }
}

plot(ARIMA_MSEs, type = "o", pch = 16, col = "steelblue", ylab = "Preidction MSE",
    main = "Prediction MSEs for ARIMA(p,1,q) Models")
text(ARIMA_MSEs, labels = names(ARIMA_MSEs), cex = 0.7, pos = c(rep(3,
    length(ARIMA_MSEs))))

top5_MSEs <- sort(ARIMA_MSEs)[1:5]  # In Order: (1,1,0), (0,1,0), (1,1,1), (2,1,2), (0,1,1)
Best_Mdls <- c(fitARIMA(df$ExchangeRate, 1, 1, 0)$AIC, fitARIMA(df$ExchangeRate,
    0, 1, 0)$AIC, fitARIMA(df$ExchangeRate, 1, 1, 1)$AIC, fitARIMA(df$ExchangeRate,
    2, 1, 2)$AIC, fitARIMA(df$ExchangeRate, 0, 1, 1)$AIC)
```