Description of the data

The data set we analyzed was one of the project data sets provided by Dr. Patrick found in the prescrip dataset in the TSA library. The data set contains data about the monthly U.S. average prescription cost in dollars from August 1986 to March 1992. By looking at the raw data in *Figure 1*, the original data does not have the full year of data for the years 1986 and 1992; therefore, we will only use the data from August 1986 to July 1991 in order to have a complete/full cycle of years. From *Figure 2*, we can see that the variance seems to increase over time, so we applied a transformation of the data with a natural log transformation to stabilize the variance. The resulting graph in *Figure 3* has a more stable variance.

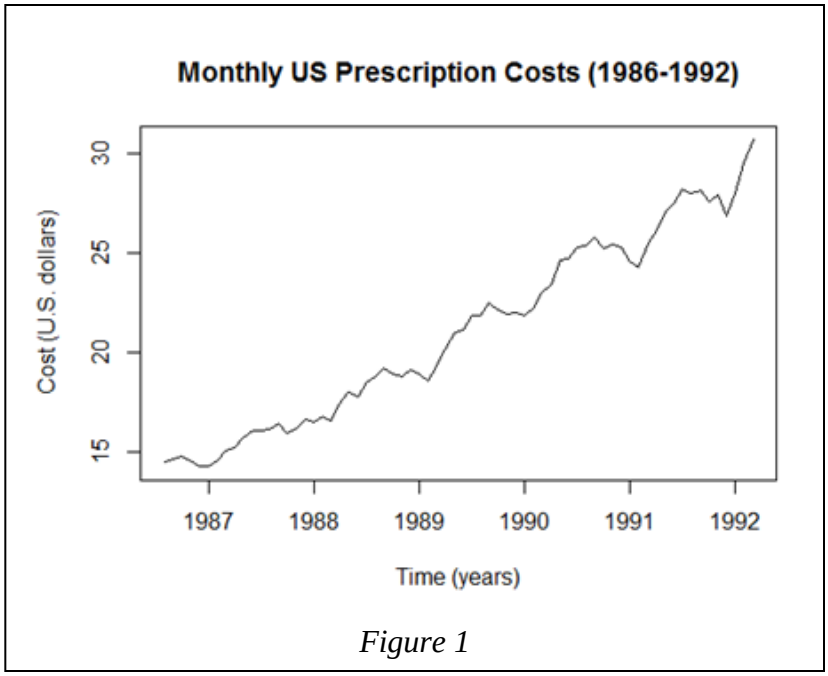

Figure 1

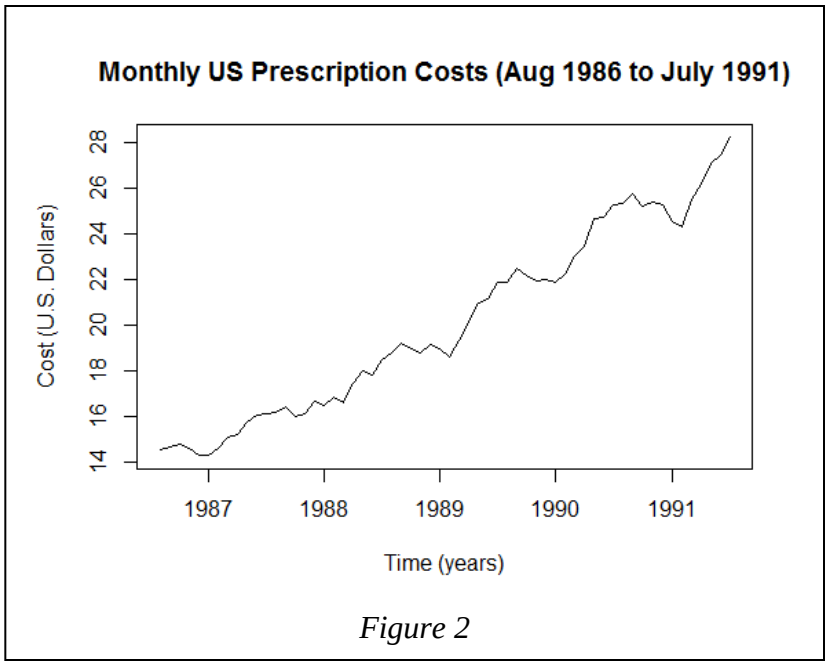

Figure 2

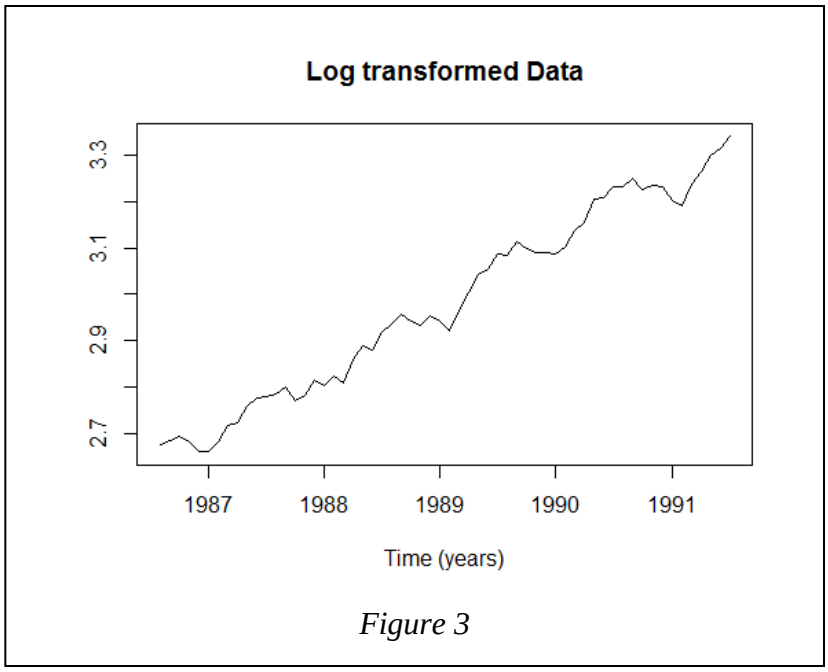

Figure 3

Deterministic components

From the plot of the log transformed data, we can see that there is trend from the continuous increase in the data. We can also see that there is a yearly seasonal component from the pattern in the data that occurs every year. To remove the trend in this data, we fit a second degree polynomial to the data. We chose a second degree polynomial and not a linear trend because the data points are not exactly linear; instead, the data points have a curvature to its shape. To remove the seasonality in this data, we used the sum of harmonics.

Figure 4 below is a plot of the log transformed data with the fitted trend. *Figure 5* is a plot of the residuals after the trend is removed. *Figure 6* is a plot of the residuals after the trend is removed with a line for the fitted seasonal component on top, and *Figure 7* is a plot of the residuals after the seasonal component has been removed.

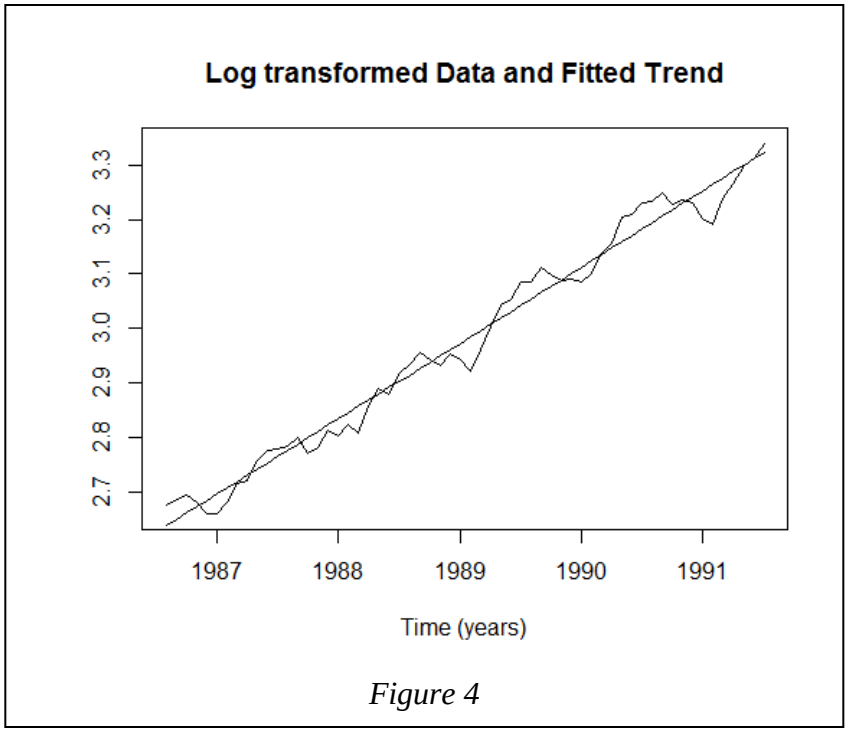

Figure 4

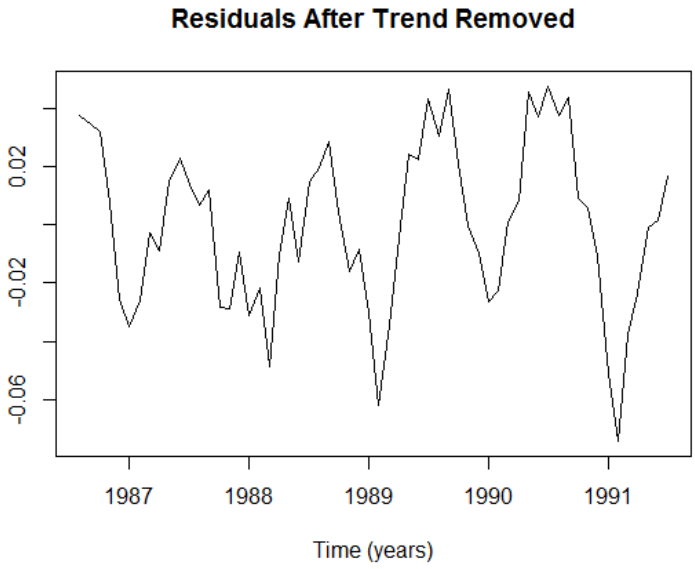

Figure 5

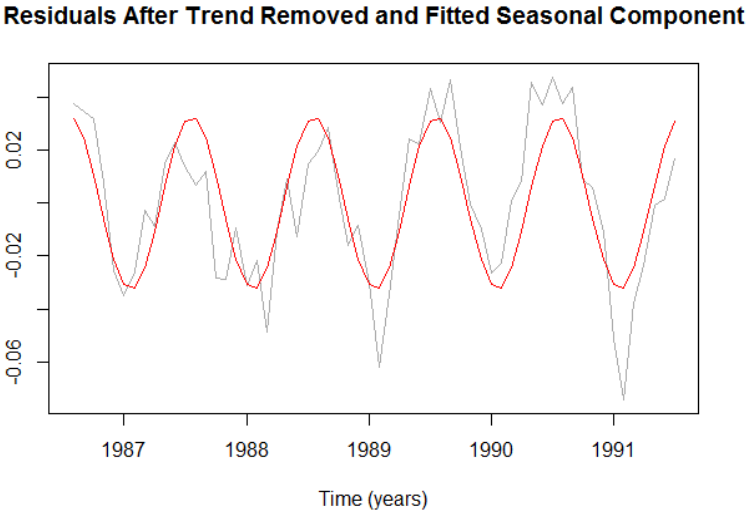

Figure 6

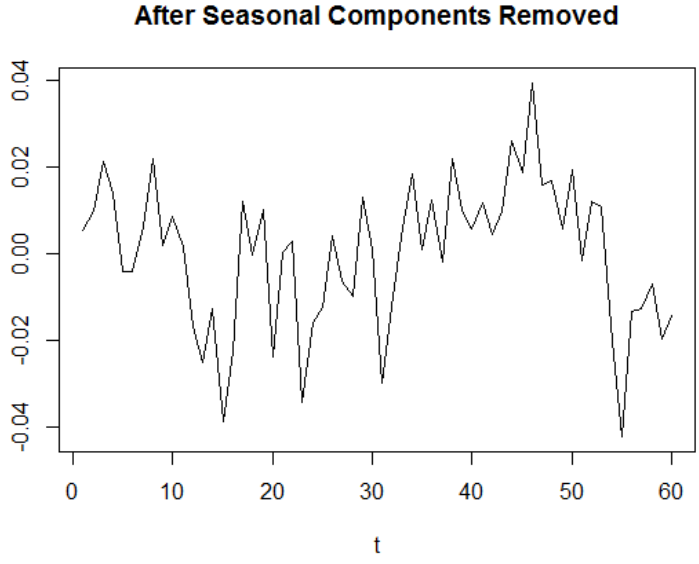

Figure 7

Time series model

After the deterministic components have been removed, the ACF and PACF plots were graphed. In *Figure 8* and *Figure 9*, since the ACF "trails-off" to 0, and PACF is 0 past lag 1, the model we considered was AR(1). Since looking at the ACF and PACF graphs are just an estimation of the model, we also applied the Hyndman-Khandakar algorithm to select the best model using the AIC criteria. The resulting model that the Hyndman-Khandakar algorithm selected was AR(1) as well (as shown in *Figure 10*).

AR(1) model:

$$X_t - 0.4840X_{t-1} = Z_t$$

$$Z_t \sim WN(0, 0.0002106)$$

After fitting an AR(1) model and checking the ACF and PACF plots in *Figure 11* and *Figure 12*, there doesn't seem to be any dependency structure remaining because there are no significant lags. To make sure that there is really no dependency structure left, we performed the Ljung-Box test as shown in *Figure 13*. Since the p-value is 0.8299 (not significant), then we cannot reject the hypothesis that the white noise (residuals) is independent.

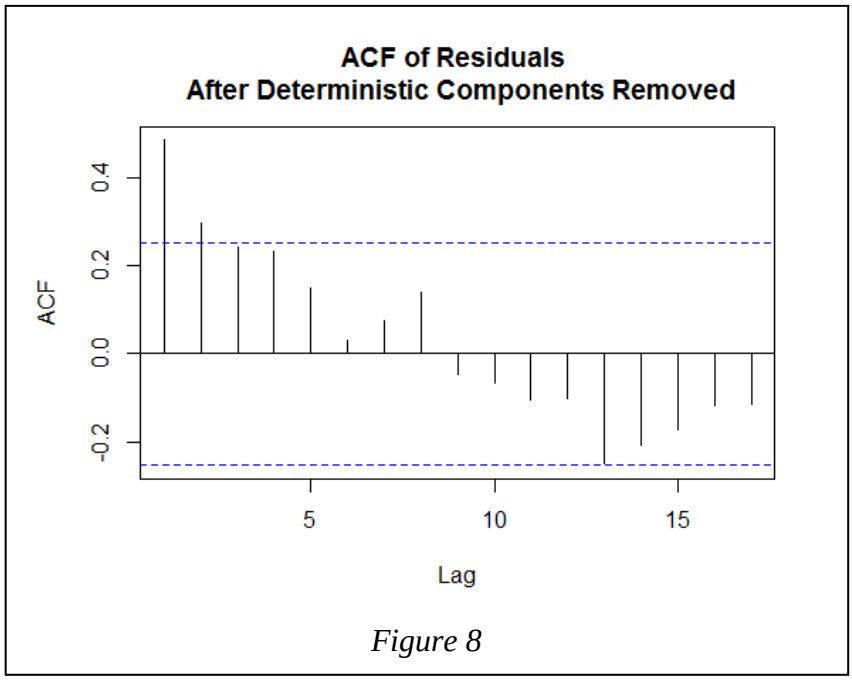

Figure 8

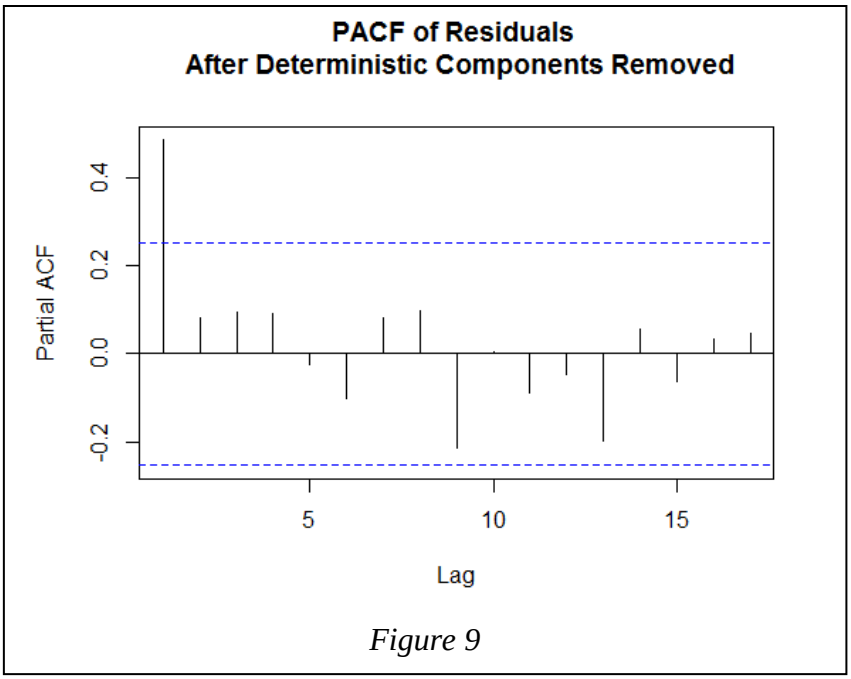

Figure 9

```
> arma.fit
Series: z
ARIMA(1,0,0) with zero mean

Coefficients:
         ar1
      0.4840
s.e.  0.1119

sigma^2 estimated as 0.0002106:  log likelihood=168.7
AIC=-333.4   AICc=-333.19   BIC=-329.21
```

Figure 10

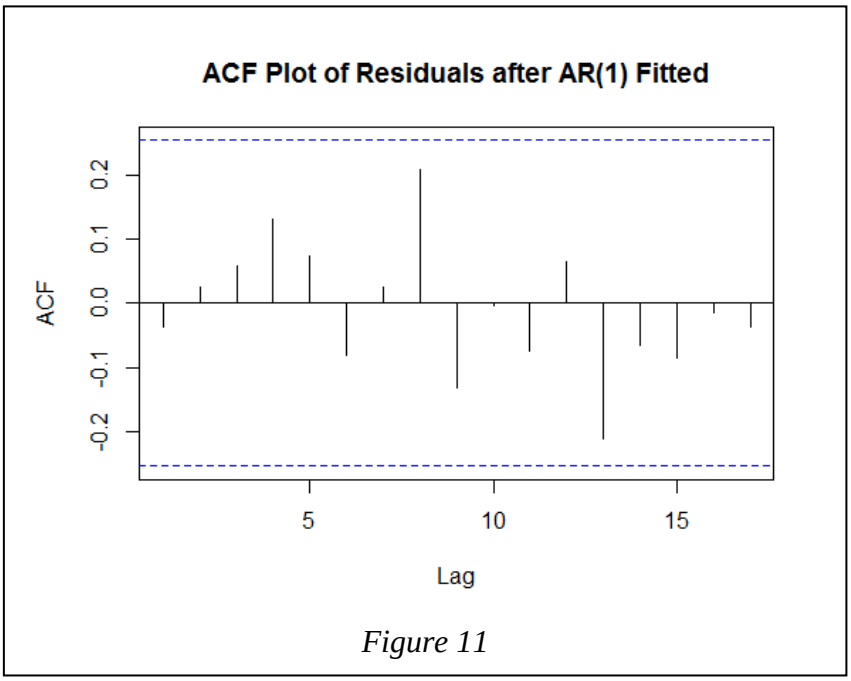

Figure 11

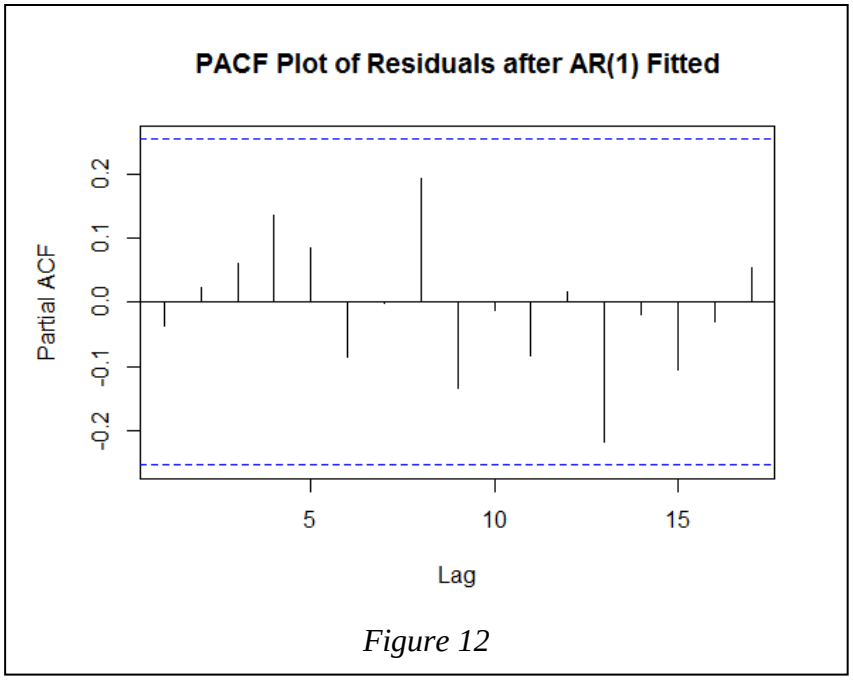

Figure 12

```
> Box.test(wn, type="Ljung-Box",lag = min(2*d, floor(n/5)) )

        Box-Ljung test

data:  wn
X-squared = 7.4718, df = 12, p-value = 0.8249
```

Figure 13

Forecasting

Using the first 60 points in the data (corresponding to August 1986 to July 1991), we forecasted the next ten time points (August 1991 to May 1992). *Figure 15* is a plot of the forecasts of the next ten months of noise. *Figure 16* shows a plot of the forecasts of the next ten months on top of the true values from the original data set, and below it, there is a plot that shows a closer view of the forecasts of the ten time points. As shown in *Figure 16*, the next ten time points are plotted over eight points of the true value including two points beyond the true values. The values of the forecasts for the next ten time points are 28.797, 29.02719, 29.01292, 28.89026, 28.80371, 28.87682, 29.87682, 29.18791, 29.75577, 30.53541, and 31.425. It is shown in *Figure 14*.

```
ex.f
Time Series:
Start = 61
End = 70
Frequency = 1
       1        2        3        4        5        6        7
28.79700 29.02719 29.01292 28.89026 28.80371 28.87682 29.18791
       8        9       10
29.75577 30.53541 31.42500
```

Figure 14

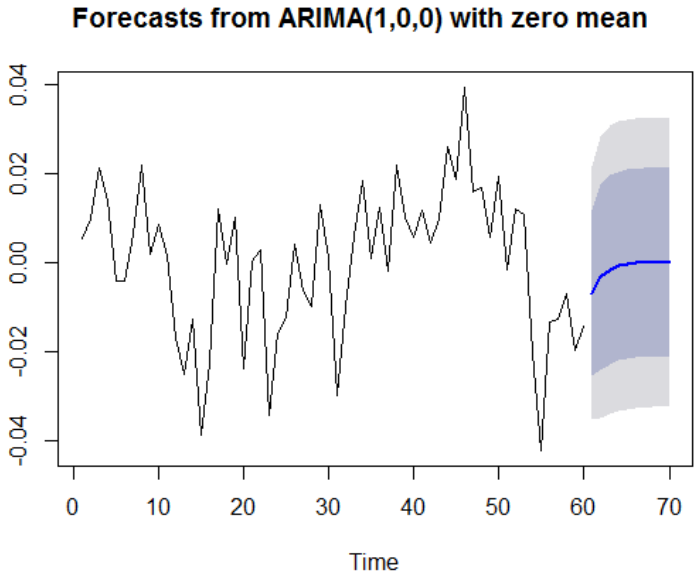

Figure 15

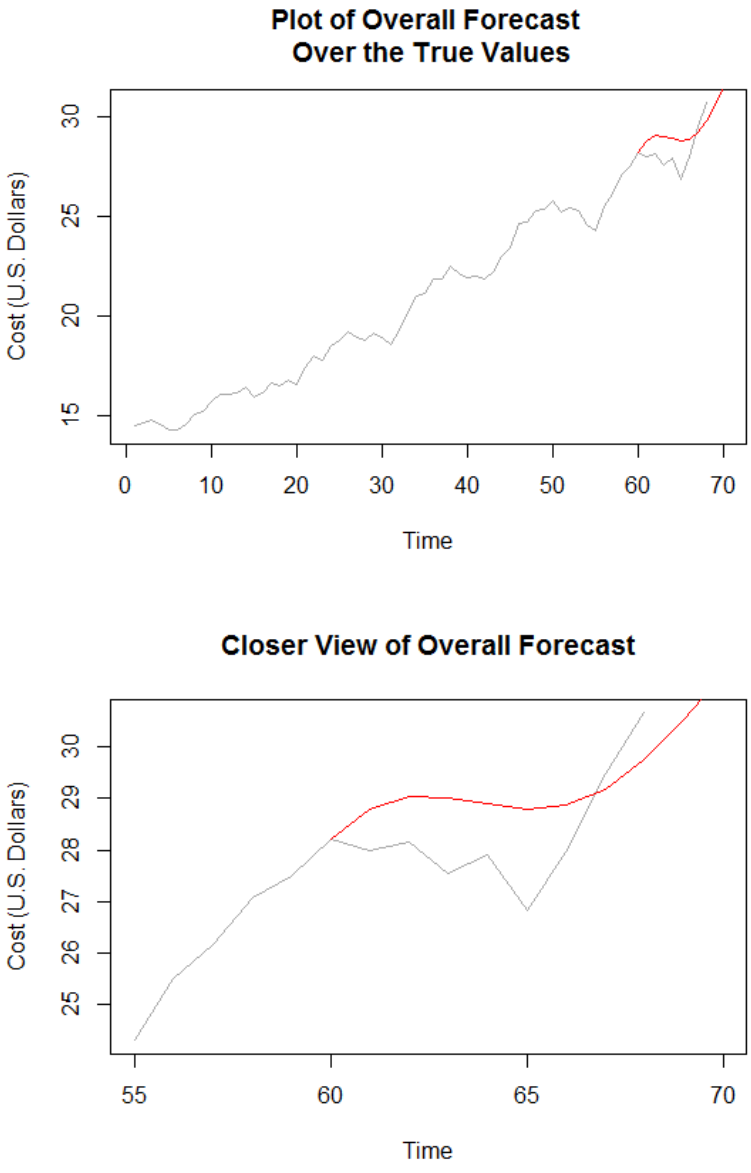

Figure 16

Before forecasting, we took a look at the Q-Q plot of the residuals and conducted the Shapiro–Wilk test to see if normality can be assumed. From the Q-Q plot in *Figure 17*, we can see that the residuals are approximately normal. The results from the Shapiro–Wilk test in *Figure 18* also provides evidence for assuming normality because the p-value is 0.08302, which is greater than 0.05, so it is not significant. We fail to reject the hypothesis that the residuals are normal. Even though we are assuming the residuals are normal, it does not necessarily mean that the forecasts are correct.

Since the overall forecast is plotted over some of the true values in *Figure 16*, we can see that the forecasting is not extremely accurate. They share a resemblance to the true values, but the forecasted points are just shifted upwards. This may have happened because the forecast overestimated the increase in variance of the data as a steady increase of variance. As a result, our ten forecasted time points are mostly greater than their true values, and it appears to continue to increase for future years.

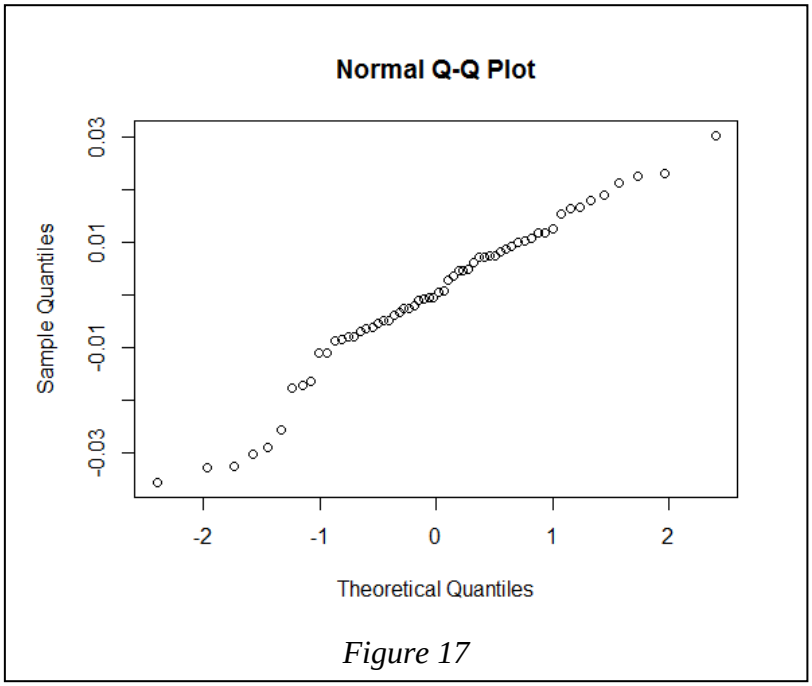

Figure 17

```
> shapiro.test(wn)

        Shapiro-Wilk normality test

data:  wn
W = 0.96502, p-value = 0.08302
```

Figure 18

Appendix

```
library(TSA)
data(prescrip)

x = as.vector(prescrip)
t = as.vector(time(prescrip))
plot(t, x, type ="l", main = "Monthly US Prescription Costs (1986-1992)",
     xlab = "Time (years)", ylab = "Cost (U.S. dollars)")

prescrip.part = window(prescrip, start = c(1986, 8), end = c(1991, 7))
x = as.vector(prescrip.part)
t = as.vector(time(prescrip.part))
plot(t, x, type = "l", xlab = "Time (years)", ylab = "Costs (U.S. dollars)",
     main = "Monthly US Prescription Costs (August 1986 - July 1991)")

# transform the data with a ln transformation to stabilize the variance
x = log(x)
plot(t, x, type = "l", xlab = "Time (years)", ylab = "", main = "Log Transformed")

# remove trend by fitting polynomial
t2 = t^2
trend.fit = lm(x ~ t + t2)

# plot fitted trend
plot(t, x, type = "l", xlab = "Time (years)", ylab = "",
     main = "Log Transformed Data and Fitted Trend")
lines(t, fitted(trend.fit))

# plot residuals after trend removed
y = residuals(trend.fit)
plot(t, y, type = "l", main = "Residuals After Trend Removed",
     xlab = "Time (years)", ylab = "")

# remove seasonal component by using a sum of harmonics
n = length(t)
t = 1:length(y)
t = (t) / n
d = 12                    # number of time points in each season
n.harm = 6               # set to [d/2]
harm = matrix(nrow = length(t), ncol = 2*n.harm)
for(i in 1:n.harm){
  harm[,i*2-1] = sin(n/d * i * 2*pi*t)
  harm[,i*2] = cos(n/d * i * 2*pi*t)
}
colnames(harm) = paste0(c("sin", "cos"), rep(1:n.harm, each = 2))

# fit on all of the sines and cosines
dat = data.frame(y, harm)
fit = lm(y~., data = dat)
summary(fit)

# setup full model and model with only an intercept
full = lm(y~., data = dat)
reduced = lm(y~1, data = dat)

# stepwise regression starting with full model
fit.back = step(full, scope = formula(reduced), direction = "both")
fit.back
```

```r
# plot residuals after trend removed and fitted seasonal component
t = as.vector(time(prescrip.part))
plot(t, y, type = "l", col = "darkgrey", xlab = "Time (years)", ylab = "",
     main = "Residuals After Trend Removed and Fitted Seasonal Component")
lines(t, fitted(fit.back), col = "red")

# plot residuals after seasonal component removed
ts.plot(residuals(fit.back), main = "After Seasonal Components Removed",
        ylab = "", xlab = "t")

z = residuals(fit.back)
acf(z, main = "ACF of Residuals")
pacf(z, main = "PACF of Residuals")

# use H-K algorithm to determine best model
require(forecast)
arma.fit = auto.arima(z, allowmean = FALSE, trace = TRUE, stepwise = FALSE)
arma.fit

# examine the residuals of the arma fit
wn = resid(arma.fit)

acf(wn, na.action = na.pass, main = "ACF Plot of Residuals after AR(1) Fitted")
pacf(wn, na.action = na.pass, main = "PACF Plot of Residuals after AR(1) Fitted")

Box.test(wn, type="Ljung-Box",lag = min(2*d, floor(n/5)) )

qqnorm(wn)
shapiro.test(wn)

# forecasting the next ten time points

noise.f = forecast(arma.fit, 10)

plot(noise.f, xlab = "Time")

# forecast the seasonal component with the noise
season.f = fitted(fit.back)[1:10]
plot(season.f + noise.f$mean)

# forecast trend
t.f = 61:70
t.f2 = t.f^2

n = length(x)
t = 1:n
t2 = t^2
trend.fit = lm(x ~ t + t2)
trend.f = predict(trend.fit, newdata = data.frame(t = t.f, t2 = t.f2))

# add all components together to get the overall forecast
x.f = trend.f + season.f + noise.f$mean

# undo the natural log transformation
ex.f = exp(x.f)
ex.f

# plot the forecasts on top of the true values
x = as.vector(prescrip)
plot(1:70, x[1:70], type="l", col="darkgrey", ylab = "Cost (U.S. Dollars)",
     main = "Plot of Overall Forecast \n Over the True Values",
     xlab = "Time")
lines(60:70, c(x[60], ex.f), col="red")

# closer view
plot(55:70, x[55:70], type="l", col="darkgrey", ylab = "Cost (U.S. Dollars)",
     xlab = "Time", main = "Closer View of Overall Forecast")
lines(60:70, c(x[60], ex.f), col="red")
```