Project 1

Xiang Yang Ng, Muiz Rahemtullah

April 18, 2018

Introduction

Our data is Consumer Price Index for all urban consumers on the purchasing power of the US dollar from January 1950 to January 2018. We obtained the data from the U.S. Bureau of Labor Statistics. THe data is last updated on April 17 2018.

Incorporating all the libraries into our project and setting the directory

```
#downloading all the libraries and setting the directory
setwd("/Users/Lenovo/Desktop/Econ 144/Project 1")
rm(list=ls(all=TRUE))
library(lattice)
library(foreign)
library(MASS)
library(car)
```

```
## Loading required package: carData
```

```
require(stats)
require(stats4)
```

```
## Loading required package: stats4
```

```
library(KernSmooth)
```

```
## KernSmooth 2.23 loaded
## Copyright M. P. Wand 1997-2009
```

```
library(fastICA)
library(cluster)
library(leaps)
library(mgcv)
```

```
## Loading required package: nlme
```

```
## This is mgcv 1.8-23. For overview type 'help("mgcv-package")'.
```

```r
library(rpart)
library(pan)
library(mgcv)
library(DAAG)
```

```
##
## Attaching package: 'DAAG'
```

```
## The following object is masked from 'package:car':
##
##     vif
```

```
## The following object is masked from 'package:MASS':
##
##     hills
```

```r
library(tis)
```

```
##
## Attaching package: 'tis'
```

```
## The following object is masked from 'package:mgcv':
##
##     ti
```

```r
require(datasets)
require(graphics)
library("forecast")
```

```
##
## Attaching package: 'forecast'
```

```
## The following object is masked from 'package:tis':
##
##     easter
```

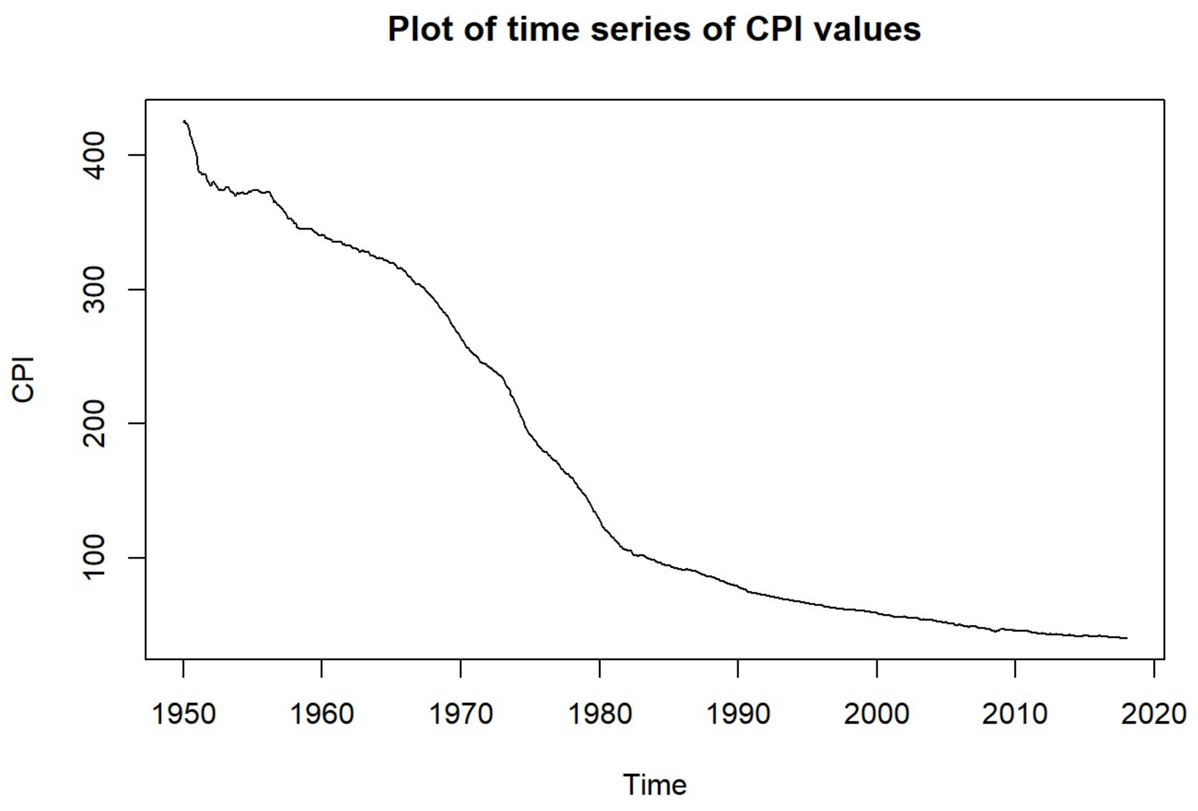
```
## The following object is masked from 'package:nlme':
##
##     getResponse
```

Feeding in the data

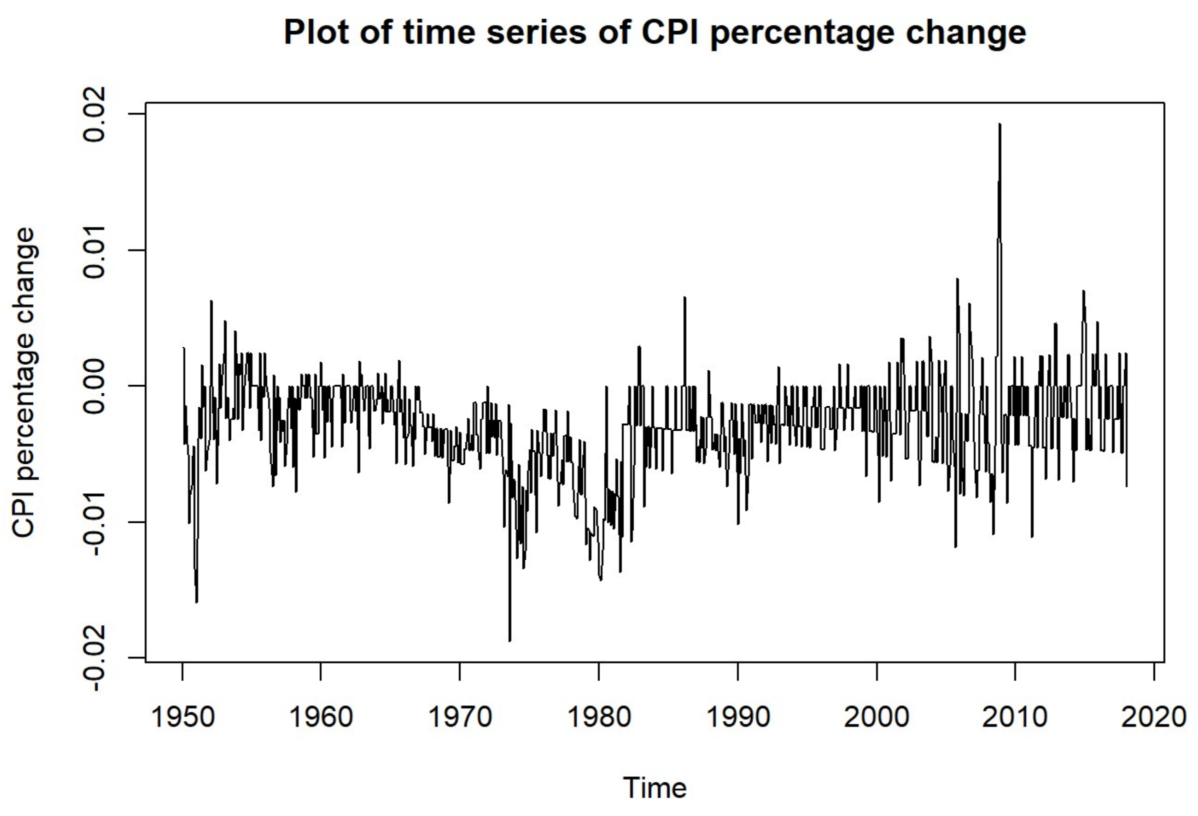
```
#Obtain the data, clean it and construct the time series for the CPI
data = read.csv("Consumer Price Index for All Urban Consumers Purchasing Power of t
he Consumer Dollar 1950_1-2018_1.csv")
colnames(data)[2] = "CPI"
attach(data)
CPI_ts = ts(CPI, start = 1950, freq = 12)
```

a. and b)

```
#Plotting both the time series of the CPI and it's difference of the log
plot(CPI_ts, main = "Plot of time series of CPI values", ylab = "CPI")
```



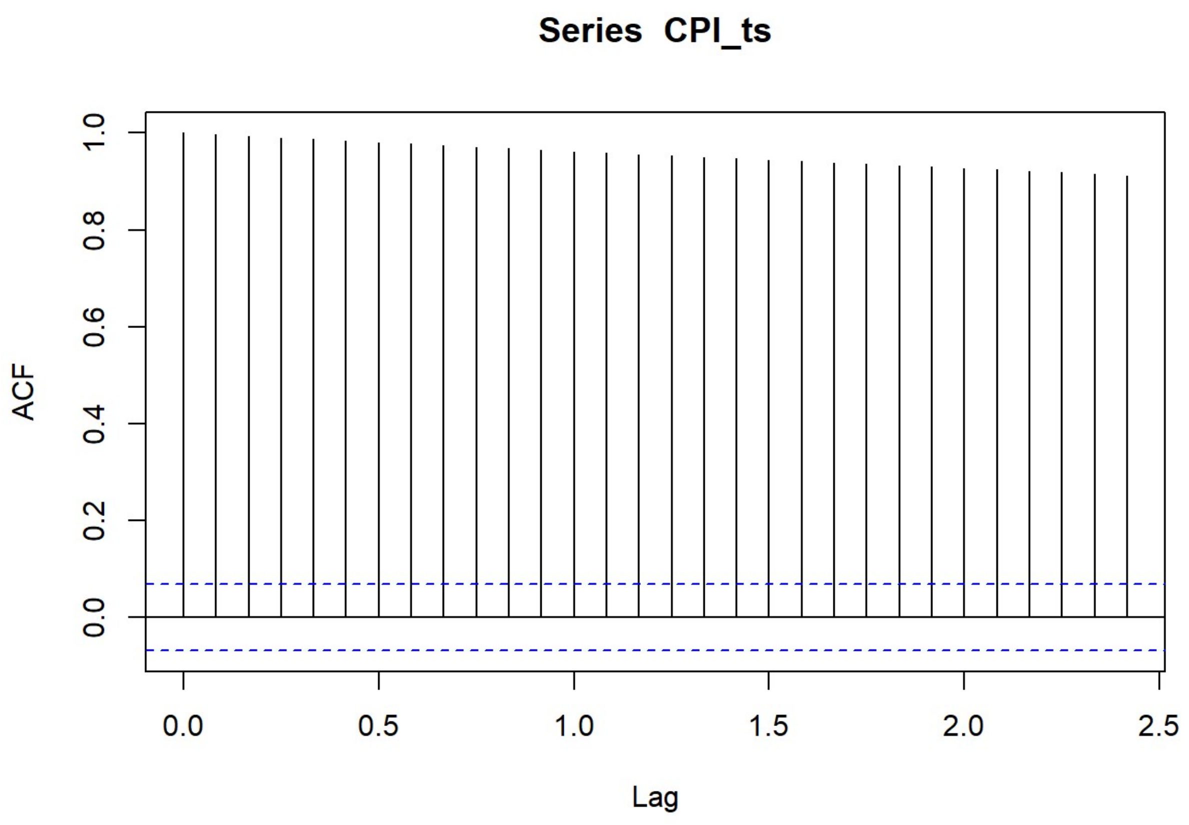
```
CPI_ts_diff = diff(log(CPI_ts))
plot(CPI_ts_diff, main = "Plot of time series of CPI percentage change", ylab = "CP
I percentage change")
```



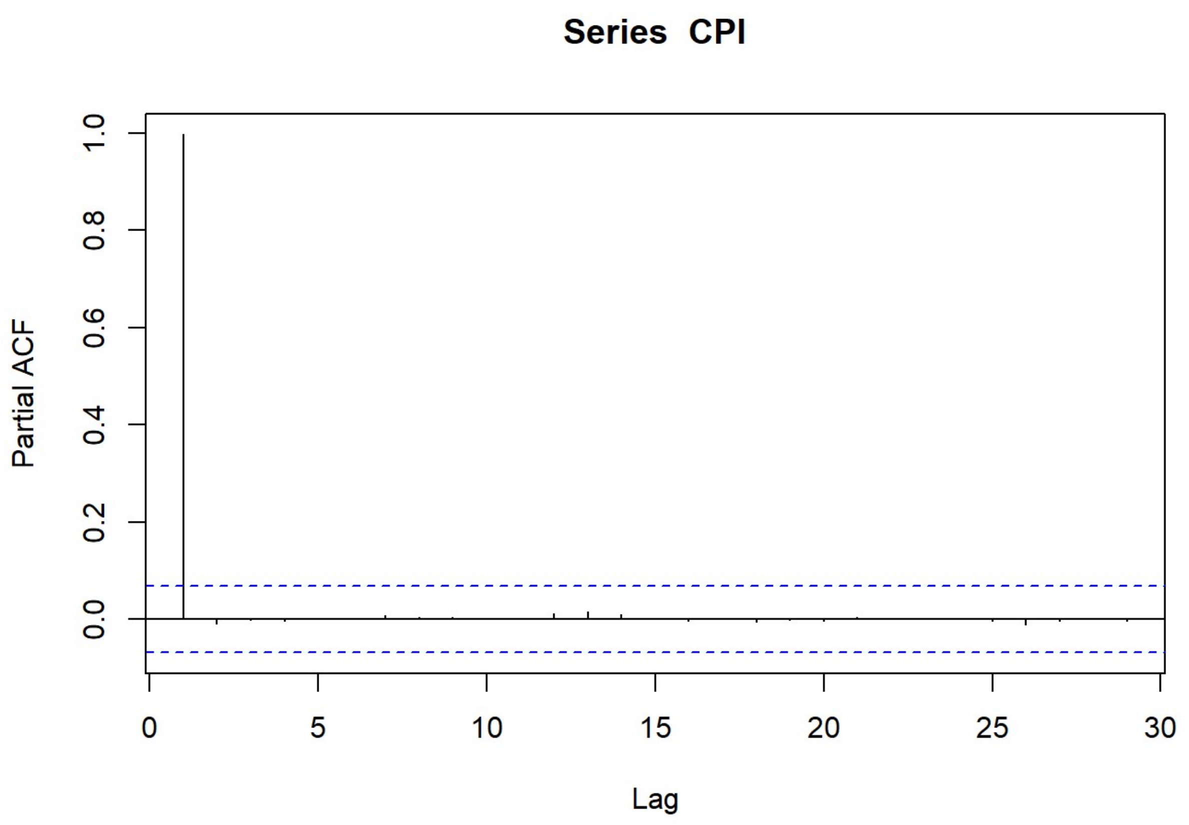
The original plot shows that the time series is not covariance stationary since the variance at each specific time point varies. However if we take a difference of the log of the time series, we could clearly see that this difference is covariance stationary.

c.

```
#Plotting the ACF and the PACF of the time series
acf(CPI_ts)
```


Series CPI_ts
ACF
Lag
0.0
0.2
0.4
0.6
0.8
1.0
0.0
0.5
1.0
1.5
2.0
2.5

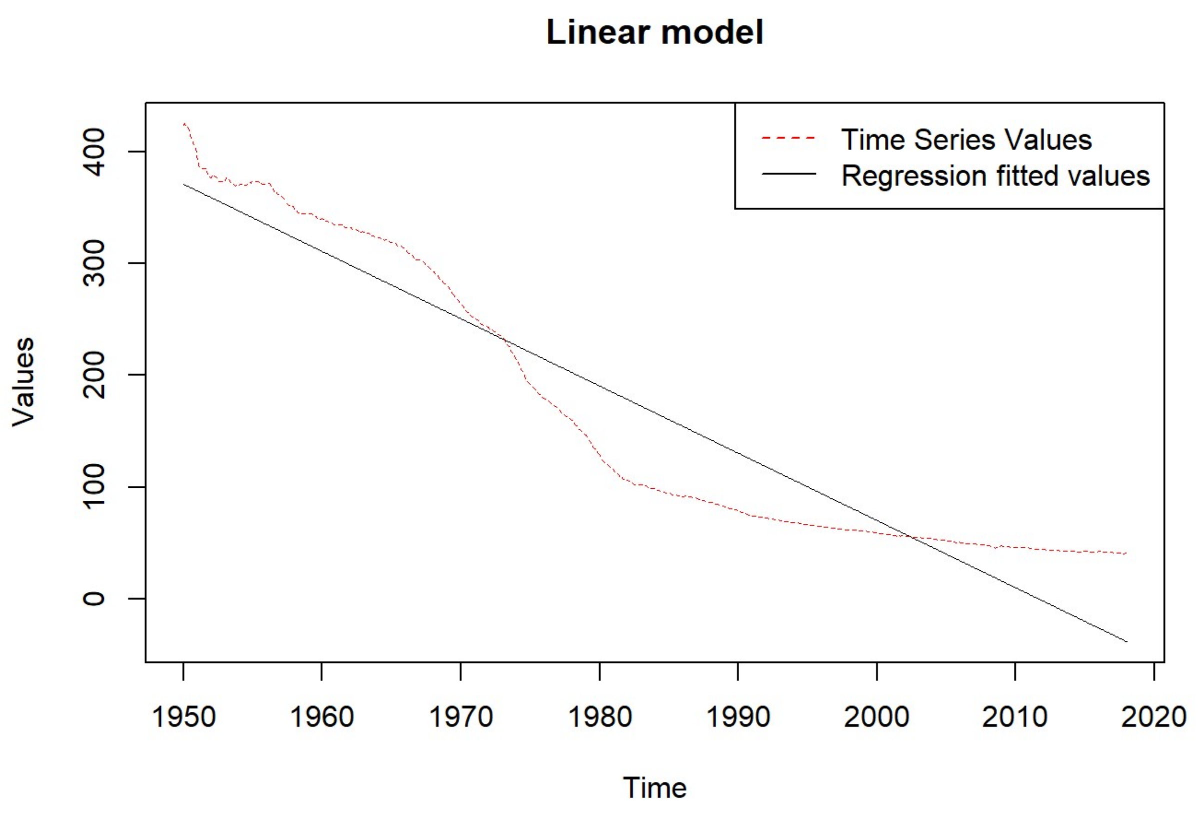
```
pacf(CPI)
```


Series CPI
Partial ACF
Lag
0.0
0.2
0.4
0.6
0.8
1.0
0
5
10
15
20
25
30

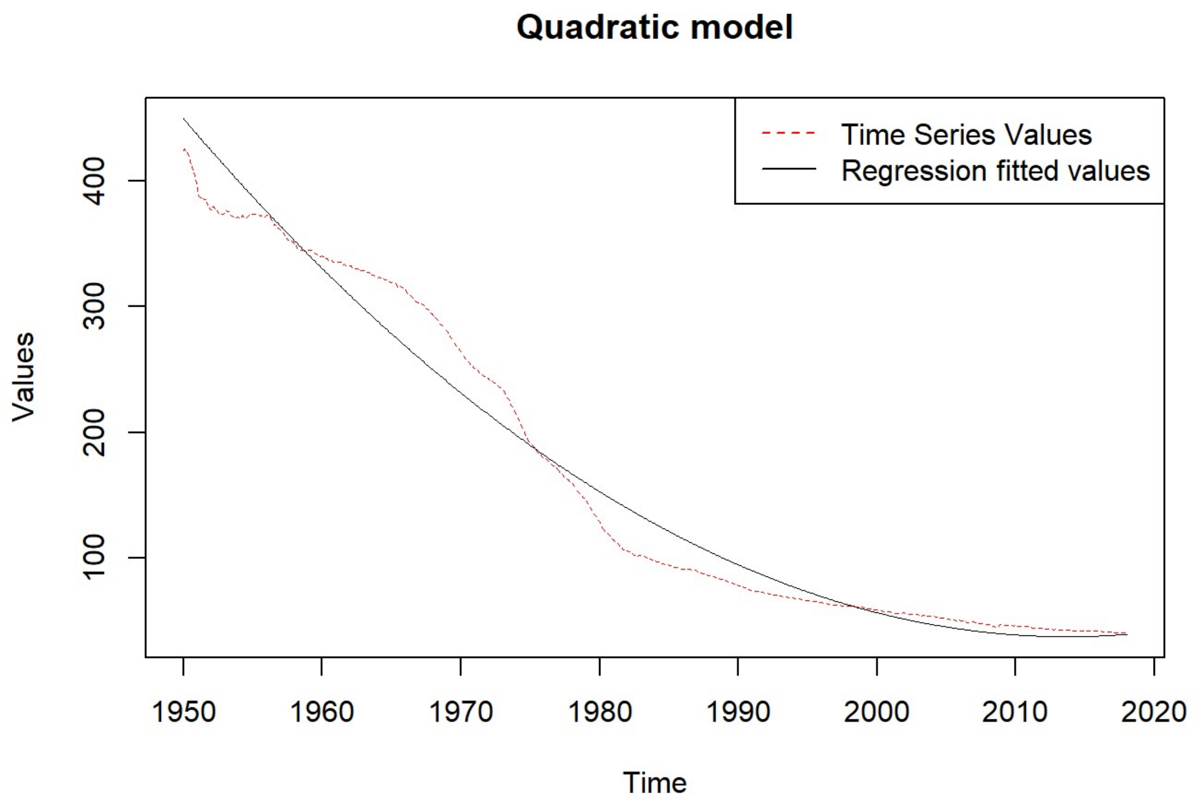
The ACF shows that there is significant correlation for lag 1, which decreases as the lag increases. This means that subsequence lags have correlation which only depends on the first lag. This is shown by the PACF, which is 1 for the first lag and zero otherwise.

d.

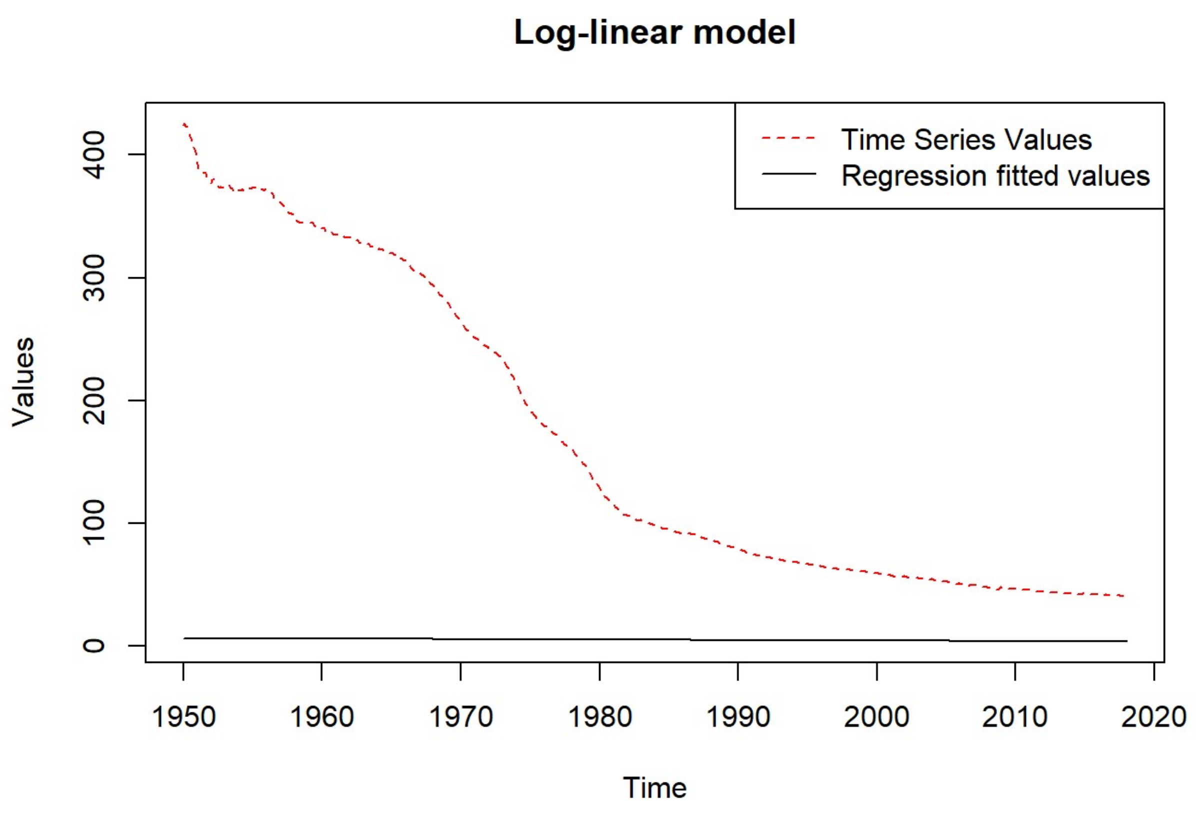
```
#we regress time series of the CPI with a linear model and nonlinear models such as
quadratic, logatithmatic, exponential, log-linear

t = seq(1950, 2018, length = length(CPI_ts))
y1 = lm(CPI_ts~t)

y2 = lm(CPI_ts~t+I(t^2))

y3=lm(log(CPI_ts) ~ t)

ds=data.frame(x=t, y=CPI_ts)
y4=nls(y ~ exp(a + b * t),data=ds, start = list(a = 0, b = 0))

y5=lm(CPI_ts ~ log(t))

#plotting the graphs of the time series values with the regression fitted values fo
r all of the models
matplot(t, cbind(y1$fitted.values, CPI), type = 'l', xlab = "Time", ylab = "Value
s", lwd = 0.1)
title(main = "Linear model")
legend("topright", legend = c("Time Series Values", "Regression fitted values"), co
l = c("red", "black"), lty = c(2,1))
```



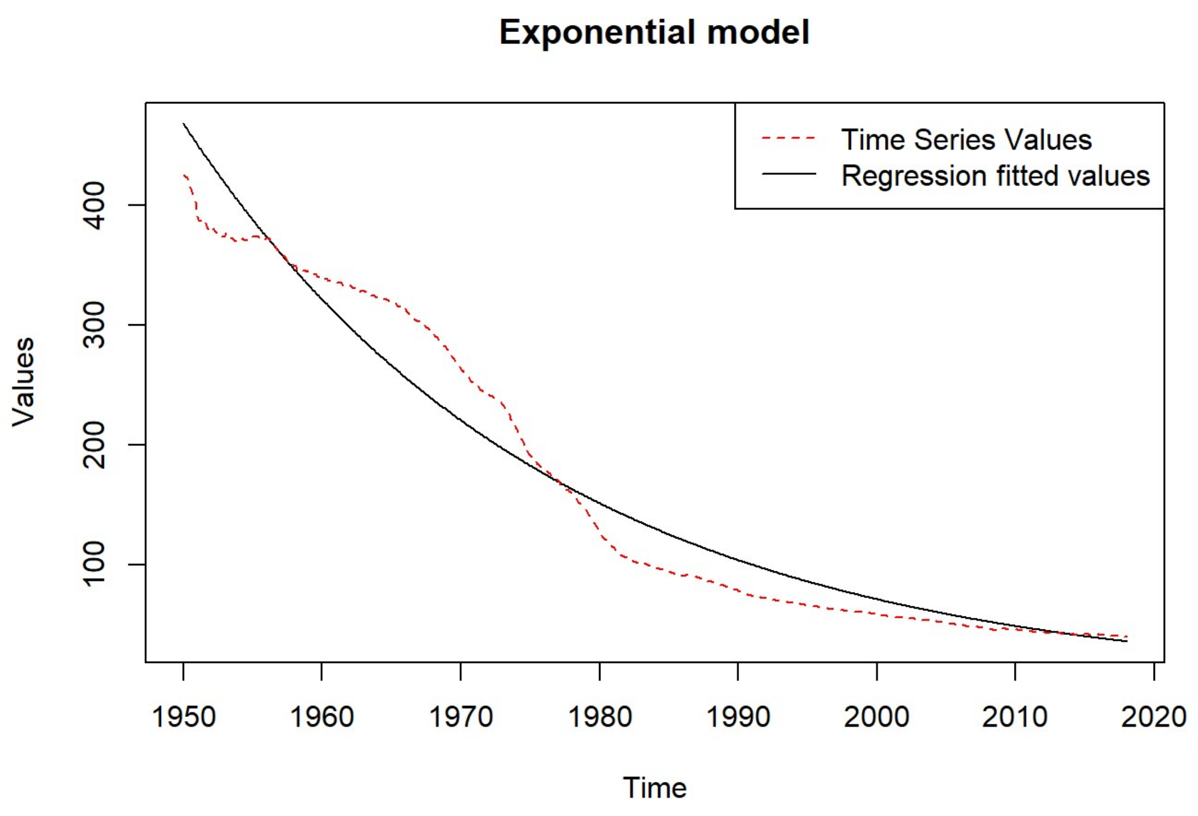
```
matplot(t, cbind(y2$fitted.values, CPI), type = 'l', xlab = "Time", ylab = "Value
s", lwd = 0.1)
title(main = "Quadratic model")
legend("topright", legend = c("Time Series Values", "Regression fitted values"), co
l = c("red", "black"), lty = c(2,1))
```



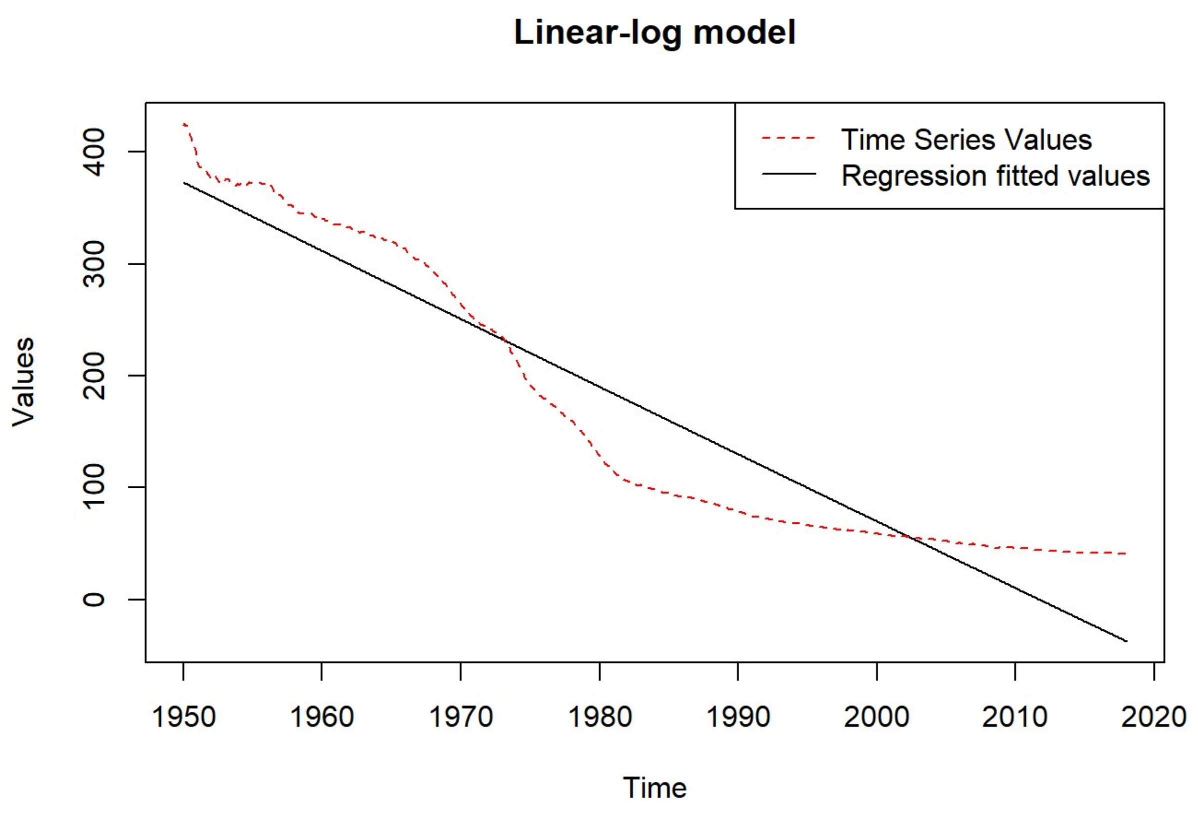
```
matplot(t, cbind(y3$fitted.values, CPI), type = 'l',xlab = "Time", ylab = "Values")
title(main = "Log-linear model")
legend("topright", legend = c("Time Series Values", "Regression fitted values"), co
l = c("red", "black"), lty = c(2,1))
```



```
matplot(t, cbind(predict(y4, list(x = ds$x)), CPI), type = 'l',xlab = "Time", ylab
= "Values")
title(main = "Exponential model")
legend("topright", legend = c("Time Series Values", "Regression fitted values"), co
l = c("red", "black"), lty = c(2,1))
```



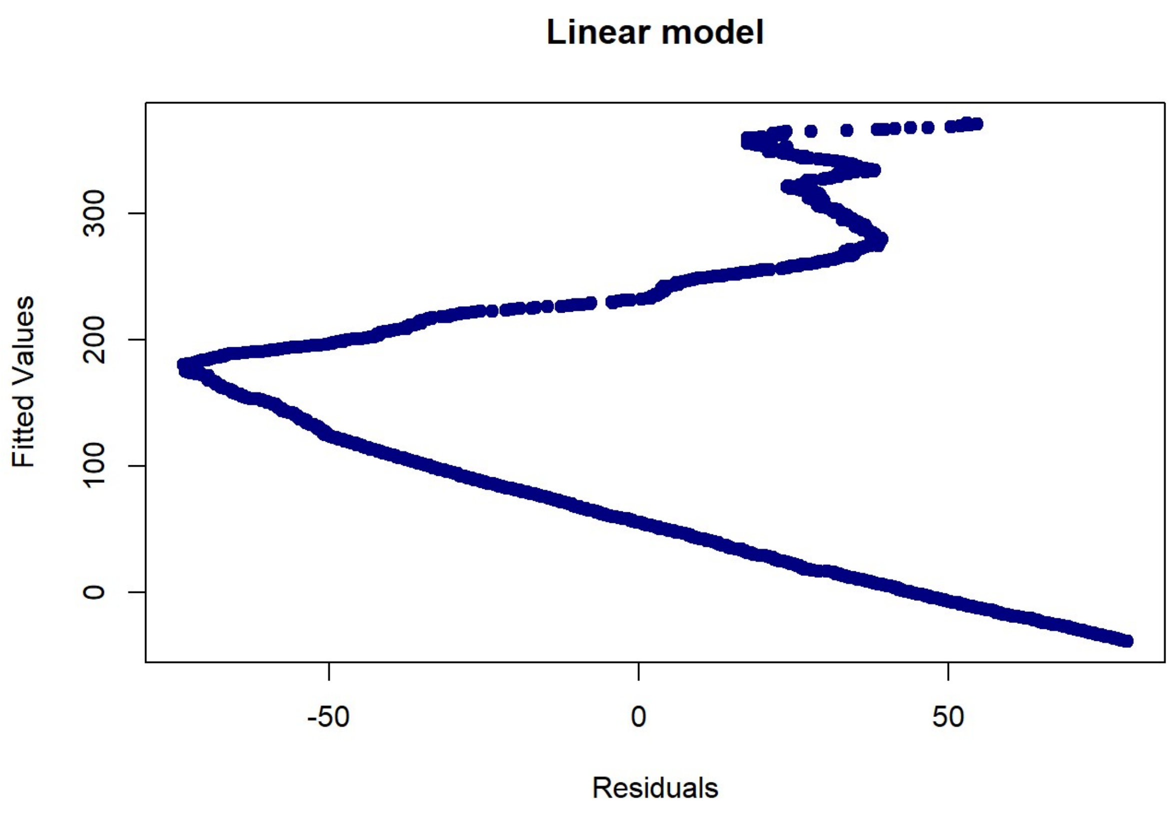
```
matplot(t, cbind(y5$fitted.values, CPI), type = 'l',xlab = "Time", ylab = "Values")
title(main = "Linear-log model")
legend("topright", legend = c("Time Series Values", "Regression fitted values"), co
l = c("red", "black"), lty = c(2,1))
```



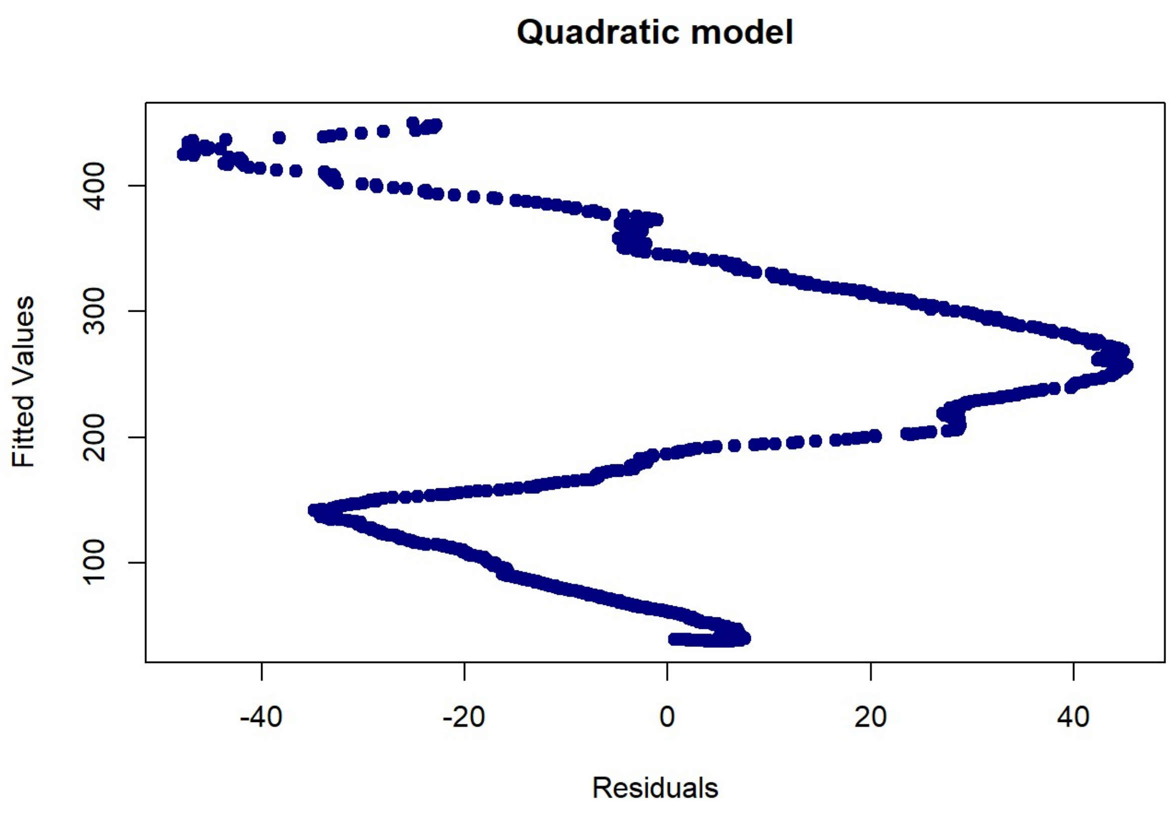
All of the lines, except the log-linear, seems to follow the trend of the data well. This means that these models fit the data well. We have to see the summary for the log-linear to determine whether the model fits well to the data.

e.

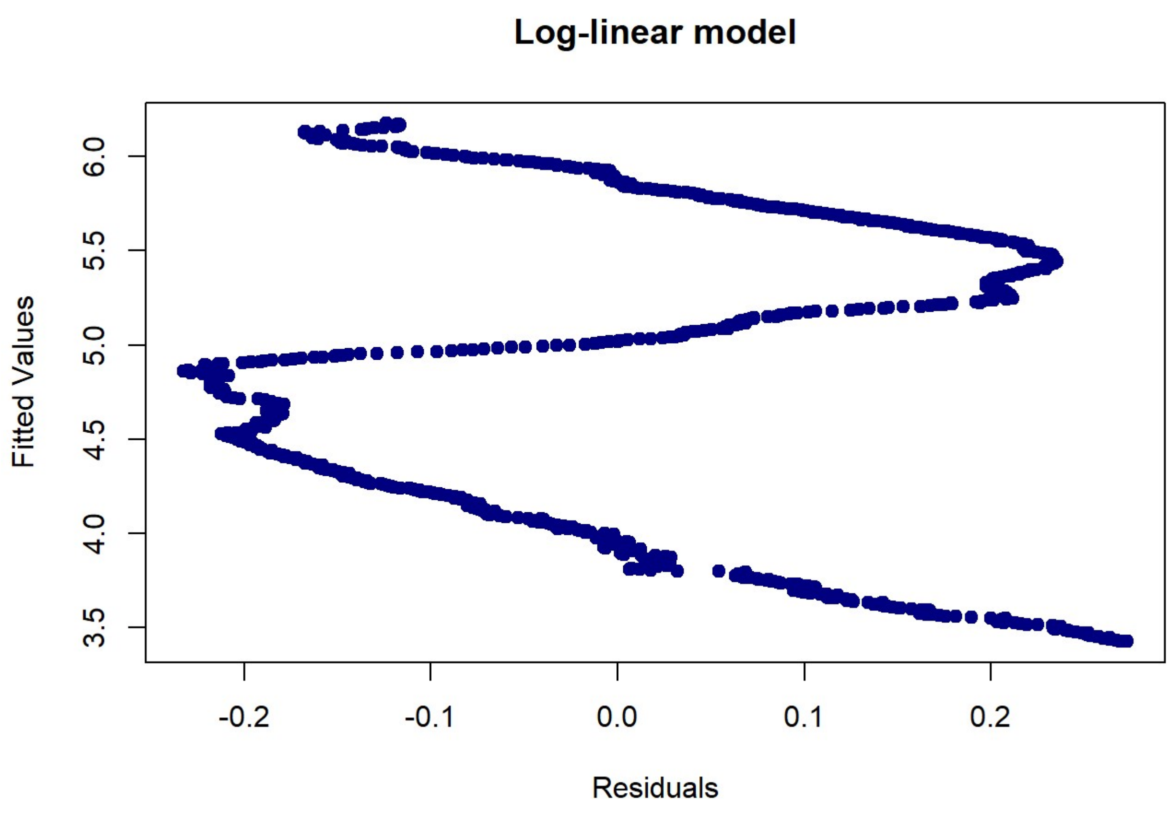
```
#Plotting the scatterplot of the residuals of all the models
plot(resid(y1), fitted(y1), xlab = "Residuals",
    ylab = "Fitted Values", main = "Linear model",
    pch = 19, col = "navyblue")
```



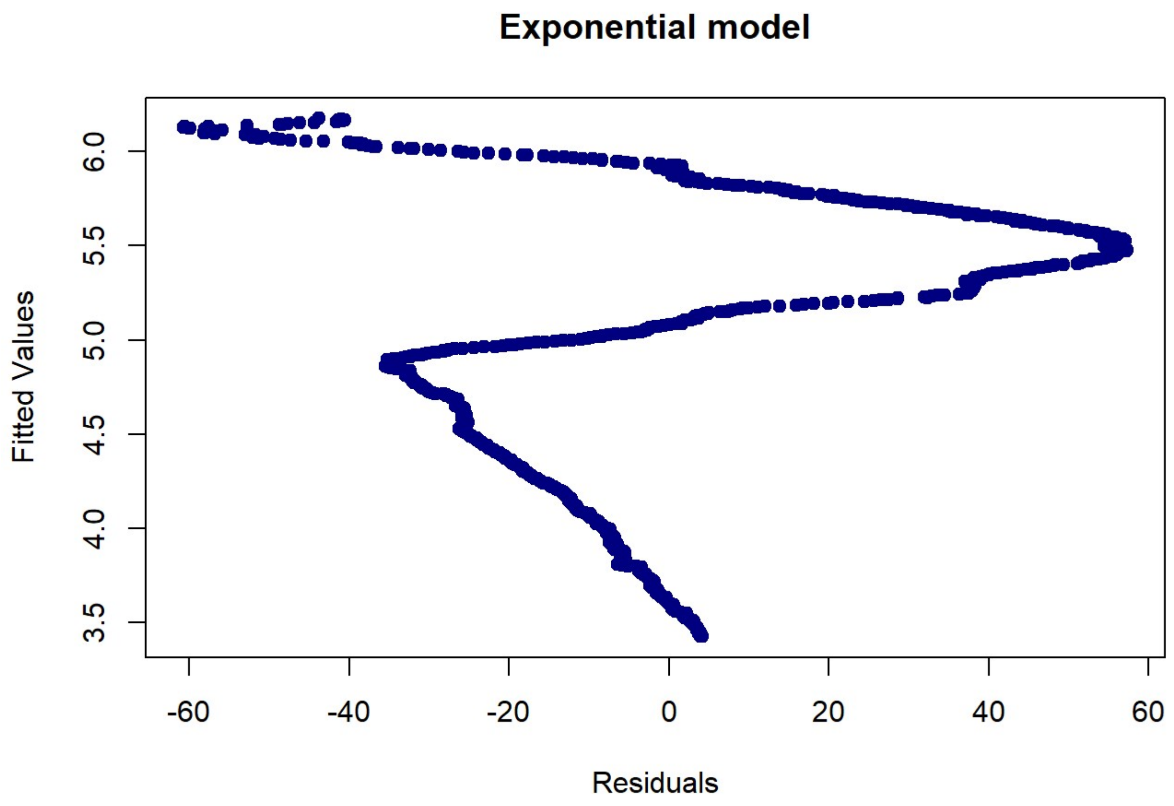
```
plot(resid(y2), fitted(y2), xlab = "Residuals",
    ylab = "Fitted Values", main = "Quadratic model",
    pch = 19, col = "navyblue")
```



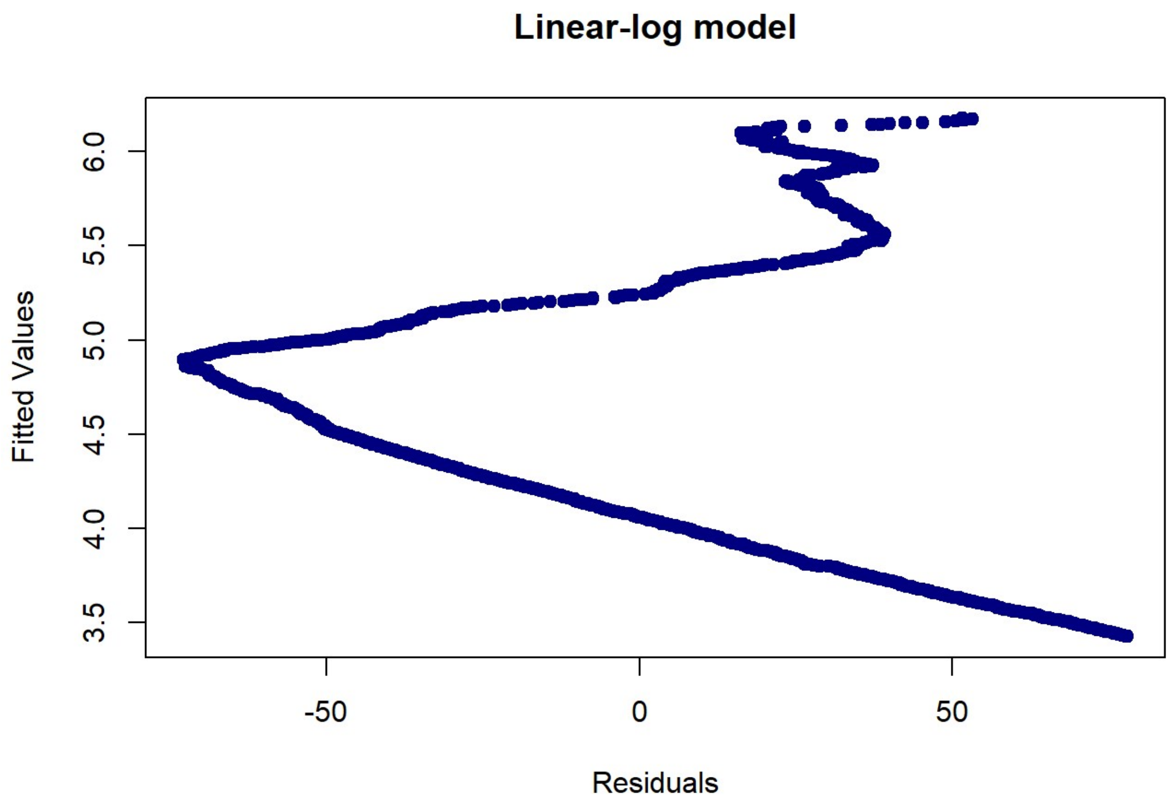
```
plot(resid(y3), fitted(y3), xlab = "Residuals",
    ylab = "Fitted Values", main = "Log-linear model",
    pch = 19, col = "navyblue")
```



```
plot(resid(y4), fitted(y3), xlab = "Residuals",
    ylab = "Fitted Values", main = "Exponential model",
    pch = 19, col = "navyblue")
```



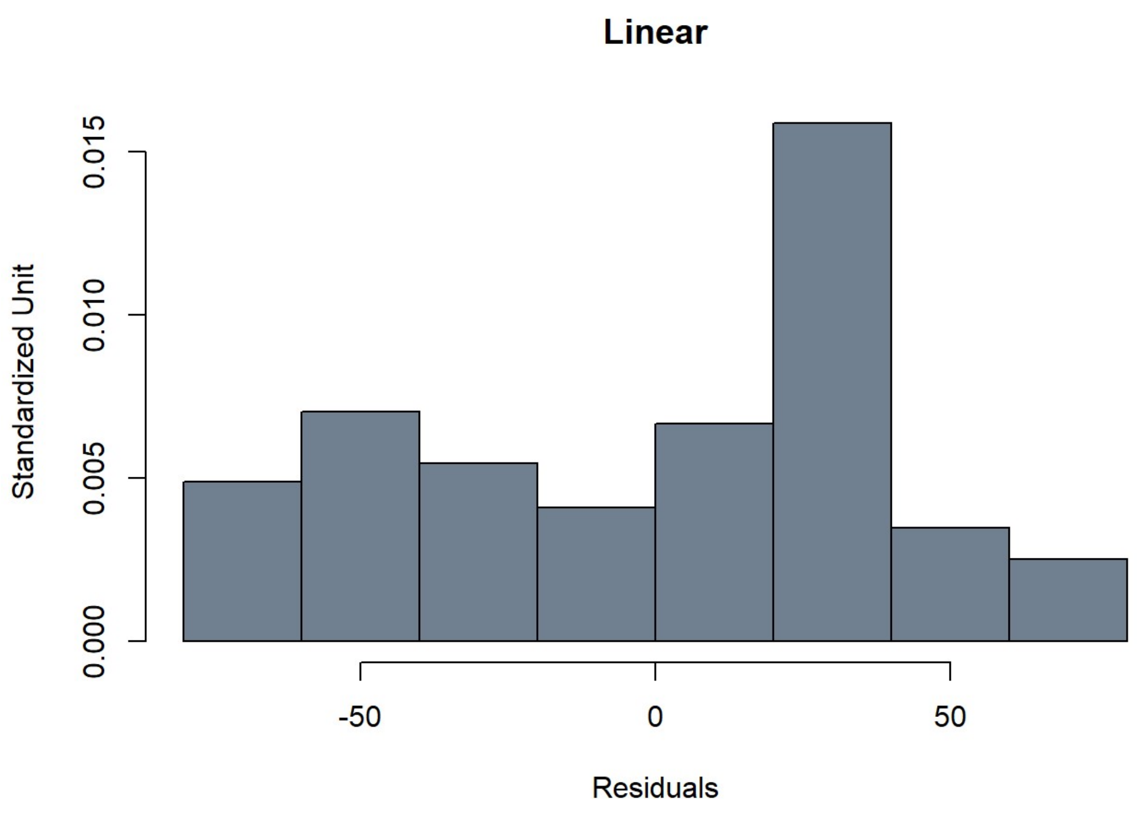
```
plot(resid(y5), fitted(y3), xlab = "Residuals",
    ylab = "Fitted Values", main = "Linear-log model",
    pch = 19, col = "navyblue")
```



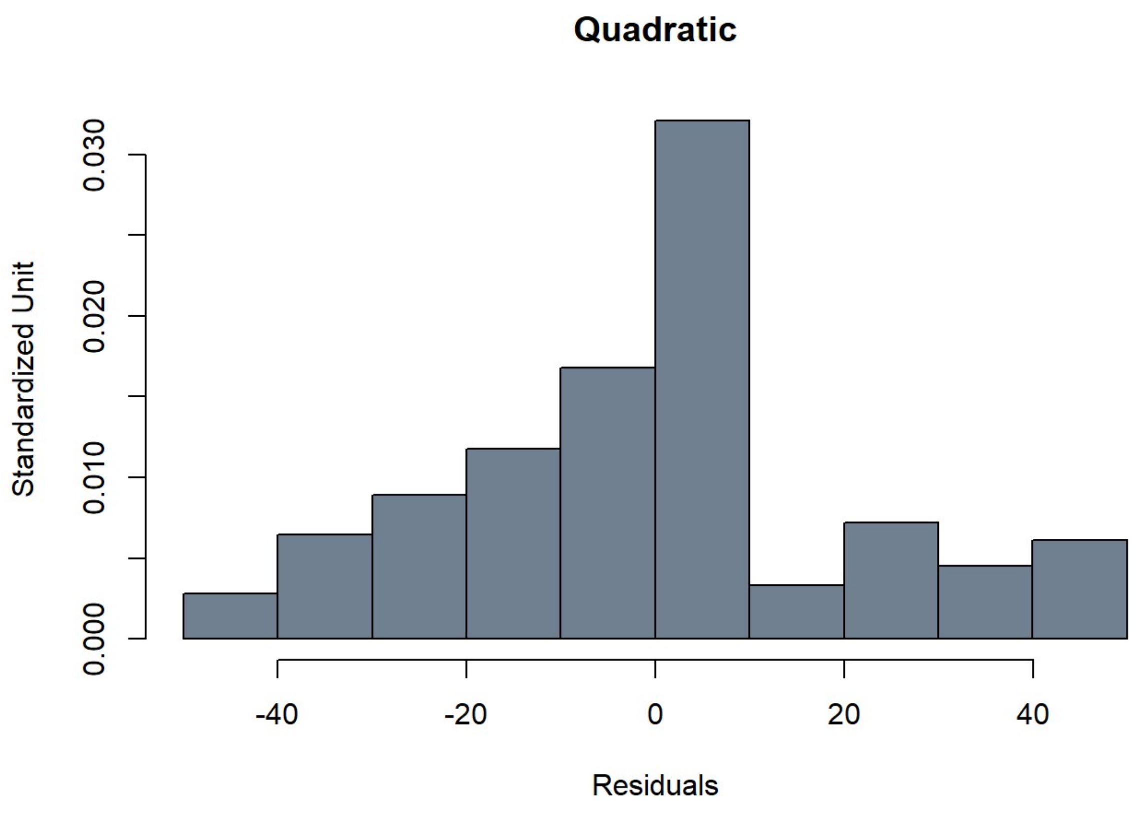
For all of the models, the residuals follow a non-linear trend, which means that there are still dynamics in the data that we have not accounted for. For both the linear and the linear-log models, there seems to be a little more residuals on the right, which indicate that the models underpredict the data, while for both the quadratic and exponential models, there seems to be little more residuals on the left, indicating that the models overpredict the data. The residuals are most balanced on both sides are from the log-linear model.

f.

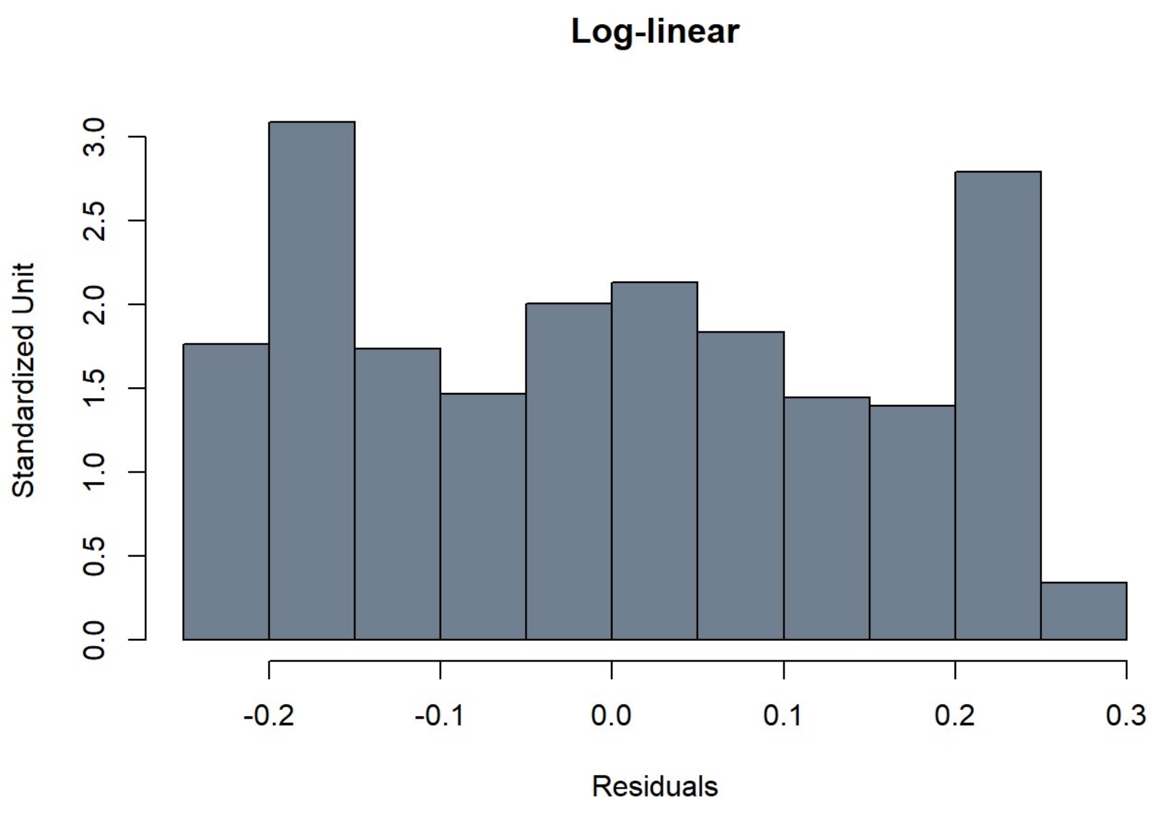
```
#Plotting the histogram of the residuals of all the models
truehist(resid(y1), main = "Linear", col = "slategrey",
    xlab = "Residuals", ylab = "Standardized Unit")
```



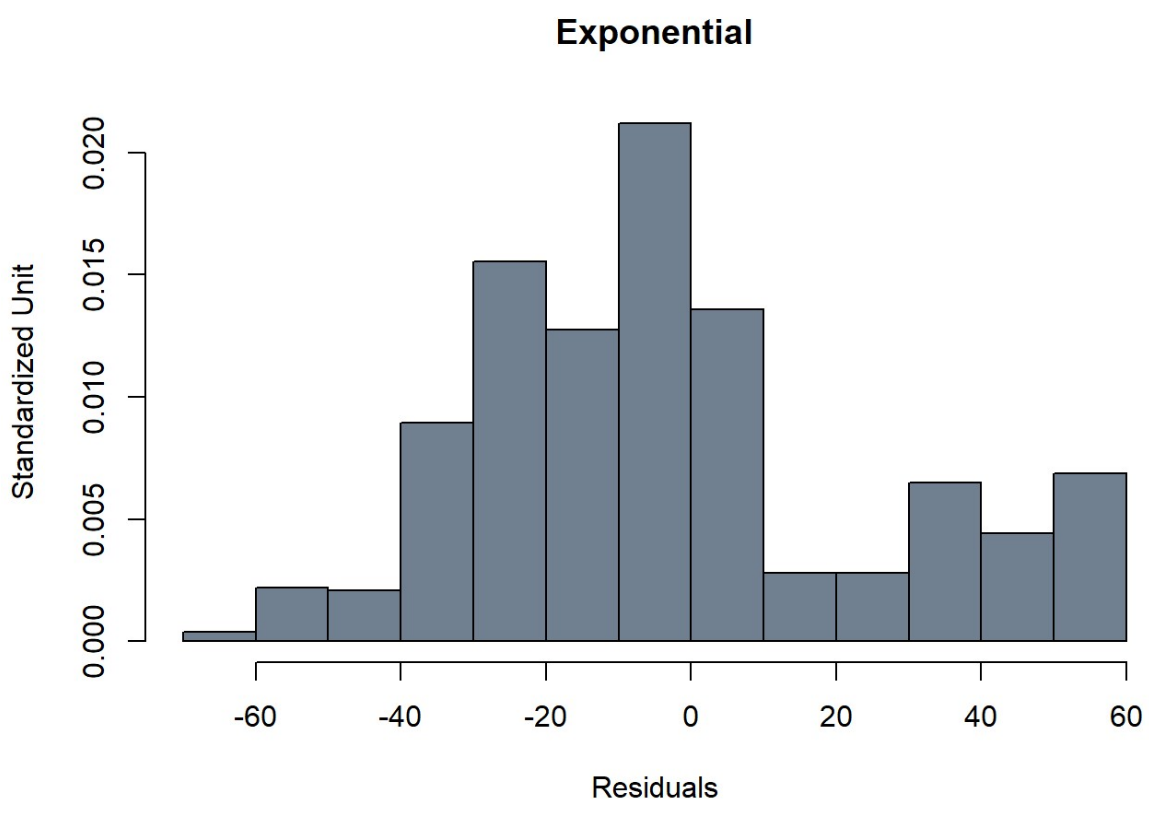
```
truehist(resid(y2), main = "Quadratic", col = "slategrey",
    xlab = "Residuals", ylab = "Standardized Unit")
```



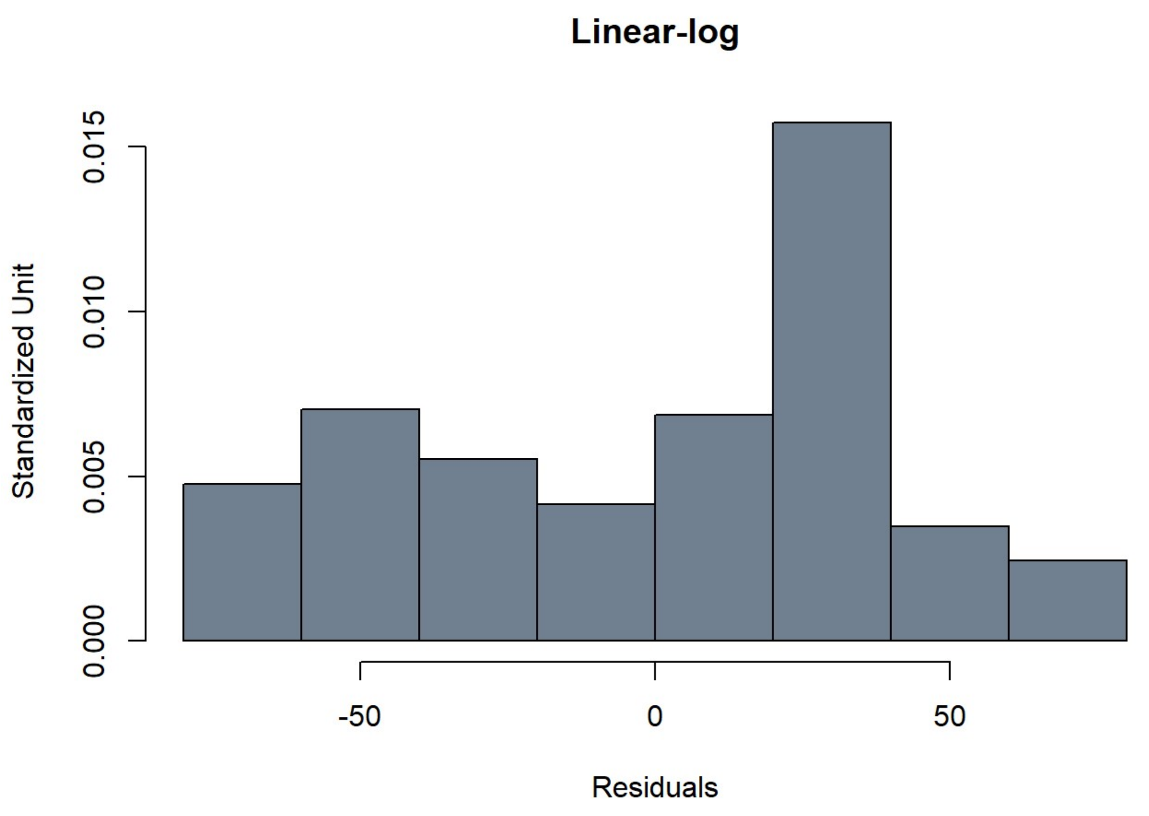
```
truehist(resid(y3), main = "Log-linear", col = "slategrey",
    xlab = "Residuals", ylab = "Standardized Unit")
```



```
truehist(resid(y4), main = "Exponential", col = "slategrey",
    xlab = "Residuals", ylab = "Standardized Unit")
```



```
truehist(resid(y5), main = "Linear-log", col = "slategrey",
    xlab = "Residuals", ylab = "Standardized Unit")
```



These histograms fully mimics the plot from e). The analysis for each model should then be the same.

f.

```
#Displaying the summary of the regression for each model
summary(y1)
```

```
##
## Call:
## lm(formula = CPI_ts ~ t)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -73.441 -37.785   9.315  32.693  78.807
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)  1.213e+04  1.450e+02   83.61   <2e-16 ***
## t           -6.029e+00  7.311e-02  -82.47   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 41.07 on 815 degrees of freedom
## Multiple R-squared:  0.893,  Adjusted R-squared:  0.8929
## F-statistic:  6801 on 1 and 815 DF,  p-value: < 2.2e-16
```

```
summary(y2)
```

```
##
## Call:
## lm(formula = CPI_ts ~ t + I(t^2))
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -47.737 -15.009   1.788   6.932  45.268
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)  4.114e+05  8.508e+03   48.36   <2e-16 ***
## t           -4.086e+02  8.577e+00  -47.64   <2e-16 ***
## I(t^2)       1.014e-01  2.162e-03   46.93   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 21.35 on 814 degrees of freedom
## Multiple R-squared:  0.9711, Adjusted R-squared:  0.9711
## F-statistic: 1.369e+04 on 2 and 814 DF,  p-value: < 2.2e-16
```

```
summary(y3)
```

```
##
## Call:
## lm(formula = log(CPI_ts) ~ t)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.23258 -0.14710 -0.00038  0.13035  0.27300
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) 84.9470434  0.5326195   159.5   <2e-16 ***
## t           -0.0403963  0.0002684  -150.5   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1508 on 815 degrees of freedom
## Multiple R-squared:  0.9653, Adjusted R-squared:  0.9652
## F-statistic: 2.265e+04 on 1 and 815 DF,  p-value: < 2.2e-16
```

```
summary(y4)
```

```
##
## Formula: y ~ exp(a + b * t)
##
## Parameters:
##     Estimate Std. Error t value Pr(>|t|)
## a 79.4715383  0.7530855  105.53   <2e-16 ***
## b -0.0376015  0.0003837  -98.01   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 27.63 on 815 degrees of freedom
##
## Number of iterations to convergence: 11
## Achieved convergence tolerance: 4.312e-06
```

```
summary(y5)
```

```
##
## Call:
## lm(formula = CPI_ts ~ log(t))
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -72.879 -37.275   9.424  32.314  77.908
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  91097.3     1089.2   83.64   <2e-16 ***
## log(t)      -11975.9      143.5  -83.48   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 40.62 on 815 degrees of freedom
## Multiple R-squared:  0.8953, Adjusted R-squared:  0.8952
## F-statistic:  6970 on 1 and 815 DF,  p-value: < 2.2e-16
```

All the models fit the data pretty well, with at least a 0.85 for the R^2. From the t- and F-distribution, we can see that the time for all models are also statistically significant, indicating that time plays a significant role in the data. Among all the models, the quadratic and the log-linear models have the best R^2, which might indicate that they are the best models to fit the data.

h.

```
#Calculating the AIC and the BIC of the models
AIC(y1,y2,y3,y4,y5)
```

```
##    df       AIC
## y1  3 8393.2655
## y2  4 7325.0133
## y3  3 -768.6082
## y4  3 7745.7949
## y5  3 8375.3592
```

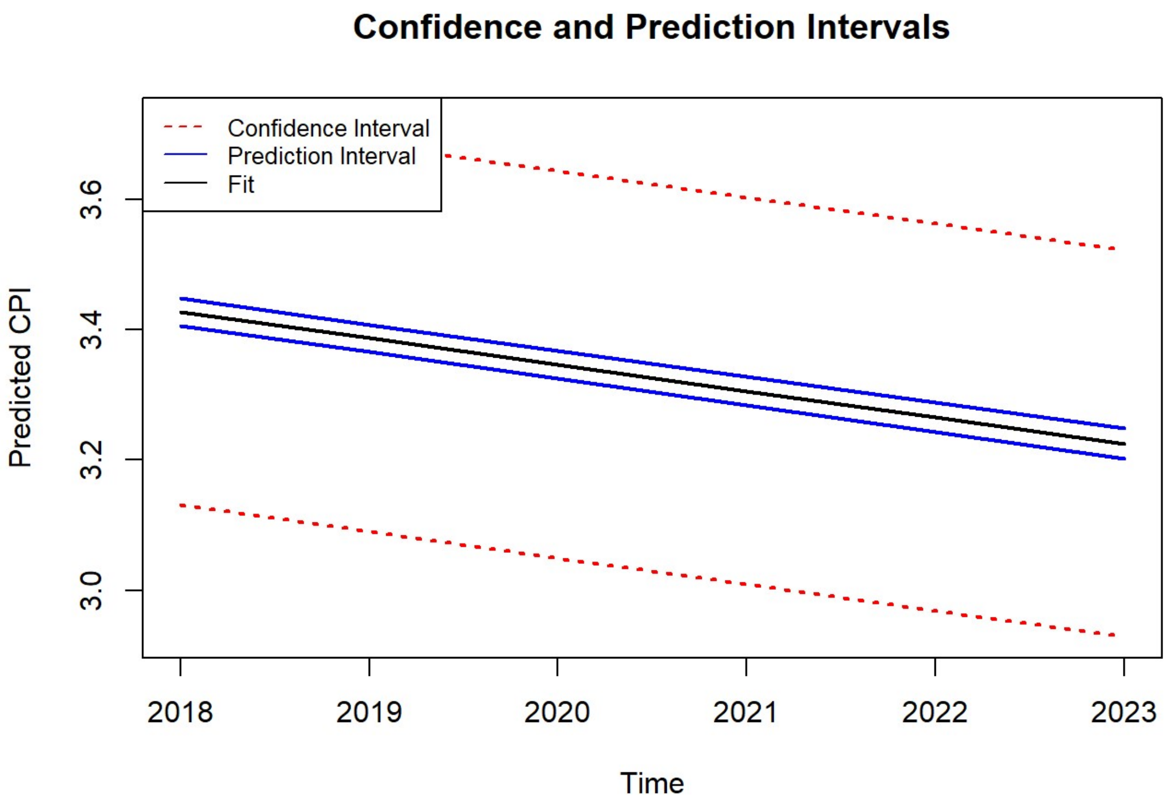
```
BIC(y1,y2,y3,y4,y5)
```

```
##    df       BIC
## y1  3 8407.3824
## y2  4 7343.8358
## y3  3 -754.4912
## y4  3 7759.9118
## y5  3 8389.4761
```

We can see that the log linear model is the best model to work with as the AIC and the BIC values are the smallest for the log linear model.

i.

```
#Plotting the fitted values, Confidence and Prediction intervals for 60 periods ahe
ad, i.e predicting for the years 2018-2023.
time <- data.frame(t = seq(2018,2023))
pred <- predict(lm(CPI~t), time, se.fit = TRUE)
pred_interval <- predict(lm(log(CPI)~t), time, level = .95, interval = "predictio
n")
conf_interval <- predict(lm(log(CPI)~t), time, level = .95, interval = "confidenc
e")

matplot(time$t, cbind(conf_interval, pred_interval[,-1]), lty = c(1,1,1,3,3),col =
c("black", "blue", "blue", "red", "red"), type = "l", lwd = 2, ylab = "Predicted CP
I", xlab = "Time", main = "Confidence and Prediction Intervals")
legend("topleft", legend = c("Confidence Interval", "Prediction Interval", "Fit"),
col = c("red", "blue", "black"), lty = c(2,1,1), cex = .8)
```



We forecast the data using the our log-linear model for 60 periods ahead or for 5 years ahead from 2018-2023. We can see that the over time, the purchasing power of the US dollar will continue to fall slightly. This makes sense as it had been falling slightly until the present time.

Problem 2

a.

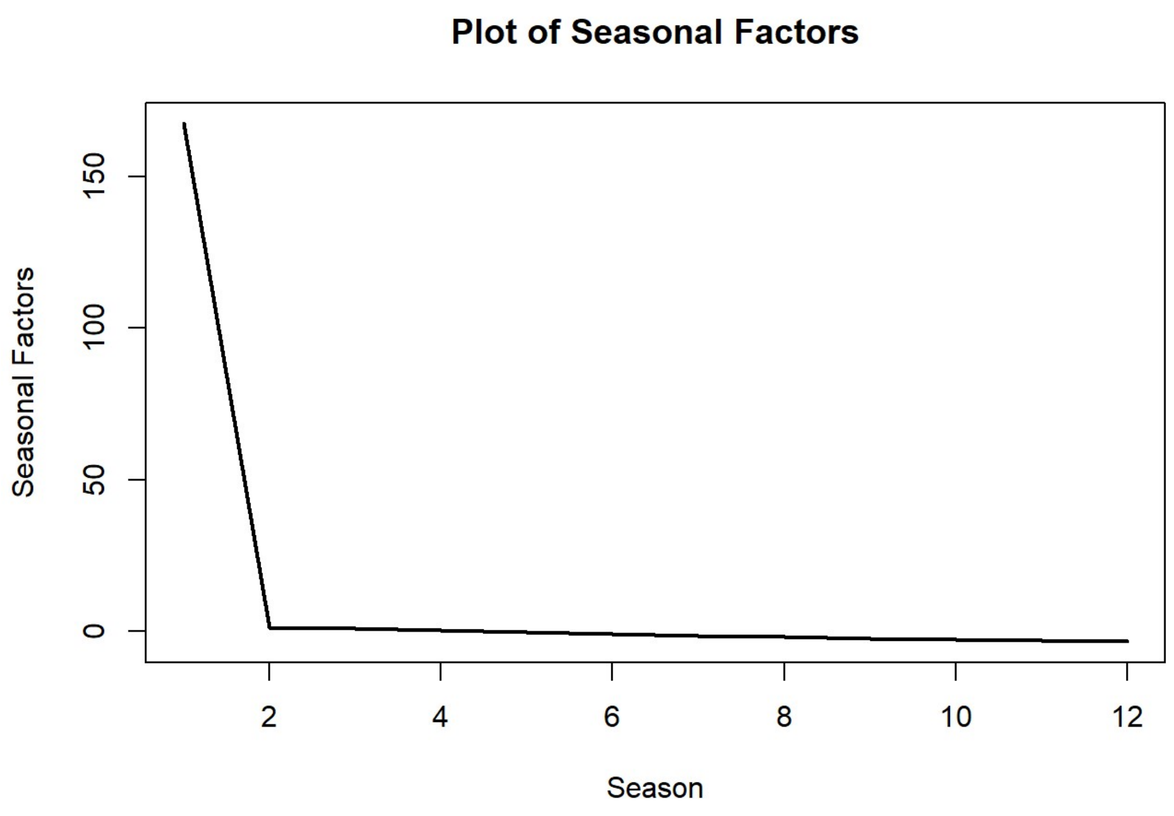
```
#Run a time series regression on the season and show the results
fit_season=tslm(CPI_ts ~ season)
summary(fit_season)
```

```
##
## Call:
## tslm(formula = CPI_ts ~ season)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -127.88 -109.09  -69.32  135.96  256.86
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 167.5406    15.2061  11.018   <2e-16 ***
## season2       1.4418    21.5835   0.067    0.947
## season3       0.8609    21.5835   0.040    0.968
## season4       0.3594    21.5835   0.017    0.987
## season5      -0.1479    21.5835  -0.007    0.995
## season6      -0.7906    21.5835  -0.037    0.971
## season7      -1.3847    21.5835  -0.064    0.949
## season8      -1.7744    21.5835  -0.082    0.934
## season9      -2.3009    21.5835  -0.107    0.915
## season10     -2.7288    21.5835  -0.126    0.899
## season11     -3.0332    21.5835  -0.141    0.888
## season12     -3.3215    21.5835  -0.154    0.878
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 126.3 on 805 degrees of freedom
## Multiple R-squared:  0.000148,   Adjusted R-squared:  -0.01351
## F-statistic: 0.01083 on 11 and 805 DF,  p-value: 1
```

All the season dummies’ coefficients are statistically insignificant, as evidenced by the high p-values for both the t- and F-statistics. This shows that seasonality weighs little on the data.

b.

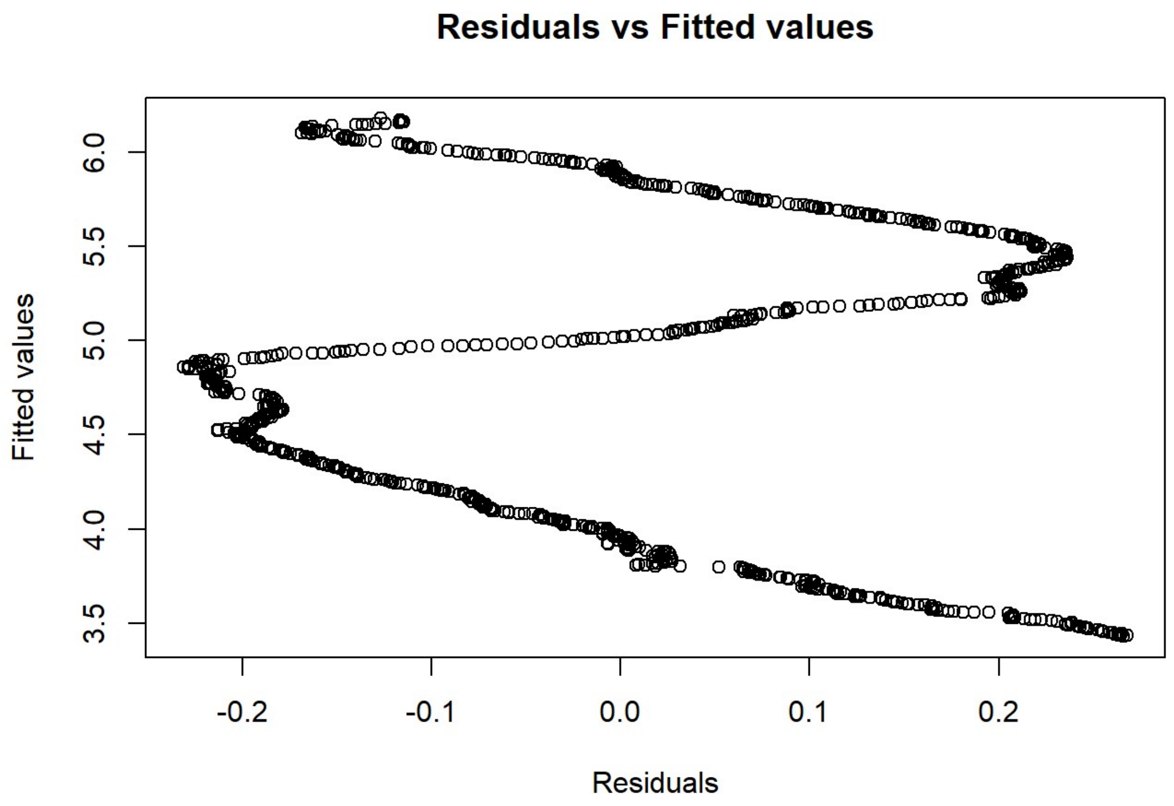
```
#Plotting the season factors
plot(fit_season$coef,type='l',ylab='Seasonal Factors', xlab="Season",lwd=2, main="P
lot of Seasonal Factors")
```



The plot of seasonal factors show that there is little significance of seasonal factors on the data. That further strengthens the idea that the trend dominates the seasonal elements.

c.

```
#First construct the time series regression model for both the trend and season
#Then plot the residuals vs fitted values graphs
fit_full = tslm(log(CPI_ts) ~ trend + season)
plot(fit_full$residuals,fit_full$fitted.values, main = "Residuals vs Fitted value
s", xlab = "Residuals", ylab = "Fitted values")
```



There seems to be a nonlinear pattern in the plot, which means that the model is heterskedastic and thus we have not accounted for all the dynamics in the data. But at the very least the number of residuals on the right seems to balance off with the number on the left, which means that model predicts the data quite well.

d.

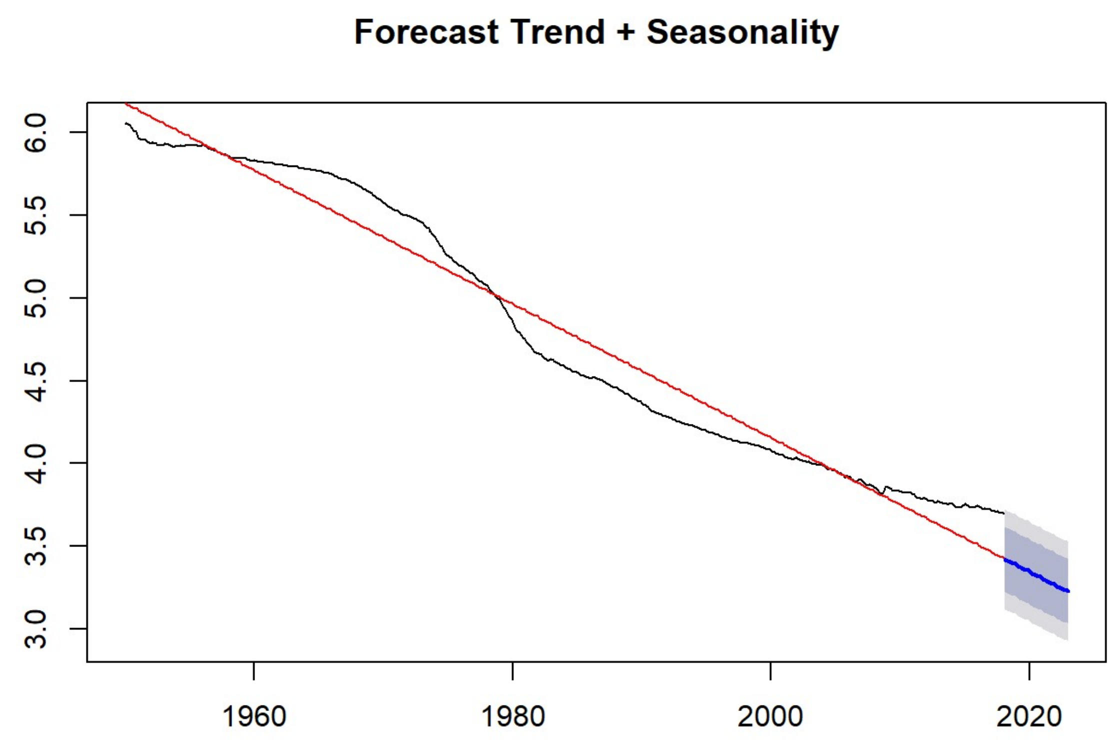
```
#Producing the summary of the regression
summary(fit_full)
```

```
##
## Call:
## tslm(formula = log(CPI_ts) ~ trend + season)
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.231069 -0.145971  0.000437  0.131494  0.267819
##
## Coefficients:
##               Estimate Std. Error  t value Pr(>|t|)
## (Intercept)  6.181e+00  2.047e-02  302.001   <2e-16 ***
## trend       -3.366e-03  2.252e-05 -149.473   <2e-16 ***
## season2     -4.025e-03  2.594e-02   -0.155    0.877
## season3     -4.887e-03  2.594e-02   -0.188    0.851
## season4     -5.013e-03  2.594e-02   -0.193    0.847
## season5     -4.931e-03  2.594e-02   -0.190    0.849
## season6     -5.269e-03  2.594e-02   -0.203    0.839
## season7     -4.698e-03  2.594e-02   -0.181    0.856
## season8     -4.048e-03  2.594e-02   -0.156    0.876
## season9     -4.011e-03  2.594e-02   -0.155    0.877
## season10    -2.819e-03  2.594e-02   -0.109    0.914
## season11    -5.633e-04  2.594e-02   -0.022    0.983
## season12     2.033e-03  2.594e-02    0.078    0.938
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1518 on 804 degrees of freedom
## Multiple R-squared:  0.9653, Adjusted R-squared:  0.9647
## F-statistic:  1862 on 12 and 804 DF,  p-value: < 2.2e-16
```

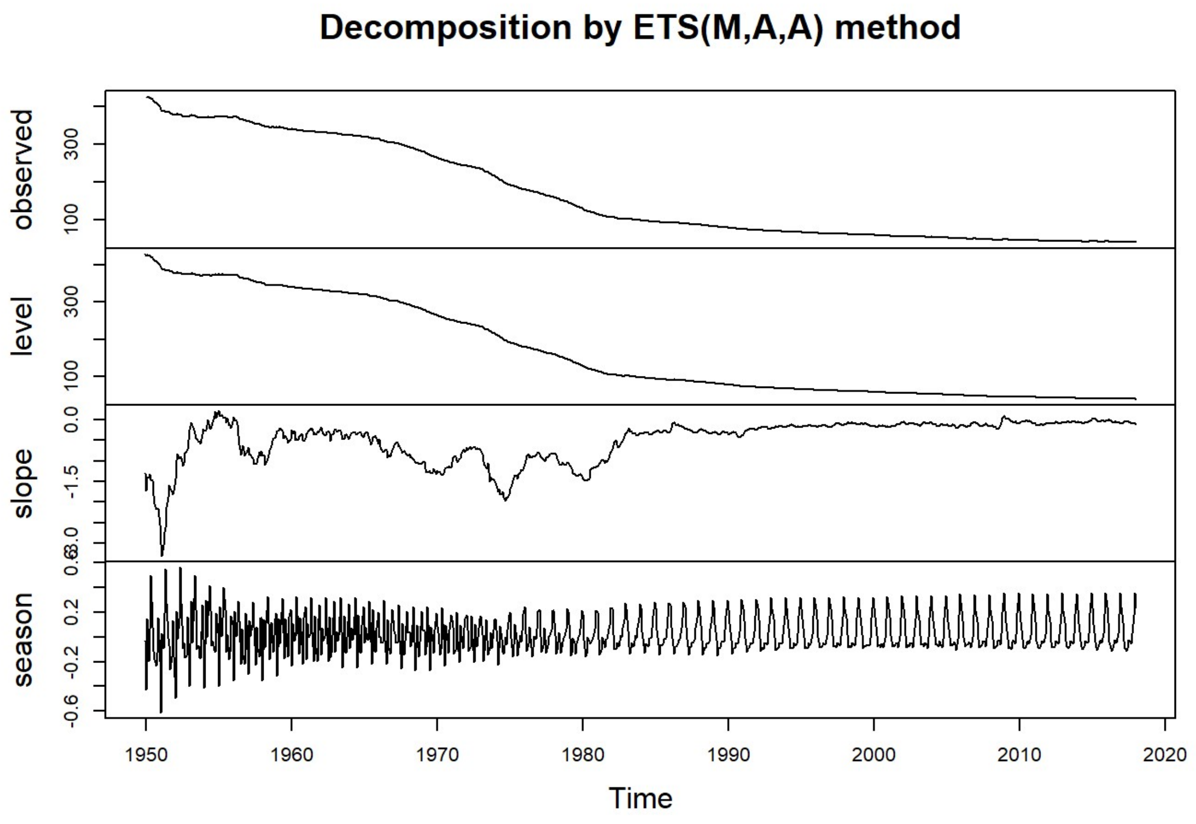
Only the trend's coefficient is statistically significant, while the seasons' coefficients are all statistically insignificant based on the t-values and it's corresponding probabilities. This suggests that there is little seasonality in the data as trend weighs more on the data than seasonality. Since the R^2 is very high, we can see that the model fits the data well. The large F-statistics, paired with a significantly low p-value, shows that all the regression coefficients are not equal zero. But this is because of the trend coefficient having large t-value and a corresponding low p-value that shows it is a statistically significant variable not equal zero.

e.

```
#Plotting the forecast of the data using the full model for 60 periods ahead
plot(forecast(fit_full,h=60),main="Forecast Trend + Seasonality")
lines(fit_full$fitted.values, col="red")
```



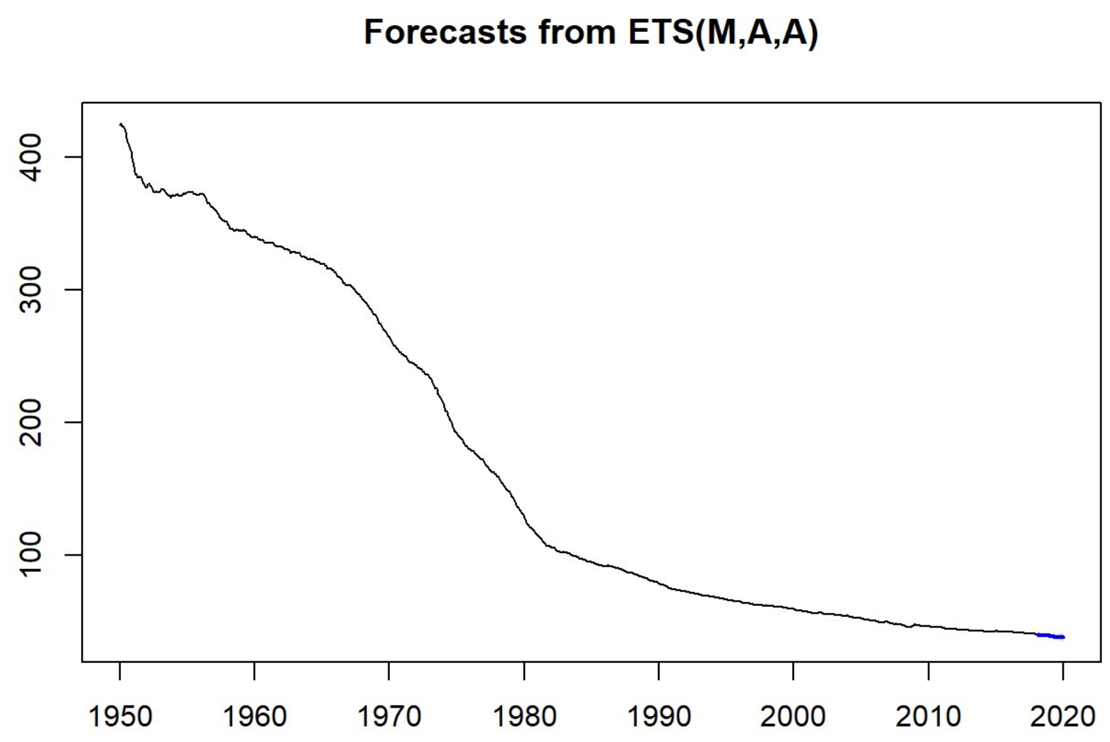
```
#Plotting the ets of the CPI_ts and finding the accuracy
fit_ets = ets(CPI_ts)
plot(fit_ets)
```



```
accuracy(fit_ets)
```

```
##                      ME      RMSE       MAE        MPE      MAPE
## Training set 0.01096641 0.5838411 0.3389124 0.01042081 0.2056646
##                    MASE      ACF1
## Training set 0.05994035 0.2198285
```

```
plot(forecast(fit_ets,level=c(50,80,95)))
```



We forecast the data using the our full model for 60 periods ahead or for 5 years ahead from 2018-2023. As we can see, just plotting the forecast of the full model yields very bad result. However, by using the ETS function, we manage to improve the forecast and produce the graphs that shows a better picture of the forecast. Both forecasts show that the CPI is trending downwards.

Conclusion

After testing with several models, we think that the log-linear model seems to fit the data best. However, looking at the residuals vs fitted values plot, the model is still heteroskedastic.

Forecasting both the log-linear model and the log-linear+seasonality yields very similar results, which means that seasonality seems to play very little role on the data.

Moving forward, we would have to add more dynamics to the log-linear model, such as incorporating ARMA to our model, to better improve the fit of the data.

References/Citation

U.S. Bureau of Labor Statistics, Consumer Price Index for All Urban Consumers: Purchasing Power of the Consumer Dollar [CUUR0000SA0R], retrieved from FRED, Federal Reserve Bank of St. Louis; https://fred.stlouisfed.org/series/CUUR0000SA0R (https://fred.stlouisfed.org/series/CUUR0000SA0R), April 25, 2018.