STAT 443: Lab 1

Alexander Grinius (20712972)

21st March, 2025

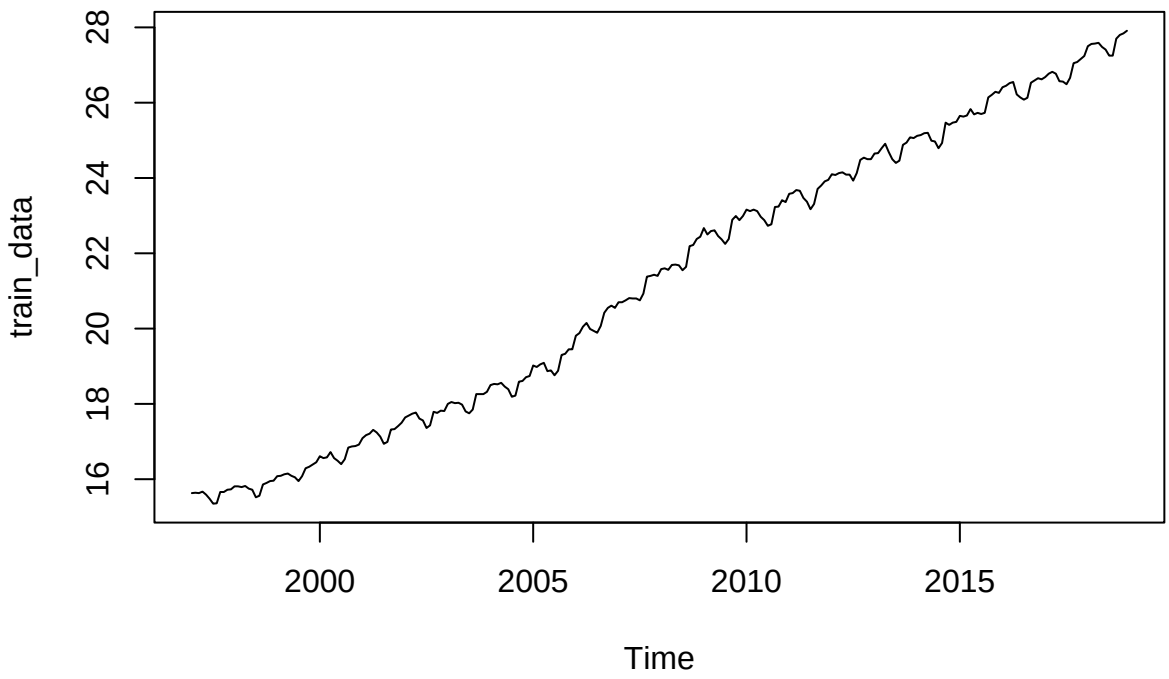
Question 1

(a)

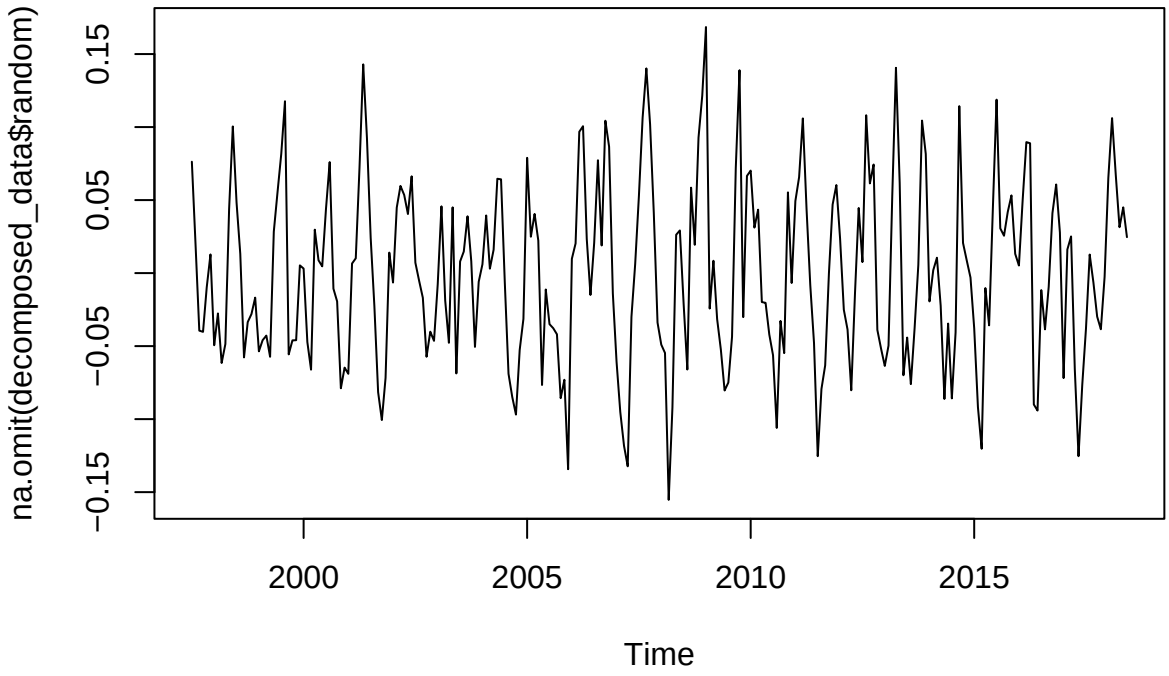
```
wages_ts <- ts(data$employee_wages, start = c(1997, 1), frequency = 12)
train_data <- window(wages_ts, start = c(1997, 1), end = c(2018, 12))
test_data <- window(wages_ts, start = c(2019, 1), end = c(2019, 12))
```

(b)

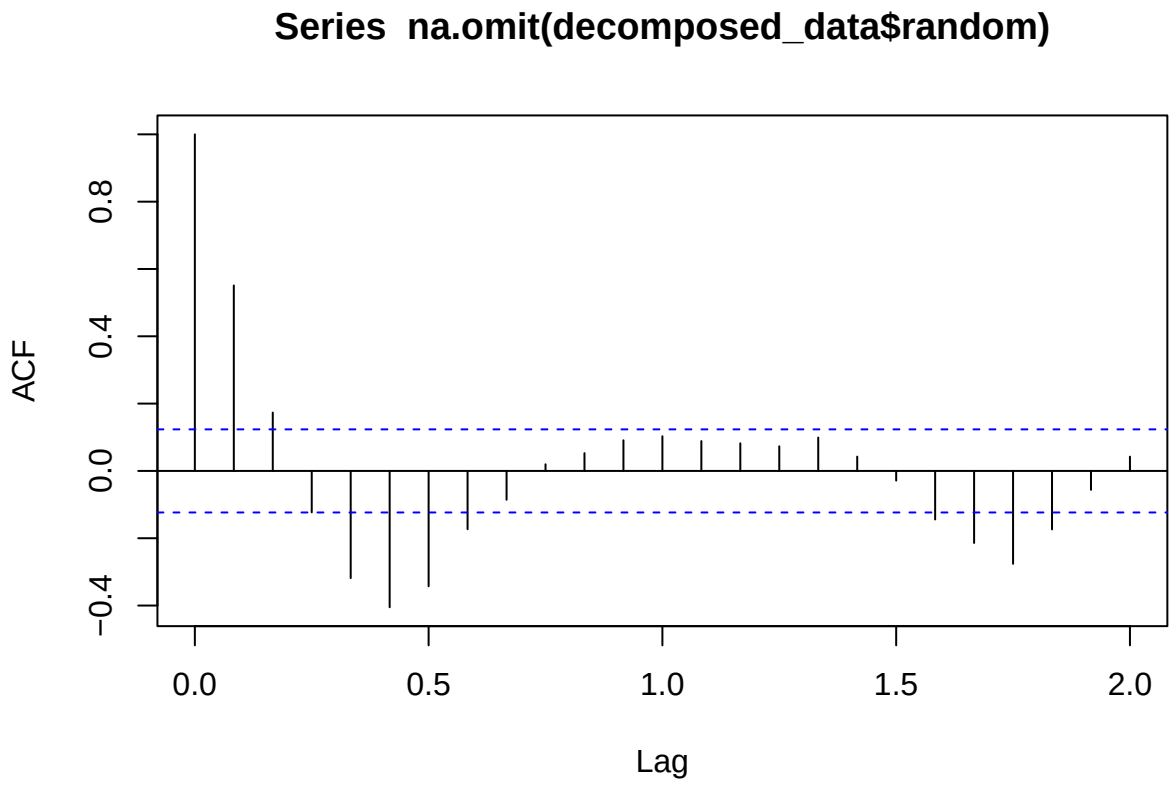
```
plot(train_data)
```



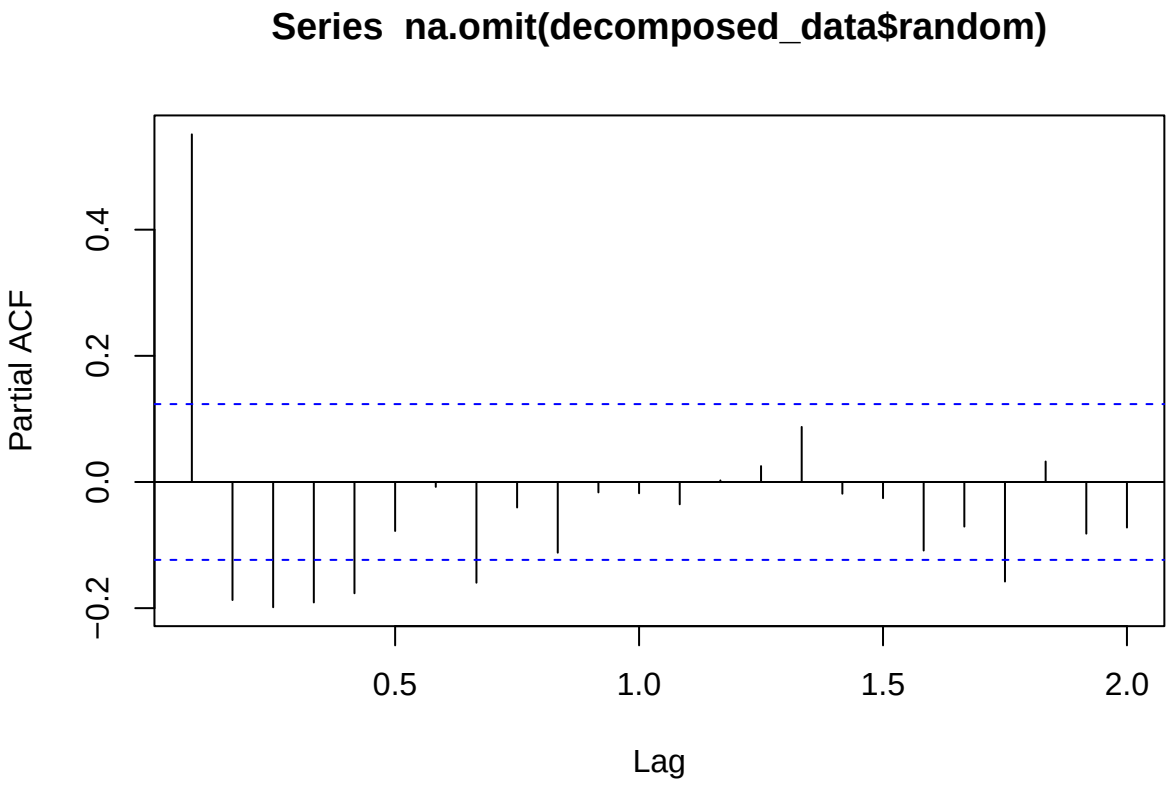
```r
decomposed_data <- decompose(train_data, type = "additive")
plot(na.omit(decomposed_data$random))
```




```r
acf(na.omit(decomposed_data$random))
```




```
pacf(na.omit(decomposed_data$random))
```




```
adf.test(na.omit(decomposed_data$random))
```

```
## Warning in adf.test(na.omit(decomposed_data$random)): p-value smaller than
## printed p-value
```

```
##
##  Augmented Dickey-Fuller Test
##
## data:  na.omit(decomposed_data$random)
## Dickey-Fuller = -7.8021, Lag order = 6, p-value = 0.01
## alternative hypothesis: stationary
```

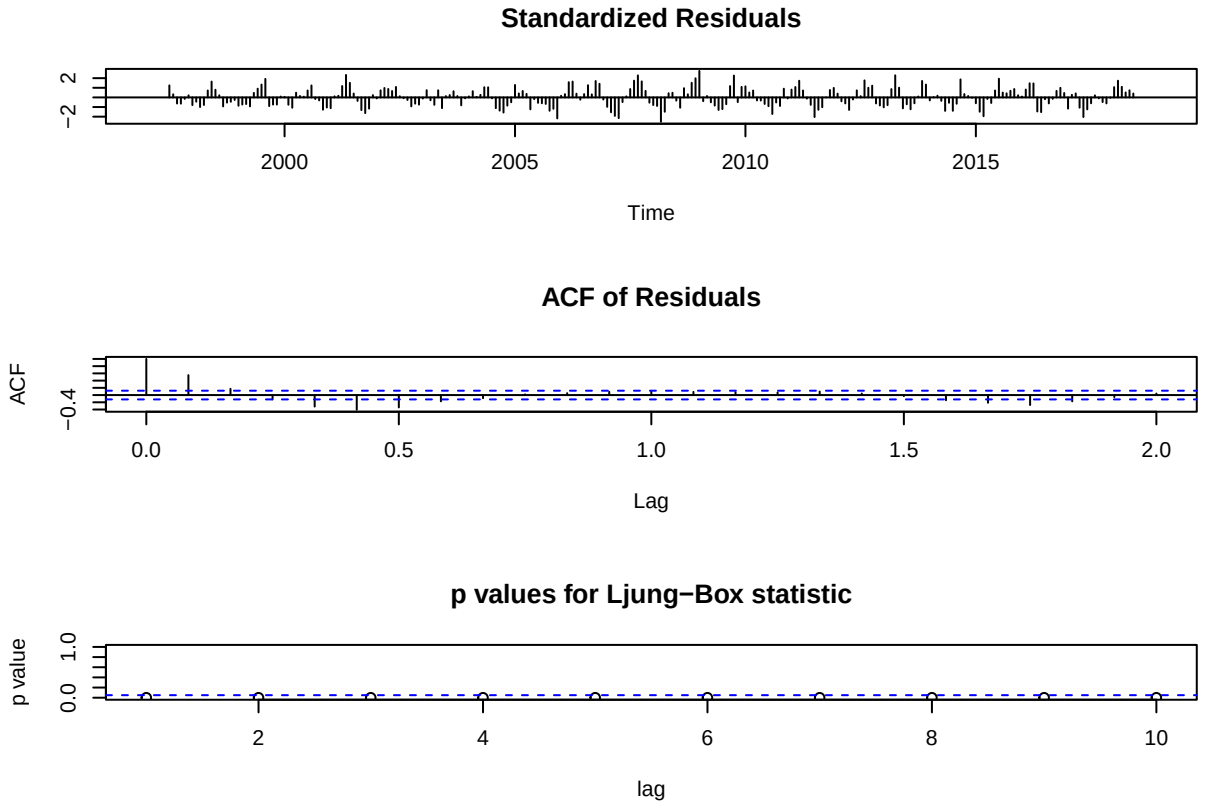
From the plot of the original data, we can see that an additive model would be most appropriate. From the ACF we can see that we want q = 7/12 months, the pacf gives p = 5/12 months. (c)

```
arma_model <- arima(decomposed_data$random, order = c(5/12, 0, 7/12))
summary(arma_model)
```

```
##
## Call:
## arima(x = decomposed_data$random, order = c(5/12, 0, 7/12))
##
## Coefficients:
##       intercept
```

```
##         -0.0004
## s.e.     0.0039
##
## sigma^2 estimated as 0.003767:  log likelihood = 345.7,  aic = -687.39
##
## Training set error measures:
##                          ME       RMSE        MAE      MPE     MAPE     MASE
## Training set -3.430887e-18 0.06137539 0.05049225 99.45379 100.0935 1.101395
##                    ACF1
## Training set 0.5510938
```

```r
tsdiag(arma_model)
```

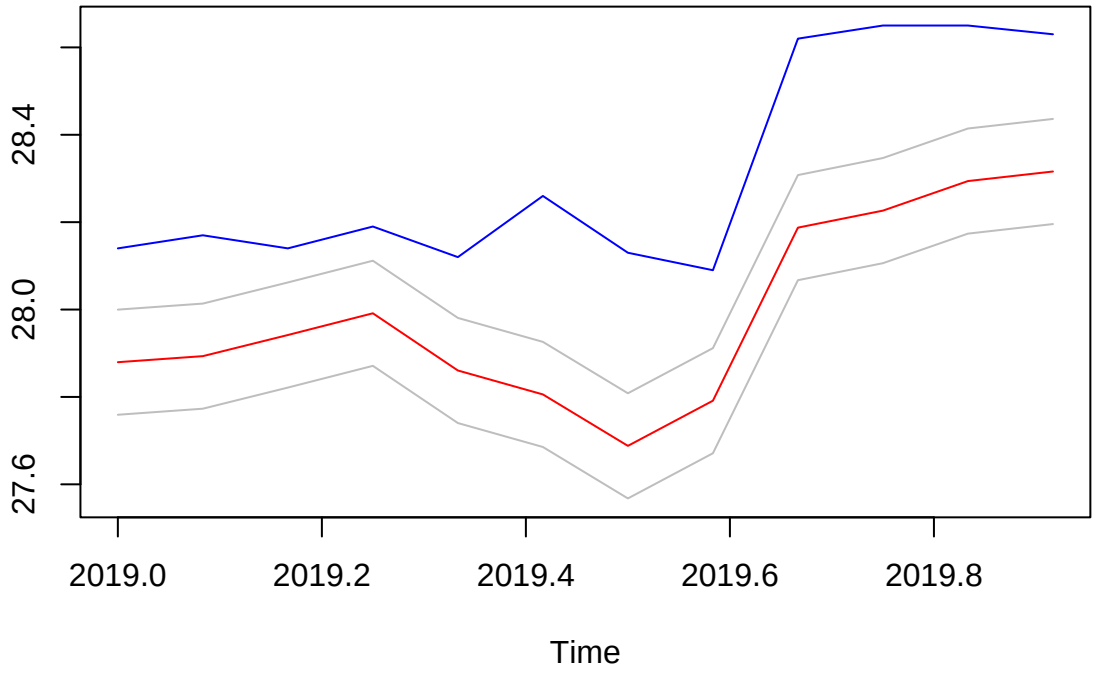


The graph and ACF plot of the standard residuals aren't particularly satisfactory; there appears to be some form of sinusodal pattern to the graphs of the residuals, and the acf has multiple significant lags before 6/12 months (0.5 on the above graph).

```r
remainder_forecast <- predict(arma_model, n.ahead=12)
trend <- predict.lm(
  lm(Trend ~ Time, data=data.frame(
    Trend = na.omit(decomposed_data$trend),
    Time = 1:252
    )
    ),
  newdata = data.frame(Time = 253:264)
)
```

```r
seasonal_component <- decomposed_data$seasonal[1:12]
forecasts_q1 <- remainder_forecast$pred + seasonal_component + trend
```

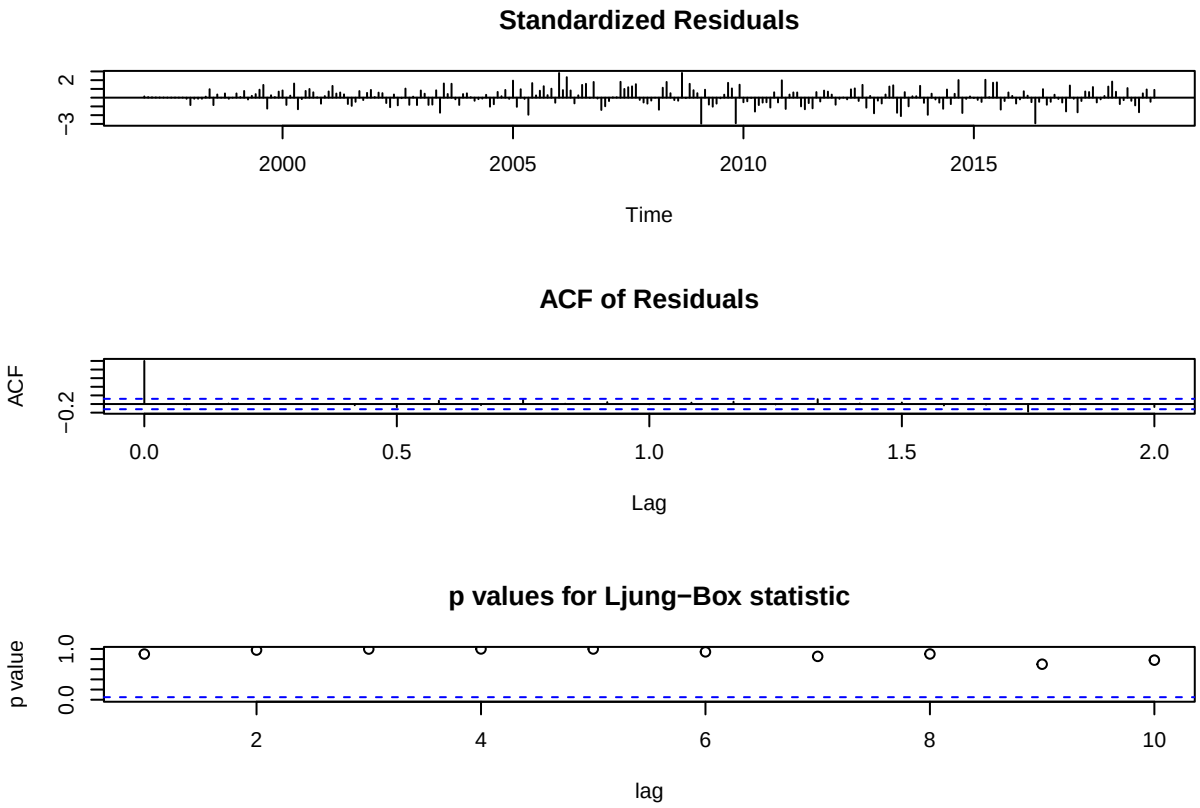
```r
comparison <- data.frame(
  Month = 1:12,
  Forecast = forecasts_q1,
  True_Value = test_data,
  Lower_Bound = forecasts_q1 - 1.96* remainder_forecast$se,
  Upper_Bound = forecasts_q1 + 1.96* remainder_forecast$se
)
#print(comparison)
ts.plot(comparison$True_Value, comparison$Forecast, comparison$Lower_Bound, comparison$Upper_Bound, col
```



The prediction seems alright, I wasn't sure how to calculate the standard error of this form of model. ### Question 2

(a)

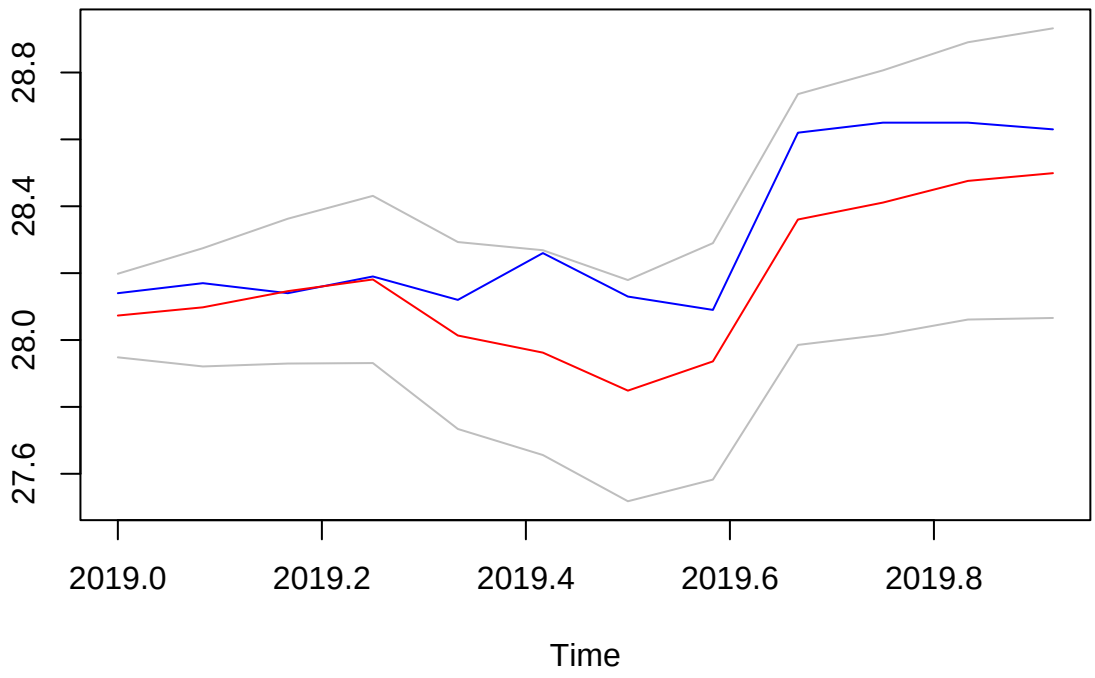
```r
Q2_model <- auto.arima(train_data)
tsdiag(Q2_model)
```



This SARIMA model appears to have a much better fit; The residual graph has less defined patterns, and the acf doesn't have any lag values with large probability.

(b)

```
forcasts_q2 <- predict(Q2_model, n.ahead = 12)
comparison <- data.frame(
  Month = 1:12,
  Forecast = forcasts_q2$pred,
  True_Value = test_data,
  Lower_Bound = forcasts_q2$pred - 1.96 * forcasts_q2$se,
  Upper_Bound = forcasts_q2$pred + 1.96 * forcasts_q2$se
)
ts.plot(comparison$True_Value, comparison$Forecast, comparison$Lower_Bound, comparison$Upper_Bound, col
```

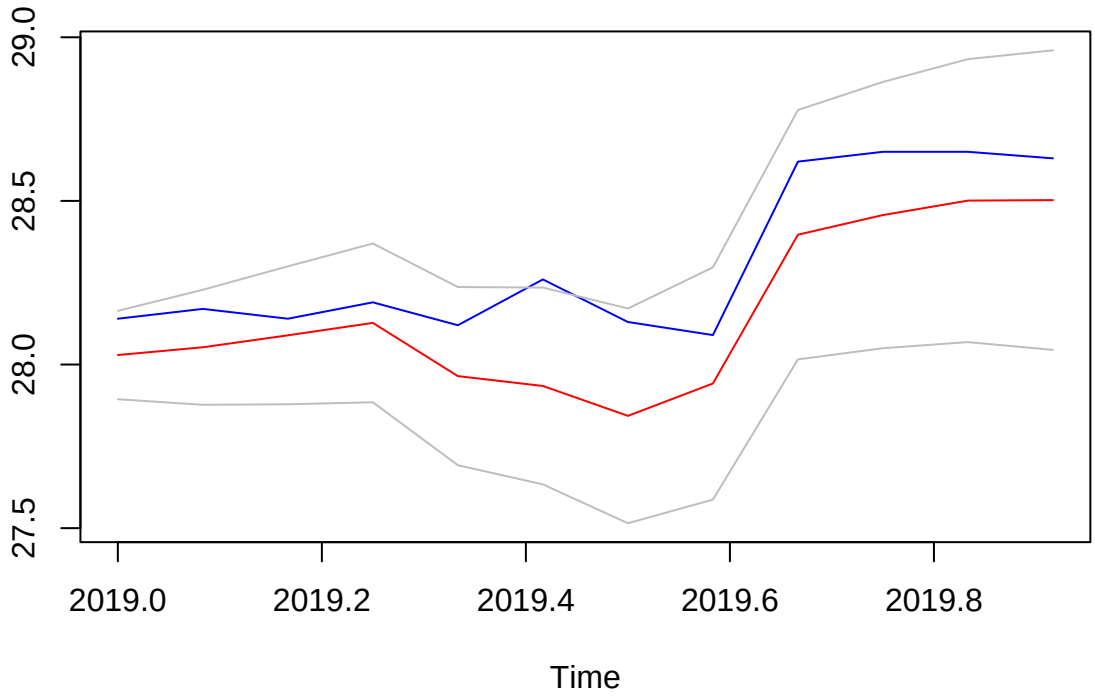


This model appears to be much closer to the real values, and the 95% confidence interval contains all the actual values.

Question 3

(a)

```
holt_winters_model <- HoltWinters(train_data,seasonal = "additive")
forcasts_q3 <- predict(holt_winters_model, n.ahead = 12, prediction.interval = TRUE)
comparison <- data.frame(
  Month = 1:12,
  Forecast = forcasts_q3[,1],
  True_Value = test_data,
  Lower_Bound = forcasts_q3[,2],
  Upper_Bound = forcasts_q3[,3]
)
ts.plot(comparison$True_Value, comparison$Forecast, comparison$Lower_Bound, comparison$Upper_Bound, col
```



This model appears to be closer to the true values than model 1, but further than model 2. The 95% ci also doesn't fully contain the true values. ### Question 4

(a)

```
cat(" SnDecomp + ARMA:", mean((forecasts_q1 - test_data)^2), "\n","Box-Jenkins    :", mean((forcasts_q2
```

```
##  SnDecomp + ARMA: 0.1144138
##  Box-Jenkins    : 0.03205542
##  Holt-Winters   : 0.03271917
```

The SARIMA model with Box-Jenkins forecasting has the lowest MSPE.

Method 1:

- Pros: able to interpret the decomposed data more precisely (what is seasonal vs. what is the trend); fast to create
- Cons: Not as accurate as the other models; code is more involved

Method 2:

- Pros: the most accurate for predictions (MSPE); code is easy to implement
- Cons: takes a longer time computationally expensive; harder to interpret

Method 3:

- Pros: Reasonably accurate predictions, only slightly worse than the best method and much more than the worst; code is also easy to implement; interpretable when you dig into the values (separates trend & seasonal components)
- Cons: The ci for the predictions was too narrow;