HW1

Read files

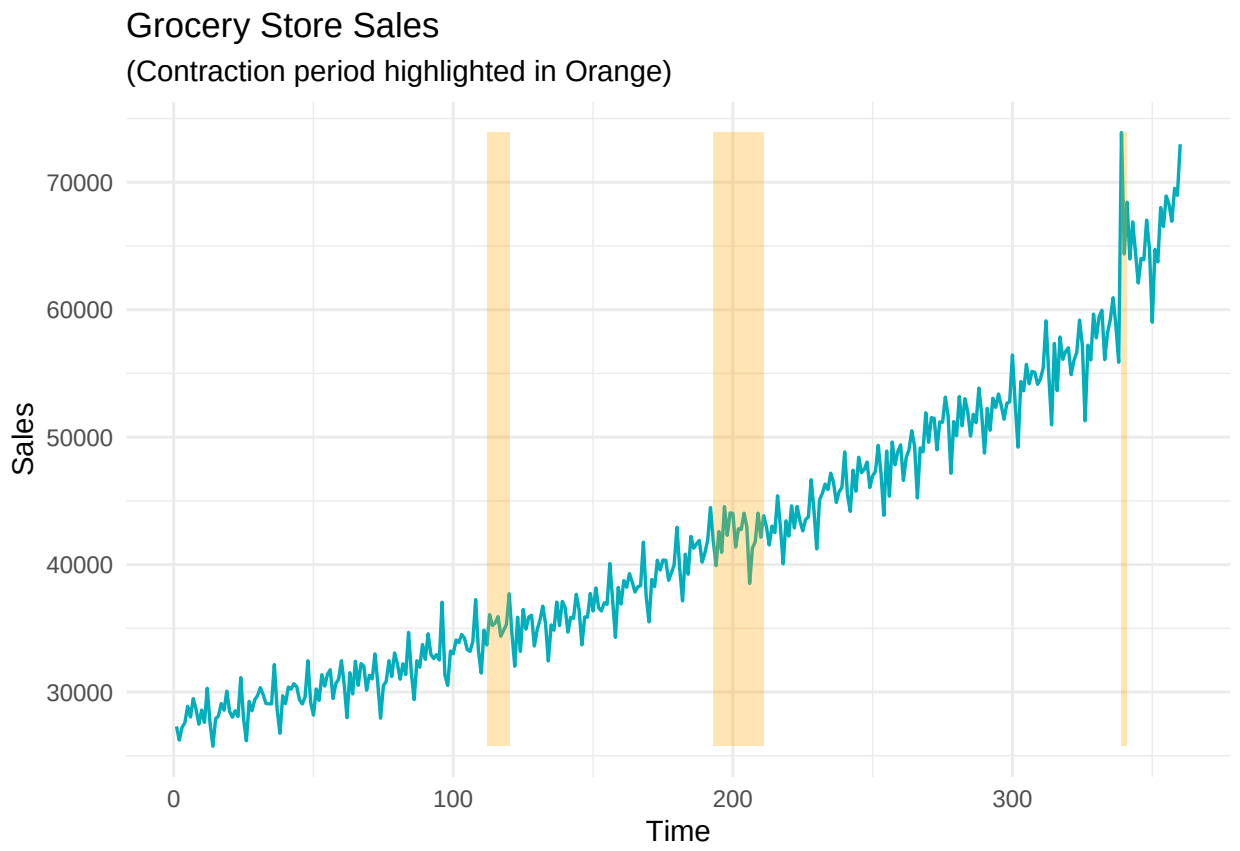
```
sales<-read.csv(file = "C:\\Users\\phili\\Desktop\\STAT 535\\STAT 535 HW1\\GroceryStoreSales.txt")

sales$Time<-as.numeric(sales$Time)
fMonth<-as.factor(sales$Month)

sales<-data.frame(sales,fMonth)
```

Q1

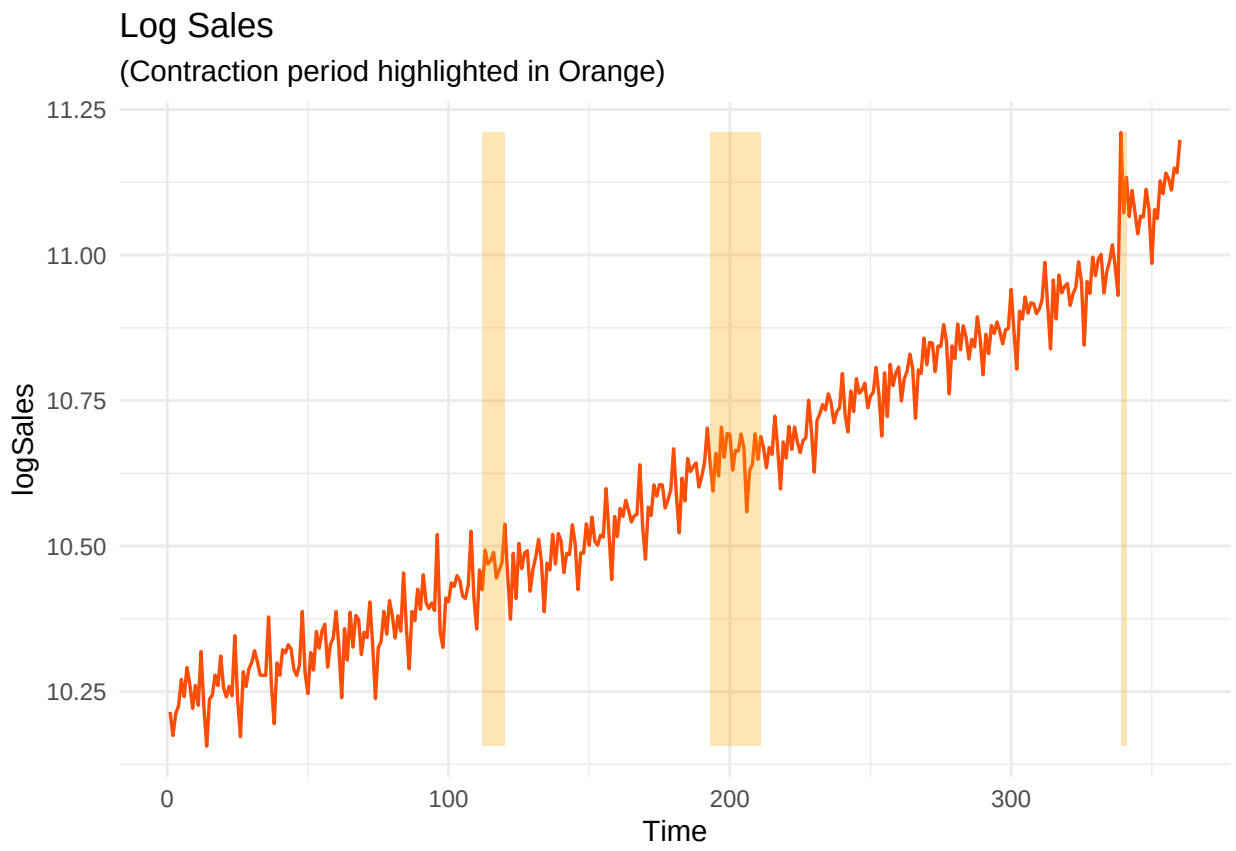
During the years 1992 to 2021 the Business Cycle Dating Committee of the National Bureau of Economic Research defined three periods of economic contraction, 2001(4) to 2001(11), 2008(1) to 2009(6), and 2020(3) to 2020(4).

1. Make separate time series plots for (i) Sales and (ii) log Sales, and mark the contraction periods on the plots.

```
## Example data
set.seed(0)
dat <- sales

## Determine highlighted regions
v <- rep(0, max(dat$Time))
v[c(112:119, 193:210, 339:340)] <- 1

## Get the start and end points for highlighted regions
inds <- diff(c(0, v))
start <- dat$Time[inds == 1]
end <- dat$Time[inds == -1]
if (length(start) > length(end)) end <- c(end, tail(dat$Time, 1))

## highlight region data
rects <- data.frame(start=start, end=end, group=seq_along(start))

ggplot(data=dat, aes(Time, Sales)) +
theme_minimal() +
geom_line(color = "#00AFBB", size = .6) +
geom_rect(data=rects, inherit.aes=FALSE, aes(xmin=start, xmax=end, ymin=min(dat$Sales),
ymax=max(dat$Sales), group=group), color="transparent", fill="orange",
alpha=0.3) +
labs(title = "Grocery Store Sales", subtitle = "(Contraction period highlighted in Orange)")
```




```
ggplot(data=dat, aes(Time, logSales)) +
theme_minimal() +
geom_line(color = "#FC4E07", size = .6) +
geom_rect(data=rects, inherit.aes=FALSE, aes(xmin=start, xmax=end, ymin=min(dat$logSales),
ymax=max(dat$logSales), group=group), color="transparent", fill="orange", alpha=0.3) +
labs(title = "Log Sales", subtitle = "(Contraction period highlighted in Orange)")
```



2. Discuss and compare the two plots. Comment on trend structure and volatility. Do the plots reveal any unusual features?

ANSWER: Upward trend is visible. In the Sales plot (blue line) Sales seem to increase exponentially or by a constant multiplicative factor, whereas in the log Sales plot (red line) log Sales seem to increase by a constant additive amount.

There is annual seasonal component. Grocery sales tend to peak in December and drop sharply in January and rises to a small peak in the summer months before peaking again in December. The cycle seems to repeat annually.

Volatility seems to increase with the number of Sales, and economic contraction does not seem to affect grocery Sales.

There is an outlier in March 2020 (observation 339), where grocery sales skyrocket due to the start of the COVID-19 pandemic.

3. If yes, describe what is notable and discuss the underlying causes. Do the two plots indicate whether an additive decomposition model or a multiplicative decomposition model should be fit to model Sales? Explain your answer.

ANSWER: The most obvious underlying cause of the peak in December Sales is Christmas holiday shopping - consumers buy more groceries to do Christmas cooking. The outlying sharp sales peak in March 2020 is due to the start of COVID-19 pandemic which led Americans to stockpile groceries.

The multiplicative decomposition model should be fit to the model as we do see evidence of increasing volatility as Sales go up.

Q2

1. Fit a multiplicative decomposition model to the Sales data. Include a polynomial trend, a seasonal component using the fMonth variable, and trigonometric variables to investigate calendar structure. If you find an outlier value which warrants use of a dummy for adjustment, form the dummy and include it in your model.) Comment on and interpret the statistical results.

ANSWER:

The trend and seasonal components are statistically significant. The partial F-test for both components yield very small p-values which means we reject the null hypothesis that their regression coefficents are simultaneously zero (see below for F-test and p-values).

The outlier variable and calendar variables are also statistically significant. We can see through the summary function that the outlier variable (obs339) also has a very small p-value. For calendar vaiables, the partial F-test also yield a very small value which means that calendar pairs are statistically significant, so we retain them in our analysis.

```
# add dummy variable for outlier, observation at t = 339
obs339<-rep(0, max(sales$Time))
obs339[339] <- 1
sales<-data.frame(sales,obs339)

model1<-lm(logSales~Time+I(Time^2)+I(Time^3)+I(Time^4)+fMonth+obs339+c348+s348+c432+s432, data = sales)
summary(model1)
```

```
##
## Call:
## lm(formula = logSales ~ Time + I(Time^2) + I(Time^3) + I(Time^4) +
##     fMonth + obs339 + c348 + s348 + c432 + s432, data = sales)
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.051396 -0.010798  0.000094  0.010042  0.061814
##
## Coefficients:
##               Estimate Std. Error  t value Pr(>|t|)
## (Intercept)  1.023e+01  5.499e-03 1860.380  < 2e-16 ***
## Time         6.266e-04  1.778e-04    3.524 0.000483 ***
## I(Time^2)    1.654e-05  1.998e-06    8.276 2.96e-15 ***
## I(Time^3)   -6.952e-08  8.311e-09   -8.365 1.59e-15 ***
## I(Time^4)    1.044e-10  1.142e-11    9.141  < 2e-16 ***
## fMonth2     -7.120e-02  4.478e-03  -15.900  < 2e-16 ***
## fMonth3      1.591e-02  4.513e-03    3.526 0.000480 ***
## fMonth4     -7.582e-03  4.476e-03   -1.694 0.091186 .
## fMonth5      4.729e-02  4.477e-03   10.562  < 2e-16 ***
## fMonth6      1.509e-02  4.476e-03    3.371 0.000836 ***
## fMonth7      4.327e-02  4.478e-03    9.663  < 2e-16 ***
## fMonth8      3.033e-02  4.477e-03    6.776 5.49e-11 ***
## fMonth9     -1.025e-02  4.479e-03   -2.288 0.022730 *
## fMonth10     1.032e-02  4.477e-03    2.306 0.021697 *
## fMonth11     1.114e-02  4.480e-03    2.486 0.013382 *
## fMonth12     7.305e-02  4.479e-03   16.309  < 2e-16 ***
## obs339       1.892e-01  1.787e-02   10.586  < 2e-16 ***
## c348        -9.583e-03  1.298e-03   -7.384 1.20e-12 ***
## s348         5.489e-05  1.292e-03    0.042 0.966133
```

```
## c432        -7.943e-04  1.298e-03   -0.612 0.540962
## s432         3.425e-03  1.293e-03    2.648 0.008470 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.01733 on 339 degrees of freedom
## Multiple R-squared:  0.9956, Adjusted R-squared:  0.9953
## F-statistic:  3800 on 20 and 339 DF,  p-value: < 2.2e-16
```

Partial F-test to determine the significance of fMonth dummy variables

```
# excluding the time polynomials
model2<-lm(logSales~fMonth+obs339+c348+s348+c432+s432, data = sales)

anova(model2, model1)
```

```
## Analysis of Variance Table
##
## Model 1: logSales ~ fMonth + obs339 + c348 + s348 + c432 + s432
## Model 2: logSales ~ Time + I(Time^2) + I(Time^3) + I(Time^4) + fMonth +
##     obs339 + c348 + s348 + c432 + s432
##   Res.Df     RSS Df Sum of Sq     F    Pr(>F)
## 1    343 21.9642
## 2    339  0.1018  4    21.862 18205 < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Partial F-test to determine the significance of fMonth dummy variables

```
# excluding the fMonth dummy variable
model3<-lm(logSales~Time+I(Time^2)+I(Time^3)+I(Time^4)+obs339+c348+s348+
c432+s432, data = sales)

anova(model3, model1)
```

```
## Analysis of Variance Table
##
## Model 1: logSales ~ Time + I(Time^2) + I(Time^3) + I(Time^4) + obs339 +
##     c348 + s348 + c432 + s432
## Model 2: logSales ~ Time + I(Time^2) + I(Time^3) + I(Time^4) + fMonth +
##     obs339 + c348 + s348 + c432 + s432
##   Res.Df     RSS Df Sum of Sq      F    Pr(>F)
## 1    350 0.52798
## 2    339 0.10177 11    0.4262 129.06 < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Partial F-test to determine the significance of calendar pair variables

```
# excluding the calendar pairs
model4<-lm(logSales~Time+I(Time^2)+I(Time^3)+I(Time^4)+fMonth+obs339, data = sales);

anova(model4, model1)
```

```
## Analysis of Variance Table
##
## Model 1: logSales ~ Time + I(Time^2) + I(Time^3) + I(Time^4) + fMonth +
##     obs339
```

```
## Model 2: logSales ~ Time + I(Time^2) + I(Time^3) + I(Time^4) + fMonth +
##     obs339 + c348 + s348 + c432 + s432
##   Res.Df     RSS Df Sum of Sq      F    Pr(>F)
## 1    343 0.12036
## 2    339 0.10177  4  0.018583 15.475 1.226e-11 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

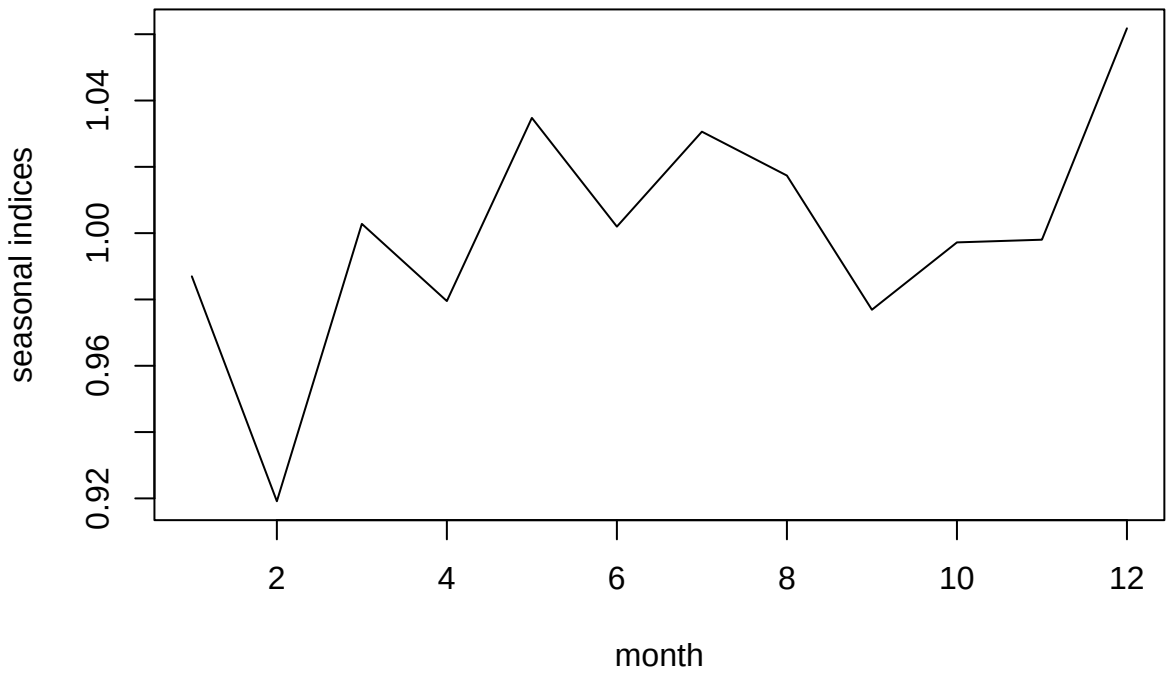
(a) Tabulate and plot the estimated static seasonal indices and give a detailed interpretation of them in the context of the data collection.

ANSWER: The output of code below produces log the seasonal indices, so we would need to exponentiate them to obtain the seasonal indices. The final indices will give us the percentage change of Sales from month to month.

For example, looking at final indices, since February has seasonal index of .918 we can expect sales to fall 8.2 percent below the level of the trend. In December, we can expect sales to increase 6.12 above the level of trend as it has an index of 1.0612.

More generally, the plot indicates that annually grocery store sales increase in December and decrease in January and February before rising again to reach a small peak in the summer months. The percentage factors by which Sales increase/decrease are listed below.

```
b1<-coef(model1)[1]
b2<-coef(model1)[6:16]+b1
b3<-c(b1,b2)
seas<-b3-mean(b3)

seas.ts<-ts(exp(seas))
plot(seas.ts,ylab="seasonal indices",xlab="month")
```




```
#cbind(seas,exp(seas))

month <- seq(12)
seas_indices <- exp(seas)
seas_df <- data.frame(month, seas, seas_indices)
print.data.frame(tbl_df(seas_df))
```

```
## Warning: `tbl_df()` was deprecated in dplyr 1.0.0.
## Please use `tibble::as_tibble()` instead.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was generated.
```

```
##    month         seas seas_indices
## 1      1 -0.013114856    0.9869708
## 2      2 -0.084313971    0.9191426
## 3      3  0.002799790    1.0028037
## 4      4 -0.020696905    0.9795158
## 5      5  0.034175159    1.0347658
## 6      6  0.001973431    1.0019754
## 7      7  0.030150337    1.0306095
## 8      8  0.017220121    1.0173692
## 9      9 -0.023364598    0.9769062
## 10    10 -0.002790114    0.9972138
## 11    11 -0.001974507    0.9980274
## 12    12  0.059936113    1.0617687
```

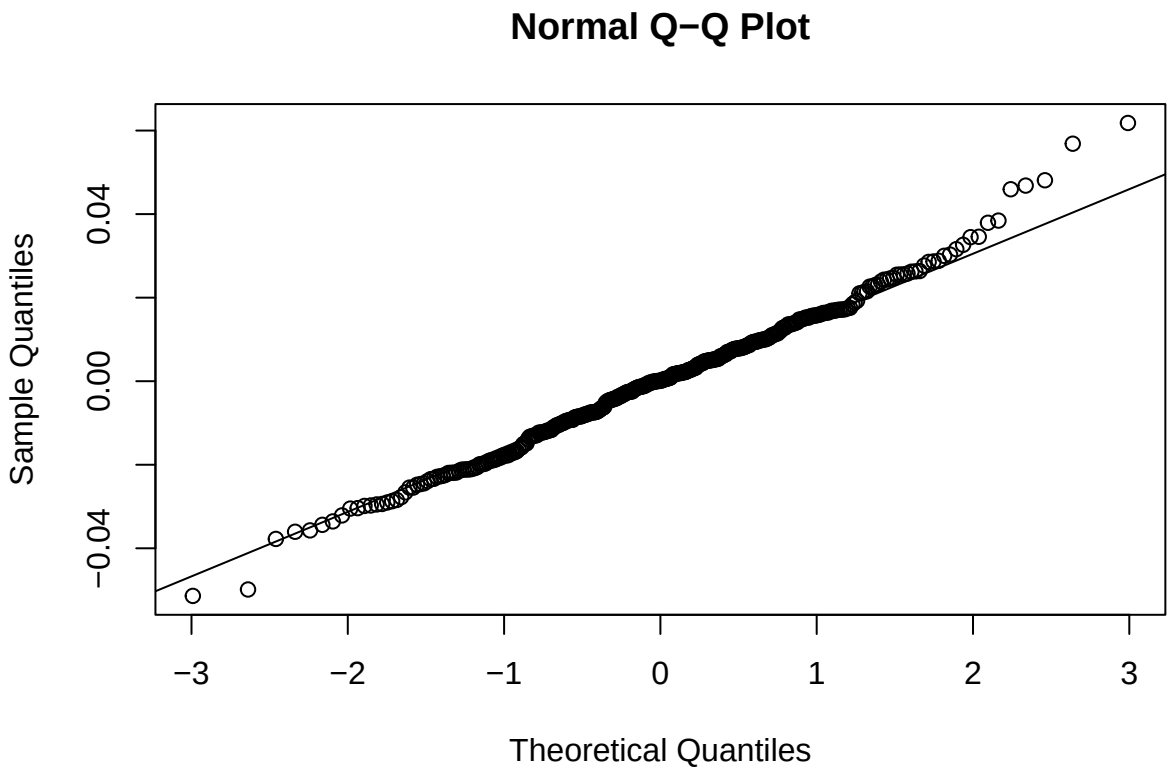
(b)

(i)Save the residuals from the fit.

```
res1 <- resid(model1)
```

(ii)Form a normal quantile plot of these residuals

```
qqnorm(res1)
qqline(res1)
```



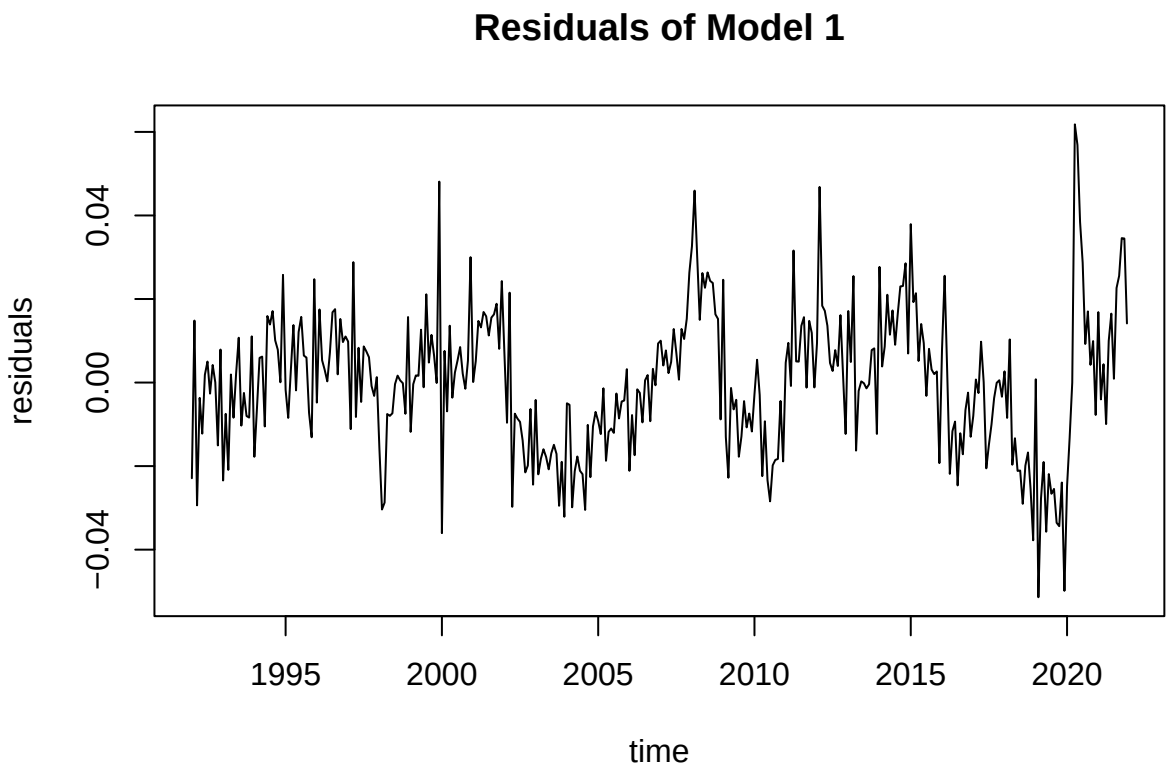
(iii) test the residuals for normality

```
shapiro.test(res1)
```

```
##
##  Shapiro-Wilk normality test
##
## data:  res1
## W = 0.99292, p-value = 0.08761
```

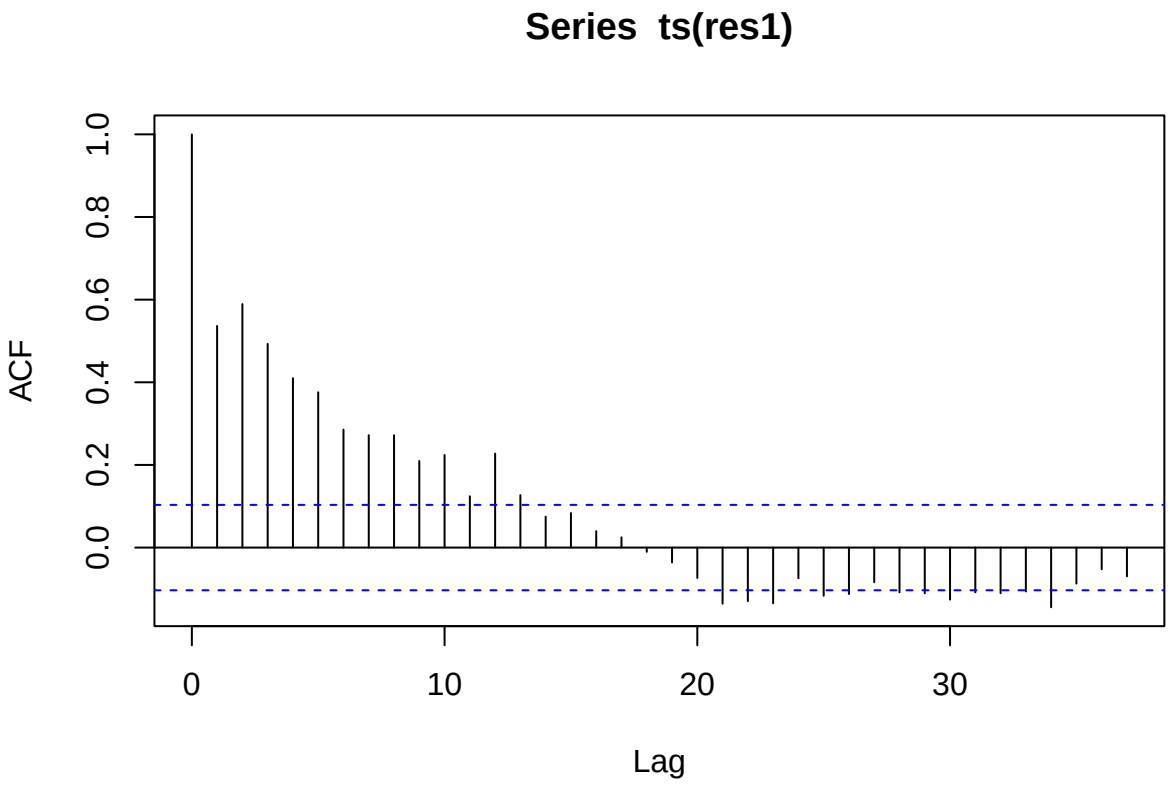
(iv) plot the residuals vs. time

```
resid1 <- ts(res1,start=c(1992,1),freq=12)
plot(resid1, xlab="time",ylab="residuals",main="Residuals of Model 1")
```



(v) and plot their autocorrelations.

```
acf(ts(res1), 37)
```



(vi) Describe each of these results. What conclusions do you draw from the residual analysis?

ANSWER: - QQPLOT: The QQPLOT seems to substantiate the assumption that the residuals are normally distributed. Most of the quantiles (observations) fall on the qqline. There are patterns of departure from normality with a few points at the upper and lower tails of the plot but not significant enough to the conclude that the residuals are not normally distributed.

- Shapiro-Wilk Normality Test: The p-value is 0.08761, which is not small enough to reject the null hypothesis that the residuals are normally distributed (given a 95% confidence level, the p-value has to fall below 0.05 to conclude that the residuals are not normal).
- Autocorrelation: The autocorrelation function plot indicates strong correlation between the residuals at lags 1-12, which indicates that the model does not adequately capture trend.
- Residual Analysis: The residuals show that the model does not track autocorrelation structure and the trend structure adequately. The residuals have a slight upward trend, which means that our model did not capture such upward trend adequately. If a model captures trend, the plot of residuals should be relatively flat.

The residuals also seem to be volatile, which indicate that there maybe autocorrelation structure which the model does not capture.

(vii)In particular, what structures in the time series has the model failed to capture?

The residuals show that the model does not track the trend structure adequately. The residuals have a slight upward trend, which means that our model did not capture such upward trend adequately. If a model captures trend, the plot of residuals should be relatively flat.

Q3

3. Use the decompose function in R with a multiplicative formulation to construct seasonal index estimates for the Sales data. Compare these static seasonal index estimates to those from part 2(a) using (i) a table and (ii) a plot. Discuss what this reveals.

ANSWER: The seasonal indices and decompose seasonal indices are nearly identical. This means that our model captures the seasonality component well.

```
sales.ts<-ts(sales$Sales,freq=12)
sales.decmpsm<-decompose(sales.ts,type="mult")
seasdmult1<-sales.decmpsm$seasonal
seasdmult<-seasdmult1[1:12]/prod(seasdmult1[1:12])^(1/12)

seasdmult
```

```
##  [1] 0.9875176 0.9182998 1.0102505 0.9792911 1.0345200 1.0008969 1.0305684
##  [8] 1.0164224 0.9756709 0.9961827 0.9963713 1.0611748
```

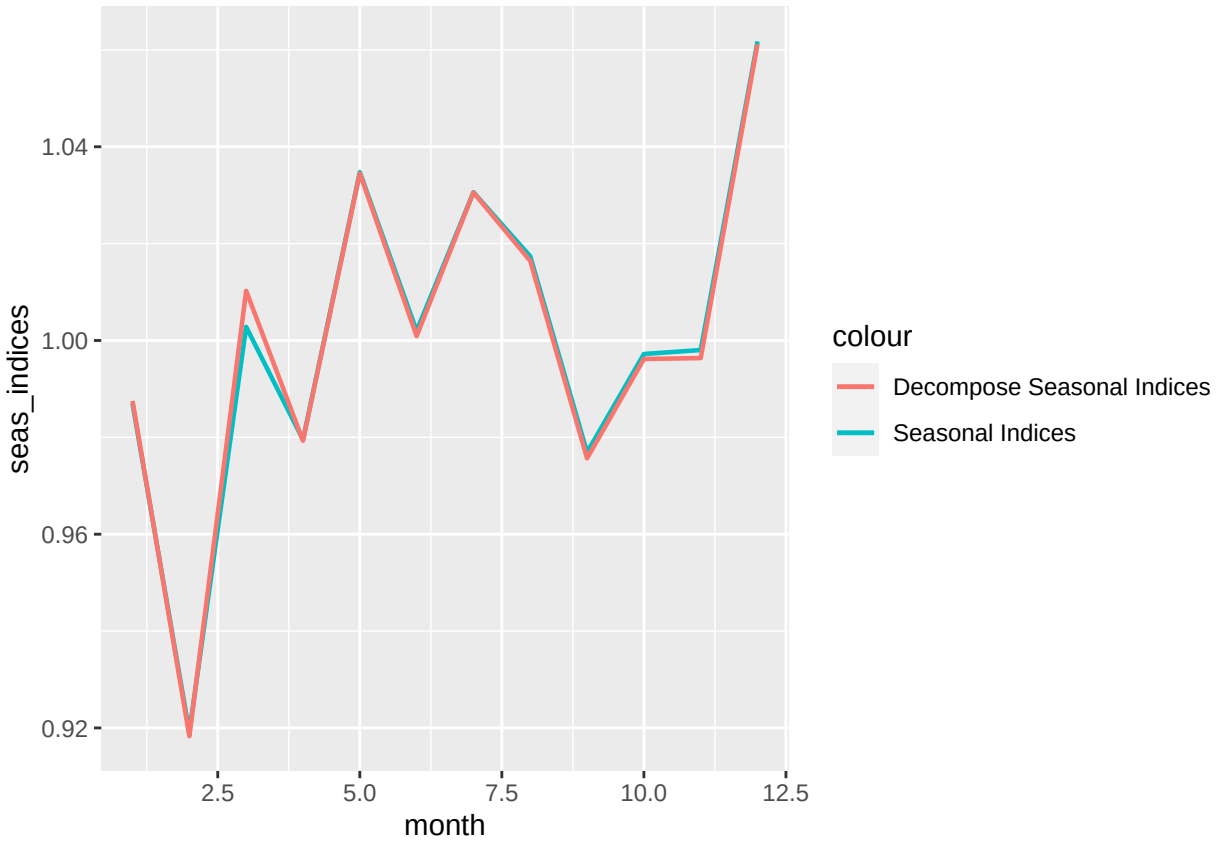
(i) Table comparison of seasonal indices and decompose seasonal indices

```
month <- seq(12)
seas_indices <- exp(seas)
seas_df <- data.frame(month, seas_indices, seasdmult)
print.data.frame(seas_df)
```

```
##             month seas_indices seasdmult
## (Intercept)     1    0.9869708 0.9875176
## fMonth2         2    0.9191426 0.9182998
## fMonth3         3    1.0028037 1.0102505
## fMonth4         4    0.9795158 0.9792911
## fMonth5         5    1.0347658 1.0345200
## fMonth6         6    1.0019754 1.0008969
## fMonth7         7    1.0306095 1.0305684
## fMonth8         8    1.0173692 1.0164224
## fMonth9         9    0.9769062 0.9756709
## fMonth10       10    0.9972138 0.9961827
## fMonth11       11    0.9980274 0.9963713
## fMonth12       12    1.0617687 1.0611748
```

(ii) Plot comparison of seasonal indices and decompose seasonal indices

```
ggplot(seas_df, aes(x = month)) +
  geom_line(aes(y = seas_indices, color = 'Seasonal Indices'), size = .8) +
  geom_line(aes(y = seasdmult, color = 'Decompose Seasonal Indices'), size =.8)
```



Q4

Calculate the lag 1, lag 2, and lag 3 residuals from the model in part 2 and add these three new variables to the model you fit in part 2. Be sure to include the calendar trigonometric pairs if they turn out to be significant now. Then perform a residual analysis for this new model. What improvements do you notice?

ANSWER: The residuals plot is now flat and less volatile, which means that the model is now closer to yielding white noise. The autocorrelation function plot now also indicates that most lags are between the blue lines.

This indicates that the adjustment of including lag variables takes into account the autocorrelation structure.

```
lresid<-c(rep(NA,360))
lag1resid<-lresid
lag2resid<-lresid
lag3resid<-lresid

lag1resid[2]<-resid(model1)[1]
lag1resid[3]<-resid(model1)[2]
lag2resid[3]<-resid(model1)[1]

for(i in 4:360){
i1<-i-1
i2<-i-2
i3<-i-3
lag1resid[i]<-resid(model1)[i1]
```

```
lag2resid[i]<-resid(model1)[i2]
lag3resid[i]<-resid(model1)[i3]
}
```

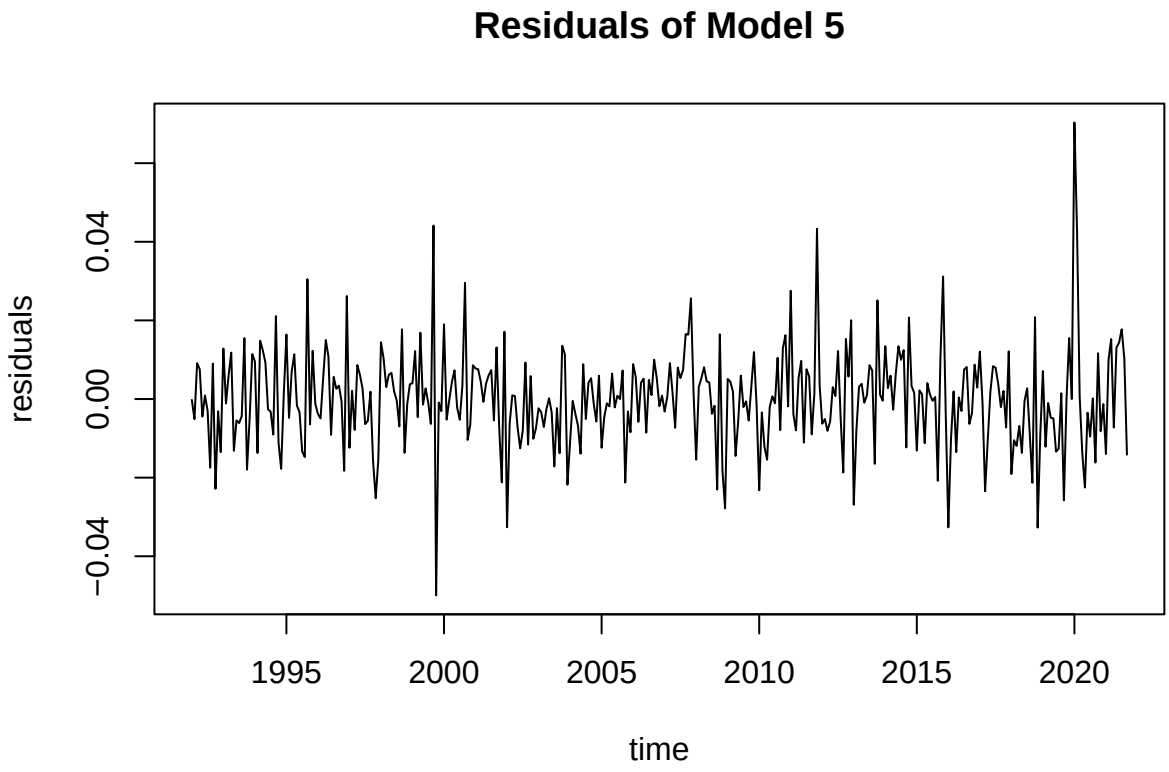
```
sales<-data.frame(sales,lag1resid,lag2resid,lag3resid)
```

```
model5<-lm(logSales~Time+I(Time^2)+I(Time^3)+I(Time^4)+fMonth+obs339+
             c348+s348+c432+s432+
             lag1resid+lag2resid+lag3resid, data = sales)

summary(model5)
```

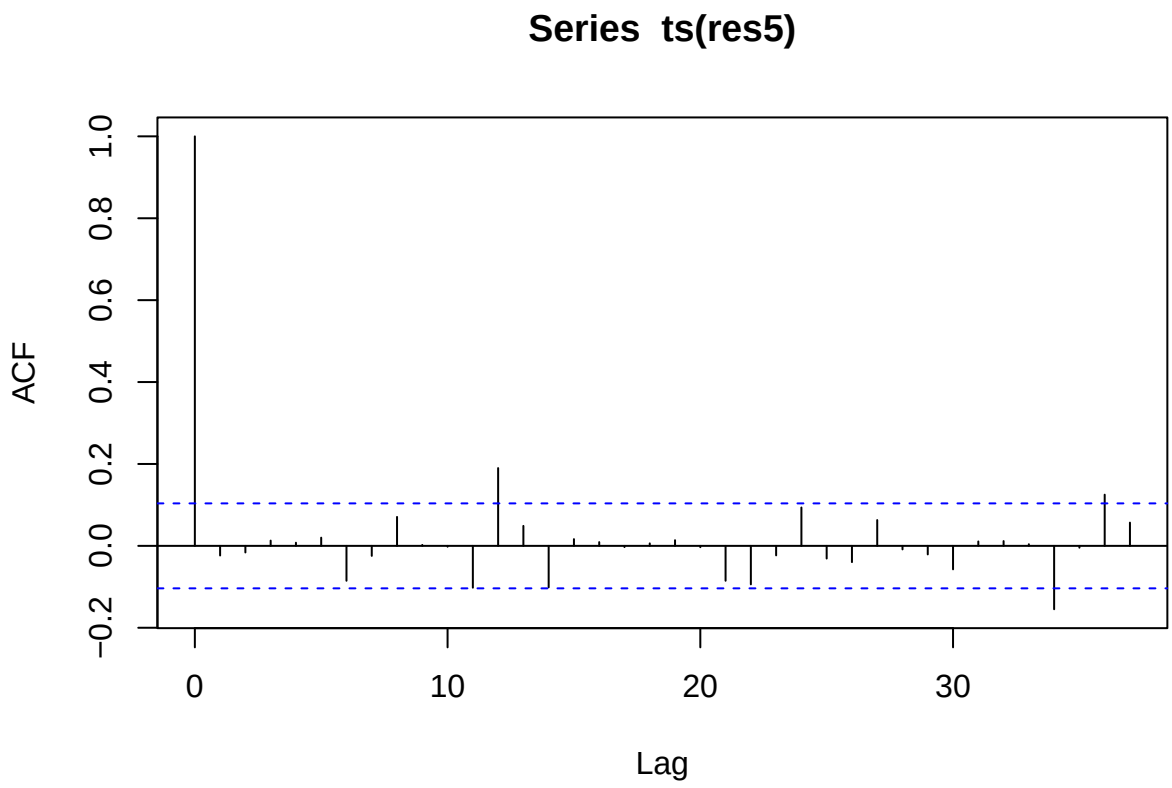
```
##
## Call:
## lm(formula = logSales ~ Time + I(Time^2) + I(Time^3) + I(Time^4) +
##     fMonth + obs339 + c348 + s348 + c432 + s432 + lag1resid +
##     lag2resid + lag3resid, data = sales)
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.049957 -0.006856 -0.000232  0.007129  0.070343
##
## Coefficients:
##               Estimate Std. Error  t value Pr(>|t|)
## (Intercept)  1.023e+01  4.519e-03 2264.839  < 2e-16 ***
## Time         5.126e-04  1.433e-04    3.576 0.000401 ***
## I(Time^2)    1.780e-05  1.580e-06   11.264  < 2e-16 ***
## I(Time^3)   -7.474e-08  6.494e-09  -11.508  < 2e-16 ***
## I(Time^4)    1.116e-10  8.856e-12   12.601  < 2e-16 ***
## fMonth2     -7.310e-02  3.392e-03  -21.548  < 2e-16 ***
## fMonth3      1.458e-02  3.424e-03    4.259 2.67e-05 ***
## fMonth4     -9.226e-03  3.364e-03   -2.742 0.006431 **
## fMonth5      4.564e-02  3.365e-03   13.560  < 2e-16 ***
## fMonth6      1.345e-02  3.364e-03    3.999 7.84e-05 ***
## fMonth7      4.160e-02  3.365e-03   12.363  < 2e-16 ***
## fMonth8      2.871e-02  3.364e-03    8.534 5.07e-16 ***
## fMonth9     -1.190e-02  3.365e-03   -3.537 0.000461 ***
## fMonth10     8.677e-03  3.364e-03    2.579 0.010336 *
## fMonth11     9.510e-03  3.365e-03    2.826 0.004997 **
## fMonth12     7.138e-02  3.366e-03   21.208  < 2e-16 ***
## obs339       2.092e-01  1.352e-02   15.477  < 2e-16 ***
## c348        -9.617e-03  9.708e-04   -9.905  < 2e-16 ***
## s348         2.286e-04  9.666e-04    0.236 0.813193
## c432        -8.274e-04  9.704e-04   -0.853 0.394493
## s432         3.677e-03  9.685e-04    3.797 0.000174 ***
## lag1resid    2.510e-01  5.384e-02    4.663 4.53e-06 ***
## lag2resid    3.736e-01  5.119e-02    7.297 2.17e-12 ***
## lag3resid    1.686e-01  5.442e-02    3.099 0.002107 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.01291 on 333 degrees of freedom
##   (3 observations deleted due to missingness)
## Multiple R-squared:  0.9975, Adjusted R-squared:  0.9974
```

```
## F-statistic:  5832 on 23 and 333 DF,  p-value: < 2.2e-16
```

```
res5 <- resid(model5)
resid5 <- ts(res5,start=c(1992,1),freq=12)
plot(resid5, xlab="time",ylab="residuals",main="Residuals of Model 5")
```




```
acf(ts(res5), 37)
```



Q5

Perform the following analysis to investigate the presence of dynamic seasonality.

(a) Analyze the data spanning the years 1992 through 2007 and calculate the static seasonal estimates.

```
# split the Time
sales_i <- sales[sales$Time <= 192, ]

# fit the model
model_i<-lm(logSales~Time+I(Time^2)+I(Time^3)+I(Time^4)+fMonth+obs339+
              c348+s348+c432+s432, data = sales_i)

# find the seasonal indices
b1<-coef(model_i)[1]
b2<-coef(model_i)[6:16]+b1
b3<-c(b1,b2)
seas_i<-b3-mean(b3)

seas_i
```

```
##   (Intercept)       fMonth2       fMonth3       fMonth4       fMonth5
## -0.0205092240 -0.0869616745  0.0004772471 -0.0223475710  0.0321216761
##       fMonth6       fMonth7       fMonth8       fMonth9      fMonth10
##  0.0024670588  0.0348806814  0.0189179547 -0.0222580956 -0.0038903604
##      fMonth11      fMonth12
```

```
## -0.0042826042  0.0713849116
```

(b) Repeat part (a) for the years 2008 through 2019.

```
# split the Time
sales_ii <- sales[sales$Time > 192 & sales$Time <= 336, ]

# fit the model
model_ii<-lm(logSales~Time+I(Time^2)+I(Time^3)+I(Time^4)+fMonth+obs339+c348+s348+c432+s432, data = sales

# find the seasonal indices
b1<-coef(model_ii)[1]
b2<-coef(model_ii)[6:16]+b1
b3<-c(b1,b2)
seas_ii<-b3-mean(b3)

seas_ii
```

```
##   (Intercept)       fMonth2       fMonth3       fMonth4       fMonth5
## -0.0021797639 -0.0780915610  0.0070070144 -0.0211859072  0.0338868547
##       fMonth6       fMonth7       fMonth8       fMonth9      fMonth10
## -0.0009865847  0.0241360169  0.0155023179 -0.0254130253 -0.0011725040
##      fMonth11      fMonth12
##  0.0009546502  0.0475424922
```

(c) Give a careful comparison, using a table and a plot, of the results in parts (a) and (b). Do you find evidence of dynamic seasonality?

ANSWER: No, there is no evidence of dynamic seasonality. If there is dynamic seasonality the seasonal indices should differ significantly, yet the 1992-2007 indices and 2008-2019 indices are relatively similar to each other. A glance at the plot below reveals that the seasonality component in the two periods are largely the same, and therefore seasonality is largely static and not dynamic.

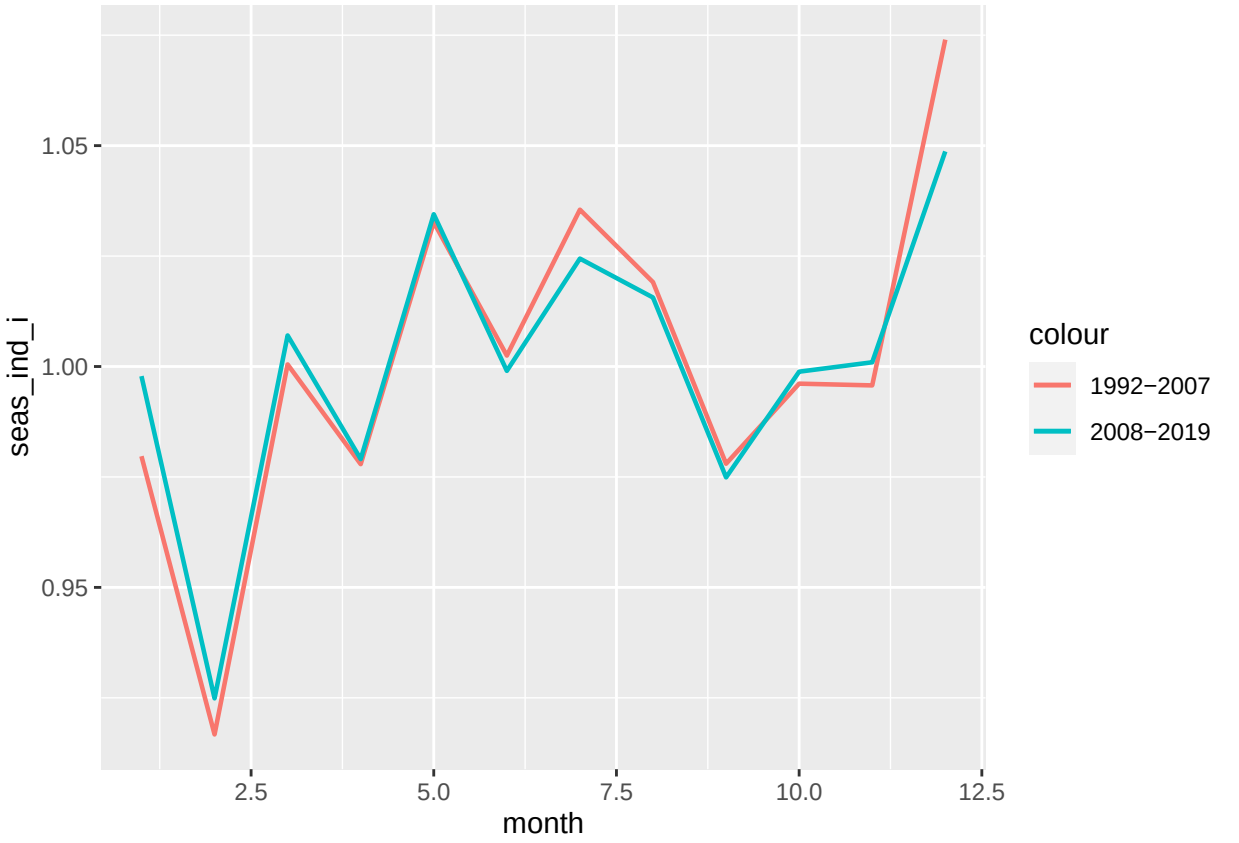
(i) Table Comparison

```
month <- seq(12)
seas_ind_i <- exp(seas_i)
seas_ind_ii <- exp(seas_ii)
seas_df1 <- data.frame(month, seas_ind_i, seas_ind_ii)
print.data.frame(seas_df1)
```

```
##             month seas_ind_i seas_ind_ii
## (Intercept)     1  0.9796997   0.9978226
## fMonth2         2  0.9167122   0.9248797
## fMonth3         3  1.0004774   1.0070316
## fMonth4         4  0.9779003   0.9790369
## fMonth5         5  1.0326431   1.0344676
## fMonth6         6  1.0024701   0.9990139
## fMonth7         7  1.0354961   1.0244296
## fMonth8         8  1.0190980   1.0156231
## fMonth9         9  0.9779878   0.9749072
## fMonth10       10  0.9961172   0.9988282
## fMonth11       11  0.9957266   1.0009551
## fMonth12       12  1.0739945   1.0486908
```

(ii) Plot Comparison

```
ggplot(seas_df1, aes(x = month)) +
  geom_line(aes(y = seas_ind_i, color = '1992-2007'), size = .8) +
  geom_line(aes(y = seas_ind_ii, color = '2008-2019'), size =.8)
```



Q6

6. Write a summary of the results obtained from parts 2 through 5.

ANSWER:

Trend: The fourth-degree polynomial trend component captures the trend well, as indicated by the relatively flat residuals shown in part 2. However, we do see a slight upward trend and volatility remain in the residuals plot, which indicate that there remains some trend component that is not yet captured.

Seasonality: The month dummy variables in the model capture seasonality well. We compare our model seasonal indices and decompose seasonal indices and found they are nearly identical.

The seasonality structure of our time series is static. We compare indices from the years 1992 to 2007 and 2008 to 2019 and found that they are nearly identical, and thus we conclude that seasonality is static and not dynamic.

Normality: QQplot and Shapiro-wilk test indicate that there is not enough evidence to conclude that the residuals are not normally distributed.

Autocorrelation: The autocorrelation function graph of model 1 indicated that there are high correlation in lags 1 to 12, which indicates that our original model did not adequately capture trend. We introduced three lag residual variables to take into account autocorrelation component of the time series, and saw that the introduction led to residual plot which is flatter and less volatile. This indicates that the new variables

capture autocorrelation well and helped the model take into account autocorrelation which was not accounted for previously, and this led to residuals that are closer to white-noise level.