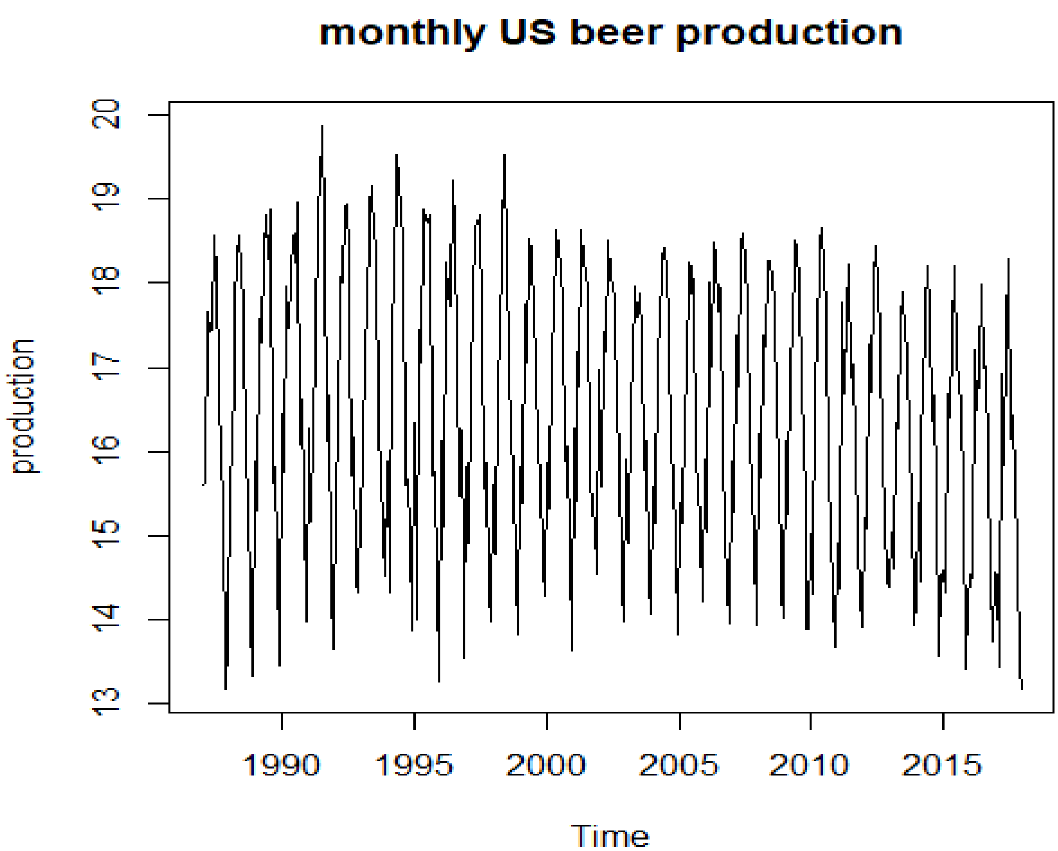
Statistics 435/535/711 24 January 2022
P. Shaman

Monthly U.S. beer production, January 1987—December 2017

We use the U.S. beer production time series to illustrate the use of decomposition models to estimate trend and seasonality.

```
> usbeer<-read.csv("F:/Stat71122Spring/beernew.txt",header=T)
> attach(usbeer)
> head(usbeer)
  year month time   beer        c348       s348       c432       s432
1 1987     1    1 15.601 -0.57757270  0.8163393 -0.9101060  0.4143756
2 1987     2    2 15.633 -0.33281954 -0.9429905  0.6565858 -0.7542514
3 1987     3    3 17.656  0.96202767  0.2729519 -0.2850193  0.9585218
4 1987     4    4 17.422 -0.77846230  0.6276914 -0.1377903 -0.9904614
5 1987     5    5 17.436 -0.06279052 -0.9980267  0.5358268  0.8443279
6 1987     6    6 18.584  0.85099448  0.5251746 -0.8375280 -0.5463943
```

Recall the plot of the production time series shown in the 31 August notes.



Additive decomposition model with month seasonal dummies. Initially the trend is increasing, and then it decreases, levels, and subsequently decreases again, suggesting that U.S. per capital beer consumption has been declining in recent years.

The plot shows presence of a modest trend with three turns. We start by fitting an additive decomposition model with a fourth-degree polynomial trend and a seasonal component. This is a reasonable start. The model is

$$y_t = \beta_0 + \beta_1 t + \beta_2 t^2 + \beta_3 t^3 + \beta_4 t^4 + S_t + \varepsilon_t.$$

The seasonal indices add to zero, $S_1 + \cdots + S_{12} = 0,$ and $S_t \equiv S_{t+12}$ for all t. The seasonal component used here is deterministic, and it is static rather than dynamic (that is, each cycle shows exactly the same pattern).

As noted in the 12 January notes, there are multiple possible formulations of dummy variables to estimate seasonal structure. The U.S. beer production time series is observed monthly, and a complete cycle spans 12 consecutive observations. We present five different formulations of the dummy variables for seasonal estimation of this time series.

(i) *Month seasonal dummies created by R with inclusion of a model intercept.* We begin by using the month dummies created by R, with inclusion of an intercept in the regression model (inclusion of an intercept is the default option in the `lm` command in R), to estimate the seasonal structure. An advantage of this approach is that R creates the dummies. Its disadvantages are twofold: we need to write several lines of R code to obtain the desired estimated seasonal indices, and the t tests for the individual dummies do not offer convenient interpretation.

The variable *month* has numerical values, but will be treated in R as a categorical variable. It needs to be formatted as a factor variable.

```
> fmonth<-as.factor(month)
> levels(fmonth)
 [1] "1"  "2"  "3"  "4"  "5"  "6"  "7"  "8"  "9"  "10" "11" "12"

> class(time)
[1] "integer"
```

To avoid a computer overflow and ensure numerical accuracy, we change the class designation for the variable *time* to numeric.

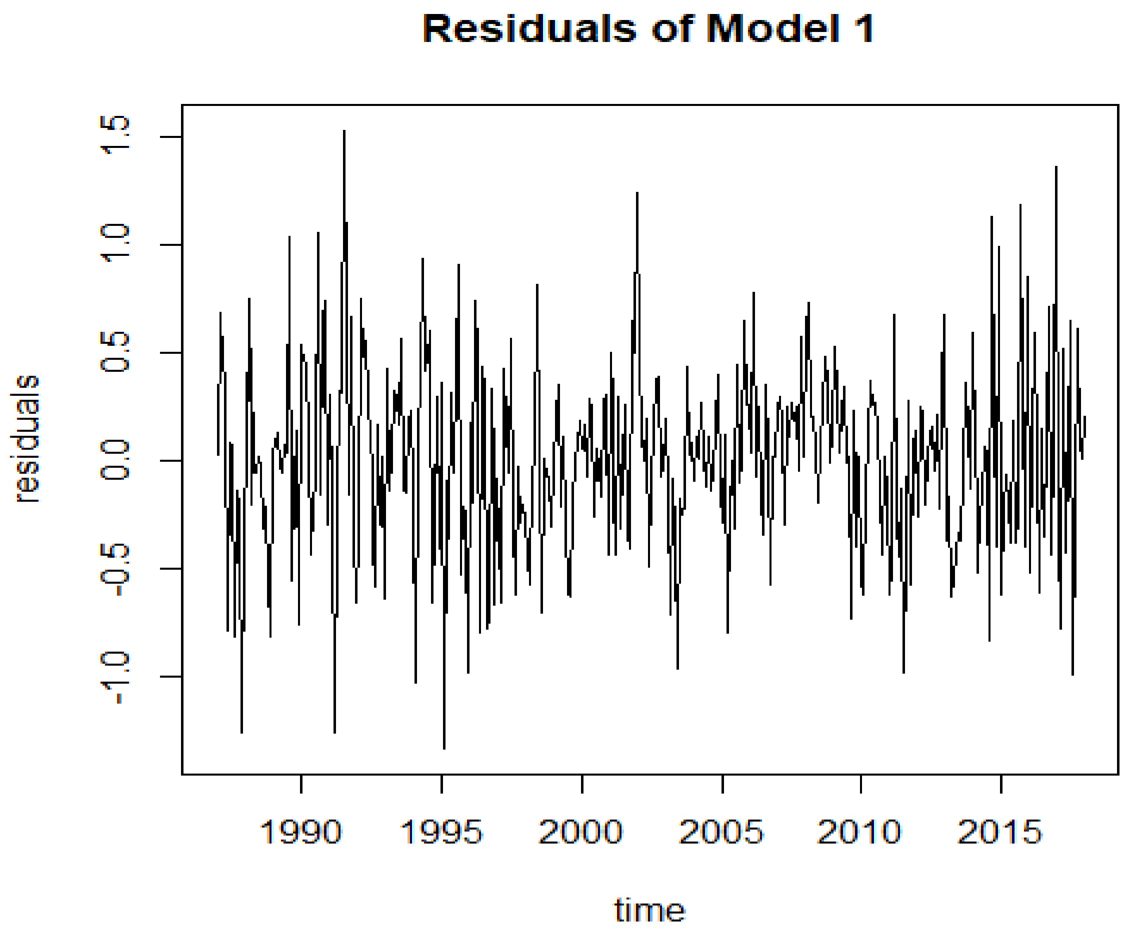
```
> time<-as.numeric(time)
> class(time)
[1] "numeric"

> model1<-
lm(beer~time+I(time^2)+I(time^3)+I(time^4)+fmonth);summary(model1)

Call:
lm(formula = beer ~ time + I(time^2) + I(time^3) + I(time^4) +
    fmonth)
```

```
Residuals:
     Min       1Q   Median       3Q      Max 
-1.33863 -0.26965  0.01012  0.25978  1.52952 

Coefficients:
              Estimate Std. Error t value Pr(>|t|)    
(Intercept)  1.556e+01  1.390e-01 111.933  < 2e-16 ***
time         1.523e-02  4.350e-03   3.502 0.000521 ***
I(time^2)   -1.675e-04  4.732e-05  -3.540 0.000453 ***
I(time^3)    6.496e-07  1.905e-07   3.410 0.000723 ***
I(time^4)   -8.780e-10  2.533e-10  -3.466 0.000594 ***
fmonth2     -6.423e-01  1.132e-01  -5.676 2.87e-08 ***
fmonth3      1.579e+00  1.132e-01  13.950  < 2e-16 ***
fmonth4      1.447e+00  1.132e-01  12.786  < 2e-16 ***
fmonth5      2.595e+00  1.132e-01  22.930  < 2e-16 ***
fmonth6      2.862e+00  1.132e-01  25.289  < 2e-16 ***
fmonth7      2.363e+00  1.132e-01  20.874  < 2e-16 ***
fmonth8      1.950e+00  1.132e-01  17.223  < 2e-16 ***
fmonth9      2.838e-01  1.132e-01   2.507 0.012627 *  
fmonth10     9.398e-03  1.132e-01   0.083 0.933896    
fmonth11    -1.264e+00  1.132e-01 -11.161  < 2e-16 ***
fmonth12    -1.700e+00  1.133e-01 -15.013  < 2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1 

Residual standard error: 0.4455 on 356 degrees of freedom
Multiple R-squared:  0.9237,    Adjusted R-squared:  0.9204 
F-statistic: 287.2 on 15 and 356 DF,  p-value: < 2.2e-16
```

When an intercept is included in the model, R creates 11 month dummies, and the month which does not receive a dummy is the one which is first (the default in R) in the ordering of the months.

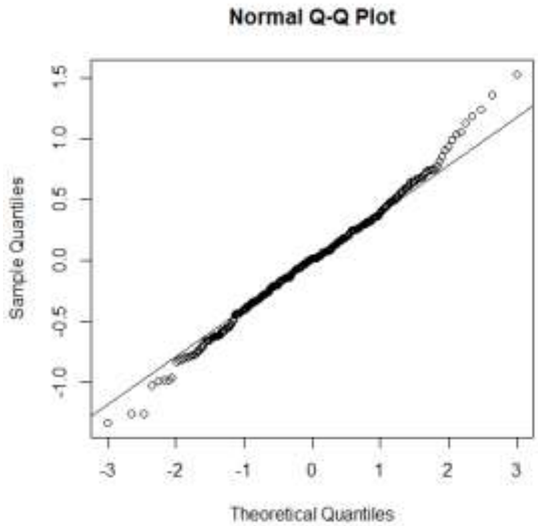
Before turning to interpretation of the model results, let’s examine the residuals from this fit. We form a time series plot of the residuals.

```
> 
plot(ts(resid(model1),start=c(1987,1),freq=12),xlab="time",ylab="residu
als",main="Residuals of Model 1")
```



According to the plot, the model does not fully capture the trend structure. In addition, there is some changing volatility, although the changes are not severe. A normal quantile plot of the residuals will check the third basic assumption (page 4 of the 12 January notes) and also help identify outliers which may need attention.

```
> qqnorm(resid(model1))
> qqline(resid(model1))
```



The tails of the residual distribution are a bit long relative to normality, primarily the upper tail. However, this pattern of departure from normality is unlikely to pose a problem for inference from the fitted model. Let's test for normality of the residuals using the Shapiro–Wilk test. The null hypothesis is normality of the residuals.

```
> shapiro.test(resid(model1))

        Shapiro-Wilk normality test

data:  resid(model1)
W = 0.99333, p-value = 0.09881
```

The result gives marginal significance, barely so, for non-normality. Nevertheless, let's proceed.

Next, we calculate and interpret the estimated seasonal indices from the model. The code which follows first selects the estimated intercept coefficient and the estimated coefficients for the *fmonth* dummies, and then it calculates the seasonal index estimates from these.

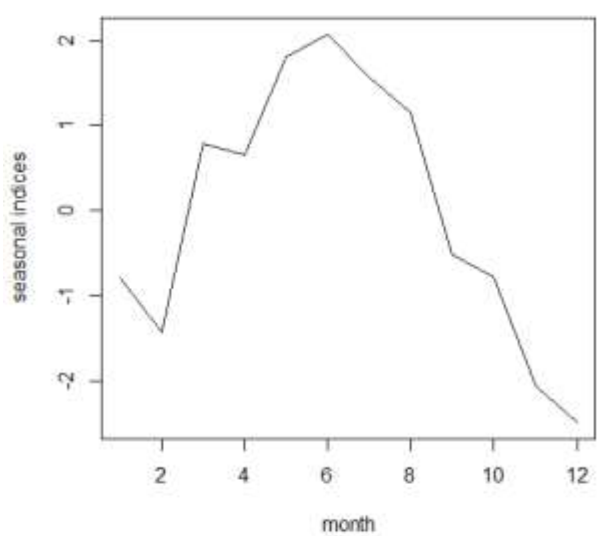
```
> b1<-coef(model1)[1]
> b2<-coef(model1)[6:16]+b1
> b3<-c(b1,b2)
> seas<-b3-mean(b3)
> seas

(Intercept)     fmonth2     fmonth3     fmonth4     fmonth5     fmonth6
 -0.7902230  -1.4325268   0.7884912   0.6568310   1.8049441   2.0721854
    fmonth7     fmonth8     fmonth9    fmonth10    fmonth11    fmonth12
  1.5725549   1.1594398  -0.5064181  -0.7808250  -2.0540390  -2.4904147
```

The rationale for the code is as follows. With the dummies produced by R (see pages 26–27 of the 12 January notes), the regression coefficients in model 1 are actually estimates of the following parameters:

(Intercept)	Intercept + S_1	fmonth7	$S_7 - S_1$
fmonth2	$S_2 - S_1$	fmonth8	$S_8 - S_1$
fmonth3	$S_3 - S_1$	fmonth9	$S_9 - S_1$
fmonth4	$S_4 - S_1$	fmonth10	$S_{10} - S_1$
fmonth5	$S_5 - S_1$	fmonth11	$S_{11} - S_1$
fmonth6	$S_6 - S_1$	fmonth12	$S_{12} - S_1$

The code above utilizes the fact that $S_1 + \cdots + S_{12} = 0$. We now change the variable *seas* to time series class and plot the estimated indices.

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



Here are the estimated indices in tabular form:

J	−0.7902	J	1.5726
F	−1.4325	A	1.1594
M	0.7885	S	−0.5064
A	0.6568	O	−0.7808
M	1.8049	N	−2.0540
J	2.0722	D	−2.4904

In June production is estimated to be 2.072 million barrels above the level of the trend, and in November it is estimated to be 2.054 million barrels below the trend level, for example. For this additive decomposition model, $y_t = T_t + S_t + \varepsilon_t$, we can *deseasonalize* the data by subtracting the estimated seasonal indices (estimates of S_t) from the data values y_t.

Of course, confidence interval estimates can be given for the seasonal indices.

Partial F test. A partial F test is used to test the null hypothesis that some of the regression coefficients are simultaneously 0. In general, here is the construction of the test statistic. Fit two models, a full model with all regressors included and a reduced model. In the latter, omit the variables corresponding to the parameters which are hypothesized to be 0. In particular, assume that k is the number of parameters (not including the intercept) in the full model and that the null hypothesis specifies that r of these parameters are 0. Thus, the number of parameters (not including the intercept) in the reduced model is $k - r$.

Suppose T is the number of observations. The F statistic for the partial F test can be calculated from the R square values of the two models. The F statistic is

$$F_{r,T-k-1} = \frac{\left(R^2(\text{full}) - R^2(\text{reduced})\right)/r}{\left(1 - R^2(\text{full})\right)/(T-k-1)}.$$

This partial F statistic has r numerator degrees of freedom and $T - k - 1$ denominator degrees of freedom. We reject the null hypothesis if the partial F statistic is sufficiently large, or equivalently, if the p-value is sufficiently small.

For the analysis above of the beer data, we can construct a partial F test of the hypothesis that all the seasonal parameters are 0 by comparing model 1 and a reduced model which includes only a fourth-degree polynomial trend.

```
> model2<-lm(beer~time+I(time^2)+I(time^3)+I(time^4));summary(model2)

Call:
lm(formula = beer ~ time + I(time^2) + I(time^3) + I(time^4))

Residuals:
    Min      1Q  Median      3Q     Max 
-3.4602 -1.1566  0.1497  1.3892  3.0871 

Coefficients:
              Estimate Std. Error t value Pr(>|t|)    
(Intercept)  1.643e+01  4.099e-01  40.077   <2e-16 ***
time         1.386e-02  1.518e-02   0.913    0.362    
I(time^2)   -1.643e-04  1.652e-04  -0.994    0.321    
I(time^3)    6.708e-07  6.651e-07   1.008    0.314    
I(time^4)   -9.418e-10  8.846e-10  -1.065    0.288    
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
Residual standard error: 1.556 on 367 degrees of freedom
Multiple R-squared:  0.04045,   Adjusted R-squared:  0.02999
F-statistic: 3.867 on 4 and 367 DF,  p-value: 0.004317
```

Because this model does not include the very strong seasonal component, the standard errors of the parameter estimates are inflated, and thus the t statistics are deflated, giving insignificant *p*-values.

```
> summary(model1)$r.squared
[1] 0.9236588
> summary(model2)$r.squared
[1] 0.04044511
```

$$F_{11,356} = \frac{(0.92366 - 0.04045)/11}{(1-0.92366)/356} = 374.43.$$

This partial F test can also be calculated with R by using the `anova` *command with two arguments.* This is convenient because the result provides the *p*-value for the test.

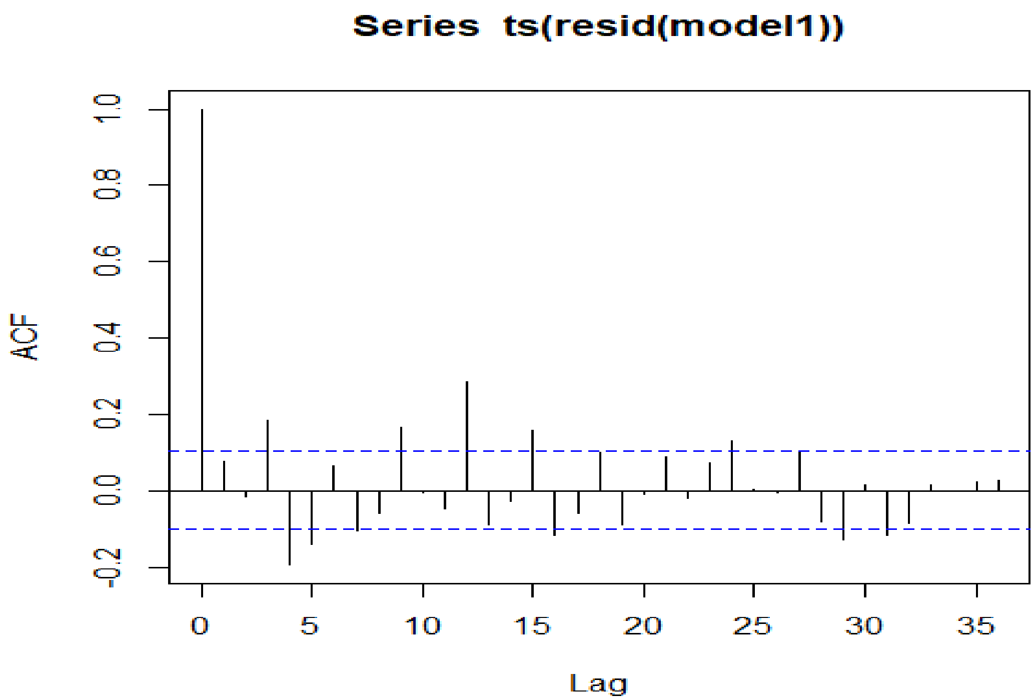
```
> anova(model2,model1)
Analysis of Variance Table

Model 1: beer ~ time + I(time^2) + I(time^3) + I(time^4)
Model 2: beer ~ time + I(time^2) + I(time^3) + I(time^4) + fmonth
  Res.Df    RSS Df Sum of Sq      F    Pr(>F)
1    367 888.27
2    356  70.67 11     817.6 374.42 < 2.2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Let's return to consideration of the residuals from model 1. Do the residuals suggest that the model disturbance term forms an uncorrelated sequence? We start with a calculation of the lag 1 correlation of the residuals, $\text{corr}(\text{resid}_t, \text{resid}_{t-1})$. This measures the correlation between residuals which are directly adjacent in time.

```
> r<-resid(model1)[2:372];r1<-resid(model1)[1:371]
> cor.test(r,r1)

        Pearson's product-moment correlation

data:  r and r1
t = 1.4767, df = 369, p-value = 0.1406
alternative hypothesis: true correlation is not equal to 0
95 percent confidence interval:
 -0.02536446  0.17708380
sample estimates:
       cor
0.07664964
```

The lag one correlation is not significant. There are other lag correlations which need to be considered, and a more complete picture of the autocorrelation structure of the residuals is obtained by examining estimates of the correlations at multiple lags. To obtain these correlations, convert the residual series to time series format and use the command `acf` (autocorrelation function). Here we obtain the calculations out to lag 36 (if no count is specified, the default in R is 26 lags).

```
> acf(ts(resid(model1)),36)
```



We see that there are significant residual correlations at many lags (significance is indicated by extension beyond the blue dashed lines).

A white noise sequence has zero correlation at all lags. We conclude that model 1 does not reduce the data to residuals which are consistent with a white noise hypothesis. Despite this, the estimates of trend and seasonal components do provide useful insight into the structure of the beer time series. The model which has been fit forces a perfectly periodic seasonal component, a static representation. The residual correlations shown above suggest that a dynamic representation of the seasonal component, one which allows the seasonal pattern to vary somewhat from year to year, would be more appropriate. The evidence for this conclusion is the significance of correlations at lags 12 and 24. The overall results also suggest, however, that the present static representation does capture major features of the seasonal component of the beer time series.

(ii) *Month seasonal dummies created by R without inclusion of a model intercept.* When an intercept is not included in the model, R creates 12 monthly dummies, rather than 11. Each month receives a dummy equal to 1 for the specified month and equal to 0 for all other months.

```
> model3<-
lm(beer~0+time+I(time^2)+I(time^3)+I(time^4)+fmonth);summary(model3)

Call:
lm(formula = beer ~ 0 + time + I(time^2) + I(time^3) + I(time^4) +
    fmonth)

Residuals:
     Min       1Q   Median       3Q      Max
-1.33863 -0.26965  0.01012  0.25978  1.52952

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
time       1.523e-02  4.350e-03   3.502 0.000521 ***
I(time^2) -1.675e-04  4.732e-05  -3.540 0.000453 ***
I(time^3)  6.496e-07  1.905e-07   3.410 0.000723 ***
I(time^4) -8.780e-10  2.533e-10  -3.466 0.000594 ***
fmonth1    1.556e+01  1.390e-01 111.933  < 2e-16 ***
fmonth2    1.492e+01  1.393e-01 107.109  < 2e-16 ***
fmonth3    1.714e+01  1.395e-01 122.834  < 2e-16 ***
fmonth4    1.701e+01  1.398e-01 121.680  < 2e-16 ***
fmonth5    1.815e+01  1.400e-01 129.682  < 2e-16 ***
fmonth6    1.842e+01  1.402e-01 131.386  < 2e-16 ***
fmonth7    1.792e+01  1.404e-01 127.634  < 2e-16 ***
fmonth8    1.751e+01  1.406e-01 124.518  < 2e-16 ***
fmonth9    1.584e+01  1.408e-01 112.523  < 2e-16 ***
fmonth10   1.557e+01  1.410e-01 110.437  < 2e-16 ***
fmonth11   1.430e+01  1.411e-01 101.287  < 2e-16 ***
fmonth12   1.386e+01  1.413e-01  98.089  < 2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.4455 on 356 degrees of freedom
Multiple R-squared:  0.9993,    Adjusted R-squared:  0.9993
F-statistic: 3.197e+04 on 16 and 356 DF,  p-value: < 2.2e-16
```

To calculate the estimated seasonal indices from this model, we use the following code. It merely selects the estimated coefficients for the month dummies and demeans them.

```
> b2<-coef(model3)[5:16]
> seas3<-b2-mean(b2)

> seas3
   fmonth1    fmonth2    fmonth3    fmonth4    fmonth5    fmonth6    fmonth7
-0.7902230 -1.4325268  0.7884912  0.6568310  1.8049441  2.0721854  1.5725549
   fmonth8    fmonth9   fmonth10   fmonth11   fmonth12
 1.1594398 -0.5064181 -0.7808250 -2.0540390 -2.4904147
```

The rationale for this code shown is simple. The coefficients for the month dummies in model 3 are estimates of the following parameters:

fmonth1	Intercept + S_1	Intercept + S_7
fmonth2	Intercept + S_2	Intercept + S_8
fmonth3	Intercept + S_3	Intercept + S_9
fmonth4	Intercept + S_4	Intercept + S_{10}
fmonth5	Intercept + S_5	Intercept + S_{11}
fmonth6	Intercept + S_6	Intercept + S_{12}

(iii) *Self-created month seasonal dummies with inclusion of a model intercept.* We illustrate this option with the dummies specified on pages 27–28 of the 12 January notes. The advantage of this approach is that the estimated coefficients for the month dummies directly give the desired seasonal index estimates, and they are easy to interpret. The disadvantage is that we have to create the dummies ourselves. Note that, when an intercept is included, one month (in this case, December) does not receive a dummy (this can be changed so that another one of the months is the one which does not receive a dummy).

```
> SJ1<-c(rep(c(1,rep(0,10),-1),31))
> SJ2<-c(rep(c(0,1,rep(0,9),-1),31))
> SJ3<-c(rep(c(rep(0,2),1,rep(0,8),-1),31))
> SJ4<-c(rep(c(rep(0,3),1,rep(0,7),-1),31))
> SJ5<-c(rep(c(rep(0,4),1,rep(0,6),-1),31))
> SJ6<-c(rep(c(rep(0,5),1,rep(0,5),-1),31))
> SJ7<-c(rep(c(rep(0,6),1,rep(0,4),-1),31))
> SJ8<-c(rep(c(rep(0,7),1,rep(0,3),-1),31))
> SJ9<-c(rep(c(rep(0,8),1,rep(0,2),-1),31))
> SJ10<-c(rep(c(rep(0,9),1,0,-1),31))
> SJ11<-c(rep(c(rep(0,10),1,-1),31))

> model4<-
lm(beer~time+I(time^2)+I(time^3)+I(time^4)+SJ1+SJ2+SJ3+SJ4+SJ5+SJ6+SJ7+
SJ8+SJ9+SJ10+SJ11);summary(model4)

Call:
lm(formula = beer ~ time + I(time^2) + I(time^3) + I(time^4) +
    SJ1 + SJ2 + SJ3 + SJ4 + SJ5 + SJ6 + SJ7 + SJ8 + SJ9 + SJ10 +
    SJ11)

Residuals:
     Min       1Q   Median       3Q      Max
-1.33863 -0.26965  0.01012  0.25978  1.52952
```

```
Coefficients:
              Estimate Std. Error t value Pr(>|t|)    
(Intercept)  1.635e+01  1.175e-01 139.182  < 2e-16 ***
time         1.523e-02  4.350e-03   3.502 0.000521 ***
I(time^2)   -1.675e-04  4.732e-05  -3.540 0.000453 ***
I(time^3)    6.496e-07  1.905e-07   3.410 0.000723 ***
I(time^4)   -8.780e-10  2.533e-10  -3.466 0.000594 ***
SJ1         -7.902e-01  7.665e-02 -10.310  < 2e-16 ***
SJ2         -1.433e+00  7.664e-02 -18.693  < 2e-16 ***
SJ3          7.885e-01  7.663e-02  10.290  < 2e-16 ***
SJ4          6.568e-01  7.662e-02   8.572 3.14e-16 ***
SJ5          1.805e+00  7.662e-02  23.558  < 2e-16 ***
SJ6          2.072e+00  7.662e-02  27.046  < 2e-16 ***
SJ7          1.573e+00  7.662e-02  20.525  < 2e-16 ***
SJ8          1.159e+00  7.662e-02  15.133  < 2e-16 ***
SJ9         -5.064e-01  7.662e-02  -6.609 1.41e-10 ***
SJ10        -7.808e-01  7.663e-02 -10.190  < 2e-16 ***
SJ11        -2.054e+00  7.664e-02 -26.802  < 2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.4455 on 356 degrees of freedom
Multiple R-squared:  0.9237,    Adjusted R-squared:  0.9204 
F-statistic: 287.2 on 15 and 356 DF,  p-value: < 2.2e-16
```

The seasonal index estimate for December is the negative sum of the other 11 estimated indices.

```
> -sum(coef(model4)[6:16])
[1] -2.490415
```

The output shows that, unlike the methods with the month dummies defined by R, the use of these month dummies directly gives significance tests for the individual seasonal indices.

(iv) *Multiplicative decomposition model with month seasonal dummies and inclusion of an intercept.* Instead of the additive decomposition structure described above, we now fit a multiplicative model to the data. Although there is evidence of only mildly changing volatility and thus a multiplicative model is not needed, it is nonetheless useful to consider the alternative interpretation offered by the multiplicative formulation. The specification we use is

(1) $$\log y_t = \beta_0 + \beta_1 t + \beta_2 t^2 + \beta_3 t^3 + \beta_4 t^4 + \log S_t + \log \varepsilon_t.$$

The $\log S_t$ values add to 0, $\log S_1 + \cdots + \log S_{12} = 0$. This implies

$$\prod_{t=1}^{12} S_t = 1,$$

and, in this multiplicative model formulation, we deseasonalize the data by dividing the observations y_t by the estimates of S_t. We let R define the month seasonal dummies and include an intercept in the model.

```
> model5<-
lm(log(beer)~time+I(time^2)+I(time^3)+I(time^4)+fmonth);summary(model5)

Call:
lm(formula = log(beer) ~ time + I(time^2) + I(time^3) + I(time^4) +
    fmonth)

Residuals:
      Min        1Q    Median        3Q       Max
-0.089963 -0.016002  0.000251  0.015319  0.089566

Coefficients:
              Estimate Std. Error t value Pr(>|t|)
(Intercept)  2.745e+00  8.532e-03 321.680  < 2e-16 ***
time         8.749e-04  2.670e-04   3.277  0.00115 **
I(time^2)   -9.611e-06  2.905e-06  -3.309  0.00103 **
I(time^3)    3.757e-08  1.169e-08   3.213  0.00143 **
I(time^4)   -5.143e-11  1.555e-11  -3.307  0.00104 **
fmonth2     -4.198e-02  6.946e-03  -6.043 3.82e-09 ***
fmonth3      9.620e-02  6.947e-03  13.849  < 2e-16 ***
fmonth4      8.858e-02  6.947e-03  12.752  < 2e-16 ***
fmonth5      1.535e-01  6.947e-03  22.094  < 2e-16 ***
fmonth6      1.682e-01  6.947e-03  24.215  < 2e-16 ***
fmonth7      1.405e-01  6.948e-03  20.218  < 2e-16 ***
fmonth8      1.172e-01  6.948e-03  16.872  < 2e-16 ***
fmonth9      1.846e-02  6.949e-03   2.656  0.00826 **
fmonth10     7.697e-04  6.950e-03   0.111  0.91188
fmonth11    -8.425e-02  6.951e-03 -12.121  < 2e-16 ***
fmonth12    -1.147e-01  6.951e-03 -16.507  < 2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.02735 on 356 degrees of freedom
Multiple R-squared:  0.9242,    Adjusted R-squared:  0.921
F-statistic: 289.3 on 15 and 356 DF,  p-value: < 2.2e-16
```

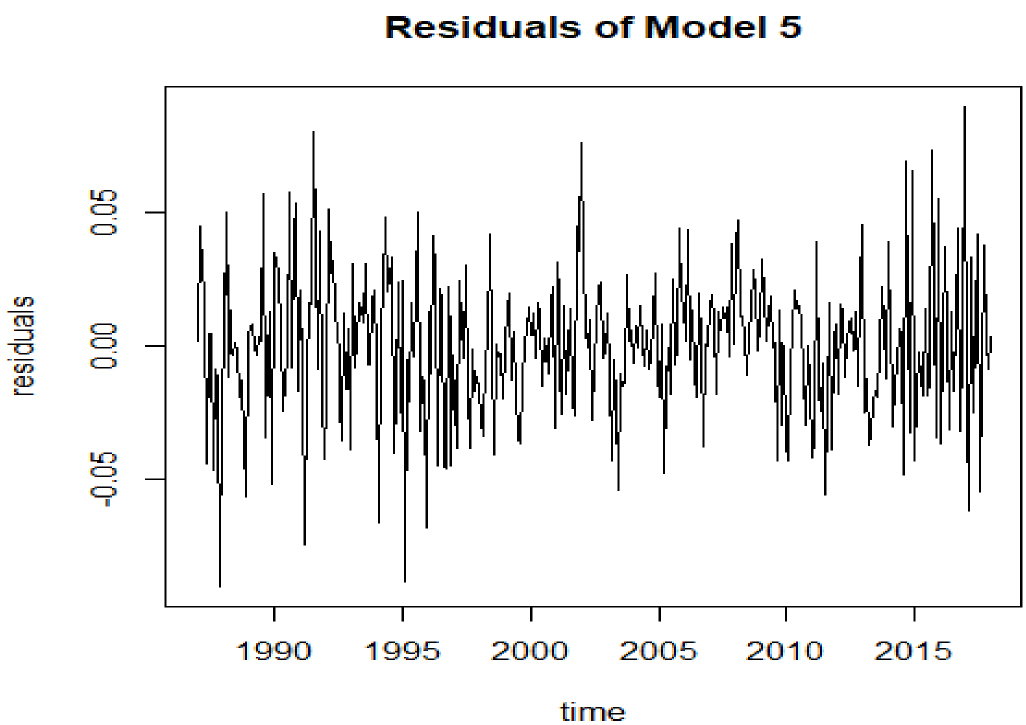
Let's obtain the estimated seasonal indices. The code produces estimates of the $\log S_t$ values, which need to be exponentiated to obtain the S_t estimates.

```
> b1<-coef(model5)[1]
> b2<-coef(model5)[6:16]+b1
> b3<-c(b1,b2)
> seas5<-b3-mean(b3)

> seas5
(Intercept)     fmonth2     fmonth3     fmonth4     fmonth5     fmonth6
-0.04520635 -0.08718237  0.05099777  0.04337636  0.10828291  0.12302736
    fmonth7     fmonth8     fmonth9    fmonth10    fmonth11    fmonth12
 0.09526824  0.07202771 -0.02674877 -0.04443668 -0.12945275 -0.1599534
```

```
> cbind(seas5,exp(seas5))
                  seas5   exp(seas5)
Jan          -0.04520635 0.9558002
Feb          -0.08718237 0.9165099
Mar           0.05099777 1.0523206
Apr           0.04337636 1.0443309
May           0.10828291 1.1143630
Jun           0.12302736 1.1309154
Jul           0.09526824 1.0999539
Aug           0.07202771 1.0746851
Sep          -0.02674877 0.9736058
Oct          -0.04443668 0.9565362
Nov          -0.12945275 0.8785761
Dec          -0.15995342 0.8521835
```

The estimated seasonal indices are interpreted in percentage terms. For example, in January production is estimated to be 4.4 percent below the level of the trend, and, in June production is estimated to be 13.1 percent above trend level. The month of lowest production is December, estimated to be 14.8 percent below trend level.

The residual time series plot is virtually identical in shape to that from the corresponding additive model fit:



However, the vertical axis is now on the log scale.

(v) *Additive decomposition model with cosine and sine seasonal dummies and inclusion of an intercept.* The cosine-sine formulation of the seasonal is an alternative to the use of month seasonal dummy variables, and it allows a different interpretation. The two approaches give identical fits if we use *all* of the cosines and sines (count parameters, to reach 11 seasonal trigonometric dummies). We illustrate with the additive model

$$y_t = \beta_0 + \beta_1 t + \beta_2 t^2 + \beta_3 t^3 + \beta_4 t^4 + \sum_{j=1}^{5}\left(\gamma_j \cos\frac{2\pi jt}{12} + \delta_j \sin\frac{2\pi jt}{12}\right) + \gamma_6 (-1)^t + \varepsilon_t .$$

The seasonal is now written as the sum of a fundamental component with period 12 and its five overtones. The *j*th component has amplitude

$$\sqrt{\gamma_j^2 + \delta_j^2}, \quad j = 1, \ldots, 5,$$

and the last component has amplitude given by the absolute value of γ_6. Note that we need to create the cosine and sine dummy variables.

```
> cosm<-matrix(nrow=length(time),ncol=6)
> sinm<-matrix(nrow=length(time),ncol=5)
> for(i in 1:5){
+ cosm[,i]<-cos(2*pi*i*time/12)
+ sinm[,i]<-sin(2*pi*i*time/12)
+ }
> cosm[,6]<-cos(pi*time)

> model6<-
lm(beer~time+I(time^2)+I(time^3)+I(time^4)+cosm[,1]+sinm[,1]+cosm[,2]+s
inm[,2]+cosm[,3]+sinm[,3]+cosm[,4]+sinm[,4]+cosm[,5]+sinm[,5]+cosm[,6])

> options(digits=10)
> summary(model6)

Call:
lm(formula = beer ~ time + I(time^2) + I(time^3) + I(time^4) +
    cosm[, 1] + sinm[, 1] + cosm[, 2] + sinm[, 2] + cosm[, 3] +
    sinm[, 3] + cosm[, 4] + sinm[, 4] + cosm[, 5] + sinm[, 5] +
    cosm[, 6])

Residuals:
        Min          1Q      Median          3Q         Max
-1.33863334 -0.26964756  0.01011539  0.25978076  1.52951509
```

```
Coefficients:
                  Estimate    Std. Error   t value   Pr(>|t|)
(Intercept)  1.634990e+01  1.174717e-01 139.18164 < 2.22e-16 ***
time         1.523140e-02  4.349744e-03   3.50168 0.00052115 ***
I(time^2)   -1.675209e-04  4.732317e-05  -3.53993 0.00045335 ***
I(time^3)    6.496385e-07  1.904884e-07   3.41038 0.00072317 ***
I(time^4)   -8.780189e-10  2.533498e-10  -3.46564 0.00059359 ***
cosm[, 1]   -1.994269e+00  3.267171e-02 -61.03962 < 2.22e-16 ***
sinm[, 1]    1.738916e-01  3.270197e-02   5.31747 1.8626e-07 ***
cosm[, 2]   -3.919055e-02  3.267142e-02  -1.19954 0.23111728
sinm[, 2]    1.273539e-01  3.267614e-02   3.89746 0.00011615 ***
cosm[, 3]   -8.882959e-02  3.267141e-02  -2.71888 0.00687115 **
sinm[, 3]    3.354931e-02  3.267141e-02   1.02687 0.30517856
cosm[, 4]   -3.403903e-02  3.267141e-02  -1.04186 0.29818368
sinm[, 4]    1.703934e-01  3.266983e-02   5.21562 3.1153e-07 ***
cosm[, 5]   -1.982014e-01  3.267141e-02  -6.06651 3.3427e-09 ***
sinm[, 5]    5.071124e-01  3.266921e-02  15.52264 < 2.22e-16 ***
cosm[, 6]   -1.358850e-01  2.310134e-02  -5.88213 9.3415e-09 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.4455463 on 356 degrees of freedom
Multiple R-squared:  0.9236588, Adjusted R-squared:  0.9204422
F-statistic: 287.1518 on 15 and 356 DF,  p-value: < 2.2204e-16
```

Note that the estimated coefficients for *time* and its powers, and the *R* square value and *F* value are the same as for model 1. We test the significance of a cosine-sine pair with a partial *F* test. If the pair is judged to be insignificant, delete both variables forming the pair. If the pair is judged to be significant, keep both variables forming the pair. *Don't delete one member of a significant pair if it is insignificant via its corresponding t test.*

Let's illustrate by testing the third harmonic pair. Here are the reduced model command and the partial *F* test calculation:

```
> model7<-
lm(beer~time+I(time^2)+I(time^3)+I(time^4)+cosm[,1]+sinm[,1]+cosm[,2]+s
inm[,2]+cosm[,4]+sinm[,4]+cosm[,5]+sinm[,5]+cosm[,6])

> anova(model7,model6)
Analysis of Variance Table

Model 1: beer ~ time + I(time^2) + I(time^3) + I(time^4) + cosm[, 1] +
    sinm[, 1] + cosm[, 2] + sinm[, 2] + cosm[, 4] + sinm[, 4] +
    cosm[, 5] + sinm[, 5] + cosm[, 6]
Model 2: beer ~ time + I(time^2) + I(time^3) + I(time^4) + cosm[, 1] +
    sinm[, 1] + cosm[, 2] + sinm[, 2] + cosm[, 3] + sinm[, 3] +
    cosm[, 4] + sinm[, 4] + cosm[, 5] + sinm[, 5] + cosm[, 6]
  Res.Df    RSS Df Sum of Sq     F Pr(>F)
1    358 72.347
2    356 70.670  2    1.6766 4.223 0.0154 *
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

The p-value is 0.015. Thus the pair is significant, and we retain both members of the pair.

Amplitude estimates are useful to measure the intensity of each harmonic component, and phase estimates indicate alignment.

```
> ampltd<-c(rep(0,times=6))
> b2<-coef(model6)[6:16]
> for(i in 1:5){
+ i1<-2*i-1
+ i2<-i1+1
+ ampltd[i]<-sqrt(b2[i1]^2+b2[i2]^2)
+ }
> ampltd[6]<-abs(b2[11])

> ampltd
[1] 2.00183602 0.13324758 0.09495395 0.17376004 0.54446922 0.13588504


> phase<-c(rep(0,times=6))
> for(i in 1:5){
+ i1<-2*i-1
+ i2<-i1+1
+ phase[i]<-atan(-b2[i2]/b2[i1])
+ if(b2[i1]<0)phase[i]<-phase[i]+pi
+ if((b2[i1]>0)&(b2[i2]>0))phase[i]<-phase[i]+2*pi
+ }
> if(b2[11]<0)phase[6]<-pi

> phase
[1] 3.228568 4.413856 3.502712 4.515217 4.339801 3.141593

> phase*180/pi
[1] 184.9833 252.8953 200.6906 258.7029 248.6523 180.0000

> peak<-c(rep(0,times=6))
> for(i in 1:5){
+ peak[i]<-(12/i)-6*phase[i]/(pi*i)
+ }
> if(phase[6]>0)peak[6]<-1

> peak
[1] 5.8338887 1.7850779 1.7701041 0.8441426 0.7423180 1.0000000
```

Amplitude and Phase Estimates

	Amplitude	Phase Degrees	Phase Radians	Peak t (period)
Fundamental	2.00	184.98	3.229	5.83 (12)
Second harmonic	0.13	252.90	4.414	1.79, 7.79 (6)
Third harmonic	0.10	200.69	3.503	1.77, 5.77, 9.77 (4)
Fourth harmonic	0.17	258.70	4.515	0.84, 3.84, 6.84, 9.84 (3)
Fifth harmonic	0.54	248.65	4.340	0 .74, 3.14, 5.54, 7.94, 10.34 (2.4)
Sixth harmonic	0.14	180.00	3.142	1.00, 3.00, 5.00, 7.00, 9.00, 11.00 (2)

Note that there are multiple peaks for the overtone components. The fundamental is the most prominent component, as judged from the amplitudes. The peak calculations point toward maximum seasonal production occurring in June.

Let's illustrate the calculation of the phase angle and peak using the estimate of the fundamental component. Recall that

$$R \cos(\lambda t + \alpha) = R \cos \alpha \cos \lambda t + (-R \sin \alpha) \sin \lambda t = A \cos \lambda t + B \sin \lambda t.$$

Ostensibly, $\alpha = \tan^{-1}(-B / A)$. The pair $(A, -B)$ can have four sign combinations, corresponding to the four quadrants in the plane. However, the ratio $-B / A$ has only two possible signs. That is, the tangent function has a cycle of 180 degrees, rather than 360 degrees. Thus, an adjustment may be needed after the inverse tangent is calculated. The following table gives the adjustments.

Signs of $(A, -B)$	Adjustment in radians	Adjustment in degrees
(+,+)	none	none
(−,+)	add π	add 180
(−,−)	add π	add 180
(+,−)	add 2π	add 360

For the fundamental component the estimate of A is –1.994269 and the estimate of B is 0.173892, and (−1.994269, −0.173892) is in the third quadrant. We calculate

$$\tan^{-1}(-0.173892/-1.994269) = \tan^{-1}(0.08720) = 4.98 \text{ degrees}.$$

Placing this in the third quadrant (where it belongs), we have 184.98 degrees, or 3.229 radians for the estimate of α. To obtain the peak, write $1 = \cos(2\pi t/12 + 3.229)$, or $2\pi t/12 + 3.229 = 2\pi$. Thus, the peak is at $t = 5.83$ months, toward the end of June.

These amplitude and phase calculations represent a kind of spectral analysis. Later we will utilize spectral methods to judge the adequacy of models fit to data.

This fit with a trigonometric basis for the seasonal can be used to form estimated seasonal indices.

Use of the `decompose` *function in R.* R provides an alternative method for estimation of trend and seasonal components in a decomposition model. In the additive case with monthly data, it estimates the trend at time t with a centered moving average,

$$\hat{T}_t = \left(\frac{1}{2}y_{t-6} + y_{t-5} + \cdots + y_{t-1} + y_t + y_{t+1} + \cdots + y_{t+5} + \frac{1}{2}y_{t+6}\right)/12, \quad t = 7, \ldots, T-6.$$

The expression shown is the smoothed value at time t, the center of the moving average span. This moving average operation kills a static seasonal structure, and it attenuates the irregular component. Thus, it essentially estimates a slowly moving trend component. Note that we lose the first six and the last six time points with this method. To obtain estimates of the seasonal indices, the procedure next performs the following steps.

1. Subtract the trend estimate from the data, yielding

$$y_t - \hat{T}_t, \quad t = 7, \ldots, T-6.$$

2. For each month, average the values obtained in part 1. This yields 12 monthly averages.

3. Center the monthly averages found in part 2. That is, subtract the mean of the 12 monthly averages from each average. This ensures that the resulting centered averages add to zero. These centered averages constitute the seasonal index estimates.

```
> beer.ts<-ts(beer,freq=12)
> beer.decmps<-decompose(beer.ts)
> seasd<-beer.decmps$seasonal

> seasd[1:12]
 [1] -0.7883671296 -1.4519185185  0.7743967593  0.6451856481
 [5]  1.8294245370  2.0652648148  1.6040203704  1.1699606481
 [9] -0.5238185185 -0.7790199074 -2.0501601852 -2.4949685185
```

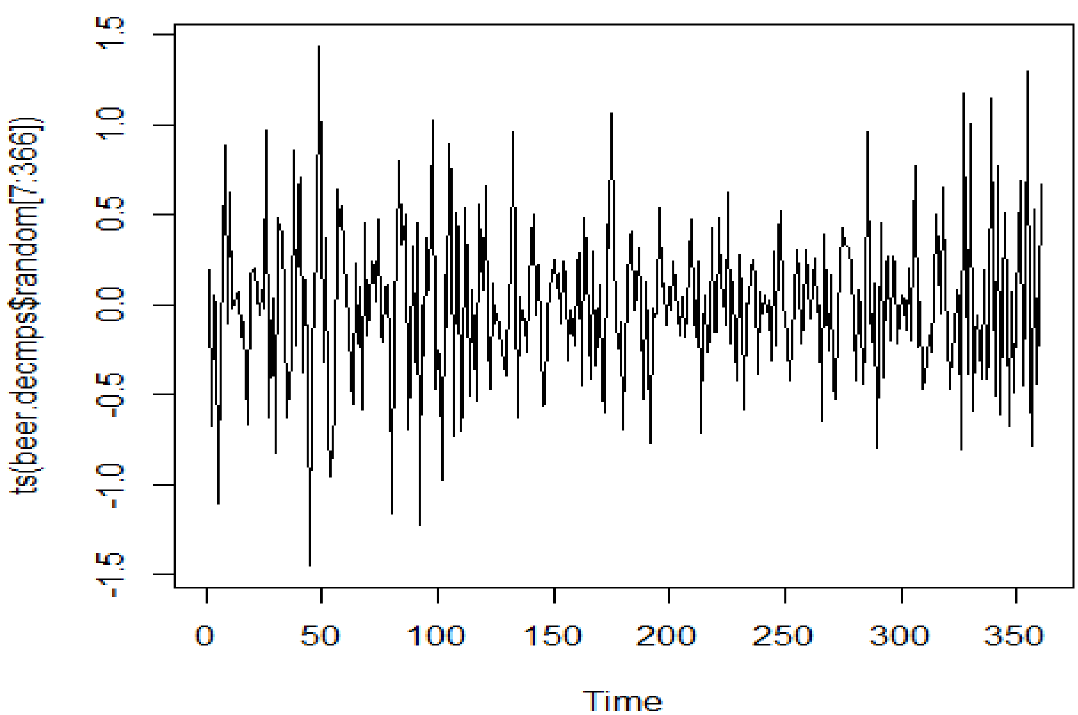
Let’s tabulate these estimates and compare them to the seasonal index estimates from model 1.

```
> options(digits=5)
> cbind(seas,seasd[1:12])
                seas    seasd
Jan         -0.79022 -0.78837
Feb         -1.43253 -1.45192
Mar          0.78849  0.77440
Apr          0.65683  0.64519
May          1.80494  1.82942
Jun          2.07219  2.06526
Jul          1.57255  1.60402
Aug          1.15944  1.16996
Sep         -0.50642 -0.52382
Oct         -0.78082 -0.77902
Nov         -2.05404 -2.05016
Dec         -2.49041 -2.49497
```

The first column gives the estimates obtained from the model 1 regression, and the second column gives estimates from the decomposition method in R.   There is little difference between the two.

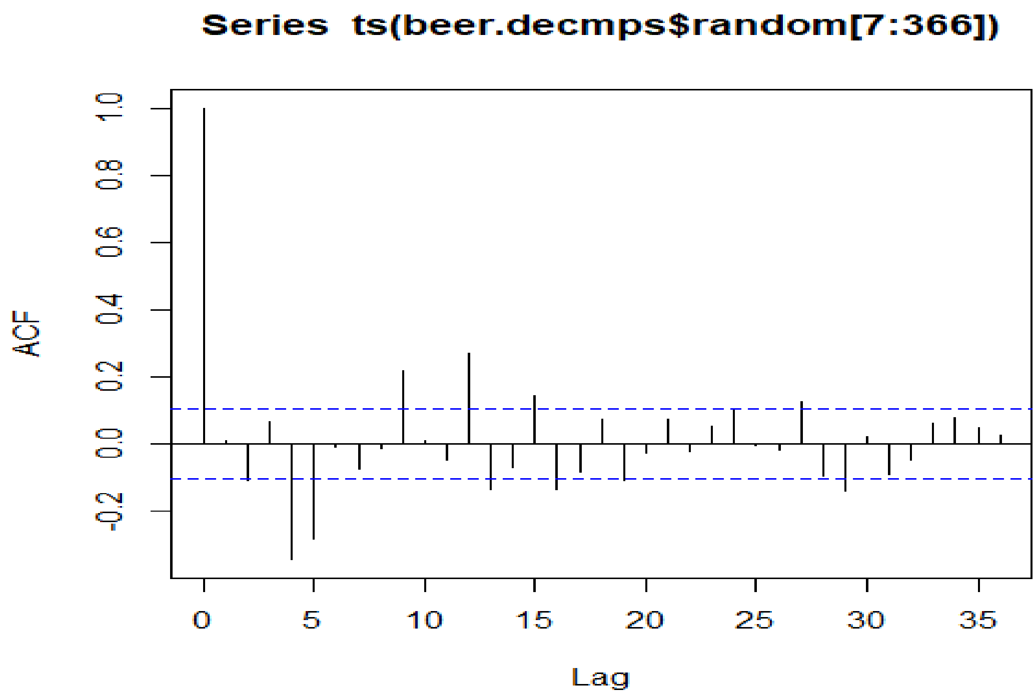
Let's look at the residuals from this decomposition fit, examining their plot and autocorrelations.

```
> plot(ts(beer.decmps$random[7:366]))
```



As noted, the estimation uses a centered moving average to estimate the trend, and the result is that the first six and last six terms of the random component cannot be calculated. Thus, we plot from time 7 to time 366 (the horizontal axis in the plot calls this range 1 to 360).

```
> acf(ts(beer.decmps$random[7:366]),36)
```



Compare the residual autocorrelation plot on page 9, which is for the model 1 regression. Both autocorrelation plots show failure to capture all of the correlation structure in the data.

R's decomposition procedure can also be applied with use of a multiplicative formulation. To finish this study of the U.S. beer data as of now, let's compare estimated seasonal indices from the model 5 regression fit, a multiplicative model, and the multiplicative decomposition procedure in R.

Calculations for the multiplicative decomposition model follow. The R code has an omission—it fails to adjust the monthly seasonal index estimates so that their product is 1. The code which follows adds a line to make this adjustment. (The R code does correctly adjust the estimates from the additive decomposition model so they add to 0.)

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

> seasdmult
 [1] 0.95587 0.91548 1.05153 1.04343 1.11566 1.13030 1.10173 1.07523 0.97263
[10] 0.95662 0.87905 0.85244
```

A table of these two sets of seasonal index estimates from multiplicative model fitting follows. The first column is from the regression fit, and the second column is from the decomposition method in R.

```
> cbind(exp(seas5),seasdmult[1:12])

           exp(seas5)seasmult
Jan         0.95580 0.95587
Feb         0.91651 0.91548
Mar         1.05232 1.05153
Apr         1.04433 1.04343
May         1.11436 1.11566
Jun         1.13092 1.13030
Jul         1.09995 1.10173
Aug         1.07469 1.07523
Sep         0.97361 0.97263
Oct         0.95654 0.95662
Nov         0.87858 0.87905
Dec         0.85218 0.85244
```

These two sets of estimates are very similar.

Summary and additional remarks

1. Several different methods have been used to fit decomposition models to the U.S. beer time series. All of the methods provide estimation of trend and static seasonal structures. The results from the different fits are comparable and lead to similar conclusions. The models seem to estimate trend and seasonal structures reasonably well, but the residuals reveal there is remaining uncaptured trend and correlation structure. That is, the models fail to reduce the data adequately to white noise for this particular time series.

2. Decomposition models have been fit to the data using both regression and the `decompose` command in R. For each of these, both additive and multiplicative structures have been employed.

3. The beer time series shows a gentle upward trend followed by a decrease, a levelling, and then a slight decline. Thus, in the regression models a fourth-degree polynomial in time is used to estimate the trend. In its additive model mode, the `decompose` command in R uses a centered moving average to estimate the trend. This command uses monthly averages after removal of the trend estimate to estimate the seasonal structure. A multiplicative model procedure is also available with `decompose` and is illustrated.

4. In the regression models fit, several different mathematical bases have been used to estimate seasonal indices. These are the month dummies employed for a factor variable in R, an alternative set of month dummies which are self-defined, and the cosine and sine functions with period 12 and their overtones. All of these mathematical bases produce exactly the same seasonal index estimates. (To obtain the same estimates with the cosines and sines as with the two sets of month dummies, one needs to use all 11 of the trigonometric functions.)

5. For the U.S. beer production time series, the months of high production are May, June, and July (11, 13, and 10 percent, respectively, above the level of the trend), and the months of low production are November and December (12 and 15 percent, respectively, below the level of the trend). When the seasonal structure is described by the trigonometric basis, the fundamental component with period 12 months is strongly dominant, but the second through fifth harmonics also contribute.

6. A partial *F* test is used for the hypothesis that some of the variables in a regression model all have zero coefficients. The test statistic may be calculated in R by fitting both a complete model and a reduced model, where the latter omits the variables for which the coefficients are hypothesized to be zero. One uses the `anova` command with two arguments to obtain the *F* statistic and its *p*-value.

Monthly Australian beer production, January 1956—August 1995.

The time series gives monthly production in megaliters, including ale and stout. Beverages with alcoholic content less than 1.15 per cent are not included. The data are in beeraustralia.txt.

The U.S. beer production data show a modest downward trend in recent years, despite increasing population. The website https://www1.racgp.org.au/newsgp/clinical/alcohol-consumption-mostly-holds-steady-in-austral indicates there has been a decline in beer consumption in Australia relative to other alcohol products:

> 'In 2017–18 beer represented 39% of all pure alcohol available for consumption and wine 38.6%... .'
>
> 'This is in stark contrast to 40 years ago when beer represented 67.6% and wine 18.6% of pure alcohol available per person aged 15 years and over, reflecting the change in consumption preferences over time.'

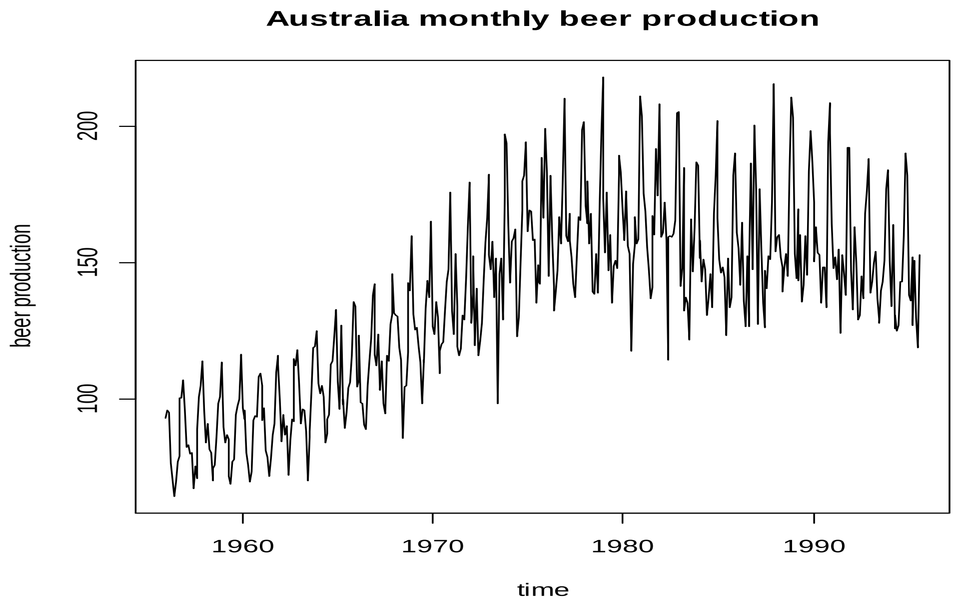
```
> ausbeer<-read.csv("F:/Stat71122Spring/beeraustralia.txt",header=T)
> attach(ausbeer)
> head(ausbeer)
  year month beer    dlogbeer obs317 obs318
1 1956     1 93.2          NA      0      0
2 1956     2 96.0  0.02960047      0      0
3 1956     3 95.2 -0.00836825      0      0
4 1956     4 77.1 -0.21087666      0      0
5 1956     5 70.9 -0.08383285      0      0
6 1956     6 64.8 -0.08996483      0      0

> fmonth<-as.factor(month)
```

The variable *dlogbeer* gives monthly log returns, the monthly differences of the log of the *beer* time series. There are also dummies for two observations—there was abnormally low production in May 1982, and a dummy is required for June 1982 for some of the models to be presented.

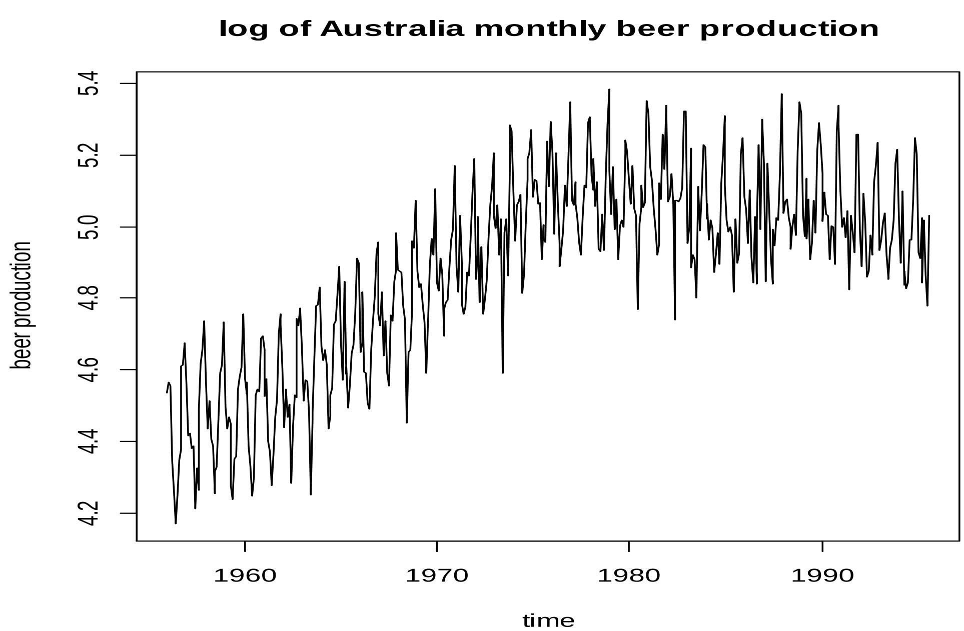
The plot of beer production vs. time follows.

```
> plot(ts(beer,start=c(1956,1),freq=12),xlab="time",ylab="beer
```

```
production",main="Australia monthly beer production")
```



The plot shows a sustained rise in production until the late 1970s. This is followed by a slight decline, a levelling, then a small increase during the 1980s, and then a decline starting in the late 1980s. As the level of the series rises, the fluctuations become more volatile. Thus, we need to build a multiplicative model. Here is the plot of the logged data:



Although the result is not perfect, the log transformation has done a decent job of stabilizing the variance. From visual inspection it appears that there are four turns in the trend. So let's begin by fitting a multiplicative decomposition model with a fifth-degree polynomial trend. We emphasize that the seasonal component used here in our model is deterministic, and static rather than dynamic.

```
> time<-as.numeric(1:length(beer))
```

Note we have converted *time* from an integer variable to a continuous variable.

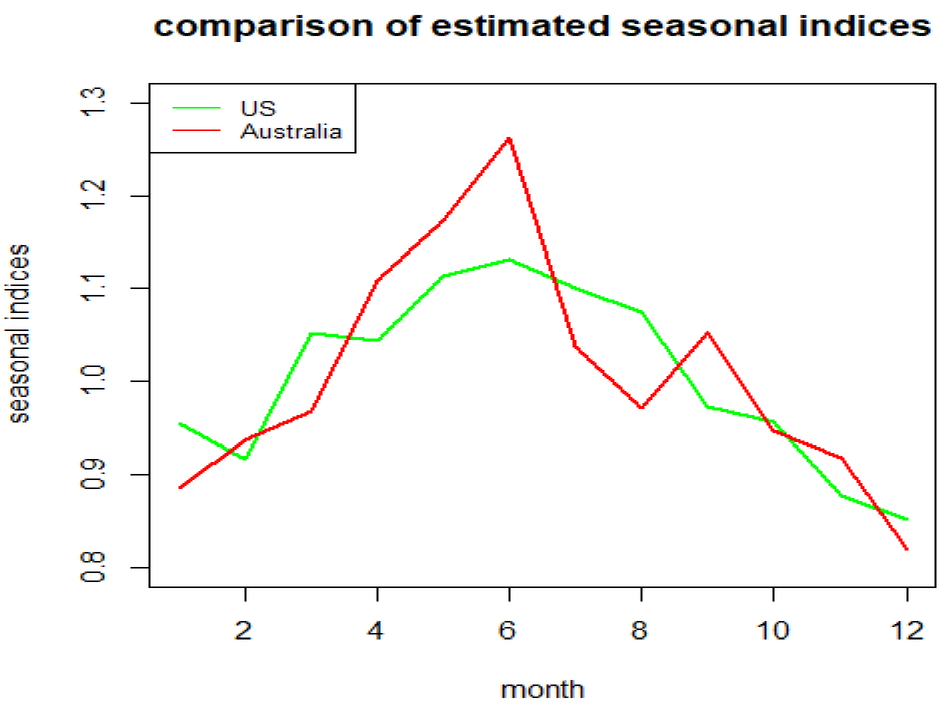
```
> modela1<-lm(log(beer)~poly(time,5)+fmonth);summary(modela1)

Call:
lm(formula = log(beer) ~ poly(time, 5) + fmonth)

Residuals:
      Min        1Q    Median        3Q       Max 
-0.262121 -0.039685  0.003353  0.045376  0.189701 

Coefficients:
               Estimate Std. Error t value Pr(>|t|)    
(Intercept)     4.91985    0.01093 450.289  < 2e-16 ***
poly(time, 5)1  4.22453    0.06909  61.144  < 2e-16 ***
poly(time, 5)2 -2.42219    0.06911 -35.049  < 2e-16 ***
poly(time, 5)3 -0.40301    0.06910  -5.832 1.03e-08 ***
poly(time, 5)4  0.75755    0.06913  10.959  < 2e-16 ***
poly(time, 5)5 -0.21521    0.06911  -3.114  0.00196 ** 
fmonth2        -0.06589    0.01545  -4.265 2.43e-05 ***
fmonth3         0.01501    0.01545   0.971  0.33188    
fmonth4        -0.08997    0.01545  -5.824 1.08e-08 ***
fmonth5        -0.12131    0.01545  -7.852 2.93e-14 ***
fmonth6        -0.23583    0.01545 -15.263  < 2e-16 ***
fmonth7        -0.15681    0.01545 -10.148  < 2e-16 ***
fmonth8        -0.10092    0.01545  -6.530 1.75e-10 ***
fmonth9        -0.06956    0.01555  -4.473 9.72e-06 ***
fmonth10        0.06737    0.01555   4.332 1.81e-05 ***
fmonth11        0.12356    0.01555   7.945 1.51e-14 ***
fmonth12        0.19610    0.01555  12.609  < 2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1 

Residual standard error: 0.06908 on 459 degrees of freedom
Multiple R-squared:  0.9342,    Adjusted R-squared:  0.9319 
F-statistic: 407.1 on 16 and 459 DF,  p-value: < 2.2e-16
```

The fifth-order term in the trend component is statistically significant. The estimated seasonal indices from this model are given next.

```
> b1<-coef(modela1)[1]
> b2<-coef(modela1)[7:17]+b1
> b3<-c(b1,b2)
> seasa1<-exp(b3-mean(b3))

> seasa1
(Intercept)     fmonth2     fmonth3     fmonth4     fmonth5     fmonth6 
  1.0371961   0.9710602   1.0528773   0.9479569   0.9187087   0.8192957 
    fmonth7     fmonth8     fmonth9    fmonth10    fmonth11    fmonth12 
  0.8866612   0.9376337   0.9674993   1.1094780   1.1736037   1.2619042
```

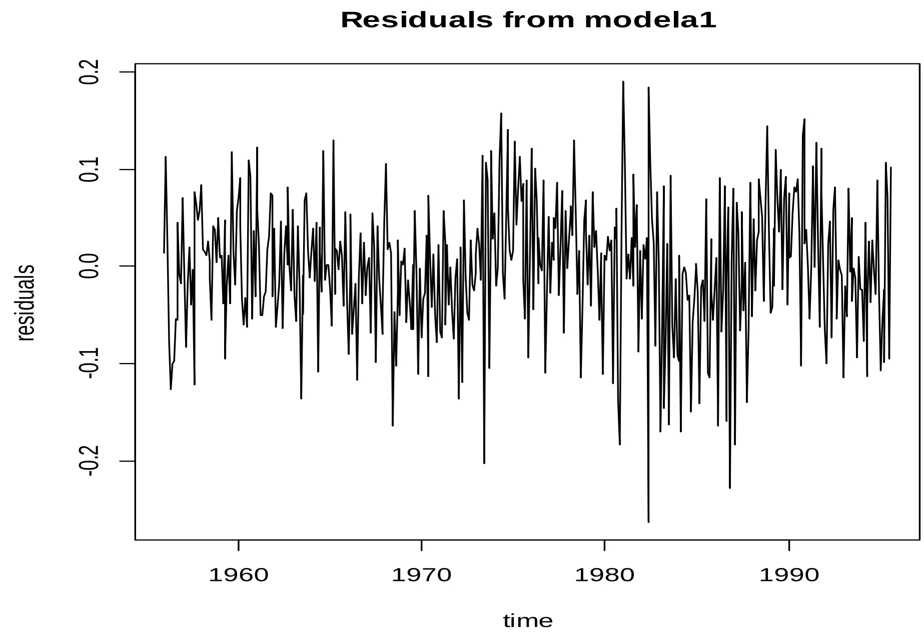
The plot which follows shows these estimated seasonal indices and the estimated seasonal indices for the U.S. beer data from model 5 for that series. In the picture the indices for the Australian data have been shifted by six months.

```
> usseas.ts<-ts(exp(seas5))
> ausseas.ts<-ts(seasa1)
> ausseas.ts<-c(ausseas.ts,ausseas.ts)[7:18]
> plot(usseas.ts,ylim=c(0.8,1.3),xlab="month",ylab="seasonal
indices",main="comparison of estimated seasonal
indices",col="green",lwd=2)
> lines(ausseas.ts,col="red",lwd=2)
>
legend("topleft",legend=c("US","Australia"),col=c("green","red"),lty=1,
cex=0.8)
```



The U.S. and Australian production series do span different stretches of time. The two seasonal patterns are somewhat similar.

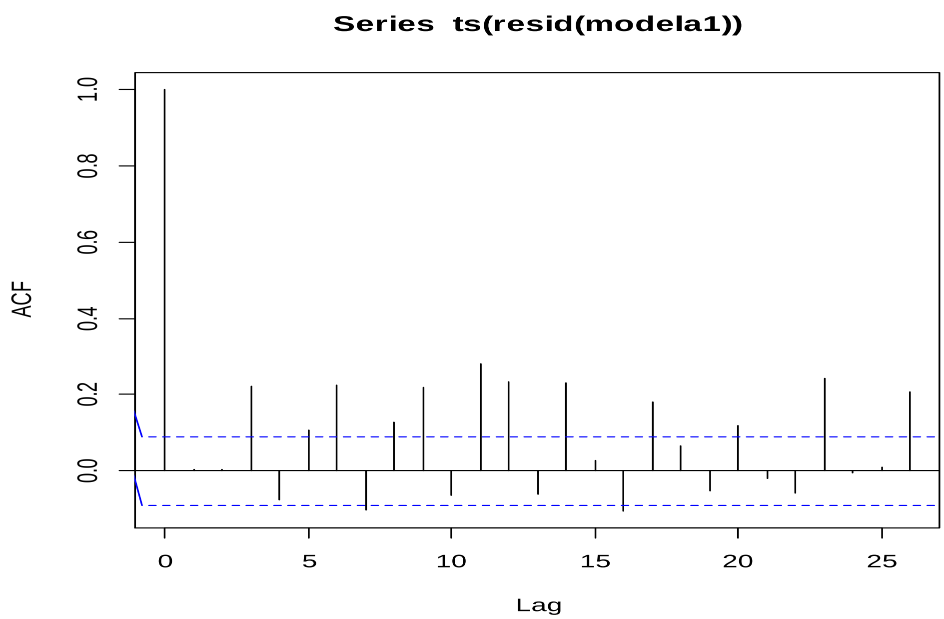
The plot of the residuals for the Australian model follows.



The residual plot shows that the fifth-degree polynomial has failed to track the trend structure adequately. Should we try fitting a higher degree polynomial? Although a high degree polynomial fit may succeed in capturing trend structure, it usually fails to behave reasonably in forecasting future observations. And trying to fit too high a degree will lead to numerical instability in the calculations.

In addition, there is failure by the model to reduce to white noise, as the following residual autocorrelation plot shows.

```
> acf(ts(resid(modela1)))
```

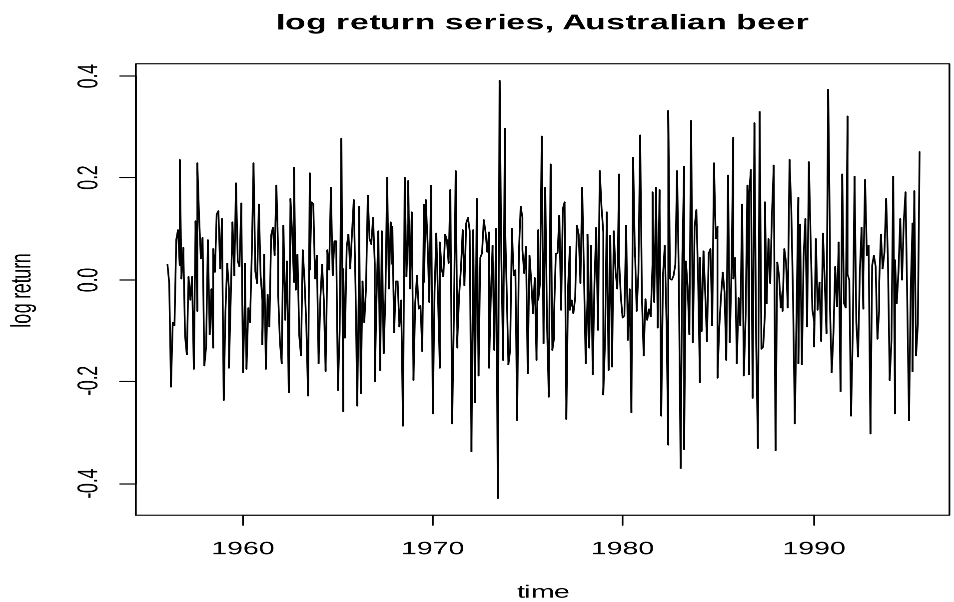


Let's try a different approach. Instead of modeling the log series directly, we consider the monthly changes in the log series, $\log y_t - \log y_{t-1}$, the *log return*. The log return is approximately equal to the percentage change of the y_t series, from one month to the next, for small percentage changes (up to about ± 0.10, that is, ± 10 per cent). Let's review the discussion on page 3 of the 12 January notes. The percentage change is $R_t = \dfrac{y_t - y_{t-1}}{y_{t-1}}$, and the log return is

$$\begin{aligned}\log y_t - \log y_{t-1} &= \log\left(y_t / y_{t-1}\right)\\ &= \log\left(1+R_t\right)\\ &\approx R_t.\end{aligned}$$

The last step follows by taking the first term in the Taylor series expansion of $\log\left(1+R_t\right)$, an approximation that is valid for small values of R_t.

Here is the plot of the log return series.



The log return series has a strong seasonal component, but it has no trend. The log return calculation involves a differencing operation, and it is common with time series that differencing eliminates a trend. The differencing operation does alter the structures of the seasonal and irregular components, though. We will estimate the seasonal indices for the log return series and then modify them to obtain estimates of the seasonal indices for the original production series. Of course, analysis of the log return series does not permit estimation of trend structure for the production series.

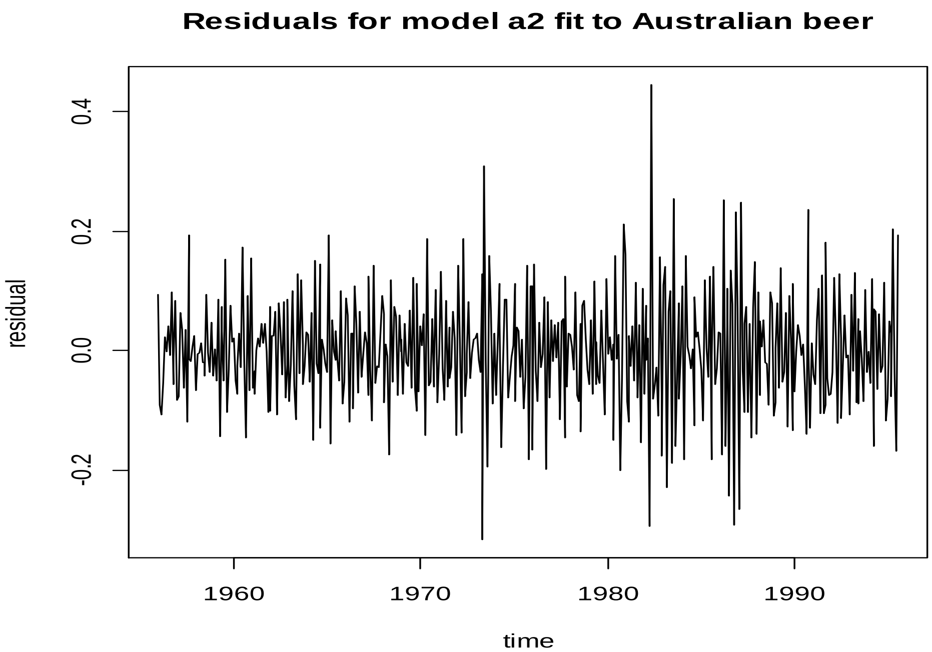
A general result of differencing time series is that fast movements are enhanced and slow movements are attenuated by the differencing operation.

For the log return data there is no statistically significant polynomial trend, and we start by fitting a model with just a seasonal component.

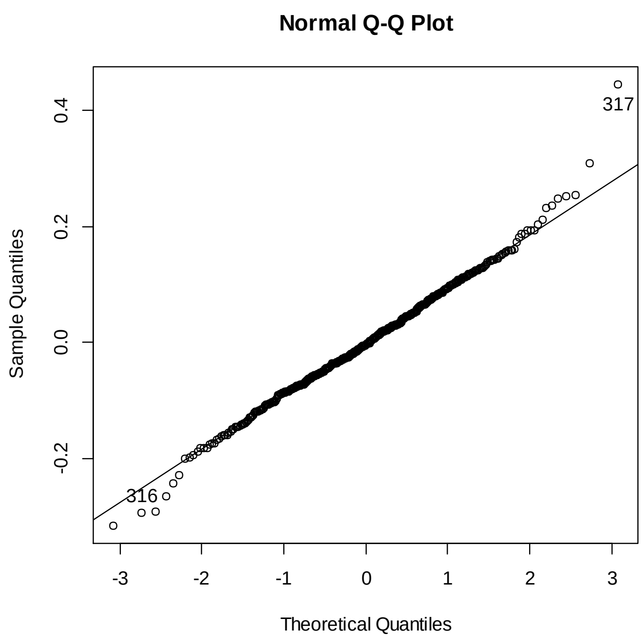
```
> modela2<-lm(dlogbeer~fmonth);summary(modela2)

Call:
lm(formula = dlogbeer ~ fmonth)

Residuals:
     Min       1Q   Median       3Q      Max 
-0.31503 -0.06119 -0.00417  0.06318  0.44470 

Coefficients:
            Estimate Std. Error t value Pr(>|t|)    
(Intercept) -0.19534    0.01554 -12.570  < 2e-16 ***
fmonth2      0.13048    0.02184   5.975 4.60e-09 ***
fmonth3      0.27727    0.02184  12.696  < 2e-16 ***
fmonth4      0.09141    0.02184   4.186 3.41e-05 ***
fmonth5      0.16505    0.02184   7.557 2.22e-13 ***
fmonth6      0.08187    0.02184   3.749    2e-04 ***
fmonth7      0.27542    0.02184  12.611  < 2e-16 ***
fmonth8      0.25230    0.02184  11.553  < 2e-16 ***
fmonth9      0.23041    0.02198  10.484  < 2e-16 ***
fmonth10     0.33339    0.02198  15.170  < 2e-16 ***
fmonth11     0.25265    0.02198  11.496  < 2e-16 ***
fmonth12     0.26901    0.02198  12.240  < 2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1 

Residual standard error: 0.09705 on 463 degrees of freedom
  (1 observation deleted due to missingness)
Multiple R-squared:  0.5037,    Adjusted R-squared:  0.4919 
F-statistic: 42.72 on 11 and 463 DF,  p-value: < 2.2e-16
```

R square for this model is 0.5037, and for the first model, fit to the production data, it is 0.9342. The lower value in the present case has resulted because the differencing operation which produces the log returns has removed the trend component, which was a major factor in raising the value of *R* square for the first model. The lower *R* square value for the present model does not mean it is an inferior model.

Here are the time plot of the residuals for this fit, and the residual normal quantile plot:




```
> qq<-qqnorm(resid(modela2))
> qqline(resid(modela2))
> identify(qq)
```



There is no trend evident in these residuals. There is a large residual value identified at time point 317, which actually corresponds to June 1982. This has resulted because of abnormally low production in May 1982. The affected time point in the residual series is

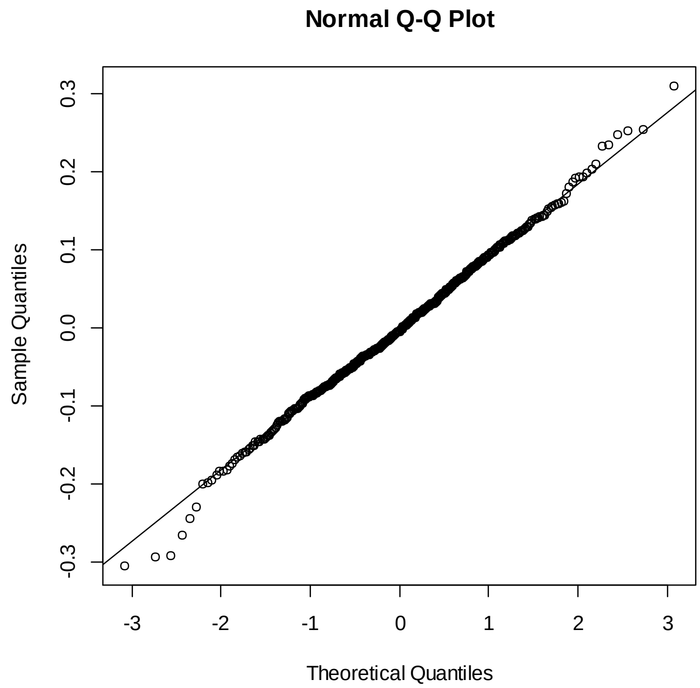
317, rather than 318, because there is no log return value for the first time point at January 1956, and the regression fit spans time points 2 to 476 of the production series.

```
> dlogbeer[316:319]
[1] -0.0829522240 -0.3236756750  0.3312228807  0.0006263702
> beer[316:319]
[1] 158.4 114.6 159.6 159.7
```

Let's add the dummy variable for observation 318 to the model.

```
> modela3<-lm(dlogbeer~fmonth+obs318);summary(modela3)

Call:
lm(formula = dlogbeer ~ fmonth + obs318)

Residuals:
     Min       1Q   Median       3Q      Max 
-0.30362 -0.06030 -0.00362  0.06335  0.31008 

Coefficients:
            Estimate Std. Error t value Pr(>|t|)    
(Intercept) -0.19534    0.01519 -12.859  < 2e-16 ***
fmonth2      0.13048    0.02135   6.112 2.09e-09 ***
fmonth3      0.27727    0.02135  12.988  < 2e-16 ***
fmonth4      0.09141    0.02135   4.282 2.26e-05 ***
fmonth5      0.16505    0.02135   7.731 6.72e-14 ***
fmonth6      0.07047    0.02148   3.280  0.00112 ** 
fmonth7      0.27542    0.02135  12.901  < 2e-16 ***
fmonth8      0.25230    0.02135  11.818  < 2e-16 ***
fmonth9      0.23041    0.02148  10.725  < 2e-16 ***
fmonth10     0.33339    0.02148  15.519  < 2e-16 ***
fmonth11     0.25265    0.02148  11.761  < 2e-16 ***
fmonth12     0.26901    0.02148  12.522  < 2e-16 ***
obs318       0.45610    0.09607   4.747 2.75e-06 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1 

Residual standard error: 0.09487 on 462 degrees of freedom
  (1 observation deleted due to missingness)
Multiple R-squared:  0.5268,    Adjusted R-squared:  0.5145 
F-statistic: 42.86 on 12 and 462 DF,  p-value: < 2.2e-16
```

As the following plot shows, the residuals from this model fit are acceptably fit by a normal distribution.



Calculation of the estimated seasonal indices from a multiplicative model fit to the monthly log return series. We can obtain estimates of the seasonal indices S_t from a multiplicative model fit to the log return series. Recall that the log representation of the multiplicative decomposition model is

$$\log y_t = \log T_t + \log S_t + \log \varepsilon_t .$$

We have monthly data, and the product of the seasonal indices is 1,

(2) $$S_1 S_2 \cdots S_{12} = 1.$$

We note that, in addition, the sum of the logged seasonal indices is 0,

(3) $$\log S_1 + \cdots + \log S_{12} = 0.$$

Also, $S_t = S_{t+12}$ for all t.

If we fit a regression model to the log returns, we work with

(4) $$\log y_t - \log y_{t-1} = \log T_t - \log T_{t-1} + \log S_t - \log S_{t-1} + \log \varepsilon_t - \log \varepsilon_{t-1}.$$

Typically, the differencing will effectively remove the trend component, that is,

$$\log T_t - \log T_{t-1}$$

will essentially be negligible. We have seen that this is the case for the present data set. Then estimation using (4) will not have to contend with a trend component that is difficult to estimate. The formulation (4) thus results in estimation of the values

(5) $$x_t = \log S_t - \log S_{t-1} = \log(S_t / S_{t-1}).$$

Observe that, by (3) and the comment directly below it, the sum of the 12 differences in (5) is equal to 0. Thus, only 11 of the differences are free to vary. Use the last 11 differences in (5) and the restriction (3) to solve for $\log S_{12}$. The solution is

(6) $$\begin{aligned} \log S_{12} &= \left(x_2 + 2x_3 + 3x_4 + \cdots + 11x_{12}\right)/12 \\ &= \left(x_1 + 2x_2 + 3x_3 + \cdots + 12x_{12}\right)/12. \end{aligned}$$

The second equality in (6) follows because $x_1 + x_2 + \cdots + x_{12} = 0$ [see (3) and (5)]. Then $\log S_1, \ldots, \log S_{11}$ are determined via

(7) $$\log S_j = x_1 + \cdots + x_j + \log S_{12}, \quad j = 1, 2, \ldots, 11.$$

Finally, exponentiate the $\log S_j$ values to obtain the estimated indices S_j.

If the data are quarterly with an annual cycle, there are only four seasonal indices, and (3), (5), (6), and (7) become

(3') $$\log S_1 + \cdots + \log S_4 = 0,$$

(5') $$x_1 = \log S_1 - \log S_4,\; x_2 = \log S_2 - \log S_1,\; x_3 = \log S_3 - \log S_2,\; x_4 = \log S_4 - \log S_3,$$

(6') $$\log S_4 = \left(x_2 + 2x_3 + 3x_4\right)/4 = \left(x_1 + 2x_2 + 3x_3 + 4x_4\right)/4.$$

(7') $$\log S_j = x_1 + \cdots + x_j + \log S_4, \quad j = 1, 2, 3.$$

If the data are daily with a weekly cycle, there are seven seasonal indices, and the details are similar.

Let’s apply the methodology described at (2)–(7) to obtain estimated seasonal indices for the production data, using model a3.

```
> b1<-coef(modela3)[1]
> b2<-coef(modela3)[2:12]+b1
> b3<-c(b1,b2)
> x<-b3-mean(b3)
```

This calculation gives the values x_t shown in (5). Next we calculate $\log S_{12}$ given by (6) and $\log S_j$ given by (7).

```
> s12<-0
> for(j in 2:12){
+ xsub<-x[j:12]
+ s12<-s12+sum(xsub)
+ }
> s12<-s12/12
> s<-c(rep(0,times=12))
> s[12]<-s12
> for(j in 1:11){
+ xsub<-x[1:j]
+ s[j]<-s[12]+sum(xsub)
+ }
> s<-exp(s)
> s

 [1] 1.0389676 0.9734147 1.0561923 0.9516341 0.9229497 0.8143480
0.8819633
 [8] 0.9333654 0.9663752 1.1090789 1.1741288 1.2634931
```

Before tabulating the seasonal index estimates, we construct one more model, by adding the dummy for May 1982 to the first model fit to the production time series.

```
> modela4<-lm(log(beer)~poly(time,5)+fmonth+obs317);summary(modela4)

Call:
lm(formula = log(beer) ~ poly(time, 5) + fmonth + obs317)

Residuals:
     Min       1Q   Median       3Q      Max
-0.22884 -0.03905  0.00293  0.04559  0.18785

Coefficients:
               Estimate Std. Error t value Pr(>|t|)
(Intercept)     4.91988    0.01076 457.269  < 2e-16 ***
poly(time, 5)1  4.23158    0.06806  62.174  < 2e-16 ***
poly(time, 5)2 -2.43177    0.06810 -35.710  < 2e-16 ***
poly(time, 5)3 -0.41637    0.06813  -6.111 2.12e-09 ***
poly(time, 5)4  0.75795    0.06807  11.135  < 2e-16 ***
poly(time, 5)5 -0.20150    0.06815  -2.957 0.003270 **
fmonth2        -0.06589    0.01521  -4.332 1.82e-05 ***
fmonth3         0.01500    0.01521   0.986 0.324786
fmonth4        -0.08998    0.01521  -5.915 6.51e-09 ***
fmonth5        -0.11456    0.01531  -7.482 3.78e-13 ***
fmonth6        -0.23586    0.01522 -15.501  < 2e-16 ***
fmonth7        -0.15684    0.01522 -10.307  < 2e-16 ***
fmonth8        -0.10095    0.01522  -6.633 9.27e-11 ***
fmonth9        -0.06960    0.01531  -4.545 7.03e-06 ***
```

```
fmonth10        0.06733    0.01531   4.397 1.37e-05 ***
fmonth11        0.12352    0.01531   8.066 6.44e-15 ***
fmonth12        0.19606    0.01531  12.802  < 2e-16 ***
obs317         -0.27077    0.06914  -3.916 0.000104 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.06803 on 458 degrees of freedom
Multiple R-squared:  0.9363,    Adjusted R-squared:  0.9339
F-statistic: 396.1 on 17 and 458 DF,  p-value: < 2.2e-16

> b1<-coef(modela4)[1]
> b2<-coef(modela4)[7:17]+b1
> b3<-c(b1,b2)
> seasa4<-exp(b3-mean(b3))

> seasa4
(Intercept)     fmonth2     fmonth3     fmonth4     fmonth5     fmonth6
  1.0366376   0.9705322   1.0522995   0.9474319   0.9244319   0.8188339
    fmonth7     fmonth8     fmonth9    fmonth10    fmonth11    fmonth12
  0.8861571   0.9370959   0.9669427   1.1088366   1.1729219   1.2611676
```

The following table compares seasonal index estimates from three models. The first column gives estimates for model a1, with log beer as the response and a fifth-degree polynomial trend. It does not include a dummy to adjust for the outlier in May 1982. In the second column, estimates are given for model a3 fit to the log return data, with the dummy for June 1982 included. The last column of the table has estimates for model a4, obtained by adding the dummy for May 1982 to model a1.

```
> options(digits=4)
> cbind(seasa1,s,seasa4)

            seasa1 s(a3)  seasa4
Jan         1.0372 1.0390 1.0366
Feb         0.9711 0.9734 0.9705
Mar         1.0529 1.0562 1.0523
Apr         0.9480 0.9516 0.9474
May         0.9187 0.9229 0.9244
Jun         0.8193 0.8143 0.8188
Jul         0.8867 0.8820 0.8862
Aug         0.9376 0.9334 0.9371
Sep         0.9675 0.9664 0.9669
Oct         1.1095 1.1091 1.1088
Nov         1.1736 1.1741 1.1729
Dec         1.2619 1.2635 1.2612
```

The three estimations have produced very similar results. As expected, the first and third columns essentially differ only for May, and the difference is very small—the adjustment made by the dummy variable for the May 1982 outlier is for only one out of 476 months. Given the seasonal index estimates from analysis of the log return data, we can conclude that failure to estimate the trend in the models applied to the production data does not hamper our ability to estimate seasonal structure.

Calculation of the estimated seasonal indices from a model fit to the monthly differenced series. The discussion above addresses estimation of seasonal indices in a decomposition model when one analyzes log return data and calculation of the log returns removes trend structure. Of course, log returns are differences of the logged time series. In some cases, one analyzes differences of unlogged data. When one does this, one is differencing an additive decomposition model, rather than the logs of a multiplicative decomposition model. Calculation of the estimated seasonal indices in this case is the same as that shown above for analysis of the log return data, with the exception that there is no exponentiation at the very end.

Recall that the additive decomposition model is

$$y_t = T_t + S_t + \varepsilon_t.$$

We have monthly data, and the sum of the seasonal indices is 0,

(8) $$S_1 + S_2 + \cdots + S_{12} = 0.$$

Also, $S_t = S_{t+12}$ for all t.

If we fit a regression model to the differences, we work with

(9) $$y_t - y_{t-1} = T_t - T_{t-1} + S_t - S_{t-1} + \varepsilon_t - \varepsilon_{t-1}.$$

Typically, the differencing will effectively remove the trend component, that is,

$$T_t - T_{t-1}$$

will essentially be negligible. Then estimation using (9) will not have to contend with a trend component that is difficult to estimate. The formulation (9) thus results in estimation of the differences

(10) $$x_t = S_t - S_{t-1}.$$

Observe that, by (8), the sum of the 12 differences in (10) is equal to 0. Thus, only 11 of the differences are free to vary. Use the last 11 differences in (10) and the restriction (8) to solve for S_{12}. The solution is

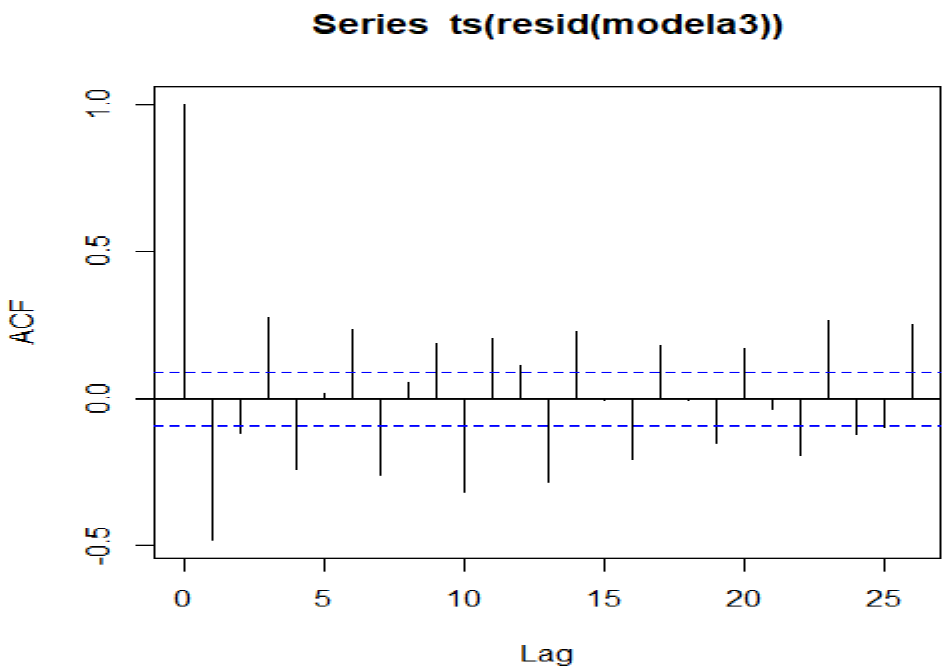
(11) $$\begin{aligned} S_{12} &= \left(x_2 + 2x_3 + 3x_4 + \cdots + 11x_{12}\right)/12 \\ &= \left(x_1 + 2x_2 + 3x_3 + \cdots + 12x_{12}\right)/12. \end{aligned}$$

The second equality in (11) follows because $x_1 + x_2 + \cdots + x_{12} = 0$ [see (8)]. Then $S_1, \ldots, S_{11}$ are determined from (8) via

(12) $$S_j = x_1 + \cdots + x_j + S_{12}, \quad j = 1, 2, \ldots, 11.$$

Let's examine the residual autocorrelations for model a3, fit to the log return data with monthly dummies and the dummy for June 1982.

```
> acf(ts(resid(modela3)))
```



There is a good deal of significant autocorrelation structure remaining in the residual series. Let's try to remedy this. To do so, we refit the log of the series (not the log return series) with a higher degree polynomial trend, the dummy for May 1982, and two trigonometric components designed to capture calendar effects. The calendar effects take into account the varying lengths of the months, leap years, and the varying number of trading days from month to month. The components are cosine-sine pairs with frequencies 0.348 and 0.432 cycles per month. Later in the course I'll discuss this in more detail. In the meantime, you may want to scan the article by Cleveland and Devlin.

```
> c348<-cos(0.696*pi*time);s348<-sin(0.696*pi*time)
> c432<-cos(0.864*pi*time);s432<-sin(0.864*pi*time)

> modela5<-
lm(log(beer)~poly(time,5)+fmonth+obs317+c348+s348+c432+s432);summary(mo
dela5)
```

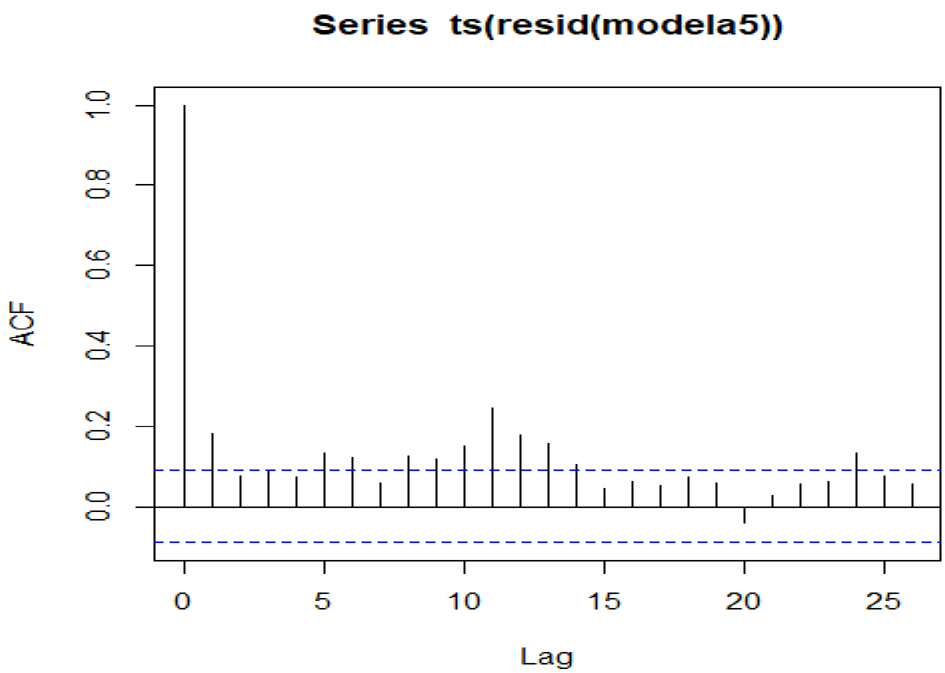
```
Call:
lm(formula = log(beer) ~ poly(time, 5) + fmonth + obs317 + c348 +
    s348 + c432 + s432)

Residuals:
     Min       1Q   Median       3Q      Max
-0.19857 -0.03784 -0.00056  0.04013  0.16549

Coefficients:
                Estimate Std. Error t value Pr(>|t|)
(Intercept)     4.919578   0.009674 508.526  < 2e-16 ***
poly(time, 5)1  4.229539   0.061188  69.123  < 2e-16 ***
poly(time, 5)2 -2.433759   0.061223 -39.752  < 2e-16 ***
poly(time, 5)3 -0.416275   0.061254  -6.796 3.39e-11 ***
poly(time, 5)4  0.753521   0.061201  12.312  < 2e-16 ***
poly(time, 5)5 -0.205500   0.061271  -3.354 0.000863 ***
fmonth2        -0.065289   0.013680  -4.773 2.46e-06 ***
fmonth3         0.014882   0.013677   1.088 0.277113
fmonth4        -0.089159   0.013679  -6.518 1.90e-10 ***
fmonth5        -0.115605   0.013768  -8.397 5.91e-16 ***
fmonth6        -0.235403   0.013680 -17.208  < 2e-16 ***
fmonth7        -0.156230   0.013684 -11.417  < 2e-16 ***
fmonth8        -0.101274   0.013682  -7.402 6.56e-13 ***
fmonth9        -0.069349   0.013769  -5.037 6.84e-07 ***
fmonth10        0.066975   0.013768   4.864 1.59e-06 ***
fmonth11        0.124737   0.013769   9.059  < 2e-16 ***
fmonth12        0.195827   0.013771  14.221  < 2e-16 ***
obs317         -0.234493   0.062437  -3.756 0.000195 ***
c348           -0.022170   0.003971  -5.584 4.06e-08 ***
s348           -0.033381   0.003968  -8.412 5.26e-16 ***
c432           -0.012030   0.003982  -3.021 0.002659 **
s432            0.005248   0.003962   1.325 0.185926
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.06116 on 454 degrees of freedom
Multiple R-squared:  0.949,     Adjusted R-squared:  0.9466
F-statistic: 402.1 on 21 and 454 DF,  p-value: < 2.2e-16
```

All of the components included in this model are significant.

Model a1, which contains a fifth-degree polynomial for the trend and month dummies, has an R square value equal to 0.9342. Model a4, which adds the dummy for May 1982, has R square equal to 0.9363. And model a5, which further adds two cosine-sine pairs for calendar effects, has R square equal to 0.9490. In addition, the monthly seasonal index estimates shown here are similar to the estimates obtained in models a1 and a4.

Let's look at the residual correlations for model a5.

```
> acf(ts(resid(modela5)))
```



There is remaining autocorrelation structure, but it is much less prominent than that shown for the residuals from model a1 (see page 28). It is evident that addition of the calendar effect components has greatly improved the fit.

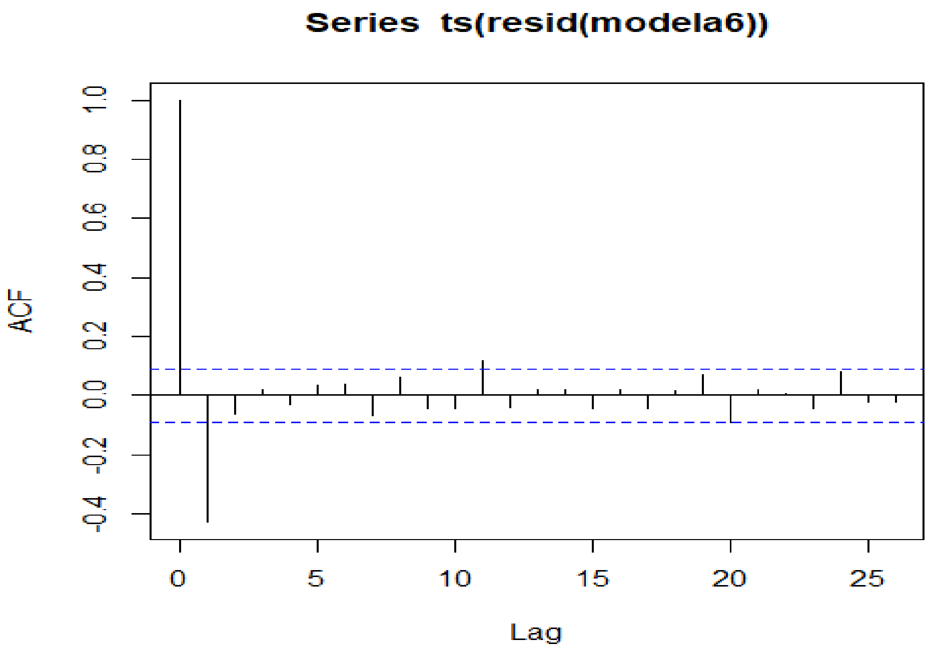
Let's fit still one more model, adding the trigonometric pairs for the calendar effects to the model fit to the log return series, model a3.

```
> modela6<-
lm(dlogbeer~fmonth+obs318+c348+s348+c432+s432);summary(modela6)

Call:
lm(formula = dlogbeer ~ fmonth + obs318 + c348 + s348 + c432 +
    s432)

Residuals:
      Min        1Q    Median        3Q       Max
-0.260048 -0.048172 -0.005019  0.051623  0.270023
```

```
Coefficients:
             Estimate Std. Error t value Pr(>|t|)    
(Intercept) -0.195185   0.012579 -15.517  < 2e-16 ***
fmonth2      0.130935   0.017679   7.406 6.30e-13 ***
fmonth3      0.276374   0.017678  15.634  < 2e-16 ***
fmonth4      0.092194   0.017678   5.215 2.79e-07 ***
fmonth5      0.163920   0.017680   9.272  < 2e-16 ***
fmonth6      0.073138   0.017791   4.111 4.67e-05 ***
fmonth7      0.275426   0.017680  15.578  < 2e-16 ***
fmonth8      0.251191   0.017677  14.210  < 2e-16 ***
fmonth9      0.229766   0.017790  12.915  < 2e-16 ***
fmonth10     0.332600   0.017789  18.697  < 2e-16 ***
fmonth11     0.254064   0.017789  14.282  < 2e-16 ***
fmonth12     0.267415   0.017791  15.031  < 2e-16 ***
obs318       0.366147   0.079903   4.582 5.94e-06 ***
c348        -0.061239   0.005104 -11.998  < 2e-16 ***
s348        -0.035055   0.005102  -6.871 2.09e-11 ***
c432        -0.021052   0.005118  -4.114 4.62e-05 ***
s432         0.015057   0.005094   2.956  0.00328 ** 
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1 

Residual standard error: 0.07855 on 458 degrees of freedom
  (1 observation deleted due to missingness)
Multiple R-squared:  0.6784,    Adjusted R-squared:  0.6671 
F-statistic: 60.37 on 16 and 458 DF,  p-value: < 2.2e-16
```

The calendar effect is strong, and the seasonal index estimates are essentially the same as those obtained with model a3. Moreover, *R* square has increased by 15 percent relative to model a3! It is 0.5268 with model a3, but 0.6784 with the present model.

The residual correlation plot is interesting.



First, compare this plot to the residual correlation plot for model a3 (page 38). The latter has many significant residual correlations, and the current plot, for model a6, is almost clean except for the very significant lag 1 residual correlation. The difference between the two models is the addition of the two calendar trigonometric pairs in model a6.

Second, what about the very significant lag 1 residual autocorrelation? It is not a feature of the data—it is an artifact introduced by the differencing operation. Recall that the multiplicative model for trend, seasonal and irregular components is logged to give

$$\log y_t = \log T_t + \log S_t + \log \varepsilon_t = T_t' + S_t' + \varepsilon_t'.$$

The differencing operation then removes the trend T_t', changes the seasonal component to $S_t' - S_{t-1}'$, and changes the irregular component to $\varepsilon_t' - \varepsilon_{t-1}'$. The residual autocorrelation plot above indicates that the irregular component ε_t' is white noise and that the differencing operation has converted it to a first-order moving average process, which is characterized by a nonzero lag-one autocorrelation. Later in the course we will study such a process.

To aid in following the details of the analysis of the Australian beer data, next is a listing of the forms of the six models which have been fit.

Model a1

logbeer vs. 1,…, t^5, *fmonth*   A residual acf plot is on page 28.

Model a2

dlogbeer vs. *fmonth*

Model a3

dlogbeer vs. *fmonth*, *obs318*   A residual acf plot is on page 38.

Model a4

logbeer vs. 1,…, t^5, *fmonth*, *obs317*

Model a5

logbeer vs. 1,…, t^5, *fmonth*, *obs317*, *c348*, *s348*, *c432*, *s432*
A residual acf plot is on page 40.

Model a6

dlogbeer vs. *fmonth*, *obs318*, *c348*, *s348*, *c432*, *s432*
A residual acf plot is on page 42.

Revisiting the American beer data analysis. There are significant calendar components for the American beer production series. We add the two calendar trigonometric pairs to model 1 (page 3).

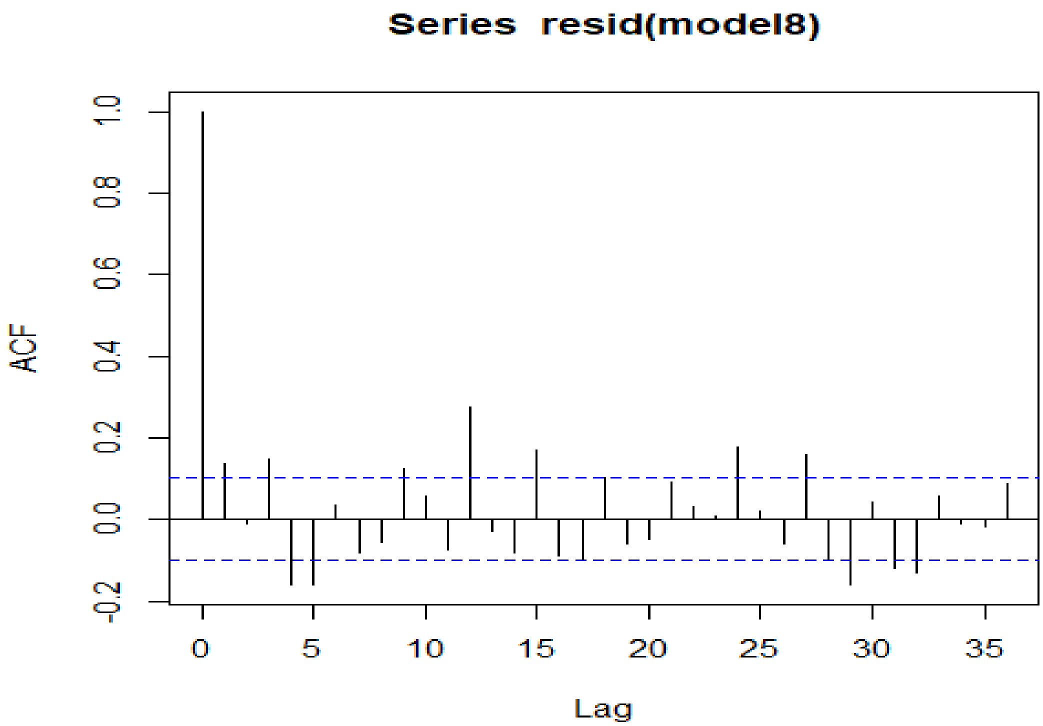
```
> model8<-
lm(beer~poly(time,4)+fmonth+c348+s348+c432+s432);summary(model8)

Call:
lm(formula = beer ~ poly(time, 4) + fmonth + c348 + s348 + c432 +
    s432)

Residuals:
   Min     1Q Median     3Q    Max
-1.412 -0.258  0.001  0.226  1.349

Coefficients:
               Estimate Std. Error t value Pr(>|t|)
(Intercept)     15.6661     0.0775  202.11  < 2e-16 ***
poly(time, 4)1  -4.8925     0.4314  -11.34  < 2e-16 ***
poly(time, 4)2  -2.6455     0.4312   -6.13  2.3e-09 ***
poly(time, 4)3  -0.1074     0.4317   -0.25  0.80370
poly(time, 4)4  -1.5382     0.4312   -3.57  0.00041 ***
fmonth2         -0.6442     0.1096   -5.88  9.7e-09 ***
```

```
fmonth3          1.5690     0.1096   14.32  < 2e-16 ***
fmonth4          1.4448     0.1096   13.19  < 2e-16 ***
fmonth5          2.5970     0.1096   23.69  < 2e-16 ***
fmonth6          2.8469     0.1096   25.98  < 2e-16 ***
fmonth7          2.3680     0.1096   21.61  < 2e-16 ***
fmonth8          1.9449     0.1096   17.74  < 2e-16 ***
fmonth9          0.2713     0.1096    2.47  0.01381 *
fmonth10         0.0166     0.1096    0.15  0.87957
fmonth11        -1.2745     0.1097  -11.62  < 2e-16 ***
fmonth12        -1.7055     0.1096  -15.56  < 2e-16 ***
c348             0.0725     0.0317    2.29  0.02261 *
s348             0.1258     0.0317    3.97  8.7e-05 ***
c432             0.0820     0.0317    2.58  0.01015 *
s432            -0.0234     0.0316   -0.74  0.45949
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.431 on 352 degrees of freedom
Multiple R-squared:  0.929,     Adjusted R-squared:  0.925
F-statistic:  244 on 19 and 352 DF,  p-value: <2e-16

> anova(model1,model8)
Analysis of Variance Table

Model 1: beer ~ time + I(time^2) + I(time^3) + I(time^4) + fmonth
Model 2: beer ~ poly(time, 4) + fmonth + c348 + s348 + c432 + s432
  Res.Df  RSS Df Sum of Sq    F  Pr(>F)
1    356 70.7
2    352 65.4  4      5.22 7.02 1.9e-05 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

The two calendar trigonometric pairs are significant—the partial F test gives a p-value equal to 0.00002.

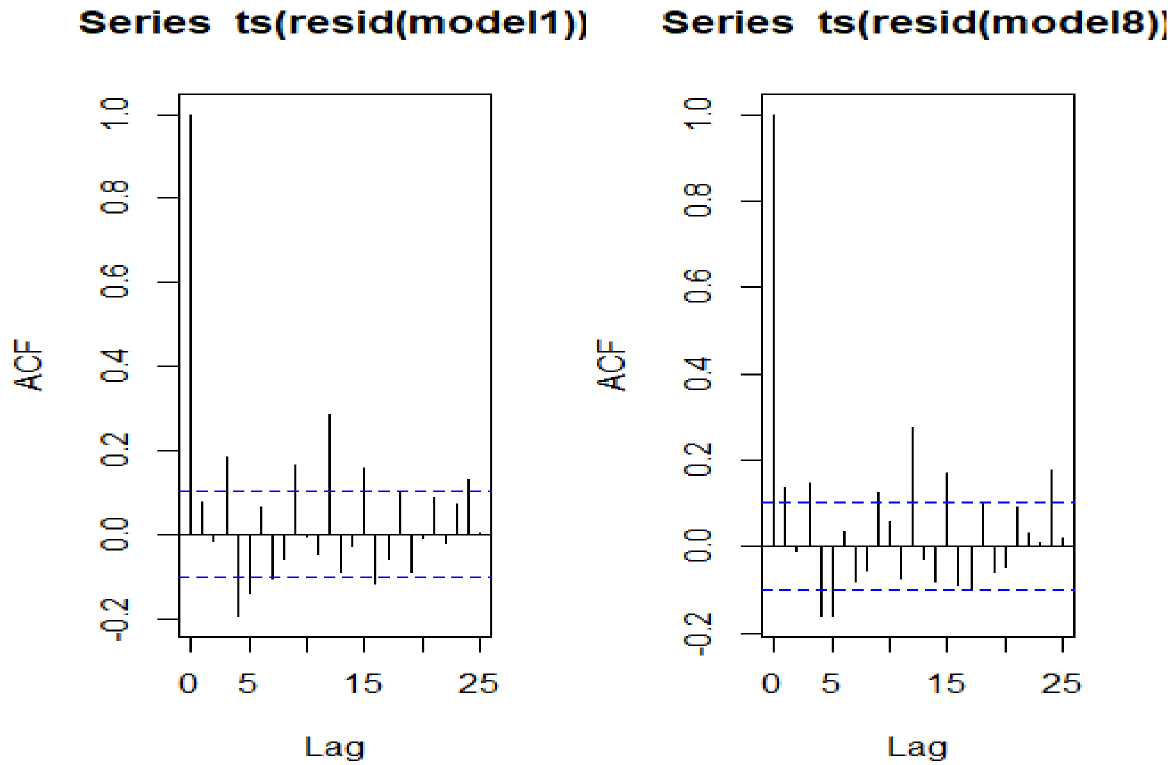
For model 1 the R square value is 0.924, and the residual standard error is 0.446. The corresponding figures for model 8 are 0.929 and 0.431. Although the calendar effects are very significant, there is not much difference between the two models.

Let's compare the residual autocorrelations for the two models. The plot for model 8 follows.



To aid in comparison, let's put the two residual acf plots side-by-side.

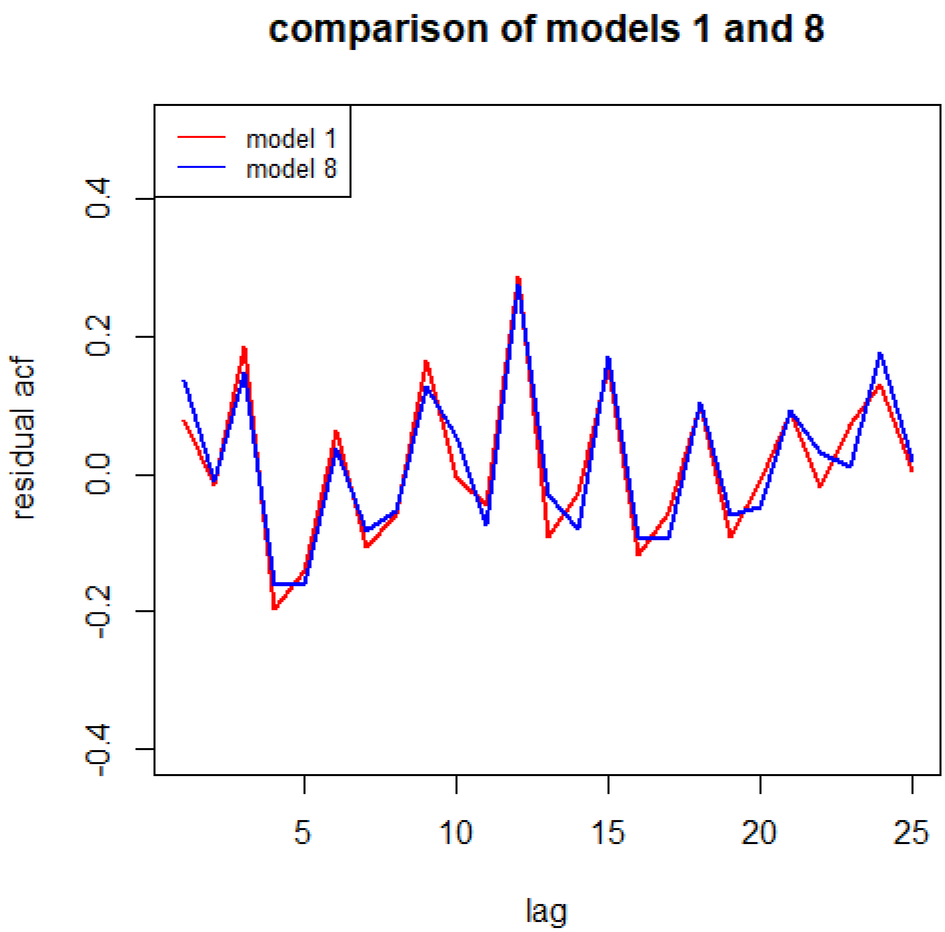
```
> par(mfrow=c(1,2))
> acf(ts(resid(model1)))
> acf(ts(resid(model8)))
```



The plot on the right is a little better, but there is not much difference between the two. Let’s compare the numerical values in a table and a plot.

```
> acf1<-acf(resid(model1),plot=F)
> acf8<-acf(resid(model8),plot=F)

> cbind(acf1$acf,acf8$acf)

  lag  model 1  model 8

    0  1.00000  1.00000
    1  0.07663  0.13552
    2 -0.01688 -0.00989
    3  0.18492  0.14696
    4 -0.19483 -0.16123
    5 -0.14189 -0.16007
    6  0.06343  0.03610
    7 -0.10614 -0.08327
    8 -0.05799 -0.05420
    9  0.16570  0.12577
   10 -0.00505  0.05536
   11 -0.04570 -0.07275
   12  0.28607  0.27382
   13 -0.09084 -0.03041
   14 -0.02744 -0.08018
   15  0.15613  0.17137
   16 -0.11606 -0.09032
   17 -0.05763 -0.09490
   18  0.09944  0.10350
   19 -0.09156 -0.06014
   20 -0.01054 -0.04882
   21  0.08784  0.09052
   22 -0.01865  0.03201
   23  0.07268  0.00920
   24  0.12908  0.17688
   25  0.00333  0.02049
```

```
> autocorr1.ts<-ts(acf1$acf[2:26])
> autocorr8.ts<-ts(acf8$acf[2:26])
> plot(autocorr1.ts,ylim=c(-0.4,0.5),xlab="lag",ylab="residual
acf",main="comparison of models 1 and 8",col="red",lwd=2)
> lines(autocorr8.ts,col="blue",lwd=2)
> legend("topleft",legend=c("model 1","model
8"),col=c("red","blue"),lty=1,cex=0.8)
```



Summary and additional remarks

1. The Australian series and the U.S. series cover different stretches of time, with ten years of overlap. The Australian series has a slightly more complicated trend structure than does the U.S. series. In addition, the variance of the Australian series increases as the level increases, and thus it is necessary to log the response, that is, to use multiplicative decomposition models for estimation.

2. A multiplicative model with a fifth-degree polynomial trend and seasonal structure does not adequately estimate the trend. However, it does provide good estimation of the (static) seasonal indices. This is a relatively common occurrence. The trend addresses

very low frequency movement, and the seasonal component deals with faster fluctuations. The inability to estimate structure in one frequency band does not preclude decent estimation of structure in other (nonoverlapping) frequency bands.

3. There are some rather small differences between the U.S. seasonal index estimates and the Australian seasonal index estimates, but the two patterns are very similar (after translating one set of index estimates by six months).

4. One model fit to the Australian beer production data uses as the response the log return data, that is, the monthly differences of the log of the response. This arises from a multiplicative decomposition model. The response is logged, and then monthly changes are calculated. The log return values are approximately equal to monthly percentage changes. For many time series, the differencing operation eliminates the trend, and this does occur for the Australian beer data. One purpose of this differencing approach is to avoid the difficulties in estimating the trend and focus on estimation of the seasonal indices. The differencing operation does alter the structure of the time series. In particular, it attenuates slow movements and enhances fast movements. And, also, fitting a model to the differences of the logs leads directly to estimation of the monthly differences of the logs of the seasonal indices. As shown, however, estimates of the correct seasonal indices of the multiplicative decomposition structure can be determined from estimates of the monthly differences of the logs of the seasonal indices. For the Australian beer data we find that the estimated seasonal indices obtained from the log return data are virtually identical to the estimated seasonal indices obtained from the log production data.

5. In some cases an additive decomposition structure is employed for a monthly time series and a model is fit to the monthly differences. The differencing operation usually eliminates the trend, but does alter the structure of the time series. Fitting a model to the monthly differences leads directly to estimation of the monthly differences of the seasonal indices. Estimates of the correct seasonal indices of the additive decomposition structure can be obtained from these.

6. Residual analysis for several initial models fit to the Australian beer data shows considerable remaining autocorrelation structure, that is, failure to reduce to white noise. Substantial improvement is obtained by adding dummy calendar trigonometric pairs with frequencies 0.348 and 0.432.

7. Revisiting the U.S. beer data, we find that the calendar trigonometric pairs with frequencies 0.348 and 0.432 are both significant additions to the additive decomposition model fit.

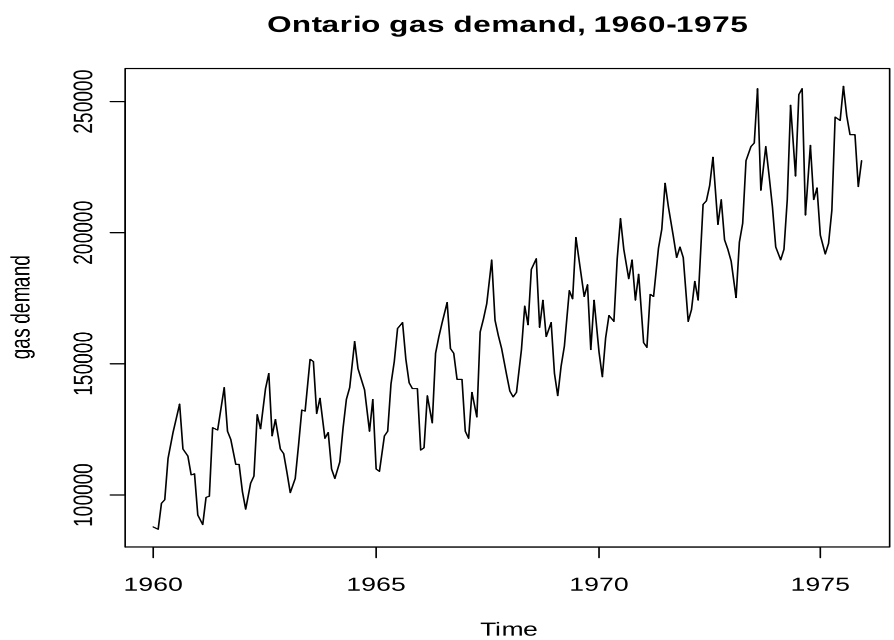
Monthly Ontario gasoline demand, January 1960—December 1975.

This time series gives monthly demand in millions of imperial gallons. An imperial gallon is 4.546 liters, whereas a U.S. gallon is 3.785 liters.

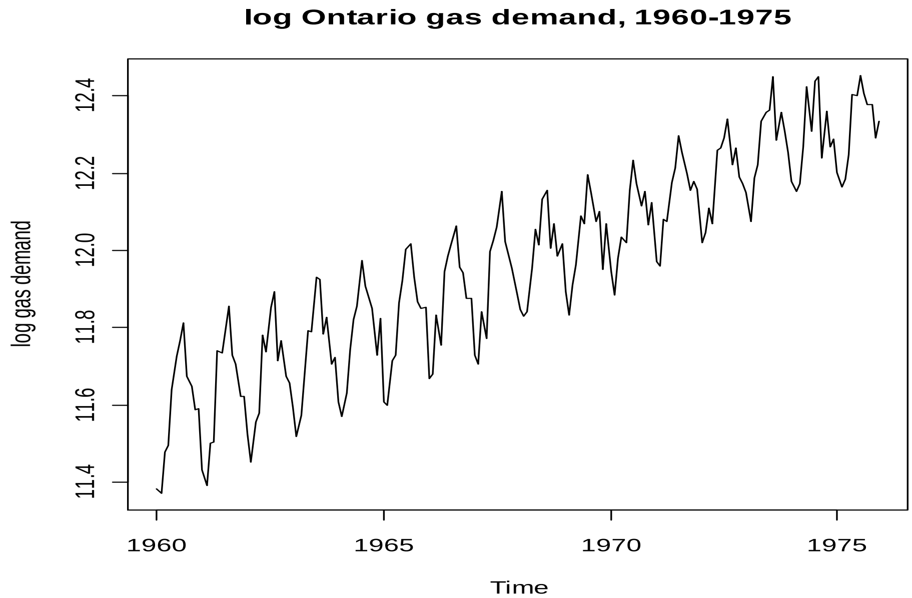
```
> ontgas<-read.csv("F:/Stat71122Spring/Ontariogasdemand.txt",header=T)
> attach(ontgas)
> head(ontgas)
  gasdemand loggasdemand year month obs61 obs125 obs177         c348       s348
1     87695     11.38162 1960     1     0      0      0 -0.57757270  0.8163393
2     86890     11.37240 1960     2     0      0      0 -0.33281954 -0.9429905
3     96442     11.47670 1960     3     0      0      0  0.96202767  0.2729519
4     98133     11.49408 1960     4     0      0      0 -0.77846230  0.6276914
5    113615     11.64057 1960     5     0      0      0 -0.06279052 -0.9980267
6    123924     11.72742 1960     6     0      0      0  0.85099448  0.5251746
```

Let's first plot the demand data.

```
> ontariogas.ts<-ts(ontgas[,1],start=c(1960,1),freq=12)
> plot(ontariogas.ts,ylab="gas demand",main="Ontario gas demand, 1960-
1975")
```



There are an upward trend and a strong seasonal component. The plot of the logged demand data follows.



The two plots have similar appearance. Each shows some changing volatility. In the first plot, volatility increases somewhat over time, and in the second plot, it decreases slightly over time. Perhaps the log transformation is a slight overadjustment. It's not clear from these plots whether the log transformation is needed to analyze the data. Let's start by fitting a model to the demand data.

```
> time<-as.numeric(1:length(gasdemand))
> class(time)
[1] "numeric"
```

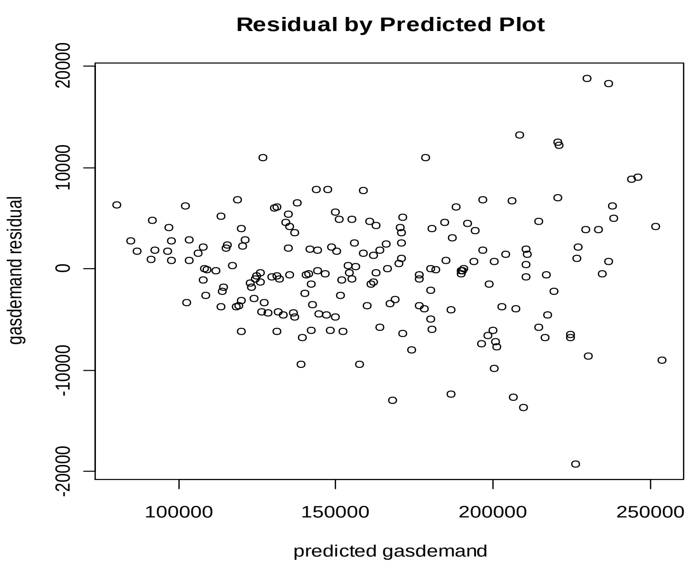
The command `1:length(gasdemand)` creates an integer variable in R, and we have changed it to a continuous variable.

```
> fmonth<-as.factor(month)
> model1<-lm(gasdemand~poly(time,4)+obs125+fmonth);summary(model1)

Call:
lm(formula = gasdemand ~ poly(time, 4) + obs125 + fmonth)

Residuals:
     Min       1Q   Median       3Q      Max
-19237.5  -3608.8   -134.2   3181.8  18778.6
```

```
Coefficients:
                 Estimate Std. Error t value Pr(>|t|)    
(Intercept)        140924       1421  99.163  < 2e-16 ***
poly(time, 4)1     516032       5692  90.656  < 2e-16 ***
poly(time, 4)2      57343       5687  10.083  < 2e-16 ***
poly(time, 4)3     -10939       5717  -1.913  0.05733 .  
poly(time, 4)4     -14312       5678  -2.520  0.01261 *  
obs125             -23616       5895  -4.006 9.11e-05 ***
fmonth2             -4961       2007  -2.472  0.01439 *  
fmonth3              5629       2007   2.805  0.00561 ** 
fmonth4              9811       2007   4.887 2.30e-06 ***
fmonth5             32668       2042  16.000  < 2e-16 ***
fmonth6             32266       2008  16.067  < 2e-16 ***
fmonth7             45213       2009  22.509  < 2e-16 ***
fmonth8             46642       2009  23.213  < 2e-16 ***
fmonth9             26062       2010  12.967  < 2e-16 ***
fmonth10            28640       2011  14.244  < 2e-16 ***
fmonth11            15855       2012   7.882 3.30e-13 ***
fmonth12            17324       2012   8.608 4.17e-15 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 5676 on 175 degrees of freedom
Multiple R-squared: 0.983,      Adjusted R-squared: 0.9814
F-statistic: 632.1 on 16 and 175 DF,  p-value: < 2.2e-16
```

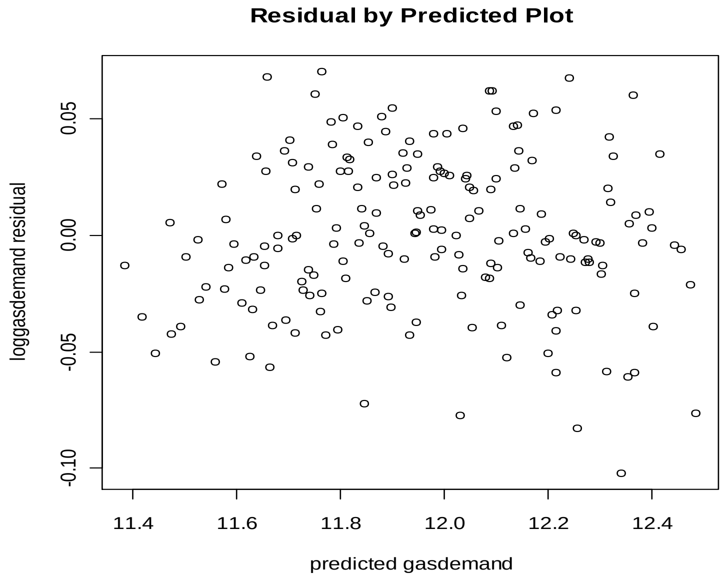
The following plot shows the residuals vs. model predicted values.

```
> plot(predict(model1),resid(model1),xlab="predicted
gasdemand",ylab="gasdemand residual",main="Residual by Predicted Plot")
```



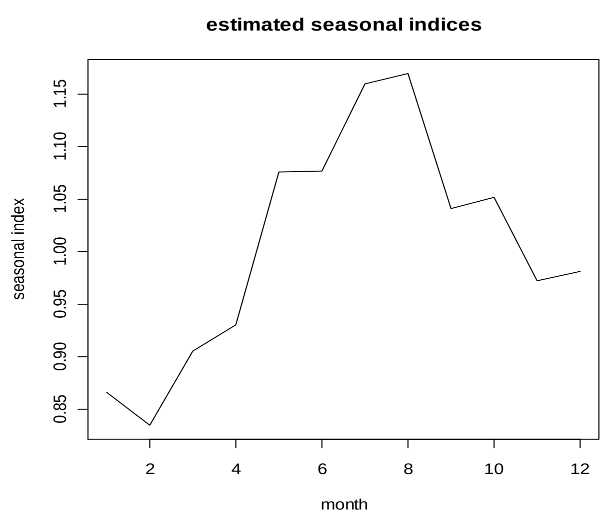
The residual plot shows the presence of heteroscedasticity (note the megaphone shape)—the residual variance increases as the predicted demand value increases. This indicates that analysis should be performed on the logged data—a multiplicative model is needed.

```
> model2<-lm(loggasdemand~poly(time,4)+obs125+fmonth);summary(model2)

Call:
lm(formula = loggasdemand ~ poly(time, 4) + obs125 + fmonth)

Residuals:
      Min        1Q    Median        3Q       Max 
-0.101949 -0.022296 -0.001644  0.025653  0.070251 

Coefficients:
                Estimate Std. Error  t value Pr(>|t|)    
(Intercept)    11.819466   0.008662 1364.493  < 2e-16 ***
poly(time, 4)1  3.231952   0.034695   93.152  < 2e-16 ***
poly(time, 4)2  0.015048   0.034665    0.434 0.664746    
poly(time, 4)3 -0.080212   0.034847   -2.302 0.022524 *  
poly(time, 4)4 -0.089321   0.034611   -2.581 0.010678 *  
obs125         -0.138381   0.035930   -3.851 0.000165 ***
fmonth2        -0.036264   0.012233   -2.964 0.003455 ** 
fmonth3         0.044396   0.012234    3.629 0.000373 ***
fmonth4         0.071825   0.012236    5.870 2.14e-08 ***
fmonth5         0.216955   0.012445   17.434  < 2e-16 ***
fmonth6         0.217813   0.012240   17.795  < 2e-16 ***
fmonth7         0.291401   0.012243   23.801  < 2e-16 ***
fmonth8         0.299885   0.012247   24.486  < 2e-16 ***
fmonth9         0.183406   0.012251   14.971  < 2e-16 ***
fmonth10        0.193823   0.012256   15.815  < 2e-16 ***
fmonth11        0.115142   0.012261    9.391  < 2e-16 ***
fmonth12        0.124821   0.012267   10.176  < 2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1 

Residual standard error: 0.0346 on 175 degrees of freedom
Multiple R-squared: 0.9839,     Adjusted R-squared: 0.9824 
F-statistic: 669.1 on 16 and 175 DF,  p-value: < 2.2e-16 


> plot(predict(model2),resid(model2),xlab="predicted
gasdemand",ylab="loggasdemand residual",main="Residual by Predicted
Plot")
```



With use of the logged demand data, the heteroscedasticity evident previously is mostly gone here. The estimated seasonal indices from this model follow.

```
> b1<-coef(model2)[1]
> b2<-coef(model2)[7:17]+b1
> b3<-c(b1,b2)
> seas<-exp(b3-mean(b3))
> seas
(Intercept)     fmonth2     fmonth3     fmonth4     fmonth5     fmonth6
  0.8662340   0.8353835   0.9055579   0.9307397   1.0761123   1.0770355
    fmonth7     fmonth8     fmonth9    fmonth10    fmonth11    fmonth12
  1.1592816   1.1691586   1.0406091   1.0515055   0.9719428   0.9813965
```



Demand is estimated to be highest in July and August, and lowest in January and February. Other relatively high demand months are May, June, September and October. For example, demand in August is estimated to be 17 percent above that of the trend level, and demand in February is estimated to be 16 percent below that of the trend level. Note the small bump in October.

The fit with cosine and sine dummies for the seasonal component, instead of the month dummies, follows.

```
> cosm<-matrix(nrow=length(time),ncol=6)
> sinm<-matrix(nrow=length(time),ncol=5)
> for(i in 1:5){
+ cosm[,i]<-cos(2*pi*i*time/12)
+ sinm[,i]<-sin(2*pi*i*time/12)
+ }
> cosm[,6]<-cos(pi*time)

> model3<-
lm(loggasdemand~poly(time,4)+obs125+cosm[,1]+sinm[,1]+cosm[,2]+sinm[,2]
+cosm[,3]+sinm[,3]+cosm[,4]+sinm[,4]+cosm[,5]+sinm[,5]+cosm[,6]);summar
y(model3)

Call:
lm(formula = loggasdemand ~ poly(time, 4) + obs125 + cosm[, 1] +
    sinm[, 1] + cosm[, 2] + sinm[, 2] + cosm[, 3] + sinm[, 3] +
    cosm[, 4] + sinm[, 4] + cosm[, 5] + sinm[, 5] + cosm[, 6])

Residuals:
      Min        1Q    Median        3Q       Max
-0.101949 -0.022296 -0.001644  0.025653  0.070251

Coefficients:
                Estimate Std. Error  t value Pr(>|t|)
(Intercept)    11.963066   0.002504 4777.580  < 2e-16 ***
poly(time, 4)1  3.231952   0.034695   93.152  < 2e-16 ***
poly(time, 4)2  0.015048   0.034665    0.434 0.664746
poly(time, 4)3 -0.080212   0.034847   -2.302 0.022524 *
poly(time, 4)4 -0.089321   0.034611   -2.581 0.010678 *
obs125         -0.138381   0.035930   -3.851 0.000165 ***
cosm[, 1]      -0.090100   0.003547  -25.400  < 2e-16 ***
sinm[, 1]      -0.105095   0.003549  -29.612  < 2e-16 ***
cosm[, 2]       0.026991   0.003537    7.630 1.45e-12 ***
sinm[, 2]      -0.006167   0.003550   -1.737 0.084095 .
cosm[, 3]       0.020193   0.003532    5.717 4.60e-08 ***
sinm[, 3]      -0.008429   0.003552   -2.373 0.018710 *
cosm[, 4]      -0.000991   0.003537   -0.280 0.779664
sinm[, 4]      -0.005581   0.003547   -1.574 0.117361
cosm[, 5]       0.023411   0.003547    6.600 4.73e-10 ***
sinm[, 5]       0.027161   0.003536    7.681 1.08e-12 ***
cosm[, 6]       0.001717   0.002504    0.686 0.493888
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
Residual standard error: 0.0346 on 175 degrees of freedom
Multiple R-squared: 0.9839,     Adjusted R-squared: 0.9824
F-statistic: 669.1 on 16 and 175 DF,  p-value: < 2.2e-16
```

Let's fit the model without the variables *cos4*, *sin4*, and *cos6*—it appears that they are insignificant.

```
> cosm<-matrix(nrow=length(time),ncol=6)
> sinm<-matrix(nrow=length(time),ncol=5)
> for(i in 1:5){
+ cosm[,i]<-cos(2*pi*i*time/12)
+ sinm[,i]<-sin(2*pi*i*time/12)
+ }
> cosm[,6]<-cos(pi*time)

> model4<-
lm(loggasdemand~poly(time,4)+obs125+cosm[,1]+sinm[,1]+cosm[,2]+sinm[,2]
+cosm[,3]+sinm[,3]+cosm[,5]+sinm[,5]);summary(model4)

Call:
lm(formula = loggasdemand ~ poly(time, 4) + obs125 + cosm[, 1] +
    sinm[, 1] + cosm[, 2] + sinm[, 2] + cosm[, 3] + sinm[, 3] +
    cosm[, 5] + sinm[, 5])

Residuals:
      Min        1Q    Median        3Q       Max
-0.104753 -0.020841 -0.002069  0.025151  0.071234

Coefficients:
                Estimate Std. Error  t value Pr(>|t|)
(Intercept)    11.963046   0.002504 4777.293  < 2e-16 ***
poly(time, 4)1  3.232304   0.034692   93.171  < 2e-16 ***
poly(time, 4)2  0.015268   0.034667    0.440 0.660173
poly(time, 4)3 -0.079174   0.034840   -2.273 0.024250 *
poly(time, 4)4 -0.089431   0.034614   -2.584 0.010578 *
obs125         -0.134512   0.035637   -3.775 0.000218 ***
cosm[, 1]      -0.090067   0.003547  -25.390  < 2e-16 ***
sinm[, 1]      -0.105106   0.003549  -29.613  < 2e-16 ***
cosm[, 2]       0.026968   0.003538    7.623 1.42e-12 ***
sinm[, 2]      -0.006127   0.003550   -1.726 0.086053 .
cosm[, 3]       0.020191   0.003533    5.715 4.53e-08 ***
sinm[, 3]      -0.008467   0.003552   -2.384 0.018178 *
cosm[, 5]       0.023374   0.003548    6.589 4.84e-10 ***
sinm[, 5]       0.027141   0.003537    7.674 1.05e-12 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.0346 on 178 degrees of freedom
Multiple R-squared: 0.9836,     Adjusted R-squared: 0.9824
F-statistic: 823.1 on 13 and 178 DF,  p-value: < 2.2e-16
```

Next, here is the partial F test for significance of the variables omitted in model 4.

```
> anova(model4,model3)
Analysis of Variance Table

Model 1: loggasdemand ~ poly(time, 4) + obs125 + cosm[, 1] + sinm[, 1]
+
    cosm[, 2] + sinm[, 2] + cosm[, 3] + sinm[, 3] + cosm[, 5] +
    sinm[, 5]
Model 2: loggasdemand ~ poly(time, 4) + obs125 + cosm[, 1] + sinm[, 1]
+
    cosm[, 2] + sinm[, 2] + cosm[, 3] + sinm[, 3] + cosm[, 4] +
    sinm[, 4] + cosm[, 5] + sinm[, 5] + cosm[, 6]

  Res.Df     RSS Df Sum of Sq      F Pr(>F)
1    178 0.21313
2    175 0.20950  3 0.0036358 1.0124 0.3886
```

Thus, the three variables can be dropped simultaneously and we use model 4. Amplitude and phase estimates for the seasonal structure follow.

```
> ampltd<-c(rep(0,times=4))
> b2<-coef(model4)[7:14]
> for(i in 1:4){
+ i1<-2*i-1
+ i2<-i1+1
+ ampltd[i]<-sqrt(b2[i1]^2+b2[i2]^2)
+ }
> ampltd
[1] 0.13841748 0.02765571 0.02189425 0.03581856

> phase<-c(rep(0,times=4))
> for(i in 1:4){
+ i1<-2*i-1
+ i2<-i1+1
+ phase[i]<-atan(-b2[i2]/b2[i1])
+ if(b2[i1]<0)phase[i]<-phase[i]+pi
+ if((b2[i1]>0)&(b2[i2]>0))phase[i]<-phase[i]+2*pi
+ }
> phase
[1] 2.2792939 0.2234112 0.3970899 5.4233466

> harm<-c(1,2,3,5)
> peak<-c(rep(0,times=4))
> for(i in 1:4){
+ ir<-harm[i]
+ peak[i]<-(12/ir)-6*phase[i]/(pi*ir)
+ }
> peak
[1] 7.6468692 5.7866580 3.7472047 0.3284342
```

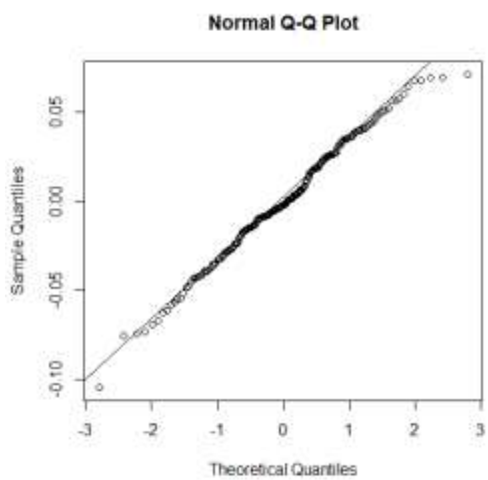
Amplitude and Phase Estimates

	Amplitude	Phase Degrees	Phase Radians	Peak t (period)
Fundamental	0.138	130.59	2.279	7.65 (12)
Second harmonic	0.028	12.80	0.223	5.79, 11.79 (6)
Third harmonic	0.022	22.75	0.397	3.75, 7.75, 11.75 (4)
Fifth harmonic	0.036	310.73	5.423	0.33, 2.73, 5.13, 7.53, 9.93 (2.4)

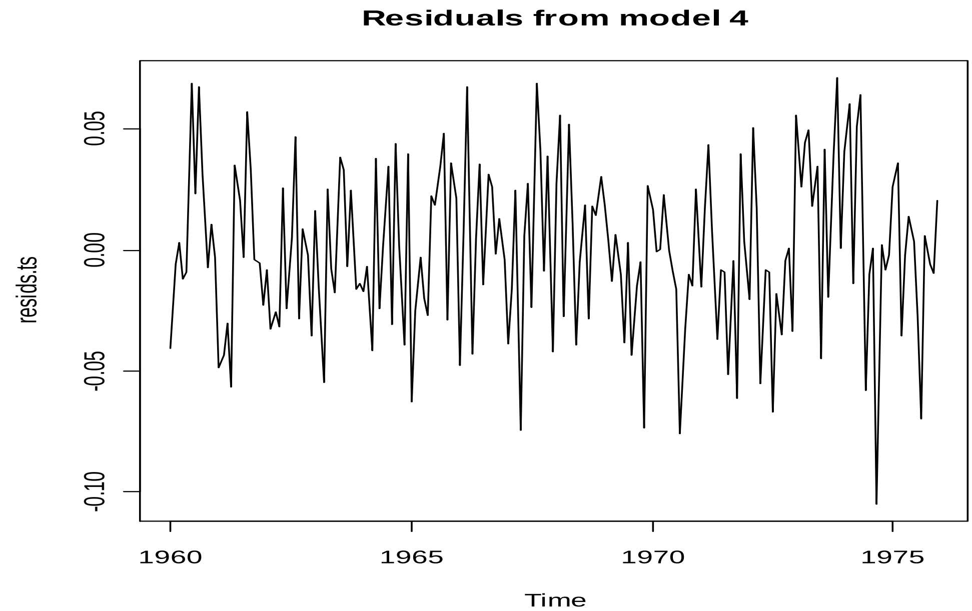
The fundamental is the dominant trigonometric component, and there are three significant overtones. The table shows evidence of the peak demand occurring during August.

Let's examine the residuals from model 4.

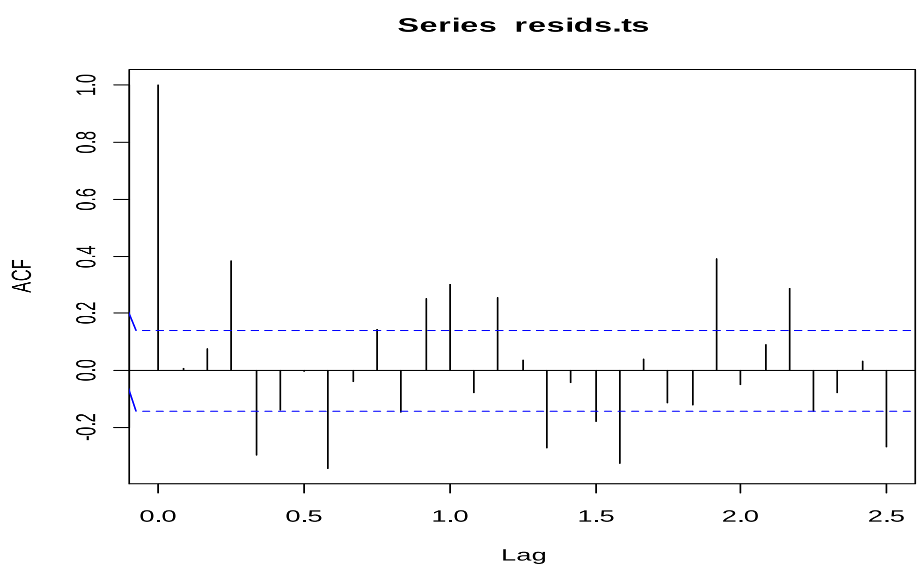
```
> qqnorm(resid(model4))
> qqline(resid(model4))
```




```
> shapiro.test(resid(model4))

        Shapiro-Wilk normality test

data:  resid(model4)
W = 0.99232, p-value = 0.4086


> resids.ts<-ts(resid(model4),start=c(1960,1),freq=12)
> plot(resids.ts,main="Residuals from model 4")
```




```
> acf(resids.ts,30)
```



In defining the time series *resids.ts*, we specified `freq=12`. This is why the horizontal axis of the residual acf plot is labelled with the cycle count instead of the lag count.

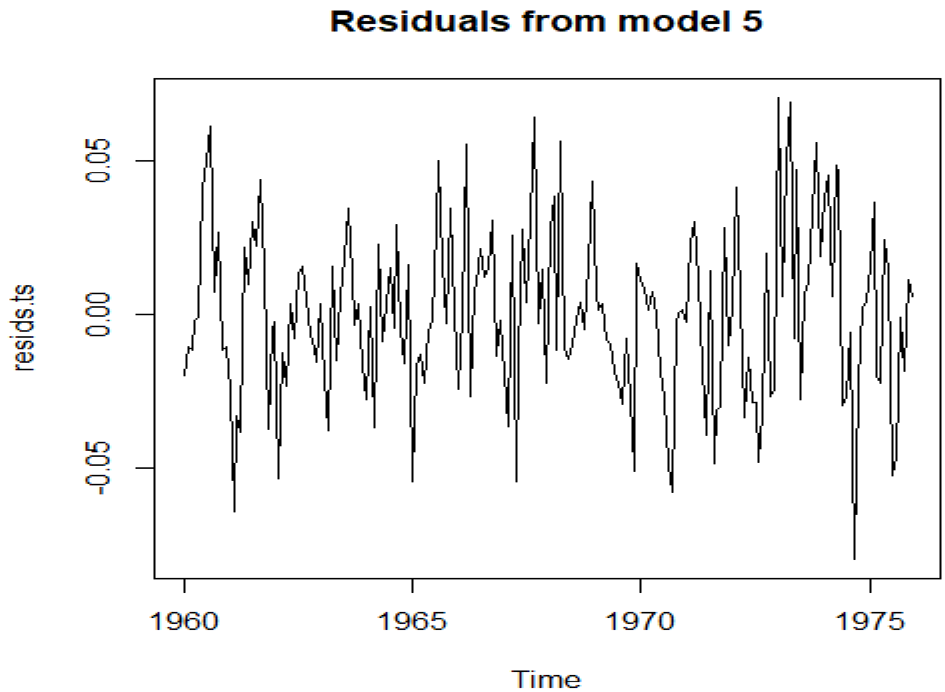
Overall, there is little remaining trend of consequence, although the last three years of data do show some uncaptured trending . More seriously, there is considerable autocorrelation structure remaining in the residuals. The residuals are well described by a normal distribution.

Next, we add the trigonometric pairs with frequencies 0.348 and 0.432 for calendar effects. We need to construct the 0.432 pair first.

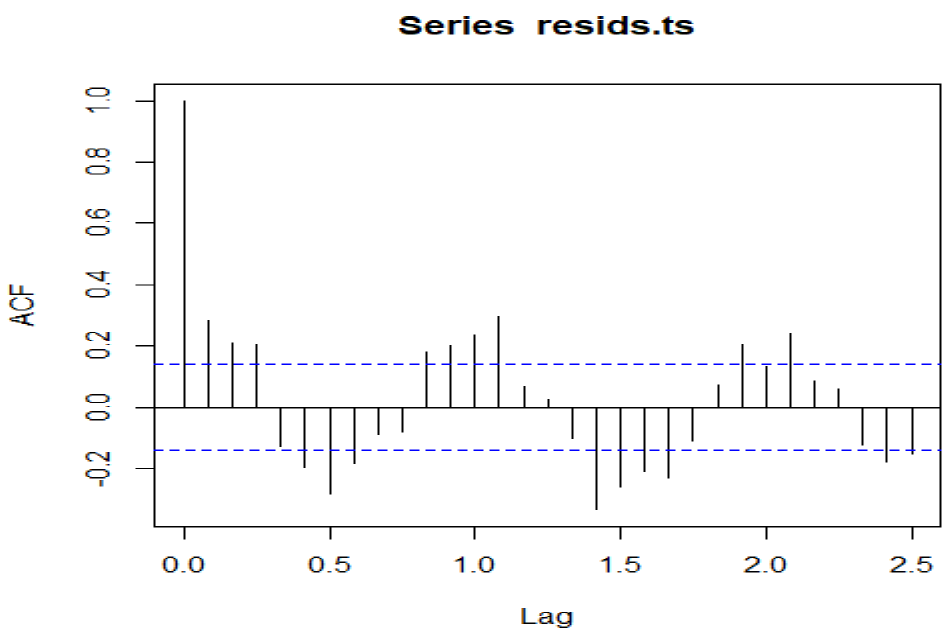
```
> c432<-cos(0.864*pi*time)
> s432<-sin(0.864*pi*time)

> model5<-
lm(loggasdemand~poly(time,4)+obs125+fmonth+c348+s348+c432+s432);summary
(model5)

Call:
lm(formula = loggasdemand ~ poly(time, 4) + obs125 + fmonth +
    c348 + s348 + c432 + s432)

Residuals:
      Min        1Q    Median        3Q       Max
-0.079616 -0.018530 -0.000547  0.015840  0.070518

Coefficients:
                Estimate Std. Error  t value Pr(>|t|)
(Intercept)    11.820125   0.007346 1609.145  < 2e-16 ***
poly(time, 4)1  3.229035   0.029415  109.777  < 2e-16 ***
poly(time, 4)2  0.018520   0.029389    0.630 0.529429
poly(time, 4)3 -0.081283   0.029554   -2.750 0.006595 **
poly(time, 4)4 -0.087328   0.029340   -2.976 0.003340 **
obs125         -0.112058   0.030840   -3.634 0.000369 ***
fmonth2        -0.038352   0.010376   -3.696 0.000294 ***
fmonth3         0.044519   0.010375    4.291 2.97e-05 ***
fmonth4         0.071719   0.010372    6.915 8.95e-11 ***
fmonth5         0.213125   0.010568   20.168  < 2e-16 ***
fmonth6         0.218354   0.010380   21.036  < 2e-16 ***
fmonth7         0.290915   0.010380   28.028  < 2e-16 ***
fmonth8         0.297764   0.010389   28.663  < 2e-16 ***
fmonth9         0.184404   0.010389   17.751  < 2e-16 ***
fmonth10        0.192645   0.010390   18.542  < 2e-16 ***
fmonth11        0.113593   0.010401   10.921  < 2e-16 ***
fmonth12        0.125708   0.010401   12.087  < 2e-16 ***
c348            0.024125   0.003025    7.976 2.08e-13 ***
s348            0.001676   0.002996    0.559 0.576652
c432            0.001175   0.003009    0.391 0.696576
s432           -0.008671   0.002999   -2.891 0.004333 **
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.02933 on 171 degrees of freedom
Multiple R-squared:  0.9887,    Adjusted R-squared:  0.9874
F-statistic: 748.6 on 20 and 171 DF,  p-value: < 2.2e-16
```

Partial *F* tests show that both calendar trigonometric pairs are significant. We next examine the residuals.

```
> resids.ts<-ts(resid(model5),start=c(1960,1),freq=12)
> plot(resids.ts,main="Residuals from model 5")
```




```
> acf(resids.ts,30)
```



Addition of the calendar pairs has certainly led to improvement—there is less autocorrelation in the residuals than previously. However, the estimation has still not produced white noise residual structure. There is autocorrelation present, especially near lags 6, 12, 18, and 24, indicating inadequate estimation of the seasonal component. The regression methodology employed here has assumed that the seasonal pattern is nonrandom and does not change from year to year, that is, it is static. A better approach to estimation of the seasonal component is needed, modelling dynamic structure. We will encounter such an approach later in the course when we consider ARIMA modelling.

Summary and additional remarks

1. Initially, plotting of the Ontario gas demand time series does not clearly indicate whether an additive decomposition model can be fit, or a multiplicative decomposition model is needed. The residuals from the additive fit show clearly that there is heteroscedasticity present and that a multiplicative model is preferable.

2. A multiplicative model is fit with a fourth-degree polynomial time trend, a dummy for an outlier observation, and the month dummies. The resulting fit does a decent job of estimating trend structure, but the residuals show substantial unmodeled correlation structure remaining. Addition to the model of the calendar trigonometric pair with frequency 0.348 improves the fit, but leaves significant correlation structure near lags 6, 12, 18, and 24 in the residuals. We conclude that the seasonal is slowly changing from year to year—it is dynamic, and a better estimation of the seasonal structure via another method is needed. Later in the course we will explore such a method.

3. A model is also fit in which cosine and sine functions are used to track seasonal structure. The fourth harmonic cosine-sine pair and the sixth harmonic cosine term are dropped because they are not statistically significant.