DSA 8020 R Session 12: Time Series Analysis II

Whitney

Contents

Seasonal Component Estimation

Now let's consider the situation that a time series consists of seasonal component only (assuming the trend has been estimated/removed). That is

$$Y_t = s_t + \eta_t.$$

with $\{s_t\}$ having period d (i.e., $s_t = s_{t+jd}$ for all integers j and t). $\sum_{t=1}^{d} s_t = 0$ and $\mathbb{E}[\eta_t] = 0$. We can use a harmonic regression or a seasonal factor model to estimate the seasonal components or to use seasonal-differencing to remove the seasonality.

Harmonic Regression

A harmonic regression model has the form

$$s_t = \sum_{j=1}^{k} A_k \cos(2\pi f_j + \phi_j).$$

For each $j = 1, \cdots, k$:

- $A_j > 0$ is the *amplitude* of the jth cosine wave.
- f_j controls the the *frequency* of the j-th cosine wave (how often waves repeats).
- $\phi_j \in [-\pi, \pi]$ is the *phase* of the j-th wave (where it starts)

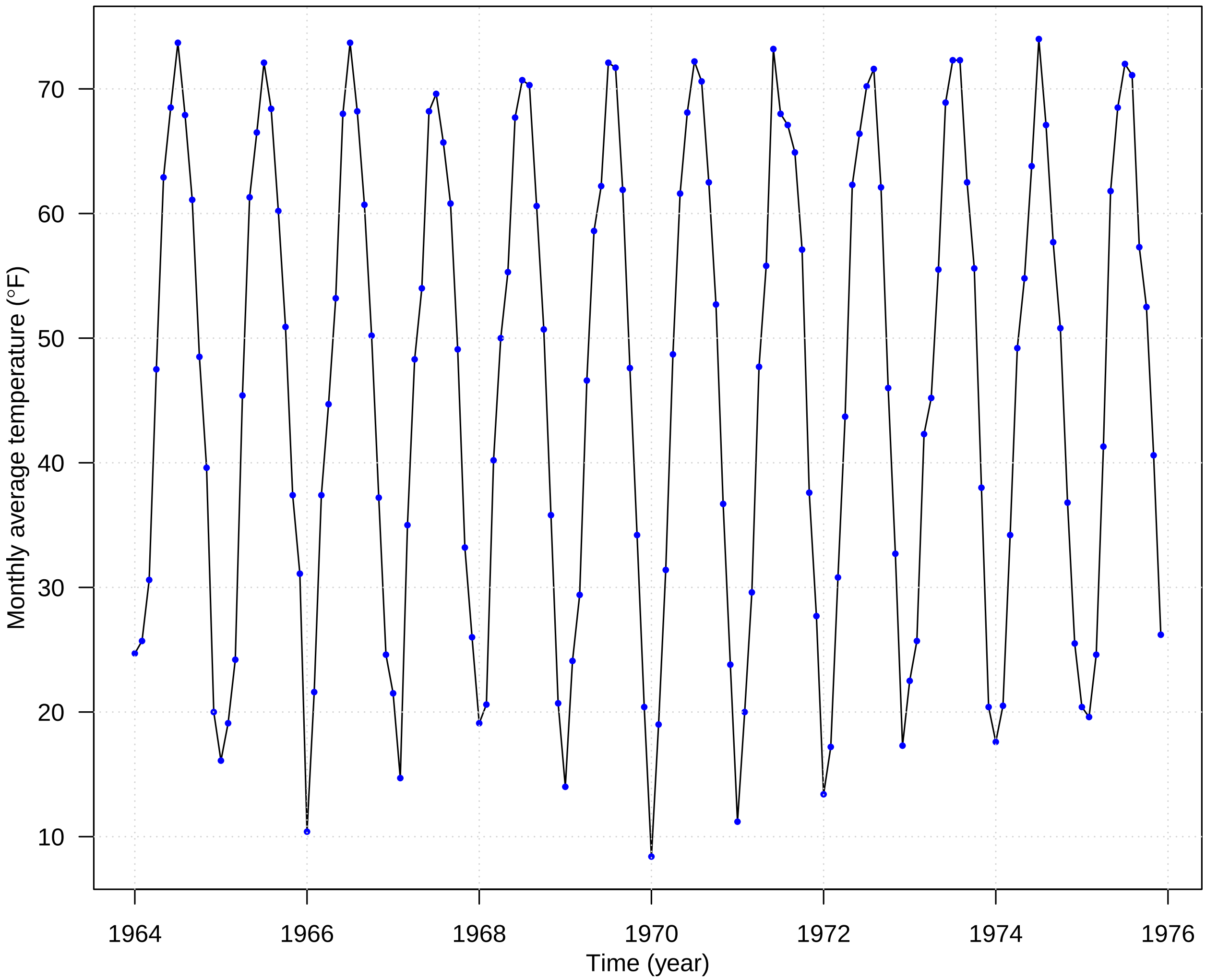
The above can be expressed as

$$\sum_{j=1}^{k} \left(\beta_{1j}\cos(2\pi f_j) + \beta_{2j}\sin(2\pi f_j)\right),$$

where $\beta_{1j} = A_j\cos(\phi_j)$ and $\beta_{2j} = A_j\sin(\phi_j)$. Therefore, if the frequencies $\{f_j\}_{j=1}^k$ are known, we can use regression techniques to estimate the parameters $\{\beta_{1j}, \beta_{2j}\}_{j=1}^k$ by treating $\{\cos(2\pi f_j)\}_{j=1}^k$ and $\{\sin(2\pi f_j)\}_{j=1}^k$ as predictor variables.

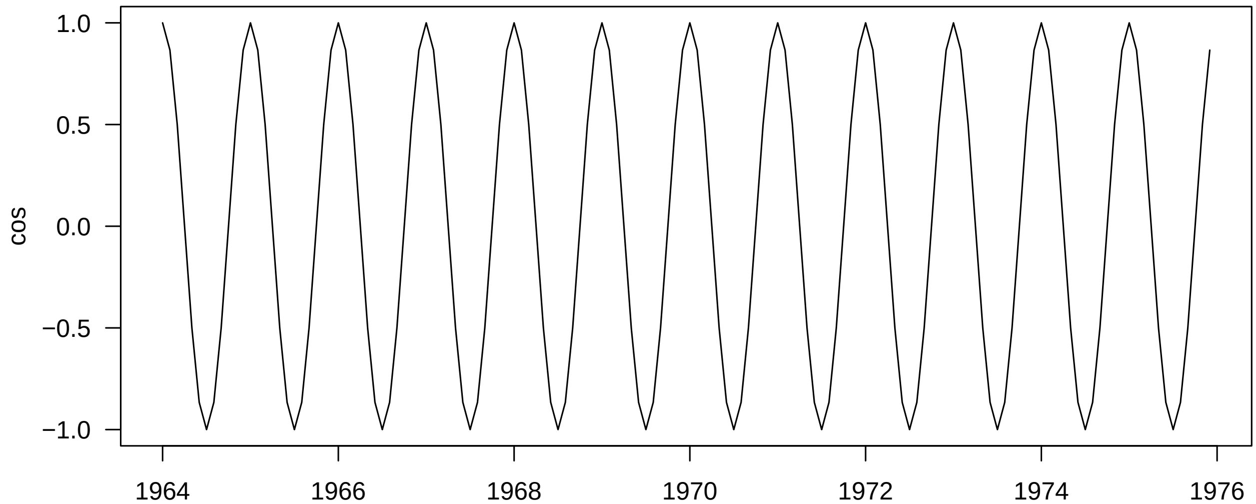
Let's use the monthly average temperature (in degrees Fahrenheit) recorded in Dubuque, IA from Jan. 1964 - Dec. 1975.

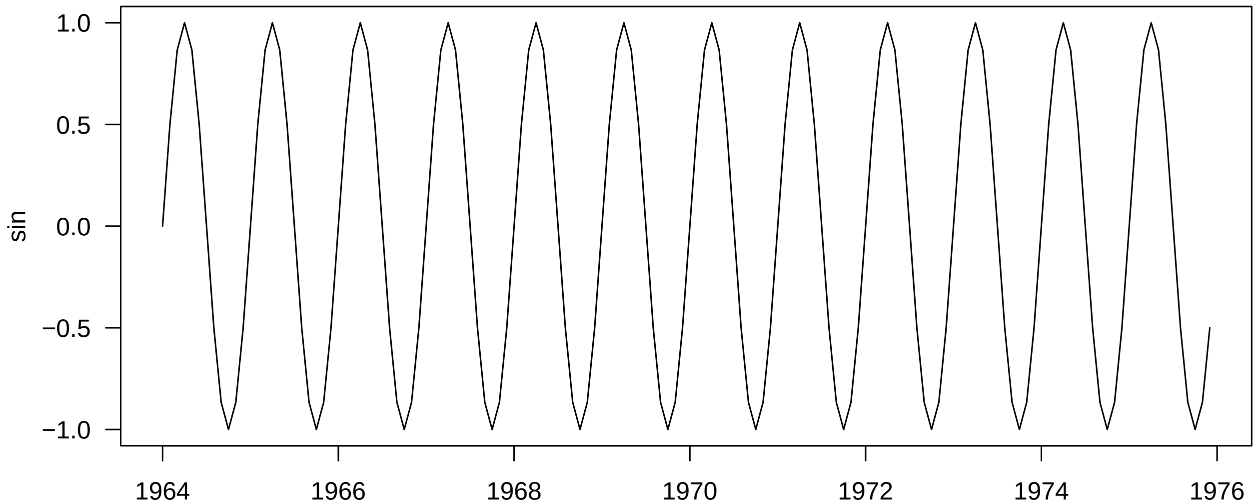
```r
library(TSA)
data(tempdub)
time <- as.numeric(time(tempdub))
par(mar = c(4, 4, 0.8, 0.6))
plot(tempdub, type = "l", las = 1, xlab = "", ylab = "")
points(tempdub, pch = 16, col = "blue", cex = 0.6)
grid()
mtext("Time (year)", side = 1, line = 2)
mtext(expression(paste("Monthly average temperature (", degree, "F)")), side = 2, line = 2)
```



First, we need to set up the harmonics (assuming yearly cycle)

```
harmonics <- harmonic(tempdub, 1)
time <- as.numeric(time(tempdub))
par(mfrow = c(2, 1), las = 1, mar = c(2, 4, 0.8, 0.6))
plot(time, harmonics[, 1], type = "l", ylab = "cos")
plot(time, harmonics[, 2], type = "l", ylab = "sin")
```





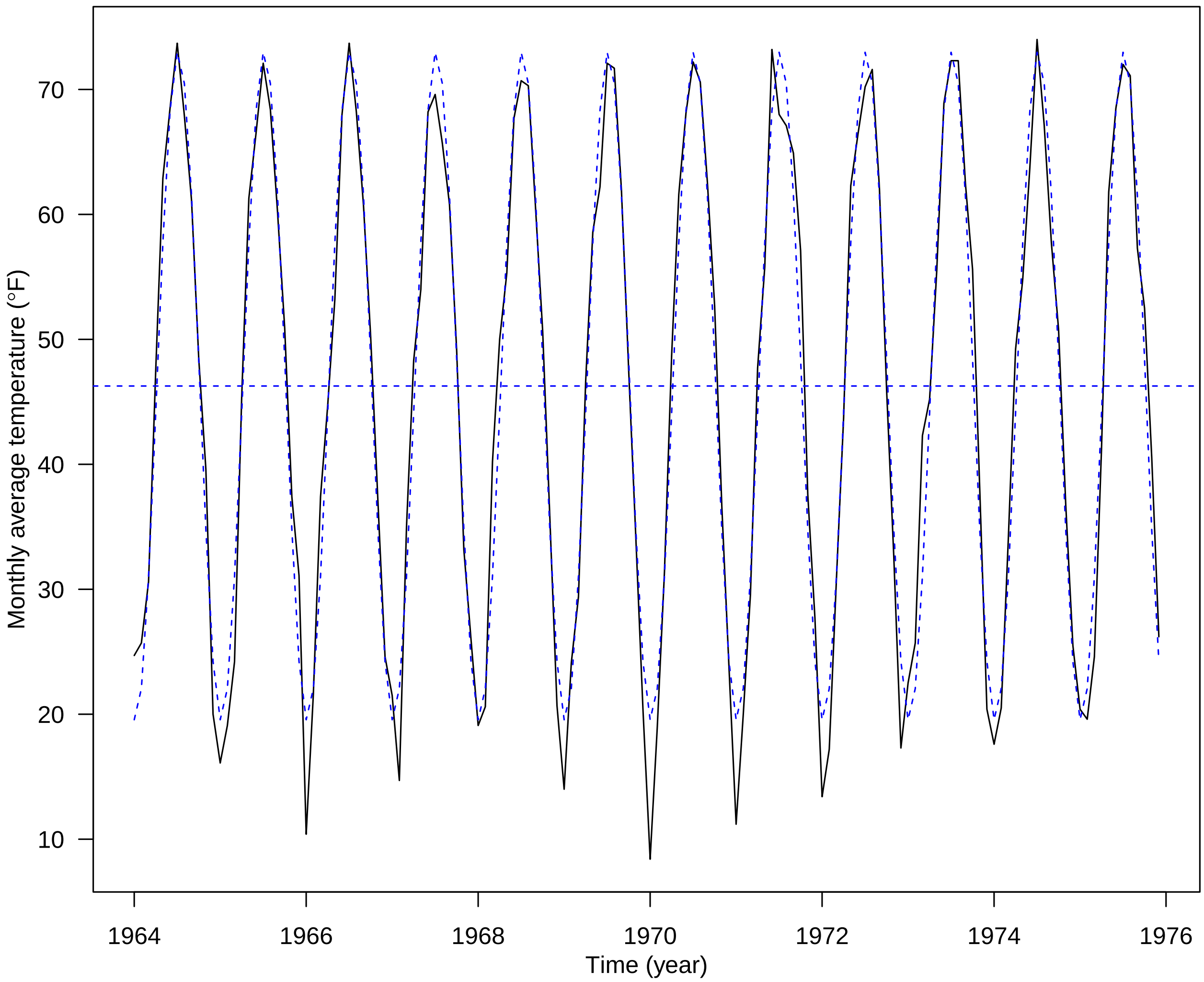
Next, perform a linear regression using the harmonics we just created as the predictors

```r
harReg <- lm(tempdub ~ harmonics)
summary(harReg)
```

```
##
## Call:
## lm(formula = tempdub ~ harmonics)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -11.1580  -2.2756  -0.1457   2.3754  11.2671
##
## Coefficients:
##                        Estimate Std. Error t value Pr(>|t|)
## (Intercept)             46.2660     0.3088 149.816  < 2e-16 ***
## harmonicscos(2*pi*t) -26.7079     0.4367 -61.154  < 2e-16 ***
## harmonicssin(2*pi*t)  -2.1697     0.4367  -4.968 1.93e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3.706 on 141 degrees of freedom
```

```
## Multiple R-squared:  0.9639, Adjusted R-squared:  0.9634
## F-statistic:  1882 on 2 and 141 DF,  p-value: < 2.2e-16
```

```r
par(mar = c(3.6, 3.6, 0.8, 0.6))
plot(time, tempdub, type = "l", las = 1, xlab = "", ylab = "")
mtext("Time (year)", side = 1, line = 2)
mtext(expression(paste("Monthly average temperature (", degree, "F)")), side = 2, line = 2)
time <- as.numeric(time(tempdub))
lines(time, harReg$fitted.values, col = "blue", lty = 2)
abline(h = harReg$coefficients[1], lty = 2, col = "blue")
```



Seasonal factors

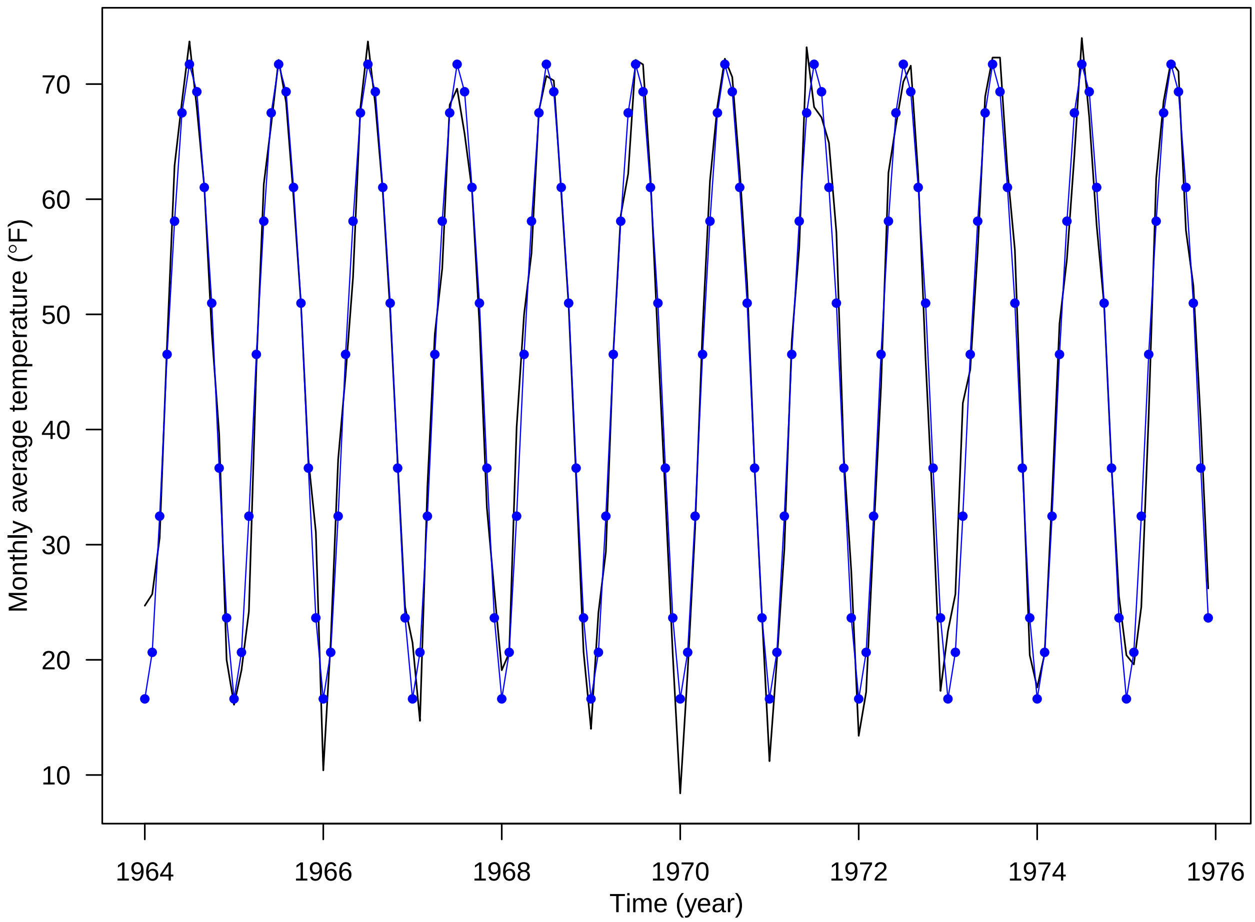
Harmonic regression assume the seasonal pattern has a regular shape, i.e. the height of the peaks is the same as the depth of the troughs. Assuming the seasonal pattern repeats itself every d time points, a less restrictive approach is to model it as

$$s_t = \begin{cases} \beta_1 & \text{for } t = 1, 1+d, 1+2d, \cdots; \\ \beta_2 & \text{for } t = 2, 2+d, 2+2d, \cdots; \\ \vdots & \vdots; \\ \beta_d & \text{for } t = d, 2d, 3d, \cdots. \end{cases}$$

```r
month = season(tempdub)
season_means <- lm(tempdub ~ month - 1)
summary(season_means)
```

```
##
## Call:
## lm(formula = tempdub ~ month - 1)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -8.2750 -2.2479  0.1125  1.8896  9.8250
##
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)
## monthJanuary     16.608      0.987   16.83   <2e-16 ***
## monthFebruary    20.650      0.987   20.92   <2e-16 ***
## monthMarch       32.475      0.987   32.90   <2e-16 ***
## monthApril       46.525      0.987   47.14   <2e-16 ***
## monthMay         58.092      0.987   58.86   <2e-16 ***
## monthJune        67.500      0.987   68.39   <2e-16 ***
## monthJuly        71.717      0.987   72.66   <2e-16 ***
## monthAugust      69.333      0.987   70.25   <2e-16 ***
## monthSeptember   61.025      0.987   61.83   <2e-16 ***
## monthOctober     50.975      0.987   51.65   <2e-16 ***
## monthNovember    36.650      0.987   37.13   <2e-16 ***
## monthDecember    23.642      0.987   23.95   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3.419 on 132 degrees of freedom
## Multiple R-squared:  0.9957, Adjusted R-squared:  0.9953
## F-statistic:  2569 on 12 and 132 DF,  p-value: < 2.2e-16
```

```r
plot(tempdub, type = "l", las = 1, xlab = "", ylab = "")
mtext("Time (year)", side = 1, line = 2)
mtext(expression(paste("Monthly average temperature (", degree, "F)")), side = 2, line = 2)
points(time, season_means$fitted.values, col = "blue", pch = 16, cex = 0.8)
lines(time, season_means$fitted.values, col = "blue", lwd = 0.75)
```



Let's put trend and seasonal variation together

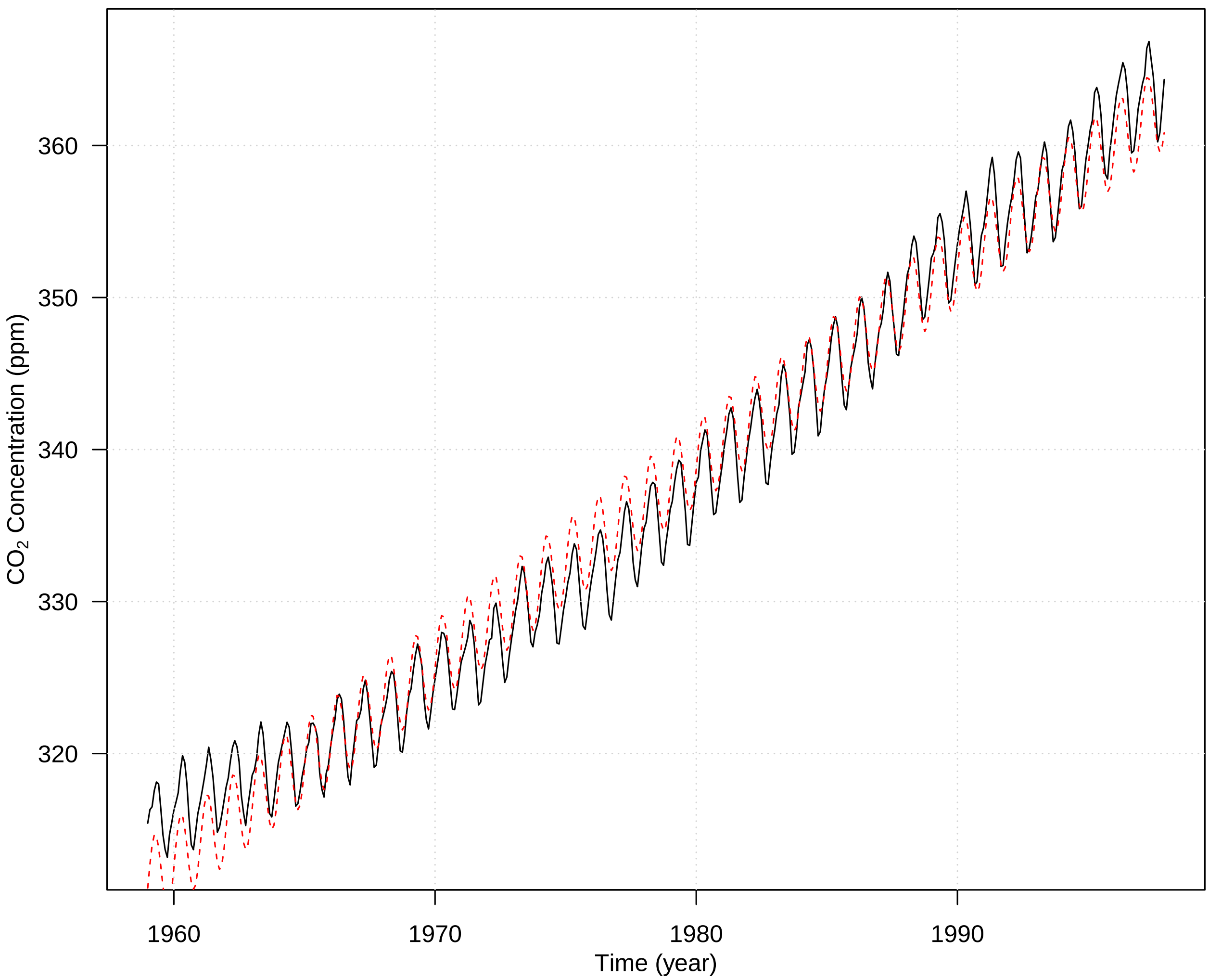
Here we using the CO_2 concentration time series is an example. First, we can perform a linear regression with both time and the harmonics as the covariates.

```
time <- as.numeric(time(co2))
harmonics <- harmonic(co2, 1)

lm_trendSeason <- lm(co2 ~ time + harmonics)
summary(lm_trendSeason)
```

```
##
## Call:
## lm(formula = co2 ~ time + harmonics)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -3.433 -1.323 -0.282  1.221  4.615
##
## Coefficients:
##                          Estimate Std. Error  t value Pr(>|t|)
## (Intercept)            -2.256e+03  1.391e+01 -162.155  < 2e-16 ***
## time                    1.311e+00  7.033e-03  186.382  < 2e-16 ***
```

```
## harmonicscos(2*pi*t) -3.889e-01  1.120e-01   -3.474  0.00056 ***
## harmonicssin(2*pi*t)  2.772e+00  1.120e-01   24.760  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.712 on 464 degrees of freedom
## Multiple R-squared:  0.987,  Adjusted R-squared:  0.9869
## F-statistic: 1.173e+04 on 3 and 464 DF,  p-value: < 2.2e-16
```

```r
par(mar = c(3.8, 4, 0.8, 0.6))
plot(time, co2, type = "l", las = 1, xlab = "", ylab = "")
#points(co2, col = "blue", pch = 16, cex = 0.25)
mtext("Time (year)", side = 1, line = 2)
mtext(expression(paste("CO"[2], " Concentration (ppm)")), side = 2, line = 2.5)
grid()
lines(time, lm_trendSeason$fitted.values, col = "red", lty = 2)
```



Airline Passengers Example

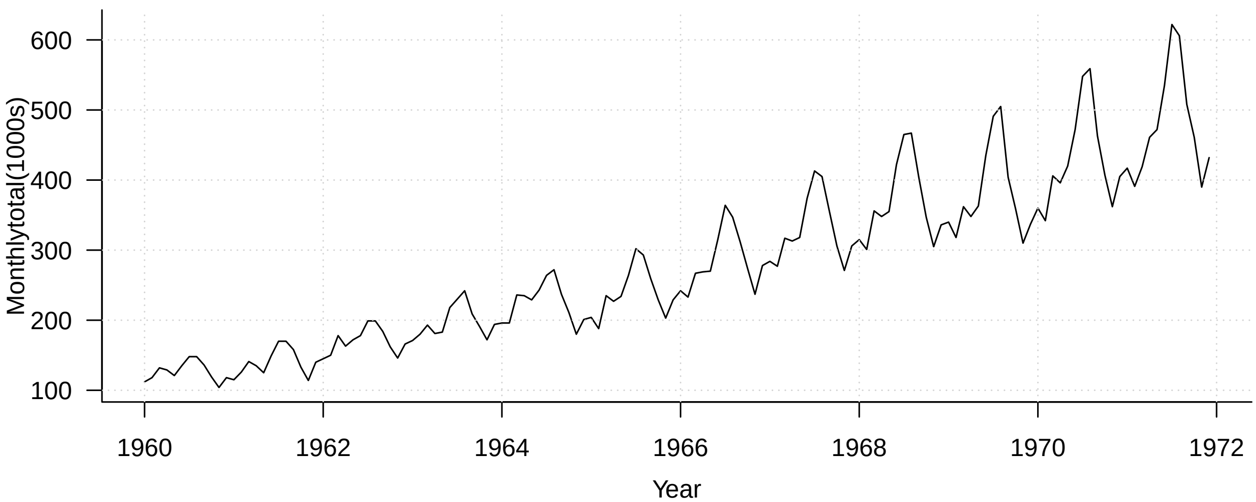
Read the data

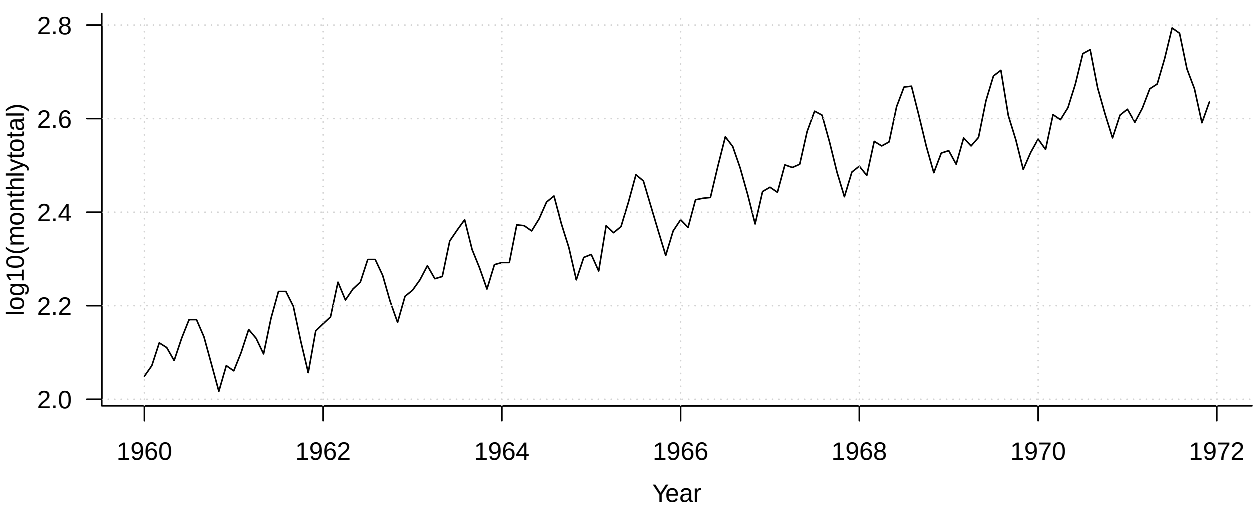
```
data(airpass)
str(airpass)
```

```
##  Time-Series [1:144] from 1960 to 1972: 112 118 132 129 121 135 148 148 136 119 ...
```

Plot the time series

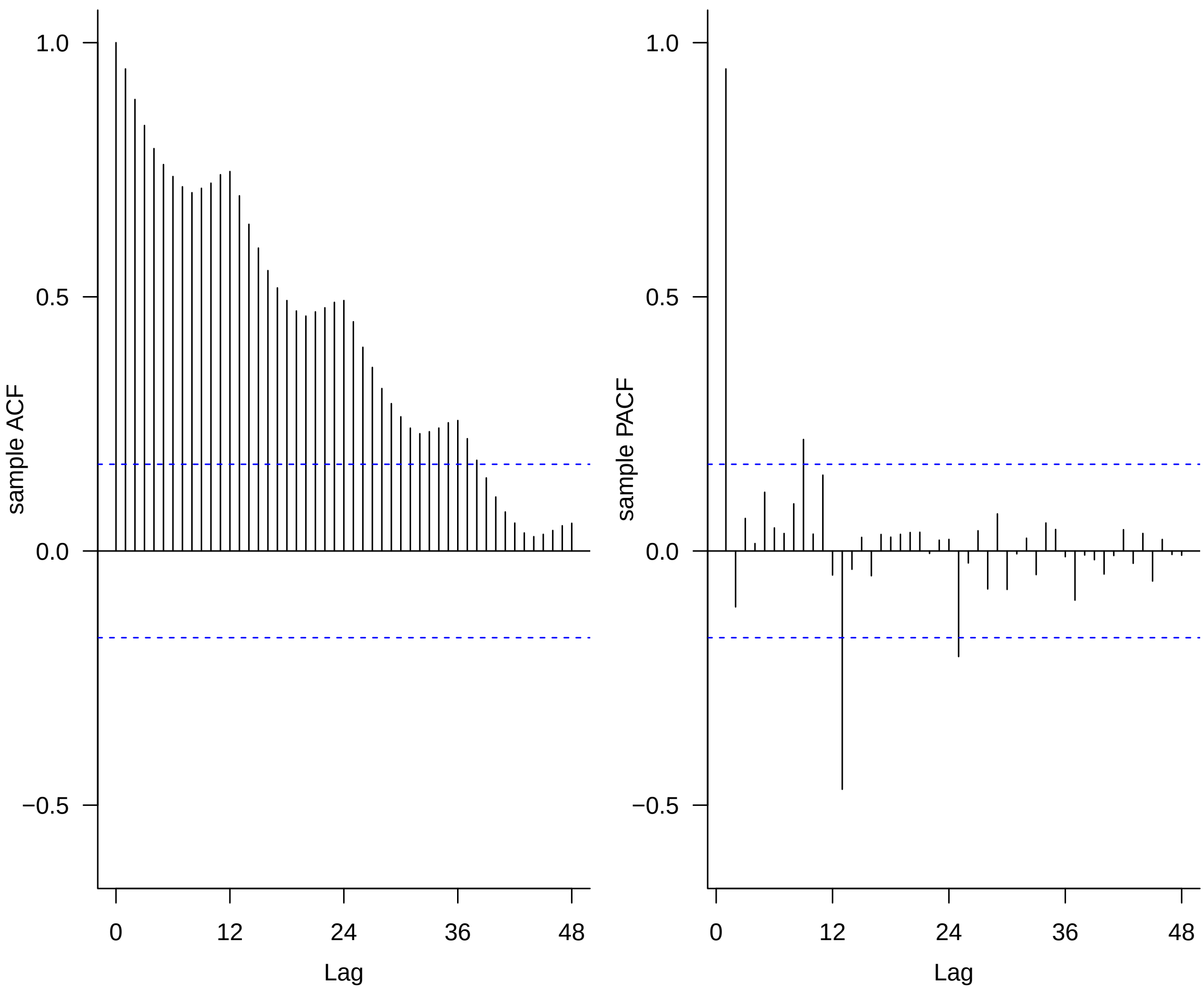
```
par(bty = "L", mar = c(3.6, 3.5, 0.8, 0.6), mgp = c(2.4, 1, 0), las = 1, mfrow = c(2, 1))
plot(airpass, xlab = "Year", ylab = "Monthlytotal(1000s)")
grid()
##take a log(to the base 10) of the airpassenger data.
log.airpass <- log10(airpass)
plot(log.airpass, type = "l", xlab = "Year", ylab = "log10(monthlytotal)")
grid()
```





Plot sample ACF/PACF

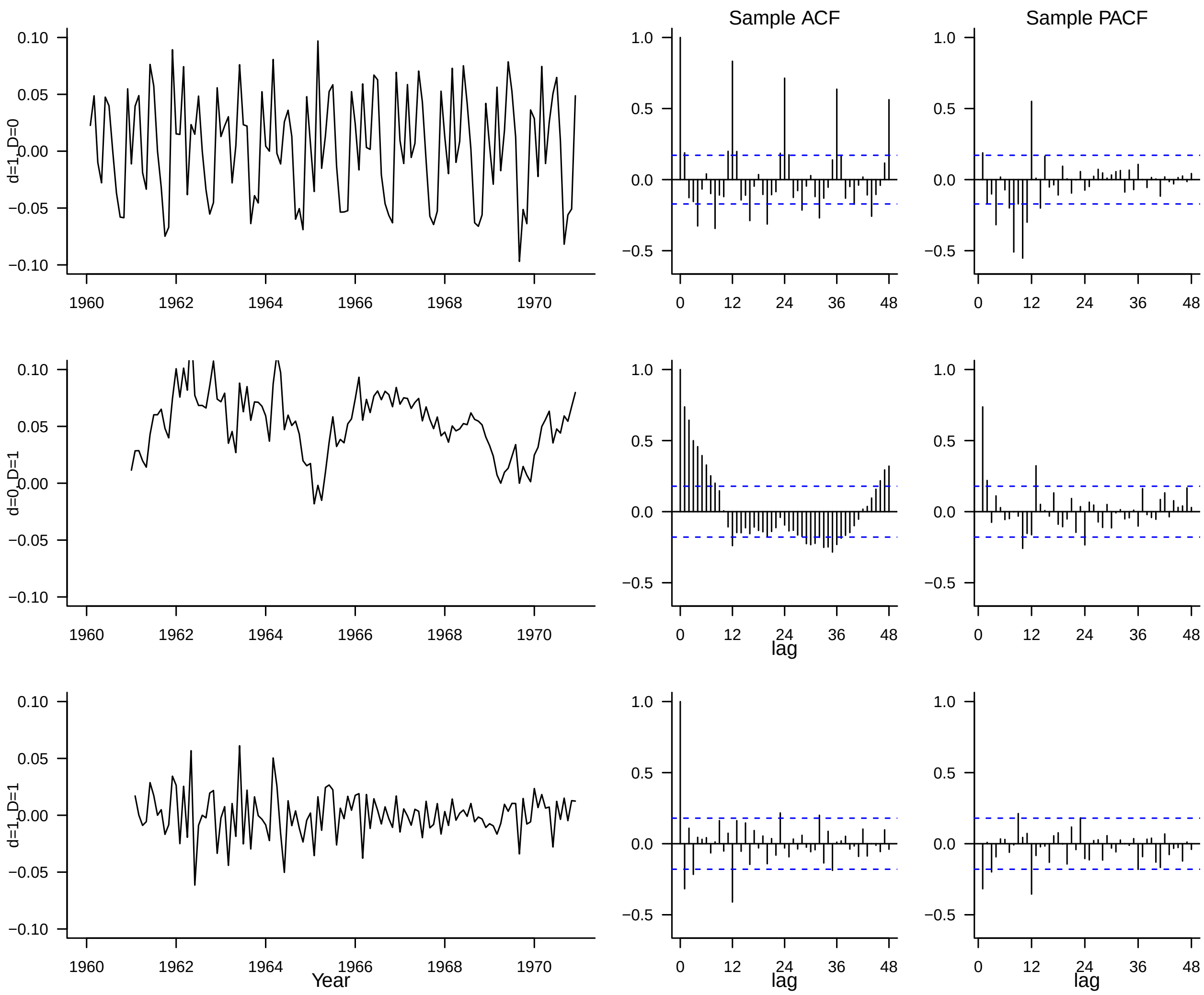
```
log.shortair <- log.airpass[1:132]
yr <- time(airpass)
shortyears <- yr[1:132]
par(bty = "L", mar = c(3.6, 3.5, 0.8, 0.6), mgp = c(2.4, 1, 0), las = 1, mfrow = c(1, 2))
stats::acf(log.shortair, ylab = "sample ACF", main = "", lag.max = 48, ylim = c(-0.6, 1), xaxt = "n")
axis(side = 1, at = seq(0, 48, 12))
stats::pacf(log.shortair, ylab = "sample PACF", main = "", lag.max = 48, ylim = c(-0.6, 1), xaxt = "n")
axis(side = 1, at = seq(0, 48, 12))
```



Trying Different Orders of Differencing

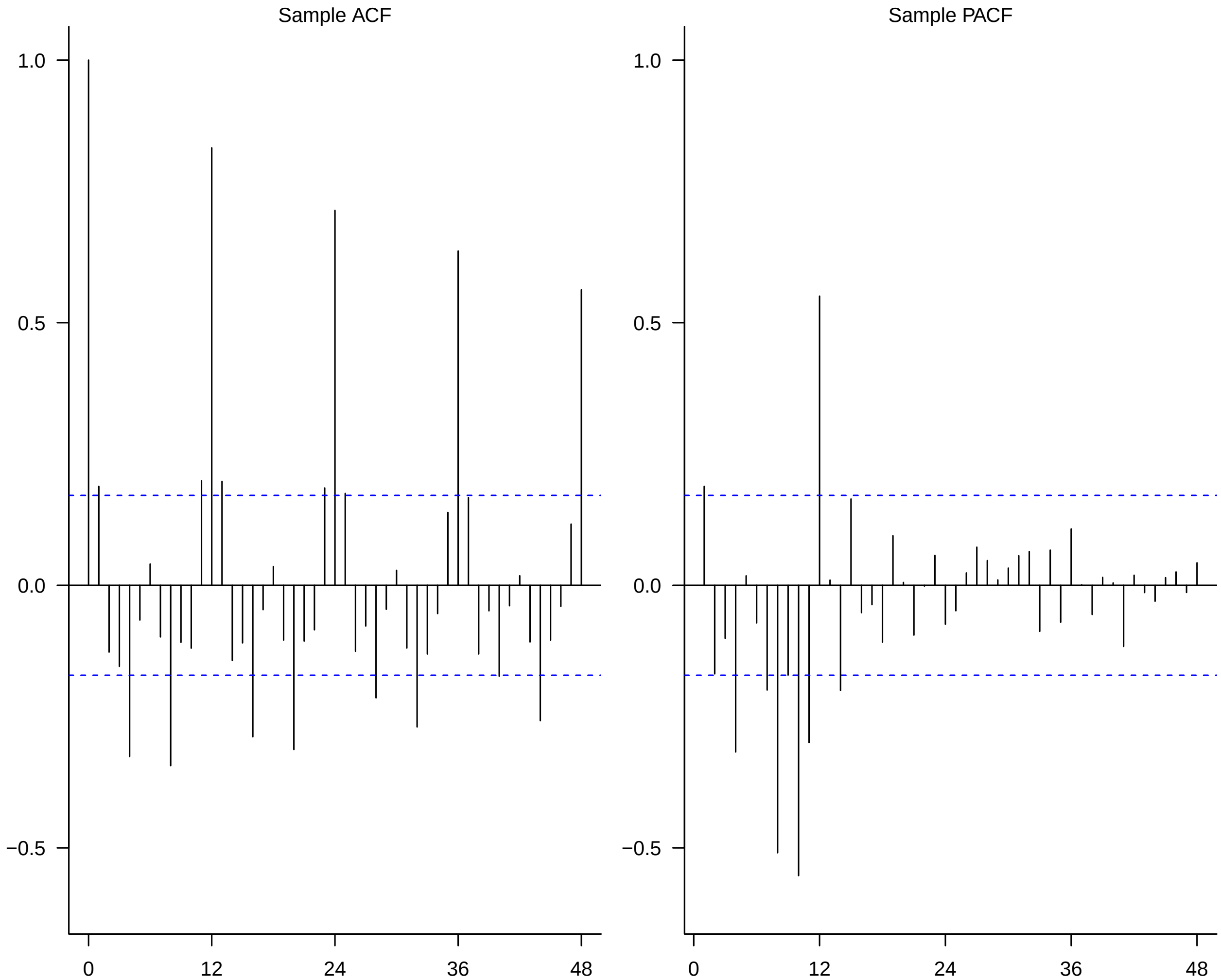
```
## take the differences Y_t = (1-B) X_t
diff.1.0 <- diff(log.shortair)
## take the seasonal differences Y_t =(1-B^(12)) X_t
diff.0.1 <- diff(log.shortair, lag = 12, diff = 1)
## take the differences Y_t = (1-B^(12)) (1-B) X_t
```

```r
diff.1.1 <- diff(diff(log.shortair, lag = 12, diff = 1))

par(bty = "L", mar = c(3.6, 3.5, 1, 0.6), mgp = c(2.4, 1, 0), las = 1)
layout.matrix <- matrix(c(1, 1, 2, 3, 4, 4, 5, 6, 7, 7, 8, 9), nrow = 3, ncol = 4, byrow = T)
layout(mat = layout.matrix)
plot(shortyears[-1], diff.1.0, xlab = "", ylab = "d=1, D=0",
     type = "l", ylim = c(-0.1, 0.1), xlim = range(shortyears))

stats::acf(diff.1.0, lag.max = 48, ylab = "", xlab = "", main = "", ylim = c(-0.6, 1), xaxt = "n")
axis(side = 1, at = seq(0, 48, 12))
mtext("Sample ACF", side = 3, line = 0, cex = 0.8)

stats::pacf(diff.1.0, lag.max = 48, ylab = "", xlab = "", main = "", ylim = c(-0.6, 1), xaxt = "n")
axis(side = 1, at = seq(0, 48, 12))
mtext("Sample PACF", side = 3, line = 0, cex = 0.8)

plot(shortyears[-c(1:12)], diff.0.1, xlab = "", ylab = "d=0, D=1",
     type = "l", ylim = c(-0.1, 0.1), xlim = range(shortyears))

stats::acf(diff.0.1, lag.max = 48, ylab = "", xlab = "", main = "", ylim = c(-0.6, 1), xaxt = "n")
axis(side = 1, at = seq(0, 48, 12))
mtext("lag", side = 1, line = 1.8, cex = 0.8)

stats::pacf(diff.0.1, lag.max = 48, ylab = "", xlab = "", main = "", ylim = c(-0.6, 1), xaxt = "n")
axis(side = 1, at = seq(0, 48, 12))

plot(shortyears[-c(1:13)], diff.1.1, xlab = "", ylab = "d=1, D=1",
     type = "l", ylim = c(-0.1, 0.1), xlim = range(shortyears))
mtext("Year", side = 1, line = 1.8, cex = 0.8)

stats::acf(diff.1.1, lag.max = 48, ylab = "", xlab = "", main = "", ylim = c(-0.6, 1), xaxt = "n")
axis(side = 1, at = seq(0, 48, 12))
mtext("lag", side = 1, line = 1.8, cex = 0.8)

stats::pacf(diff.1.1, lag.max = 48, ylab = "", xlab = "", main = "", ylim = c(-0.6, 1), xaxt = "n")
axis(side = 1, at = seq(0, 48, 12))
mtext("lag", side = 1, line = 1.8, cex = 0.8)
```



Show the ACF and PACF for the d=1, D=0 case.

```
par(mfrow = c(1, 2), cex = 0.8, bty = "L", mar = c(3.6, 3, 1, 0.6), mgp = c(2.4, 1, 0), las = 1)
stats::acf(diff.1.0, lag.max = 48, ylab = "", xlab = "", main = "", ylim = c(-0.6, 1), xaxt = "n")
axis(side = 1, at = seq(0, 48, 12))
mtext("Sample ACF", side = 3, cex = 0.8)

stats::pacf(diff.1.0, lag.max = 48, ylab = "", xlab = "", main = "", ylim = c(-0.6, 1), xaxt = "n")
axis(side = 1, at = seq(0, 48, 12))
mtext("Sample PACF", side = 3, cex = 0.8)
```



A useful function for the model diagnostics (courtesy of Peter Craigmile at OSU)

```
plot.residuals <- function (x, y = NULL, lag.max = NULL, fitdf = NULL, mean.line = TRUE,
                  acf.ylim = c(-0.25, 1), mfrow = c(2, 2),
                              lags = NULL, ...){
  if (!is.null(mfrow))
    par(mfrow = mfrow)
  if (is.null(y)){
    y <- x
    x <- seq(length(y))
  } else {
    x <- as.numeric(x)
    y <- as.numeric(y)
  }

  if (is.null(lag.max)) {
    lag.max <-  floor(10 * log10(length(x)))
  }
  plot(x, y, type = "l", ...)
  if (mean.line) abline(h = 0, lty = 2)
```

```r
  qqnorm(y, main = "", las = 1); qqline(y)
  if (is.null(lags)) {
    acf(y, main = "", lag.max = lag.max, xlim = c(0, lag.max), ylim = acf.ylim,
        ylab = "sample ACF", las = 1)

    pacf(y, main = "", lag.max = lag.max, xlim = c(0, lag.max), ylim = acf.ylim,
         ylab = "sample PACF", las = 1)
  }
  else {
    stats::acf(y, main = "", lag.max = lag.max, xlim = c(0, lag.max), ylim = acf.ylim,
        ylab = "sample ACF", xaxt = "n", las = 1)
    axis(side = 1, at = lags)

    stats::pacf(y, main = "", lag.max = lag.max, xlim = c(0, lag.max), ylim = acf.ylim,
         ylab = "sample PACF", xaxt = "n", las = 1)
    axis(side = 1, at = lags)
  }
  Box.test(y, lag.max, type = "Ljung-Box", fitdf)
}
```

Fitting the SARIMA$(1,1,0) \times (1,0,0)$ model

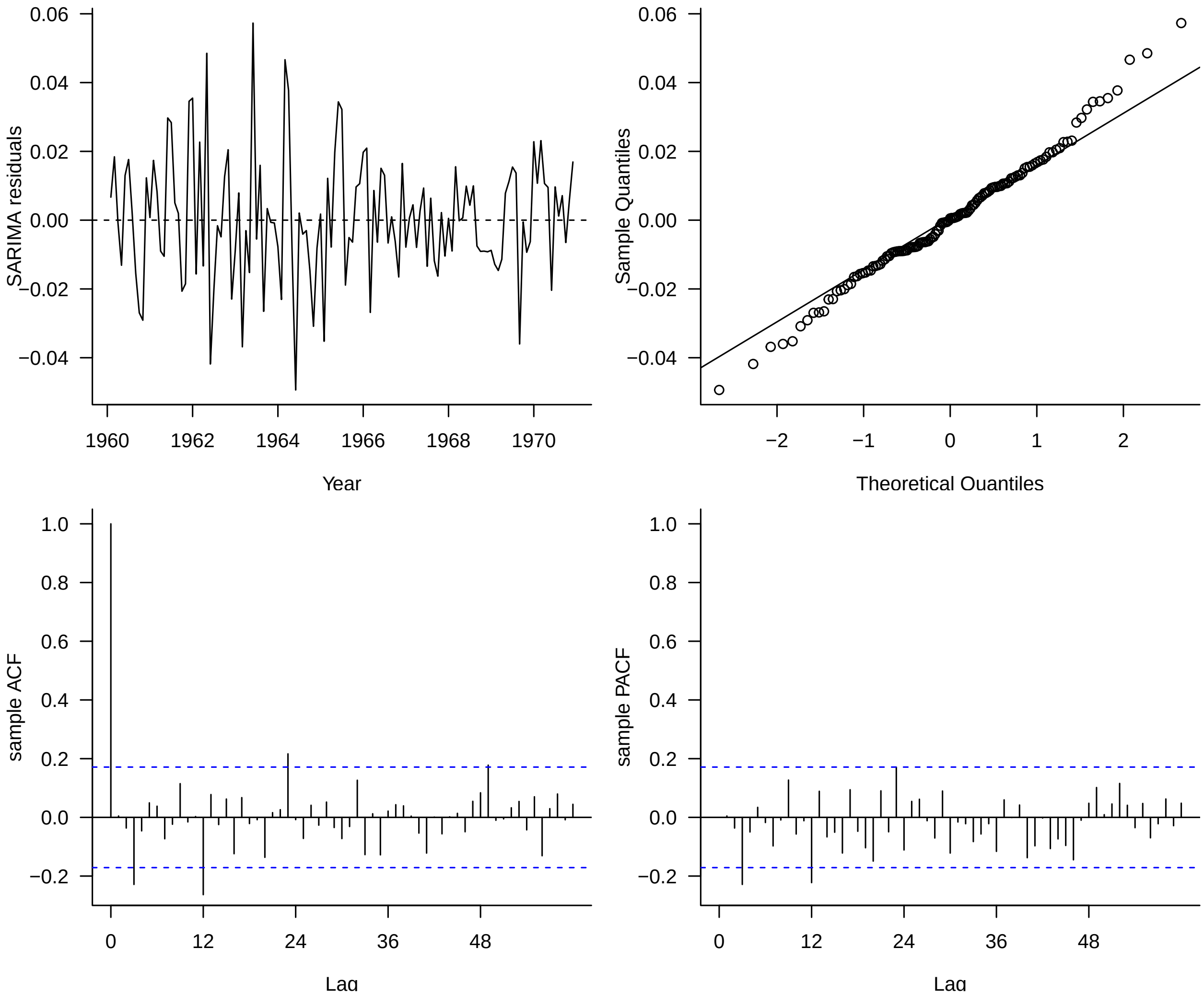
```r
(fit1 <- arima(diff.1.0, order = c(1, 0, 0), seasonal = list(order = c(1, 0, 0), period = 12)))
```

```
##
## Call:
## arima(x = diff.1.0, order = c(1, 0, 0), seasonal = list(order = c(1, 0, 0),
##     period = 12))
##
## Coefficients:
##           ar1    sar1  intercept
##       -0.2667  0.9291     0.0039
## s.e.   0.0865  0.0235     0.0096
##
## sigma^2 estimated as 0.0003298:  log likelihood = 327.27,  aic = -648.54
```

```r
Box.test(fit1$residuals, lag = 60, type = "Ljung-Box", fitdf = 3)
```

```
##
##  Box-Ljung test
##
## data:  fit1$residuals
## X-squared = 70.751, df = 57, p-value = 0.1042
```

```r
par(mfrow = c(2, 2), cex = 0.8, bty = "L", mar = c(3.6, 4, 0.8, 0.6),
    mgp = c(2.8, 1, 0), las = 1)
plot.residuals(shortyears[-1], resid(fit1), lag.max = 60, fitdf = 3,
               ylab = "SARIMA residuals", xlab = "Year", lags = seq(0, 48, 12))
```




```
##
##  Box-Ljung test
##
## data:  y
## X-squared = 70.751, df = 57, p-value = 0.1042
```

Fitting the SARIMA$(0,1,0) \times (1,0,0)$ model

```
(fit2 <- arima(diff.1.0, seasonal = list(order = c(1, 0, 0), period = 12)))
```

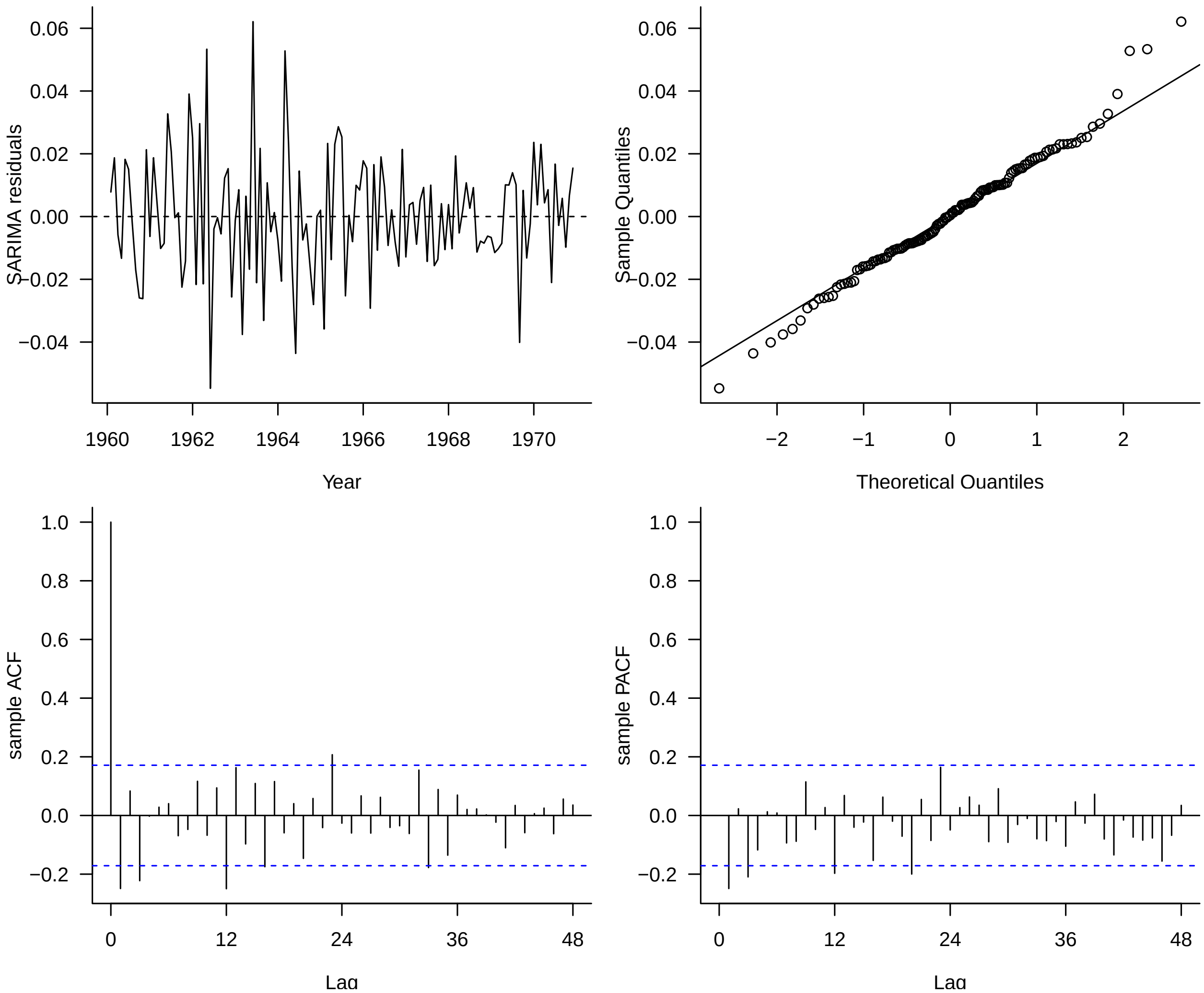
```
##
## Call:
## arima(x = diff.1.0, seasonal = list(order = c(1, 0, 0), period = 12))
##
## Coefficients:
##         sar1  intercept
##       0.9081     0.0040
## s.e.  0.0278     0.0108
```

```
##
## sigma^2 estimated as 0.0003616:  log likelihood = 322.75,  aic = -641.51
```

```r
Box.test(fit2$residuals, lag = 48, type = "Ljung-Box", fitdf = 2)
```

```
##
##  Box-Ljung test
##
## data:  fit2$residuals
## X-squared = 80.641, df = 46, p-value = 0.001201
```

```r
par(mfrow = c(2, 2), cex = 0.8, bty = "L", mar = c(3.6, 4, 0.8, 0.6),
    mgp = c(2.8, 1, 0), las = 1)
plot.residuals(shortyears[-1], resid(fit2), lag.max = 48, fitdf = 2,
               ylab = "SARIMA residuals", xlab = "Year", lags = seq(0, 48, 12))
```




```
##
##  Box-Ljung test
##
## data:  y
## X-squared = 80.641, df = 46, p-value = 0.001201
```

Forecasting 1971 Data

```r
## fit the first full model
fit1 <- arima(log.shortair, order = c(1, 1, 0),
                     seasonal = list(order = c(1, 0, 0), period = 12))
fit1
```

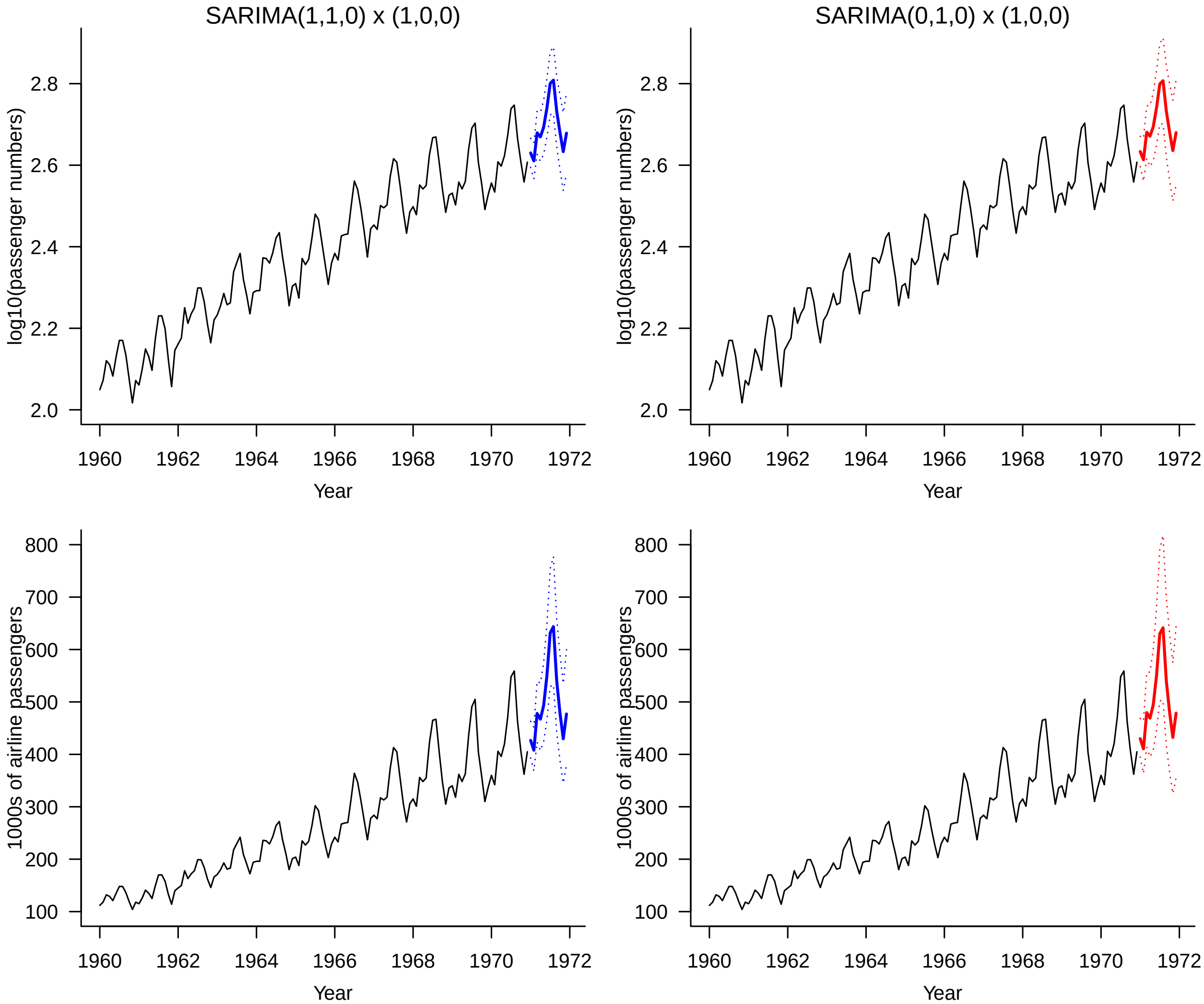
```
##
## Call:
## arima(x = log.shortair, order = c(1, 1, 0), seasonal = list(order = c(1, 0,
##     0), period = 12))
##
## Coefficients:
##           ar1    sar1
##       -0.2665  0.9298
## s.e.   0.0866  0.0233
##
## sigma^2 estimated as 0.0003299:  log likelihood = 327.19,  aic = -650.38
```

```r
## fit the second full model
fit2 <- arima(log.shortair, order = c(0, 1, 0),
                     seasonal = list(order = c(1, 0, 0), period = 12))
fit2
```

```
##
## Call:
## arima(x = log.shortair, order = c(0, 1, 0), seasonal = list(order = c(1, 0,
##     0), period = 12))
##
## Coefficients:
##         sar1
##       0.9088
## s.e.  0.0276
##
## sigma^2 estimated as 0.0003617:  log likelihood = 322.69,  aic = -643.38
```

```r
## define the forecasting time points
fyears <- yr[133:144]

preds1 <- predict(fit1, 12)
forecast1 <- preds1$pred
flimits1  <- qnorm(0.975) * preds1$se

preds2 <- predict(fit2, 12)
forecast2 <- preds2$pred
flimits2  <- qnorm(0.975) * preds2$se

par(mfrow = c(2, 2), cex = 0.8, bty = "L", mar = c(3.6, 4, 1, 0.6),
    mgp = c(2.4, 1, 0), las = 1)

plot(shortyears, log.shortair, type = "l", xlab = "Year",
     ylab = "log10(passenger numbers)", xlim = range(yr), ylim = c(2, 2.9))
```

```r
mtext("SARIMA(1,1,0) x (1,0,0)")
## plots the forecasts
lines(fyears, forecast1, lwd = 2, col = "blue")
## plot the 95% prediction intervals.
lines(fyears, forecast1 + flimits1, lty = 3, col = "blue")
lines(fyears, forecast1 - flimits1, lty = 3, col = "blue")

plot(shortyears, log.shortair, type = "l", xlab = "Year",
     ylab = "log10(passenger numbers)", xlim = range(yr), ylim = c(2, 2.9))
mtext("SARIMA(0,1,0) x (1,0,0)")
## plots the forecasts
lines(fyears, forecast2, lwd = 2, col = "red")
## plot the 95% prediction intervals.
lines(fyears, forecast2 + flimits2, lty = 3, col = "red")
lines(fyears, forecast2 - flimits2, lty = 3, col = "red")

plot(shortyears, 10^log.shortair, type = "l", xlab = "Year",
     ylab="1000s of airline passengers", xlim = range(yr), ylim = c(100, 800))
lines(fyears, 10^forecast1, lwd = 2, col = "blue")
lines(fyears, 10^(forecast1 + flimits1), lty = 3, col = "blue")
lines(fyears, 10^(forecast1 - flimits1), lty = 3, col = "blue")

plot(shortyears, 10^log.shortair, type = "l", xlab = "Year",
     ylab="1000s of airline passengers", xlim = range(yr), ylim = c(100, 800))
lines(fyears, 10^forecast2, lwd = 2, col = "red")
lines(fyears, 10^(forecast2 + flimits2), lty = 3, col = "red")
lines(fyears, 10^(forecast2 - flimits2), lty = 3, col = "red")
```



Evaluating Forecast Performance

```
## calculate the root mean square error (RMSE)
sqrt(mean((10^forecast1 - 10^log.airpass[133:144])^2))
```

```
## [1] 30.36384
```

```
sqrt(mean((10^forecast2 - 10^log.airpass[133:144])^2))
```

```
## [1] 31.32376
```

```
## calculate the mean relative prediction error.
mean((10^forecast1 - 10^log.airpass[133:144]) / 10^log.airpass[133:144])
```

```
## [1] 0.05671086
```

```r
mean((10^forecast2 - 10^log.airpass[133:144]) / 10^log.airpass[133:144])
```

```
## [1] 0.05951677
```

```r
## calculate the empirical coverage rate
PI_fit1 <- cbind(as.numeric(10^(forecast1 + flimits1)),
                 as.numeric(10^(forecast1 - flimits1)))
sum(apply(PI_fit1 - 10^log.airpass[133:144], 1, prod) < 0) / length(10^log.airpass[133:144])
```

```
## [1] 0.9166667
```

```r
PI_fit2 <- cbind(as.numeric(10^(forecast2 + flimits2)),
                 as.numeric(10^(forecast2 - flimits2)))
sum(apply(PI_fit2 - 10^log.airpass[133:144], 1, prod) < 0) / length(10^log.airpass[133:144])
```

```
## [1] 1
```