MATH 8090:

Whitney Huang, Clemson University

9/21-9/23/2021

Contents

NOAA wind data example

This example is taken from Don Percival's time series course (UW Stat 519).

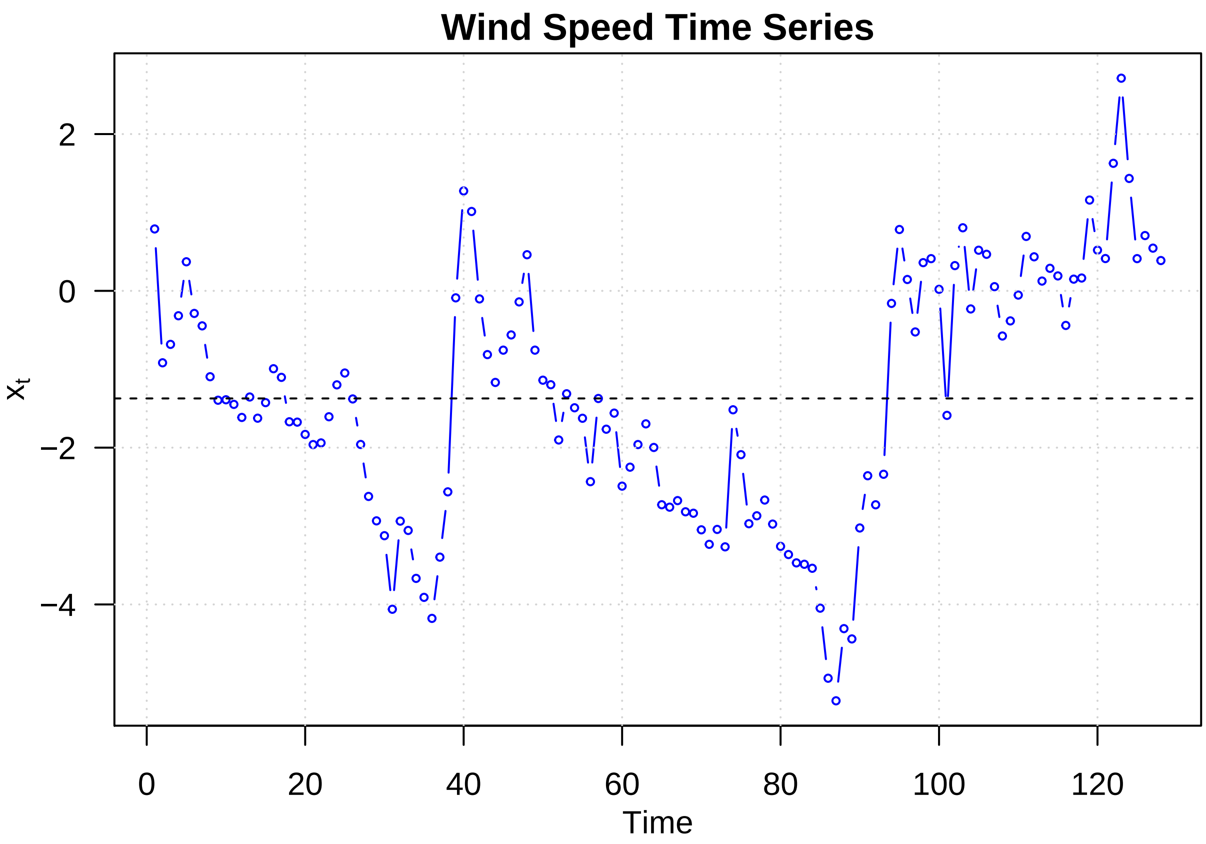
The one-step-ahead forecast of an AR(1) process is:

$$P_n X_{n+1} = \hat{\mu} + \hat{\phi}(X_n - \hat{\mu}),$$

where $\hat{\phi}$ is our estimate of ϕ, and $\hat{\mu}$ is an estimate of μ.

Load and plot the data

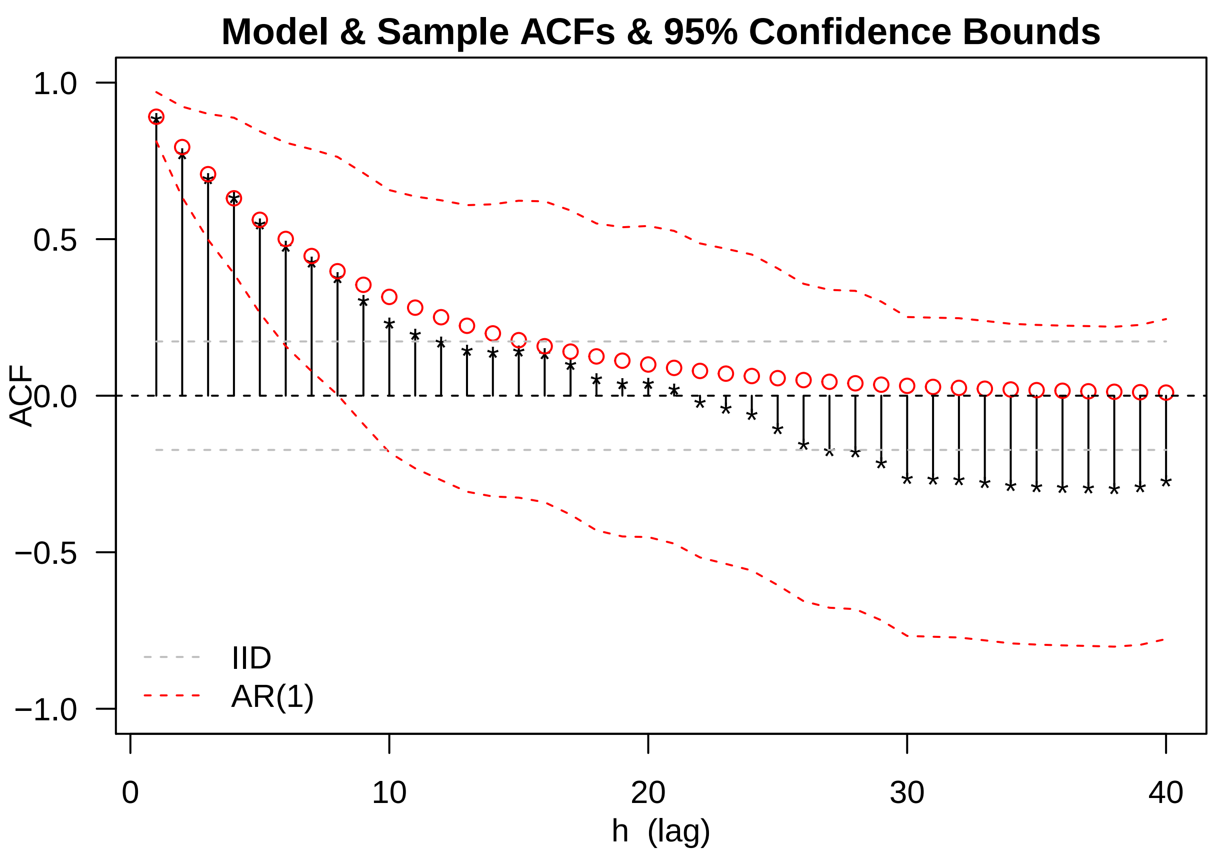
```
ws <- scan("http://faculty.washington.edu/dbp/s519/Data/wind-speed.txt")
n <- length(ws)
par(las = 1, mgp = c(2, 1, 0), mar = c(3.6, 3.6, 1.4, 0.6))
plot(ws, col = "blue", xlab = "Time", typ = "b",
     ylab = expression(x[t]), main = "Wind Speed Time Series", cex = 0.5)
grid()
xbar_ws <- mean(ws)
abline(h = xbar_ws, lty = 2)
```



"Estimate" ϕ using sample ACF and center the data

```
acf.ws <- acf(ws, lag.max = 40, plot = FALSE)$acf
phi.ws <- acf.ws[2] # this is an estimate for the coefficient of AR(1)

gen.whh.ar <- function(h, phi){
    p.2 <- phi^2; p.2h <- p.2^h
    - 2 * h * p.2h + (1 - p.2h) * (1 + p.2) / (1 - p.2)
}

plot.ACFbartlettAR <- function(ts, n.lags = 40){
    n.ts <- length(ts)
    lags <- 1:n.lags
    acf.est <- acf(ts, lag.max = n.lags, plot = FALSE)$acf[-1]
    acf.model <- acf.est[1]^lags
    plot(lags, acf.est, type = "h", xlab = "h  (lag)",
```

```
        ylab = "ACF", ylim = c(-1, 1),
        main = "Model & Sample ACFs & 95% Confidence Bounds", las = 1)
    points(lags, acf.est, pch = "*")
    points(lags, acf.model, col = "red")
    CI.AR <- 1.96 * sqrt(sapply(lags, function(h) gen.whh.ar(h, acf.est[1]))) / sqrt(n.ts)
    lines(lags, acf.est + CI.AR, col = "red", lty = 2)
    lines(lags, acf.est - CI.AR, col = "red", lty = 2)
    abline(h = 0, lty = "dashed")
    CI.IID <- rep(1.96 / sqrt(n), n.lags)
    lines(lags, -CI.IID, col = "gray", lty = 2)
    lines(lags, CI.IID, col = "gray", lty = 2)
    legend("bottomleft", legend = c("IID", "AR(1)"), lty = "dashed",
           col = c("gray", "red"), bty = "n")
}

par(mgp = c(2, 1, 0), mar = c(3.5, 3.5, 1.4, 0.6))
plot.ACFbartlettAR(ws)
```




```
## Altyernatively, we can estiamte phi using MLE
(phi_hat <- arima(ws, order = c(1, 0, 0)))
```

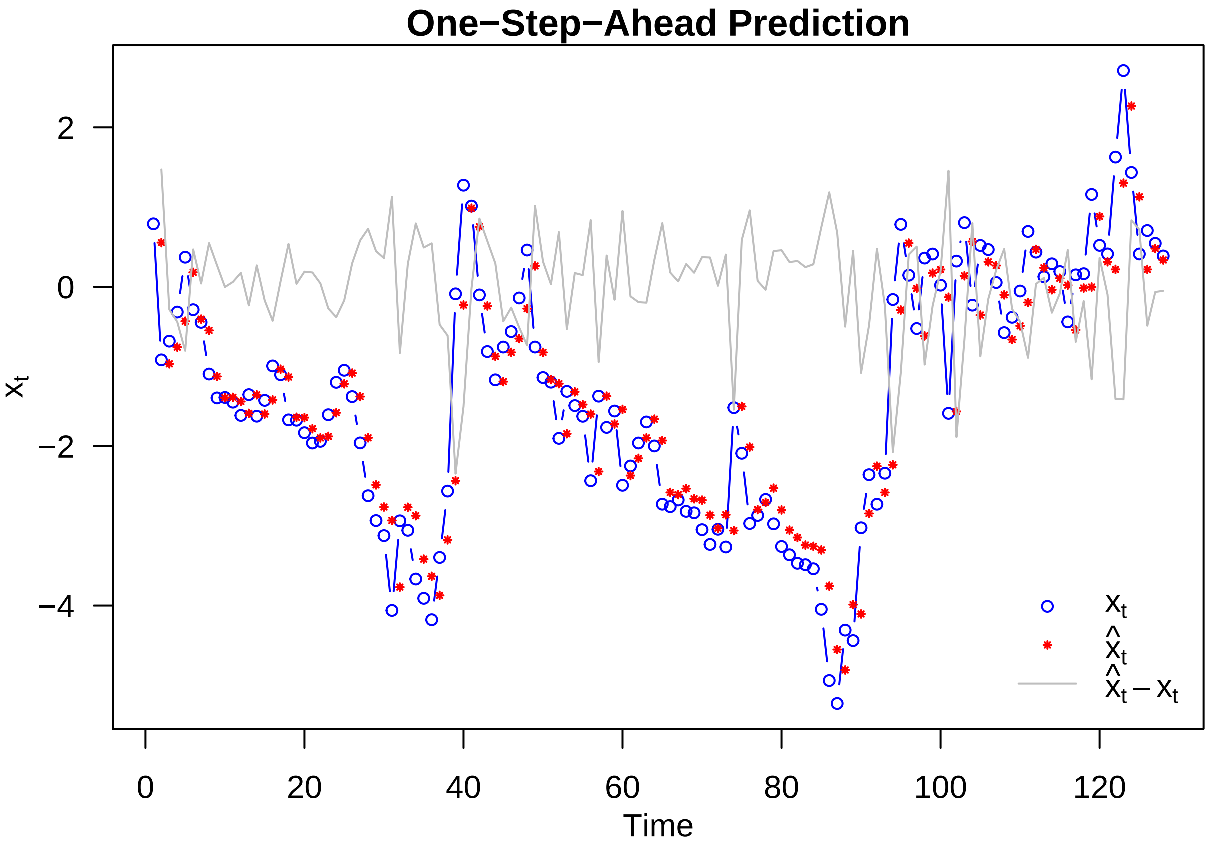
```
##
## Call:
## arima(x = ws, order = c(1, 0, 0))
##
## Coefficients:
##          ar1  intercept
```

```
##        0.906    -1.1136
## s.e.   0.037     0.6035
##
## sigma^2 estimated as 0.4615:  log likelihood = -132.99,  aic = 271.99
```

```
ws.centered <- ws - xbar_ws
```

One-step-ahead forecast

```
ws.hat <- phi.ws * ws.centered[1:(n - 1)] + xbar_ws
## prediction errors
zt.ws <- ws.hat - ws[2:n]
## plot it
par(las = 1, mgp = c(2, 1, 0), mar = c(3.5, 3.5, 1.2, 0.6))
plot(ws, col = "blue", xlab = "Time", type = "b", ylab = expression(x[t]),
     main = "One-Step-Ahead Prediction", cex = 0.75)
points(2:n, ws.hat, pch = 8, col = "red", cex = 0.375)
lines(2:n, zt.ws, col = "gray")
legend("bottomright", legend = expression(x[t], hat(x)[t], hat(x)[t] - x[t]),
       col = c("blue", "red", "gray"), pch = c(1, 8, NA),
       lty = c(NA, NA, "solid"), pt.cex = c(0.75, 0.375, 1), inset = 0.01,
       bty = "n")
```




```
var(zt.ws) # sample prediction variance
```

```
## [1] 0.4629379
```

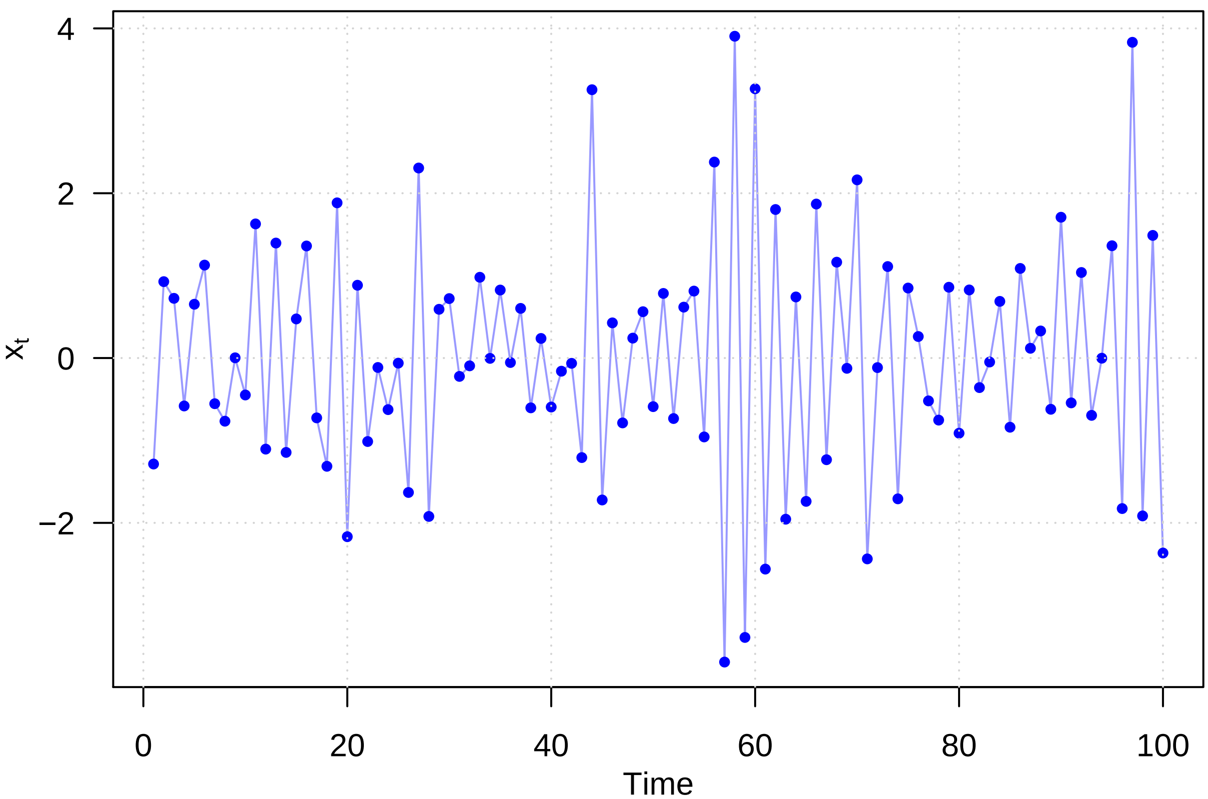
```r
var(ws) # sample variance
```

```
## [1] 2.50251
```

Fill in missing value example

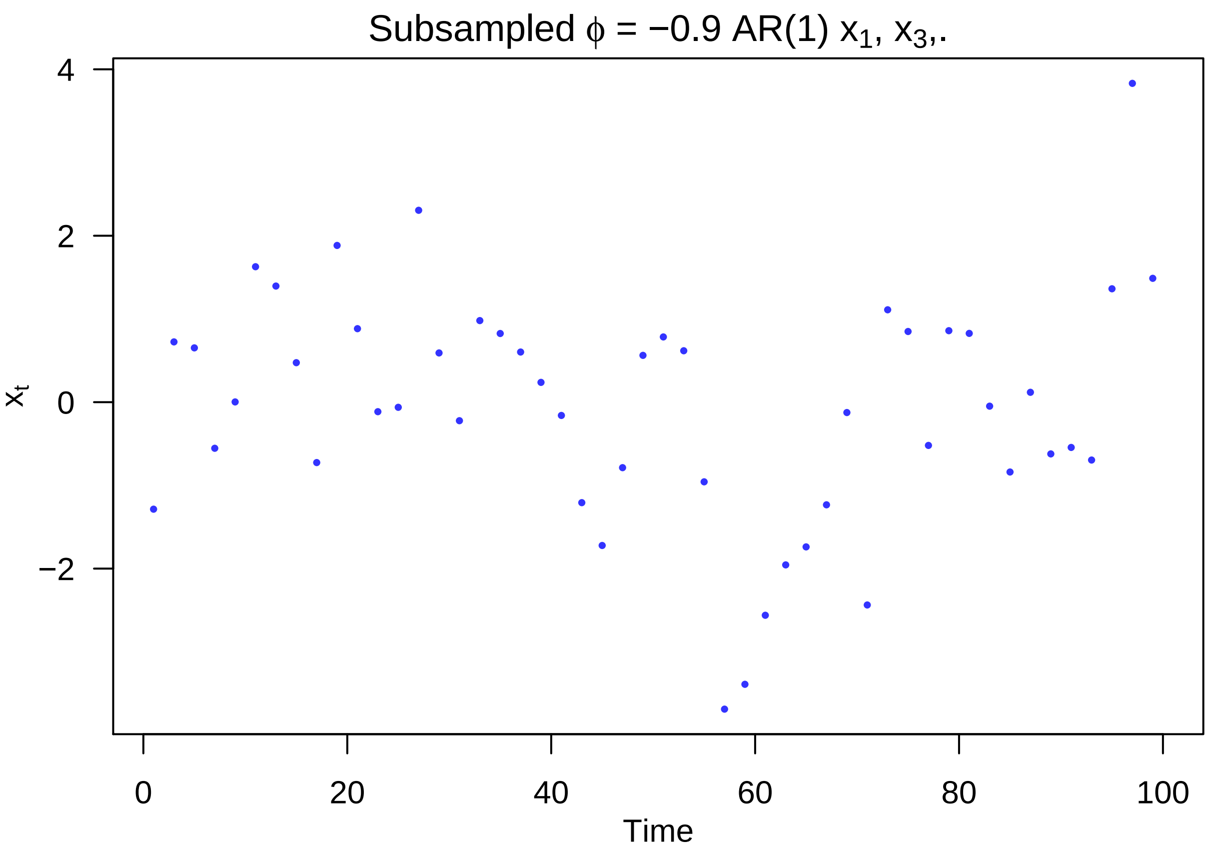
Simulate an AR(-0.9)

```r
generate.AR1.ts <- function(phi = 0.0){
    ts <- rep(0, 100)
    ts[1] <- rnorm(1) / sqrt(1 - phi^2)
    for(i in 2:100) ts[i] <- phi * ts[i - 1] + rnorm(1)
    ts
}
set.seed(123)
ar1.ts <- generate.AR1.ts(-0.9)

library(scales)
par(las = 1, mgp = c(2, 1, 0), mar = c(3.5, 3.5, 1.4, 0.6))
plot(ar1.ts, col = alpha("blue", 0.4), xlab = "Time", type = "l",
     ylab = expression(x[t]), cex = 0.5)
points(ar1.ts, pch = 16, cex = 0.75, col = "blue")
grid()
```



Let's remove some data to illustrate how to fill in missing values using forecasting algorithm

```
ar1.ts.subsampled <- ar1.ts
ar1.ts.subsampled[seq(2, 100, 2)] <- NA
par(las = 1, mgp = c(2, 1, 0), mar = c(3.5, 3.5, 1.4, 0.6))
plot(ar1.ts.subsampled, xlab = "Time", type = "b", ylab = expression(x[t]),
     main = expression(paste("Subsampled ", phi, " = -0.9 AR(1) ", x[1], ", ",x[3], ",.")),
     cex = 0.5, col = alpha("blue", 0.8), pch = 16)
```



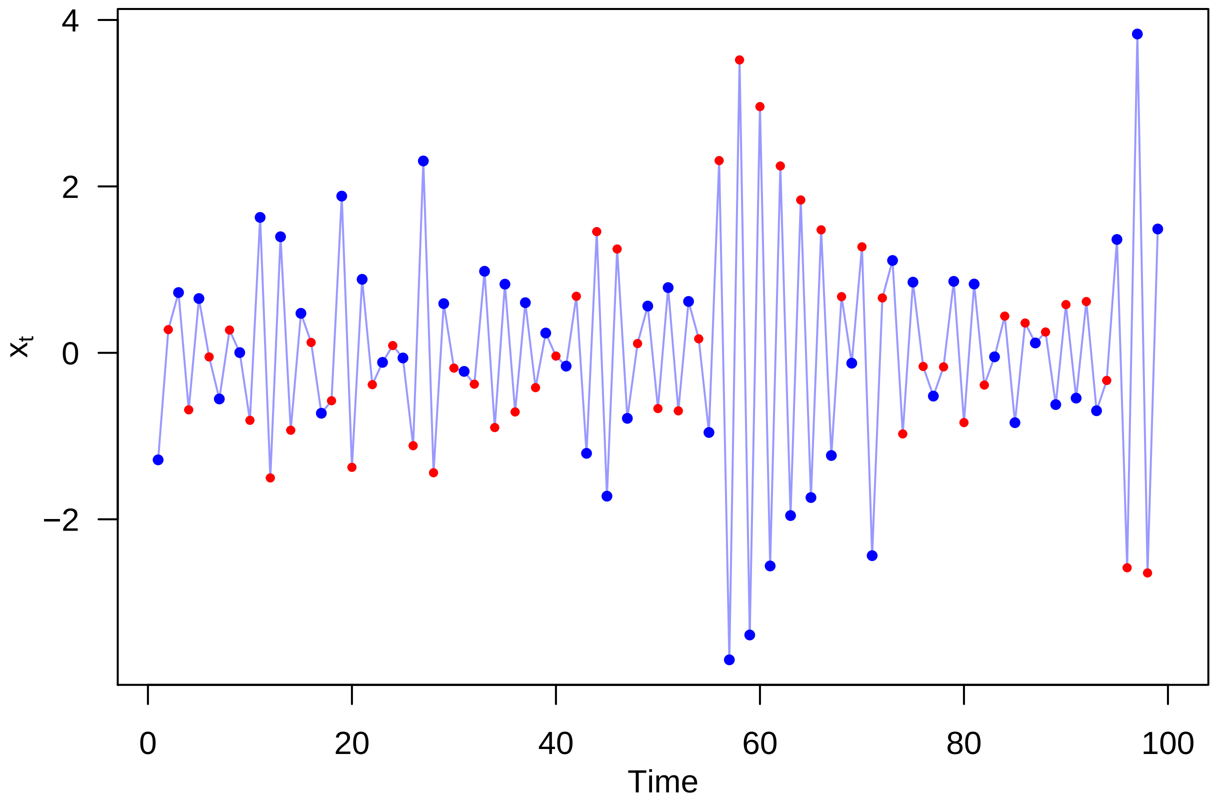
Fill in "missing" values

$\hat{X}_2 = \phi(X_1 + X_3)/(1 + \phi^2)$

$\text{MSPE} = \frac{\sigma^2}{1+\phi^2}$

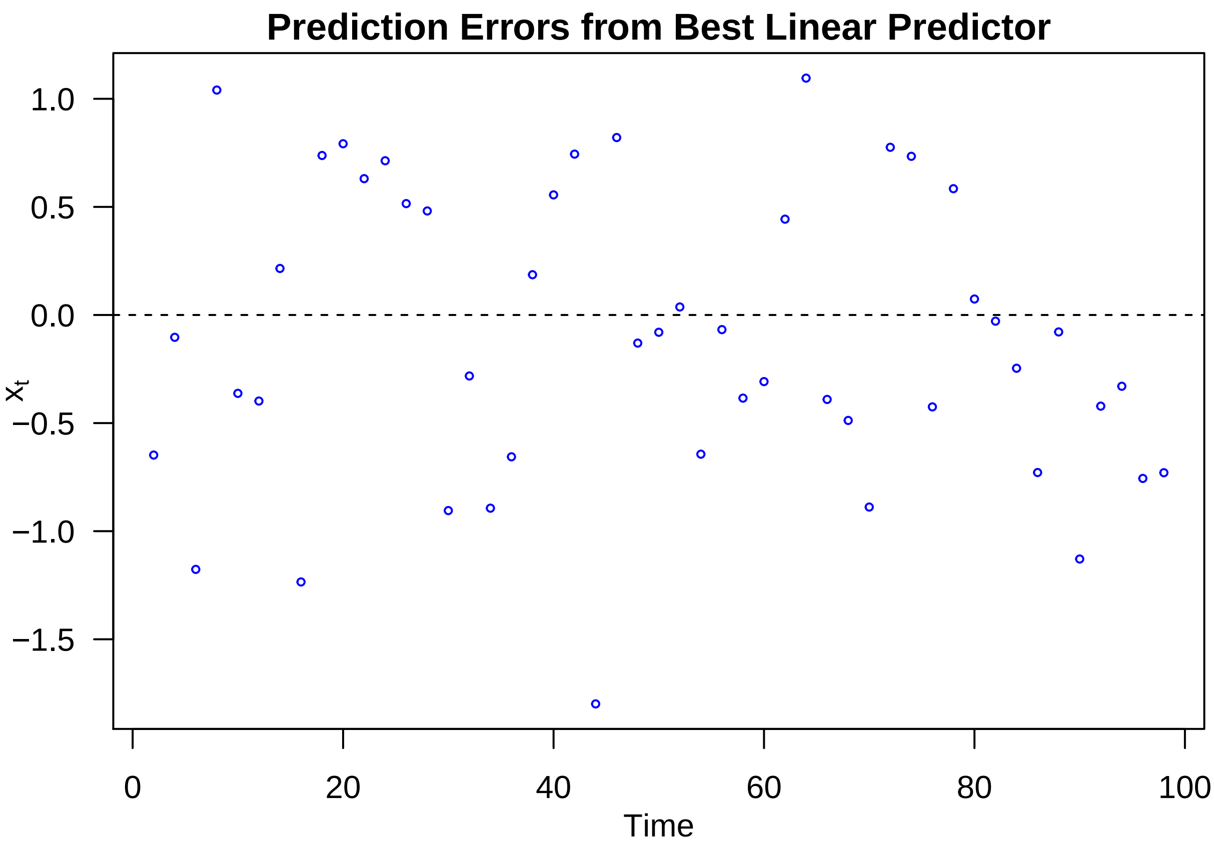
```
ar1.ts.predicted <- ar1.ts
ar1.ts.predicted[seq(2, 98, 2)] <- -0.9 * (ar1.ts[seq(1, 97, 2)] + ar1.ts[seq(3, 99, 2)]) / 1.81
ar1.ts.predicted[100] <- NA


par(las = 1, mgp = c(2, 1, 0), mar = c(3.5, 3.5, 1.4, 0.6))
plot(ar1.ts.predicted, col = alpha("blue", 0.4), xlab = "Time", type = "l",
     ylab = expression(x[t]), cex = 0.5)
xs <- seq(2, 98, 2)
points(xs, ar1.ts.predicted[xs], pch = 19, col = "red", cex = 0.5)
```

```r
xo <- seq(1, 99, 2)
points(xo, ar1.ts.subsampled[xo], col = "blue", cex = 0.75, pch = 16)
```



Prediction Errors from Best Linear Predictor

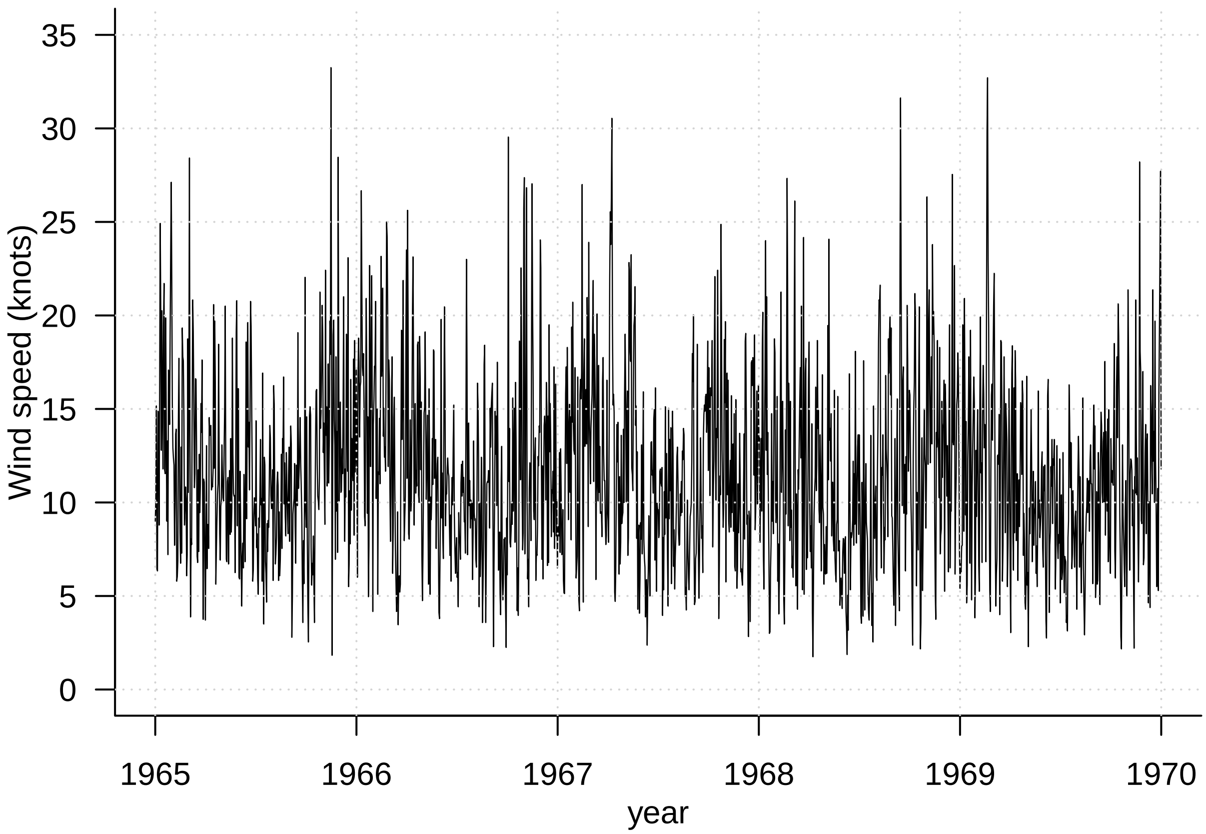
```r
par(las = 1, mgp = c(2, 1, 0), mar = c(3.5, 3.5, 1.4, 0.6))
plot(xs, (ar1.ts.predicted - ar1.ts)[xs], col = "blue", xlab = "Time", type = "p",
     ylab = expression(x[t]), main = "Prediction Errors from Best Linear Predictor",
     cex=0.5)
abline(h = 0, lty = 2)
```



Ireland wind data case study

Load and plot the data

```
rosslare <- scan("http://www.stat.osu.edu/~pfc/teaching/6550/datasets/daily_rosslare.txt")
## set up  the year variable
year <- seq(from = 1965, by = 1 / 365.25, length = length(rosslare))
par(bty = "L", mar = c(3.6, 3.6, 0.5, 0.6), mgp = c(2, 1, 0), las = 1)
plot(year, rosslare, type = "l", ylim = c(0, 35), lwd = 0.6,
     xlab = "year", ylab = "Wind speed (knots)")
grid()
```



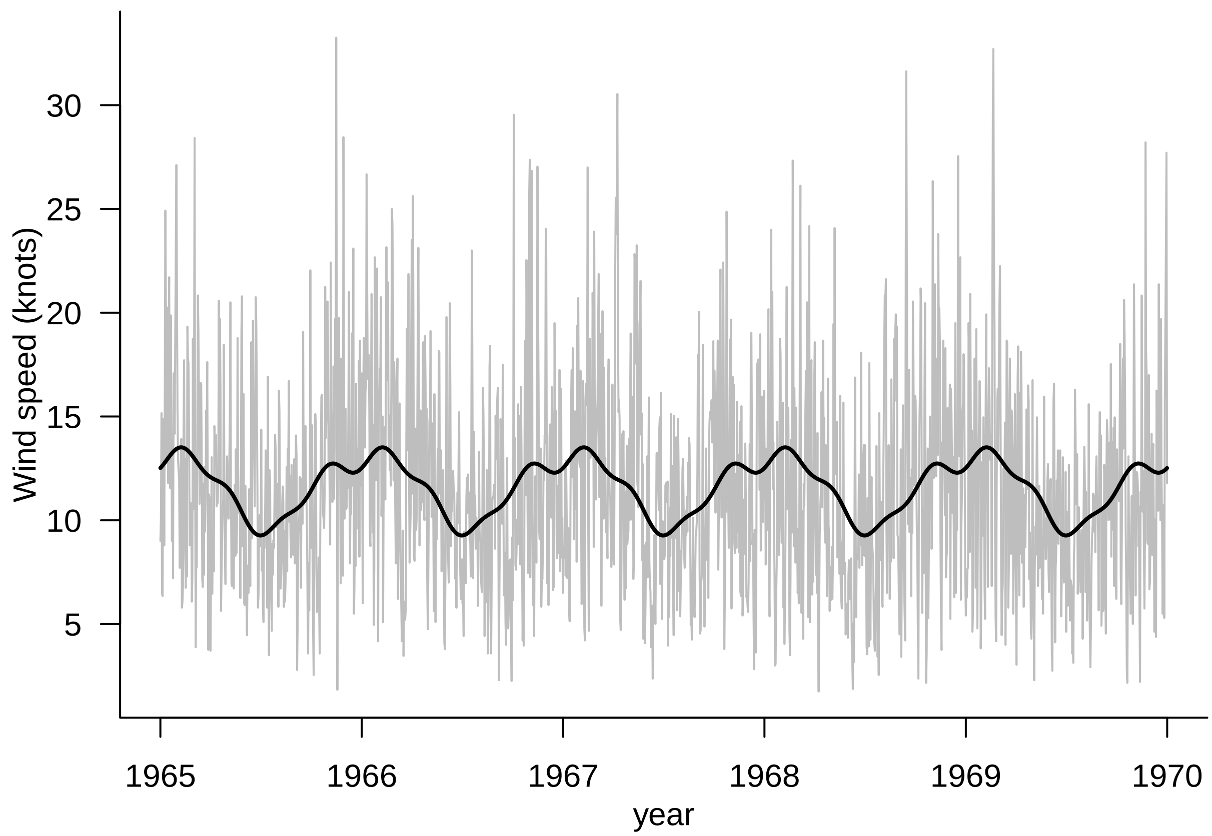
Deseasonalization: Harmonic Regression

```
## create harmonic terms
Harmonic <- function(year, K){
  t <- outer(2 * pi * year, 1:K)
  return(cbind(apply(t, 2, cos), apply(t, 2, sin)))
}
harmonics <- Harmonic(year, 4)
## fit a harmonic regression
harm.model <- lm(rosslare ~ harmonics)
summary(harm.model)
```

```
##
## Call:
## lm(formula = rosslare ~ harmonics)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -10.854  -3.378  -0.492   2.839  20.829
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 11.583744   0.112316 103.135  < 2e-16 ***
## harmonics1   1.686674   0.158807  10.621  < 2e-16 ***
## harmonics2  -0.436067   0.158807  -2.746  0.00609 **
## harmonics3  -0.060841   0.158807  -0.383  0.70168
```

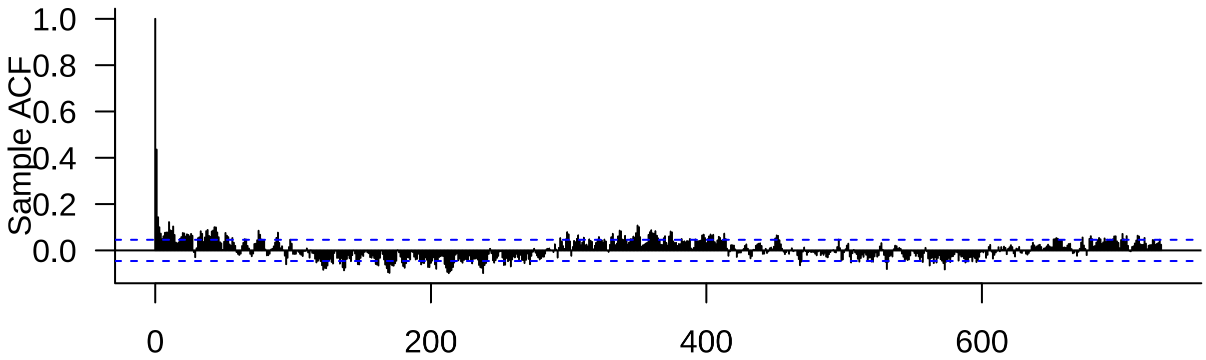
```
## harmonics4  -0.252191   0.158807  -1.588  0.11245
## harmonics5   0.412367   0.158872   2.596  0.00952 **
## harmonics6   0.003881   0.158872   0.024  0.98051
## harmonics7   0.107255   0.158872   0.675  0.49970
## harmonics8   0.217883   0.158872   1.371  0.17041
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 4.801 on 1818 degrees of freedom
## Multiple R-squared:  0.0677, Adjusted R-squared:  0.0636
## F-statistic:  16.5 on 8 and 1818 DF,  p-value: < 2.2e-16
```

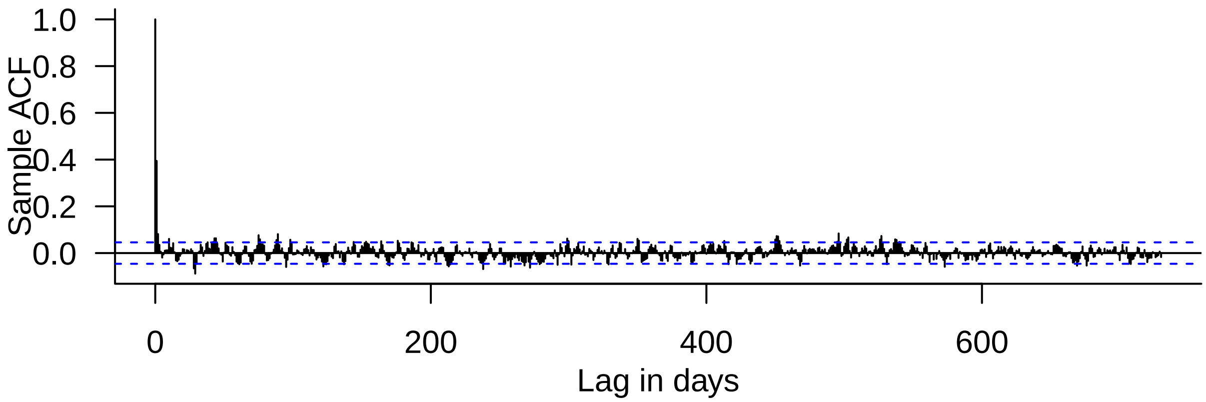
```r
par(bty = "L", mar = c(3.6, 3.6, 0.5, 0.6), mgp = c(2, 1, 0), las = 1)
plot(year, rosslare, type = "l",
     xlab = "year", ylab = "Wind speed (knots)", col = "grey")
lines(year, fitted(harm.model), lwd = 2)
```



ACF Plots: Original and Deseasonalized Series

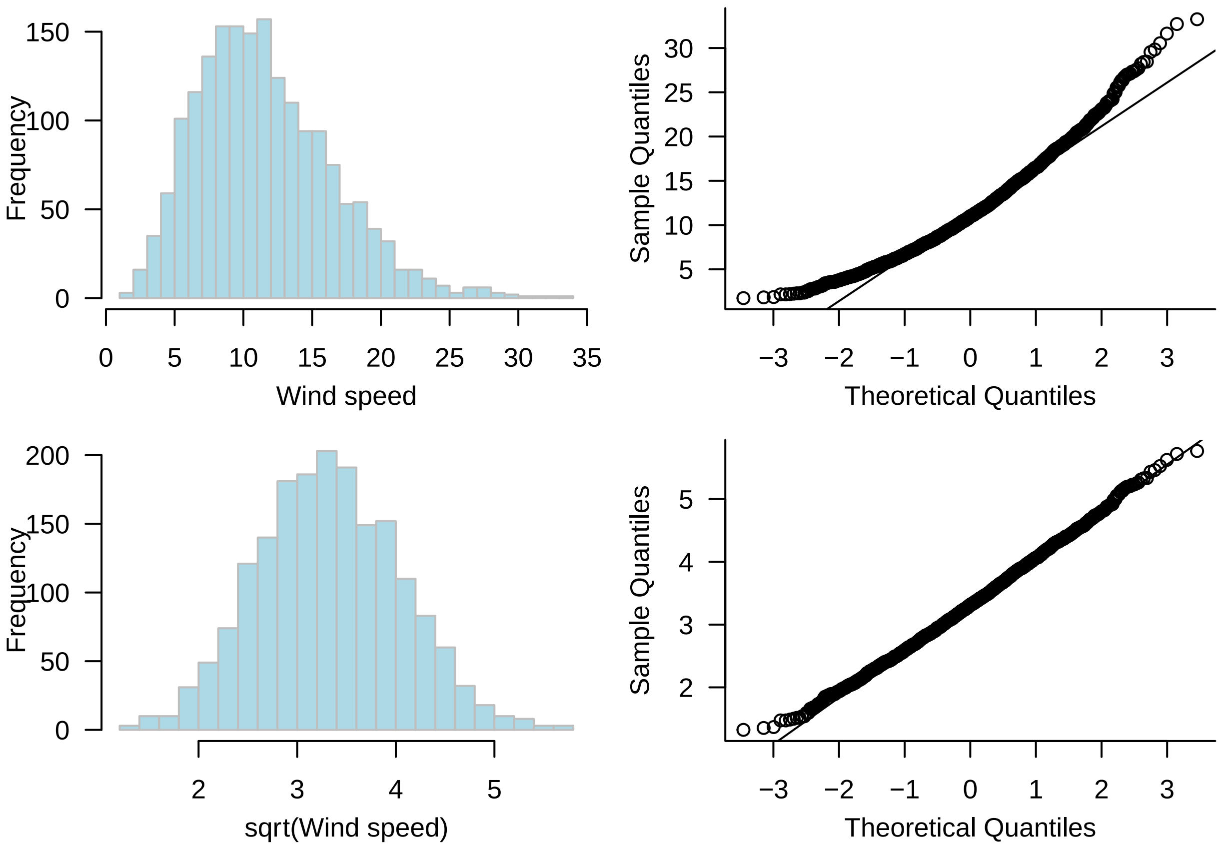
```r
par(bty = "L", mar = c(3.6, 3.6, 0.5, 0.6), mgp = c(2, 1, 0), las = 1,
    mfrow = c(2, 1))
acf(rosslare, lag.max = 365 * 2, xlab = "", ylab = "Sample ACF", main = "")
acf(resid(harm.model), lag.max = 365 * 2, xlab = "Lag in days",
    ylab = "Sample ACF", main = "")
```





Apply transformation to make wind speed more Gaussian like

```
## Now look at a histogram of the values, along with the
## normal quantile-quantile plot.
par(mfrow = c(2, 2), bty = "L", mar = c(3.6, 3.6, 0.5, 0.6), las = 1,
    mgp = c(2.2, 1, 0))
hist(rosslare, 40, main = "", xlab = "Wind speed", col = "lightblue", border = "gray")
qqnorm(rosslare, main = "")
qqline(rosslare)
## Histogram/Q-Q plot of 1/2 root transformation
hist(sqrt(rosslare), 30, main = "", xlab = "sqrt(Wind speed)", col = "lightblue", border = "gray")
qqnorm(sqrt(rosslare), main=""); qqline(sqrt(rosslare))
```



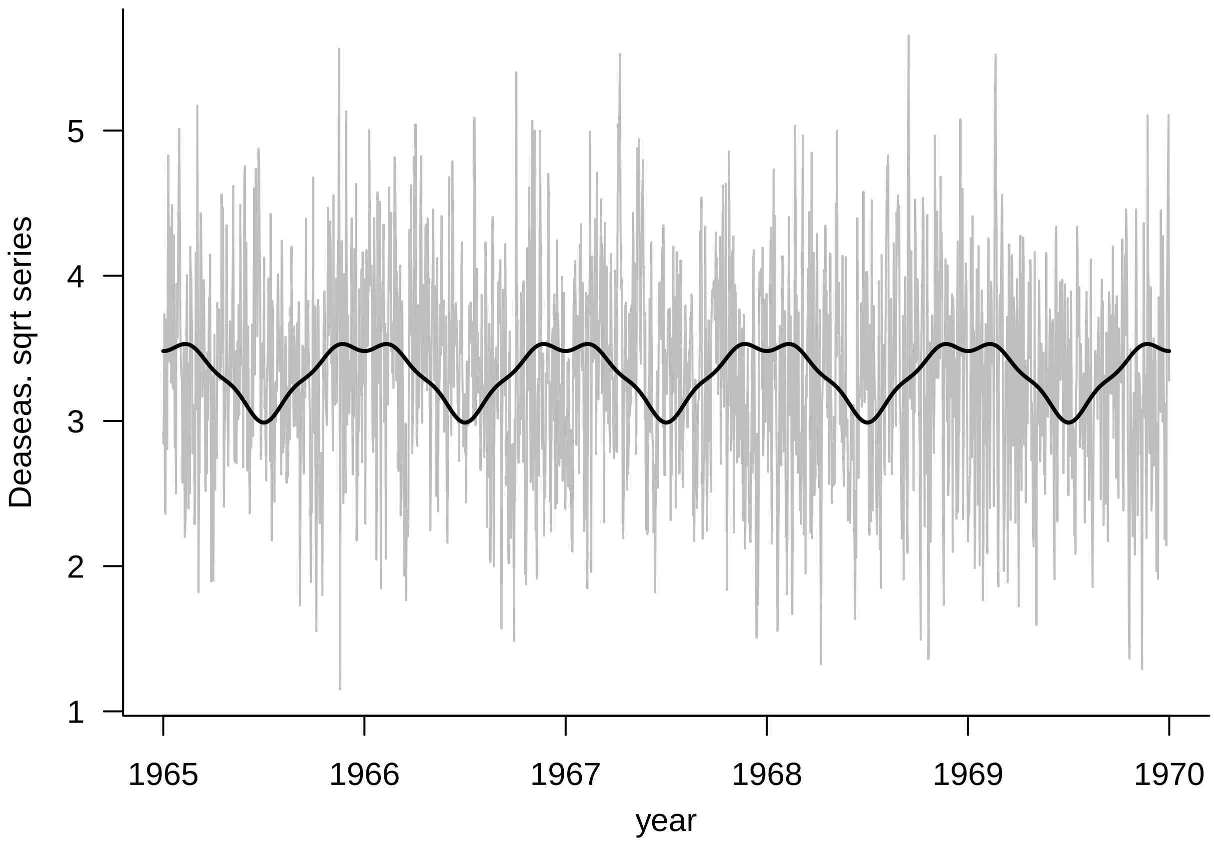
Now take square roots of the original data and deseasonalizeagain!

```r
## now we start again from the beginning with a sqrt transformation
sqrt.rosslare <- sqrt(rosslare)
## refit the periodicity, without the intercept term
harm.model <- lm(sqrt.rosslare ~ harmonics[, 1:4] - 1)
summary(harm.model)
```

```
##
## Call:
## lm(formula = sqrt.rosslare ~ harmonics[, 1:4] - 1)
##
## Residuals:
##    Min     1Q Median     3Q    Max
##  1.149  2.846  3.317  3.800  5.656
##
## Coefficients:
##                    Estimate Std. Error t value Pr(>|t|)
## harmonics[, 1:4]1  0.242704   0.112557   2.156   0.0312 *
## harmonics[, 1:4]2 -0.057059   0.112557  -0.507   0.6123
## harmonics[, 1:4]3  0.003434   0.112557   0.031   0.9757
## harmonics[, 1:4]4 -0.032795   0.112557  -0.291   0.7708
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3.403 on 1823 degrees of freedom
```

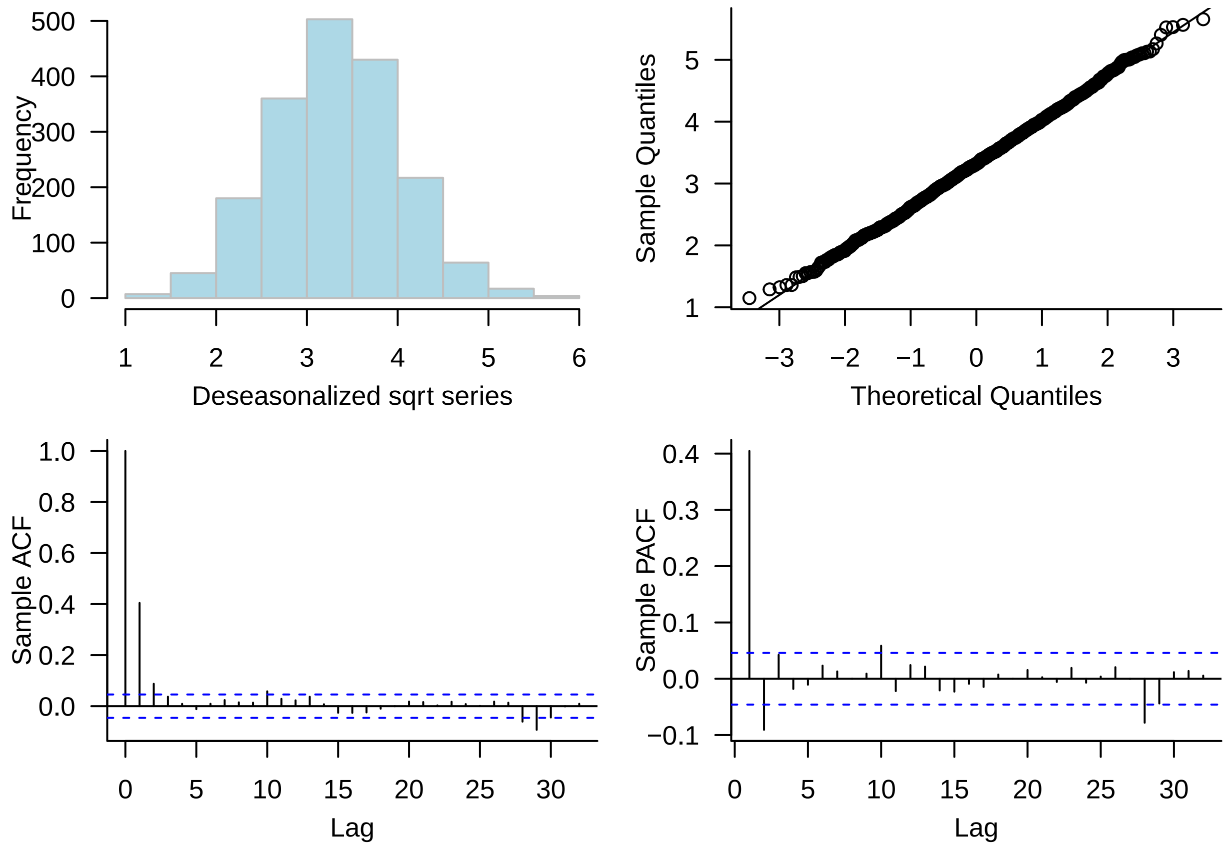
```
## Multiple R-squared:  0.00273,    Adjusted R-squared:  0.0005415
## F-statistic: 1.247 on 4 and 1823 DF,  p-value: 0.2888
```

```r
## calculate the estimate of the deseasonalized series
sqrt.rosslare.ds <- resid(harm.model)
## Produce time series plots of the sqrt data
par(bty = "L", mar = c(3.6, 3.6, 0.5, 0.6), las = 1, mgp = c(2.2, 1, 0))
plot(year, sqrt.rosslare.ds, type = "l", col = "gray",
     xlab = "year", ylab = "Deaseas. sqrt series")
lines(year, fitted(harm.model) + mean(sqrt.rosslare.ds), lwd = 2)
```



Checking Normality ACF/PACF

```r
## And check the distribution
par(bty = "L", mar = c(3.6, 3.6, 0.5, 0.6), las = 1, mgp = c(2.2, 1, 0),
    mfrow = c(2, 2))
hist(sqrt.rosslare.ds, main = "", xlab = "Deseasonalized sqrt series",
     col = "lightblue", border = "gray")
qqnorm(sqrt.rosslare.ds, main="")
qqline(sqrt.rosslare.ds)
## Now let's examine the sample ACF and PACF
acf(sqrt.rosslare.ds, main = "", ylab = "Sample ACF")
pacf(sqrt.rosslare.ds, main = "", ylab = "Sample PACF")
```



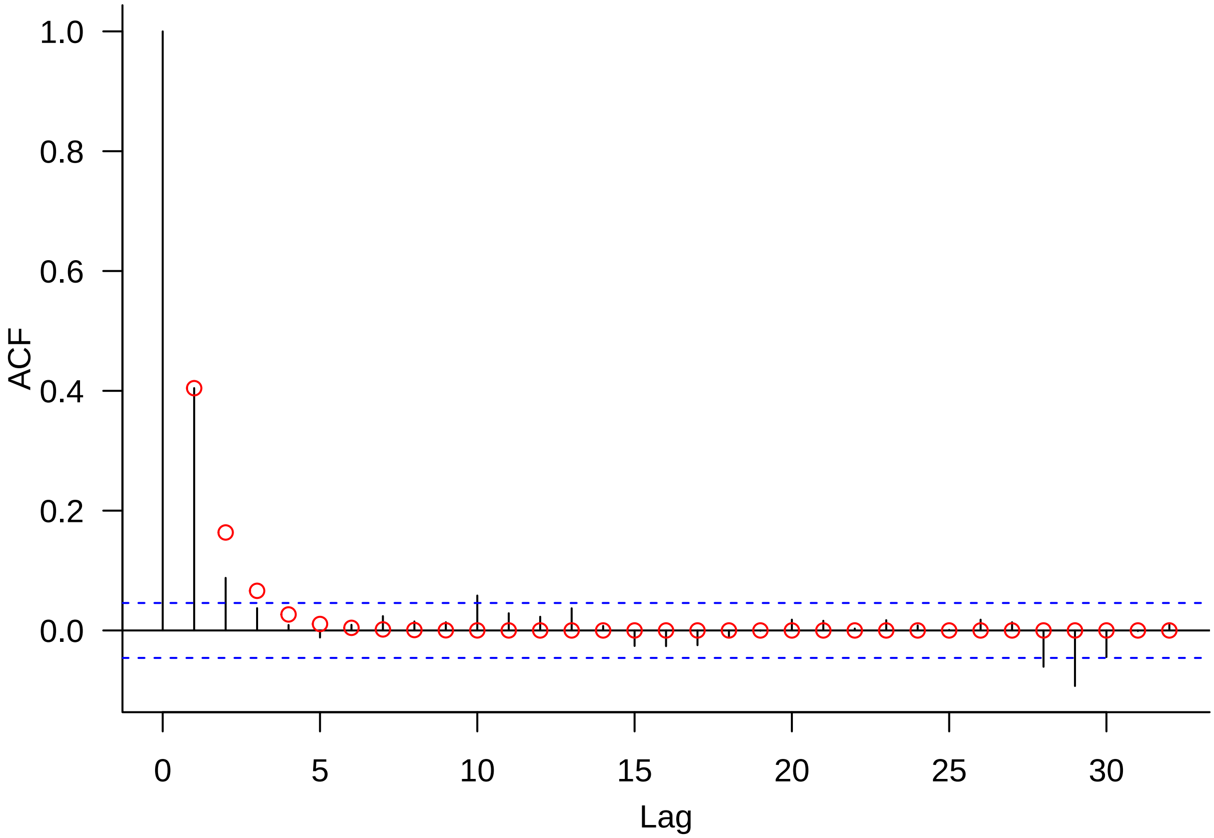
Model identifcation, fitting, and selection

Let's first fit an AR(1)

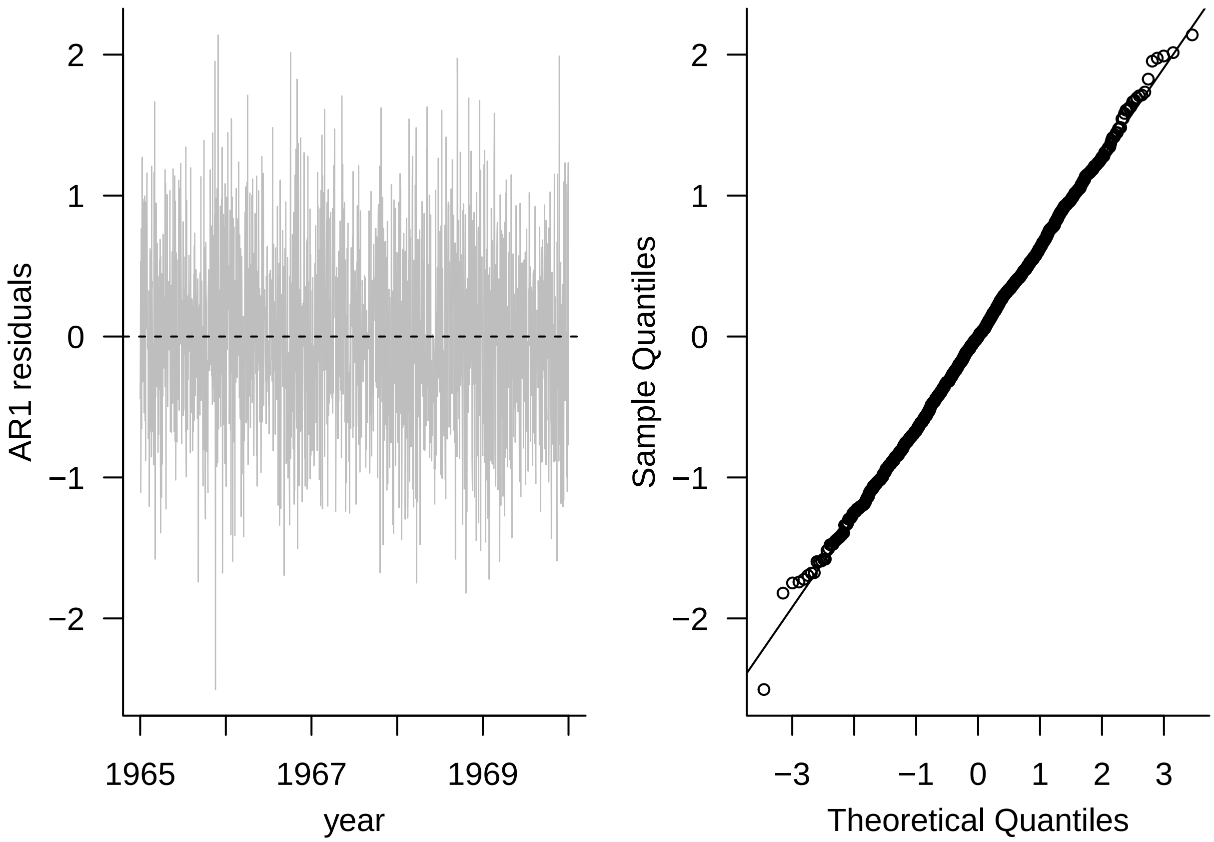
```r
## Fit an AR(1) model
ar1.model <- arima(sqrt.rosslare.ds, order = c(1, 0, 0))
## summarize the model
ar1.model
```

```
##
## Call:
## arima(x = sqrt.rosslare.ds, order = c(1, 0, 0))
##
## Coefficients:
##          ar1  intercept
##       0.4044     3.3251
## s.e.  0.0214     0.0253
##
## sigma^2 estimated as 0.4149:  log likelihood = -1788.91,  aic = 3583.82
```

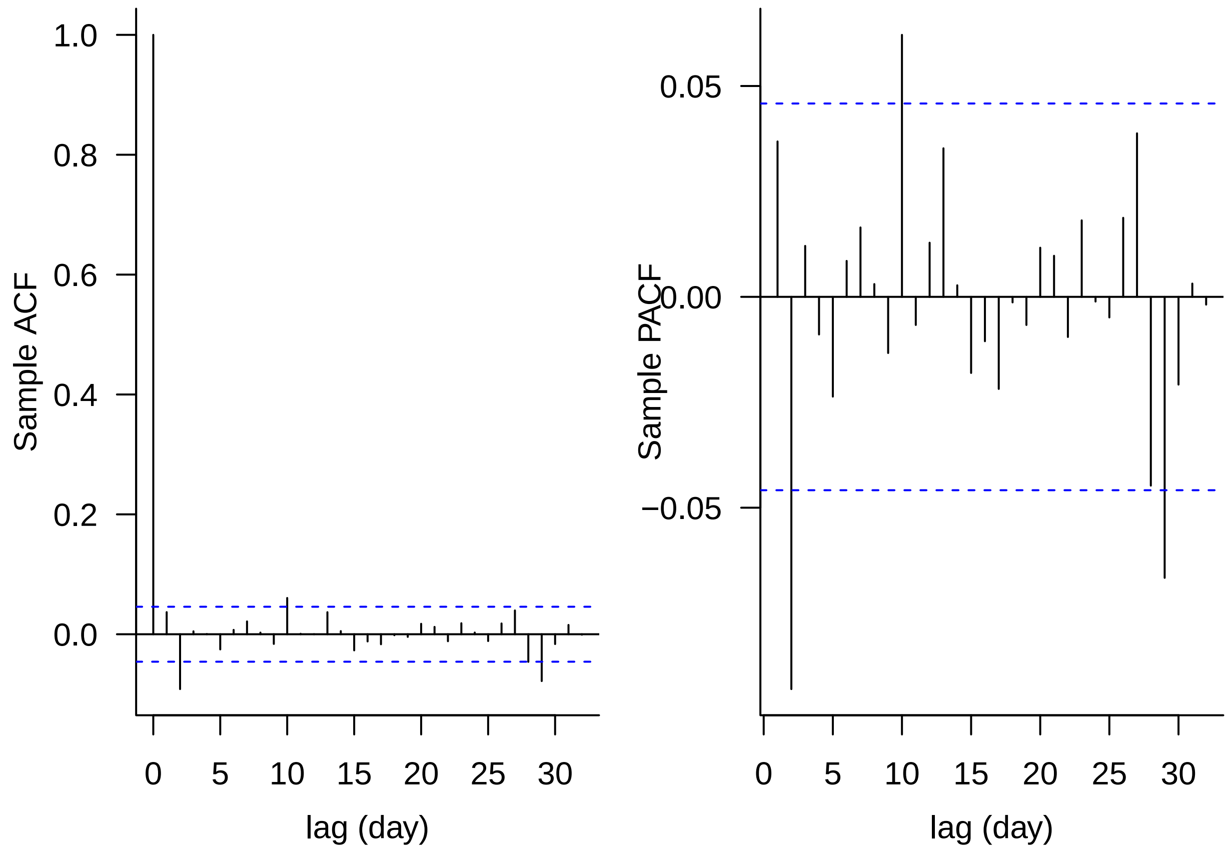
```r
par(bty = "L", mar = c(3.6, 3.6, 0.5, 0.6), las = 1, mgp = c(2.2, 1, 0))
acf(sqrt.rosslare.ds, main = "")
acf_true <- ARMAacf(ar = c(ar1.model$coef[1]), lag.max = 32)[-1]
points(1:32, acf_true, col = "red")
```




```
## extract the residuals
ar1.resids <- resid(ar1.model)

## time series plot of the residuals
par(bty = "L", mar = c(3.6, 3.6, 0.5, 0.6), las = 1, mgp = c(2.2, 1, 0),
    mfrow = c(1, 2))
plot(year, ar1.resids, type = "l", xlab = "year", ylab = "AR1 residuals",
     lwd = 0.6, col = "gray")
abline(h = 0, lty = 2)
## Normal Q-Q plot for the residuals
qqnorm(ar1.resids, main = "", cex = 0.75); qqline(ar1.resids)
```




```
## Sample ACF and PACF of the residuals
par(bty = "L", mar = c(3.6, 3.6, 0.5, 0.6), las = 1, mgp = c(2.4, 1, 0),
    mfrow = c(1, 2))
acf(ar1.resids, ylab = "Sample ACF", xlab = "lag (day)", main = "")
pacf(ar1.resids, ylab = "Sample PACF", xlab = "lag (day)")
```




```r
## Carry out the Box-Pierce test
Box.test(ar1.resids, lag = 32, type = "Ljung-Box")
```

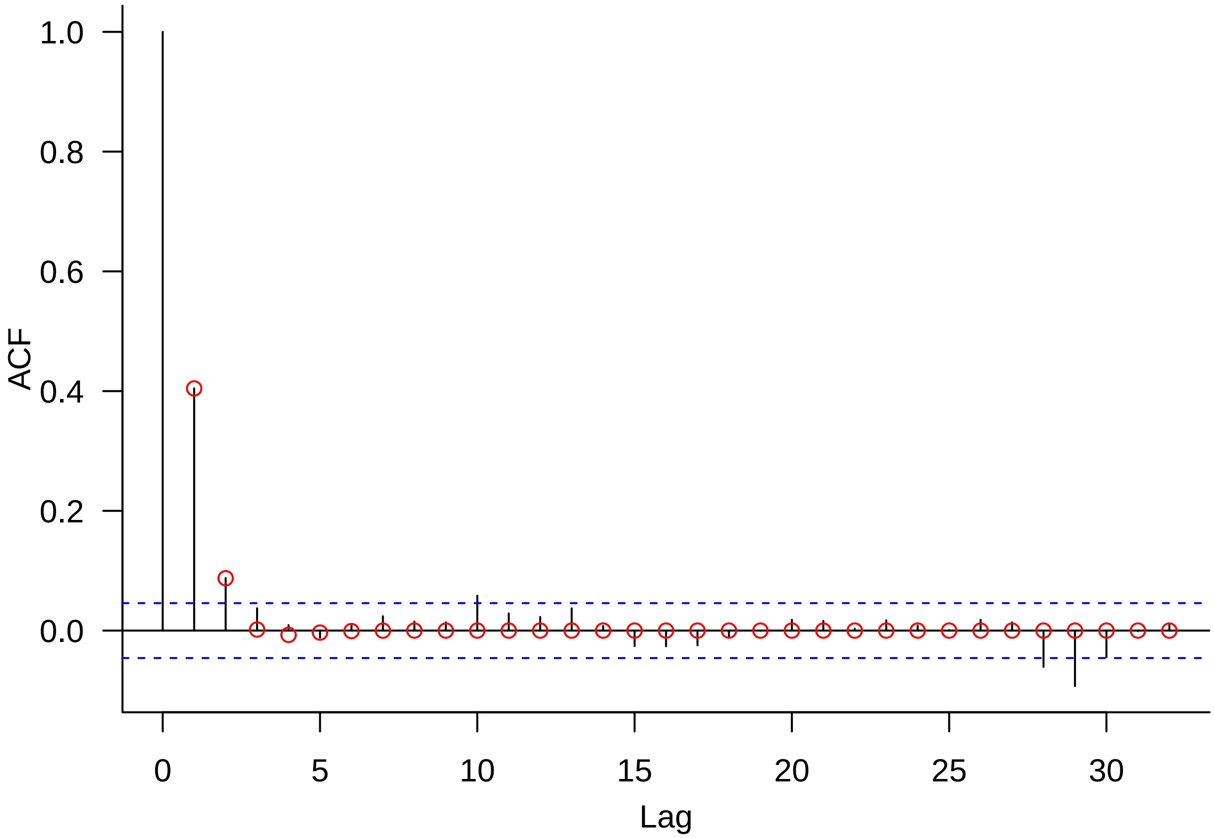
```
##
##  Box-Ljung test
##
## data:  ar1.resids
## X-squared = 53.656, df = 32, p-value = 0.009603
```

Fit an AR(2) model

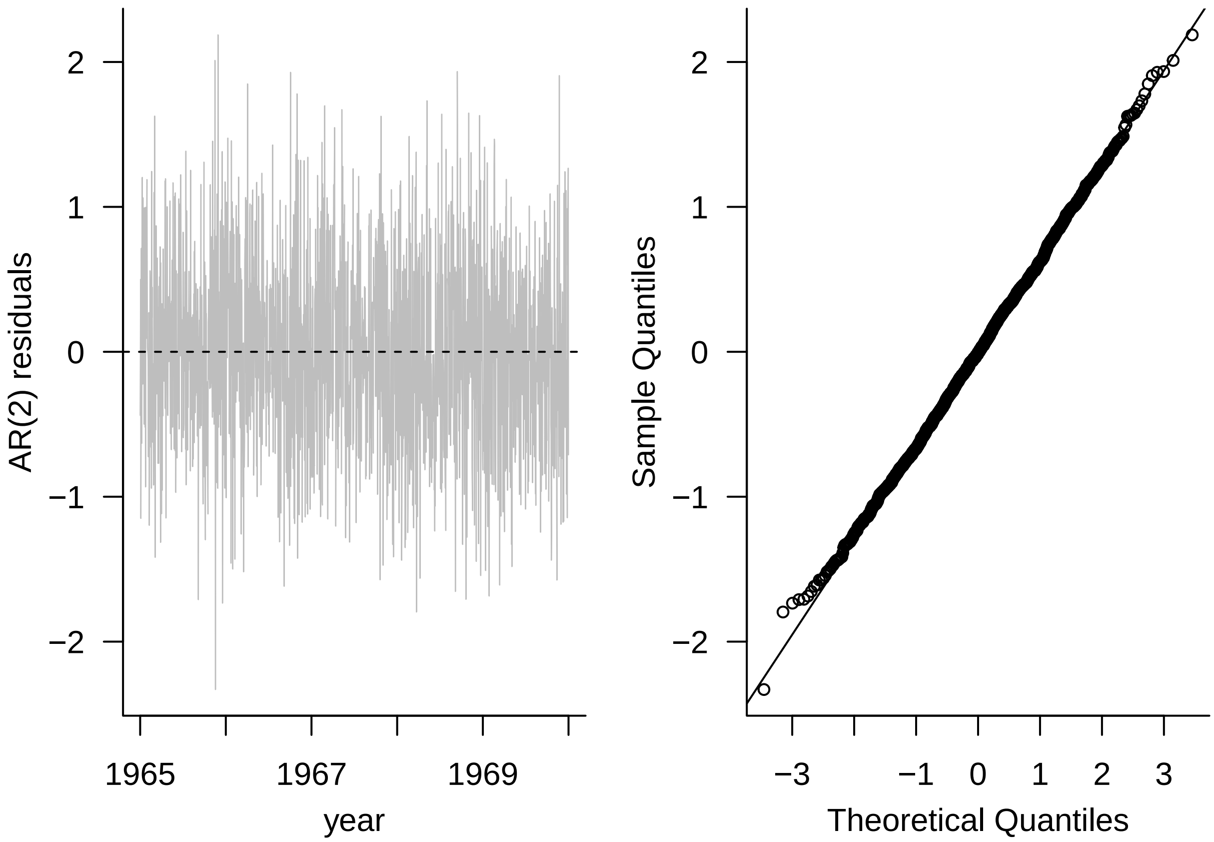
```r
## Fit an AR(2) model
ar2.model <- arima(sqrt.rosslare.ds, order = c(2, 0, 0))
## summarize the model
ar2.model
```

```
##
## Call:
## arima(x = sqrt.rosslare.ds, order = c(2, 0, 0))
##
## Coefficients:
##          ar1      ar2  intercept
##       0.4413  -0.0911     3.3252
## s.e.  0.0233   0.0233     0.0231
##
## sigma^2 estimated as 0.4115:  log likelihood = -1781.32,  aic = 3570.65
```

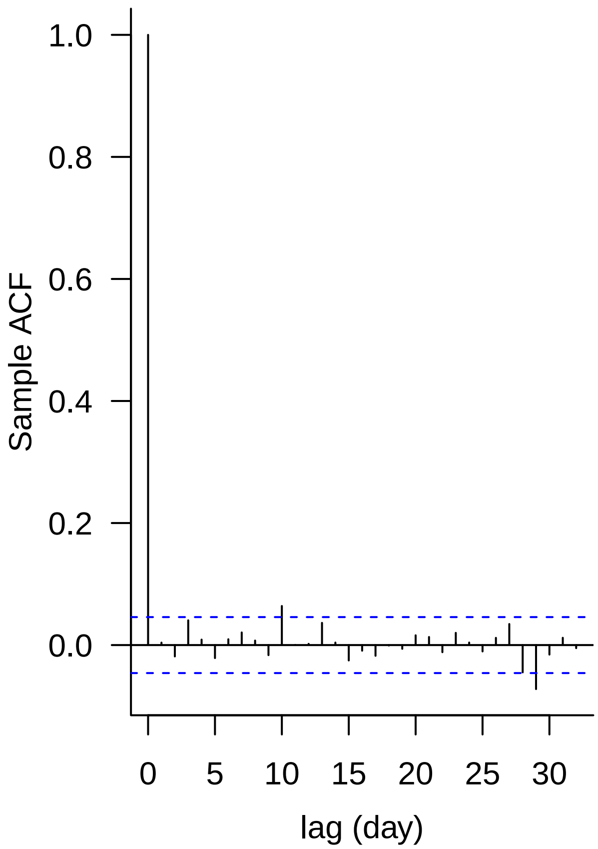
```r
par(bty = "L", mar = c(3.6, 3.6, 0.5, 0.6), las = 1, mgp = c(2.2, 1, 0))
acf(sqrt.rosslare.ds, main = "")
acf_true <- ARMAacf(ar = c(ar2.model$coef[1:2]), lag.max = 32)[-1]
points(1:32, acf_true, col = "red")
```

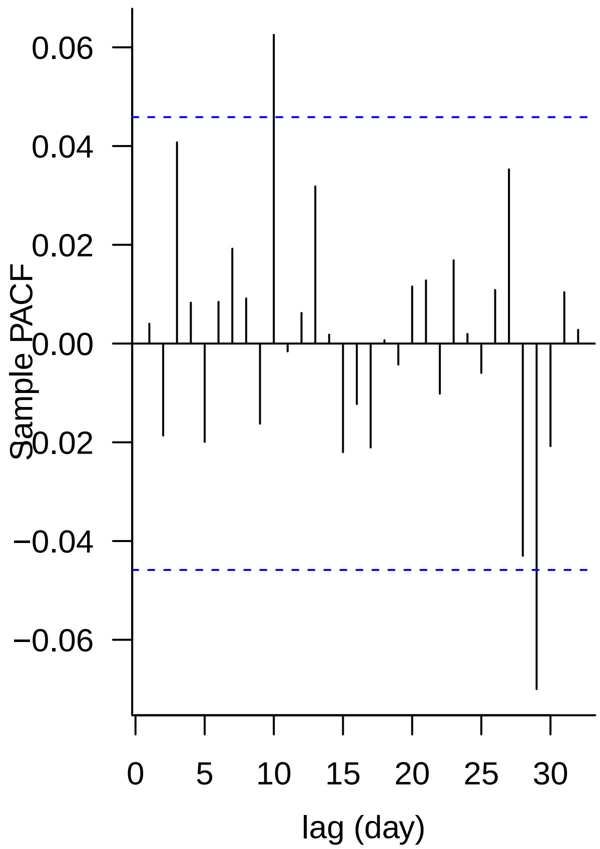



```r
## extract the residuals
ar2.resids <- resid(ar2.model)

## time series plot of the residuals
par(bty = "L", mar = c(3.6, 3.6, 0.5, 0.6), las = 1, mgp = c(2.2, 1, 0), mfrow = c(1, 2))
plot(year, ar2.resids, type = "l", xlab = "year",
     ylab = "AR(2) residuals", lwd = 0.6, col = "gray")
abline(h = 0, lty = 2)
## Normal Q-Q plot for the residuals
qqnorm(ar2.resids, main = "", cex = 0.75); qqline(ar2.resids)
```




```r
## Sample ACF and PACF of the residuals
par(bty = "L", mar = c(3.6, 3.6, 0.5, 0.6), las = 1, mgp = c(2.4, 1, 0),
    mfrow = c(1, 2))
acf(ar2.resids, ylab = "Sample ACF", xlab = "lag (day)", main = "")
pacf(ar2.resids, ylab = "Sample PACF", xlab = "lag (day)")
```






```r
## Carry out the Box-Pierce test
Box.test(ar2.resids, lag = 32, type = "Ljung-Box")
```

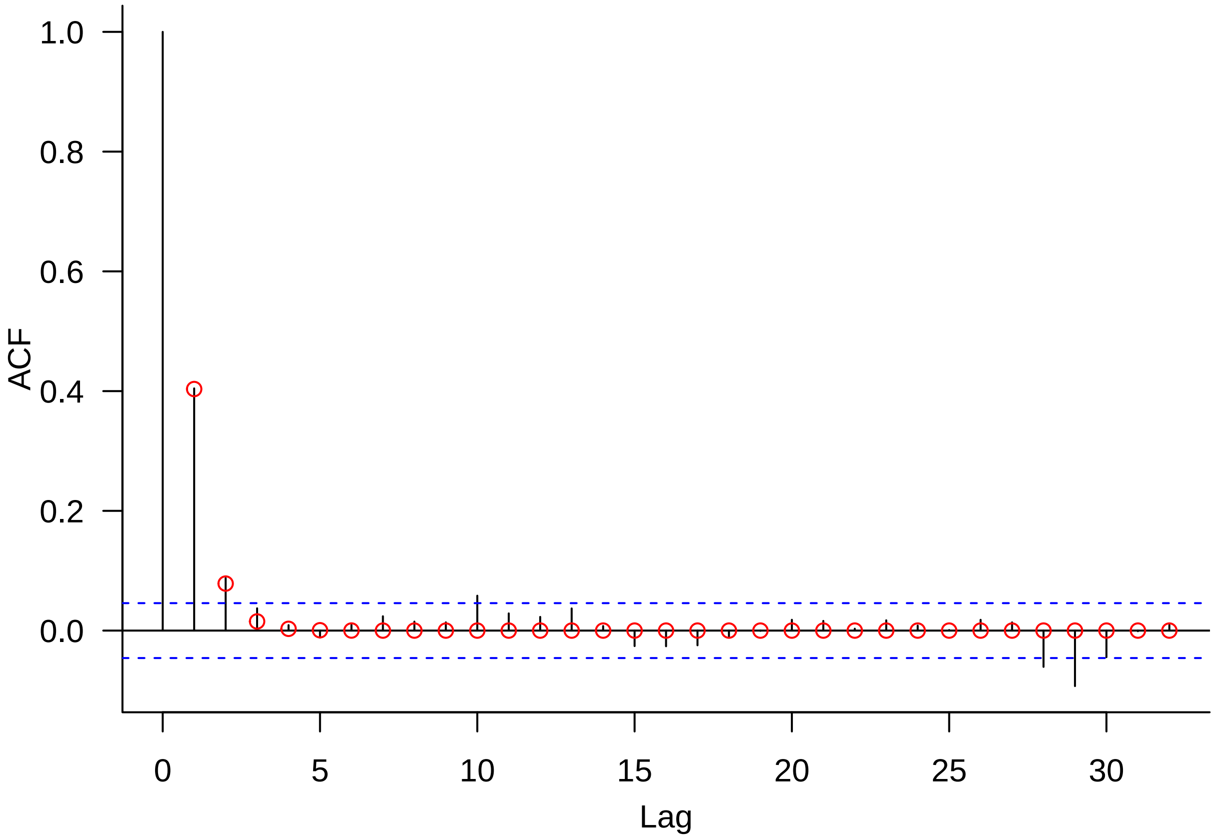
```
##
##  Box-Ljung test
##
## data:  ar2.resids
## X-squared = 36.852, df = 32, p-value = 0.2544
```

Fit an ARMA(1,1) model

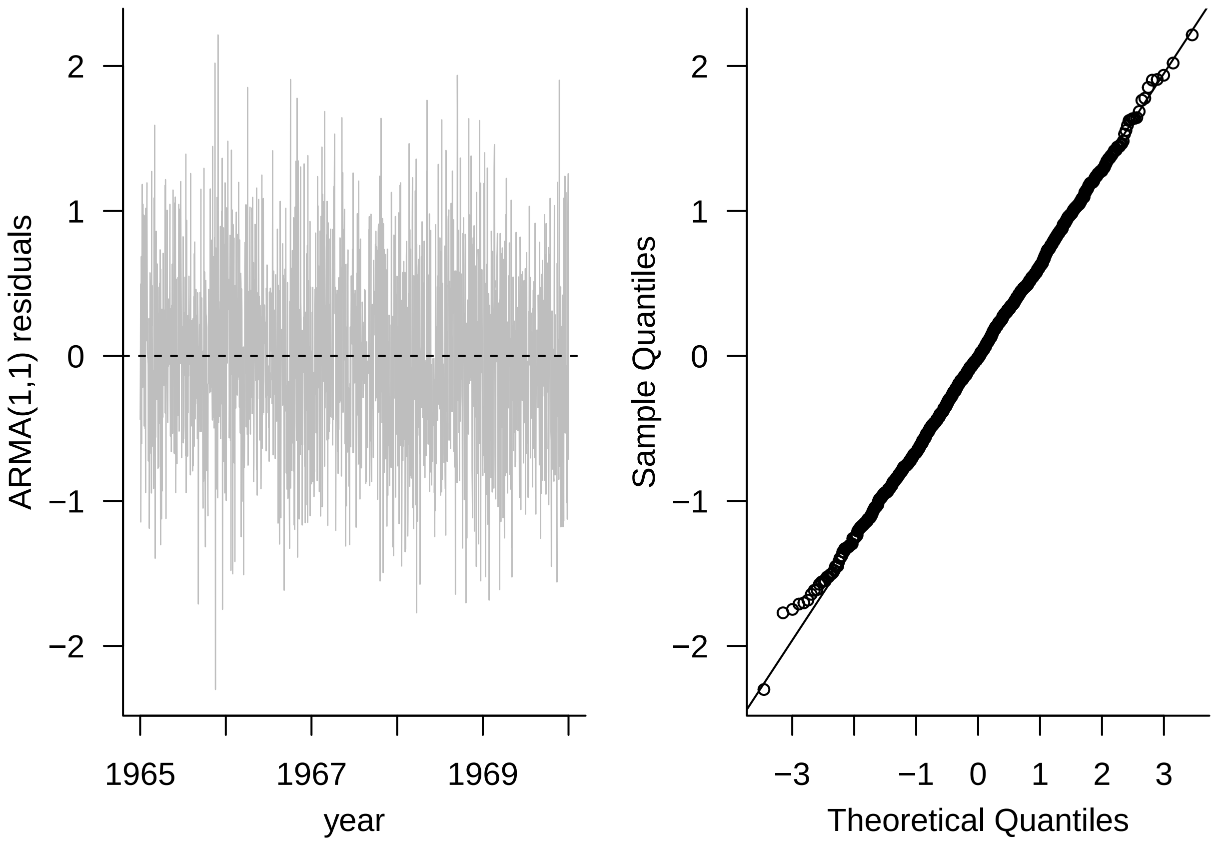
```r
## Fit an ARMA(1,1) model
arma11.model <- arima(sqrt.rosslare.ds, order = c(1, 0, 1))
## summarize the model
arma11.model
```

```
##
## Call:
## arima(x = sqrt.rosslare.ds, order = c(1, 0, 1))
##
## Coefficients:
##          ar1     ma1  intercept
##       0.1947  0.2521     3.3250
## s.e.  0.0556  0.0553     0.0233
##
## sigma^2 estimated as 0.4108:  log likelihood = -1779.92,  aic = 3567.83
```

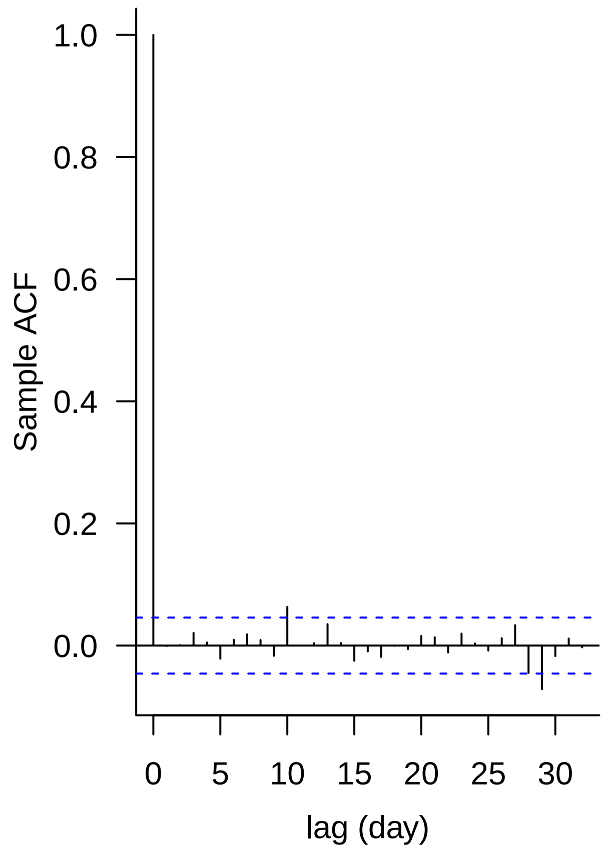
```r
par(bty = "L", mar = c(3.6, 3.6, 0.5, 0.6), las = 1, mgp = c(2.2, 1, 0))
acf(sqrt.rosslare.ds, main = "")
acf_true <- ARMAacf(ar = arma11.model$coef[1], ma = arma11.model$coef[2], lag.max = 32)[-1]
points(1:32, acf_true, col = "red")
```

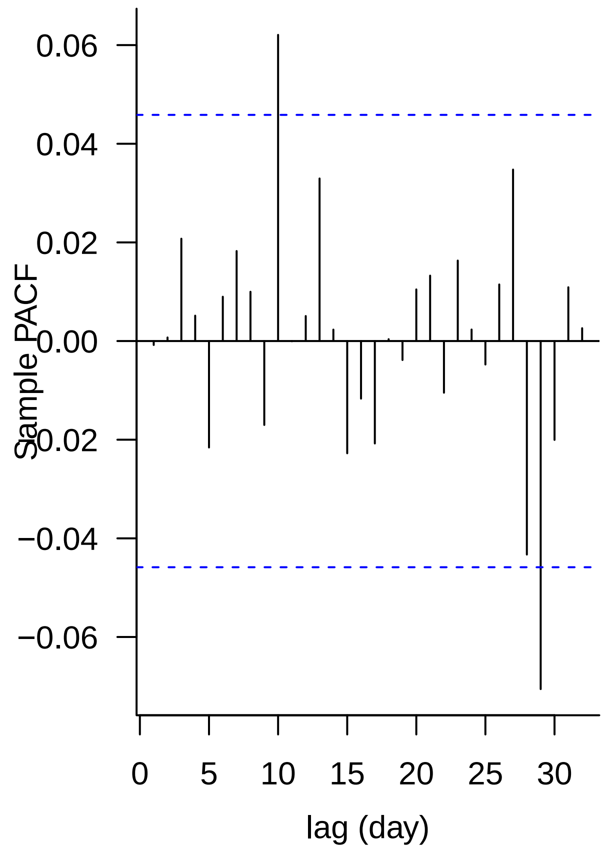



```r
## extract the residuals
arma11.resids <- resid(arma11.model)

## time series plot of the residuals
par(bty = "L", mar = c(3.6, 3.6, 0.5, 0.6), las = 1, mgp = c(2.2, 1, 0),
    mfrow = c(1, 2))
plot(year, arma11.resids, type = "l", xlab = "year",
     ylab = "ARMA(1,1) residuals", lwd = 0.6, col = "gray")
abline(h = 0, lty = 2)
## Normal Q-Q plot for the residuals
qqnorm(arma11.resids, main = "", cex = 0.75); qqline(arma11.resids)
```




```r
## Sample ACF and PACF of the residuals
par(bty = "L", mar = c(3.6, 3.6, 0.5, 0.6), las = 1, mgp = c(2.4, 1, 0),
    mfrow = c(1, 2))
acf(arma11.resids, ylab = "Sample ACF", xlab = "lag (day)", main = "")
pacf(arma11.resids, ylab = "Sample PACF", xlab = "lag (day)")
```






```r
## Carry out the Box-Pierce test
Box.test(arma11.resids, lag = 32, type = "Ljung-Box")
```

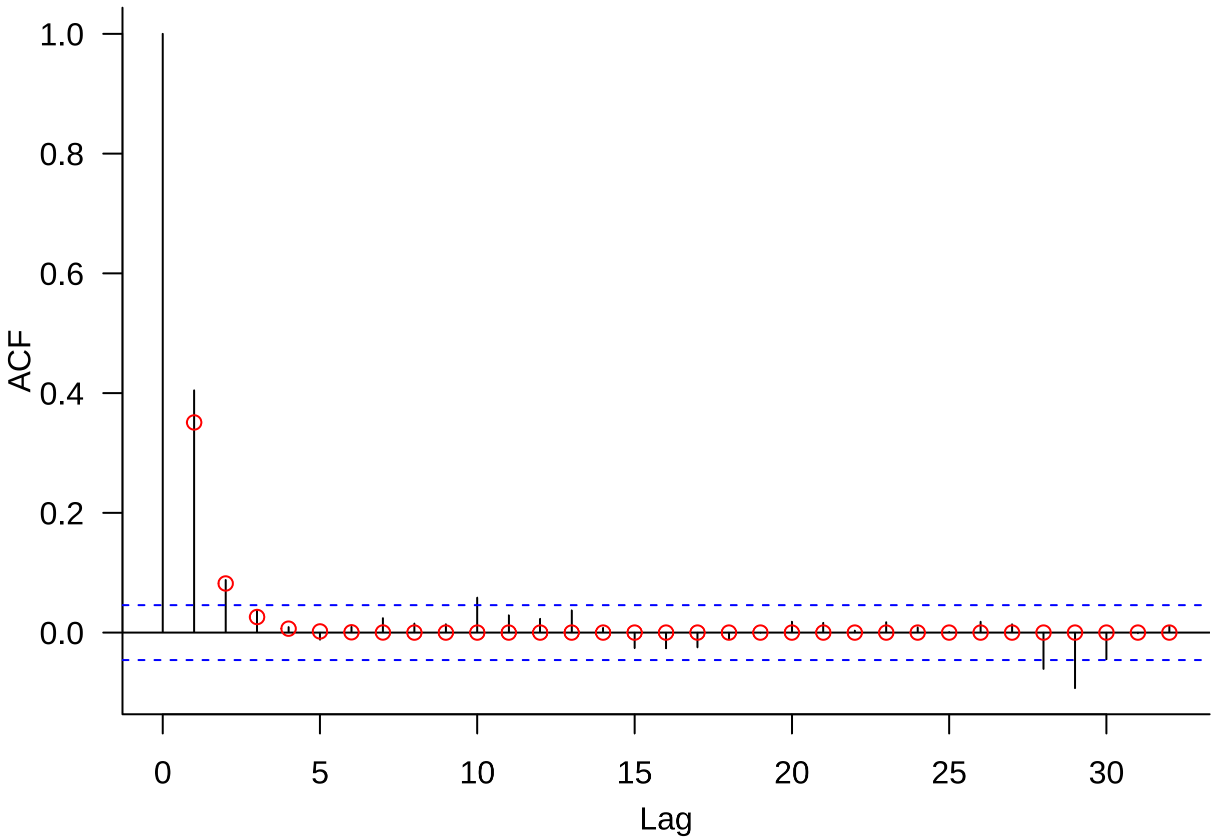
```
##
##  Box-Ljung test
##
## data:  arma11.resids
## X-squared = 33.09, df = 32, p-value = 0.4137
```

Fit an ARMA(2,1) model

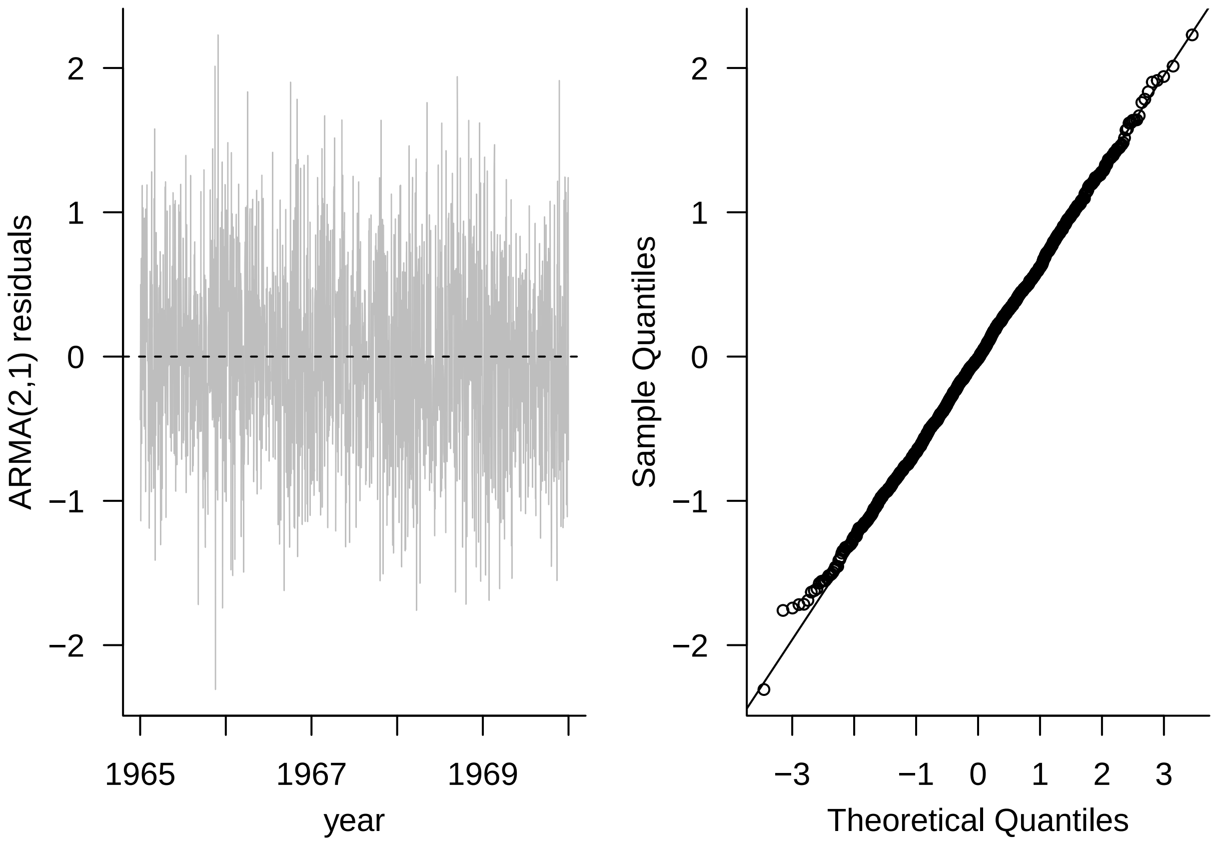
```r
## Fit an ARMA(2,1) model
arma21.model <- arima(sqrt.rosslare.ds, order = c(2, 0, 1))
## summarize the model
arma21.model
```

```
##
## Call:
## arima(x = sqrt.rosslare.ds, order = c(2, 0, 1))
##
## Coefficients:
##          ar1     ar2     ma1  intercept
##       0.0674  0.0584  0.3785     3.3247
## s.e.  0.1693  0.0772  0.1665     0.0236
##
## sigma^2 estimated as 0.4107:  log likelihood = -1779.66,  aic = 3569.32
```

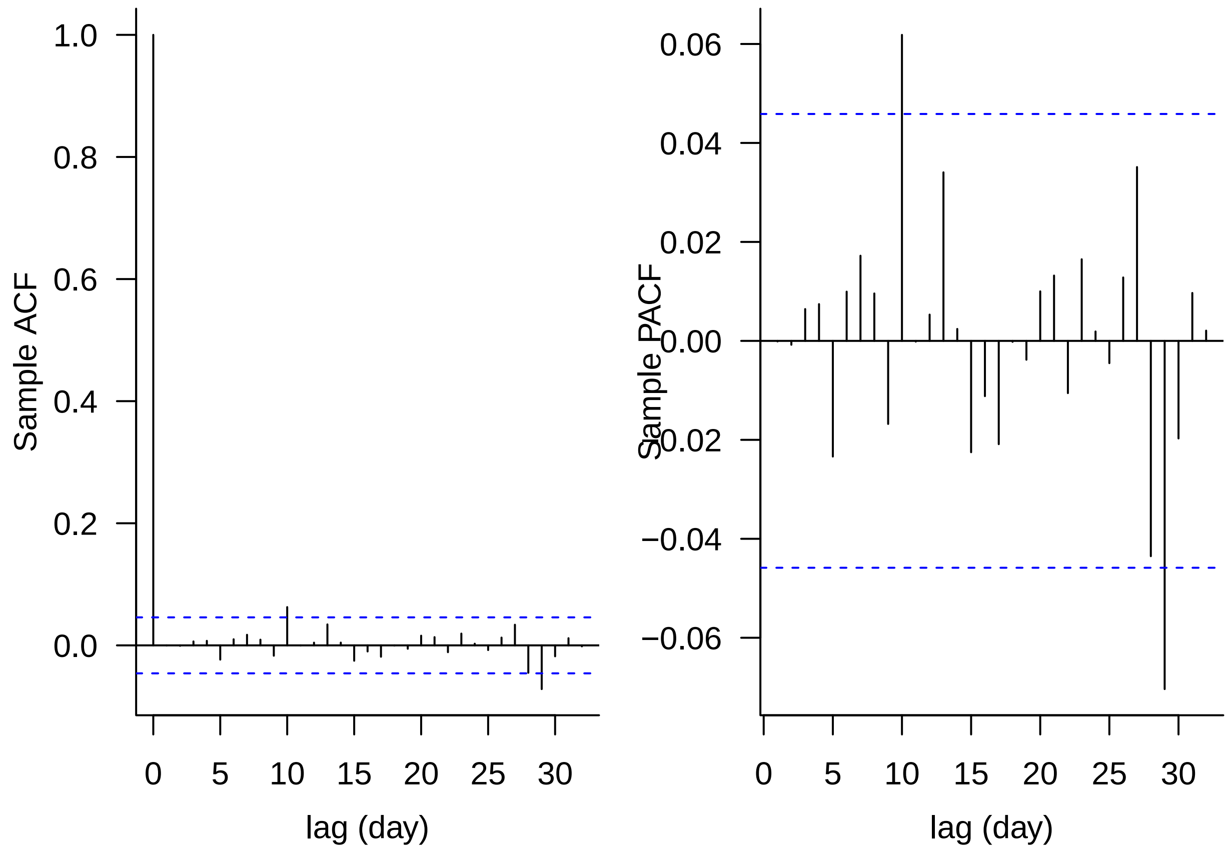
```r
par(bty = "L", mar = c(3.6, 3.6, 0.5, 0.6), las = 1, mgp = c(2.2, 1, 0))
acf(sqrt.rosslare.ds, main = "")
acf_true <- ARMAacf(ar = arma21.model$coef[1:2], ma = arma11.model$coef[3], lag.max = 32)[-1]
points(1:32, acf_true, col = "red")
```




```r
## extract the residuals
arma21.resids <- resid(arma21.model)

## time series plot of the residuals
par(bty = "L", mar = c(3.6, 3.6, 0.5, 0.6), las = 1, mgp = c(2.2, 1, 0),
    mfrow = c(1, 2))
plot(year, arma21.resids, type = "l", xlab = "year",
     ylab = "ARMA(2,1) residuals", lwd = 0.6, col = "gray")
abline(h = 0, lty = 2)
## Normal Q-Q plot for the residuals
qqnorm(arma21.resids, main = "", cex = 0.75); qqline(arma21.resids)
```




```r
## Sample ACF and PACF of the residuals
par(bty = "L", mar = c(3.6, 3.6, 0.5, 0.6), las = 1, mgp = c(2.4, 1, 0),
    mfrow = c(1, 2))
acf(arma21.resids, ylab = "Sample ACF", xlab = "lag (day)", main = "")
pacf(arma21.resids, ylab = "Sample PACF", xlab = "lag (day)")
```




```r
## Carry out the Box-Pierce test
Box.test(arma21.resids, lag = 32, type = "Ljung-Box")
```

```
##
##  Box-Ljung test
##
## data:  arma21.resids
## X-squared = 32.537, df = 32, p-value = 0.4404
```

Use AIC to conduct model selection

```r
AIC.to.AICC <- function (aic, n, npars) {
  aic - 2 * npars * ( 1 - n/(n-1-npars))
}
# calculate the length of the time series
n <- length(sqrt.rosslare.ds)

# Here are the AIC values
ar1.model$aic
```

```
## [1] 3583.817
```

```r
ar2.model$aic
```

```
## [1] 3570.65
```

```r
arma11.model$aic
```

```
## [1] 3567.833
```

```r
arma21.model$aic
```

```
## [1] 3569.319
```

```r
# convert the AIC values to AICC values.
AIC.to.AICC(ar1.model$aic, n, 2)
```

```
## [1] 3583.824
```

```r
AIC.to.AICC(ar2.model$aic, n, 3)
```

```
## [1] 3570.663
```

```r
AIC.to.AICC(arma11.model$aic, n, 3)
```

```
## [1] 3567.847
```

```r
AIC.to.AICC(arma21.model$aic, n, 4)
```

```
## [1] 3569.341
```

Forecasting

```r
## How many days will we predict into the future?
h <- 31
 ## Predict 'h' days into the future using the ARMA(1,1) model.
sqrt.rosslare.forecast <- predict(arma11.model, h)
sqrt.rosslare.forecast$pred
```

```
## Time Series:
## Start = 1828
## End = 1858
## Frequency = 1
##  [1] 3.136357 3.288312 3.317896 3.323656 3.324778 3.324996 3.325039 3.325047
##  [9] 3.325049 3.325049 3.325049 3.325049 3.325049 3.325049 3.325049 3.325049
## [17] 3.325049 3.325049 3.325049 3.325049 3.325049 3.325049 3.325049 3.325049
## [25] 3.325049 3.325049 3.325049 3.325049 3.325049 3.325049 3.325049
```

```r
sqrt.rosslare.forecast$se
```

```
## Time Series:
## Start = 1828
## End = 1858
## Frequency = 1
##  [1] 0.6409755 0.7020359 0.7042464 0.7043300 0.7043332 0.7043333 0.7043333
##  [8] 0.7043333 0.7043333 0.7043333 0.7043333 0.7043333 0.7043333 0.7043333
## [15] 0.7043333 0.7043333 0.7043333 0.7043333 0.7043333 0.7043333 0.7043333
## [22] 0.7043333 0.7043333 0.7043333 0.7043333 0.7043333 0.7043333 0.7043333
## [29] 0.7043333 0.7043333 0.7043333
```

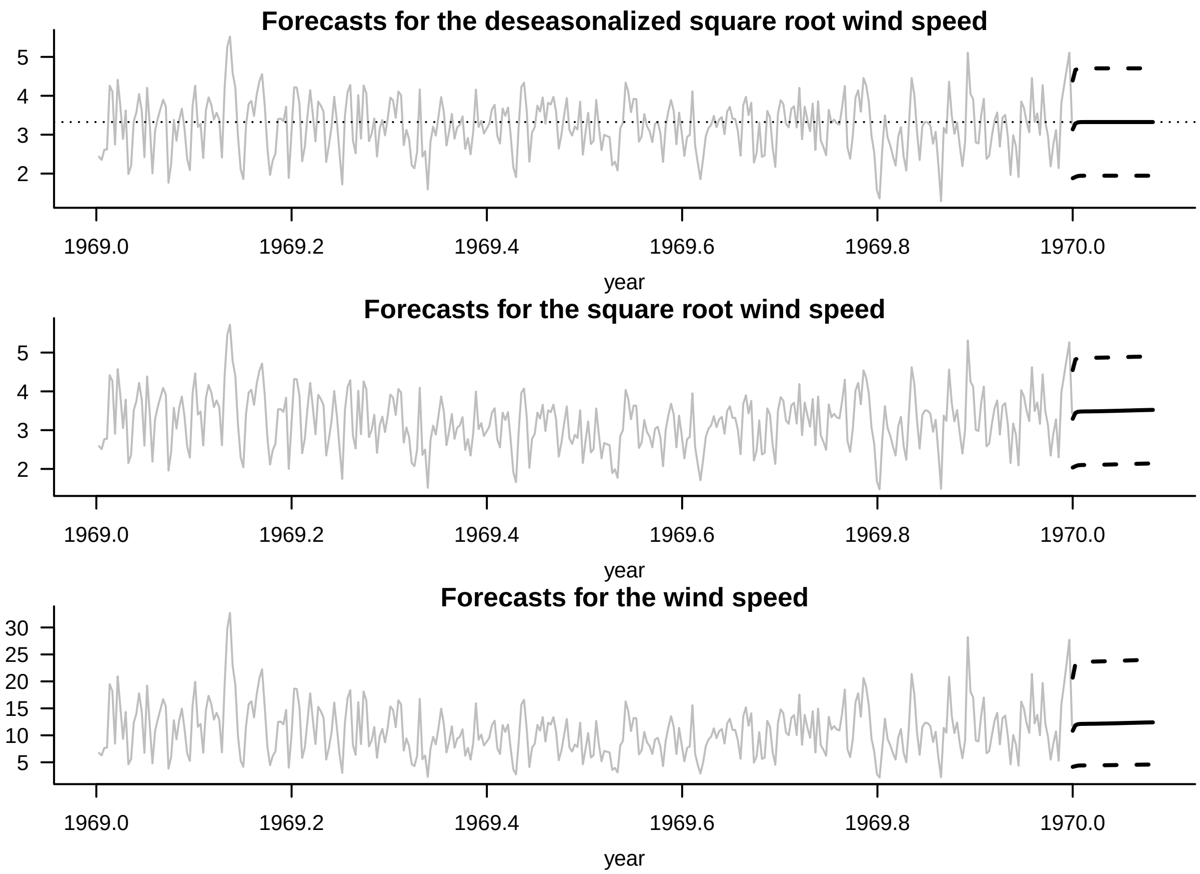
```r
## define the forecast variable
forecast <- sqrt.rosslare.forecast$pred
## The plus or minus value is the z critical value
## times the standard error for the forecast
plus.or.minus <- qnorm(0.975) * sqrt.rosslare.forecast$se
lower <- forecast - plus.or.minus
```

```r
upper <- forecast + plus.or.minus
## Define the prediction time
fyear <- 1970 + (0:(h - 1)) / 365.25
```

Visualizing the Forecasts

```r
par(bty = "L", mar = c(3.6, 3.6, 0.75, 0.6), las = 1, mgp = c(2.4, 1, 0),
    mfrow = c(3, 1))
## Show the data for 1969 onwards
plot(year[year>1969], sqrt.rosslare.ds[year>1969], type="l",
     xlim=c(1969, max(fyear)), col="grey", xlab="year",
     ylab="")

## Add the BLUP, along with the prediction limits
lines(fyear, forecast, lwd=2)
lines(fyear, lower, lty=2, lwd=2)
lines(fyear, upper, lty=2, lwd=2)

## add a horizontal line at the mean
abline(h=mean(sqrt.rosslare.ds), lty=3)

title("Forecasts for the deseasonalized square root wind speed")

## now add the seasonality estimate for the first 31 days in a year.
adj.forecast <- fitted(harm.model)[1:h] + sqrt.rosslare.forecast$pred

## adjust the lower and upper values of the interval
lower <- adj.forecast - plus.or.minus
upper <- adj.forecast + plus.or.minus

## Show the data for 1969 onwards
plot(year[year>1969], sqrt.rosslare[year>1969], type="l",
     xlim=c(1969, max(fyear)), col="grey", xlab="year", ylab="")

title("Forecasts for the square root wind speed")

## Add the BLUP, along with the prediction limits
lines(fyear, adj.forecast, lwd=2)
lines(fyear, lower, lty=2, lwd=2)
lines(fyear, upper, lty=2, lwd=2)

## We square everything (forecast, lower limit, and upper limit)
## to get the forecast on the original wind speed (knots) scale.

## Show the data for 1969 onwards
plot(year[year>1969], rosslare[year>1969], type="l",
     xlim=c(1969, max(fyear)), col="grey", xlab="year", ylab="")

title("Forecasts for the wind speed")

## Add the BLUP, along with the prediction limits
lines(fyear, adj.forecast^2, lwd = 2)
```

```r
lines(fyear, lower^2, lty = 2, lwd = 2)
lines(fyear, upper^2, lty = 2, lwd = 2)
```



References