MATH 8090: ARMA Models: Prediction and Forecasting, Modeling Case Study

Whitney Huang, Clemson University

9/30-10/2/2025

Contents

NOAA wind data example

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

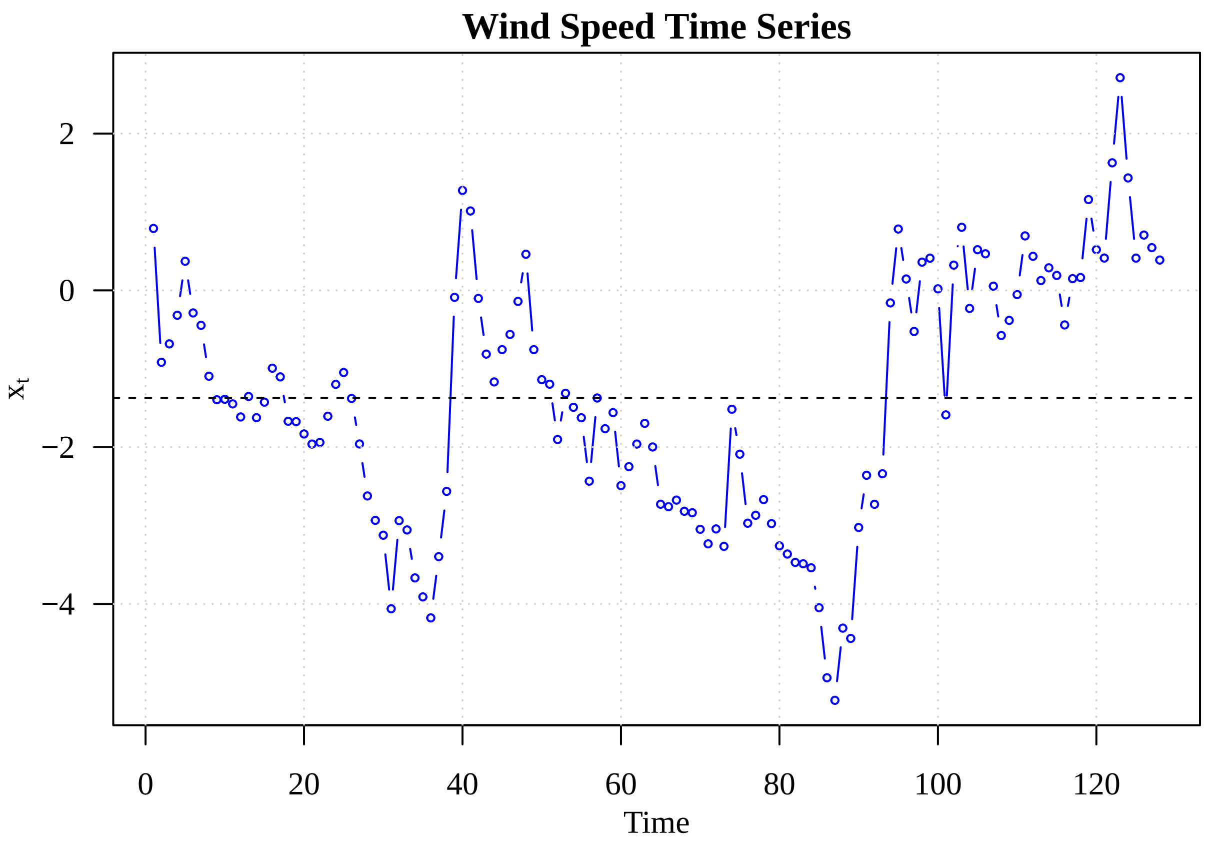
The one-step-ahead forecast of an AR(1) process is:

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

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

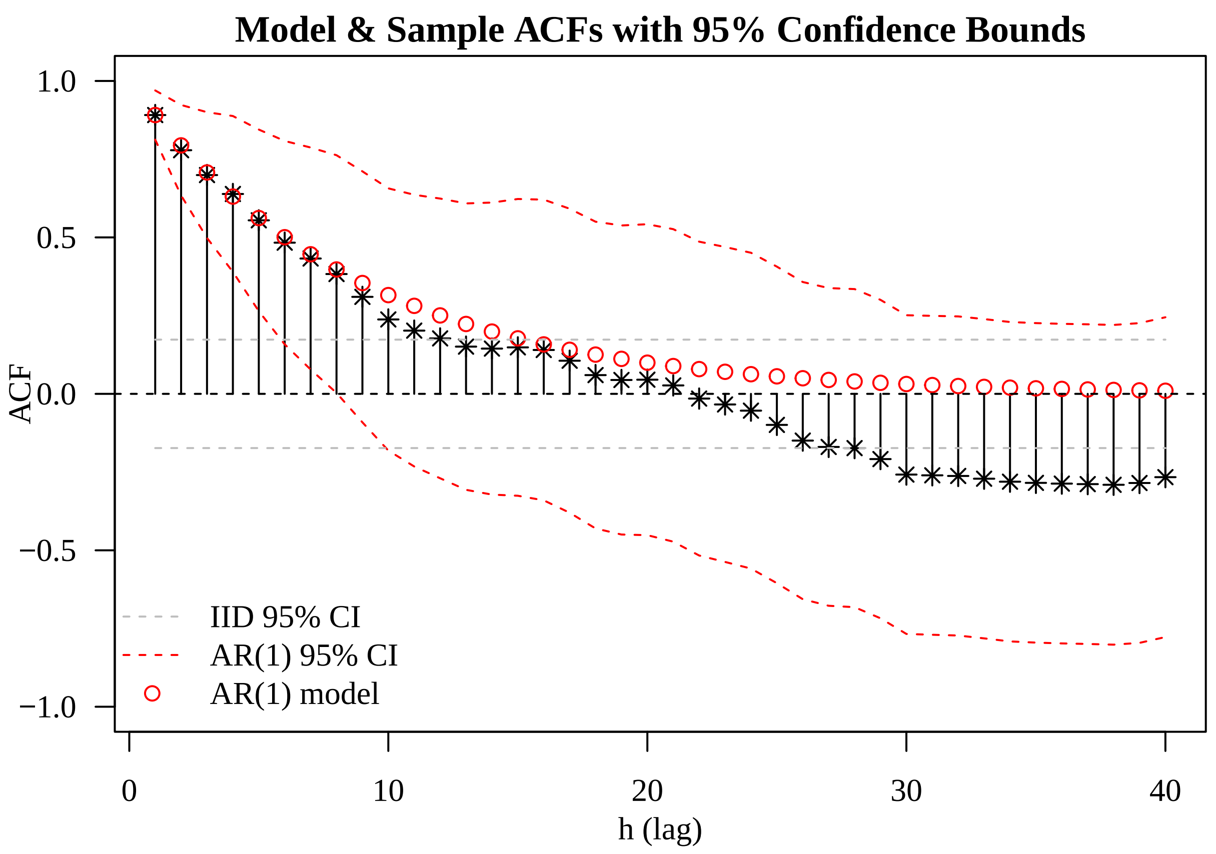
Load and plot the data

```r
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), family = "serif")
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

```r
# --- Quick AR(1) coefficient from lag-1 ACF --------------------------------
acf.ws <- acf(ws, lag.max = 40, plot = FALSE)$acf  # includes lag 0
phi.ws <- acf.ws[2]  # lag-1 sample ACF as a quick AR(1) phi estimate

# --- Bartlett variance factor for AR(1): w_hh(phi) ------------------------
# Var( rho_hat[h] )  w_hh(phi) / n
gen.whh.ar <- function(h, phi){
  p2  <- phi^2
  p2h <- p2^h
  -2 * h * p2h + (1 - p2h) * (1 + p2) / (1 - p2)
}

# --- Plot Sample ACF, AR(1) Model ACF, and 95% CIs (Bartlett & IID) -------
plot.ACFbartlettAR <- function(ts, n.lags = 40){
  stopifnot(is.numeric(ts), length(ts) > 5)
  n.ts <- length(ts)
  lags <- 1:n.lags

  # Sample ACF for lags 1..n.lags (remove lag 0)
  acf.full  <- acf(ts, lag.max = n.lags, plot = FALSE)$acf
  acf.est   <- acf.full[-1]

  # AR(1) model ACF using phi_hat = sample acf at lag 1
  phi.hat   <- acf.est[1]
  acf.model <- phi.hat^lags

  # Plot the sample ACF (stems + points)
  plot(lags, acf.est, type = "h",
       xlab = "h (lag)", ylab = "ACF",
       ylim = c(-1, 1),
       main = "Model & Sample ACFs with 95% Confidence Bounds",
       las = 1)
  points(lags, acf.est, pch = 8)  # star

  # Add AR(1) model ACF
  points(lags, acf.model, col = "red")

  # Bartlett 95% CI under AR(1)
  CI.AR <- 1.96 * sqrt(sapply(lags, function(h) gen.whh.ar(h, phi.hat))) / sqrt(n.ts)
  lines(lags, acf.est + CI.AR, col = "red", lty = 2)
  lines(lags, acf.est - CI.AR, col = "red", lty = 2)

  # IID 95% CI for comparison: ±1.96/sqrt(n)
  CI.IID <- rep(1.96 / sqrt(n.ts), n.lags)
  lines(lags, -CI.IID, col = "gray", lty = 2)
  lines(lags,  CI.IID, col = "gray", lty = 2)

  abline(h = 0, lty = "dashed")
  legend("bottomleft",
         legend = c("IID 95% CI", "AR(1) 95% CI", "AR(1) model"),
         lty    = c(2, 2, NA),
         pch    = c(NA, NA, 1),
         col    = c("gray", "red", "red"),
         bty     = "n")
}

# --- Plot styling---------------------------------------------
par(mgp = c(2, 1, 0), mar = c(3.5, 3.5, 1.4, 0.6), family = "serif")
plot.ACFbartlettAR(ws)
```



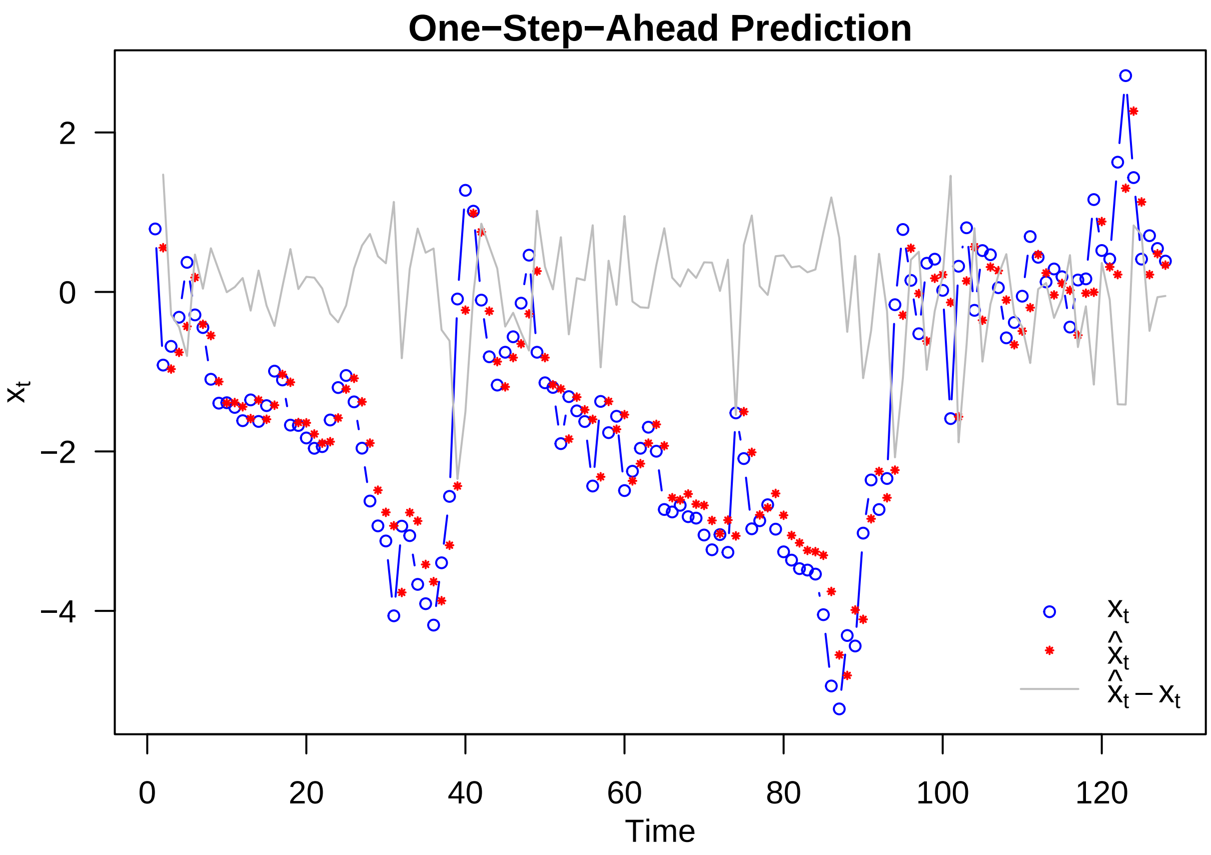
```r
# --- Alternatively: estimate phi via MLE ----------------------------------
phi_hat_mle <- arima(ws, order = c(1, 0, 0))  # AR(1) fit by MLE
phi_hat_mle
```

```
##
## 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
```

```r
# --- Center the series if you need a mean-zero version --------------------
ws.centered <- ws - mean(ws)
```

One-step-ahead forecast

```r
# -- One-step-ahead predictions and errors ----------------------------------
# Predictions available for t = 2..n using x_{t-1}
ws.hat <- phi.ws * ws.centered[1:(n - 1)] + xbar_ws        # \hat{x}_t
zt.ws  <- ws.hat - ws[2:n]                                  # e_t = \hat{x}_t - x_t
## 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")
```



```r
# -- Variance comparison (smaller error variance suggests better predictive fit)
cat("\nQuick AR(1) estimate (lag-1 ACF): phi_hat =", round(phi.ws, 4), "\n")
```

```
##
## Quick AR(1) estimate (lag-1 ACF): phi_hat = 0.8911
```

```r
cat("Var(prediction errors) =", round(var(zt.ws), 4), "\n")
```

```
## Var(prediction errors) = 0.4629
```

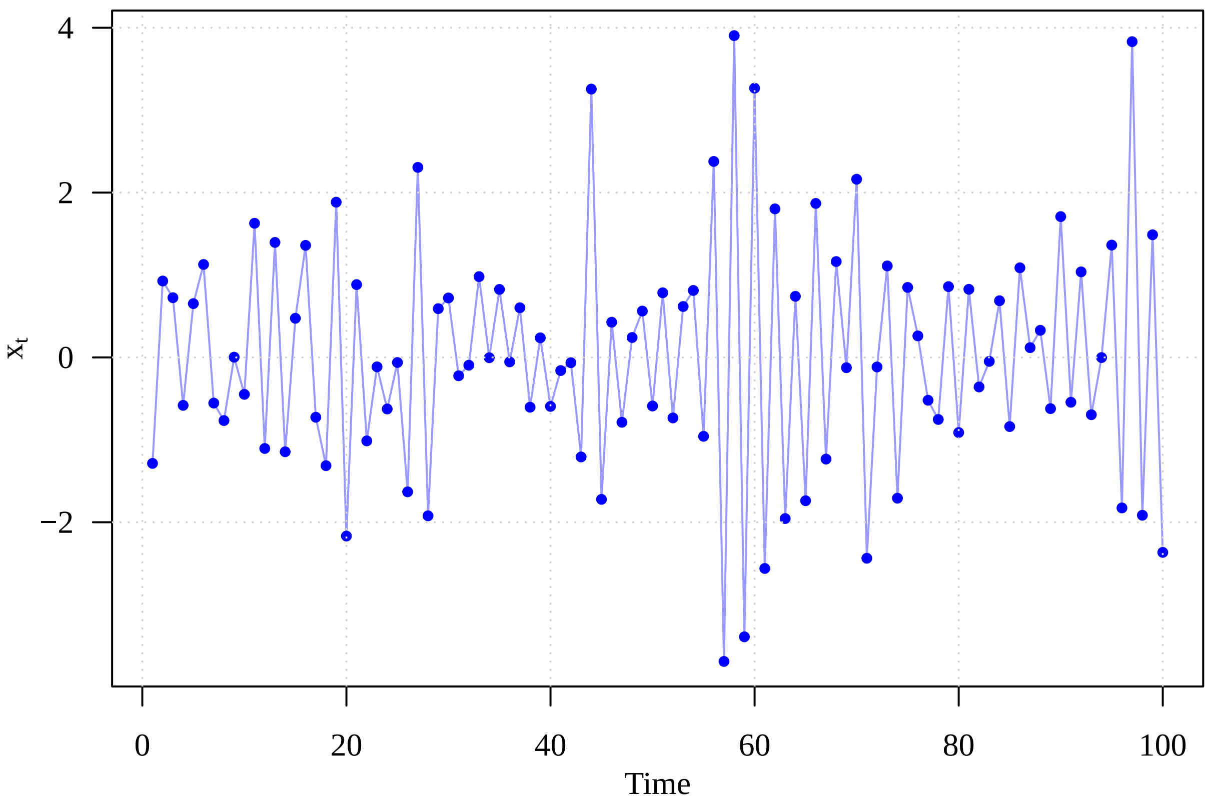
```r
cat("Var(series)            =", round(var(ws), 4), "\n")
```

```
## Var(series)            = 2.5025
```

Fill in missing value example

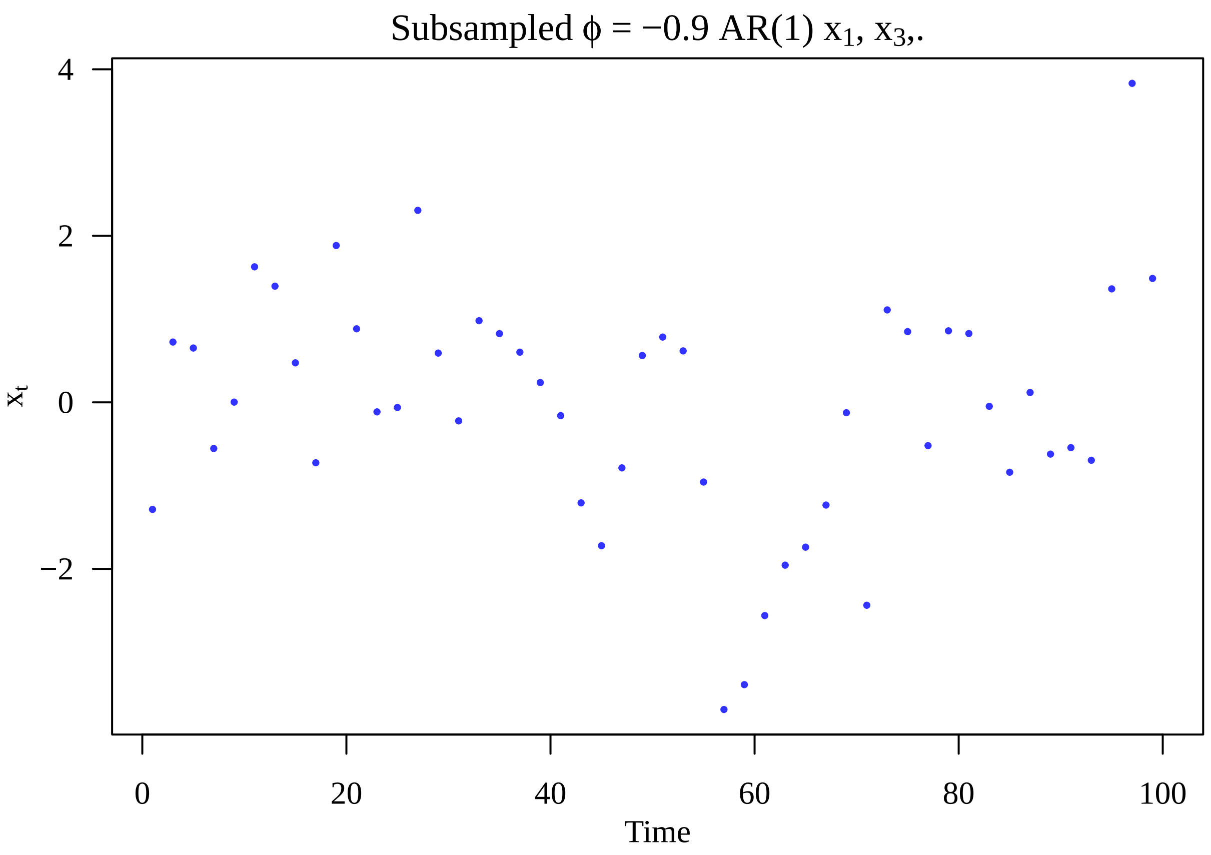
Simulate an AR(-0.9)

```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), family = "serif")
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

```r
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), family = "serif")
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)
```



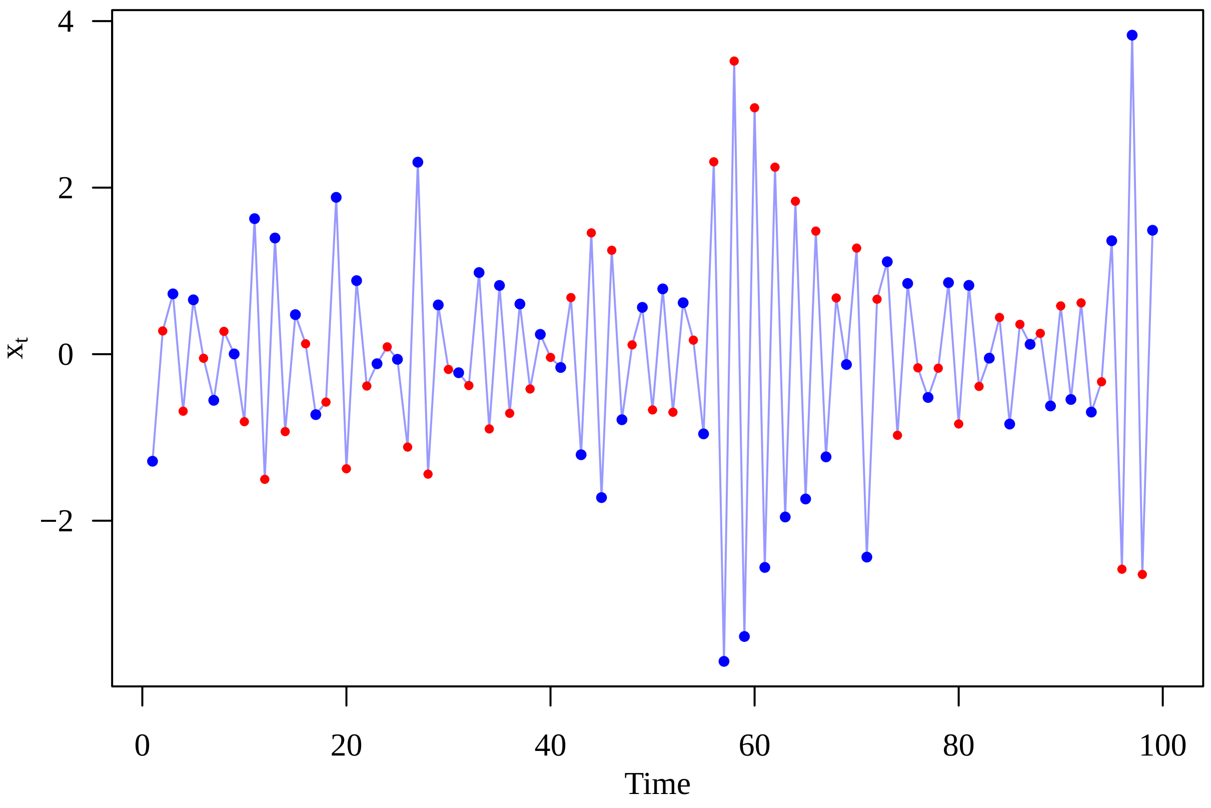
Fill in "missing" values

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

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

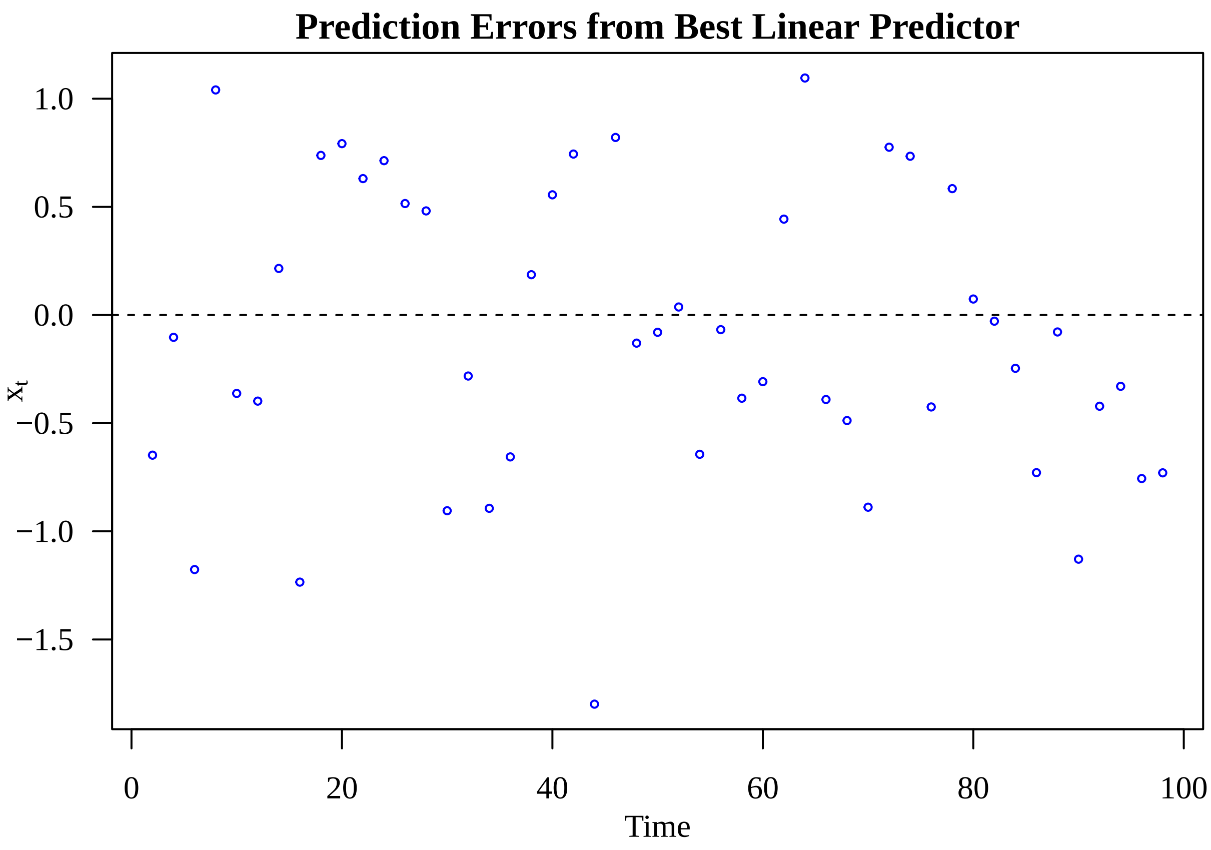
```r
ar1.ts.predicted <- ar1.ts
# Fill in even indices (2, 4, ..., 98) with prediction based on neighbors.
# Formula comes from conditional expectation of AR(1) with phi = -0.9
ar1.ts.predicted[seq(2, 98, 2)] <-
  -0.9 * (ar1.ts[seq(1, 97, 2)] + ar1.ts[seq(3, 99, 2)]) / 1.81

# Last element (t=100) set to NA since it has no neighbor on the right
ar1.ts.predicted[100] <- NA

# Plot styling
par(las = 1, mgp = c(2, 1, 0), mar = c(3.5, 3.5, 1.4, 0.6), family = "serif")

# Base plot: predicted series in light blue
plot(ar1.ts.predicted, col = alpha("blue", 0.4), xlab = "Time", type = "l",
     ylab = expression(x[t]), cex = 0.5)

# Mark predicted even-indexed points (red)
xs <- seq(2, 98, 2)
points(xs, ar1.ts.predicted[xs], pch = 19, col = "red", cex = 0.5)
```

```r
# Mark observed odd-indexed points (blue)
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), family = "serif")
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)
```

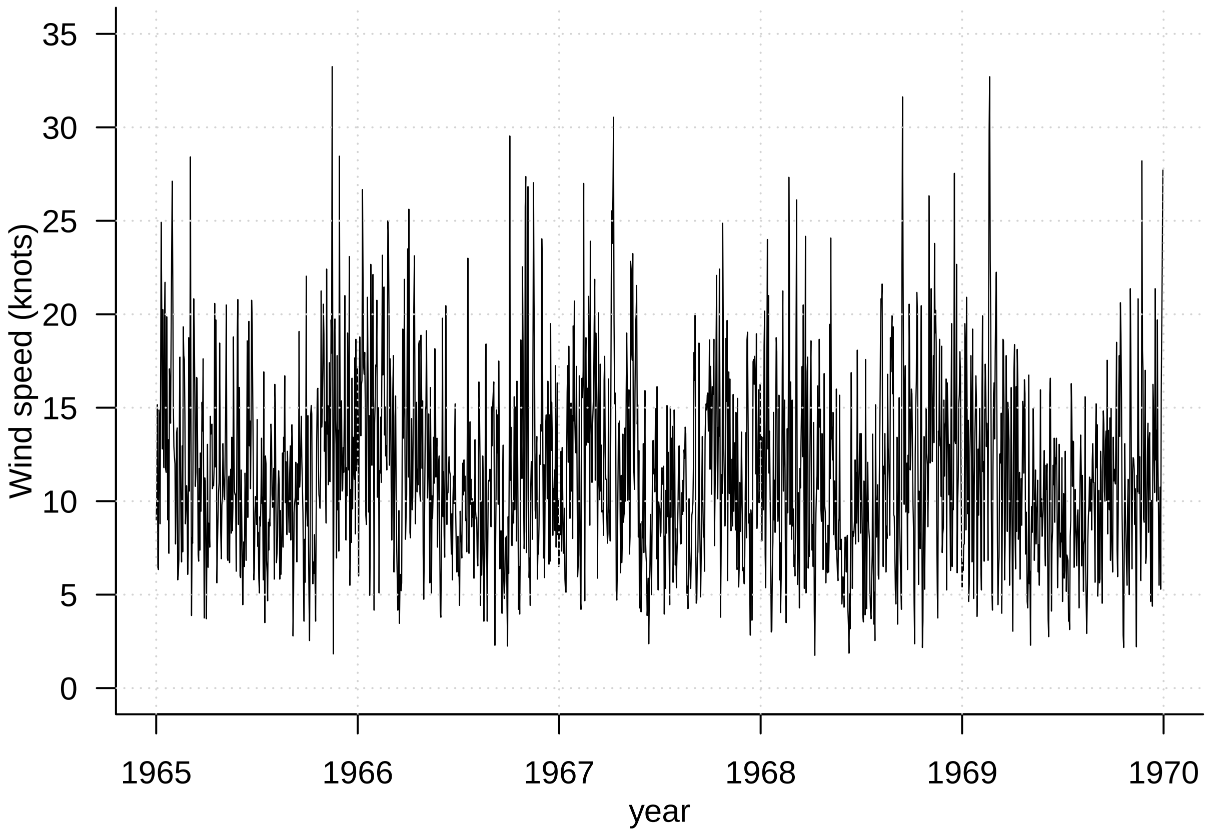


Ireland wind data case study

Load and plot the data

In this case study, we use the data at the Rosslare station from 1965 to 1969.

```r
library(gstat)
data(wind)
id <- which(wind$year %in% 65:69)
rosslare <- wind$ROS[id]
## 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()
```



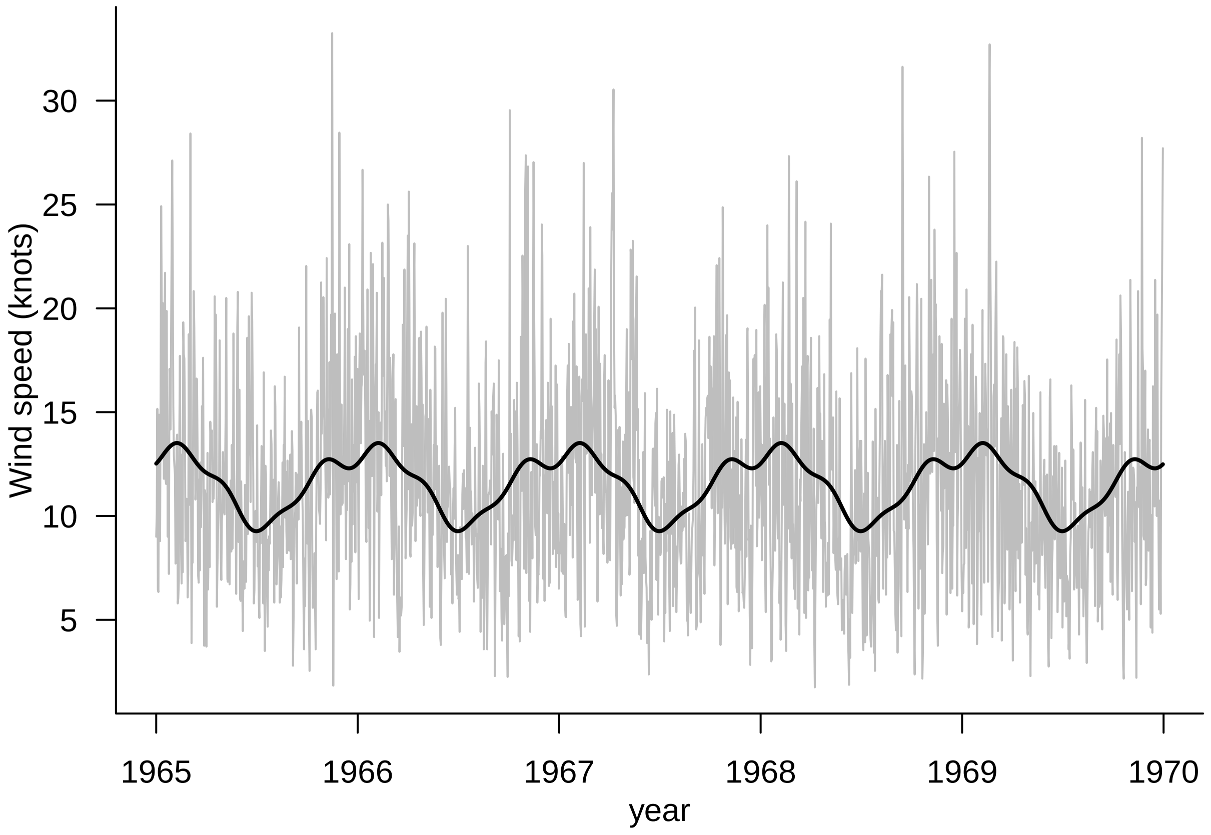
Deseasonalization: Harmonic Regression

We use harmonic regression with 4 harmonics per year to model the seasonal components.

```
## 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
## -10.8538  -3.3813  -0.4892   2.8395  20.8290
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 11.584141   0.112377 103.083  < 2e-16 ***
## harmonics1   1.687468   0.158936  10.617  < 2e-16 ***
## harmonics2  -0.435273   0.158936  -2.739  0.00623 **
## harmonics3  -0.060047   0.158936  -0.378  0.70562
## harmonics4  -0.251396   0.158936  -1.582  0.11388
## harmonics5   0.412363   0.158915   2.595  0.00954 **
## harmonics6   0.003874   0.158915   0.024  0.98055
## harmonics7   0.107245   0.158915   0.675  0.49985
## harmonics8   0.217870   0.158915   1.371  0.17055
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 4.802 on 1817 degrees of freedom
## Multiple R-squared:  0.06771,    Adjusted R-squared:  0.06361
## F-statistic:  16.5 on 8 and 1817 DF,  p-value: < 2.2e-16
```

```
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

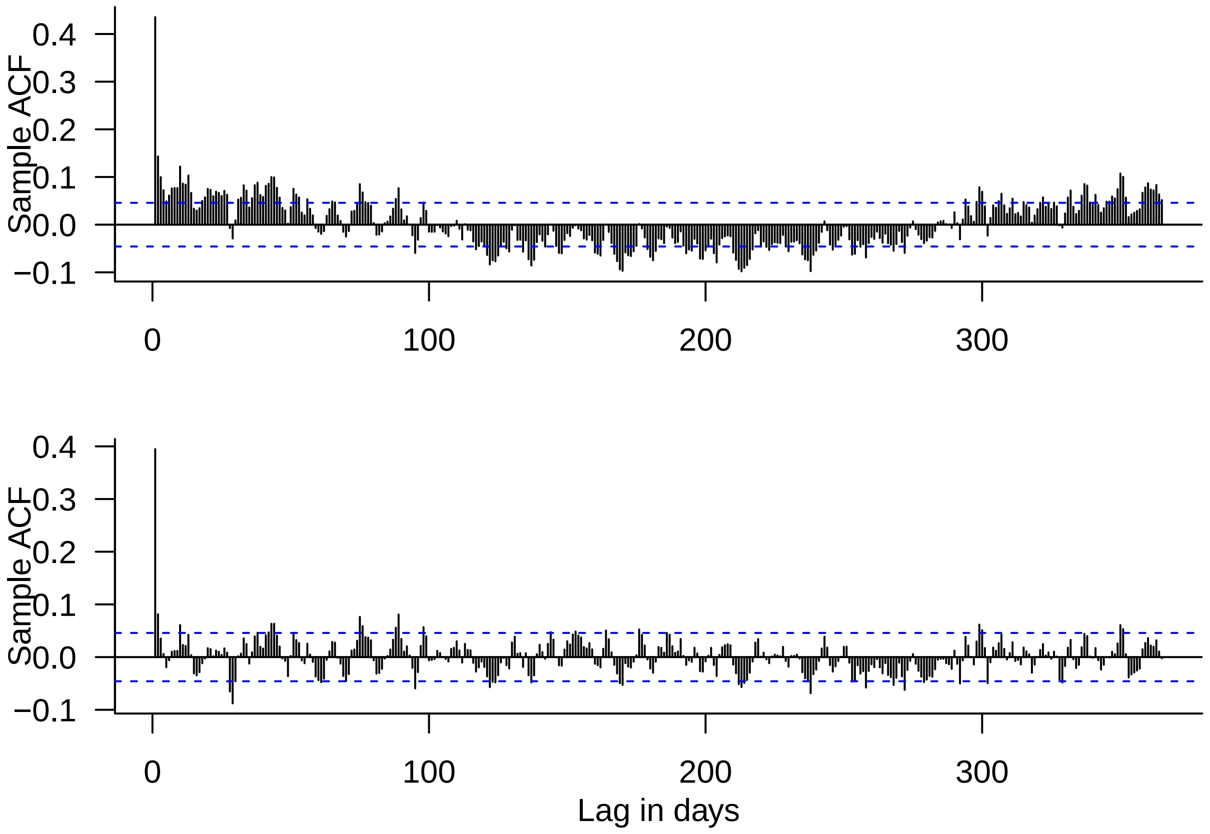
Let's plot the ACF and PACF plots to investigate the possible order for the ARMA model.

```
library(forecast)
```

```
## Registered S3 method overwritten by 'quantmod':
```

```
##   method             from
##   as.zoo.data.frame zoo
```

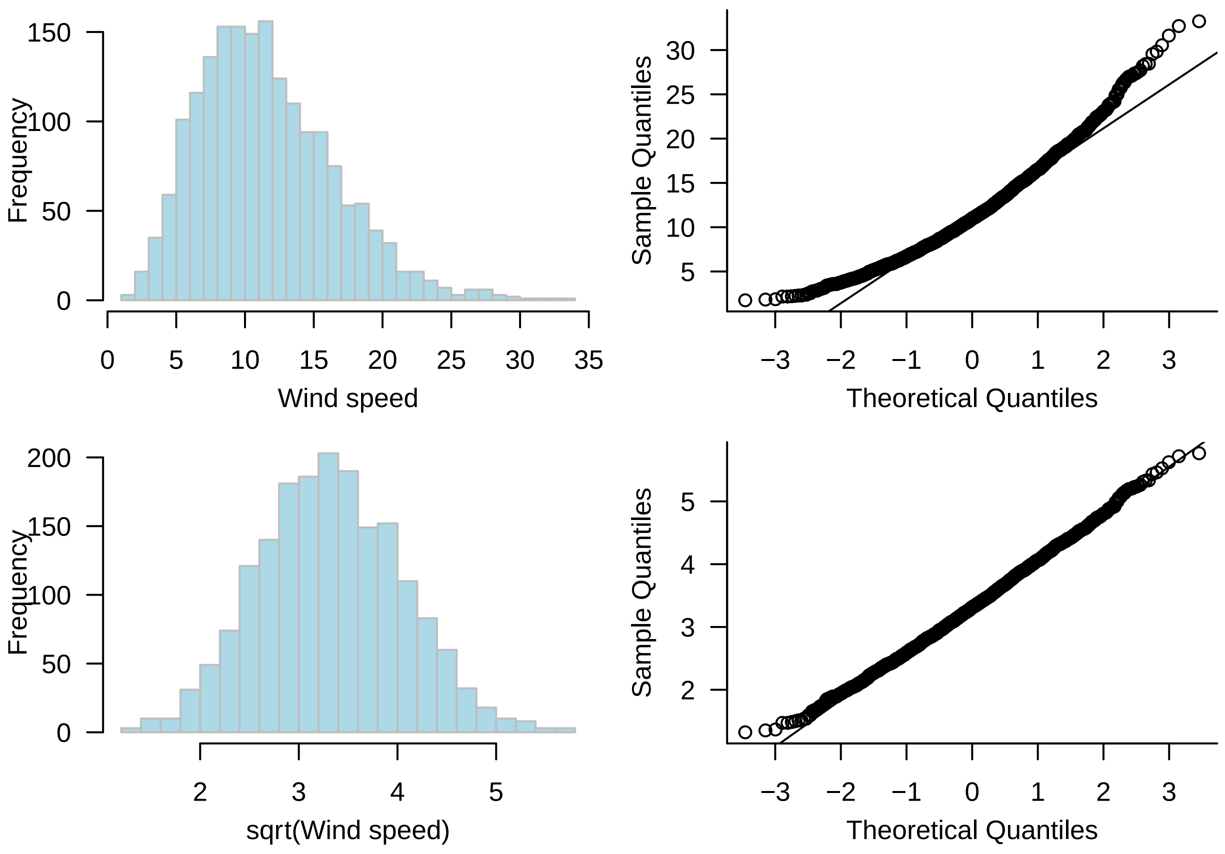
```
par(bty = "L", mar = c(3.6, 3.6, 0.5, 0.6), mgp = c(2, 1, 0), las = 1,
    mfrow = c(2, 1))
Acf(rosslare, lag.max = 365, xlab = "", ylab = "Sample ACF", main = "")
Acf(resid(harm.model), lag.max = 365, xlab = "Lag in days",
    ylab = "Sample ACF", main = "")
```



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), 25, main = "", xlab = "sqrt(Wind speed)", col = "lightblue", border = "gray")
qqnorm(sqrt(rosslare), main=""); qqline(sqrt(rosslare))
```



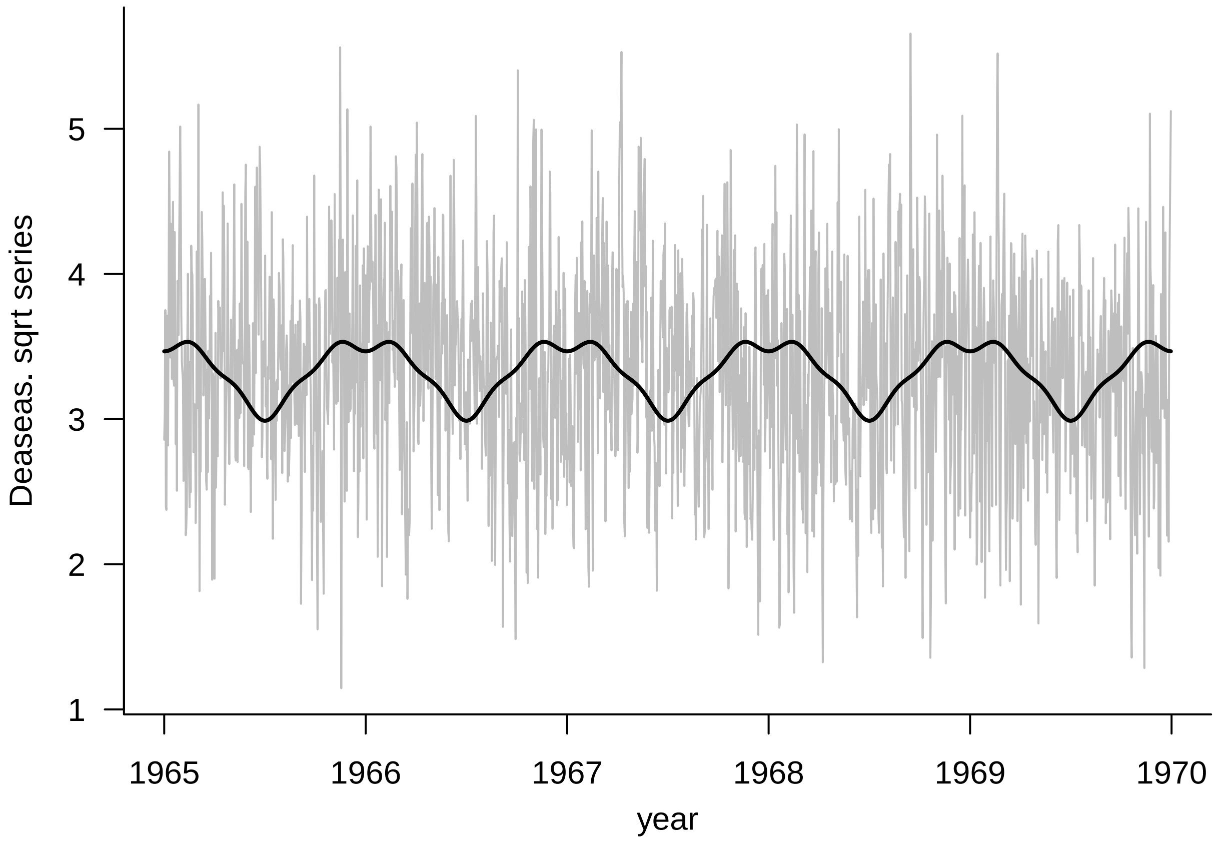
Now take square roots of the original data and deseasonalizeagain!

```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
##  1.146  2.848  3.316  3.799  5.656
##
## Coefficients:
##                       Estimate Std. Error t value Pr(>|t|)
## harmonics[, 1:4]1  0.2391111  0.1126203   2.123   0.0339 *
## harmonics[, 1:4]2 -0.0606520  0.1126203  -0.539   0.5903
## harmonics[, 1:4]3 -0.0001588  0.1126203  -0.001   0.9989
## harmonics[, 1:4]4 -0.0363877  0.1126202  -0.323   0.7467
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

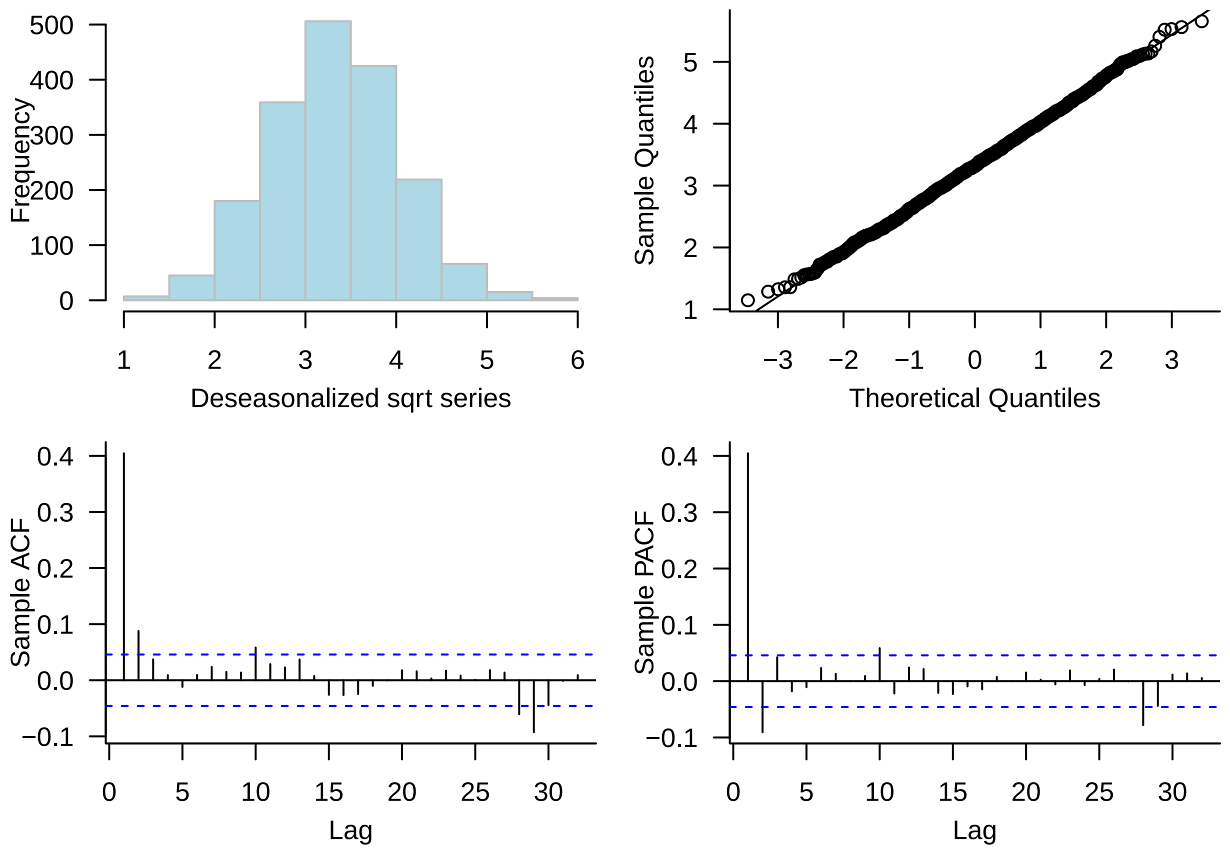
```
##
## Residual standard error: 3.403 on 1822 degrees of freedom
## Multiple R-squared:  0.002684,   Adjusted R-squared:  0.0004944
## F-statistic: 1.226 on 4 and 1822 DF,  p-value: 0.2978
```

```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")
Acf(sqrt.rosslare.ds, main = "", type = "partial", ylab = "Sample PACF")
```



Model identifcation, fitting, and selection

Let's first fit an AR(1) Fit an AR(1) model

```
ar1.model <- arima(sqrt.rosslare.ds, order = c(1, 0, 0))
```

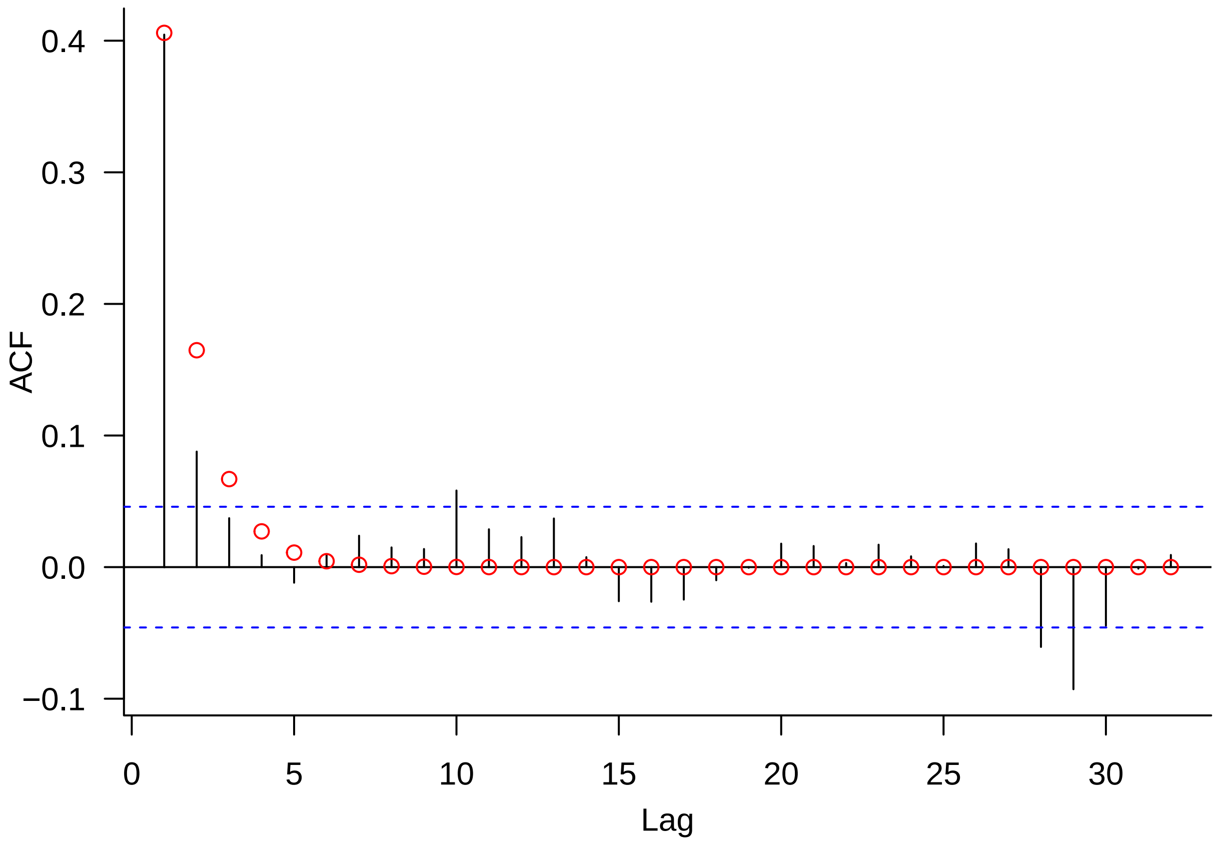
Summarize the fitted model

```
ar1.model
```

```
##
## Call:
## arima(x = sqrt.rosslare.ds, order = c(1, 0, 0))
##
## Coefficients:
##          ar1  intercept
##       0.4060     3.3257
## s.e.  0.0214     0.0254
##
## sigma^2 estimated as 0.4148:  log likelihood = -1787.72,  aic = 3581.43
```

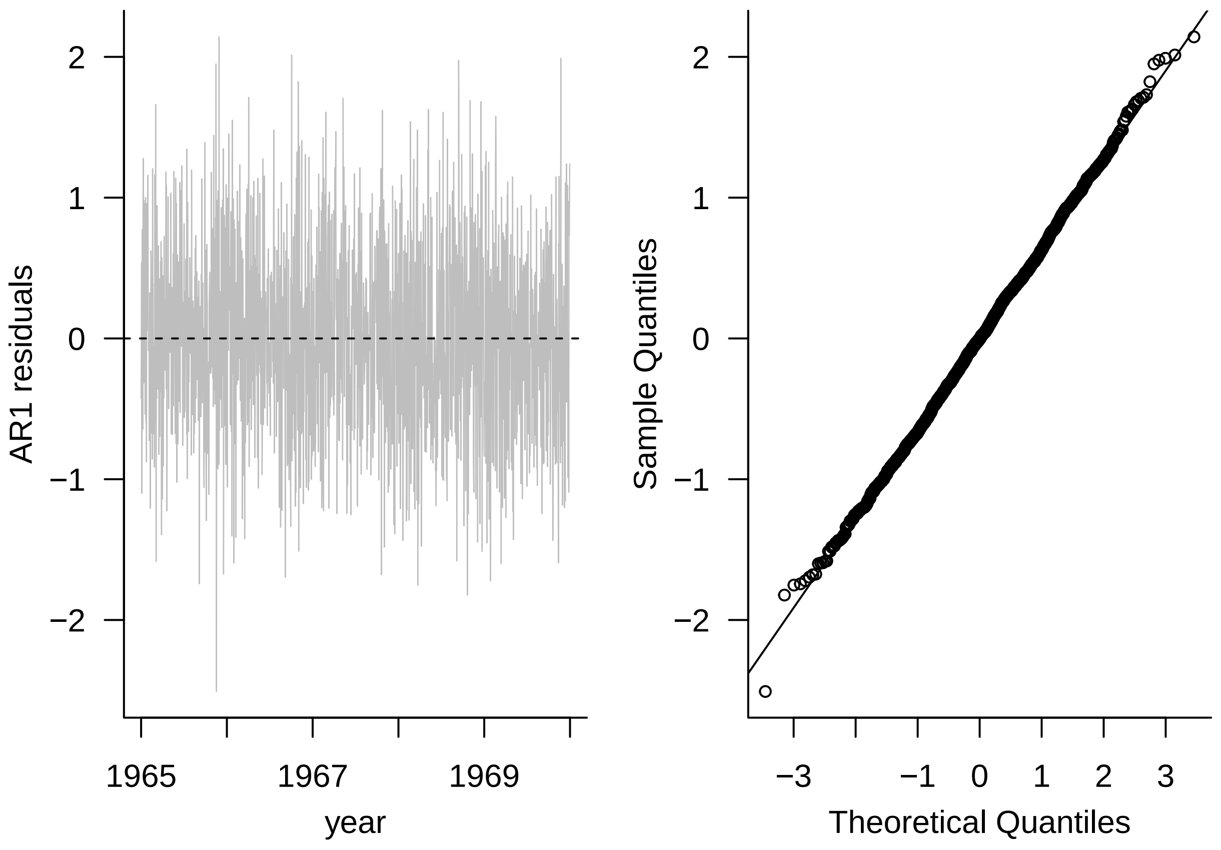
Sample and fitted ACF

```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")
```



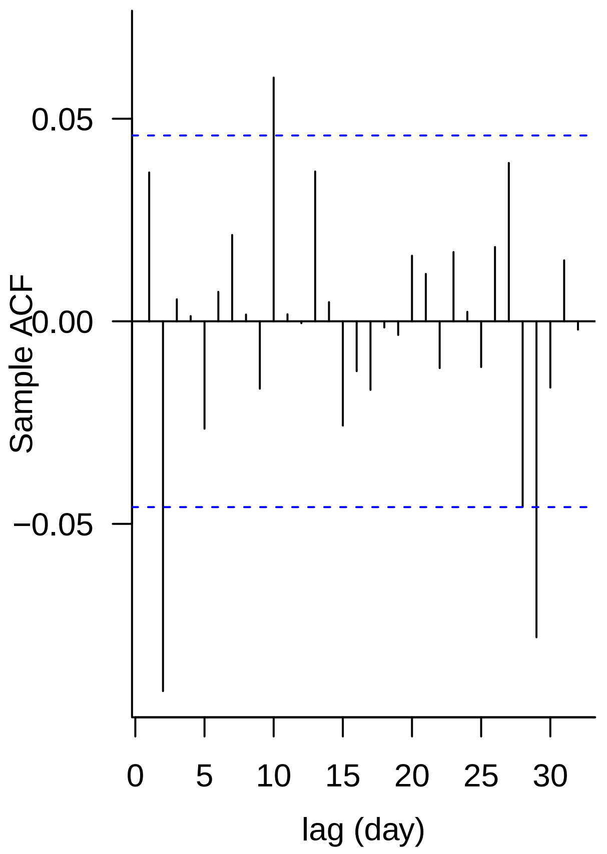
Extract residuals

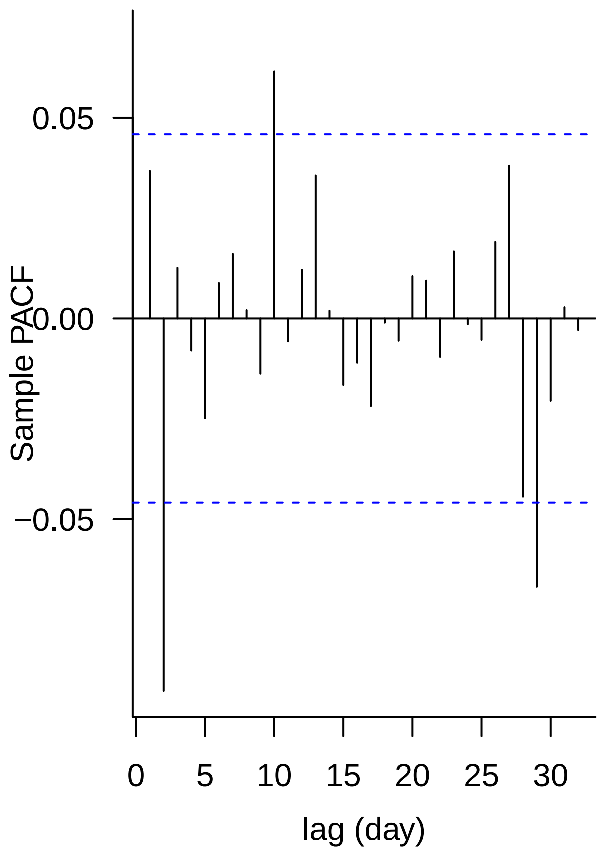
```r
ar1.resids <- resid(ar1.model)
## time series plot of the residuals
par(bty = "L", mar = c(3.6, 3.6, 0.5, 0.6), las = 1, mgp = c(2.2, 1, 0),
    mfrow = c(1, 2))
plot(year, ar1.resids, type = "l", xlab = "year", ylab = "AR1 residuals",
     lwd = 0.6, col = "gray")
abline(h = 0, lty = 2)
## Normal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



Sample ACF and PACF of the residuals

```r
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 = "")
Acf(ar1.resids, ylab = "Sample PACF",type = "partial",  xlab = "lag (day)")
```





```
## Carry out the Ljung-Box test
Box.test(ar1.resids, lag = 32, type = "Ljung-Box", fitdf = 1)
```

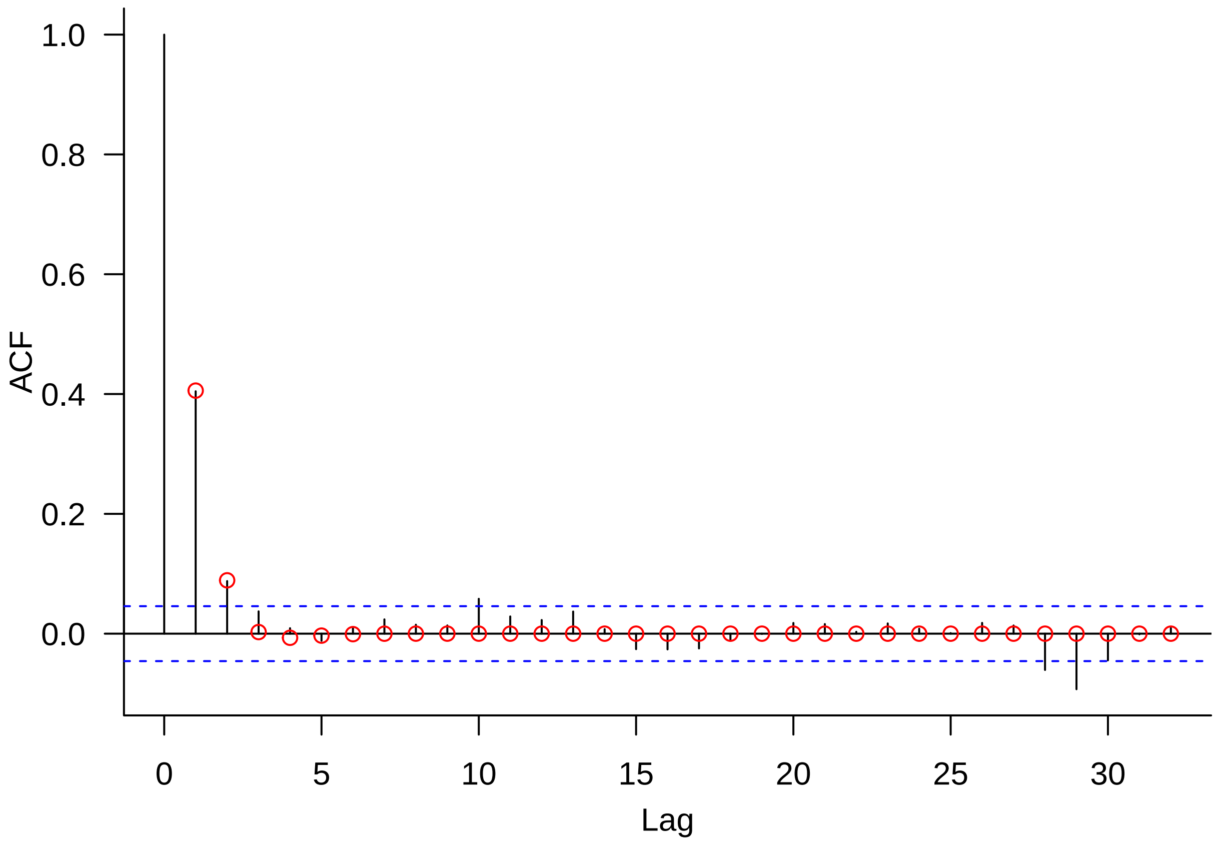
```
##
##  Box-Ljung test
##
## data:  ar1.resids
## X-squared = 53.142, df = 31, p-value = 0.00794
```

```
(ar2.model <- arima(sqrt.rosslare.ds, order = c(2, 0, 0)))
```

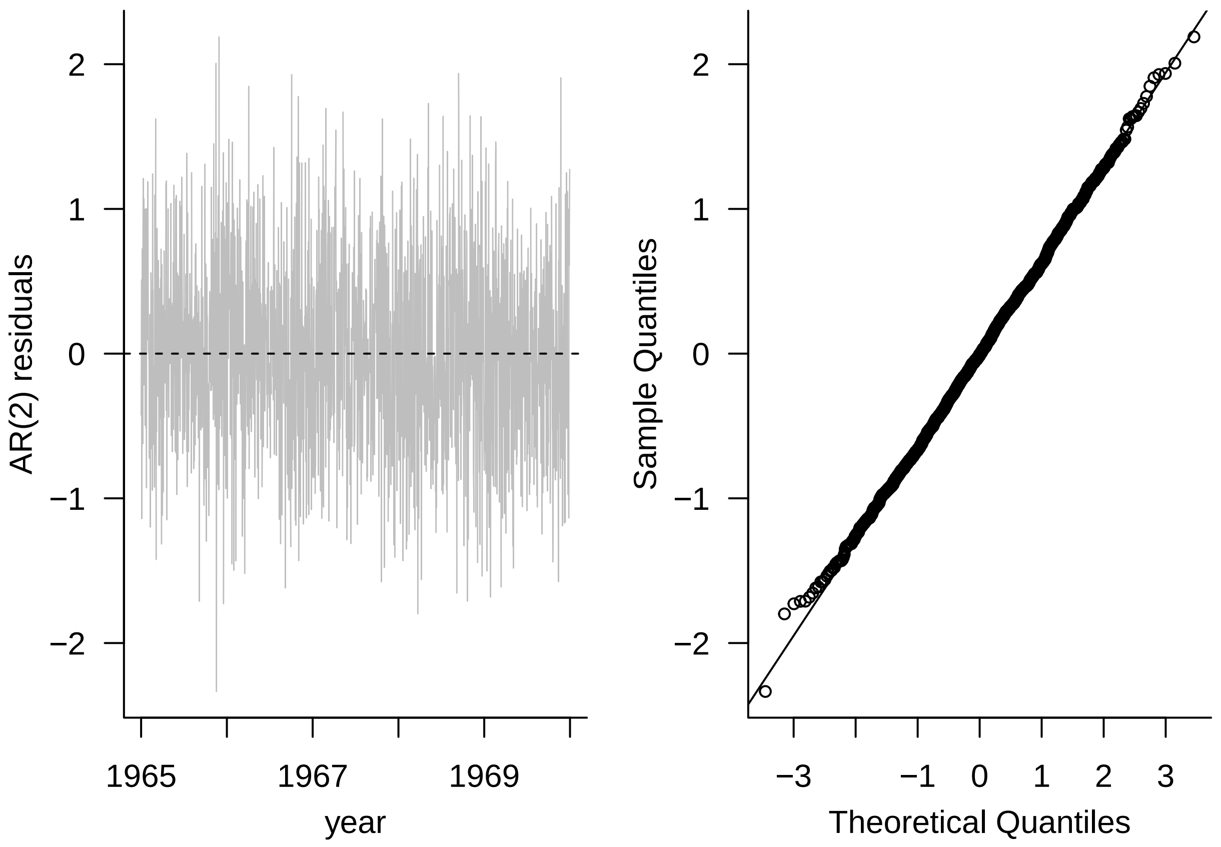
Fit an AR(2) model

```
##
## Call:
## arima(x = sqrt.rosslare.ds, order = c(2, 0, 0))
##
## Coefficients:
##          ar1      ar2  intercept
##       0.4425  -0.0905     3.3254
## s.e.  0.0233   0.0233     0.0232
##
## sigma^2 estimated as 0.4114:  log likelihood = -1780.23,  aic = 3568.46
```

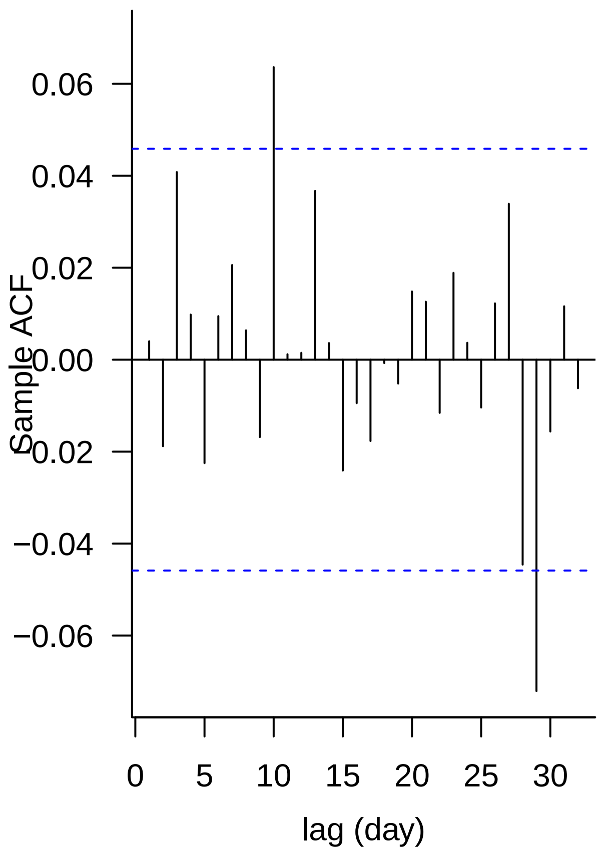
```r
par(bty = "L", mar = c(3.6, 3.6, 0.5, 0.6), las = 1, mgp = c(2.2, 1, 0))
acf(sqrt.rosslare.ds, main = "")
acf_true <- ARMAacf(ar = c(ar2.model$coef[1:2]), lag.max = 32)[-1]
points(1:32, acf_true, col = "red")
```

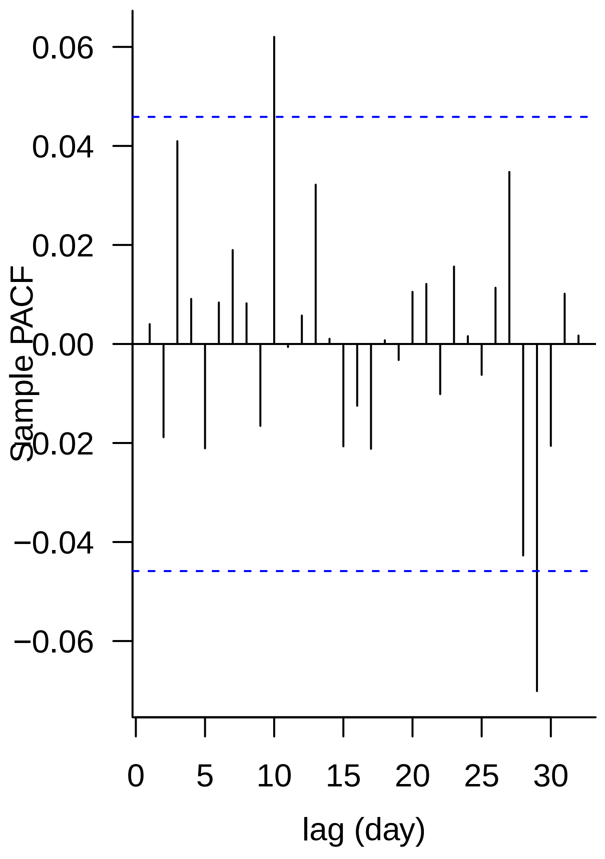


```r
## extract the residuals
ar2.resids <- resid(ar2.model)

## 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)")
```





```
## Carry out the Ljung-Box test
Box.test(ar2.resids, lag = 32, type = "Ljung-Box", fitdf = 2)
```

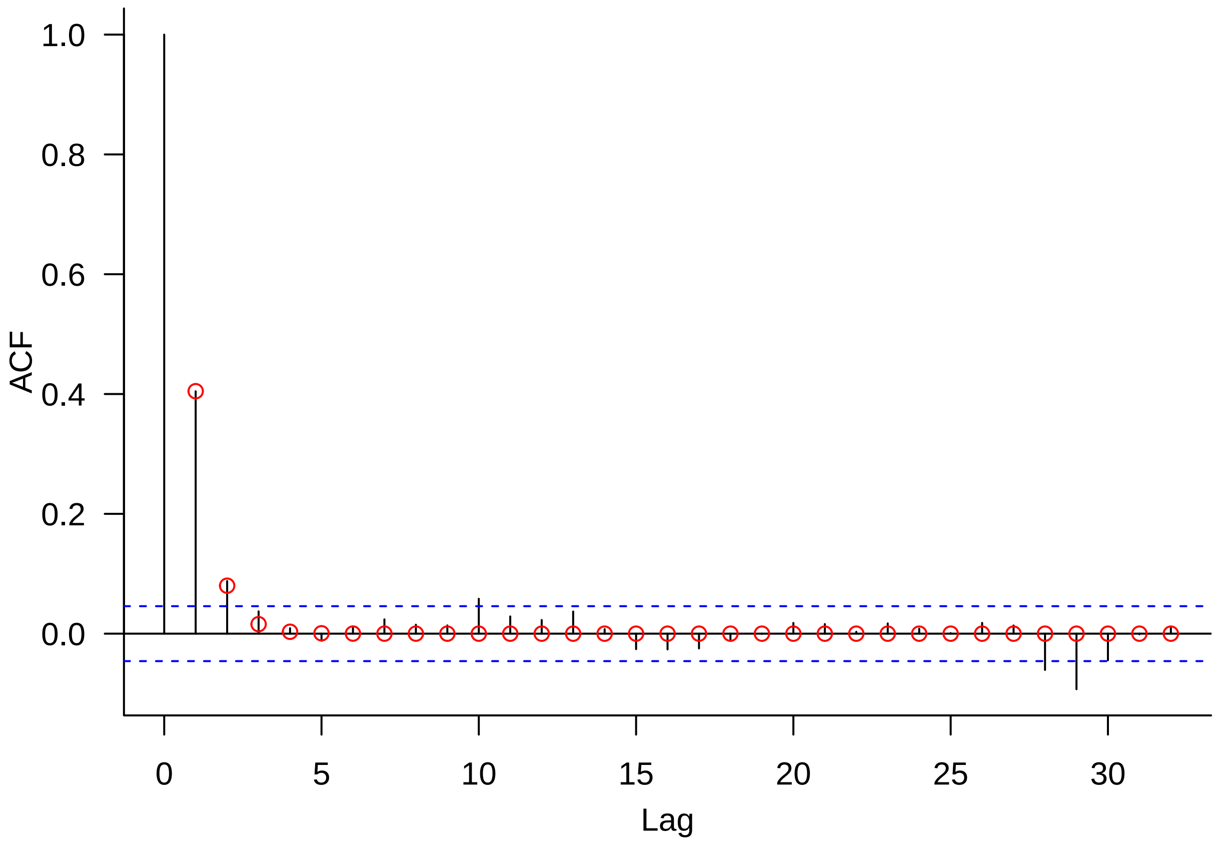
```
##
##  Box-Ljung test
##
## data:  ar2.resids
## X-squared = 36.548, df = 30, p-value = 0.1907
```

```

(arma11.model <- arima(sqrt.rosslare.ds, order = c(1, 0, 1)))
```

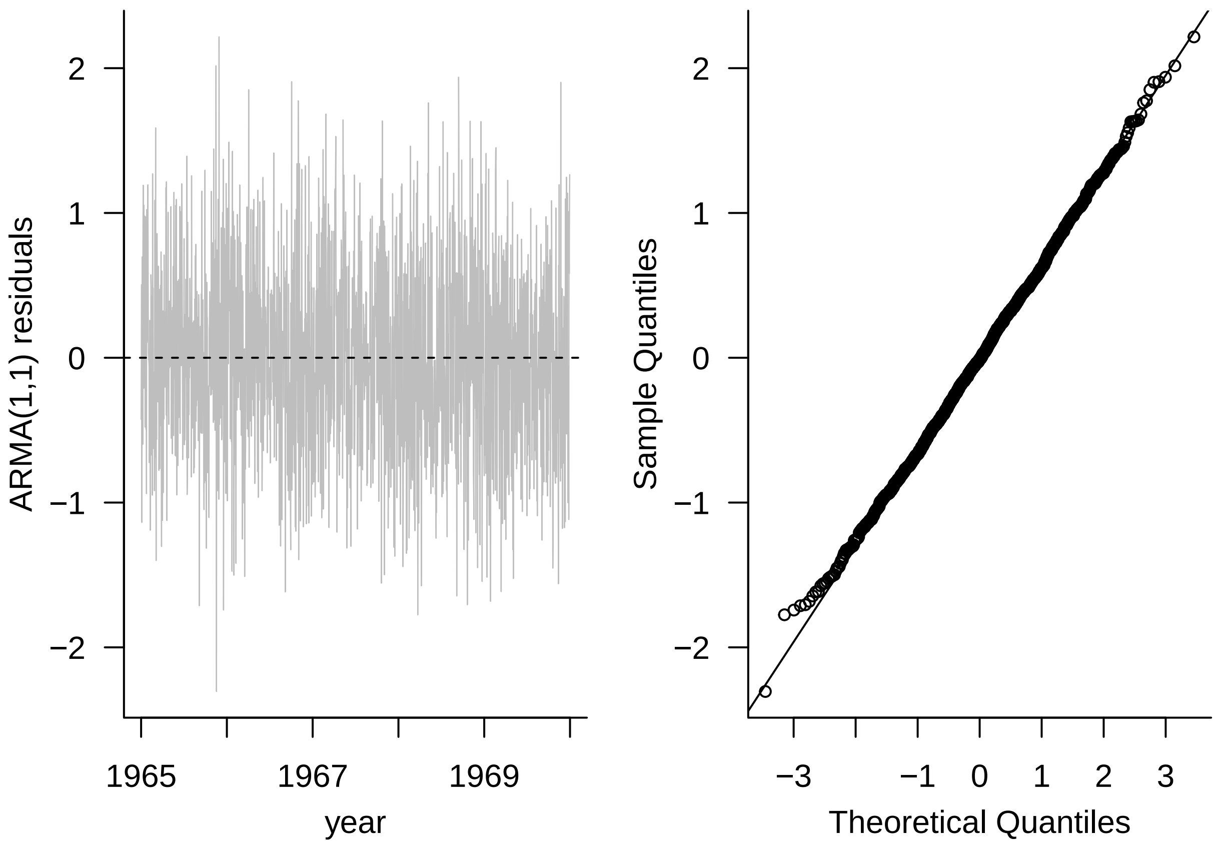
Fit an ARMA(1,1) model

```
##
## Call:
## arima(x = sqrt.rosslare.ds, order = c(1, 0, 1))
##
## Coefficients:
##          ar1     ma1  intercept
##       0.1978  0.2502     3.3254
## s.e.  0.0556  0.0553     0.0234
##
## sigma^2 estimated as 0.4108:  log likelihood = -1778.82,  aic = 3565.64
```

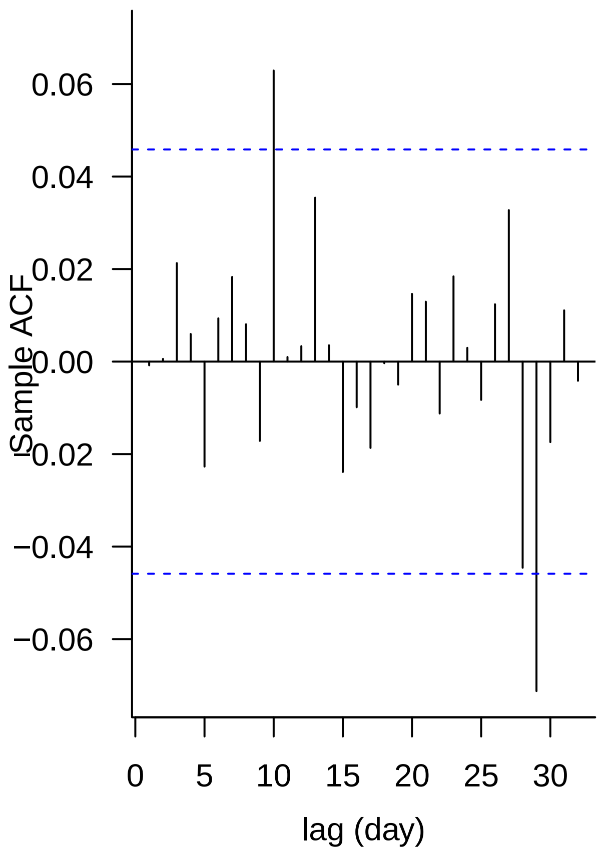
```r
par(bty = "L", mar = c(3.6, 3.6, 0.5, 0.6), las = 1, mgp = c(2.2, 1, 0))
acf(sqrt.rosslare.ds, main = "")
acf_true <- ARMAacf(ar = arma11.model$coef[1], ma = arma11.model$coef[2], lag.max = 32)[-1]
points(1:32, acf_true, col = "red")
```

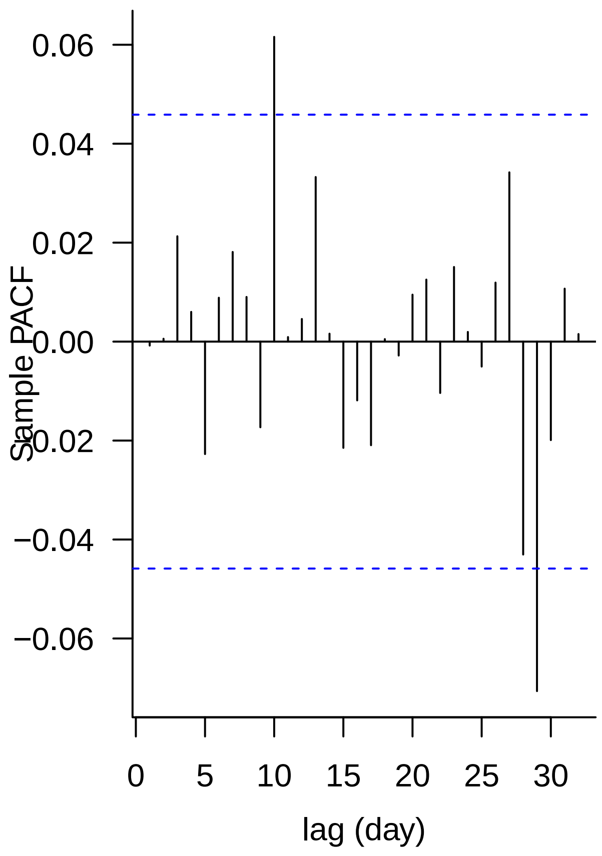


```r
## extract the residuals
arma11.resids <- resid(arma11.model)
## time series plot of the residuals
par(bty = "L", mar = c(3.6, 3.6, 0.5, 0.6), las = 1, mgp = c(2.2, 1, 0), mfrow = c(1, 2))
plot(year, arma11.resids, type = "l", xlab = "year",
     ylab = "ARMA(1,1) residuals", lwd = 0.6, col = "gray")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



```
## 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 Ljung-Box test
Box.test(arma11.resids, lag = 32, type = "Ljung-Box", fitdf = 2)
```

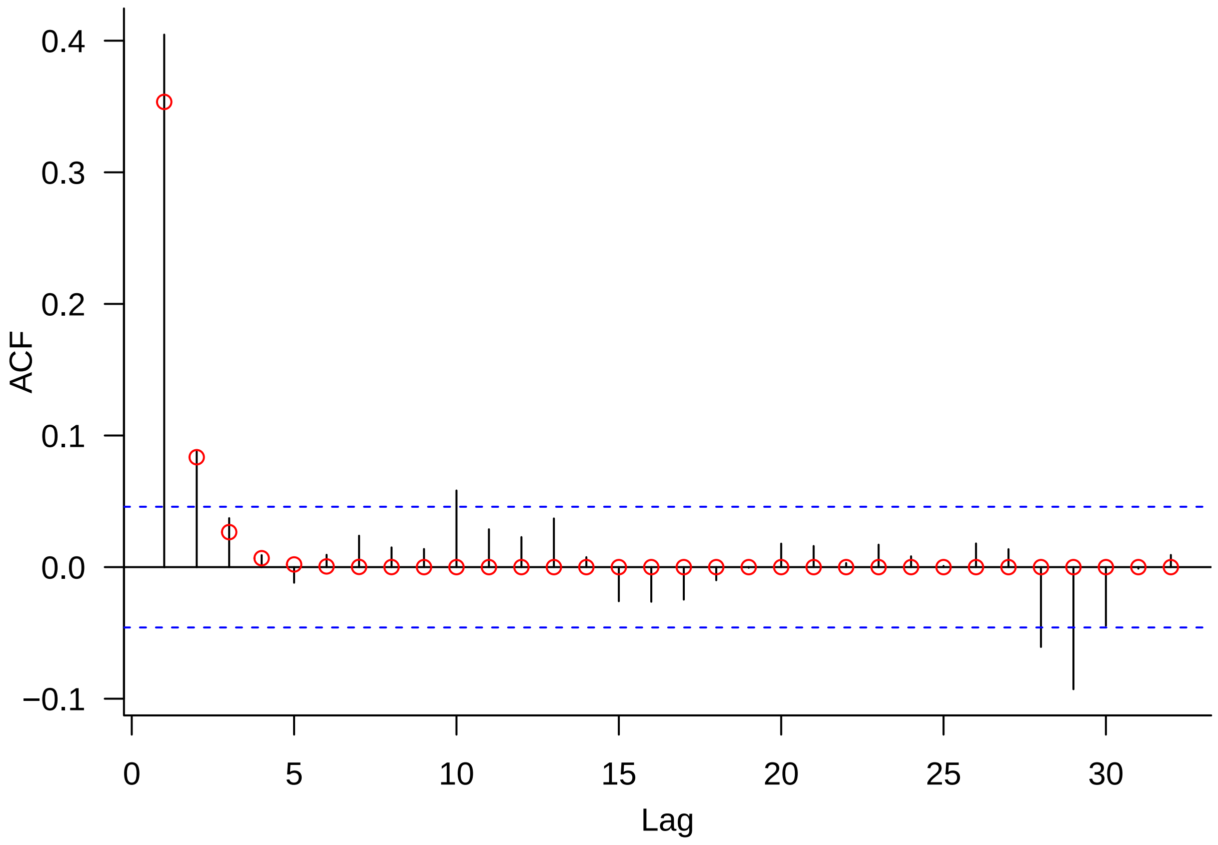
```
##
##  Box-Ljung test
##
## data:  arma11.resids
## X-squared = 32.757, df = 30, p-value = 0.3332
```

```r
(arma21.model <- arima(sqrt.rosslare.ds, order = c(2, 0, 1)))
```

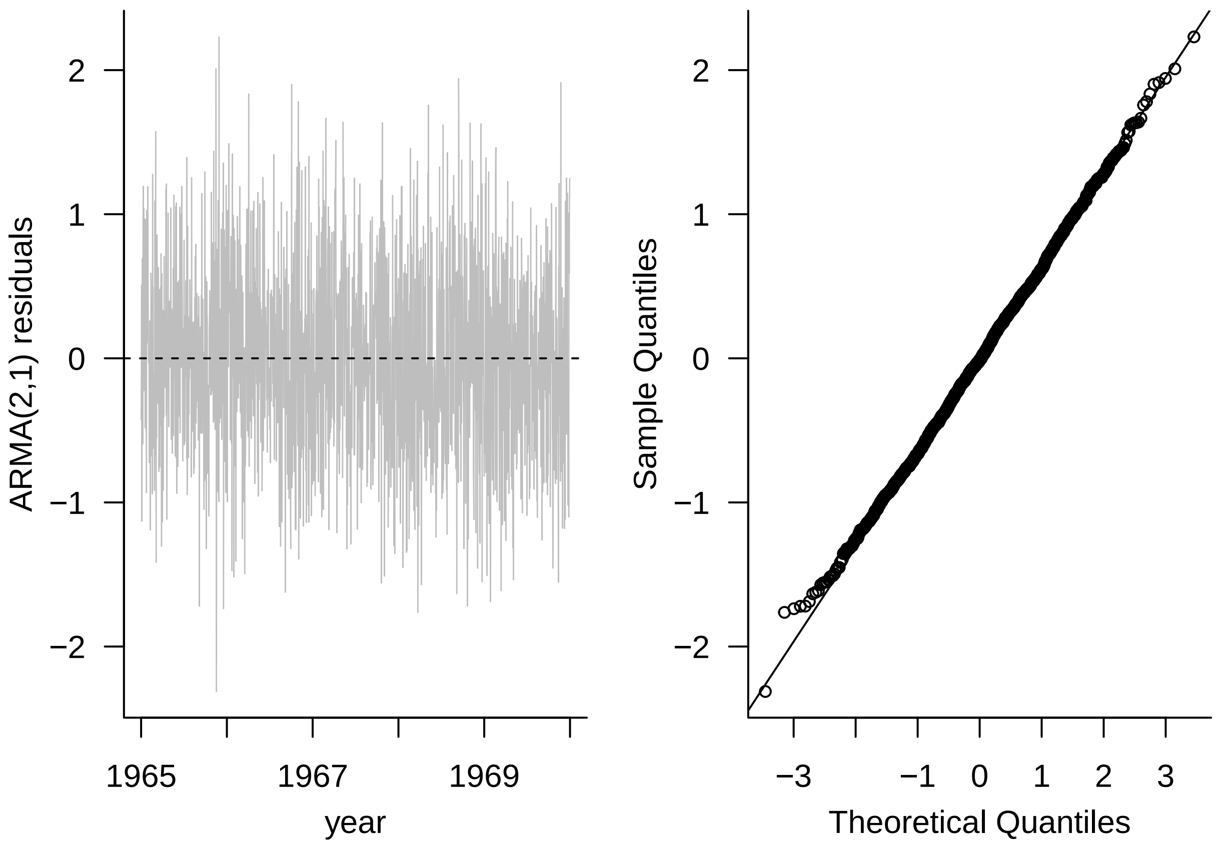
Fit an ARMA(2,1) model

```
##
## Call:
## arima(x = sqrt.rosslare.ds, order = c(2, 0, 1))
##
## Coefficients:
##          ar1     ar2     ma1  intercept
##       0.0703  0.0587  0.3768     3.3253
## s.e.  0.1691  0.0772  0.1663     0.0237
##
## sigma^2 estimated as 0.4107:  log likelihood = -1778.56,  aic = 3567.11
```

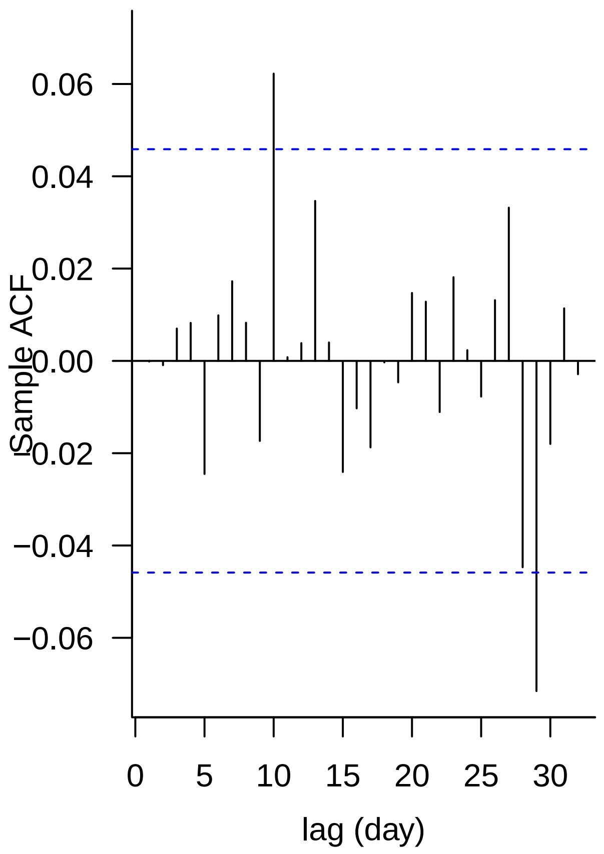
```r
par(bty = "L", mar = c(3.6, 3.6, 0.5, 0.6), las = 1, mgp = c(2.2, 1, 0))
Acf(sqrt.rosslare.ds, main = "")
acf_true <- ARMAacf(ar = arma21.model$coef[1:2], ma = arma11.model$coef[3], lag.max = 32)[-1]
points(1:32, acf_true, col = "red")
```

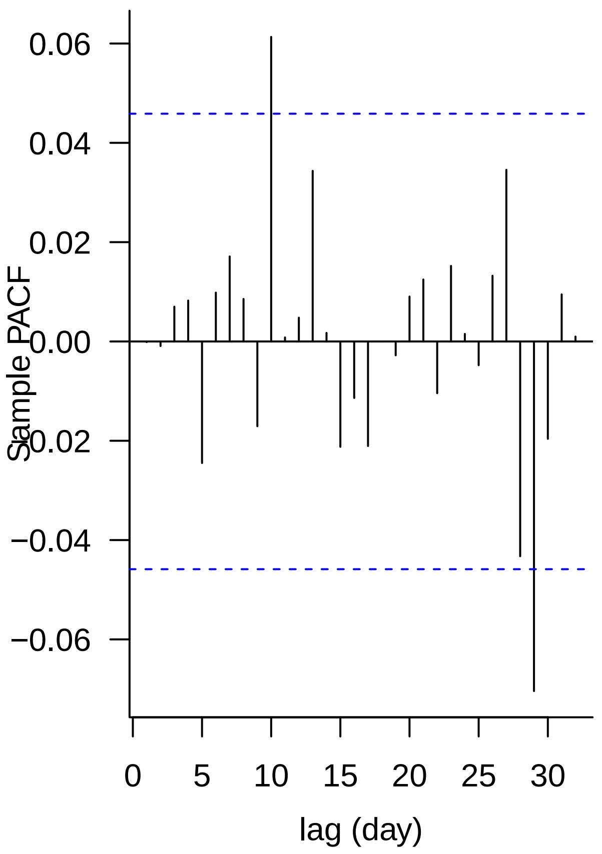


```r
## extract the residuals
arma21.resids <- resid(arma21.model)
## time series plot of the residuals
par(bty = "L", mar = c(3.6, 3.6, 0.5, 0.6), las = 1, mgp = c(2.2, 1, 0), mfrow = c(1, 2))
plot(year, arma21.resids, type = "l", xlab = "year",
     ylab = "ARMA(2,1) residuals", lwd = 0.6, col = "gray")
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 Ljung-Box test
Box.test(arma21.resids, lag = 32, type = "Ljung-Box", fitdf = 3)
```

```
##
##  Box-Ljung test
##
## data:  arma21.resids
## X-squared = 32.171, df = 29, p-value = 0.3124
```

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] 3581.432
```

```
ar2.model$aic
```

```
## [1] 3568.46
```

```
arma11.model$aic
```

```
## [1] 3565.642
```

```
arma21.model$aic
```

```
## [1] 3567.112
```

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

```
## [1] 3581.438
```

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

```
## [1] 3568.473
```

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

```
## [1] 3565.655
```

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

```
## [1] 3567.134
```

Based on the AIC (and AICc as well), we choose the ARMA(1,1) model.

Forecasting

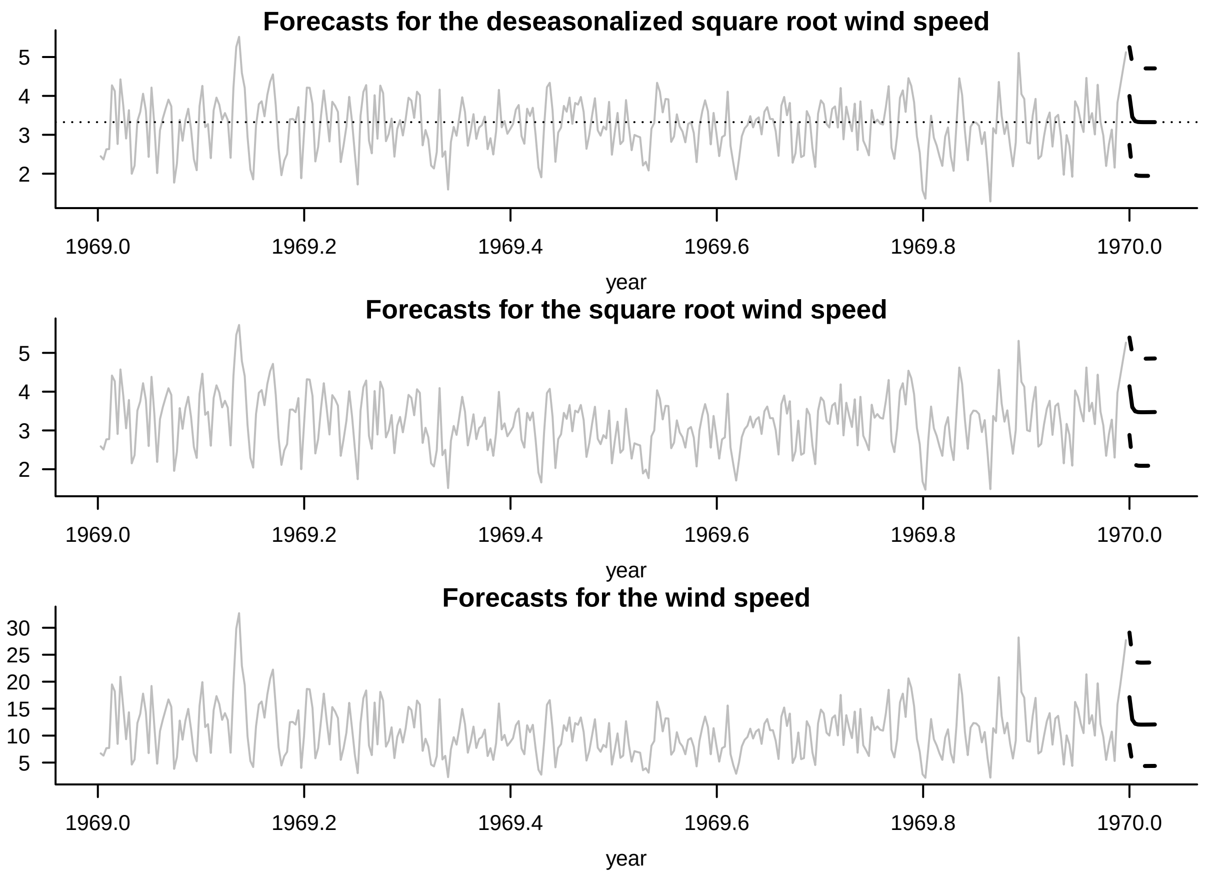
```
## How many days will we predict into the future?
h <- 10
## Predict 'h' days into the future using the ARMA(1,1) model.
sqrt.rosslare.forecast <- predict(arma11.model, h)
sqrt.rosslare.forecast$pred; sqrt.rosslare.forecast$se
```

```
## Time Series:
## Start = 1827
## End = 1836
## Frequency = 1
##  [1] 3.997161 3.458299 3.351724 3.330646 3.326477 3.325652 3.325489 3.325457
##  [9] 3.325451 3.325449
## Time Series:
## Start = 1827
## End = 1836
## Frequency = 1
##  [1] 0.6409326 0.7022959 0.7045876 0.7046771 0.7046806 0.7046807 0.7046807
##  [8] 0.7046807 0.7046807 0.7046807
```

```
## define the forecast variable
forecast <- sqrt.rosslare.forecast$pred
## The plus or minus value is the z critical value
## times the standard error for the forecast
me <- qnorm(0.975) * sqrt.rosslare.forecast$se
lower <- forecast - me
upper <- forecast + me
## Define the prediction time
fyear <- 1970 + (0:(h - 1)) / 365.25
```

Visualizing the Forecasts

```
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 10 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 - me
upper <- adj.forecast + me

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


Forecasts for the deseasonalized square root wind speed
year
Forecasts for the square root wind speed
year
Forecasts for the wind speed
year