

Business 4720 - Class 14

Time Series

Joerg Evermann

Faculty of Business Administration
Memorial University of Newfoundland
jevermann@mun.ca



Unless otherwise indicated, the copyright in this material is owned by Joerg Evermann. This material is licensed to you under the [Creative Commons by-attribution non-commercial license \(CC BY-NC 4.0\)](#)

What You Will Learn:

- ▶ Time Series Models
 - ▶ Time series basics
 - ▶ Smoothing methods
 - ▶ ARIMA models
 - ▶ Seasonal models
 - ▶ GARCH Models

Robert H. Shumway and David S. Stoffer (2017) *Time Series Analysis and Its Applications*, 4th Edition. Springer.

<https://www.stat.pitt.edu/stoffer/tsa4/>

Rob J. Hyndman and George Athanasopoulos (2018) *Forecasting: Principles and Practice*, 2nd edition. OTexts.

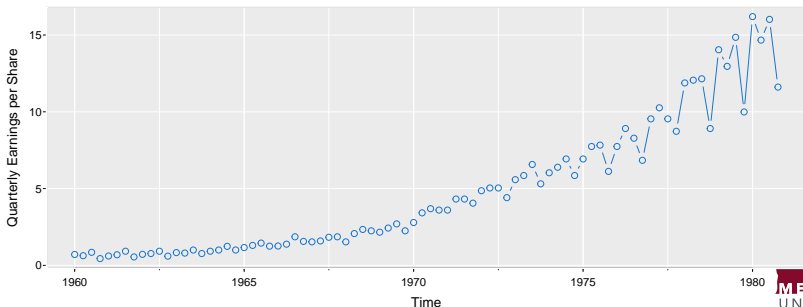
<https://otexts.com/fpp2/>

Useful Tutorials

- ▶ <https://github.com/nickpoison/tsa4>
- ▶ <https://a-little-book-of-r-for-time-series.readthedocs.io/en/latest/src/timeseries.html>
- ▶ <https://rc2e.com/timeseriesanalysis>
- ▶ <https://atsa-es.github.io/atsa-labs/chap-tslab.html>

Time Series Application Examples

- ▶ Finance: Stock market predictions
- ▶ Economics: Unemployment rates
- ▶ Social science: School enrollment
- ▶ Natural sciences: Global temperature trends
- ▶ Ecology: Fish population forecasting
- ▶ Epidemiology: COVID incidence rates



Characteristics of Time Series

Time-Domain Approach

- ▶ Focuses on lagged relationships
- ▶ *Example:* How does yesterday's stock performance affect today's stock performance?

Frequency-Domain Approach

- ▶ Focuses on cycles
- ▶ *Example:* What is the economic cycle through periods of expansion and recession?

Time Series Statistical Models

- ▶ Moving Averages
- ▶ Autoregressive Model
- ▶ Random walk with drift
- ▶ Signal in noise

Moving Averages

Example:

$$v_t = \frac{1}{3}(w_t + w_{t-1} + w_{t-2})$$

(Sum of coefficients equals 1)

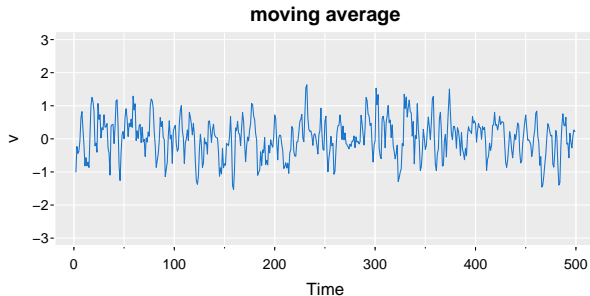
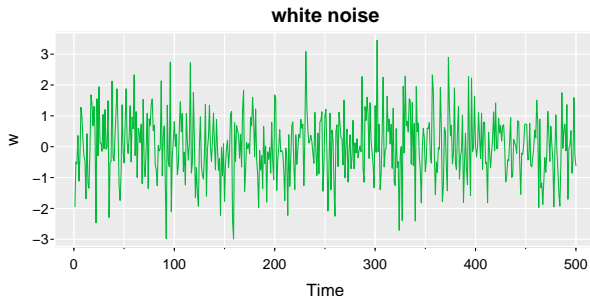
```
# Load ASTSA library
library(astsa)

# Random numbers
w = rnorm(500,0,1)

# Moving average
v = filter(w, sides=2, filter=rep(1/3,3))

# Plot timeseries
par(mfrow=c(2,1))
tsplot(w, ylim=c(-3,3), main="white noise", col=3)
tsplot(v, ylim=c(-3,3), main="moving average", col=4)
```

Moving Averages [cont'd]



Autoregressions

Example:

$$x_t = x_{t-1} - 0.9x_{t-2} + w_t$$

(Coefficients less or equal 1)

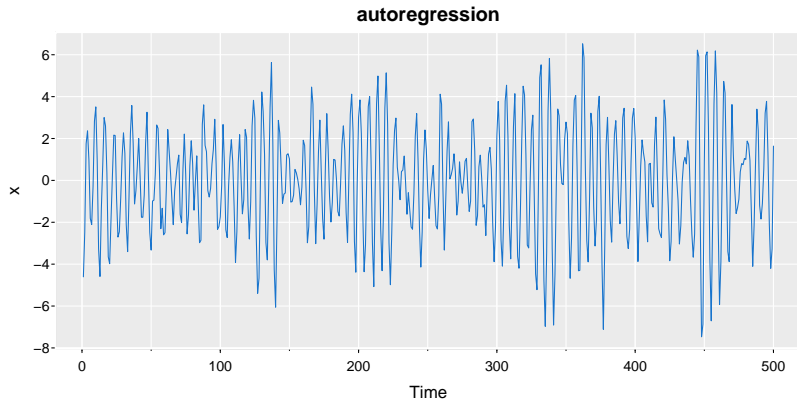
```
# Random numbers (errors)
w = rnorm(550,0,1)

# Autoregressive time series
x = filter(w, filter=c(1,-.9), method="recursive")

# Remove first 50 values for startup
x = x[-(1:50)]

# Plot time series
tsplot(x, main="autoregression", col=4)
```

Autoregressions [cont'd]



Random Walk with Drift

Example:

$$\begin{aligned}x_t &= \delta + x_{t-1} + w_t \\ &= \delta t + \sum_{j=1}^t w_j\end{aligned}$$

(Coefficient of x_{t-1} is 1)

```
# White noise
w = rnorm(200)
# Random walk
x = cumsum(w)

# Drift
drift = .2
# White noise with drift
w.drift = w + drift;
# Random walk with drift
x.drift = cumsum(w.drift)

# Plot
tsplot(x.drift, main="random walk", col=3,
        ylab='', ylim=c(-10,55))
abline(a=0, b=drift, lty=2, col=3)
lines(x, col=4)
abline(a=0, b=0, lty=2, col=4)
```

Random Walk with Drift [cont'd]



Signal in Noise

Example:

$$x_t = A \cos(2\pi\omega t + \phi)$$

$$A = 2$$

amplitude

$$\omega = 1/50$$

frequency

$$\phi = .6\pi$$

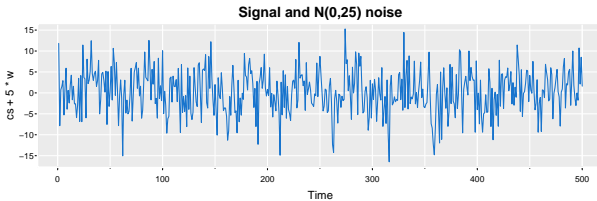
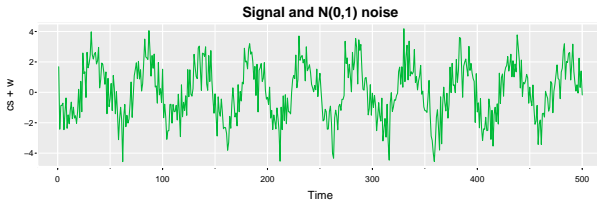
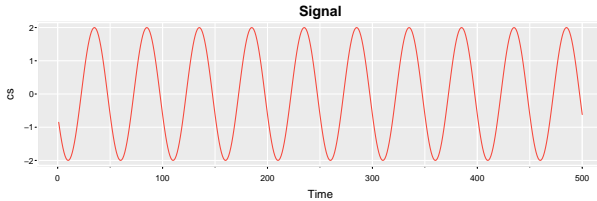
phase shift

```
# Create signal
cs = 2*cos(2*pi*1:500/50 + .6*pi)

# Create noise
w = rnorm(500,0,1)

# Plot signal with different amount of noise
par(mfrow=c(3,1), mar=c(3,2,2,1), cex.main=1.5)
tsplot(cs, main='Signal', col=2)
tsplot(cs+w, main='Signal and N(0,1) noise', col=3)
tsplot(cs+5*w, main='Signal and N(0,25) noise', col=4)
```

Signal in Noise [cont'd]



Basic Time Series Operations in R

Building a time series:

```
# Creating a time series object with monthly data  
ts_data <- ts(1:24, frequency = 12, start = c(2020, 1))
```

Dealing with missing values:

```
library(zoo)  
  
# Introduce NA values  
ts_data[c(5, 10, 15)] <- NA  
  
# Remove leading/trailing NA values  
trimmed_ts <- na.trim(ts_data)  
  
# Last Observation Carried Forward  
locf_ts <- na.locf(ts_data)  
  
# Interpolate NA values  
approx_ts <- na.approx(ts_data)
```

Basic Time Series Operations in R

First and last observations:

```
head(ts_data)
tail(ts_data)
```

Merging time series:

```
# Creating another time series
ts_data2 <- ts(c(1:24), frequency = 12, start = c(2020, 7))

# Find intersection of two time series
intersect_ts <- ts.intersect(ts_data, ts_data2)

# Find union of two time series
union_ts <- ts.union(ts_data, ts_data2)
```


Basic Time Series Operations in R

Lagging a series (shifting forward or backward in time):

```
# Positive k shifts backwards  
lag(ts_data, 6)  
  
# Negative k shifts forwards  
lag(ts_data, -3)
```

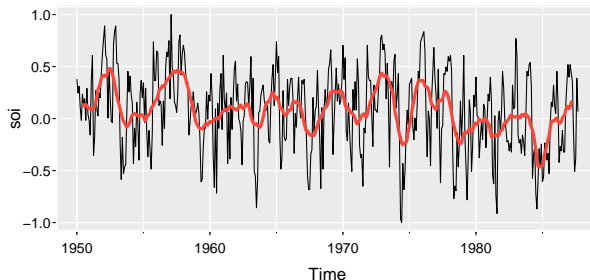
Smoothing a Time Series

- ▶ Moving average
- ▶ Kernel smoothing
- ▶ KNN regression / Lowess
- ▶ Smoothing splines

Moving Average

$$m_t = \sum_{j=-k}^k a_j x_{t-j} \quad \text{where} \quad \sum_{j=-k}^k a_j = 1$$

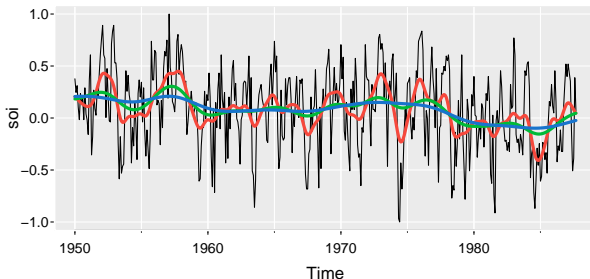
```
f = c(.5, rep(1, 11), .5)/12  
filter(soi, sides=2, filter=f)
```



Kernel Smoothing

$$a_i(t) = K\left(\frac{t-i}{b}\right) / \sum_{j=1}^n K\left(\frac{t-j}{b}\right) \text{ and } K(z) = \frac{1}{\sqrt{2\pi}} \exp(-z^2/2)$$

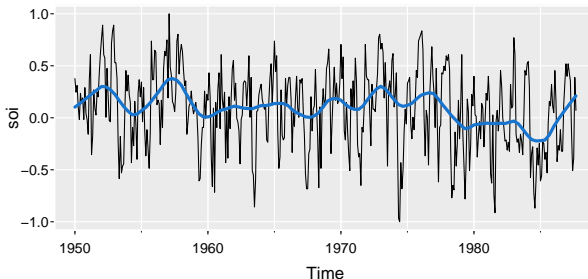
```
ksmooth(time(soi), soi, 'normal', bandwidth=1)
```



Locally Weighted Regression

- Identify proportion f of closest points
- Use weighted least squares regression to predict values

```
lowess(soi, f=0.1)
```

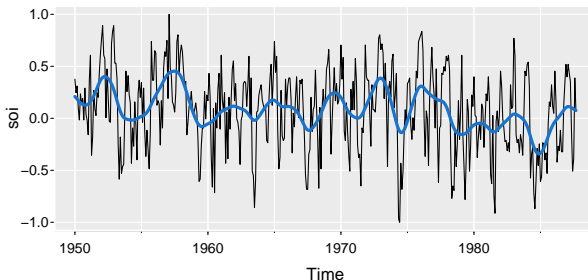


Smoothing Splines

- ▶ Penalized polynomial regression
- ▶ Fit $m_t = \beta_0 + \beta_1 t + \beta_2 t^2$
- ▶ Minimize loss function:

$$\sum_{t=1}^n (x_t - m_t)^2 + \lambda \int \left(\frac{d^2 m}{dt^2} \right)^2 dt$$

```
smooth.spline(time(soi), soi, spar=0.5)
```



Hands-On Exercises

- 1 Generate 100 observations from the autoregression model $x_t = -.9x_{t-2} + w_t$ with $\sigma_w^2 = 1$
 - 1.1 Smooth the time series using a moving average filter $v_t = (x_t + x_{t-1} + x_{t-2} + x_{t-3})/4$
 - 1.2 Plot x_t as a line and superimpose v_t
 - 1.3 Comment on the behaviour of x_t and how applying the moving average filter changes that behavior
- 2 Repeat the smoothing but with $x_t = \cos(2\pi t/4)$
- 3 Repeat the smoothing but with added $N(0, 1)$ noise, that is smooth $x_t = \cos(2\pi t/4) + w_t$
- 4 Compare and contrast the three exercises: How does the moving average change each series

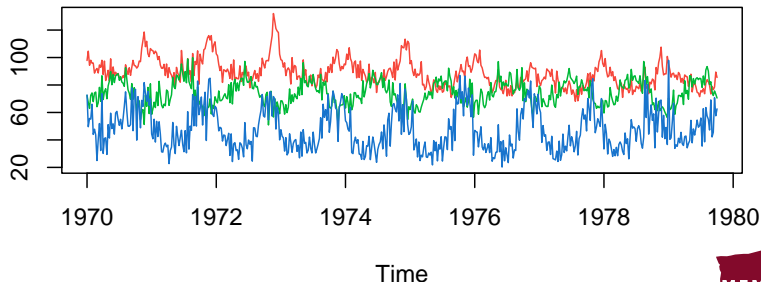
-
- ▶ Use `x = filter(w, filter=c(a, b), method="recursive")`
 - ▶ Use `v = filter(x, rep(1/4, 4), sides=1)` as above.
 - ▶ Use `c = cos(2*pi*1:500/4)` as above.
 - ▶ Use `lines(..., lty=2)` to add lines to a plot

Time Series Regression Example

Epidemiology Example:

- ▶ Cardiovascular mortality `cmort`
- ▶ Temperate `tempr`
- ▶ Particulate pollution `part`

```
# IMPORTANT: Use ts.plot, NOT tsplot:  
ts.plot(cmort, tempr, part, col=2:4)
```



Time Series Regression Example [cont'd]

Fit different linear regression models (at the same time points):

```
# Center Temperature
temp = tempr - mean(tempr)

# Square Temperature
temp.2 = temp^2

# Fit different linear models and provide summaries
summary(lm(cmort ~ time(cmort)))
summary(lm(cmort ~ time(cmort) + temp))
summary(lm(cmort ~ time(cmort) + temp + temp.2))
summary(lm(cmort ~ time(cmort) + temp + temp.2 + part))
```

Time Series Regression Example

With Lagged Variables

```
# Lag the temperature
temp.l.2 = lag(temp, 2)
temp.l.4 = lag(temp, 4)

# Intersect all time series to
# omit leading/trailing NA
temp.df <- ts.intersect(cmort, time(cmort), part,
                        temp, temp.2, temp.l.2,
                        temp.l.4,
                        dframe=TRUE)

# Fit the linear model including lagged temperature
summary(lm(cmort ~ time.cmort. + temp + temp.2 +
           temp.l.2 + temp.l.4 + part,
           data=temp.df))
```

A **strictly stationary** time series is one for which the probabilistic behaviour of every collection of values

$$\{x_{t_1}, x_{t_2}, \dots, x_{t_k}\}$$

is identical to that of the shifted set

$$\{x_{t_1+h}, x_{t_2+h}, \dots, x_{t_k+h}\}.$$

That is,

$$\Pr\{x_{t_1} \leq c_1, \dots, x_{t_k} \leq c_k\} = \Pr\{x_{t_1+h} \leq c_1, \dots, x_{t_k+h} \leq c_k\}$$

Measures of Dependence

Autocovariance

- Covariance between two points $t, t + h$ on time series x

$$\gamma(h) = \text{cov}(x_t, x_{t+h}) = E[(x_t - \mu)(x_{t+h} - \mu)]$$

- Large autocovariance \rightarrow smooth time series
- Small autocovariance \rightarrow choppy time series

Sample Autocovariance for Lag h

$$\hat{\gamma}(h) = \frac{1}{(q-p)} \sum_{t=p}^{q-h} (x_t - \bar{x})(x_{t+h} - \bar{x})$$

Autocorrelation Function (ACF)

$$\rho_x(h) = \frac{\gamma(t+h, t)}{\sqrt{\gamma(t+h, t+h)\gamma(t, t)}} = \frac{\gamma(h)}{\gamma(0)} \quad (\text{weak stationarity})$$

Sample Autocorrelation for Lag h

$$\hat{\rho}_x(h) = \frac{\hat{\gamma}(h)}{\sqrt{\hat{\gamma}(h)\hat{\gamma}(0)}} = \frac{\hat{\gamma}(h)}{\hat{\gamma}(0)} \quad (\text{weak stationarity})$$

Weak Stationarity

A **weakly stationary** time series is a process such that

- 1 the mean is constant and does not depend on time: $\mu_t = \mu$
- 2 the autocovariance γ depends on s and t only through their difference $h = |s - t|$.

Let $s = t + h$, then:

$$\gamma(s, t) = \gamma(t + h, t) = \text{cov}(x_{t+h}, x_t) = \text{cov}(x_h, x_0) = \gamma(h)$$

and

$$\rho(h) = \gamma(h)/\gamma(0)$$

The large-sample distribution of $\hat{\rho}_x(h)$ is normal with mean 0 and standard deviation

$$\sigma_{\hat{\rho}_x} = 1/\sqrt{n}$$

if the processes is independent white noise.

Hence, the approximate 95% confidence interval on the ACF is

$$-\frac{1}{n} \pm \frac{2}{\sqrt{n}}$$

Autocorrelation Function (ACF) Example 1

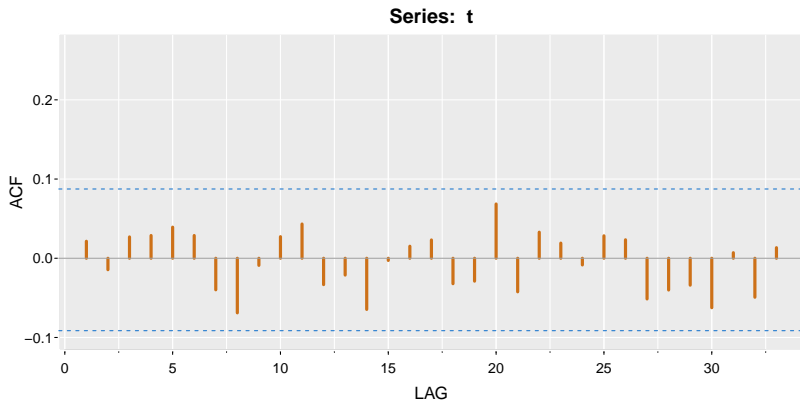
Gaussian white noise example:

```
library(astsa)
t <- ts(rnorm(500))

# The hard way:
cor(ts.intersect(t, lag(t,1), dframe=T))
cor(ts.intersect(t, lag(t,2), dframe=T))
# etc.

# The easy way:
# Without plot
acf <- acf1(t, plot=FALSE)
# With plot
acf1(t, col=7, lwd=3)
```


Autocorrelation Function (ACF) Example 1 [cont'd]



Autocorrelation Function (ACF) Example 2

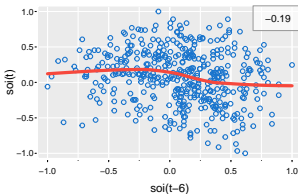
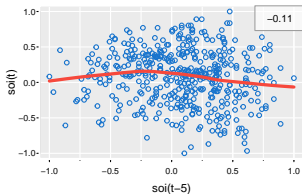
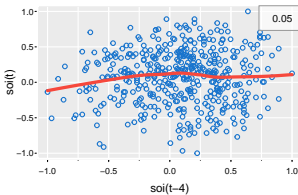
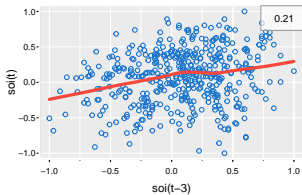
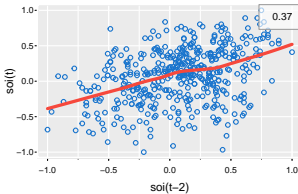
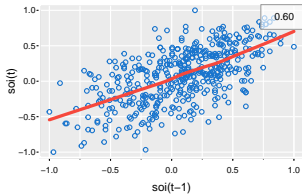
Example using the `soi` data set (sea level air pressure index) from the `astsa` library:

```
library(astsa)

# Compute and plot the ACF for different lags
acfl(soi, co=3, lwd=2)

# Scatterplot of original versus or lags up to 6, with ACF
lag1.plot(soi, max.lag = 6, col=4, lwl=3)
```

Autocorrelation Function (ACF) Example 2 [cont'd]



Dealing with Non-Stationarity

Log Transformation

$$y_t = \log x_t$$

Box-Cox power transformation

$$y_t = \begin{cases} (x_t^\lambda - 1)/\lambda & \lambda \neq 0 \\ \log x_t & \lambda = 0 \end{cases}$$

Dealing with Non-Stationarity

Assume that $x_t = \mu_t + y_t$ where μ_t is the trend and y_t a stationary process.

Detrending

- ▶ Estimate trend, e.g. with an LM such as $\mu_t = \beta_0 + \beta_1 t$
- ▶ Work with residuals, e.g. $\hat{y}_t = x_t - \hat{\mu}_t = x_t - \hat{\beta}_0 - \hat{\beta}_1 t$

```
# Simulate a time series with a linear trend
t <- ts(1:100 + rnorm(100) * 10)

# Fit a linear model to the time series
trend_model <- lm(t ~ time(t))
# Calculate detrended series
detrended <- residuals(trend_model)

# Plot original and detrended
par(mfrow=c(2,1))
tsplot(t, type="l", main="original", col=3)
tsplot(detrended, type="l", main="detrend", col=2)
```

Detrending Example



Dealing with Non-Stationarity

Assume that $x_t = \mu_t + y_t$ where μ_t is the trend and y_t a stationary process.

Differencing

- ▶ Model the trend stochastically as a random walk with drift:
 $\mu_t = \delta + \mu_{t-1} + w_t$ where w_t is white noise
- ▶ Differencing then yields

$$\begin{aligned}x_t - x_{t-1} &= (\mu_t + y_t) - (\mu_{t-1} + y_{t-1}) \\&= \mu_t - \mu_{t-1} + y_t - y_{t-1} \\&= \delta + w_t + y_t - y_{t-1}\end{aligned}$$

which is stationary

- ▶ First difference eliminates linear trend
- ▶ Second difference eliminates quadratic trend
- ▶ ...

Differencing in R

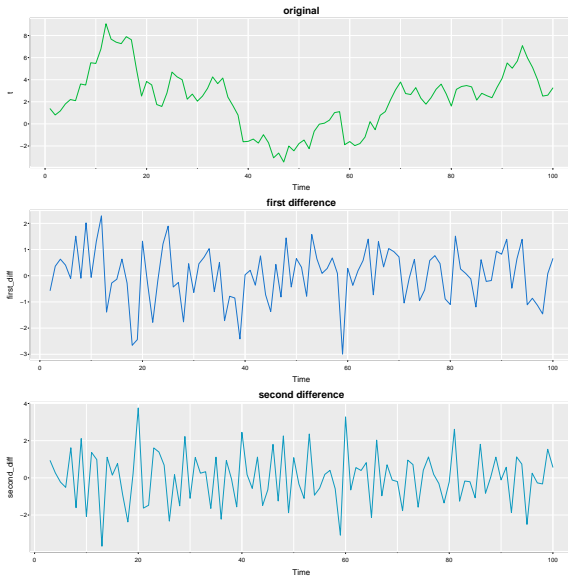
```
set.seed(42)
# Simulating a time series with trend
t <- ts(cumsum(rnorm(100)))

# Plot original, first, and second difference
par(mfrow=c(3,1))
tsplot(t, type="l",
       main="original", col=3)

tsplot(diff(t, differences = 1), type="l",
       main="first difference", col=4)

tsplot(diff(t, differences = 2), type="l",
       main="second difference", col=5)
```


Differencing in R



Detrending and Differencing in R – ACF

Compare the ACF of the original, detrended, and differenced series:

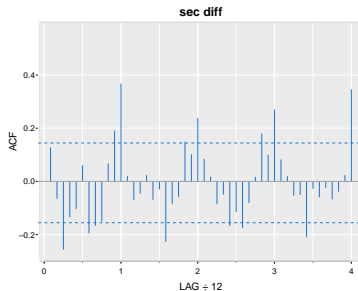
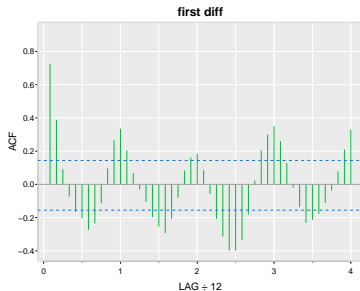
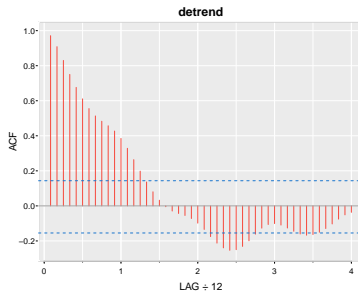
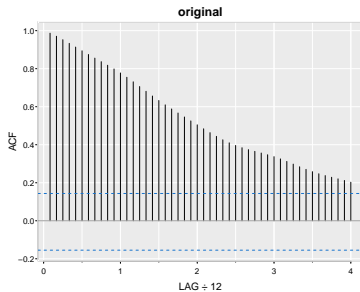
```
acf1(chicken, max.lag=48, main="original", col=1)

acf1(resid(fit), max.lag=48, main="detrend", col=2)

acf1(diff(chicken, differences = 1), max.lag=48,
      main="first diff", col=3)

acf1(diff(chicken, differences = 2), max.lag=48,
      main="sec diff", col=4)
```

Detrending and Differencing in R



Hands-On Exercises

- 1 Extend the mortality, temperature and pollution/particulate model by adding another component to the regression that accounts to the particulate four weeks prior; that is, add P_{t-4} to the regression.
- 2 Draw a scatterplot matrix of M_t , T_t , P_t and P_{t-4} , then calculate the pairwise correlations between them. Compare the relationship between M_t and P_t versus M_t and P_{t-4}
- 3 Detrend the `soi` time series data by fitting a regression of S_t on time t . Is there a significant trend in the surface pressure?
- 4 Use two different smoothing techniques to estimate the trend in the global temperature series `gtemp_both` in the `astsa` library.

Consider the two weekly time series `oil` and `gas` in the `astsa` library. The oil series is in dollars per barrel, while the gas series is in cents per gallon.

- 1 Plot the data on the same graph. Do you believe the series are stationary?
- 2 Apply the transformation $y_t = \nabla \log x_t$ to the data for both series
- 3 Plot the transformed series on the same graph, and calculate the ACFs for both series
- 4 Plot the CCF of the transformed series and comment.

- ▶ **AR:** Autoregressive models
- ▶ **MA:** Moving average models
- ▶ **ARMA:** Autoregressive moving-average models
- ▶ **ARIMA:** Autoregressive integrated moving-average models
(for non-stationary models with trend)

Backshift / Lag Operator

$$B x_t = x_{t-1}$$

$$B^k x_t = x_{t-k}$$

AR(p) Models

An **autoregressive model** of order p is of the form:

$$x_t = \phi_1 x_{t-1} + \phi_2 x_{t-2} + \cdots + \phi_p x_{t-p} + w_t$$

where w_t is white noise and ϕ_i are model parameters.

The **autoregressive operator** $\phi(B)$ is defined using the backshift operator as:

$$\begin{aligned}\phi(B) &= 1 - \phi_1 B - \phi_2 B^2 - \cdots - \phi_p B^p \\ &= \left(1 - \sum_{j=1}^p \phi_j B^j \right)\end{aligned}$$

so that the AR(p) model becomes:

$$\phi(B)x_t = w_t$$

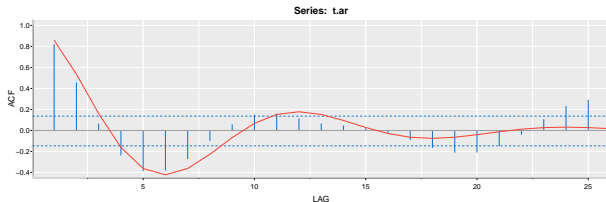
AR(p) Models – Theoretical ACF and Simulations

Theoretical ACF of an AR(2) model

```
ARMAacf(ar=c(1.5, -.75), lag.max=10)
```

Simulate an ARIMA(2,0,0) model with those AR coefficients

```
t.ar = arima.sim(list(ar=c(1.5, -.75)), n=200)
# Compute and plot the ACF of the simulated series
acf1(t.ar, max.lag=25, gg=T, lwd=2, col=4)
# Add the theoretical values for comparison
lines(ARMAacf(ar=c(1.5, -.75), lag.max=26)[-1], lwd=2, col=2)
```



Note the slow tailing off of the ACF

MA(p) Models

A **moving average model** of order q is defined as:

$$x_t = w_t + \theta_1 w_{t-1} + \theta_2 w_{t-2} + \cdots + \theta_q w_{t-q}$$

where w_t are Gaussian errors and θ_i are model parameters.

The **moving average operator** $\theta(B)$ is defined using the backshift operator as

$$\begin{aligned}\theta(B) &= 1 + \theta_1 B + \theta_2 B^2 + \cdots + \theta_q B^q \\ &= \left(1 + \sum_{j=1}^q \theta_j B^j \right)\end{aligned}$$

so that the MA(q) model becomes:

$$x_t = \theta(B)w_t$$

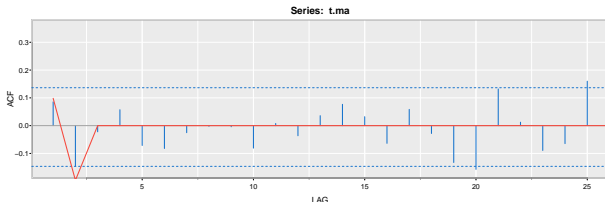
MA(p) Models – Theoretical ACF and Simulations

Theoretical ACF of an MA(2) model

```
ARMAacf(ma=c(1.5, -.75), lag.max=10)
```

Simulate an ARIMA(0,0,2) model with those MA coefficients

```
t.ma = arima.sim(list(ma=c(1.5, -.75)), n=200)
# Compute and plot the ACF of the simulated series
acf1(t.ma, gg=T, lwd=2, col=4)
# Add the theoretical values for comparison
lines(ARMAacf(ma=c(1.5, -.75), lag.max=26)[-1], lwd=2, col=2)
```



Note the sharp cut off of the ACF

ARMA(p, q) Models

A time series is **ARMA(p,q)** if it is stationary and

$$x_t = \alpha + \phi_1 x_{t-1} + \cdots + \phi_p x_{t-p} + w_t + \theta_1 w_{t-1} + \cdots + \theta_q w_{t-q}$$

where $\phi_p \neq 0, \theta_q \neq 0, \alpha = \mu(1 - \phi_1 - \cdots - \phi_p)$ and w_t is Gaussian.

This can be written as

$$\phi(B)x_t = \theta(B)w_t$$

Partial Autocorrelation Function (PACF)

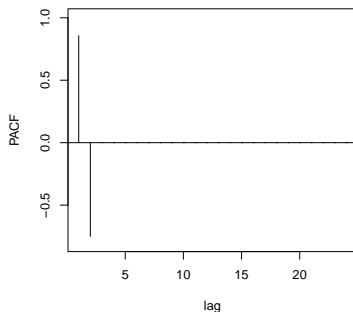
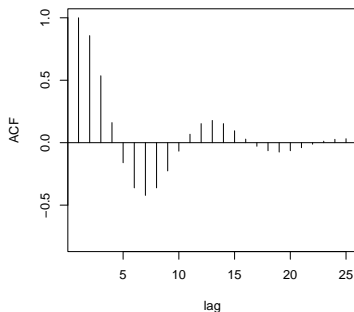
The PACF is the correlation between x_{t+h} and x_t with the linear dependence of $\{x_{t+1}, \dots, x_{t+h-1}\}$ on each, removed:

$$\phi_{hh} = \begin{cases} \rho(1) & h = 1 \\ \text{corr}(x_{t+h} - \hat{x}_{t+h}, x_t - \hat{x}_t) & h \geq 2 \end{cases}$$

Theoretical Partial Autocorrelation Function (PACF)

```
ACF = ARMAacf(ar=c(1.5,-.75), ma=0, 24)
PACF = ARMAacf(ar=c(1.5,-.75), ma=0, 24, pacf=TRUE)

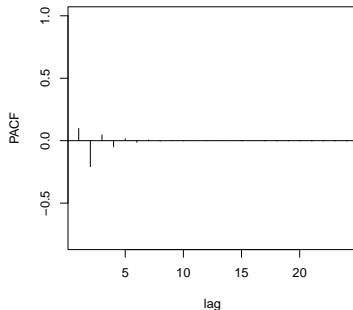
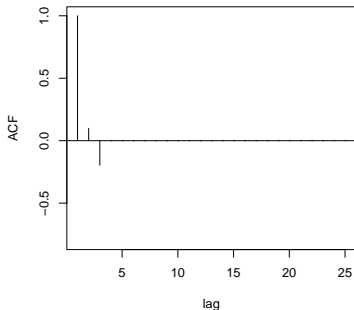
# Plot the ACF
plot(ACF, type="h", xlab="lag", ylim=c(-.8,1))
abline(h=0)
# Plot the PACF
plot(PACF, type="h", xlab="lag", ylim=c(-.8,1))
abline(h=0)
```



Theoretical Partial Autocorrelation Function (PACF)

```
ACF = ARMAacf(ar=0, ma=c(1.5,-.75), 24)
PACF = ARMAacf(ar=0, ma=c(1.5,-.75), 24, pacf=TRUE)

# Plot the ACF
plot(ACF, type="h", xlab="lag", ylim=c(-.8,1))
abline(h=0)
# Plot the PACF
plot(PACF, type="h", xlab="lag", ylim=c(-.8,1))
abline(h=0)
```



ARMA(p , q) Models – Model Selection

	AR(p)	MA(q)	ARMA (p , q)
ACF	Tails off	Cuts off after lag q	Tails off
PACF	Cuts off after lag p	Tails off	Tails off

Source: Shumway & Stoffer, Table 3.1

ARIMA(p, d, q) Models

If the AR operator can be factorized by $(1 - B)$, then:

$$\phi(B) = \left(1 - \sum_{j=1}^{p'} \phi_j B^j\right) = \left(1 - \sum_{j=1}^{p'-d} \phi_j B^j\right) (1 - B)^d$$

With $p = p' - d$, the **ARIMA(p,d,q)** model is then:

$$\left(1 - \sum_{j=1}^p \phi_j B^j\right) (1 - B)^d x_t = \left(1 + \sum_{j=1}^q \theta_j B^j\right) w_t$$

Here $(1 - B)$ is the **differencing operator** ∇ . For example:

$$(1 - B)x_t = x_t - x_{t-1} = \nabla x_t$$

$$(1 - B)^2 x_t = (1 - 2B + B^2)x_t = x_t - 2x_{t-1} + x_{t-2} = \nabla^2 x_t$$

...

Building ARIMA Models

- 1 Plot the data
- 2 Possibly transform the data
- 3 Assess stationarity
- 4 Possibly difference the data
- 5 Identify the dependence orders (p, q) of the model
- 6 Estimate parameters
- 7 Model diagnostics
- 8 Model selection

Identifying Dependence Order

Differencing d

- ▶ Slow decay in sample ACF $\hat{\rho}(h)$ indicates need for differencing
- ▶ Over-differencing can introduce dependence where non exists
- ▶ Difference once using `diff(x, differences = 1)`, then check ACF again

Preliminary p and q

- ▶ Examine sample ACF and PACF of differenced data

	AR(p)	MA(q)	ARMA (p, q)
ACF	Tails off	Cuts off after lag q	Tails off
PACF	Cuts off after lag p	Tails off	Tails off

Source: Shumway & Stoffer, Table 3.1

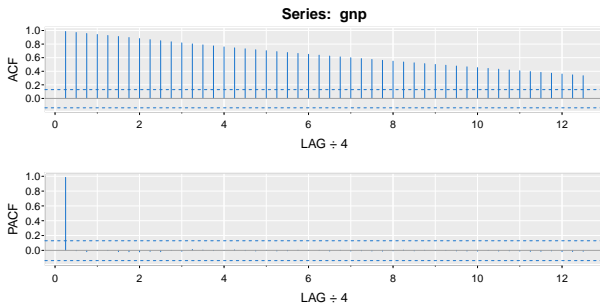
Example

Examine data and transform:

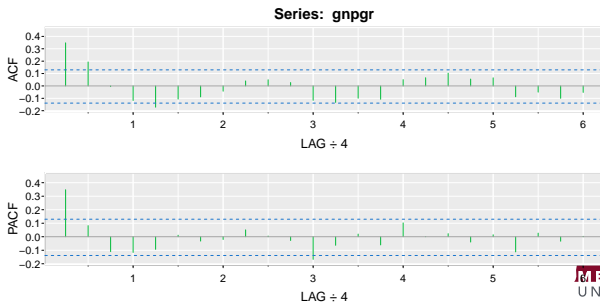
```
# Plot data  
plot(gnp)  
# Plot ACF  
acf2(gnp, 50)  
# Log transform, and first order differencing  
gnpgr = diff(log(gnp))  
# Plot transformed and differenced data  
plot(gnpgr)  
# Plot ACF of transformed and differenced data  
acf2(gnpgr, 24)
```

Example [cont'd]

Original
ACF and
PACF



Transformed
and differ-
enced
ACF and
PACF



Example [cont'd]

Fit initial models

```
# Fit an AR(1) model
sarima(gnpgr, 1, 0, 0)

# Fit an MA(2) model
sarima(gnpgr, 0, 0, 2)
```

The models show similar fit and all coefficients are significant:

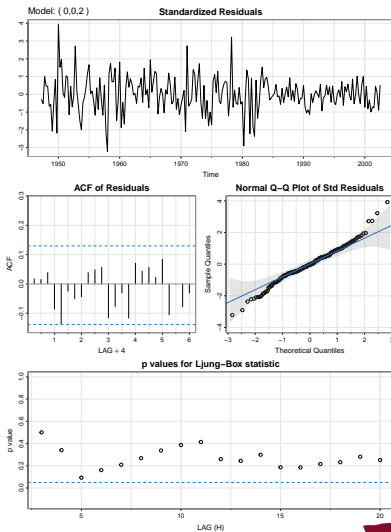
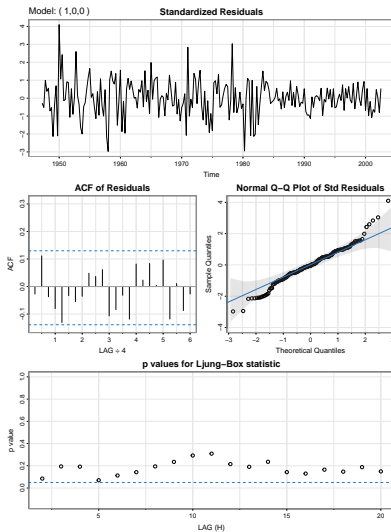
	Estimate	SE	t.value	p.value
ar1	0.3467	0.0627	5.5255	0
xmean	0.0083	0.0010	8.5398	0

AIC = -6.44694 AICc = -6.446693 BIC = -6.400958

	Estimate	SE	t.value	p.value
ma1	0.3028	0.0654	4.6272	0.0000
ma2	0.2035	0.0644	3.1594	0.0018
xmean	0.0083	0.0010	8.7178	0.0000

AIC = -6.450133 AICc = -6.449637 BIC = -6.388823

Example [cont'd]

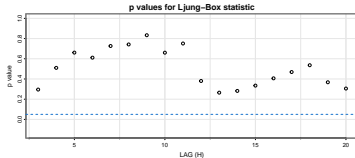
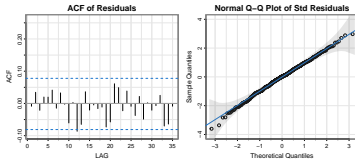
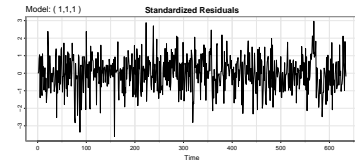
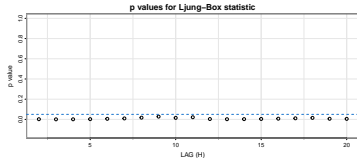
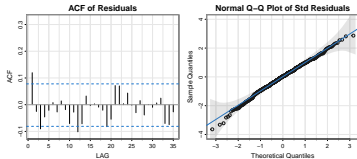
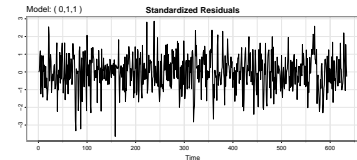


Diagnostics

- ▶ Standardized residuals are normal random ($\mu = 0$, $sd = 1$)
- ▶ Residuals are not autocorrelated
- ▶ Residual ACF are random normal with $\mu = 0$ and $sd = 1/\sqrt{n}$
- ▶ Ljung-Box statistic Q of the error ACF $\hat{\rho}_e$ for different maximum lags H is larger than the $1 - \alpha$ quantile of the χ^2_{H-p-q} distribution (i.e. the test statistic is not statistically significantly different from 0)

Mis-Fit Example

```
sarima(log(varve), 0, 1, 1, no.constant=T)  
sarima(log(varve), 1, 1, 1, no.constant=T)
```



Model Selection

For MLE estimated models, model choice is typically based on information criteria

- ▶ Functions of the log-likelihood L
- ▶ Adjusted for model complexity (number of parameters) k
- ▶ Adjusted for sample size n
- ▶ Express *relative* quality of fit: **Smaller is better**

$$AIC = -2 \log L + 2k$$

Akaike Information Criterion

$$AIC_c = AIC + \frac{2k(k+1)}{n-k-1}$$

Akaike Information Criterion, corrected

$$BIC = -2 \log L + k \log n$$

Bayesian Information Criterion

Example [cont'd]

```
> sarima(gnpgr, 1, 0, 0)
```

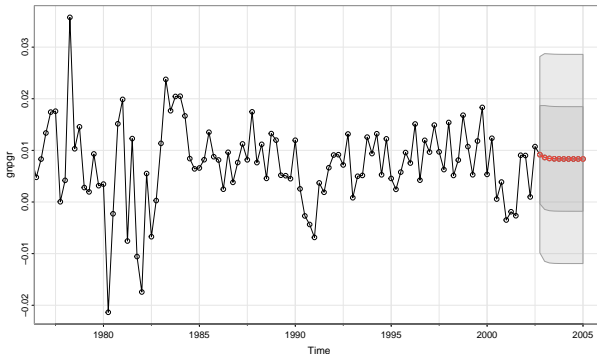
```
AIC = -6.44694   AICc = -6.446693   BIC = -6.400958
```

```
> sarima(gnpgr, 0, 0, 2)
```

```
AIC = -6.450133   AICc = -6.449637   BIC = -6.388823
```

Example – Forecasting

```
forecasts <- sarima.for(gnpgr, n.ahead=10, p=1,d=0,q=0)
```



ARMA predictions quickly settle to the mean with a constant prediction error

General Autoregressive Conditional Heteroscedastic (GARCH) Models

ARCH models the variance of a series of returns:

$$r_t = \frac{x_t - x_{t-1}}{x_{t-1}} \quad (\text{"Return"})$$

Example: ARCH(1) Model – The variance depends on the prior return.

$$\begin{aligned} r_t &= \sigma_t \epsilon_t \\ \sigma_t^2 &= \alpha_0 + \alpha_1 r_{t-1}^2 \end{aligned}$$

where ϵ_t is Gaussian.

Example: Combined AR(1) and ARCH(1) Model

AR(1) Model with ARCH(1) Errors:

$$x_t = \phi_0 + \phi_1 x_{t-1} + \sigma_t \epsilon_t \quad \text{where} \quad \sigma_t = \alpha_0 + \alpha_1 x_{t-1}^2$$

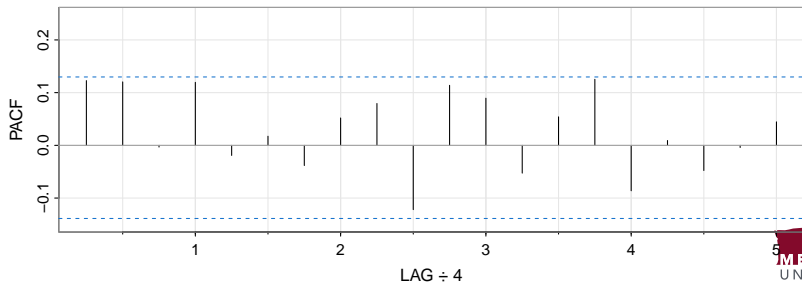
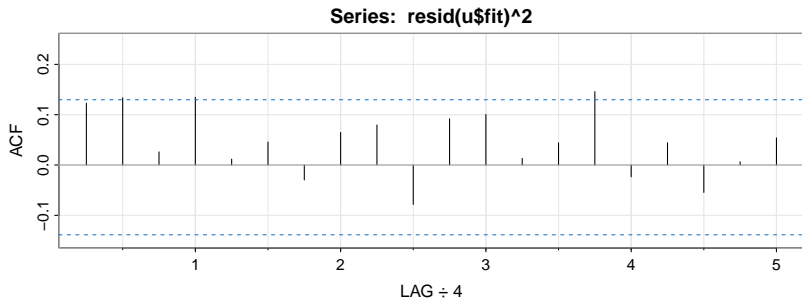
Example: US GNP Data

```
u = sarima(diff(log(gnp)), 1, 0, 0)
acf2(resid(u$fit)^2, 20)
```

Squared residuals may have some dependence.

```
library(fGarch)
summary(garchFit(~arma(1,0)+garch(1,0), diff(log(gnp))))
```

Example: Combined AR(1) and ARCH(1) Model



ARCH(q) Model

Extend the ARCH(1) Model to multiple previous returns:

$$\sigma_t^2 = \alpha_0 + \alpha_1 r_{t-1}^2 + \cdots + \alpha_p r_{t-p}^2 = \alpha_0 + \sum_{i=1}^q \alpha_i r_{t-i}^2$$

GARCH(p, q) Model

Variance depends also on prior variances:

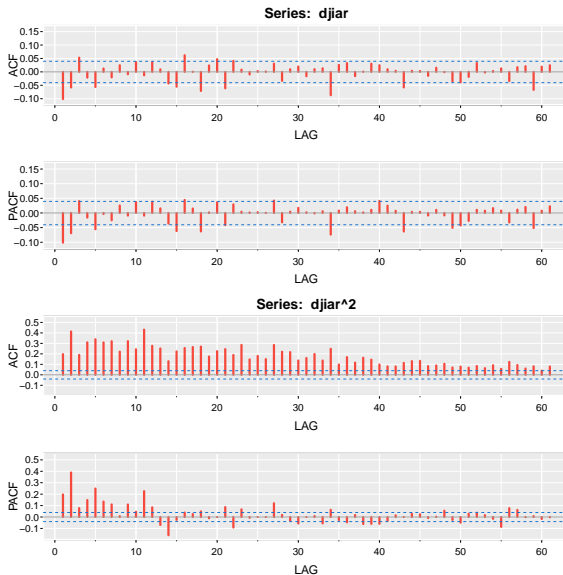
$$\begin{aligned}\sigma_t^2 &= \omega + \alpha_1 r_{t-1}^2 + \cdots + \alpha_q r_{t-q}^2 \\ &\quad + \beta_1 \sigma_{t-1}^2 + \cdots + \beta_p \sigma_{t-p}^2 \\ &= \omega + \sum_{j=1}^q \alpha_j r_{t-j}^2 + \sum_{j=1}^p \beta_j \sigma_{t-j}^2\end{aligned}$$

GARCH Example – DJIA Returns

```
library(fGarch)
# Log transform
djiar = diff(log(djia$Close))[-1]
# Fit an AR(1) + GARCH(1,1) model
djiar.g <- garchFit(~arma(1,0)+garch(1,1), data=djiar)
# Show summary information
summary(djiar.g)
# Different plots available
plot(djiar.g, which=3)
```

ARCH Example – DJIA Returns [cont'd]

ACF



GARCH Example – DJIA Returns [cont'd]

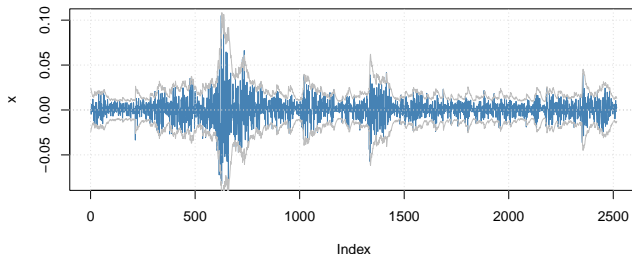
Results

	Estimate	Std. Error	t value	Pr(> t)	
mu	8.585e-04	1.470e-04	5.842	5.16e-09	***
arl	-5.532e-02	2.023e-02	-2.735	0.006238	**
omega	1.610e-06	4.459e-07	3.611	0.000305	***
alpha1	1.244e-01	1.660e-02	7.496	6.55e-14	***
beta1	8.700e-01	1.526e-02	57.022	< 2e-16	***
shape	5.979e+00	7.917e-01	7.551	4.31e-14	***

Log Likelihood:

8249.619 normalized: 3.27756

Series with 2 Conditional SD Superimposed



Appendix – Basic Time Series Functions in R

<code>filter</code>	Filters time series, through moving averages or autoregression
<code>lag</code>	Creates a lagged version of a time series by shifting the time-base back
<code>diff</code>	Creates lagged differences
<code>plot.ts</code>	Plot a time series
<code>ts.plot</code>	Plot multiple time series
<code>lag.plot</code>	Scatterplot of lagged values
<code>acf</code>	ACF and plot
<code>ccf</code>	CCF and plot

Appendix – Basic Time Series Functions in R

<code>time</code>	Creates the vector or times at which a time series was sampled
<code>cycle</code>	Gives the positions in the cycle of each observation
<code>frequency</code>	Number of samples per unit time
<code>ts.intersect</code>	Bind time series together that have a common frequency. Restrict to time covered by all series
<code>ts.union</code>	Bind time series together that have a common frequency. Pad with NA if necessary
<code>ar</code>	Fit an autoregressive model
<code>arima</code>	Fit an ARIMA model

Appendix – Time Series Functions in the `astsa` library

<code>tsplot</code>	Plot a time series
<code>acf1</code>	ACF and plot
<code>ccf2</code>	CCF and plot
<code>sarima</code>	Fit seasonal ARIMA models (and nice diagnostic plots)
<code>lag1.plot</code>	Scatterplot of lagged values