Time Series Analysis and Forecasting

Math 667

Al Nosedal

Department of Mathematics

Indiana University of Pennsylvania

Introduction

Many decision-making applications depend on a forecast of some quantity. Here are a couple of examples.

- When a service organization, such as a fast-food restaurant, plans its staffing over some time period, it must forecast the customer demand as a function of time.

Introduction

Many decision-making applications depend on a forecast of some quantity. Here are a couple of examples.

- When a service organization, such as a fast-food restaurant, plans its staffing over some time period, it must forecast the customer demand as a function of time.
- When a company plans its ordering or production schedule for a product it sells to the public, it must forecast the customer demand for this product so that it can stock appropriate quantities-neither too much nor too little.

Forecasting Methods: An Overview

There are many forecasting methods available, these methods can generally be divided into three groups:

- 1. Judgemental methods.

Forecasting Methods: An Overview

There are many forecasting methods available, these methods can generally be divided into three groups:

- 1. Judgemental methods.
- 2. Extrapolation (or Time Series) methods, and

Forecasting Methods: An Overview

There are many forecasting methods available, these methods can generally be divided into three groups:

- 1. Judgemental methods.
- 2. Extrapolation (or Time Series) methods, and
- 3. Econometric (or causal) methods.

Extrapolation Methods

Extrapolation methods are quantitative methods that use past data of a time series variable-and nothing else, except possibly time itself-to forecast future values of the variable. The idea is that we can use past movements of a variable, such as a company sales to forecast its future values.

Extrapolation Methods

There are many extrapolation methods available, including trend-based regression, moving averages, and autoregression models. All these extrapolation methods search for patterns in the historical series and then extrapolate these patterns into the future.

IUP

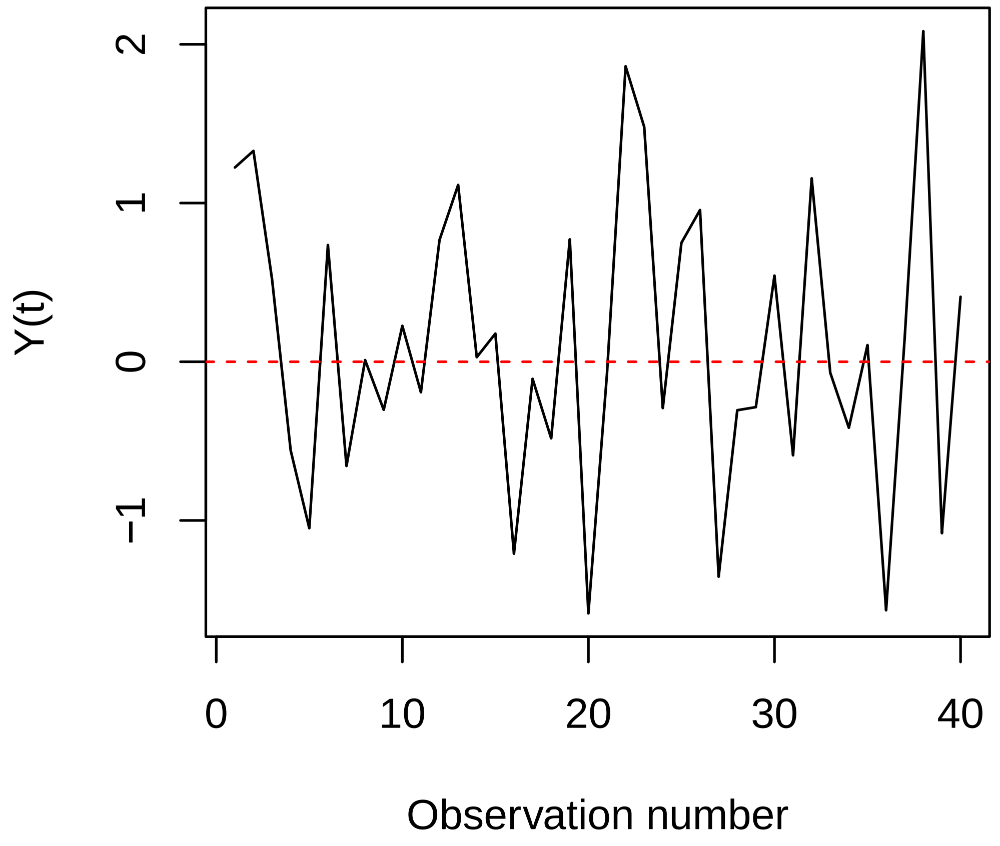
Random Series

- The simplest time series model is the random model. A random model can be written as

$$Y(t) = \mu + \epsilon(t) \tag{1}$$

- Here, μ is a constant, the average of the $Y(t)'s$, and $\epsilon(t)$ is the residual (or error) term. We assume that the residuals have mean 0, variance σ^2, and are probabilistically independent of one another.

Random series (Graph)



Random Series (1)

There are two situations where random time series occur.

- The first is when the original time series is random. For example, when studying the time pattern of diameters of individual parts from a manufacturing process, we might discover that the successive diameters behave like a random time series.

IUP

Random Series (1)

There are two situations where random time series occur.

- The first is when the original time series is random. For example, when studying the time pattern of diameters of individual parts from a manufacturing process, we might discover that the successive diameters behave like a random time series.

Random Series (1)

There are two situations where random time series occur.

- The first is when the original time series is random. For example, when studying the time pattern of diameters of individual parts from a manufacturing process, we might discover that the successive diameters behave like a random time series.

Random Series (2)

- The second situation where a random series occurs is more common. This is when we fit a model to a time series to obtain an equation of the form

$$Y(t) = fitted\,part + residual. \tag{2}$$

Although the fitted part varies from model to model, its essential feature is that it describes any underlying time series pattern in the original data. The residual is then whatever is left, and we hope that the series of residuals is random with mean $\mu = 0$. The fitted part is used to make forecasts, and the residuals, often called noise, are forecasted to be zero.

Random Series (2)

- The second situation where a random series occurs is more common. This is when we fit a model to a time series to obtain an equation of the form

$$Y(t) = fitted\,part + residual. \tag{3}$$

Although the fitted part varies from model to model, its essential feature is that it describes any underlying time series pattern in the original data. The residual is then whatever is left, and we hope that the series of residuals is random with mean $\mu = 0$. The fitted part is used to make forecasts, and the residuals, often called noise, are forecasted to be zero.

Runs test

For each observation $Y(t)$ we associate a 1 if $Y(t) \geq \bar{y}$ and 0 if $Y(t) < \bar{y}$. A run is a consecutive sequence of 0's or 1's. Let T be the number of observations, let T_a above the mean, and let T_b the number below the mean. Also let R be the observed number of runs. Then it can be show that for a random series

$$E(R) = \frac{T + 2T_a T_b}{T} \tag{4}$$

$$Stdev(R) = \sqrt{\frac{2T_a T_b(2T_a T_b - T)}{T^2(T-1)}} \tag{5}$$

When $T > 20$, the distribution of R is roughly Normal.

Example: Runs test

Suppose that the successive observations are 87, 69, 53, 57, 94, 81, 44, 68, and 77, with mean $\bar{Y} = 70$. It is possible that this series is random. Does the runs test support this conjecture?

Example: Runs test (solution)

The preceding sequence has five runs: 1; 0 0 0; 1 1; 0 0; and 1. Then, we have $T = 9, T_a = 4, T_b = 5$ and $R = 5$. Under a randomness hypothesis,

$$E(R) = 5.44 \tag{6}$$

$$Stdev(R) = 1.38 \tag{7}$$

$$Z = \frac{R - E(R)}{Stdev(R)} = -0.32 \tag{8}$$

Exercise

Write a function in R that performs a run test. My version of this function will be posted on our website tomorrow.

Autocorrelation (1)

Recall that the successive observations in a random series are probabilistically independent of one another. Many time series violate this property and are instead autocorrelated. For example, in the most common form of autocorrelation, positive autocorrelation, large observations tend to follow large observations, and small observations tend to follow small observations. In this case the runs test is likely to pick it up. Another way to check for the same nonrandomness property is to calculate the autocorrelations of the time series.

IUP

Autocorrelation (2)

To understand autocorrelation it is first necessary to understand what it means to lag a time series.
$Y(t) = Y_t =$(3,4,5,6,7,8,9,10,11,12)
Y lagged 1 $= Y(t-1) = Y_{t-1} =$(*,3,4,5,6,7,8,9,10,11)

Autocovariance Function

The autocovariance function is defined as follows:

(9) $$\gamma(k) = E(Y(t) - \mu)(Y(t+k) - \mu),\ k = 0, \pm 1, \pm 2, \ldots$$

where $Y(t)$ represents the values of the series, μ is the mean of the series and k is the lag at which the autocovariance is being considered.

Autocorrelation Function

A standardized version of $\gamma(k)$ is the autocorrelation function, defined as

$$\rho = \frac{\gamma(k)}{\gamma(0)} \tag{10}$$

Estimator of the autocovariance function

The autocovariance function at lag k can be estimated by

$$\hat{\gamma}(k) = \frac{1}{n} \sum_{t=1}^{n-k} (Y(t) - \bar{Y})(Y(t+k) - \bar{Y}) \tag{11}$$

Estimator of the autocorrelation function

The autocorrelation function estimate at lag k is simply

$$\hat{\rho}(k) = \frac{\hat{\gamma}(k)}{\hat{\gamma}(0)} \tag{12}$$

Example

Find the autocorrelation at lag 1 for the following time series:

$Y(t) = (3, 4, 5, 6, 7, 8, 9, 10, 11, 12)$

Answer.

$\hat{\rho}(1) = 0.7$

Finding the autocorrelation at lag 1

```
y<-3:12

### function to find autocorrelation when
###lag=1
my.auto<-function(x){
    n<-length(x)
    denom<-(x-mean(x))%*%(x-mean(x))/n
    num<-(x[-n]-mean(x))%*%(x[-1]-mean(x))/n
    result<-num/denom

    return(result)
            }
my.auto(y)
```

An easier way of finding autocorrelations

Using R, we can easily find the autocorrelation at any lag k.

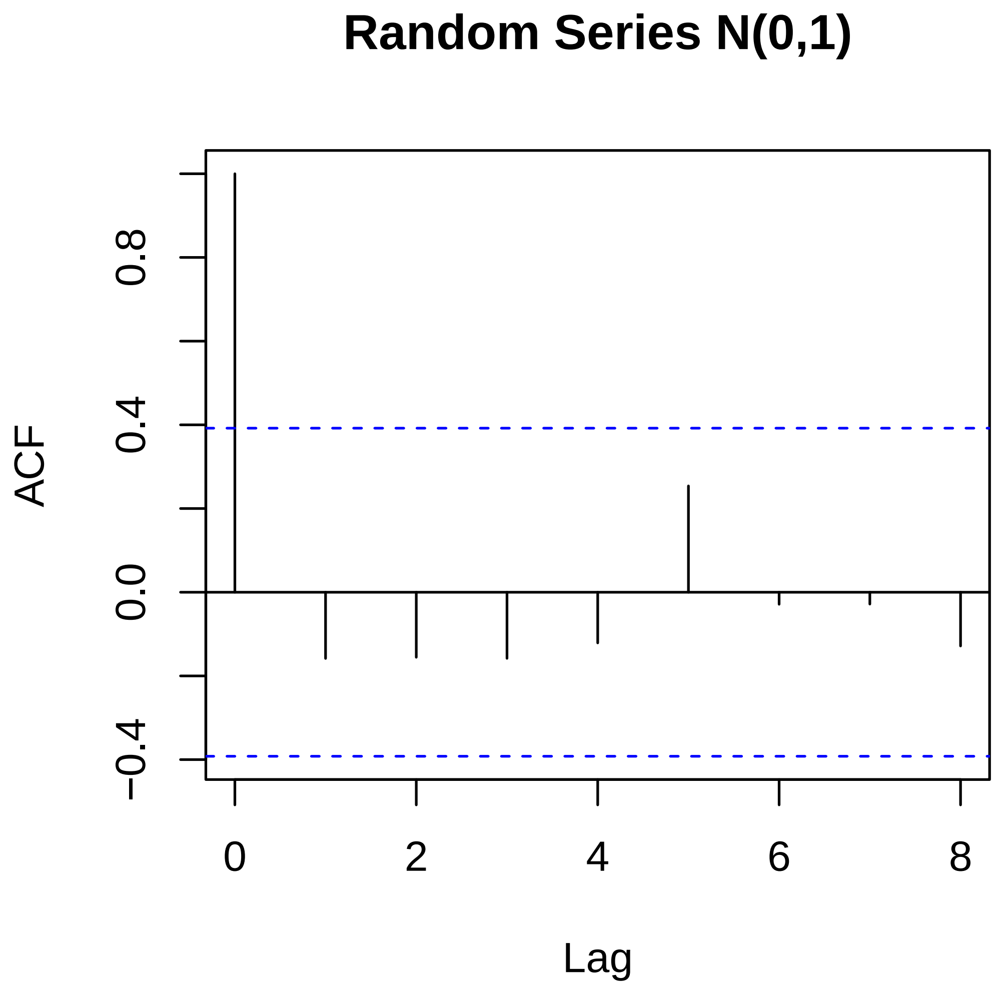
```
y<-3:12

auto.lag1<-acf(y,lag=1)$acf[2]

auto.lag1
```

Graph of autocorrelations

A graph of the lags against the corresponding autocorrelations is called a correlogram. The following lines of code can be used to make a correlogram in R.

```
### autocorrelation function for
###a random series

y<-rnorm(25)

acf(y,lag=8,main='Random Series N(0,1)')
```

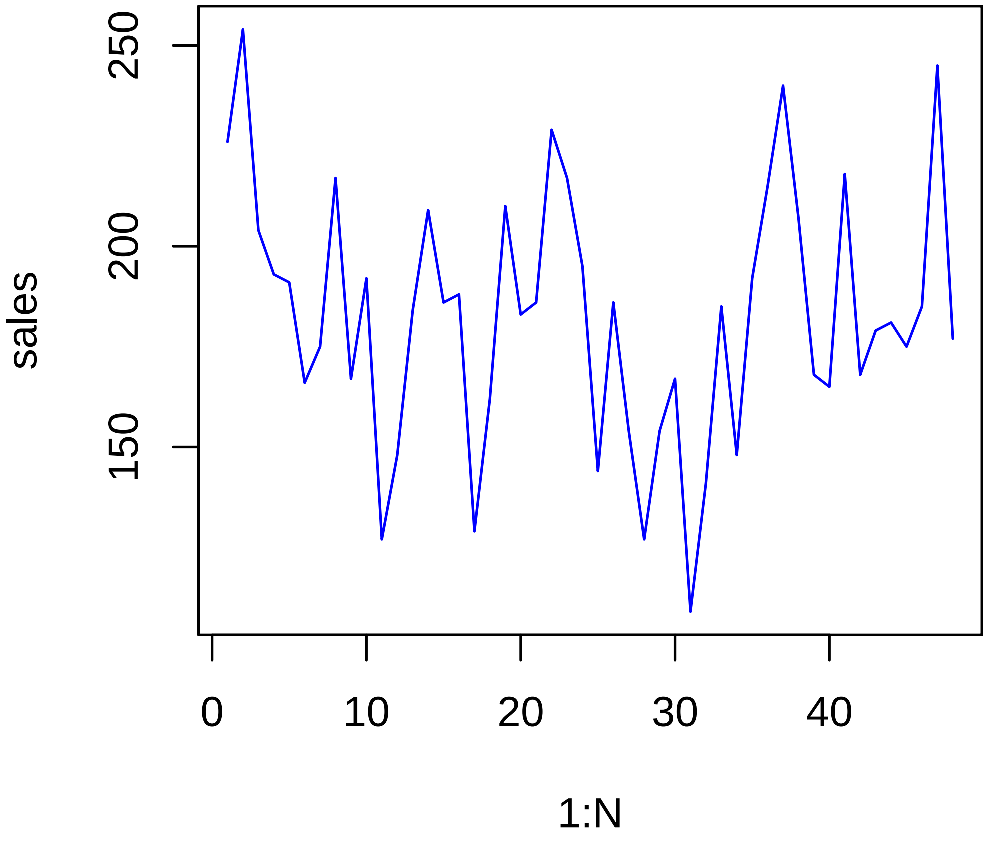
Correlogram (Graph)



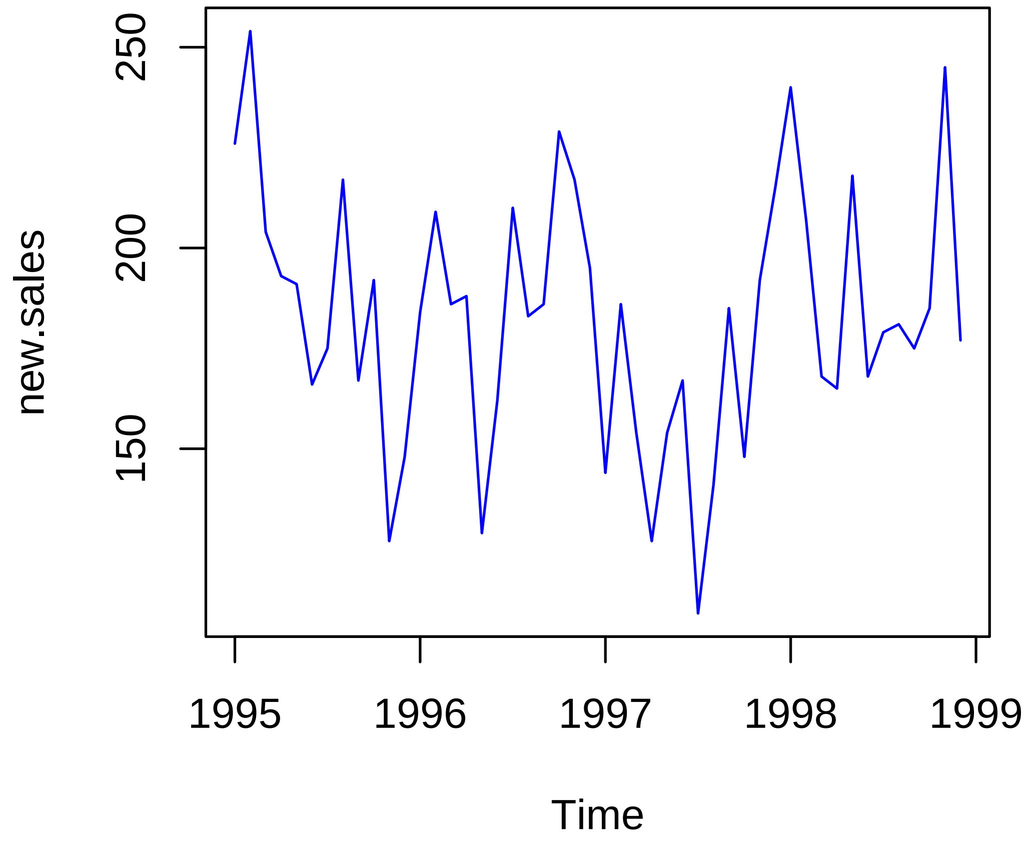
Example: Stereo Retailers

Monthly sales for a chain of stereo retailers are listed in the file stereo.txt. They cover the period from the beginning of 1995 to the end of 1998, during which there was no upward or downward trend in sales and no clear seasonal peaks or valleys. This behavior is apparent in the time series chart of sales shown in our next slide. It is possible that this series is random. Do autocorrelations support this conclusion?

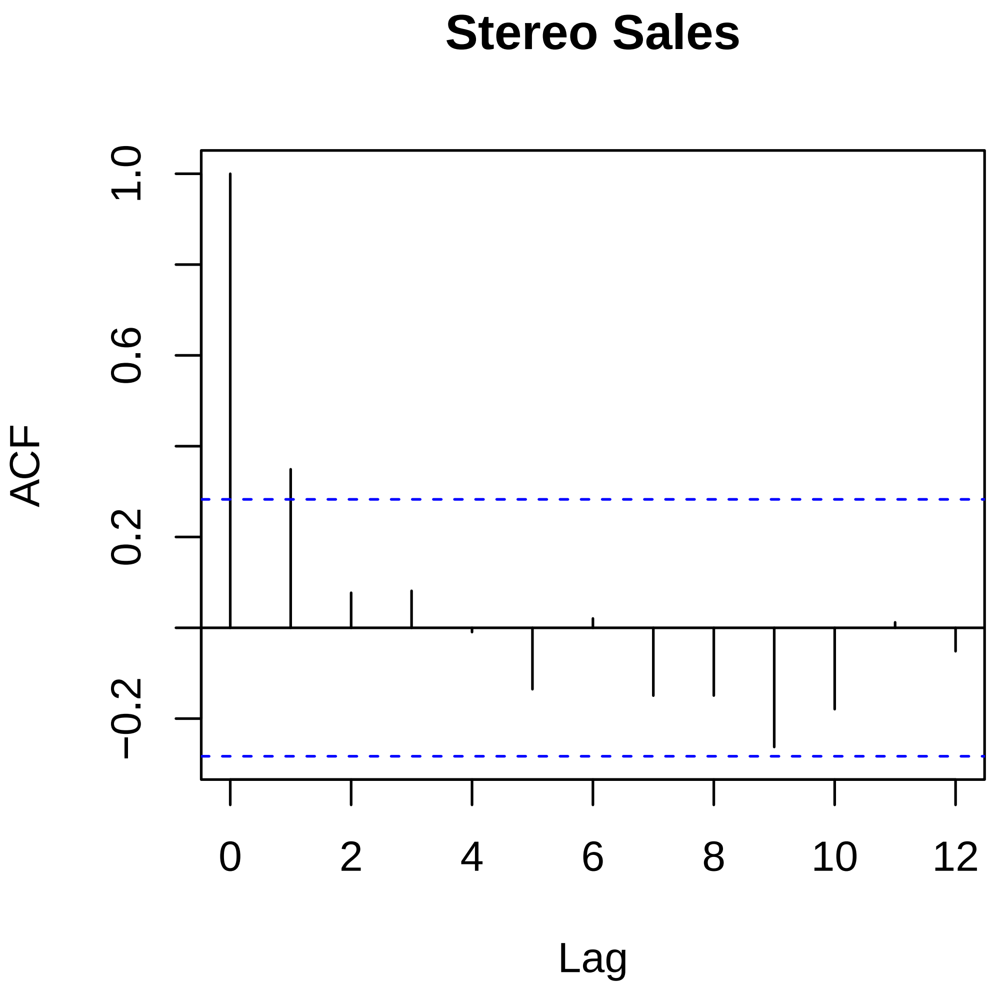
Time Series Plot (stereo.txt)



Time Series Plot (stereo.txt)



Correlogram (stereo.txt)



How large is a "large" autocorrelation?

Suppose that $Y_1, ..., Y_T$ are independent and identically distributed random variables with arbitrary mean. Then it can be shown that

(13) $$E(\hat{\rho}_k) \simeq \frac{-1}{T}$$

(14) $$Var(\hat{\rho}_k) \simeq \frac{1}{T}$$

Thus, having plotted the correlogram, we can plot approximate 95% confidence limits at $-1/T \pm 2/\sqrt{T}$, which are often further approximated to $\pm 2/\sqrt{T}$.

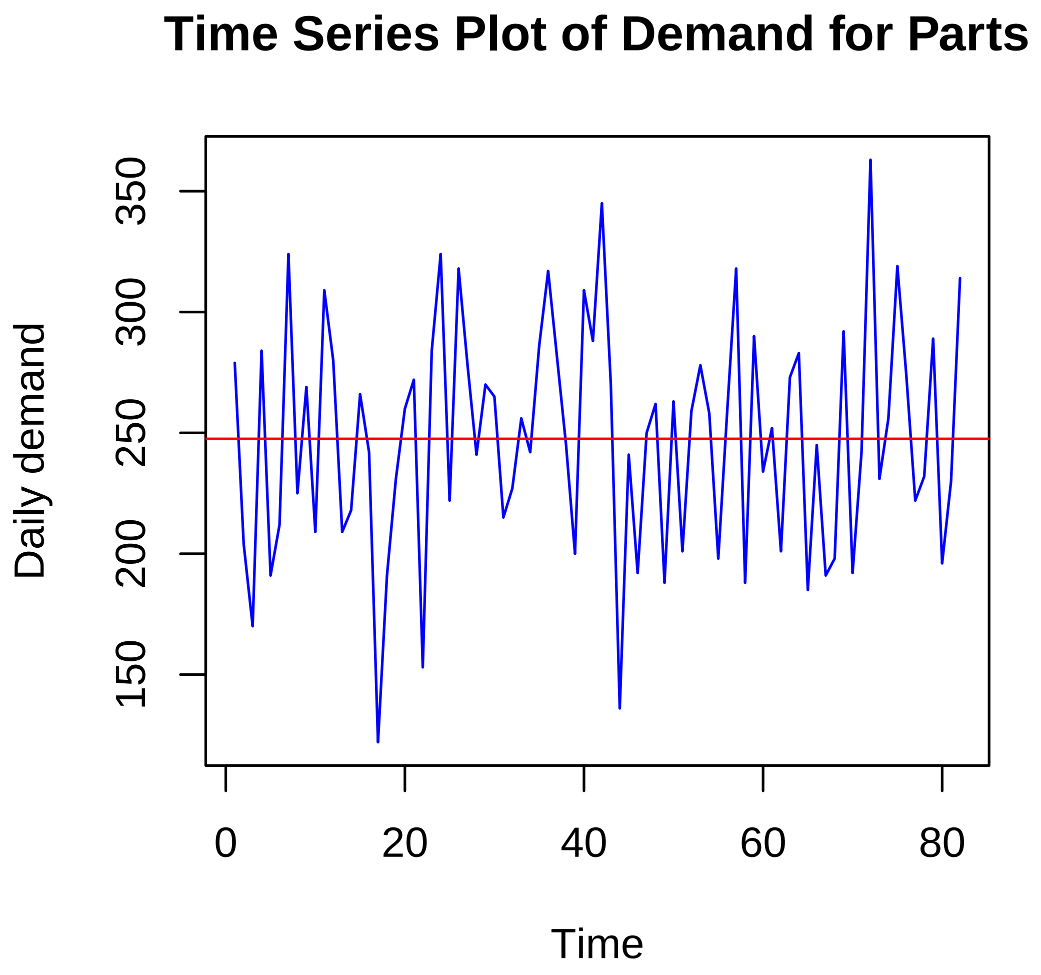
Conclusion (Stereo Sales)

In this case, $\sqrt{V(\hat{\rho}_k)} = 1/\sqrt{48}$. If the series is truly random, then only an occasional autocorrelation should be larger than two standard errors in magnitude. Therefore, any autocorrelation that is larger than two standard errors in magnitude is worth our attention. The only "large" autocorrelation for the sales data is the first, or lag 1. It seems that the pattern of sales is not completely random.

Example: Demand

The dollar demand for a certain class of parts at a local retail store has been recorded for 82 consecutive days. (See the file demand.txt) A time series plot of these demands appears in the next slide. The store manager wants to forecast future demands. In particular, he wants to know whether is any significant time pattern to the historical demands or whether the series is essentially random.

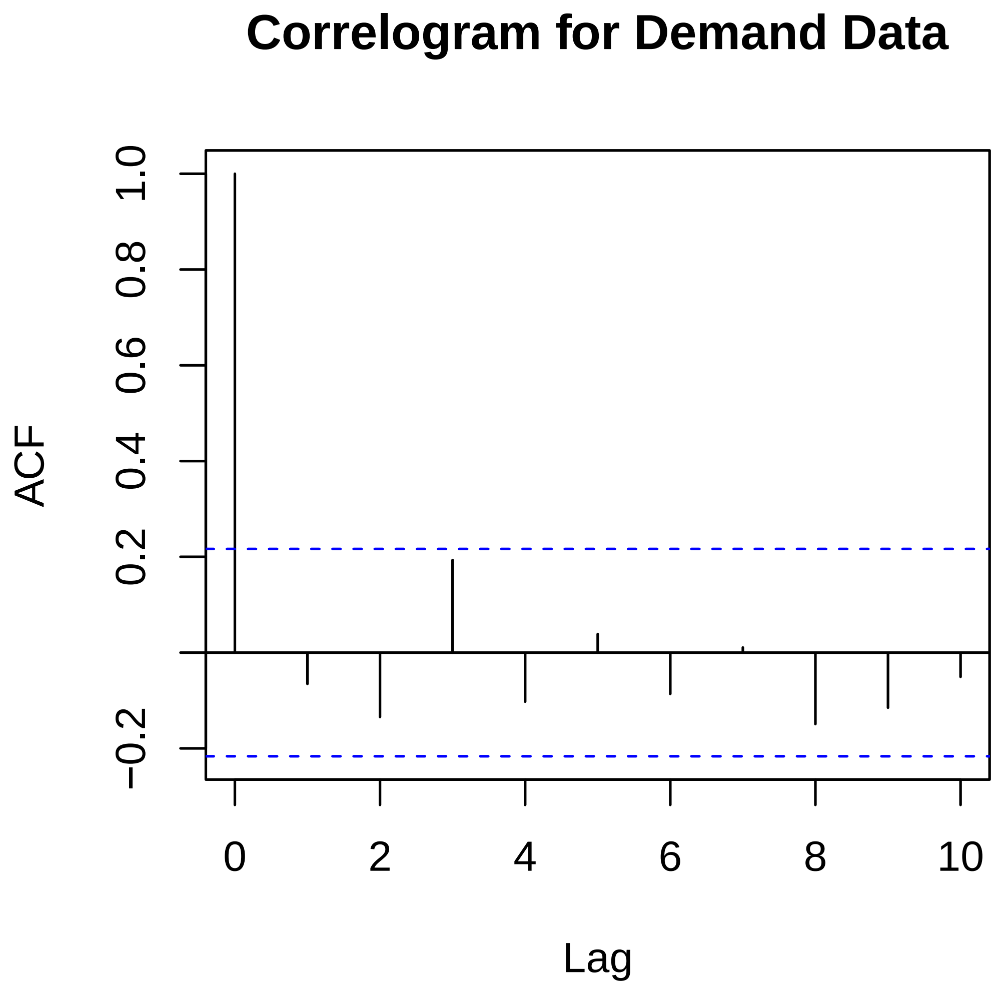
Time Series Plot of Demand for Parts



Solution

A visual inspection of the time series graph in the previous slide shows that demands vary randomly around the sample mean of $247.54 (shown as the horizontal centerline). The variance appears to be constant through time, and there are no obvious time series patterns. To check formally whether this apparent randomness holds, we perform the runs test and calculate the first 10 autocorrelations. (The associated correlogram appears in the next slide). The p-value for the runs test is relatively large 0.118 and none of the autocorrelations is significantly large. These findings are consistent with randomness.

Graph



Conclusion

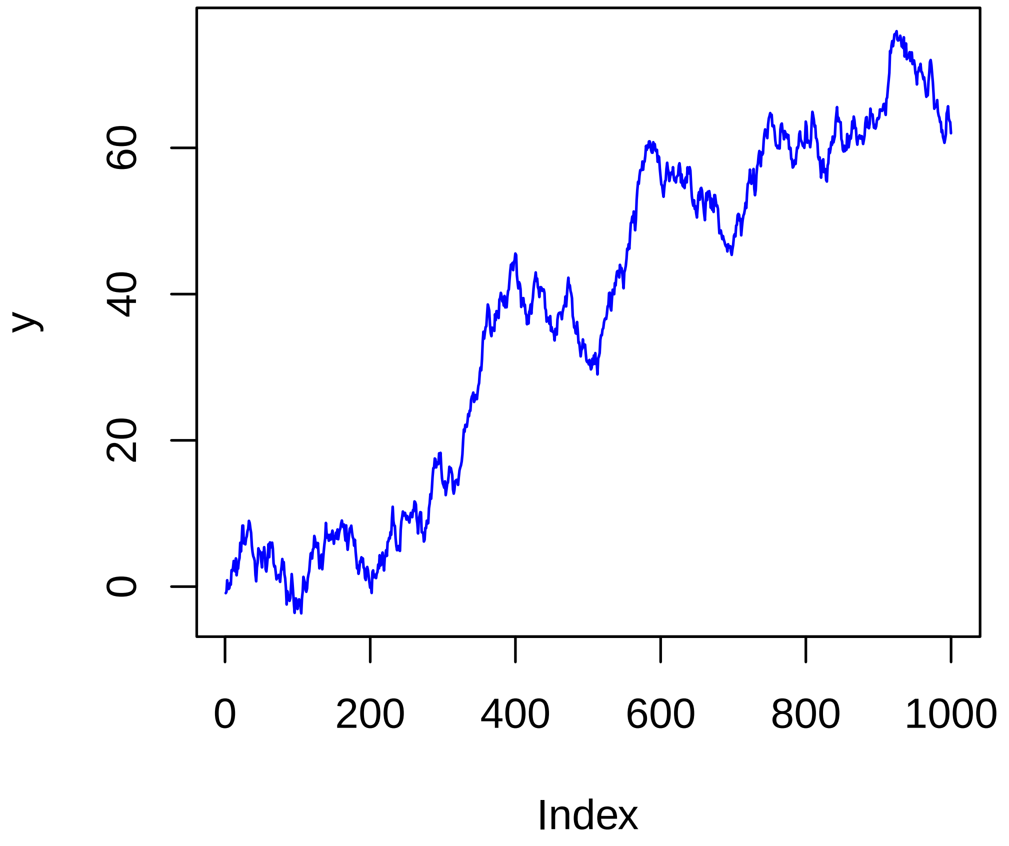
For all practical purposes there is no time series pattern to these demand data. It is as if every day’s demand is an independent draw from a distribution with mean \$247.54 and standard deviation \$47.78. Therefore, the manager might as well forecast that demand for any day in the future will be \$247.54. If he does so, about 95% of his forecasts should be within two standard deviations (about \$95) of the actual demands.

A simulation

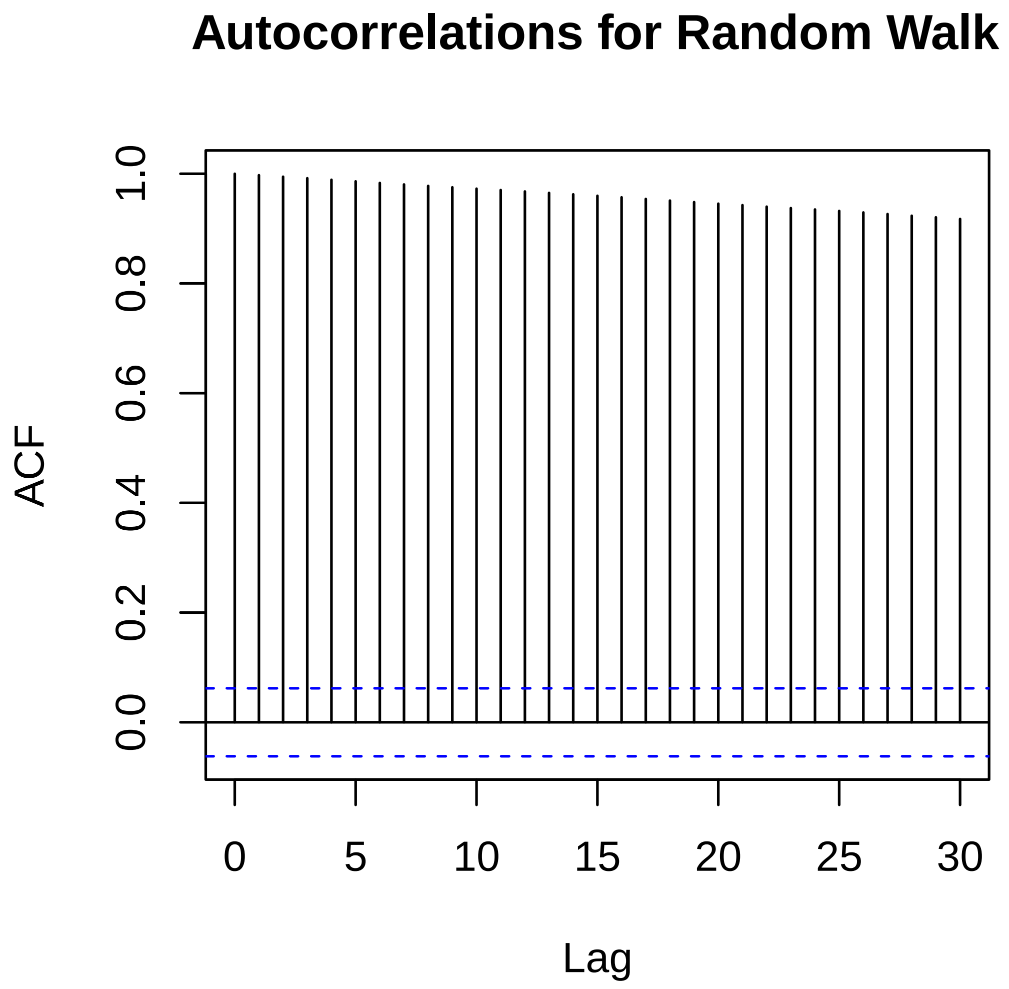
It is often helpful to study a time series model by simulation. This enables the main features of the model to be ’observed’ in plots, so that when historical data exhibit similar features the model may be selected as a potential candidate. The following commands can be used to simulate random walk data for y.

```
set.seed(2) #we do this to get
#the same random numbers
y<-e<-rnorm(1000)
for (t in 2:1000){ y[t] <- y[t-1] + e[t]}
plot(y, type=’l’,col=’blue’)
```

Simulation (plot)



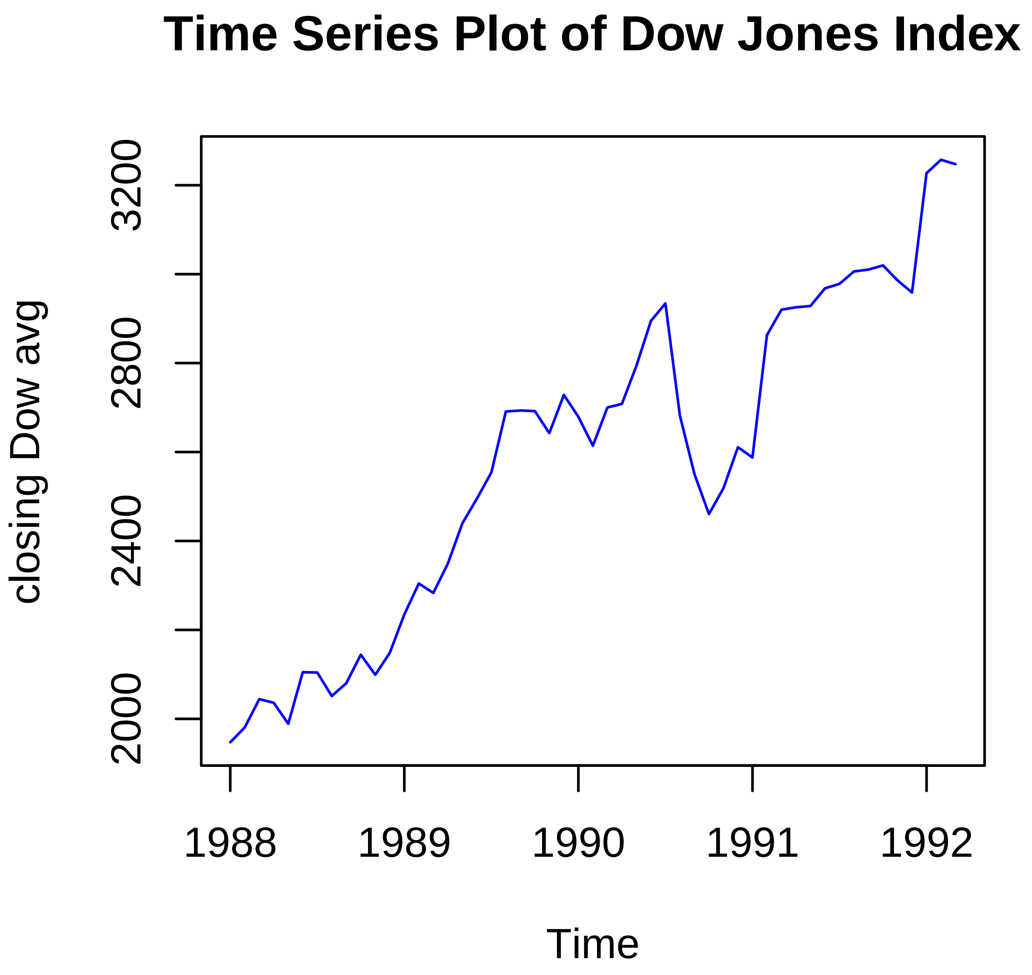
Autocorrelations for Random Walk



Random Walk Model

Random series are sometimes building blocks for other time series models. The model we now discuss, the random walk model, is an example of this. In a random walk model the series itself is not random. However, its differences - that is, the changes from one period to the next - are random. This type of behavior is typical of stock price data. For example, the graph in the next slide shows monthly Dow Jones averages from January 1988 through March 1992. (See the file dow.txt)

Time series chart of closing Dow average



More about Dow Jones

This series is not random, as can be seen from its gradual upward trend (Although the runs test and autocorrelations are not shown here, they confirm that the series is not random.)

If we were standing in March 1992 and were asked to forecast the Dow Jones average for the next few months, it is intuitive that we would not use the average of the historical values as our forecast. This forecast would probably be too low because the series has an upward trend. Instead, we would base our forecast on the most recent observation. This is exactly what the random walk model does.

Random walk (equation)

An equation for the random walk model is

$$Y_t = Y_{t-1} + \mu + \epsilon_t \tag{15}$$

where μ is a constant and ϵ_t is a random series with mean 0 and some standard deviation σ. If we let $DY_t = Y_t - Y_{t-1}$, the change in the series from time t to time $t-1$, then we can write the random walk model as

$$DY_t = \mu + \epsilon_t \tag{16}$$

This implies that the differences form a random series with mean μ and standard deviation σ.

Random walk (cont.)

An estimate of μ is the average of the differences, labeled $\bar{Y}_D$, and an estimate of σ is the sample standard deviation of the differences , labeled s_D. In words, a series that behaves according to this random walk model has random differences, and the series tends to trend upward (if $\mu > 0$) or downward (if $\mu < 0$) by an amount μ each period. If we are standing in period t and want to make a forecast F_{t+1} of Y_{t+1}, then a reasonable forecast is

$$F_{t+1} = Y_t + \bar{Y}_D \quad (17)$$

That is, we add the estimated trend to the current observation to forecast the next observation.

Example: Dow Jones revisited

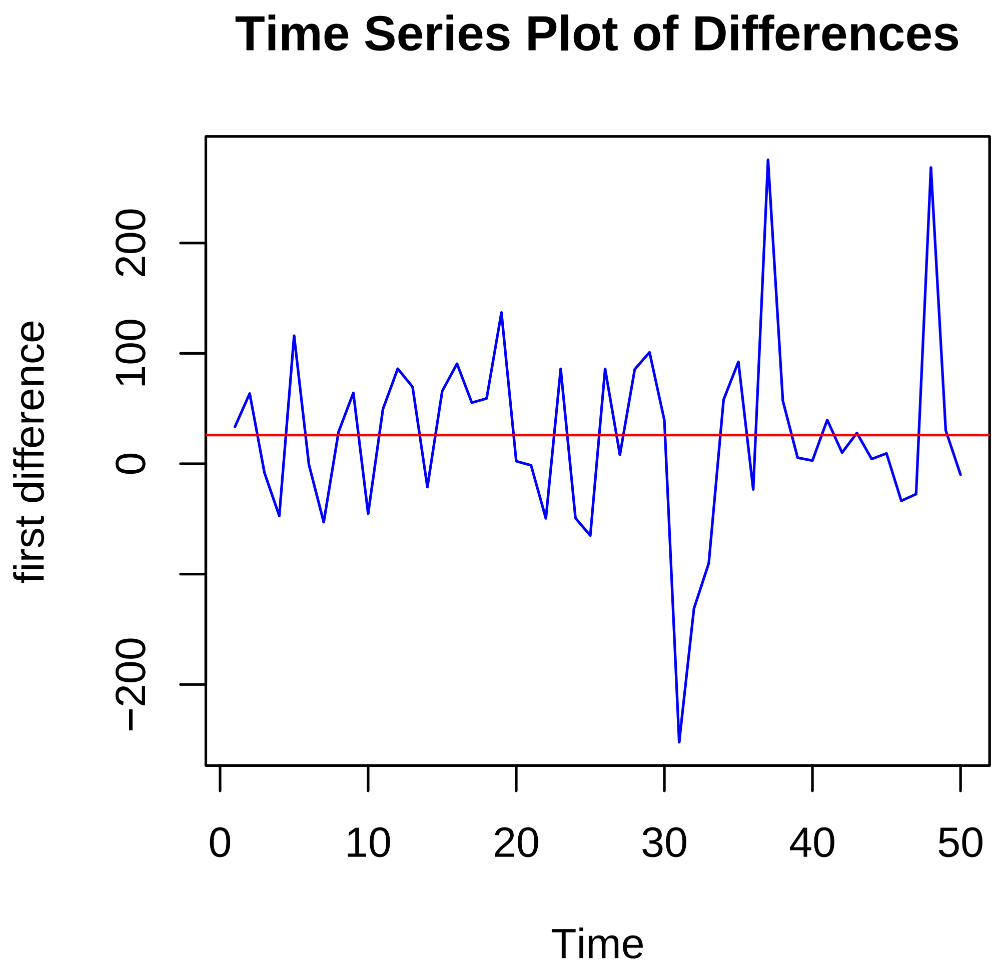
Given the monthly Dow Jones data in the file dow.txt, check that it satisfies the assumptions of a random walk, and use the random walk model to forecast the value for April 1992.

Solution

We have already seen that the Dow Jones series itself is not random, due to the upward trend, so we form the differences

```
diff.dow<-diff(index)
ts.diff<-ts(diff.dow)
plot(ts.diff,col='blue',
ylab='first difference', main='Time Series
Plot of Differences')
abline(h=mean(diff.dow),col='red')
```

Time Series Plot of Dow Differences



Dow Differences (cont.)

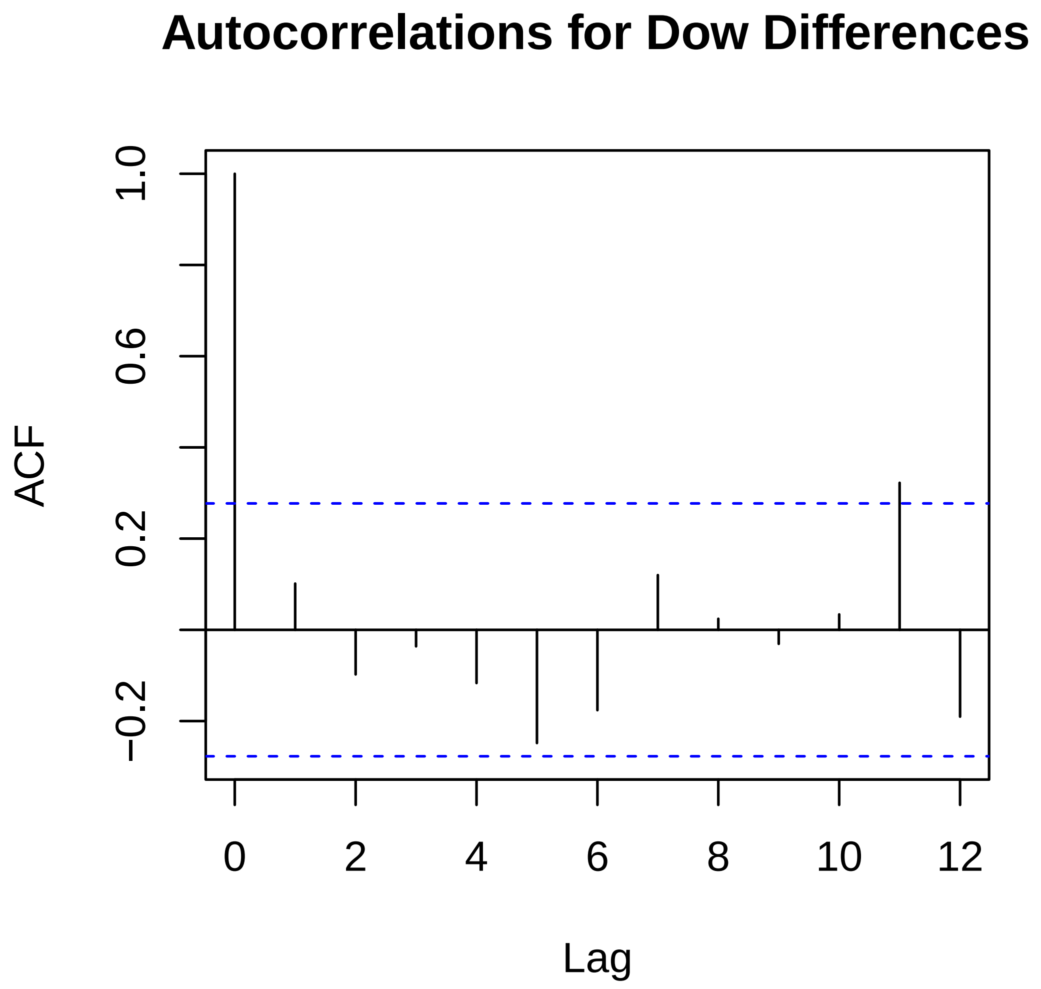
It appears to be a much more random series, varying around the mean difference 26. The runs test shows that there is absolutely no evidence of nonrandom differences.

```
runs(diff.dow)
$T [1] 50
$Ta [1] 26
$Tb [1] 24
$R [1] 26
$Z [1] 0.01144965
$‘E(R)‘ [1] 25.96
$‘P value‘ [1] 0.9908647
```

Dow Differences (cont.)

Similarly, the autocorrelations are all small except for a random "blip" at lag 11. Because there is probably no reason to believe that values 11 months apart really are related, we would tend to ignore this autocorrelation.

Correlogram for Dow Differences



Forecast of April 1992

Assuming the random walk model is adequate, the forecast of April 1992 made in March 1992 is the observed March value, 3247.42, plus the mean difference, 26, or 3273.42. A measure of the forecast accuracy is provided by the standard deviation, s_D=84.65, of the differences. Provided that our assumptions hold, we can be 95 % confident that our forecast is off by no more than $2s_D$, or about 170.

Exercise

Write a function in R that gives you a one-step-ahead forecast for a Random Walk Model.

One solution

```
forecast<-function(y){

diff<-diff(y)
y.diff.bar<-mean(diff)   #average difference
last<-length(y) #last observation
F.next<-y[last]+y.diff.bar
new.y<-c(y,F.next)

list('Y(t)'=y,'Y(t+1)'=new.y,'F(t+1)'=F.next)

                }
```

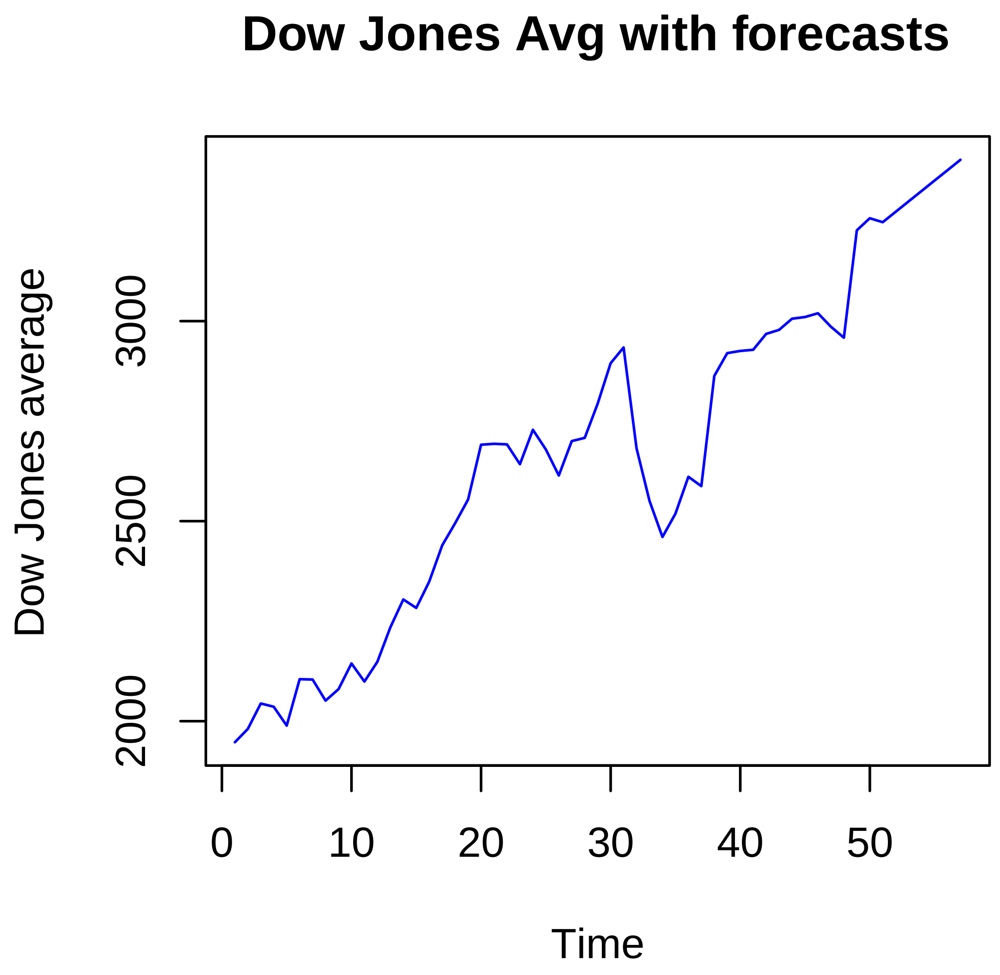
Exercise

Using your one-step-ahead forecast function, write another function in R that computes forecast for times: t+1,t+2,...,t+N, where t represents the length of your original time series.

One solution

```
forecast.N<-function(y,N){

original<-y    #original time series

for (i in 1:N){
new<-forecast(original)$'Y(t+1)'
original<-new
                }

list('Y(t)'=y,'N'=N,'Y(t+N)'=original)

               }
```

Forecasts for Dow Jones Avg



Another simulation

```
set.seed(1)
y <- e <- rnorm(100)
for (t in 2:100) {
y[t]<-0.7*y[t-1] + e[t]}
plot(y,type='l',col='blue')
acf(y)
```

Autoregression Models

A regression-based extrapolation method is to regress the current value of the time series on past (lagged) values. This is called autoregression, where the "auto" means that the explanatory variables in the equation are lagged values of the response variable, so that we are regressing the response variable on lagged versions of itself. Some trial and error is generally required to see how many lags are useful in the regression equation. The following exercise illustrates the procedure.

Exercise

- Plot your time series
- Make a correlogram
- Fit an autoregressive model of order p (p suggested by your correlogram)
- Determine if your model is adequate
- Fit another model if necessary

IUP

Exercise: Hammers

A retailer has recorded its weekly sales of hammers (units purchased) for the past 42 weeks. (See the file hammers.txt) How useful is autoregression for modeling these data and how would it be used for forecasting?

Exercise: Hammers (cont.)

- Plot your time series
- Make a correlogram
- Fit an autoregressive model of order p (p suggested by your correlogram)
- Determine if your model is adequate
- Fit another model if necessary

IUP

Regression-Based Trend Models

Many time series follow a long-term trend except for random variation. This trend can be upward or downward. A straightforward way to model this trend is to estimate a regression equation for Y_t, using time t as the single explanatory variable. In this "section" we will discuss the two most frequently used trend models, linear trend and exponential trend.

Linear Trend

A linear trend means that the time series variable changes by a constant amount each time period. The relevant equation is

$$Y_t = \alpha + \beta t + \epsilon_t \tag{18}$$

where, as in previous regression equations, α is the intercept, β is the slope, and ϵ_t is an error term.

Exercise: Reebok

The file reebok.txt includes quarterly sales data for Reebok from the first quarter 1986 through second quarter 1996. Sales increase from $ 174.52 million in the first quarter to $ 817.57 million in the final quarter. How well does a linear trend fit these data? Are the residuals from this fit random?

Exercise: Reebok (cont.)

- Plot your time series
- Fit a linear regression model and interpret your results
- Find the residuals
- Determine if the residuals are random

Exponential Trend

In contrast to a linear trend, an exponential trend is appropriate when the time series changes by a constant percentage (as opposed to a constant dollar amount) each period. Then the appropriate regression equation is

$$Y_t = c \exp(bt) u_t \tag{19}$$

where c and b are constants, and u_t represents a multiplicative error term. By taking logarithms of both sides, and letting $a = ln(c)$ and $\epsilon_t = ln(u_t)$, we obtain a linear equation that can be estimated by the usual linear regression method. However, note that the response variable is now the logarithm of Y_t:

$$ln(Y_t) = a + bt + \epsilon_t \tag{20}$$

Exercise: Intel

The file intel.txt contains quarterly sales data for the chip manufacturing firm Intel from the beginning of 1986 through the second quarter of 1996. Each sales value is expressed in millions of dollars. Check that an exponential trend fits these sales data fairly well. Then estimate the relationship and interpret it.

Exercise: Intel (cont)

- Plot your time series
- If your original time series shows an exponential trend, apply ln to the series
- Use the transformed series to estimate the relationship
- Express the estimated relationship in the original scale
- Find an estimate of the standard deviation

Moving Averages

Perhaps the simplest and one of the most frequently used extrapolation methods is the method of moving averages. To implement the moving averages method, we first choose a span, the number of terms in each moving average. Let's say the data are monthly and we choose a span of 6 months. Then the forecast of next month's value is the average of the most recent 6 month's values. For example, we average January-June to forecast July, we average February-July to forecast August, and so on. This procedure is the reason for the term moving averages.

Exercise (toy example)

Suppose that $Y(t) = [4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15]$.

- Plot your time series
- Write a function in R that finds a one-step-ahead moving average forecast for a span=2
- Write a function in R that does a moving average "smoothing" to a time series

Exercise: Dow Jones revisited

We again look at the Dow Jones monthly data from January 1988 through March 1992. (See the file dow.txt). How well do moving averages track this series when the span is 3 months; when the span is 12 months?

Exponential Smoothing

There are two possible criticisms of the moving averages method. First, it puts equal weight on each value in a typical moving average when making a forecast. Many people would argue that if next month's forecast is to be based on the previous 12 months' observations, then more weight ought to be placed on the more recent observations. The second criticism is that moving averages method requires a lot of data storage. This is particularly true for companies that routinely make forecasts of hundreds or even thousands of items. If 12-month moving averages are used for 1000 items, then 12000 values are needed for next month's forecasts. This may or may not be a concern considering today's relatively inexpensive computer storage capabilities.

Exponential Smoothing (cont.)

Exponential smoothing is a method that addresses both of these criticisms. It bases its forecasts on a weighted average of past observations, with more weight put on the more recent observations, and it requires very little data storage. There are many versions of exponential smoothing. The simplest is called simple exponential smoothing. It is relevant when there is no pronounced trend or seasonality in the series. If there is a trend but no seasonality, then Holt's method is applicable. If, in addition, there is seasonality, then Winter's method can be used.

Simple Exponential Smoothing

We now examine simple exponential smoothing in some detail. We first introduce two new terms. Every exponential model has at least one smoothing constant, which is always between 0 and 1. Simple exponential smoothing has a single smoothing constant denoted by α. The second new term is L_t, called the level of the series at time t. This value is not observable but can only be estimated. Essentially, it is where we think the series would be at time t if there were no random noise.

Simple Exponential Smoothing (cont.)

Then the simple exponential smoothing method is defined by the following two equations, where F_{t+k} is the forecast of Y_{t+k} made at time t:

(21) $$L_t = \alpha Y_t + (1-\alpha)L_{t-1}$$

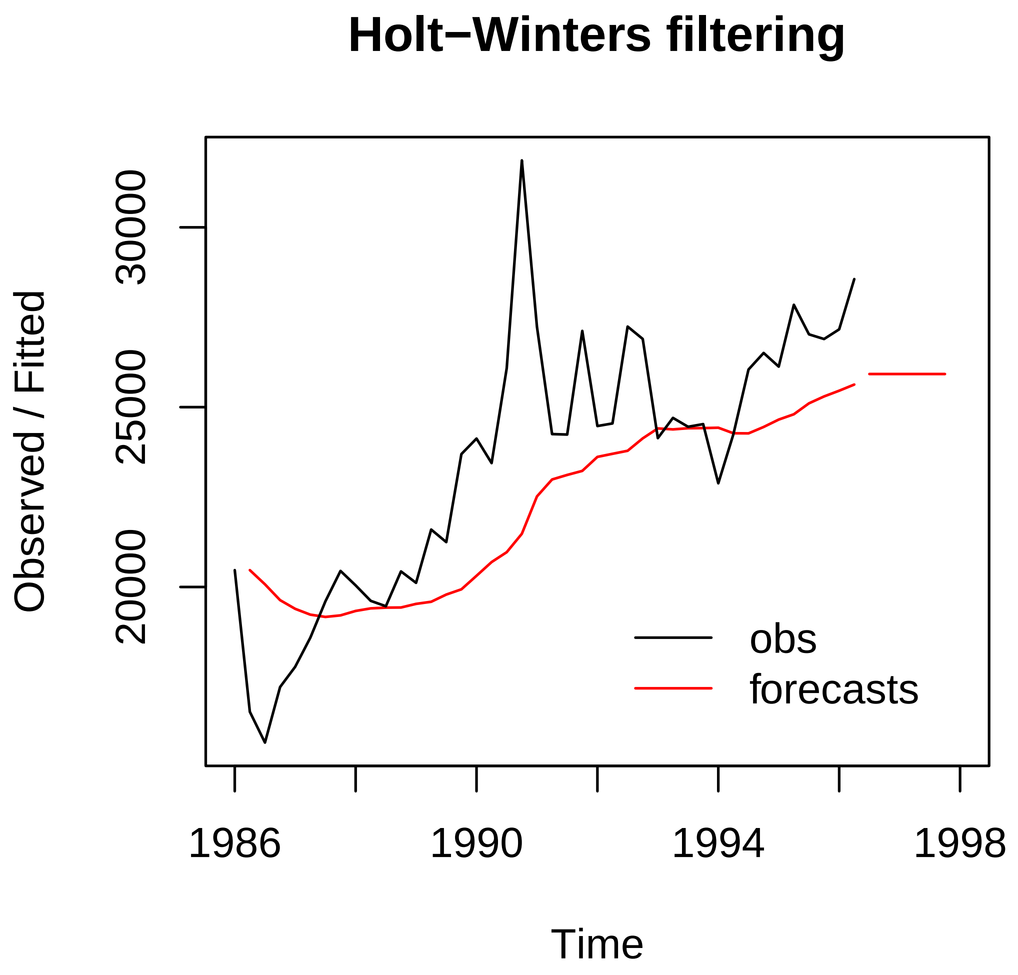
(22) $$F_{t+k} = L_t$$

Example:Exxon

The file exxon.txt contains data on quarterly sales (in millions of dollars) for the period from 1986 through the second quarter of 1996. Does a simple exponential smoothing model track these data well? How do the forecasts depend on the smoothing constant α?

Example: Exxon (cont.)

Simple Exponential Smoothing with $\alpha = 0.01$



Example: Exxon (R code)

```
exxon<-read.table(file="exxon.txt",
header=TRUE)
sales<-exxon$Sales
new.sales<-ts(sales,start=c(1986,1),
end=c(1996,2),freq=4)
mod1<-HoltWinters(new.sales,alpha=0.1,
beta=FALSE,gamma=FALSE)
plot(mod1,xlim=c(1986,1998))
lines(predict(mod1,n.ahead=6),col="red")
legend(1992,20000,c("obs","forecasts"),
col=c("black","red"),lty=c(1,1),bty="n")
```

Exxon (changing α)

The following lines of code produce a "movie" that illustrates the effect of changing α.

```
n<-100
for (i in 1:n){
    mod1<-HoltWinters(new.sales,
    alpha=(1/n)*i,beta=FALSE,gamma=FALSE)
    plot(mod1,xlim=c(1986,1997))
    Sys.sleep(0.3)

        }
```

Choosing α.

What value of α should we use? There is no universally accepted answer to this question. Some practitioners recommend always using a value around 0.1 or 0.2. Others recommend experimenting with different values of α until a measure such as the Mean Square Error (MSE) is minimized.

R can find this optimal value of α as follows:

```
mod2<-HoltWinters(new.sales,
beta=FALSE,gamma=FALSE)
```

Holt’s Model for Trend

The simple exponential smoothing model generally works well if there is no obvious trend in the series. But if there is a trend, then this method consistently lags behind it. Holt’s method rectifies this by dealing with trend explicitly. In addition to the level of the series, L_t, Holt’s method includes a trend term, T_t, and a corresponding smoothing constant β. The interpretation of L_t is exactly as before.

Holt's Model for Trend (cont.)

The interpretation of T_t is that it represents an estimate of the change in the series from one period to the next. The equations for Holt's model are as follows:

$$L_t = \alpha Y_t + (1-\alpha)(L_{t-1} + T_{t-1}) \tag{23}$$

$$T_t = \beta(L_t - L_{t-1}) + (1-\beta)T_{t-1} \tag{24}$$

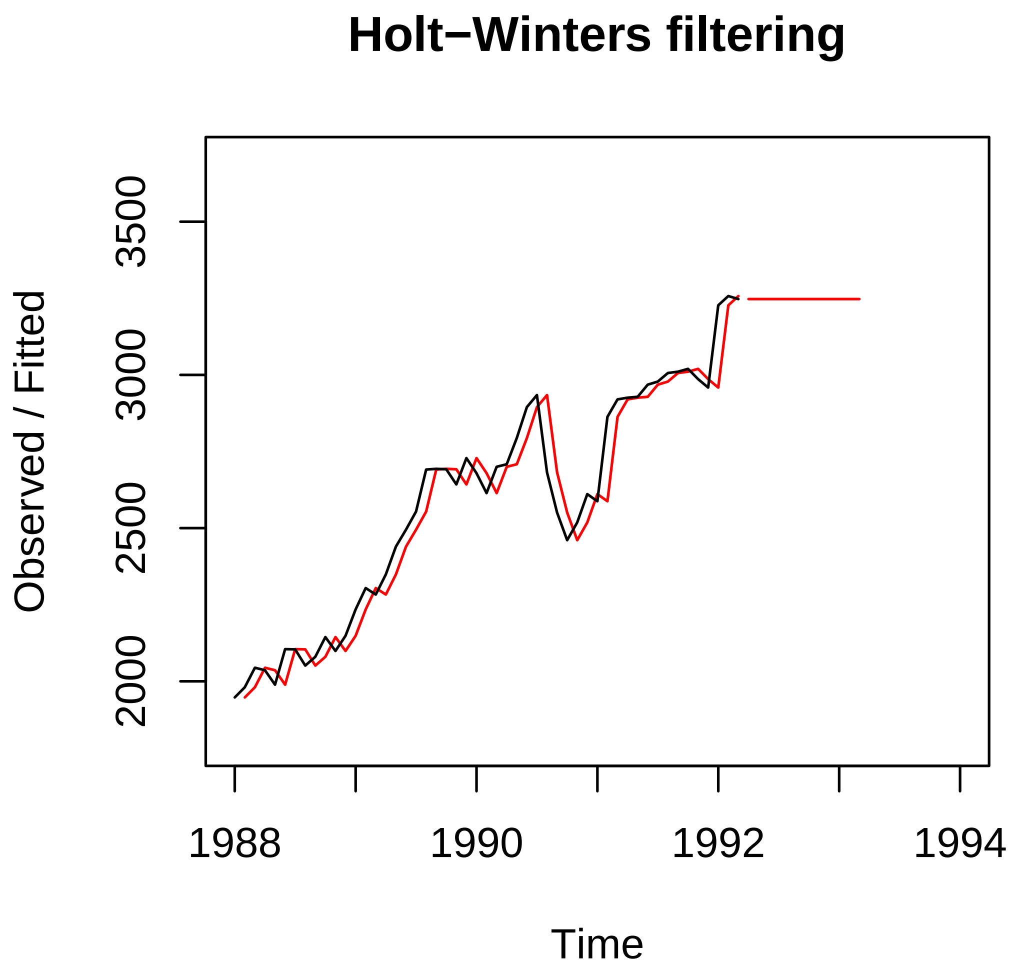
$$F_{t+k} = L_t + kT_t \tag{25}$$

Example: Dow Jones revisited

We return to the Dow Jones data (see file dow.txt). Again, these are average monthly closing prices from January 1988 through March 1992. Recall that there is a definite upward trend in this series. In this example we investigate whether simple exponential smoothing can capture the upward trend. Then we see whether Holt's exponential smoothing method can make an improvement.

Example: Dow Jones (cont.)

Simple Exponential Smoothing with optimal α

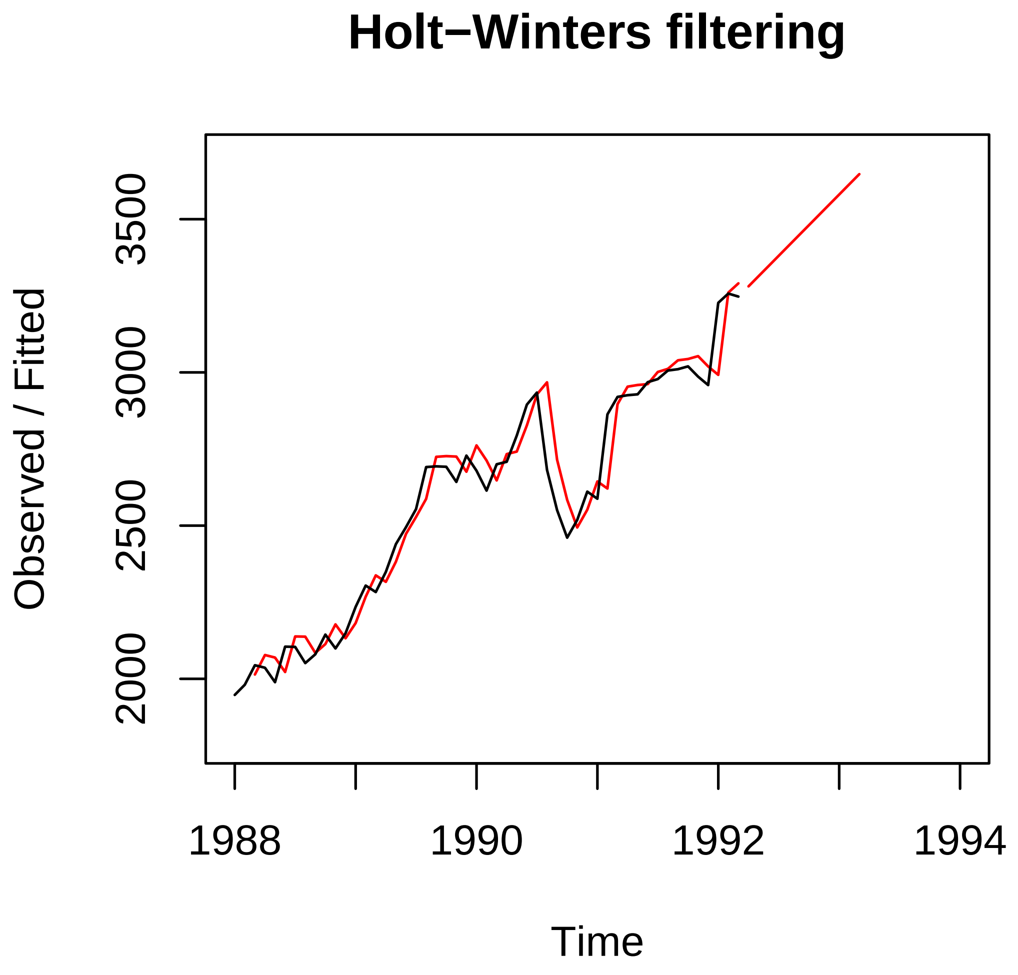


Example: Dow Jones (R code)

```
dow<-read.table(file="dow.txt",header=TRUE)
index<-dow$Dow
DJI<-ts(index,start=c(1988,1),end=c(1992,3),
freq=12)
mod1<-HoltWinters(DJI,beta=FALSE,gamma=FALSE)

### predictions
plot(mod1,xlim=c(1988,1994),
ylim=c(1800,3700))
preds<-predict(mod1,n.ahead=12)
lines(preds,col="red")
```

Dow Jones (Holt's Model)

Holt's Model with optimal α and β.



R code (Holt’s Model)

```
mod2<-HoltWinters(DJI,gamma=FALSE)
#Fitting Holt’s model

### predictions

plot(mod2,xlim=c(1988,1994),
ylim=c(1800,3700))
preds<-predict(mod2,n.ahead=12)
lines(preds,col=’red’)
```

Winter's Model

So far we have said practically nothing about seasonality. Seasonality is defined as the consistent month-to-month (or quarter-to-quarter) differences that occur each year. For example, there is seasonality in beer sales - high in the summer months, lower in other months.
How do we know whether there is seasonality in a time series? The easiest way is to check whether a plot of the time series has a regular pattern of ups and /or downs in particular months or quarters.

IUP

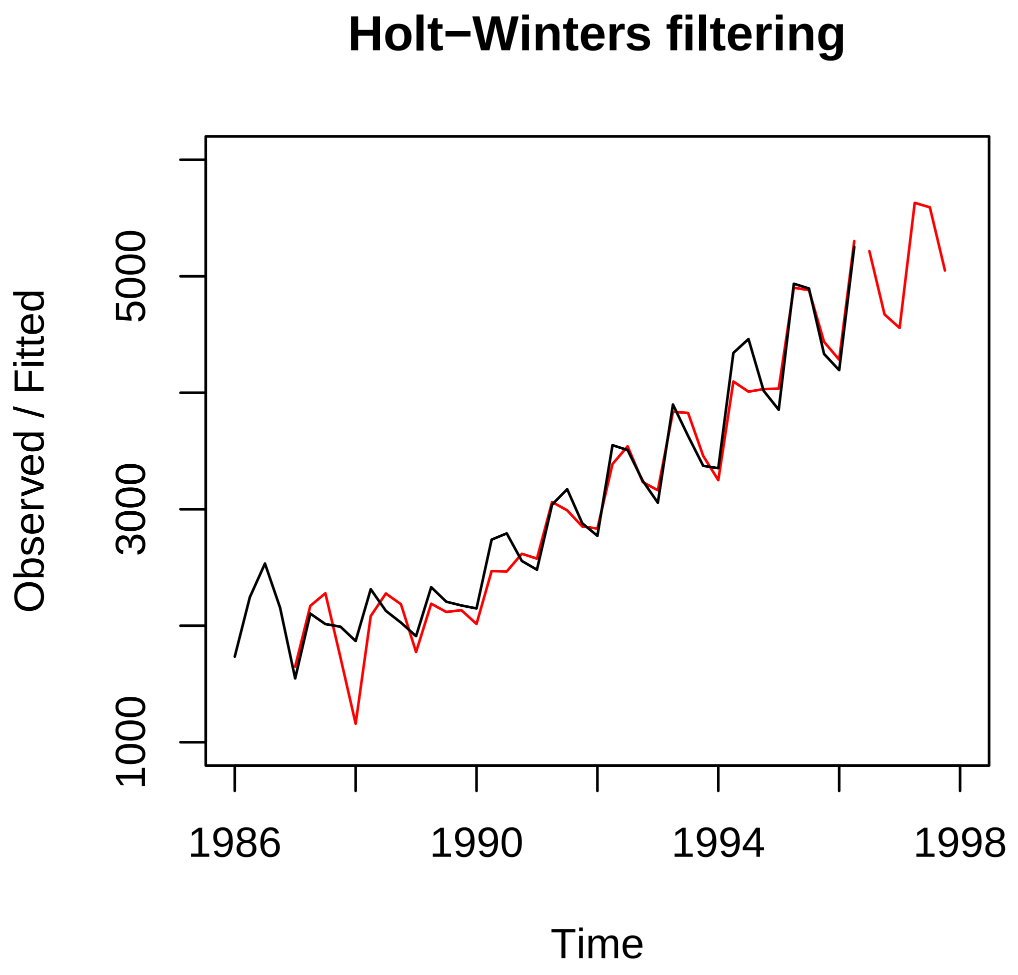
Winter's Model (cont.)

Winter's exponential smoothing model is very similar to Holt's model- it again has a level and a trend terms and corresponding smoothing constants α and β- but it also has seasonal indexes and a corresponding smoothing constant γ. This new smoothing constant γ controls how quickly the method reacts to perceived changes in the pattern of seasonality.

Example: Coke

The data in the coke.txt file represent quarterly sales (in millions of dollars) for Coca Cola from quarter 1 of 1986 through quarter 2 of 1996. As we might expect, there has been an upward trend in sales during this period, and there is also a fairly regular seasonal pattern, as shown in the next slide. Sales in the warmer quarters, 2 and 3, are consistently higher than in colder quarters, 1 and 4. How well can Winter's method track this upward trend and seasonal pattern?

Coke sales with forecasts



R code

```
mod.coke<-HoltWinters(sales)
# Holt-Winter’s model

### predictions

plot(mod.coke,ylim=c(1000,6000),
xlim=c(1986,1998))

preds<-predict(mod.coke,n.ahead=6)

lines(preds,col=’red’)
```

ARIMA-Models

Forecasting based on ARIMA (autoregressive integrated moving averages) models, commonly known as the Box-Jenkins approach, comprises the following stages:

- Model Identification
- Parameter estimation
- Diagnostic checking
- Iteration

These stages are repeated until a "suitable" model for the given data has been identified.

Identification

Various methods can be useful for identification:

- Time Series Plot
- ACF / Correlogram
- PACF (Partial Autocorrelation Function)
- Test for White Noise

Important Processes

- White Noise
- Moving Average Process (MA)
- Autoregressive Process (AR)

White Noise

White Noise. A stationary time series for which $Y(t)$ and $Y(t+k)$ are uncorrelated, i.e., $E[Y(t)-\mu][Y(t+k)-\mu]=0$ for all integers $k \neq 0$, is called white noise. Such process is sometimes loosely termed a "purely random process".

Moving Average Process

Moving Average Process. A moving average process of order q, denoted $MA(q)$, is defined by the equation

$$Y(t) = \mu + \beta_0\epsilon_t + \beta_1\epsilon_{t-1} + ... + \beta_q\epsilon_{t-q} \tag{26}$$

where ϵ_t is a white noise process.

Autoregressive Process

Consider a time series $Y(t)$ that satisfies the difference equation

$$Y(t) = \alpha_1 Y(t-1) + \alpha_2 Y(t-2) + ... + \alpha_p Y(t-p) + \epsilon_t \tag{27}$$

where ϵ_t is a white noise process with zero mean and finite variance σ^2. The time series $Y(t)$ is called an autoregressive process of order p and is denoted $AR(p)$.

Analysis of ACF and PACF

A first step in analyzing time series is to examine the autocorrelations (ACF) and partial autocorrelations ($PACF$). R provides the functions $acf()$ and $pacf()$ for computing and plotting ACF and PACF. The order of "pure" AR and MA processes can be identified from the ACF and $PACF$.

Simulating AR(1) and MA(1)

```
#simulation of an AR(1) process

sim.ar<-arima.sim(list(ar=c(0.7)),n=1000)

#simulation of a MA(1) process

sim.ma<-arima.sim(list(ma=c(0.6)),n=1000)
```

ACF and PACF for our simulations

```
# setting for a "matrix" of plots

par(mfrow=c(2,1))

## AR process

acf(sim.ar,main="ACF of AR(1) process")

pacf(sim.ar, main="PACF of AR(1) process")
```

ACF and PACF for our simulations

```
# setting for a "matrix" of plots

par(mfrow=c(2,1))

## MA process

acf(sim.ma,main="ACF of MA(1) process")

pacf(sim.ma, main="PACF of MA(1) process")
```

Simulating AR(2) and MA(2)

```
#simulation of an AR(2) process

sim.ar<-arima.sim(list(ar=c(0.4,0.4)),n=1000)

#simulation of a MA(2) process

sim.ma<-arima.sim(list(ma=c(0.6,-0.4)),n=1000
```

ACF and PACF for our simulations

```
# setting for a "matrix" of plots

par(mfrow=c(2,1))

## AR process

acf(sim.ar,main="ACF of AR(2) process")

pacf(sim.ar, main="PACF of AR(2) process")
```

ACF and PACF for our simulations

```
# setting for a "matrix" of plots

par(mfrow=c(2,1))

## MA process

acf(sim.ma,main="ACF of MA(2) process")

pacf(sim.ma, main="PACF of MA(2) process")
```

Simulating AR(3) and MA(3)

```
#simulation of an AR(3) process

sim.ar<-arima.sim(list(ar=c(0.4,0.3,0.2)),
n=1000)

#simulation of a MA(3) process

sim.ma<-arima.sim(list(ma=c(0.6,-0.4,0.5)),
n=1000)
```

ACF and PACF for our simulations

```
# setting for a "matrix" of plots

par(mfrow=c(2,1))

## AR process

acf(sim.ar,main="ACF of AR(3) process")

pacf(sim.ar, main="PACF of AR(3) process")
```

ACF and PACF for our simulations

```
# setting for a "matrix" of plots

par(mfrow=c(2,1))

## MA process

acf(sim.ma,main="ACF of MA(3) process")

pacf(sim.ma, main="PACF of MA(3) process")
```

Important conclusions from simulations

- We know that for the moving average model $MA(q)$ the theoretical ACF has the property that $\rho_h = 0$ for $h > q$, so we can expect the sample ACF $r_h \approx 0$ when $h > q$.
- If the process $Y(t)$ is really $AR(p)$, then there is no dependence of $Y(t)$ on $Y(t-p-1), Y(t-p-2), ...$ once $Y(t-1), ..., Y(t-p)$ have been taken into account. Thus we might expect that $\alpha_h \approx 0$ for $h > p$, i.e. that the $PACF$ cuts off beyond p.

Behavior of Theoretical ACF and PACF

- $MA(q)$: ACF cuts off after lag q.
- $MA(q)$: $PACF$ shows exponential decay and/or dampened sinusoid.
- $AR(p)$: ACF shows exponential decay and/or dampened sinusoid.
- $AR(p)$: $PACF$ cuts off after lag q.
- $ARMA(p, q)$:ACF shows exponential decay and/or dampened sinusoid.
- $ARMA(p, q)$:$PACF$ shows exponential decay and/or dampened sinusoid.

Fitting

Suppose that the observed series has, if necessary, been differenced to reduce it to stationarity and that we wish to fit an $ARMA(p,q)$ model with known p and q to the stationary series. Fitting the model then amounts to estimating the $p+q+1$ parameters, the coefficients for the $AR(p)$ process, the coefficients for the $MA(q)$ process and σ^2.

The three main methods of estimation are (Gaussian) maximum likelihood, least squares and solving the Yule-Walker equations. (We are not going to talk about these methods in detail, Sorry!)

Diagnostics

Having estimated parameters we need to check whether the model fits, and, if it doesn't, detect where it is going wrong. Specific checks could include:

- Time Plot of Residuals. A plot of residuals, e_t, against t should not show any structure. Similarly for a plot of e_t against $\hat{Y}(t)$.
- ACF and PACF of Residuals. The hope is that these will show the White Noise pattern: approximately independent, mean zero, normal values, each with variance n^{-1}.

Diagnostics (cont.)

- The Box-Pierce (and Ljung-Box) test examines the Null of independently distributed residuals. It's derived from the idea that the residuals of a "correctly specified" model are independently distributed. If the residuals are not, then they come from a miss-specified model.

Summary of R commands

- $arima.sim()$ was used to simulate $ARIMA(p, d, q)$ models
- $acf()$ plots the Autocorrelation Function
- $pacf()$ plots the Partial Autocorrelation Function
- $arima(data, order = c(p, d, q))$ this function can be used to estimate the parameters of an $ARIMA(p, d, q)$ model
- $tsdiag()$ produces a diagnostic plot of fitted time series model
- $predict()$ can be used for predicting future values of the levels under the model

Exercises

- Fit an ARIMA model to ts1.txt
- Fit an ARIMA model to ts2.txt
- Fit an ARIMA model to ts3.txt
- Fit an ARIMA model to ts4.txt

THANK YOU!!!