Forecasting Princples and Practice

Koji Mizumura

August 28, 2018(Beg) - (End)

Table of Contents

This document is html\_notebook output (not html\_document).

# Resources

[Forecasting: princples and practice](https://otexts.org/fpp2/)  
[Slides](https://robjhyndman.com/seminars/uwa2017/)  
[Github](https://github.com/nealxun/ForecastingPrinciplePractices)

From the Github, we can install the package fpp2.

install.packages("fpp2",dependencies = TRUE

# Time series regression models

In this chapter, we discuss regression models. The basic concept is that we forecast the time series of interest assuming that it has a linear relationship with other time series .

For example, we might wish to forecast monthly sales using total advertising spend as a predictor. OR we might forecast daily electricity demand using temparature and the day of week as predictors.

The **forecast variable**  is sometimes called the regressor, dependent or explained variables. The predictor variables are sometimes also called the regressors, independent or explanatory variables. In this book we will always refer to them as the “forecast” variable and “predictor” variables.

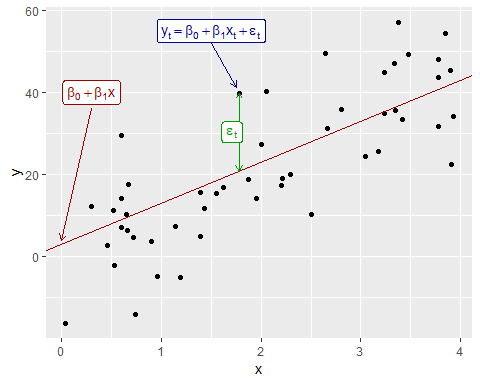
## The linear model

### Simple linear regression

In this simplest case, the regression model allows for a linear relationship between the forecast variable and a single predictor variable :

An rtificial example of data from such a model is shown in Figure 5.1. The coefficients and denote the intercept and the slope of the line respectively. The intercept represents the predicted value of when . The slope represents the average predicted change in resulting from a one unit increase in .

set.seed(2)  
x <- runif(50, 0, 4)  
df <- data.frame(x=x,  
 y=3 + 10\*x + rnorm(50, 0, 10))  
ggplot(df, aes(x, y)) +  
 geom\_point() +  
 geom\_abline(slope=10, intercept=3,  
 col="#990000", size=0.3) +  
 geom\_label(x=.3, y=40, parse=TRUE, col="#990000",  
 label=" beta[0] + beta[1] \* x") +  
 geom\_segment(x=.3, y=36, xend=0, yend=4,  
 arrow=arrow(length = unit(0.02, "npc")),  
 size=0.2, col='#990000') +  
 geom\_label(x=1.5, y=55, parse=TRUE, col="#000099",  
 label="y[t] == beta[0] + beta[1] \* x[t] + epsilon[t]") +  
 geom\_segment(x=1.5, y=52, xend=df$x[19]-0.03, yend=df$y[19]+1.5,  
 arrow=arrow(length = unit(0.02, "npc")),  
 size=0.2, col='#000099') +  
 geom\_segment(x=df$x[19], y=df$y[19],  
 xend=df$x[19], yend=3+10\*df$x[19],  
 col="#009900", size=0.2,  
 arrow=arrow(length=unit(0.02,"npc"), ends="both")) +  
 geom\_label(x=df$x[19]-0.07,  
 y=(df$y[19]+ 3+10\*df$x[19])/2,  
 col="#009900", label="epsilon[t]",  
 parse=TRUE)

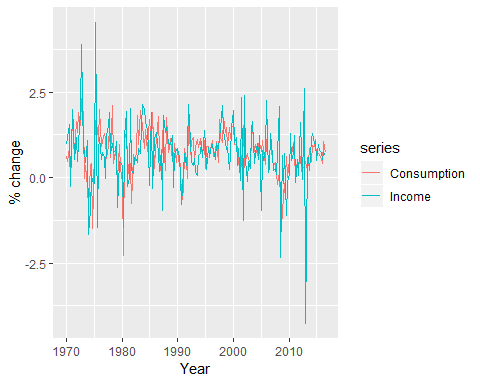


Notice that the observations do not lie on the straight line but are scattered around it. We can think of each observation consisting of the systematic or explained part of the model, , and the random “error”, . The “error” term does not imply a mistake, but a deviation from the underlying straight line model. It captures anything that may affect other than .

#### Example: US consumption expenditure

Figure @ref(fig:ConsInc) shows time series of quarterly percentage changes (growth rates) of real personal consumption expenditure () and real personal disposable income () for the US from 1970 Q1 to 2016 Q3.

autoplot(uschange[,c("Consumption","Income")]) +  
 ylab("% change") + xlab("Year")

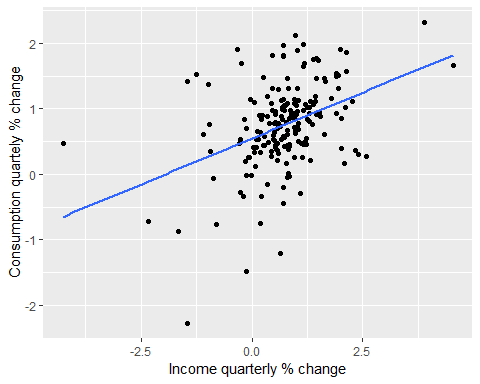


Percentage changes in personal consumption expenditure and personal income for the US.

A scatter plot of consumption against income is shown in Figure @ref(fig:ConsInc2) along with the estimated regression line

(We put a “hat” above to indicate that this is the value of predicted by the model.)

uschange %>%   
 as.data.frame() %>%   
 ggplot(aes(x=Income,y=Consumption))+  
 ylab("Consumption quartely % change")+  
 xlab("Income quarterly % change")+  
 geom\_point()+  
 geom\_smooth(method="lm",se=FALSE)



Scatterplot of consumption versus income and the fitted regression line.

This equation is estimated in R using the tslm function:

tslm(Consumption~Income,data=uschange)

##   
## Call:  
## tslm(formula = Consumption ~ Income, data = uschange)  
##   
## Coefficients:  
## (Intercept) Income   
## 0.5451 0.2806

We will discuss how tslm() computes the coefficients in Section @ref(Regr-LSprinciple).

The fitted line has a positive slope, reflecting the positive relationship between income and consumption. The slope coefficient shows that a one unit increase in (a 1% increase in personal disposable income) results on average in 0.28 units increase in (an average 0.28% increase in personal consumption expenditure). Alternatively the estimated equation shows that a value of 1 for (the percentage increase in personal disposable income) will result in a forecast value of for (the percentage increase in personal consumption expenditure).

The interpretation of the intercept requires that a value of makes sense. In this case when (i.e., when there is no change in personal disposable income since the last quarter) the predicted value of is 0.55 (i.e., an average increase in personal consumption expenditure of 0.55%). Even when does not make sense, the intercept is an important part of the model. Without it, the slope coefficient can be distorted unnecessarily. The intercept should always be included unless the requirement is to force the regression line “through the origin”. In what follows we assume that an intercept is always included in the model.

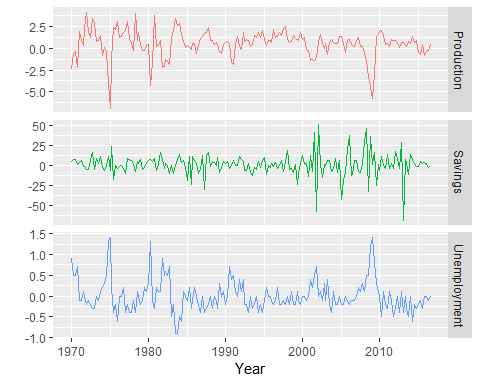
### Multiple linear regression

When there are two or more predictor variables, the model is called a “multiple regression model”. The general form of a multiple regression model is

where is the variable to be forecast and are the predictor variables. Each of the predictor variables must be numerical. The coefficients measure the effect of each predictor after taking account of the effect of all other predictors in the model. Thus, the coefficients measure the *marginal effects* of the predictor variables.

#### Example: US consumption expenditure

Figure @ref(fig:MultiPredictors) shows additional predictors that may be useful for forecasting US consumption expenditure. These are quarterly percentage changes in industrial production and personal savings, and quarterly changes in the unemployment rate (as this is already a percentage). Building a multiple linear regression model can potentially generate more accurate forecasts as we expect consumption expenditure to not only depend on personal income but on other predictors as well.



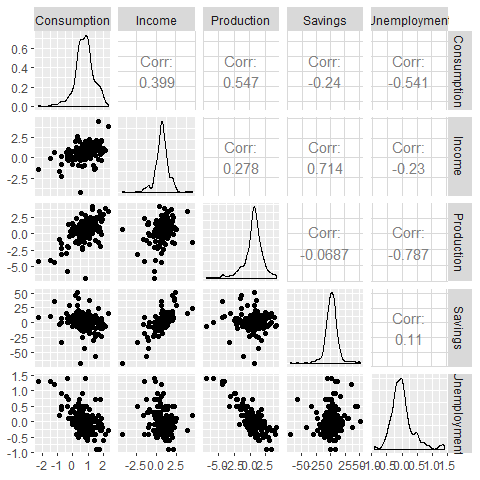
Quarterly percentage changes in industrial production and personal savings and quarterly changes in the unemployment rate for the US over the period 1960Q1-2016Q3.

The first column of Figure @ref(fig:ScatterMatrix) below shows the relationships between consumption and each of the predictors. The scatterplots show positive relationships with income and industrial production, and negative relationships with savings and unemployment. The strength of these relationships are shown by the correlation coefficients across the first row. The remaining scatterplots and correlation coefficients show the relationships between the predictors

head(uschange)

## Consumption Income Production Savings Unemployment  
## 1970 Q1 0.6159862 0.9722610 -2.4527003 4.8103115 0.9  
## 1970 Q2 0.4603757 1.1690847 -0.5515251 7.2879923 0.5  
## 1970 Q3 0.8767914 1.5532705 -0.3587079 7.2890131 0.5  
## 1970 Q4 -0.2742451 -0.2552724 -2.1854549 0.9852296 0.7  
## 1971 Q1 1.8973708 1.9871536 1.9097341 3.6577706 -0.1  
## 1971 Q2 0.9119929 1.4473342 0.9015358 6.0513418 -0.1

uschange %>%   
 as.data.frame() %>%   
 GGally::ggpairs()



A scatterplot matrix of US consumption expenditure and the four predictors.

### Assumptions

When we use a linear regression model, we are implicitly making some assumptions about the variables in Equation @ref(eq:lm).

First, we assume that the model is a reasonable approximation to reality; that is, that the relationship between the forecast variable and the predictor variables satisfies this linear equation.

Second, we make the following assumptions about the errors :

* they have mean zero; otherwise the forecasts will be systematically biased.
* they are not autocorrelated; otherwise the forecasts will be inefficient as there is more information to be exploited in the data.
* they are unrelated to the predictor variables; otherwise there would be more information that should be included in the systematic part of the model.

It is also useful to have the errors are normally distributed with constant variance in order to easily produce prediction intervals.

Another improtant assumption in the linear regression model is that each predictor is not a random variable. If we are performing a controlled experiement in a lab, we could control the values of each (so they would not be random) and observe the resulting values of . With observational data (including most data in business and economics) it is not possible to control the value of , and hence we make this an assumption.

## Least square estimation

In practice, of course, we have a collection of observations but we do not know the values of the coefficients . These need to be estimated from the data.

The least squares principle provides a way of choosing the coefficients effectively by minimizing the sum of the squared errors. That is, we choose the values of that minimize

This is called “least squares” estimation because it gives the least value for the sum of squared errors. Finding the best estimates of the coefficients is often called “fitting” the model to the data, or sometimes “learning” or “training” the model. The line shown in Figure @ref(fig:ConsInc2) was obtained in this way.

When we refer to the *estimated* coefficients, we will use the notation . The equations for these will be given in Section @ref(Regr-MatrixEquations).

The tslm function fits a linear regression model to time series data. It is very similar to the lm function which is widely used for linear models, but tslm provides additional facilities for handling time series.

### Example: US consumption on expenditure (revisited)

A multiple linear regression model for US consumption is

where is the percentage change in real personal consumption expenditure,  
 is the percentage change in real personal disposable income, is the percentage change in industrial production, is the percentage change in personal savings and is the change in the unemployment rate.

The following output provides information about the fitted model. The first column of Coefficients gives an estimate of each coefficient and the second column gives its standard error (i.e., the standard deviation which would be obtained from repeatedly estimating the coefficients on similar data sets). The standard error gives a measure of the uncertainty in the estimated  
 coefficient.

fit.consMR <- tslm(  
 Consumption~Income+Production+Unemployment+Savings,  
 data=uschange  
)  
summary(fit.consMR)

##   
## Call:  
## tslm(formula = Consumption ~ Income + Production + Unemployment +   
## Savings, data = uschange)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -0.88296 -0.17638 -0.03679 0.15251 1.20553   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 0.26729 0.03721 7.184 1.68e-11 \*\*\*  
## Income 0.71449 0.04219 16.934 < 2e-16 \*\*\*  
## Production 0.04589 0.02588 1.773 0.0778 .   
## Unemployment -0.20477 0.10550 -1.941 0.0538 .   
## Savings -0.04527 0.00278 -16.287 < 2e-16 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 0.3286 on 182 degrees of freedom  
## Multiple R-squared: 0.754, Adjusted R-squared: 0.7486   
## F-statistic: 139.5 on 4 and 182 DF, p-value: < 2.2e-16

For forecasting purposes, the final two columns are of limited interest. The “t value” is the ratio of an estimated coefficient to its standard error and the last column gives the p-value: the probability of the estimated coefficient being as large as it is if there was no real relationship between consumption and the corresponding predictor. This is useful when studying the effect of each predictor, but is not particularly useful for forecasting.

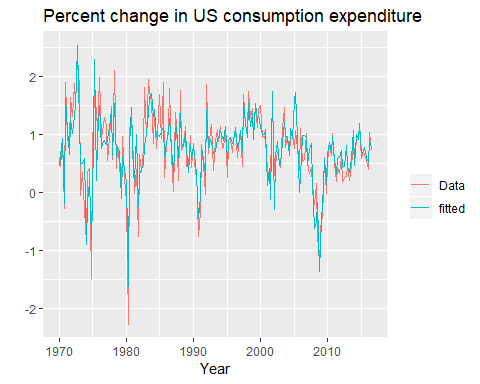
### Fitter values

Predictions of can be obtained by using the estimated coefficients in the regression equation and setting the error term to zero. In general we write,

Plugging in the values of for returns predictions of within the training-sample, referred to as *fitted values*. Note that these are predictions of the data used to estimate the model not genuine forecasts of future values of .

The following plots show the actual values compared to the fitted values for the percentage change in the US consumption expenditure series. The time plot in Figure @ref(fig:usfitted1) shows that the fitted values follow the actual data fairly closely. This is verified by the strong positive relationship shown by the scatterplot in Figure @ref(fig:usfitted2).

autoplot(uschange[,"Consumption"],series="Data")+  
 autolayer(fitted(fit.consMR),series="fitted")+  
 labs(x="Year",y="")+  
 ggtitle("Percent change in US consumption expenditure")+  
 guides(color=guide\_legend(title=" "))

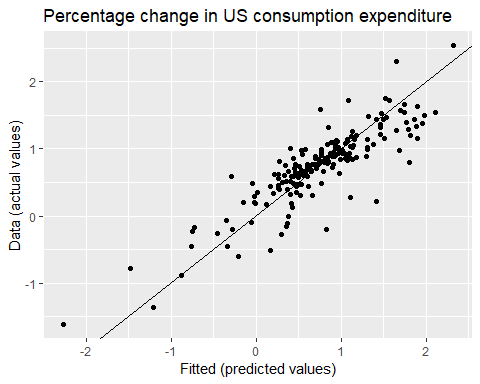


Time plot of US consumption expenditure and predicted expenditure.

fit.consMR %>% summary()

##   
## Call:  
## tslm(formula = Consumption ~ Income + Production + Unemployment +   
## Savings, data = uschange)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -0.88296 -0.17638 -0.03679 0.15251 1.20553   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 0.26729 0.03721 7.184 1.68e-11 \*\*\*  
## Income 0.71449 0.04219 16.934 < 2e-16 \*\*\*  
## Production 0.04589 0.02588 1.773 0.0778 .   
## Unemployment -0.20477 0.10550 -1.941 0.0538 .   
## Savings -0.04527 0.00278 -16.287 < 2e-16 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 0.3286 on 182 degrees of freedom  
## Multiple R-squared: 0.754, Adjusted R-squared: 0.7486   
## F-statistic: 139.5 on 4 and 182 DF, p-value: < 2.2e-16

cbind(Data=uschange[,"Consumption"], Fitted=fitted(fit.consMR)) %>%  
 as.data.frame() %>%  
 ggplot(aes(x=Data, y=Fitted)) +  
 geom\_point() +  
 xlab("Fitted (predicted values)") +  
 ylab("Data (actual values)") +  
 ggtitle("Percentage change in US consumption expenditure") +  
 geom\_abline(intercept=0, slope=1)



Actual US consumption expenditure plotted against predicted US consumption expenditure.

### Goodness of fit

A common way to summarise how well a linear regression model fits the data is via the coefficient of determination, or . This can be calculated as the square of the correlation between the observed values and the predicted values. Alternatively, it can also be calculated as,

where the summations are over all observations. Thus, it reflects the proportion of variation in the forecast variable that is accounted for (or explained) by the regression model.

In simple linear regression, the value of is also equal to the square of the correlation between and (provided an intercept has been included).

If the predictions are close to the actual values, we would expect to be close to 1. On the other hand, if the predictions are unrelated to the actual values, then (again, assuming there is an intercept). In all cases, lies between 0 and 1.

The value is commonly used, often incorrectly, in forecasting. will never decrease when adding an extra predictor to the model and this can lead to over-fitting. There are no set rules for what is a good value, and typical values of depend on the type of data used. Validating a model’s forecasting performance on the test data is much better than measuring the value on the training data.

#### Example

Recall the example of US consumption data discussed in Section @ref(Regr-LSprinciple). The output for the estimated model is reproduced here for convenience:

fit.consMR <- tslm(Consumption ~ Income + Production + Unemployment + Savings,  
 data=uschange)  
summary(fit.consMR)

##   
## Call:  
## tslm(formula = Consumption ~ Income + Production + Unemployment +   
## Savings, data = uschange)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -0.88296 -0.17638 -0.03679 0.15251 1.20553   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 0.26729 0.03721 7.184 1.68e-11 \*\*\*  
## Income 0.71449 0.04219 16.934 < 2e-16 \*\*\*  
## Production 0.04589 0.02588 1.773 0.0778 .   
## Unemployment -0.20477 0.10550 -1.941 0.0538 .   
## Savings -0.04527 0.00278 -16.287 < 2e-16 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 0.3286 on 182 degrees of freedom  
## Multiple R-squared: 0.754, Adjusted R-squared: 0.7486   
## F-statistic: 139.5 on 4 and 182 DF, p-value: < 2.2e-16

## [1] 0.8683274

Figure @ref(fig:usfitted2) plots the actual consumption expenditure values versus the fitted values. The correlation between these variables is hence % (shown in the output above). In this case model does an excellent job as it explains most of the variation in the consumption data. Compare that to the value of 16% obtained from the simple regression with the same data set in Section @ref(Regr-Intro). Adding the three extra predictors has allowed a lot more of the variance in the consumption data to be explained.

### Standard error of the regression

Another measure of how well the model has fitted the data is the standard deviation of the residuals, which is often known as the “residual standard error”. This is shown in the above output with the value 0.3286.

It is calculated by

Notice that here we divide by in order to account for the number of estimated parameters in computing the residuals. Normally, we only estimate the mean (i.e., one parameter) when computing a standard deviation. The divisor is always minus the number of parameters estimated in the calculation. Here, we divide by because we have estimated parameters (the intercept and a coefficient for each predictor variable) in computing the residuals.

The standard error is related to the size of the average error that the model produces. We can compare this error to the sample mean of or with the standard deviation of to gain some perspective on the accuracy of the model.

We should warn here that the evaluation of the standard error can be highly subjective as it is scale dependent. The main reason we introduce it here is that it is required when generating prediction intervals, discussed in Section @ref(Regr-ForeWithRegr).

## Evaluating the regression model

The difference between the observied values and the corresponding fitted values are the trainig set errors or “residuals” defined as,

for . Each residual is the unpredictable component of the associated observation.

The residuals have some useful properties including the following two:

As a result of these properties, it is clear that the average of the residual is zero, and that the correlation between the residuals and the observations for the predictor variable is also zero. (This is not necessarily true when the intercept is omitted from the model.)

After selecting the regression variables and fitting a regression model, it is necessary to plot the residuals to check that the assumptions of the model have been satisfied. There are a series of plots that should be produced in order to check different aspects of the fitted model and the underlying assumptions.

### ACF plot of residuals

With time series data, it is highly likely that the value of a variable observed in the current time period will be similar to its value in the previous period, or even the period before that, and so on. Therefore when fitting a regression model to time series data, it is very common to find *autocorrelation* in the residuals. In this case, the estimated model violates the assumption of no autocorrelation in the errors, and our forecasts may be inefficient — there is some information left over which should be accounted for in the model in order to obtain better forecasts. The forecasts from a model with autocorrelated errors are still unbiased, and so are not “wrong”, but they will usually have larger prediction intervals than they need to. Therefore we should always look at an ACF plot of the residuals.

Another useful test of *autocorrelation* in the residuals designed to take account for the regression model is the **Breusch-Godfrey** test, also referred to as the LM (Lagrange Multiplier) test for *serial correlation*. It is used to test the joint hypothesis that there is no autocorrelation in the residuals up to a certain specified order. A small p-value indicates there is significant autocorrelation remaining in the residuals.

The Breusch-Godfrey test is similar to the Ljung-Box test, but it is specifically designed for use with regression models.

### Histograms of residuals

It is always a good idea to check if the residuals are normally distributed. As explained earlier, this is not essential for forecasting, but it does make the calcultion of prediction intervals much easier.

#### Example

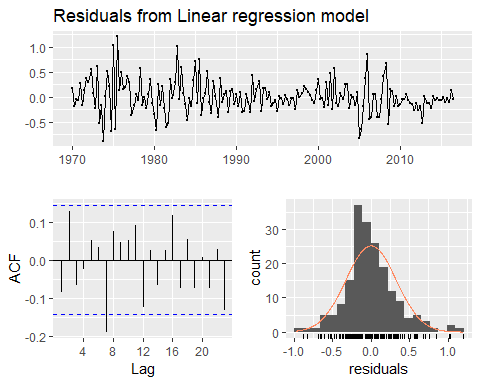
Using the checkresiduals function, introduced in Section @ref(residuals), we can obtain all the useful residual diagnostics mentioned above. Figure @ref(fig:beerresidcheck) shows a time plot, the ACF and the histogram of the residuals from the model fitted to the beer production data as well as the Breusch-Godfrey test for testing up to 8th order autocorrelation. (The checkresiduals function will use the *Breusch-Godfrey* test for regression models, but the *Ljung-Box* test otherwise.)

The time plot shows a large negative value in the residuals at 2004 Q4, which suggests that something unusual may have happened in that quarter. It would be worth investigating to see if there were any unusual circumstances or events which may have significantly reduced beer production for that quarter.

The remaining residuals show that the model has captured the patterns in the data quite well. However, there is a small amount of autocorrelation left in the residuals, seen in the significant spike in the ACF plot but not detected to be significant by the Breusch-Godfrey test. This suggests that there may be some information remaining in the residuals and the model can be slightly improved to capture this. However, it is unlikely that it will make much difference to the resulting forecasts. The forecasts from the current model are still unbiased, but will have larger prediction intervals than they need to. In Chapter @ref(ch-dynamic) we discuss dynamic regression models used for better capturing information left in the residuals.

The histogram shows that the residuals from modelling the beer data seem to be slightly negatively skewed, although that is due to the large negative value in 2004 Q4.

checkresiduals(fit.consMR)



Analysing the residuals from a regression model for beer production.

##   
## Breusch-Godfrey test for serial correlation of order up to 8  
##   
## data: Residuals from Linear regression model  
## LM test = 14.874, df = 8, p-value = 0.06163

Figure 5.8 shows a time plot, the ACF and the histogram of the residuals from the multiple regression model fitted to the US quarterly consumption data, as well as the Breusch-Godfrey test for jointly testing up to 8th order autocorrelation. (The checkresiduals() function will use the *Breusch-Godfrey* test for regression models, but the Ljung-Box test otherwise.)

The timeplot shows some changin variation over time, but is otherwise relatively unremakable. This *heteroschedasticity* will potentially make potentially make the prediction interval coverage inaccurate.

The histogram shows that the residuals seem to be slightly *skewed*, which may also affect the coverage probability of the prediction intervals.

The autocorrelation plot shows a significant spike at lag 7, but it is not quite enough for the \***Breusch-Godfrey** to be significantly at the 5% level. In any case, the autocorrelation is not particularly large, and lag 7 it is unlikely to have any noticeable impact on the forecasts or the prediction intervals. In Chapter 9, we discuss dynamic regression modesl used for better capturing information left in the residuals.

### Residual plots against predictors

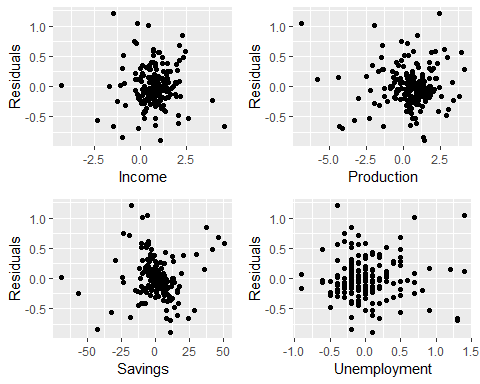
We would expect the residuals to be randomly scattered without showing any systematic patterns. A simple and quick way to check this is to examine *scatterplots* of the residuals against each of the predictor variables. If these scatterplots show a pattern, then the relationship may be nonlinear and the model will need to be modified accordingly. See Section 5.8 for a discussion of nonlinear regression.

It is also necessary to plot the residuals against any predictors that are not in the model. If any of these *show a pattern*, then the corresponding predictor may need to be added to the model (possibly in a nonlinear form).

#### Example

The residuals from the multiple regression model for forecasting US consumption plotted against each predictor in Figure 5.9 seem to be randomly scattered. Therefore we are satisfied with these in this case.

df <- as.data.frame(uschange)  
df[,"Residuals"] <- as.numeric(residuals(fit.consMR))  
  
p1 <- ggplot(df,aes(Income,Residuals))+geom\_point()  
p2 <- ggplot(df,aes(Production,Residuals))+geom\_point()  
p3 <- ggplot(df,aes(Savings, Residuals))+geom\_point()  
p4 <- ggplot(df,aes(Unemployment, Residuals))+geom\_point()  
  
gridExtra::grid.arrange(p1,p2,p3,p4, nrow=2)



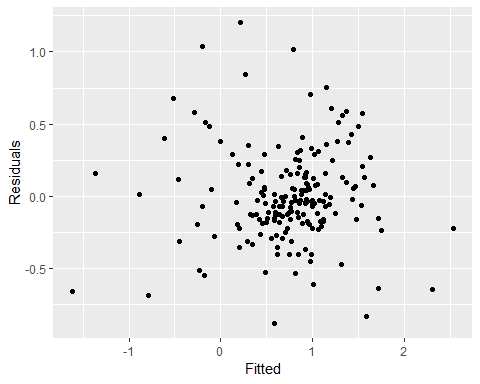
### Residual plots against fitted values

A plot of the residuals against the fitted value should also show no pattern. If a pattern is observed, there may be “heteroskedasticity” in the errors which means that the variance of the residuals may not be constant. If this problem occurs, a transformation of the forecast variable such as a logarithm or square root may be required.

#### Example

Continuing the previous example, Figure 5.10 shows the residuals plotted against the fitted values. The random scatter suggests the errors are **homoscedastic**.

cbind(Fitted=fitted(fit.consMR),  
 Residuals=residuals(fit.consMR)) %>%   
 as.data.frame() %>%   
 ggplot(aes(Fitted,Residuals))+geom\_point()



### Outliers and influential observations

Observations that take extreme values compared to the majority of the data are called **outliers**. Observations that have a large influence on the estimated coefficients of a regression model are called **influential observations**. Usually, influential observations are also outliers that are extreme in the direction.

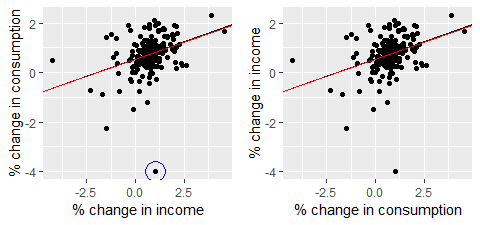
There are formal methods for detecting outliers and influential observations that are beyond the scope of this book. As we suggested at the beginning of Chapter 2, becoming faimilar with your data prior to perfoming any analysis of vital importance. A scatter plot of against each is always a useful point in regression analysis, and often helps to identify usual observations.

One source of outliers is *incorrect data entry*. Simple descriptive statistics of your data can identify minima and maxima that are sensible. If such an observation is identified, and it has been recored incorrectly, it should be corrected or removed from the sample immediately.

Outliers also occur when some observations are simply different. In this case it may not be wise for these observations to be removed. IF an observation has been identified as a likely outlier, it is important to study it and analyze the possible reasons behind it. The decision to remove or retain an observation can be a challenging one (especially when outliers are influential observations). It is wise to report results both with and without the removal of such observations.

#### Example

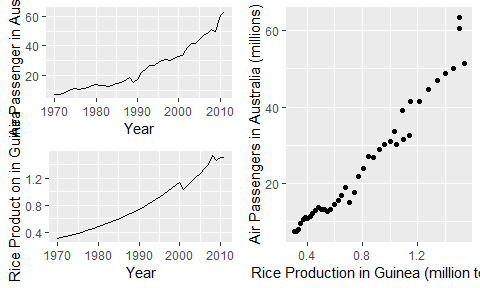
Figure @ref(fig:outlier) highlights the effect of a single outlier when regressing US consumption on income (the example introduced in Section @ref(Regr-Intro)). In the left panel the outlier is only extreme in the direction of as the percentage change in consumption has been incorrectly recorded as -4%. The red line is the regression line fitted to the data which includes the outlier, compared to the black line which is the line fitted to the data without the outlier. In the right panel the outlier now is also extreme in the direction of with the 4% decrease in consumption corresponding to a 6% increase in income. In this case the outlier is very influential as the red line now deviates substantially from the black line.



The effect of outliers and influential observations on regression

### Spurious regression

More often than not, time series data are **“non-stationary”**; that is, the values of the time series do not fluctuate around a constant mean or with a constant variance. We will deal with time series stationarity in more detail in Chapter 8, but here we need to address the effect that non-stationary data can have on regression models.

For example consider the two variables plotted below in Figure @ref(fig:spurious). These appear to be related simply because they both trend upwards in the same manner. However, air passenger traffic in Australia has nothing to do with rice production in Guinea. 

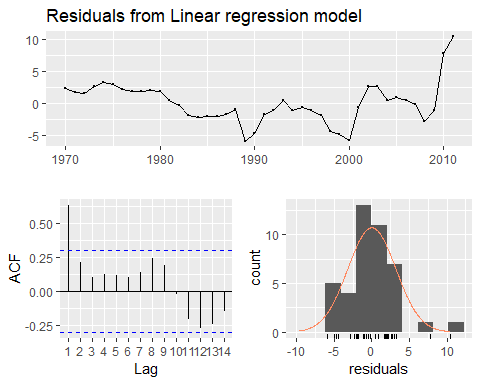
Regressing *non-stationary* time series can lead to **spurious regressions**. The output of regressing Australian air passengers on rice production in Guinea is shown below. High and **high residual autocorrelation** can be signs of **spurious regression**. We discuss the issues surrounding non-stationary data and spurious regressions in more detail in Chapter @ref(ch-dynamic).

Cases of spurious regression might appear to give reasonable short-term forecasts, but they will generally not continue to work into the future.

aussies <- window(ausair, end=2011)  
fit <- tslm(aussies~guinearice)  
summary(fit)

##   
## Call:  
## tslm(formula = aussies ~ guinearice)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -5.9448 -1.8917 -0.3272 1.8620 10.4210   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) -7.493 1.203 -6.229 2.25e-07 \*\*\*  
## guinearice 40.288 1.337 30.135 < 2e-16 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 3.239 on 40 degrees of freedom  
## Multiple R-squared: 0.9578, Adjusted R-squared: 0.9568   
## F-statistic: 908.1 on 1 and 40 DF, p-value: < 2.2e-16

checkresiduals(fit)



##   
## Breusch-Godfrey test for serial correlation of order up to 8  
##   
## data: Residuals from Linear regression model  
## LM test = 28.813, df = 8, p-value = 0.000342

## Selection predictors

When there are many possible predictors, we need some strategy to select the best predictors to use in a regression model.

A common approach that is *not recommended* is to plot the forecast variable against a particular predictor and if it shows no noticeable relationship, drop it. This is invalid because it is not always possible to see the relationship from a scatterplot, especially when the effects of other predictors have not been accounted for.

Another common approach which is also invalid is to do a multiple linear regression on all the predictors and disregard all variables whose -values are greater than 0.05. To start with, statistical significance does not always indicate predictive value. Even if forecasting is not the goal, this is not a good strategy because the -values can be misleading when two or more predictors are correlated with each other (see Section @ref(Regr-MultiCol)).

Instead, we will use a measure of predictive accuracy. Five such measures are introduced in this section.

## Some useful predictors

There are several useful predictors that occur frequently when using regression for time series data.

### Trend

It is very common for time series data to be trending. A linear trend can be modelled by simply using as a predictor,

where . A trend variable can be specified in the tslm function using the trend predictor. In Section @ref(Regr-NonLinear) we discuss how we can also model a nonlinear trends.

### Dummy variables

So far, we have assumed that each predictor takes numerical values. But what about when a predictor is a categorical variable taking only two values (e.g., “yes” and “no”). Such a variable might arise, for example, when forecasting daily sales and you want to take account of whether the day is a **public holiday** or not. So the predictor takes value “yes” on a public holiday, and “no” otherwise.

This situation can still be handled within the framework of multiple regression models by creating a “dummy variable” taking value 1 corresponding to “yes” and 0 corresponding to “no”. A dummy variable is also known as an “indicator variable”.

A dummy variable can also be used to account for an **outlier** in the data. Rather than omit the outlier, a dummy variable removes its effect. In this case, the dummy variable takes value 1 for that observation and 0 everywhere else. An example is the case where a special event has occurred. For example when forecasting tourist arrivals to Brazil, we will need to account for the effect of the Rio de Janeiro summer Olympics in 2016.

If there are more than two categories, then the variable can be coded using several dummy variables (one fewer than the total number of categories). tslm will automatically handle this case if you specify a factor variable as a predictor. There is usually no need to manually create the corresponding dummy variables.

### Seasonal dummy varabiles

For example, suppose we are forecasting daily electricity demand and we want to account for the day of the week as a predictor. Then the following dummy variables can be created.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |
| Monday | 1 | 0 | 0 | 0 | 0 | 0 |
| Tuesday | 0 | 1 | 0 | 0 | 0 | 0 |
| Wednesday | 0 | 0 | 1 | 0 | 0 | 0 |
| Thursday | 0 | 0 | 0 | 1 | 0 | 0 |
| Friday | 0 | 0 | 0 | 0 | 1 | 0 |
| Saturday | 0 | 0 | 0 | 0 | 0 | 1 |
| Sunday | 0 | 0 | 0 | 0 | 0 | 0 |
| Monday | 1 | 0 | 0 | 0 | 0 | 0 |
| Tuesday | 0 | 1 | 0 | 0 | 0 | 0 |
| Wednesday | 0 | 0 | 1 | 0 | 0 | 0 |
| Thursday | 0 | 0 | 0 | 1 | 0 | 0 |
| Friday | 0 | 0 | 0 | 0 | 1 | 0 |
| ⋮ | ⋮ | ⋮ | ⋮ | ⋮ | ⋮ | ⋮ |

Notice that only six dummy variables are needed to code seven categories. That is because the seventh category (in this case Sunday) is specified when all dummy variables are all set to zero and is captured by the intercept.

Many beginners will try to add a seventh dummy variable for the seventh category. This is known as the “dummy variable trap” because it will cause the regression to fail. There will be one too many parameters to estimate when an intercept is also included. The general rule is to use one fewer dummy variables than categories. So for quarterly data, use three dummy variables; for monthly data, use 11 dummy variables; and for daily data, use six dummy variables, and so on.

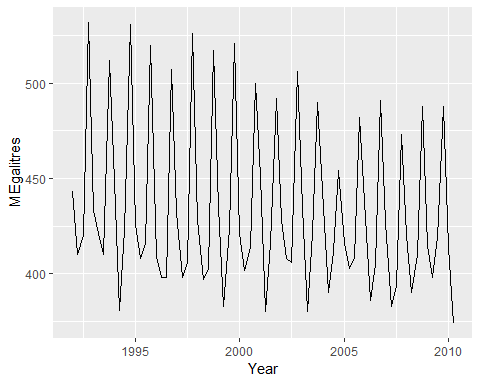
The interpretation of each of the coefficients associated with the dummy variables is that it is *a measure of the effect of that category relative to the omitted category*. In the above example, the coefficient of associated with Monday will measure the effect of Monday compared to Sunday on the forecast variable. An example of interpreting estimated dummy variable coefficients capturing the quarterly seasonality of Australian beer production follows.

The tslm function will automatically handle this situation if you specify the predictor season.

#### Example Australian quarterly beer production

Recall the Australian quarterly beer production data shown again in Figure @ref(fig:beeragain) below.

beer2 <- window(ausbeer,start=1992)  
autoplot(beer2)+xlab("Year")+ylab("MEgalitres")



Australian quarterly beer production.

We want to forecast the value of future beer production. We can model this data using a regression model with a linear trend and quarterly dummy variables.

where if is in quarter and 0 otherwise. The first quarter variable has been omitted, so the coefficients associated with the other quarters are measures of the difference between those quarters and the first quarter.

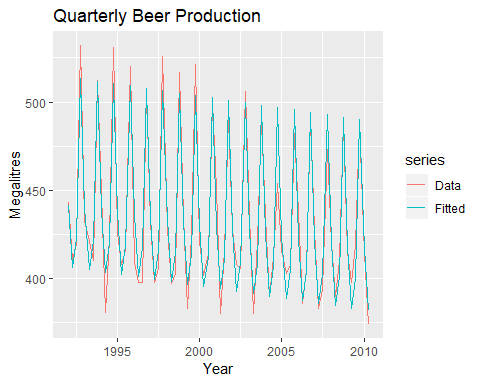
fit.beer <- tslm(beer2 ~ trend + season)  
summary(fit.beer)

##   
## Call:  
## tslm(formula = beer2 ~ trend + season)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -42.903 -7.599 -0.459 7.991 21.789   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 441.80044 3.73353 118.333 < 2e-16 \*\*\*  
## trend -0.34027 0.06657 -5.111 2.73e-06 \*\*\*  
## season2 -34.65973 3.96832 -8.734 9.10e-13 \*\*\*  
## season3 -17.82164 4.02249 -4.430 3.45e-05 \*\*\*  
## season4 72.79641 4.02305 18.095 < 2e-16 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 12.23 on 69 degrees of freedom  
## Multiple R-squared: 0.9243, Adjusted R-squared: 0.9199   
## F-statistic: 210.7 on 4 and 69 DF, p-value: < 2.2e-16

Note that trend and season are not objects in the R workspace; they are created automatically by tslm when specified in this way.

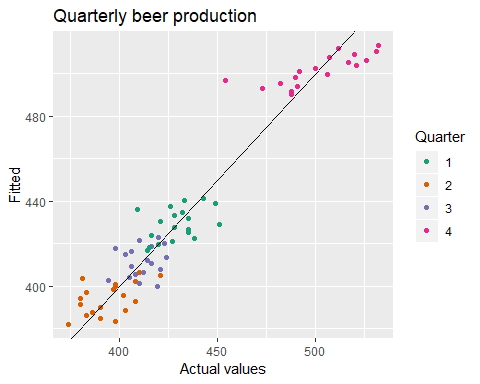
There is a downward trend of -0.34 megalitres per quarter. On average, the second quarter has production of 34.7 megalitres lower than the first quarter, the third quarter has production of 17.8 megalitres lower than the first quarter, and the fourth quarter has production of 72.8 megalitres higher than the first quarter.

autoplot(beer2, series="Data")+  
 autolayer(fitted(fit.beer),series="Fitted")+  
 xlab("Year")+ylab("Megalitres")+  
 ggtitle("Quarterly Beer Production")



Time plot of beer production and predicted beer production.

cbind(Data=beer2, Fitted=fitted(fit.beer)) %>%   
 as.data.frame() %>%   
 ggplot(aes(x=Data,y=Fitted, colour=as.factor(cycle(beer2))))+  
 geom\_point()+  
 ylab("Fitted")+xlab("Actual values")+  
 ggtitle("Quarterly beer production")+  
 scale\_color\_brewer(palette="Dark2",name="Quarter")+  
 geom\_abline(intercept=0, slope=1)



Actual beer production plotted against predicted beer production.

### Intervention variables

It is often necessary to model interventions that may have affected the variable to be forecast. For example, competitor activity, advertising expenditure, industrial action, and so on, can all have an effect.

When the effect lasts only for one period, we use a spike variable. This is a dummy variable taking value one in the period of the intervention and zero elsewhere. A spike variable is equivalent to a dummy variable for handling an outlier.

Other interventions have an immediate and permanent effect. If an intervention causes a level shift (i.e., the value of the series changes suddenly and permanently from the time of intervention), then we use a step variable. A step variable takes value zero before the intervention and one from the time of intervention onward.

Another form of permanent effect is a change of slope. Here the intervention is handled using a piecewise linear trend; a trend that bends at the time of intervention and hence is nonlinear. We will discuss this in Section @ref(Regr-NonLinear).

### Trading days

The number of trading days in a month can vary considerably and can have a substantial effect on sales data. To allow for this, the number of trading days in each month can be included as a predictor.

For monthly or quarterly data, the bizdays function will compute the number of trading days in each period.

An alternative that allows for the effects of different days of the week has the following predictors:

### Distributed lags

It is often useful to include advertising expenditure as a predictor. However, since the effect of advertising can last beyond the actual campaign, we need to include lagged values of advertising expenditure. So the following predictors may be used.

It is common to require the coefficients to decrease as the lag increases, although this is beyond the scope of this book.

### Easter

Easter is different from most holidays because it is not held on the same date each year and the effect can last for several days. In this case, a dummy variable can be used with value one where the holiday falls in the particular time period and zero otherwise.

For example, with monthly data, when Easter falls in March then the dummy variable takes value 1 in March, when it falls in April, the dummy variable takes value 1 in April, and when it starts in March and finishes in April, the dummy variable is split proportionally between months.

The easter function will compute the dummy variable for you.

### Fourier series

As alternative to using seasonal dummy variables, especially for long seasonal periods, is to use **Fourier terms**. Jean-Baptiste Fourier was a French mathmatician, born in the 1700s, who showed that a series of sine and cosine terms of the right frequencies can approximate any periodic function. We can use them for seasonal patterns.

If is the seasonal period, then the first few Fourier terms are given by

$$
x\_{1,t} = \sin\left(\textstyle\frac{2\pi t}{m}\right),
x\_{2,t} = \cos\left(\textstyle\frac{2\pi t}{m}\right),
x\_{3,t} = \sin\left(\textstyle\frac{4\pi t}{m}\right),
x\_{4,t} = \cos\left(\textstyle\frac{4\pi t}{m}\right),
x\_{5,t} = \sin\left(\textstyle\frac{6\pi t}{m}\right),
x\_{6,t} = \cos\left(\textstyle\frac{6\pi t}{m}\right),
$$

and so on. If we have monthly seasonality, and we use the first 11 of these predictor variables, then we will get exactly the same forecasts as using 11 dummy variables.

The advantage of using Fourier terms is that we can often use fewer predictor variables than we need to with dummy variables, especially when is large. This makes them useful for weekly data, for example, where . For short seasonal periods such as with quarterly data, there is little advantage in using Fourier series over seasonal dummy variables.

In R, these Fourier terms are produced using the fourier function. For example, the Australian beer data can be modelled like this.

fourier.beer <- tslm(beer2~trend+fourier(beer2, K=2))  
summary(fourier.beer)

##   
## Call:  
## tslm(formula = beer2 ~ trend + fourier(beer2, K = 2))  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -42.903 -7.599 -0.459 7.991 21.789   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 446.87920 2.87321 155.533 < 2e-16 \*\*\*  
## trend -0.34027 0.06657 -5.111 2.73e-06 \*\*\*  
## fourier(beer2, K = 2)S1-4 8.91082 2.01125 4.430 3.45e-05 \*\*\*  
## fourier(beer2, K = 2)C1-4 53.72807 2.01125 26.714 < 2e-16 \*\*\*  
## fourier(beer2, K = 2)C2-4 13.98958 1.42256 9.834 9.26e-15 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 12.23 on 69 degrees of freedom  
## Multiple R-squared: 0.9243, Adjusted R-squared: 0.9199   
## F-statistic: 210.7 on 4 and 69 DF, p-value: < 2.2e-16

The first argument to fourier just allows it to identify the seasonal period and the length of the predictors to return. The second K argument specifies how many pairs of sin and cos terms to include. The maximum number allowed is where is the seasonal period. Because we have used the maximum here, the results are identical to those obtained when using seasonal dummy variables.

If only the first two Fourier terms are used ( and ), the seasonal pattern will follow a simple sine wave. A regression model containing Fourier terms is often called a **harmonic regression** because the successive Fourier terms represent harmonics of the first two Fourier terms.

## Selecting predictors

When there are many possible predictors, we need some strategy for selecting the best predictors to use in a regression model.

A common approach that is *not recommended* is to plot the forecast variable against a particular predictor and if there is no noticebable relationship, drop that predictor from the model. This is invalid because it is not always possible to see the relationship from a scatter plot, especially when the effects of other predictors have not been accounted for.

Another common approach which is also invalid is to do a multiple linear regression on all the predictors and disregard all variables whose *p* values are greater than 0.05. To start with, statistical sigfinicance does not always indicate predictive value. Even if forecasting is not the goal, this is not a good strategy because the -value can be misleading when two or more predictors are correlated with each other (see Section 5.9).

Instead, we will use a measure of predictive accuracy. Five such measures are introduced in this section. They can be calculated using the cv() function, here applied to the model for US consumption:

CV(fit.consMR)

## CV AIC AICc BIC AdjR2   
## 0.1163477 -409.2980298 -408.8313631 -389.9113781 0.7485856

We compare these values against the corresponding values from other models. For the CV, AIC, AICc and BIC measures, we want to find the model with the lowest value; for adjusted , we seek the model with the highest value.

### Adjusted

Computer output for a regression will always give the value, discussed in Section 5.2. However, it is not a good measure of the predictive ability of a model. Imagine a model which produces forecats that are exactly 20% of the actual values. In that case, the value would be 1 (indicating perfect correlation), but the forecasts are not very close to the actual values.

In addition, does not allow for “degrees of freedom”. Adding *any* variable tends to increase the value of , even if that variable is irrelevant. For these reasons, forecasters should not use to determine whether a model will give good predictions.

An equivalent idea is to select the model which gives the minimum sum of squared errors (SSE), given by

Minimizing the SSE is equivalent to maximizing and will always choose the model with the most variables, and so is not a valid way of selecting predictors.

An alternative, designed to overcome these problems, is the adjusted (also called “R-bar-squared”):

where is the number of observations and is the number of predictors. This is an improvement on as it will no longer increase with each added predictor. Using this measure, the best model will be the one with the largest value of . Maximizing is equivalent to minimizing the standard error given in Equation @ref(#eq:Regr-se).

Maximizing works quite well as a method of selecting predictors, although it does tend to err on the side of selecting too many predictors.

### Cross-validation

Section @ref(accuracy) introduced **time series cross-validation** as a general and useful tool for determining the predictive ability of a model. For regression models, it is also possible to use classical *leave-one-out cross-validation* to selection predictors [@BHK15]. This is faster and makes more efficient use of the data. The procedure uses the following steps:

1. Remove observation from the data set, and fit the model using the remining data. Then compute the error for the omitted observation (This is not the same as the residual because the th observation are not used in estimating the value of )
2. Repeat step1 for .
3. Compute the MSE from . We shall call this the **CV**.

Although this looks like a time-consuming procedure, there are fast methods of calculating CV, so that it takes no longer than fitting one model to the full data set. The equation for computing CV efficiently is given in Section 5.7. Under this criterion, the best model is the one with the smallest value of CV.

### Akaike’S information criterion

A closely-related method os Akaike’s information citerion, which we define as

where is the number of observations used for estimation and is the number of predictors in the model. Different computer packages use slightly different definitions for the AIC, although they should all lead to the same model being selected. The part of the equation occurs because there are parameters in the model: the coefficients for the predictors, the intercept and the variance of the residuals. The idea here is to penalize the fit of the model (SSE) with the number of parameters that need to be estimated.

The model with the minimum value of the AIC is often the best model for forecasting. For large values of , minimizing the AIC is equivalent to minimizing the CV value.

### Corrected Akaike’s information criterion

For small values of , the AIC tends to select too many predictors, and so a bias-corrected version of the AIC has been developed,

As with the AIC, the AICc should be minimized.

### Schwarz bayesian information criterion

A related measure is Schwarz’s Bayesian Information Criterion (known as SBIC, BIC or SC),

As with the AIC, minimizing the BIC is intended to give the best model. The model chosen by BIC is either the same as that chosen by AIC, or one with fewer terms. This is because the BIC penalizes the number of parameters more heavily than the AIC. For large values of , minimizing BIC is similar to leave--out cross-validation when .

### Which measure should we use

While is widely used, and has been around longer than the other measures, its tendency to select too many predictor variables makes it less suitable for forecasting.

Many statisticians like to use the **BIC** because it has the feature that if there is a true underlying model, the BIC will select that model given enough data. However, in reality, there is rarely, if ever, a true underlying model, and even if there was a true underlying model, selecting that model will not necessarily give the best forecasts (because the parameter estimates may not be accurate).

Consequently, we recommend that one of the AICc, AIC or CV statistics be used, each of which has forecasting as their objective. If the value of is large enough, they will all lead to the same model. In most of the examples in this book, we use the AIcc value to select the forecasting model.

#### Example: US consumption

To obtain all these measures in R, use CV(fit). In the multiple regression example for forecasting US consumption we considered four predictors. With four predictors, there are possible models. Now we can check if all four predictors are actually useful, or whether we can drop one or more of them. All 16 models were fitted and the results are summarised below in Table @ref(tab:tblusMR). A “1” indicates that the predictor was included in the model, and a “0” means that the predictor was not included in the model. Hence the first row shows the measures of predictive accuracy for a model including all four predictors.

The results have been sorted according to the AICc and therefore the best models are given at the top of the table, and the worst at the bottom of the table.

All 16 possible models for forecasting US consumption with 4 predictors.

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Income | Production | Savings | uneployment | CV | AIC | AICc | BIC | AdjR2 |
| 1 | 1 | 1 | 1 | 0.116 | -409.298 | -408.831 | -389.911 | 0.749 |
| 1 | 0 | 1 | 1 | 0.116 | -408.094 | -407.763 | -391.939 | 0.746 |
| 1 | 1 | 1 | 0 | 0.118 | -407.467 | -407.135 | -391.311 | 0.745 |
| 1 | 0 | 1 | 0 | 0.129 | -388.727 | -388.507 | -375.803 | 0.716 |
| 1 | 1 | 0 | 1 | 0.278 | -243.164 | -242.832 | -227.008 | 0.386 |
| 1 | 0 | 0 | 1 | 0.283 | -237.928 | -237.708 | -225.003 | 0.365 |
| 1 | 1 | 0 | 0 | 0.289 | -236.125 | -235.906 | -223.201 | 0.359 |
| 0 | 1 | 1 | 1 | 0.293 | -234.373 | -234.042 | -218.218 | 0.356 |
| 0 | 1 | 1 | 0 | 0.300 | -228.942 | -228.722 | -216.018 | 0.334 |
| 0 | 1 | 0 | 1 | 0.303 | -226.298 | -226.078 | -213.374 | 0.324 |
| 0 | 0 | 1 | 1 | 0.306 | -224.575 | -224.355 | -211.650 | 0.318 |
| 0 | 1 | 0 | 0 | 0.314 | -219.627 | -219.496 | -209.934 | 0.296 |
| 0 | 0 | 0 | 1 | 0.314 | -217.677 | -217.546 | -207.984 | 0.288 |
| 1 | 0 | 0 | 0 | 0.372 | -185.438 | -185.307 | -175.744 | 0.154 |
| 0 | 0 | 1 | 0 | 0.414 | -164.135 | -164.004 | -154.442 | 0.052 |
| 0 | 0 | 0 | 0 | 0.432 | -155.051 | -154.985 | -148.588 | 0.000 |

The best model contains all four predictors. However, a closer look at the results reveals some interesting features. There is clear separation between the models in the first four rows and the ones below. This indicates that Income and Savings are both more important variables than Production and Unemployment. Also, the first two rows have almost identical values of *CV*, *AIC* and *AICc*. So we could possibly drop the Production variable and get very similar forecasts. Note that Production and Unemployment are highly (negatively) correlated, as shown in Figure @ref(fig:ScatterMatrix), so most of the predictive information in Production is also contained in the Unemployment variable.

### Best subset regression

Where possible, all potential regression models should be fitted (as was done in the above example) and the best model should be selected based on one of the measures discussed. This is known as “best subsets” regression or “all possible subsets” regression.

It is recommended that one of CV, AIC or AICc be used for this purpose. If the value of is large enough, they will all lead to the same model.

While is very widely used, and has been around longer than the other measures, its tendency to select too many predictor variables makes it less suitable for forecasting than either CV, AIC or AICc. Also, the tendency of BIC to select too few variables makes it less suitable for forecasting than either CV, AIC or AICc.

### Stepwise regression

If there are a large number of predictors, it is not possible to fit all possible models. For example, 40 predictors leads to 1 trillion possible models! Consequently, a strategy is required to limit the number of models to be explored.

An approach that works quite well is *backwards stepwise regression*:

* Start with the model containing all potential predictors.
* Remove one predictor at a time. Keep the model if it improves the measure of predictive accuracy.

If the number of potential predictors is too large, then the backwards stepwise regression will not work and *forward stepwise regression* can be used instead. This procedure starts with a model that includes only the intercept. Predictors are added one at a time, and the one that most improves the measure of predictive accuracy is retained in the model. The procedure is repeated until no further improvement can be achieved.

Alternatively for either the backward or forward direction, a starting model can be one that includes a subset of potential predictors. In this case, an extra step needs to be included. For the backwards procedure we should also consider adding a predictor with each step, and for the forward procedure we should also consider dropping a predictor with each step. These are referred to as *hybrid* procedures.

It is important to realise that any stepwise approach is not guaranteed to lead to the best possible model, but it almost always leads to a good model. For further details see [@ISLR.](<https://www-bcf.usc.edu/~gareth/ISL/>)

### Beware of inference after selecting predictors

We do not discuss statistical inference of the predictors in this book (e.g., looking at -values associated with each predictor). If you do wish to look at the statistical significance of the predictors, beware that *any* procedure involving selecting predictors first will invalidate the assumptions behind the -values. The procedures we recommend for selecting predictors are helpful when the model is used for forecasting; they are not helpful if you wish to study the effect of any predictor on the forecast variable.

## Forecasting with regression

Recall that predictoons of can be obtained using

which comprises the estimated coefficients and ignores the error in the regression equation. Plugging in the values of the predictor variables for returned the fitted (training-sample) values of . What we are interested in here is forecasting *future* values of .

### Ex-ante versus ex-post forecasts

When using regression models for time series data, we need to distinguish between the different types of forecasts that can be produced, depending on what is assumed to be known when the forecasts are computed.

*Ex ante forecasts* are those that are made using only the information that is available in advance. For example, ex-ante forecasts for the percentage change in US consumption for quarters following the end of the sample, should only use information that was available *up to and including* 2016 Q3. These are genuine forecasts, made in advance using whatever information is available at the time. Therefore in order to generate ex-ante forecasts, the model requires future values (forecasts) of the predictors. To obtain these we can use one of the simple methods introduced in Section @ref(sec-2-methods) or more sophisticated pure time series approaches that follow in Chapters @ref(ch-expsmooth) and @ref(ch-arima). Alternatively, forecasts from some other source, such as a government agency, may be available and can be used.

*Ex post forecasts* are those that are made using later information on the predictors. For example, ex post forecasts of consumption may use the actual observations of the predictors, once these have been observed. These are not genuine forecasts, but are useful for studying the behaviour of forecasting models.

The model fro mwhich ex-post forecasts are produced should not be estimated using data from the forecast period. That is, ex-post forecasts can assume knowledege of the predictor variables (the variables), but should not assume knowledge of the data that are to be forecast (the variable).

A comparative evaluation of ex-ante forecasts and ex-post forecasts can help to separate out the sources of forecast uncertainty. This will show whether forecast errors have arisen due to poor forecasts of the predictor or due to a poor forecasting model.

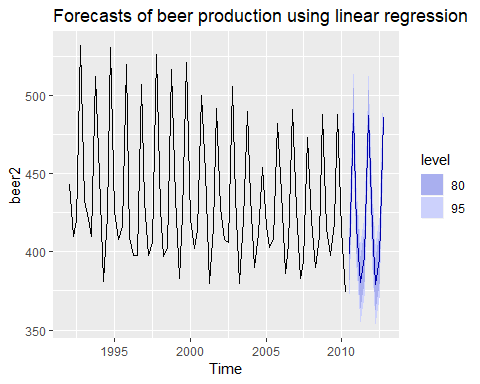
#### Example: Australian quarterly beer production [-]

Normally, we cannot use actual future values of the predictor variables when producing ex-ante forecasts because their values will not be known in advance. However, the special predictors introduced in Section @ref(Regr-UsefulPredictors) are all known in advance, as they are based on calendar variables (e.g., seasonal dummy variables or public holiday indicators) or deterministic functions of time (e.g. time trend). In such cases, there is no difference between ex ante and ex post forecasts.

beer2 <- window(ausbeer, start=1992)  
head(beer2)

## Qtr1 Qtr2 Qtr3 Qtr4  
## 1992 443 410 420 532  
## 1993 433 421

fit.beer <- tslm(beer2 ~ trend + season)  
fcast <- forecast(fit.beer)  
autoplot(fcast) +  
 ggtitle("Forecasts of beer production using linear regression")

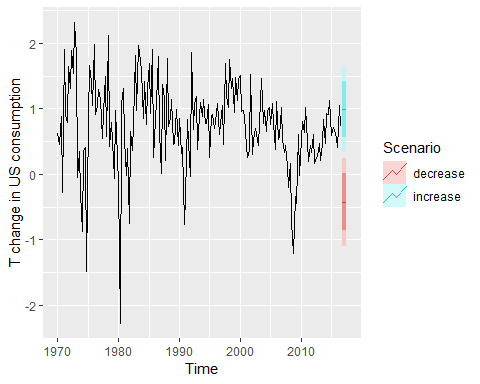


Forecasts from the regression model for beer production. The dark shaded region shows 80% prediction intervals and the light shaded region shows 95% prediction intervals.

### Scenario based forecasting

In this setting, the forecaster assumes possible scenarios for the predictor variables that are of interest. For example, a US policy maker may be interested in comparing the predicted change in consumption when there is a constant growth of 1% and 0.5% respectively for income and savings with no change in the employment rate, versus a respective decline of 1% and 0.5%, for each of the four quarters following the end of the sample. The resulting forecasts are calculated below and shown in Figure @ref(fig:ConsInc4). We should note that prediction intervals for scenario based forecasts do not include the uncertainty associated with the future values of the predictor variables. They assume that the values of the predictors are known in advance.

fit.consBest <- tslm(Consumption ~ Income + Savings + Unemployment, data = uschange)  
h <- 4  
  
newdata <-  
 cbind(Income=c(1,1,1,1), Savings=c(0.5,0.5,0.5,0.5), Unemployment=c(0,0,0,0)) %>%  
 as.data.frame()  
fcast.up <- forecast(fit.consBest, newdata=newdata)  
  
newdata <-  
 cbind(Income=rep(-1,h), Savings=rep(-0.5,h), Unemployment=rep(0,h)) %>%  
 as.data.frame()  
fcast.down <- forecast(fit.consBest, newdata=newdata)  
  
autoplot(uschange[,1])+ylab("T change in US consumption")+  
 forecast::autolayer(fcast.up, PI=TRUE,series="increase")+  
 autolayer(fcast.down,PI=TRUE,series="decrease")+  
 guides(colour=guide\_legend(title="Scenario"))



Forecasting percentage changes in personal consumption expenditure for the US under scenario based forecasting.

### Builing a predictive regression model

The great advantage of regression models is that they can be used to capture important relationships between the forecast variable of interest and the predictor variables. A major challenge however, is that in order to generate ex-ante forecasts the model requires future values of each predictor. If scenario based forecasting is of interest then these models are extremely useful. However, if ex-ante forecasting is the main focus, obtaining forecasts of the predictors can be very challenging (in many cases generating forecasts for the predictor variables can be more challenging than forecasting directly the forecast variable without using predictors).

An alternative formulation is to use as predictors their lagged values. Assuming that we are interested in generating a -step ahead forecast we write

for . The predictor set is formed by values of the s that are observed time periods prior to observing . Therefore when the estimated model is projected into the future, i.e., beyond the end of the sample , all predictor values are available.

Including lagged values of the predictors does not only make the model operational for easily generating forecasts, it also makes it intuitively appealling. For example, the effect of a policy change with the aim of increasing production may not have an instantaneous effect on consumption expenditure. It is most likely that this will happend with a lagging effect. We touched upon this in Section @ref(Regr-UsefulPredictors) when briefly introducing distributed lags as predictors. Several directions for generalising regression models to better incorporate the rich dynamics observed in time series are discussed In Section @ref(ch-dynamic).

### Prediction intervals

With each forecast for the change in consumption Figure @ref(fig:ConsInc4) 95% and 80% prediction intervals are also included. The general formulation of how to calculate prediction intervals for multiple regression models is presented in Section @ref(Regr-MatrixEquations). As this involves some advanced matrix algebra we present here the case for calculating prediction intervals for a simple regression, where a forecast can be generated using the equation,

Assuming that the regression errors are normally distributed, an approximate 95% prediction interval (also called a prediction interval) associated with this forecast is given by

where is the total number of observations, is the mean of the observed values, is the standard deviation of the observed values and is the standard error of the regression given by Equation @ref(eq:Regr-se). Similarly, an 80% prediction interval can be obtained by replacing 1.96 by 1.28. Other prediction intervals can be obtained by replacing the 1.96 with the appropriate value given in Table @ref(tab:pcmultipliers). If R is used to obtain prediction intervals, more exact calculations are obtained (especially for small values of ) than what is given by Equation @ref(eq:Regr-pi).

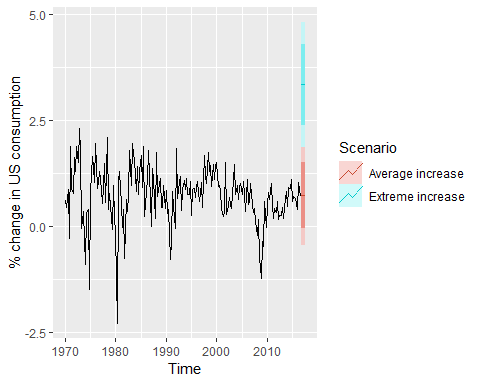
Equation @ref(eq:Regr-pi) shows that the prediction interval is wider when is far from . That is, we are more certain about our forecasts when considering values of the predictor variable close to its sample mean.

#### Example

The estimated simple regression line in the US consumption example is

Assuming that for the next four quarters personal income will increase by its historical mean value of , consumption is forecast to increase by and the corresponding and prediction intervals are and respectively (calcualted using R). If we assume an extreme increase of in income then the prediction intervals are considerably wider as shown in Figure @ref(fig:conSimplePI).

autoplot(uschange[,"Consumption"]) +  
 ylab("% change in US consumption") +  
 forecast::autolayer(fcast.up, PI=TRUE, series="Average increase") +  
 forecast::autolayer(fcast.down, PI=TRUE, series="Extreme increase") +  
 guides(colour=guide\_legend(title="Scenario"))



Prediction intervals if income is increased by its historical mean versus an extreme increase of 10%.

## Matrix formulation

*Warning: this is a more advanced optional section and assumes knowledge of matrix algebra.*

Recall that multiple regression model can be written as

where has mean zero and variance . This expresses the relationship between a single value of the forecast variable and the predictors.

It can be convenient to write this in matrix form where all the values of the forecast variable are given in a single equation. Let , , and

Then

where has mean and variance . Note that the matrix has rows reflecting the number of observations and columns reflecting the intercept which is represented by the column of ones plus the number of predictors.

## Least squares estimation

Least squares estimation is obtained by minimizing the expression . It can be shown that this is minimized when takes the value

This is sometimes known as the “normal equation”. The estimated coefficients require the inversion of the matrix . If is not of full column rank then matrix is singular and the model cannot be estimated. This will occur, for example, if you fall for the “dummy variable trap”, i.e., having the same number of dummy variables as there are categories of a categorical predictor, as discussed in Section @ref(Regr-UsefulPredictors).

The residual variance is estimated using

### Fitted values and cross-validation

The normal equation shows that the fitted values can be calculated using

where is known as the “hat-matrix” because it is used to compute (“y-hat”).

If the diagonal values of are denoted by , then the cross-validation statistic can be computed using

where is the residual obtained from fitting the model to all observations. Thus, it is not necessary to actually fit separate models when computing the CV statistic.

### Forecasts and prediction intervals

Let be a row vector containing the values of the predictors (in the same format as ) for which we want to generate a forecast . Then the forecast is given by

and its variance is given by

A 95% prediction interval can be calculated (assuming normally distributed errors) as

This takes account of the uncertainty due to the error term and the uncertainty in the coefficient estimates. However, it ignores any errors in . So if the future values of the predictors are uncertain, then the prediction interval calculated using this expression will be too narrow.

## Nonlinear regression

Although the linear relationship assumed so far in this chapter is often adequate, there are many cases for which a nonlinear functional form is more suitable. To keep things simple in this section we assume that we only have one predictor .

Simply transforming the forecast variable and/or the predictor variable and then estimating a regression model using the transformed variables is the simplest way of obtaining a nonlinear specification. We should note that these specifications and the specification that follow are still linear in the parameters. The most commonly used transformation is the (natural) logarithmic (see Section @ref(sec-transformations)).

A **log-log** functional form is specified as

In this model, the slope is interpreted as an elasticity, is the average percentage change in resulting from a increase in . Other useful forms can also be specified. The **log-linear** form is specified by only transforming the forecast variable and the **linear-log** form is obtained by transforming the predictor.

Recall that in order to perform a logarithmic transformation to a variable, all its observed values must be greater than zero. In the case that variable contains zeros we use the transformation , i.e., we add one to the value of the variable and then take logarithms. This has a similar effect to taking logarithms but avoids the problem of zeros. It also has the neat side-effect of zeros on the original scale remaining zeros on the transformed scale.

There are cases for which simply transforming the data will not be adequate and a more general specification may be required. Then the model we use is

where is a nonlinear function. In standard (linear) regression, . In the specification of nonlinear regression that follows, we allow to be a more flexible nonlinear function of , compared to simply a logarithmic or other transformation.

One of the simplest specifications is to make **piecewise linear**. That is, we introduce points where the slope of can change. These points are called “knots”. This can be achieved by letting and introducing variable such that

The notation means the value , if it is positive and 0 otherwise. This forces the slope to bend at point . Additional bends can be included in the relationship by adding further variables of the above form.

An example of this follows by considering and fitting a piecewise linear trend to a times series.

Piecewise linear relationships constructed in this way are a special case of **regression splines**. In general, a linear regression spline is obtained using

where are the knots (the points at which the line can bend). Selecting the number of knots () and where they should be positioned can be difficult and somewhat arbitrary. Some automatic knot selection algorithms are available in some software, but are not yet widely used.

A smoother result is obtained using piecewise cubics rather than piecewise lines. These are constrained so they are continuous (they join up) and they are smooth (so there are no sudden changes of direction as we see with piecewise linear splines). In general, a cubic regression spline is written as

Cubic splines usually give a better fit to the data. However, forecasting values of when is outside the range of the historical data becomes very unreliable.

### Forecasting with nonlinear trend

In Section @ref(Regr-UsefulPredictors) fitting a linear trend to a time series by setting was introduced. The simplest way of fitting a nonlinear trend is using quadratic or higher order trends obtained by specifying

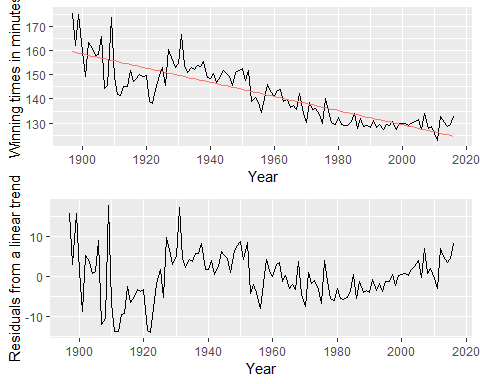
However, it is not recommended that quadratic or higher order trends are used in forecasting. When they are extrapolated, the resulting forecasts are often very unrealistic.

A better approach is to use the piecewise specification introduced above and fit a piecwise linear trend which bends at some point in time. We can think of this as a nonlinear trend constructed of linear pieces. If the trend bends at time , then it can be specified by simply replacing and above such that we include the predictors,

in the model. If the associated coefficients of and are and , then gives the slope of the trend before time , while the slope of the line after time is given by . Additional bends can be included in the relationship by adding further variables of the form where is the “knot” or point in time at which the line should bend.

#### Example: Boston marathon winning times

The top panel of Figure 5.20 shows the Boston marathon winning times (in minutes) since it started in 1897. The time series shows a general downward trend as the winning times have been improving over the years. The bottom panel shows the residuals from fitting a linear trend to the data. The plot shows an obvious nonlinear pattern which has not been captured by the linear trend. There is also some heteroscedasticity, with decreasing variation over time.



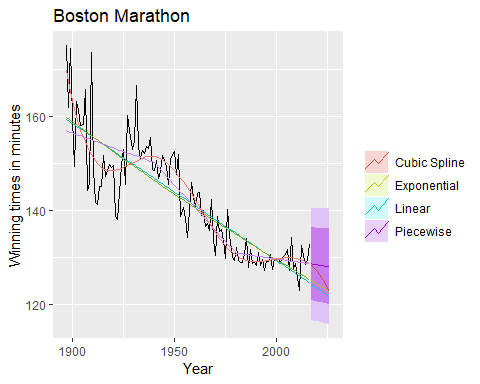
Fitting a linear trend to the Boston marathon winning times is inadequate

Fitting an exponential trend (equivalent to a log-linear regression) to the data can be achieved by transforming the variable so that the model to be fitted is,

This also addresses the **heteroskedasticity**. Thei fitted exponential trend and forecasts are shown in Fugre 5.21. Although the exponential trend does not seem to fit the data much better than the linear trend, it gives a more sensible projection in that the winning times will decrease in the future but at a decaying rate rather than a fixed linear rate.

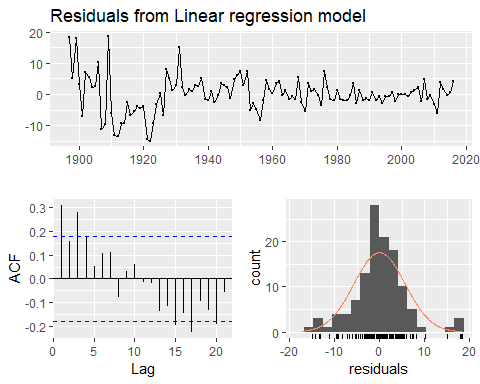
The plot of winning times reveals three different periods. There is a lot of volatility in the winning times up to about 1940, with the winning times decreasing overall but with significant increases during the 1920s. After 1940 there is a near-linear decrease in times, followed by a flattening out after the 1980s, with the suggestion of an upturn towards the end of the sample. To account for these changes, we specify the years 1940 and 1980 as knots. We should warn here that subjective identification of knots can lead to over-fitting, which can be detrimental to the forecast performance of a model, and should be performed with caution.

h <- 10  
fit.lin <- tslm(marathon ~ trend)  
fcasts.lin <- forecast(fit.lin, h = h)  
fit.exp <- tslm(marathon ~ trend, lambda = 0)  
fcasts.exp <- forecast(fit.exp, h = h)  
  
t <- time(marathon)  
t.break1 <- 1940  
t.break2 <- 1980  
tb1 <- ts(pmax(0, t - t.break1), start = 1897)  
tb2 <- ts(pmax(0, t - t.break2), start = 1897)  
  
fit.pw <- tslm(marathon ~ t + tb1 + tb2)  
t.new <- t[length(t)] + seq(h)  
tb1.new <- tb1[length(tb1)] + seq(h)  
tb2.new <- tb2[length(tb2)] + seq(h)  
  
newdata <- cbind(t=t.new, tb1=tb1.new, tb2=tb2.new) %>%  
 as.data.frame()  
fcasts.pw <- forecast(fit.pw, newdata = newdata)  
  
fit.spline <- tslm(marathon ~ t + I(t^2) + I(t^3) +  
 I(tb1^3) + I(tb2^3))  
fcasts.spl <- forecast(fit.spline, newdata = newdata)  
  
autoplot(marathon) +  
 autolayer(fitted(fit.lin), series = "Linear") +  
 autolayer(fitted(fit.exp), series = "Exponential") +  
 autolayer(fitted(fit.pw), series = "Piecewise") +  
 autolayer(fitted(fit.spline), series = "Cubic Spline") +  
 autolayer(fcasts.pw, series="Piecewise") +  
 autolayer(fcasts.lin, series="Linear", PI=FALSE) +  
 autolayer(fcasts.exp, series="Exponential", PI=FALSE) +  
 autolayer(fcasts.spl, series="Cubic Spline", PI=FALSE) +  
 xlab("Year") + ylab("Winning times in minutes") +  
 ggtitle("Boston Marathon") +  
 guides(colour = guide\_legend(title = " "))



Projecting forecasts from a linear, exponential, piecewise linear trends and a cubic spline for the Boston marathon winning times

checkresiduals(fit.pw)



Residuals from the piecewise trend.

##   
## Breusch-Godfrey test for serial correlation of order up to 10  
##   
## data: Residuals from Linear regression model  
## LM test = 21.597, df = 10, p-value = 0.0173

There is an alternative formulation of cubic splines (called **natural cubic smoothing splines**) that imposes some constraints, so the spline function is linear at the end, which usually gives much better forecasts without compromising the fit. In Figure 5.22, we have used the splinef() function to produce the cubic spline forecasts. This uses many more knots than we used in Figure 5.21, but the coefficients are constrained to prevent over-fitting, and the curve is linear at both ends. This has the added advantage that knot selection is not subjective. We have also used a **log transformation** (lambda=0) to handle the **heteroscedasticity**.

marathon %>%   
 splinef(lambda=0) %>%   
 autoplot()

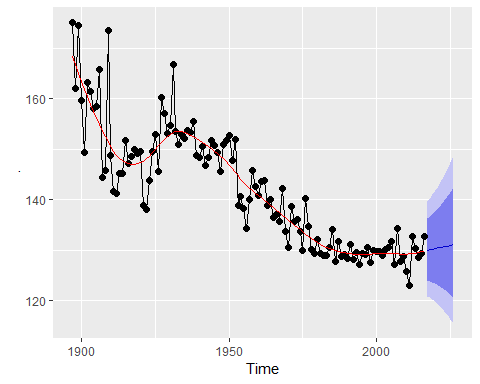


Figure 5.22

The residuals plotted in Figure 5.23 show that this model has captured the trend well, although there is some **heteroscedasticity** remaining. The wide prediction interval associated with the forecasts reflects the volatility observed in the historical winning times.

marathon %>%   
 splinef(lambda=0) %>%   
 checkresiduals()

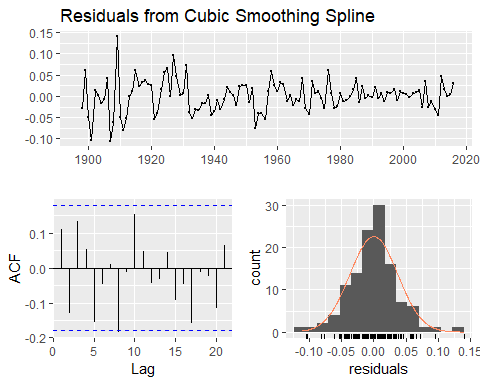


Figure 5-23: Residuals from the cubic spline trend

## Correlation, causation and forecasting

### Correlation is not causation

It is important not to confuse correlation with causation, or causation with forecasting. A variable may be useful for predicting a variable , but that does not mean is causing . It is possible that *is* causing , but it may be that the relationship between them is more complicated than simple causality.

For example, it is possible to model the number of drownings at a beach resort each month with the number of ice-creams sold in the same period. The model can give reasonable forecasts, not because ice-creams cause drownings, but because people eat more ice-creams on hot days when they are also more likely to go swimming. So the two variables (ice-cream sales and drownings) are correlated, but one is not causing the other. It is important to understand that **correlations are useful for forecasting, even when there is no causal relationship between the two variables**.

However, often a better model is possible if a causal mechanism can be determined. In this example, both ice-cream sales and drownings will be affected by the temperature and by the numbers of people visiting the beach resort. Again, high temperatures do not actually *cause* people to drown, but they are more directly related to why people are swimming. So a better model for drownings will probably include temperatures and visitor numbers and exclude ice-cream sales.

### Confounded predictors

A related issue involves confounding variables. Suppose we are forecasting monthly sales of a company for 2012, using data from 2000–2011. In January 2008 a new competitor came into the market and started taking some market share. At the same time, the economy began to decline. In your forecasting model, you include both competitor activity (measured using advertising time on a local television station) and the health of the economy (measured using GDP). It will not be possible to separate the effects of these two predictors because they are correlated. We say two variables are **confounded** when their effects on the forecast variable cannot be separated. Any pair of correlated predictors will have some level of confounding, but we would not normally describe them as confounded unless there was a relatively high level of correlation between them.

Confounding is not really a problem for forecasting, as we can still compute forecasts without needing to separate out the effects of the predictors. However, it becomes a problem with scenario forecasting as the scenarios should take account of the relationships between predictors. It is also a problem if some historical analysis of the contributions of various predictors is required.

### Multicollnearity and forecasting

A closely related issue is **multicollinearity** which occurs when similar information is provided by two or more of the predictor variables in a multiple regression. It can occur in a number of ways.

Two predictors are highly correlated with each other (that is, they have a correlation coefficient close to +1 or -1). In this case, knowing the value of one of the variables tells you a lot about the value of the other variable. Hence, they are providing similar information.

A linear combination of predictors is highly correlated with another linear combination of predictors. In this case, knowing the value of the first group of predictors tells you a lot about the value of the second group of predictors. Hence, they are providing similar information.

The dummy variable trap is a special case of multicollinearity. Suppose you have quarterly data and use four dummy variables, and . Then , so there is perfect correlation between and .

TODO: I have changed this below - change back if not improved

The dummy variable trap is a special case of multicollinearity, referred to as perfect multicollinearity. Suppose you have quarterly data and use four dummy variables, and . Then and therefore there is perfect correlation between the column of ones representing the constant and the linear combination, in this case the sum, of the dummies.

In the case of **perfect correlation** (i.e., a correlation of +1 or -1, such as in the dummy variable trap), it is not possible to estimate the regression model. If there is **high correlation** (close to but not equal to +1 or -1), then the estimation of the regression coefficients is computationally difficult. In fact, some software (notably Microsoft Excel) may give highly **inaccurate estimates** of the coefficients. Most reputable statistical software will use algorithms to limit the effect of multicollinearity on the coefficient estimates, but you do need to be careful. The major software packages such as R, SPSS, SAS and Stata all use estimation algorithms to avoid the problem as much as possible.

The uncertainty associated with individual regression coefficients will be large. This is because they are difficult to estimate. Consequently, statistical tests (e.g., t-tests) on regression coefficients are unreliable. (In forecasting we are rarely interested in such tests.) Also, it will not be possible to make accurate statements about the contribution of each separate predictor to the forecast.

Forecasts will be unreliable if the values of the future predictors are outside the range of the historical values of the predictors. For example, suppose you have fitted a regression model with predictors and which are highly correlated with each other, and suppose that the values of in the fitting data ranged between 0 and 100. Then forecasts based on or will be unreliable. It is always a little dangerous when future values of the predictors lie much outside the historical range, but it is especially problematic when multicollinearity is present.

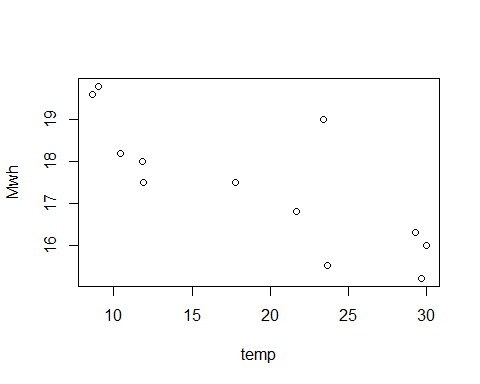
## Exercise

1. Electricity consumption was recorded for a small town on 12 consecutive days. The following maximum temperatures (degrees Celsius) and consumption (megawatt-hours) were recorded for each day. TODO: change the econsumption to a ts of 12 concecutive days - change the lm to tslm below

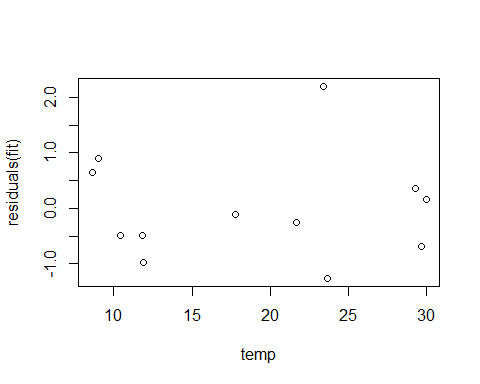
|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 |
| Mwh | 16.3 | 16.8 | 15.5 | 18.2 | 15.2 | 17.5 | 19.8 | 19.0 | 17.5 | 16.0 | 19.6 | 18.0 |
| Temp | 29.3 | 21.7 | 23.7 | 10.4 | 29.7 | 11.9 | 9.0 | 23.4 | 17.8 | 30.0 | 8.6 | 11.8 |

1. Plot the data and find the regression model for Mwh with temperature as an explanatory variable. Why is there a negative relationship?
2. Produce a residual plot. Is the model adequate? Are there any outliers or influential observations?
3. Use the model to predict the electricity consumption that you would expect for the next day if the maximum temperature was and compare it with the forecast if the with maximum temperature was . Do you believe these predictions?
4. Give prediction intervals for your forecasts. The following R code will get you started:

plot(Mwh ~ temp, data=econsumption)



fit <- lm(Mwh ~ temp, data=econsumption)  
plot(residuals(fit) ~ temp, data=econsumption)



forecast(fit, newdata=data.frame(temp=c(10,35)))

## Point Forecast Lo 80 Hi 80 Lo 95 Hi 95  
## 1 18.74795 17.27010 20.22579 16.34824 21.14766  
## 2 15.11902 13.50469 16.73335 12.49768 17.74035

1. Data set olympic contains the winning times (in seconds) for the men’s 400 meters final in each Olympic Games from 1896 to 2012.
2. Plot the winning time against the year. Describe the main features of the scatterplot.
3. Fit a regression line to the data. Obviously the winning times have been decreasing, but at what *average* rate per year?
4. Plot the residuals against the year. What does this indicate about the suitability of the fitted line?
5. Predict the winning time for the men’s 400 meters final in the 2000, 2004, 2008 and 2012 Olympics. Give a prediction interval for each of your forecasts. What assumptions have you made in these calculations?
6. Find out the actual winning times for these Olympics (see <www.databaseolympics.com>). How good were your forecasts and prediction intervals?
7. Type easter(ausbeer) and interpret what you see.
8. An elasticity coefficient is the ratio of the percentage change in the forecast variable () to the percentage change in the predictor variable (). Mathematically, the elasticity is defined as . Consider the log-log model,
9. Express as a function of and show that the coefficient is the elasticity coefficient.
10. The data set fancy concerns the monthly sales figures of a shop which opened in January 1987 and sells gifts, souvenirs, and novelties. The shop is situated on the wharf at a beach resort town in Queensland, Australia. The sales volume varies with the seasonal population of tourists. There is a large influx of visitors to the town at Christmas and for the local surfing festival, held every March since 1988. Over time, the shop has expanded its premises, range of products, and staff.
11. Produce a time plot of the data and describe the patterns in the graph. Identify any unusual or unexpected fluctuations in the time series.
12. Explain why it is necessary to take logarithms of these data before fitting a model.
13. Use R to fit a regression model to the logarithms of these sales data with a linear trend, seasonal dummies and a “surfing festival” dummy variable.
14. Plot the residuals against time and against the fitted values. Do these plots reveal any problems with the model?
15. Do boxplots of the residuals for each month. Does this reveal any problems with the model?
    1. What do the values of the coefficients tell you about each variable?
    2. What does the Breusch-Godfrey test tell you about your model?
    3. Regardless of your answers to the above questions, use your regression model to predict the monthly sales for 1994, 1995, and 1996. Produce prediction intervals for each of your forecasts.
    4. Transform your predictions and intervals to obtain predictions and intervals for the raw data.
    5. How could you improve these predictions by modifying the model?
16. TODO: you got to this before me ;-) The gasoline series consists of weekly data for supplies of US finished motor gasoline product, from 2 February 1991 to 20 January 2017. The units are in “thousand barrels per day”. Consider only the data to the end of 2004.
    1. Fit a harmonic regression with trend to the data. Select the appropriate number of Fourier terms to include by minimizing the AICc or CV value.
    2. Check the residuals of the final model using the checkresiduals() function. Even though the residuals fail the correlation tests, the results are probably not severe enough to make much difference to the forecasts and prediction intervals. (Note that the correlations are relatively small, even though they are significant.)
    3. To forecast using harmonic regression, you will need to generate the future values of the Fourier terms. This can be done as follows.

fc <- forecast(fit, fourier(x, K, h))

where fit is the fitted model using tslm, K is the number of Fourier terms used in creating fit, and h is the forecast horizon required.

Forecast the next year of data.

1. Plot the forecasts along with the actual data for 2005. What do you find?
2. *(For advanced readers following on from Section @ref(Regr-MatrixEquations))*.

Using matrix notation it was shown that if , where has mean and variance matrix , the estimated coefficients are given by and a forecast is given by where is a row vector containing the values of the regressors for the forecast (in the same format as ), and the forecast variance is given by

Consider the simple time trend model where . Using the following results,

derive the following expressions: a.

## Further reading

Regression with time series data

* Shumway, R. H. and D. S. Stoffer (2011). Time series analysis and its applications: with R examples. 3rd ed. New York: Springer.

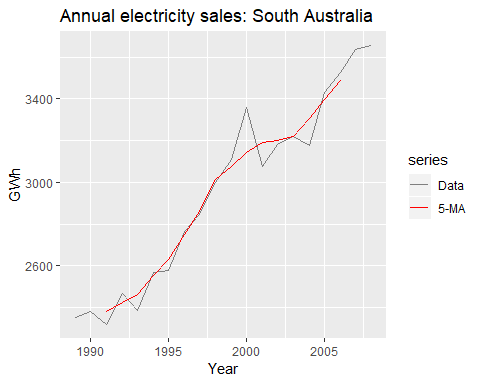
Note that if you are using good statistical software, if you are not interested in the specific contributions of each predictor, and if the future values of your predictor variables are within their historical ranges, there is nothing to worry about — multicollinearity is not a problem.

|  |
| --- |
| # Time series decomposition Time series data can exhibit a variety of patterns, and it is often helpful to split a time series into several components, each representing an underlying pattern category. |
| In Section 2.3 we discussed three types of time series patterns: trend, seasonality and cycles. When we decompose a time series into components, we usually combine the trend and cycle into a single trend-cycle component (sometimes called the trend for simplicity). Thus we think of a time series as comprising three components: a trend-cycle component, a seasonal component, and a remainder component (containing anything else in the time series). |
| In this chapter, we consider some common methods for extracting these components from a time series. Often this is done to help improve understanding of the time series, but it can also be used to improve forecast accuracy. |
| ## Time series components If we assume an additive model, then we can write |
| **The additive model** is the most appropriate if the magnitude of the *seasonal fluctuations*, or the variation around the *trend-cycle*, does not vary with the level of the time series. When the variation in the seasonal pattern, or the variation around the trend-cycle, appears to be proportional to the level of the time series, then a multiplicative model is more appropriate. Multiplicative models are common with economic time series. |
| An alternative to using a multiplicative model is to first transform the data until the variation in the series appears to be stable over time, then use an additive model. When a log transformation has been used, this is equivalent to using a multiplicative decomposition because  ### Electrical equipment manufacturing We will look at several methods for obtaining the components , and later in this chapter, but first, it is helpful to see an example. We will decompose the new orders index for electrical equipment shown in Figure @ref(fig:elecequip-trend). These data show the numbers of new orders for electrical equipment (computer, electronic and optical products) in the Euro area (16 countries). The data have been adjusted by working days and normalized so that a value of 100 corresponds to 2005. |
| Electrical equipment orders: the trend-cycle component (red) and the raw data (grey). |
| Figure @ref(fig:elecequip-trend) shows the **trend-cycle component**, , in red and the original data, , in grey. The trend-cycle shows the overall movement in the series, ignoring the seasonality and any small random fluctuations. |
| Figure @ref(fig:elecequip-stl) shows an additive decomposition of these data. The method used for estimating components in this example is STL, which is discussed in Section @ref(sec-6-stl). |
| The electricial equipment orders (top) and its three additive components. |
| The three components are shown separately in the bottom three panels of Figure @ref(fig:elecequip-stl). These components can be added together to reconstruct the data shown in the top panel. Notice that the seasonal component changes very slowly over time, so that any two consecutive years have very similar patterns, but years far apart may have different seasonal patterns. The remainder component shown in the bottom panel is what is left over when the seasonal and trend-cycle components have been subtracted from the data. |
| The grey bars to the right of each panel show the relative scales of the components. Each grey bar represents the same length but because the plots are on different scales, the bars vary in size. The large grey bar in the bottom panel shows that the variation in the remainder component is small compared to the variation in the data, which has a bar about one quarter the size. If we shrunk the bottom three panels until their bars became the same size as that in the data panel, then all the panels would be on the same scale. |
| ### Seasonally adjusted data |
| If the seasonal component is removed from the original data, the resulting values are called the “seasonally adjusted” data. For an additive model, the seasonally adjusted data are given by , and for multiplicative data, the seasonally adjusted values are obtained using . |
| Figure @ref(fig:elecequip-sa) shows the seasonally adjusted electrical equipment orders. Seasonally adjusted electrical equipment orders (red) and the original data (grey). |
| If the variation due to seasonality is not of primary interest, the **seasonally adjusted sereis** can be useful. For example, monthly unemployment data are usually seaonally adjusted in order to highlight variation due to the underlying state of the economy rather than the seasonal variation. An increase in unemployment due to school leavers seeking work is seasonal variation, while an increase in unemployment due to large employers laying off workers is non-seasonal. Most people who study unemployment data are more interested in the non-seasonal variation. Consequently, employment data (and many other economic series) are usually seasonally adjusted. |
| **Seasonally adjusted series** contain the remainder component as well as the trend-cycle. Therefore, they are not “smooth”, and “downturns” or “upturns” can be misleading. If the purpose is to look for turning points in the series, and interpret any changes in the series, then it is better to use the trend-cycle component rather than the seasonally adjusted data. |
| ## Moving averages The classical method of time series decomposition originated in the 1920s and was widely used until the 1950s. IT still forms the basis of many time sereis decompsotion methods, si it is important to understand how it works. The first step in a classical decomposition is to use a moving average method to estimate the trend-cycle, so we begin by discussing moving averages. |
| ### Moving average smoothing Moving average of order can be written as |
| ,where . That is, the estimate of the trend-cycle at time is obtained by averaging values of the time series within periods of . Observations that are nearby in time are also likely to be close in value, and the average eliminates some of the randomness in the data, leaving a smooth trend-cycle component. We call this an “-MA”, meaning a moving average of order . |
| r autoplot(elecsales)+xlab("Year")+ylab("GWh")+ ggtitle("Annual electricity sales: South Australia") |
| Residential electricity sales (excluding hot water) for South Australia: 1989–2008. |
| For example, consider Figure @ref(fig:ressales1) which shows the volume of electricity sold to residential customers in South Australia each year from 1989 to 2008 (hot water sales have been excluded). The data are also shown in Table @ref(tab:elecsales). |
| r ma5 <- ma(elecsales,5) autoplot(ma5) |
|  |
| r # ts(as.data.frame(elecsales),frequency = 1, start=1989) autoplot(elecsales)+ autolayer(ma5, sereis="MA") |
| ## Warning: Ignoring unknown parameters: sereis |
| ## Warning: Removed 4 rows containing missing values (geom\_path). |
|  |
| Table: Annual electricity sales to residential customers in South Australia. 1989-2008. |
| Year Sales (GWh) 5-MA |

1989 2354.34  
1990 2379.71  
1991 2318.52 2381.53 1992 2468.99 2424.56 1993 2386.09 2463.76 1994 2569.47 2552.60 1995 2575.72 2627.70 1996 2762.72 2750.62 1997 2844.50 2858.35 1998 3000.70 3014.70 1999 3108.10 3077.30 2000 3357.50 3144.52 2001 3075.70 3188.70 2002 3180.60 3202.32 2003 3221.60 3216.94 2004 3176.20 3307.30 2005 3430.60 3398.75 2006 3527.48 3485.43 2007 3637.89  
2008 3655.00

In the second column of this table, amoving average of order 5 is shown, providing an estimate of the trend-cycle. The first value in this column is the average of the first observations(1989–1993); the second value in the 5-MA column is the average of the values for 1990–1994; and so on. Each value in the 5-MA column is the average of the observations in the five year period centered on the corresponding year. There are no values for either the first two years or the last two years, because we do not have two observations on either side. In the formula above, column 5-MA contains the values of with . To see what the trend-cycle estimate looks like, we plot it along with the original data in Figure @ref(fig:ressales2).

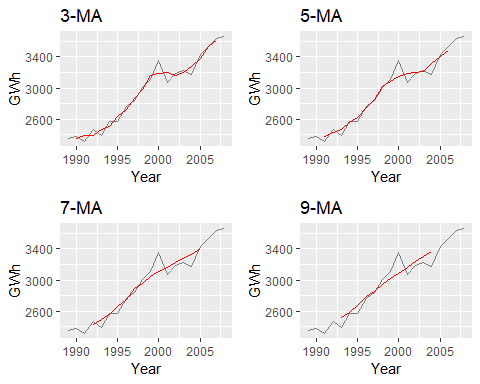
autoplot(elecsales, series="Data")+  
 forecast::autolayer(ma(elecsales,5),series="5-MA")+  
 xlab("Year")+ylab("GWh")+  
 ggtitle("Annual electricity sales: South Australia")+  
 scale\_colour\_manual(values=c("Data"="grey50","5-MA"="red"),  
 breaks=c("Data","5-MA"))



Residential electricity sales (black) along with the 5-MA estimate of the trend-cycle (red).

Notice that the *trend-cycle* (in red) is smooother than the original data, and captures the main movement of the time series without all of the minor fluctuations. The moving average method does not allow estimates of when is close to the ends of the series; hence the red line does not extend to the edges of the graph on either side. Later we will use more sophisticated methods of trend-cycle estimation which do allow estimates near the endpoints.

The order of the moving average determines the smoothness of the trend-cycle estimate. In general, a larger order means a smoother curve. Figure @ref(fig:ressales3) shows the effect of changing the order of the moving average for the residential electricity sales data.



Different moving averages applied to the residential electricity sales data.

Simple moving averages such as these are usually of an odd order (e.g., 3, 5, 7, etc.) This is so they are symmetric: in a moving average of order , there are earlier observations, later observations and the middle observation that are averaged. But if was even, it would no longer be symmetric.

### Moving averages of moving averages

It is possible to apply a moving average to a moving average. One reason for doing this is to make an even-order moving average symmetric.

For example, we might take a moving average of order 4, and then apply another moving average of order 2 to the results. In the following table, this has been done for the first few years of the Australian quarterly beer production data.

beer2 <- window(ausbeer, start=1992)  
ma4 <- ma(beer2, order=4, centre=FALSE)  
ma2x4 <- ma(beer2, order=4, centre = TRUE)

A moving average of order 4 applied to the quarterly beer data,followed by a moving average of order 2.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Year | Quarter | Observation | 4-MA | 2x4-MA |
| 1992 | Q1 | 443 |  |  |
| 1992 | Q2 | 410 | 451.25 |  |
| 1992 | Q3 | 420 | 448.75 | 450.000 |
| 1992 | Q4 | 532 | 451.50 | 450.125 |
| 1993 | Q1 | 433 | 449.00 | 450.250 |
| 1993 | Q2 | 421 | 444.00 | 446.500 |
| 1993 | Q3 | 410 | 448.00 | 446.000 |
| 1993 | Q4 | 512 | 438.00 | 443.000 |
| 1994 | Q1 | 449 | 441.25 | 439.625 |
| 1994 | Q2 | 381 | 446.00 | 443.625 |
| 1994 | Q3 | 423 | 440.25 | 443.125 |
| 1994 | Q4 | 531 | 447.00 | 443.625 |
| 1995 | Q1 | 426 | 445.25 | 446.125 |
| 1995 | Q2 | 408 | 442.50 | 443.875 |
| 1995 | Q3 | 416 | 438.25 | 440.375 |
| 1995 | Q4 | 520 | 435.75 | 437.000 |
| 1996 | Q1 | 409 | 431.25 | 433.500 |
| 1996 | Q2 | 398 | 428.00 | 429.625 |
| 1996 | Q3 | 398 | 433.75 | 430.875 |
| 1996 | Q4 | 507 | 433.75 | 433.750 |

The notation “-MA” in the last column means a 4-MA followed by a 2-MA. The values in the last column are obtained by taking a moving average of order 2 of the values in the previous column. For example, the first two values in the 4-MA column are 451.25=(443+410+420+532)/4 and 448.75=(410+420+532+433)/4. The first value in the 2x4-MA column is the average of these two: 450.00=(451.25+448.75)/2.

When a 2-MA follows a moving average of an even order (such as 4), it is called a “centered moving average of order 4”. This is because the results are now symmetric. To see that this is the case, we can write the -MA as follows:

It is now a weighted average of observations, but it is symmetric.

Other combinations of moving averages are also possible. For example, a -MA is often used, and consists of a moving average of order 3 followed by another moving average of order 3. In general, an even order MA should be followed by an even order MA to make it symmetric. Similarly, an odd order MA should be followed by an odd order MA.

### Estimating the trend-cycle with seasonal data

The most common use of centered moving averages is in estimating the **trend-cycle** from seasonal data. Consider the -MA:

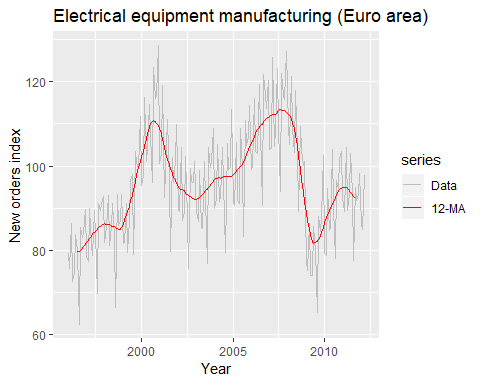
When applied to quarterly data, each quarter of the year is given equal weight as the first and last terms apply to the same quarter in consecutive years. Consequently, the seasonal variation will be averaged out and the resulting values of will have little or no seasonal variation remaining. A similar effect would be obtained using a -MA or a -MA to quarterly data.

In general, a -MA is equivalent to a weighted moving average of order where all observations take the weight , except for the first and last terms which take weights . So, if the seasonal period is even and of order , use a -MA to estimate the trend-cycle. If the seasonal period is odd and of order , use a -MA to estimate the trend-cycle. For example, a -MA can be used to estimate the trend-cycle of monthly data and a 7-MA can be used to estimate the trend-cycle of daily data with a weekly seasonality.

Other choices for the order of the MA will usually result in trend-cycle estimates being contaminated by the seasonality in the data.

### Example: Electrical equipment manufacturing

autoplot(elecequip, series="Data")+  
 forecast::autolayer(ma(elecequip,12),series="12-MA")+  
 xlab("Year")+ylab("New orders index")+  
 ggtitle("Electrical equipment manufacturing (Euro area)")+  
 scale\_color\_manual(values=c("Data"="grey","12-MA"="red"),  
 breaks=c("Data","12-MA"))



A 2x12-MA applied to the electrical equipment orders index.

Figure @ref(fig:elecequip2) shows a -MA applied to the electrical equipment orders index. Notice that the smooth line shows no seasonality; it is almost the same as the trend-cycle shown in Figure @ref(fig:elecequip-trend), which was estimated using a much more sophisticated method than moving averages. Any other choice for the order of the moving average (except for 24, 36, etc.) would have resulted in a smooth line that showed some seasonal fluctuations.

### weighted moving averages

Combinations of moving averages result in weighted moving averages. For example, the -MA discussed above is equivalent to a weighted 5-MA with weights given by . In general, a weighted -MA can be written as

where , and the weights are given by . It is important that the weights all sum to one and that they are symmetric so that . The simple -MA is a special case where all of the weights are equal to .

A major advantage of weighted moving averages is that they yield a smoother estimate of the trend-cycle. Instead of observations entering and leaving the calculation at full weight, their weights slowly increase and then slowly decrease, resulting in a smoother curve.

Some specific sets of weights are widely used due to their mathematical properties. Some of these are given in Table @ref(tab:weights).

Commonly used weights in weighted moving averages. S=Spencer’s weighted moving average. H=Henderson’s weighted moving average

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |  |  |  |  |  |  |
| 3-MA | 0.333 | 0.333 |  |  |  |  |  |  |  |  |  |  |
| 5-MA | 0.200 | 0.200 | 0.200 |  |  |  |  |  |  |  |  |  |
| 2x12-MA | 0.083 | 0.083 | 0.083 | 0.083 | 0.083 | 0.083 | 0.042 |  |  |  |  |  |
| 3x3-MA | 0.333 | 0.222 | 0.111 |  |  |  |  |  |  |  |  |  |
| 3x5-MA | 0.200 | 0.200 | 0.133 | 0.067 |  |  |  |  |  |  |  |  |
| S15-MA | 0.231 | 0.209 | 0.144 | 0.066 | 0.009 | -0.016 | -0.019 | -0.009 |  |  |  |  |
| S21-MA | 0.171 | 0.163 | 0.134 | 0.094 | 0.051 | 0.017 | -0.006 | -0.014 | -0.014 | -0.009 | -0.003 |  |
| H5-MA | 0.558 | 0.294 | -0.073 |  |  |  |  |  |  |  |  |  |
| H9-MA | 0.330 | 0.267 | 0.119 | -0.010 | -0.041 |  |  |  |  |  |  |  |
| H13-MA | 0.240 | 0.214 | 0.147 | 0.066 | 0.000 | -0.028 | -0.019 |  |  |  |  |  |
| H23-MA | 0.148 | 0.138 | 0.122 | 0.097 | 0.068 | 0.039 | 0.013 | -0.005 | -0.015 | -0.016 | -0.011 | -0.004 |

## Classical decomposition

The classical decomposition method originated in the 1920s. It is a relatively simple procedure, and forms the starting point for most other methods of time series decomposition. There are two forms of classical decomposition: an **additive decomposition** and a **multiplicative decomposition**. These are described below for a time series with seasonal period (e.g., for quarterly data, for monthly data, for daily data with a weekly pattern).

In classical decomposition, we assume that the seasonal component is constant from year to year. For multiplicative seasonality, the values that form the seasonal component are sometimes called the “seasonal indices”.

### Additive decomposition

Step 1

If is an even number, compute the trend-cycle component using a -MA to obtain . If is an odd number, compute the trend-cycle component using an -MA to obtain .

Step 2

Calculate the detrended series: .

Step 3

To estimate the seasonal component for each season, simply average the detrended values for that season. For example, with monthly data, the seasonal component for March is the average of all the detrended March values in the data. These seasonal component values are then adjusted to ensure that they add to zero. The seasonal component is obtained by stringing together these values for each year of data. This gives .

Step 4

The remainder component is calculated by subtracting the estimated seasonal and trend-cycle components: .

### Multiplicative decomposition

A classical multiplicative decomposition is very similar, except that the subtractions are replaced by divisions.

Step 1

If is an even number, compute the trend-cycle component using a -MA to obtain . If is an odd number, compute the trend-cycle component using an -MA to obtain .

Step 2

Calculate the detrended series: .

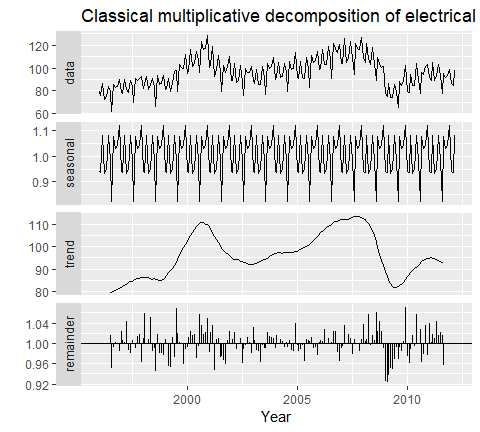
Step 3

To estimate the seasonal component for each season, simply average the detrended values for that season. For example, with monthly data, the seasonal index for March is the average of all the detrended March values in the data. These seasonal indexes are then adjusted to ensure that they add to . The seasonal component is obtained by stringing together all of the seasonal indices for each year of data. This gives .

Step 4

The remainder component is calculated by dividing out the estimated seasonal and trend-cycle components: .

elecequip %>%   
 decompose(type="multiplicative") %>%   
 autoplot()+xlab("Year")+  
 ggtitle("Classical multiplicative decomposition of electrical equipment index")



A classical multiplicative decomposition of the new orders index for electrical equipment.

Figure @ref(fig:classical-elecequip) shows a classical decomposition of the electrical equipment index. Compare this decomposition with that shown in Figure @ref(fig:elecequip-trend). The run of large negative remainder values in 2009 suggests that there is some “leakage” of the trend-cycle component into the remainder component. The trend-cycle estimate has over-smoothed the drop in the data, and the corresponding remainder values have been affected by the poor trend-cycle estimate.

### Comments on classical decomposition

While classical decomposition is still widely used, it is **not recommended**, as there are now several much better methods. Some of the problems with classical decomposition are summarised below.

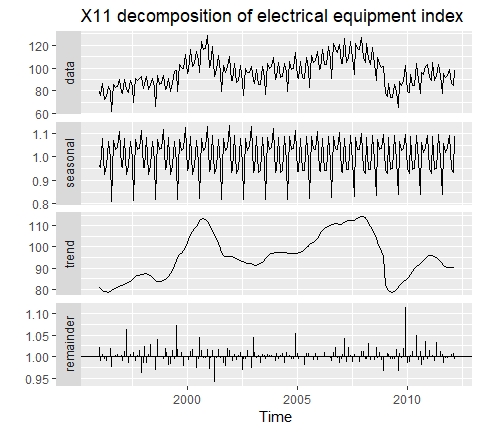
* The estimate of the trend-cycle is unavailable for the first few and last few observations. For example, if , there is no trend-cycle estimate for the first six or the last six observations. Consequently, there is also no estimate of the remainder component for the same time periods.
* The trend-cycle estimate tends to over-smooth rapid rises and falls in the data (as seen in the above example).
* Classical decomposition methods assume that the seasonal component repeats from year to year. For many series, this is a reasonable assumption, but for some longer series it is not. For example, electricity demand patterns have changed over time as air conditioning has become more widespread. Specifically, in many locations, the seasonal usage pattern from several decades ago had its maximum demand in winter (due to heating), while the current seasonal pattern has its maximum demand in summer (due to air conditioning). The classical decomposition methods are unable to capture these seasonal changes over time.
* Occasionally, the values of the time series in a small number of periods may be particularly unusual. For example, the monthly air passenger traffic may be affected by an industrial dispute, making the traffic during the dispute very different from usual. The classical method is not robust to these kinds of unusual values.

## X11 decomposition

Another popular method for decomposing quarterly and monthly data is the **X11 method** which originated in the US Census Bureau and Statistics Canada.

This method is based on classical decomposition, but includes many *extra steps and features* in order to overcome the drawbacks of classical decomposition that were discussed in the previous section. In particular, **trend-cycle estimates** are available for all observations including the end points, and the seasonal component is allowed to vary slowly over time. X11 also has some sophisticated methods for handling trading day variation, holiday effects and the effects of known predictors. It handles both additive and multiplicative decomposition. The process is entirely automatic and tends to be highly robust to outliers and level shifts in the time series.

The details of the X11 method are described in @Dagum2016. Here we will only demonstrate how to use the automatic procedure in R.

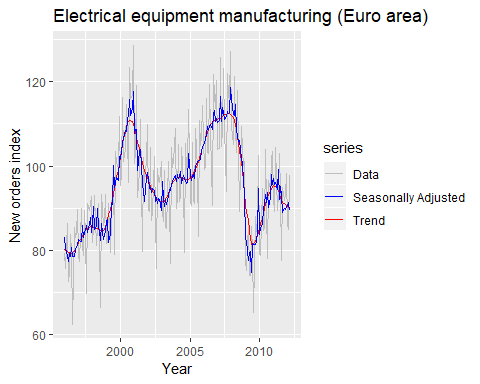
The X11 method is available using the seas function from the seasonal package for R. 

Compare this decomposition with the **STL decomposition** shown in Figure @ref(fig:elecequip-trend) and the classical decomposition shown in Figure @ref(fig:classical-elecequip). The X11 trend-cycle has captured the sudden fall in the data in early 2009 better than either of the other two methods, and the unusual observation at the end of 2009 is now more clearly seen in the remainder component.

Given the output from the seas function, seasonal() will extract the seasonal component, trendcycle() will extract the trend-cycle component, remainder() will extract the remainder component, and seasadj() will compute the seasonally adjusted time series.

For example, Figure @ref(fig:x11-seasadj) shows the trend-cycle component and the seasonally adjusted data, along with the original data.

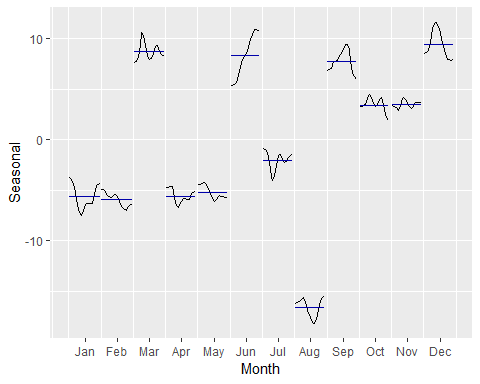
autoplot(elecequip, series="Data") +  
 forecast::autolayer(trendcycle(fit), series="Trend") +  
 forecast::autolayer(seasadj(fit), series="Seasonally Adjusted") +  
 xlab("Year") + ylab("New orders index") +  
 ggtitle("Electrical equipment manufacturing (Euro area)") +  
 scale\_colour\_manual(values=c("gray","blue","red"),  
 breaks=c("Data","Seasonally Adjusted","Trend"))



Electrical equipment orders: the original data (grey), the trend-cycle component (red) and the seasonally adjusted data (blue).

It can be useful to use seasonal plots and seasonal sub-series plots of the seasonal component. These help us to visualize the variation in the seasonal component over time. Figure @ref(fig:elecequip3) shows a seasonal sub-series plot of the seasonal component from Figure @ref(fig:x11). In this case, there are only very small changes over time.

ggsubseriesplot(seasonal(fit)) + ylab("Seasonal")



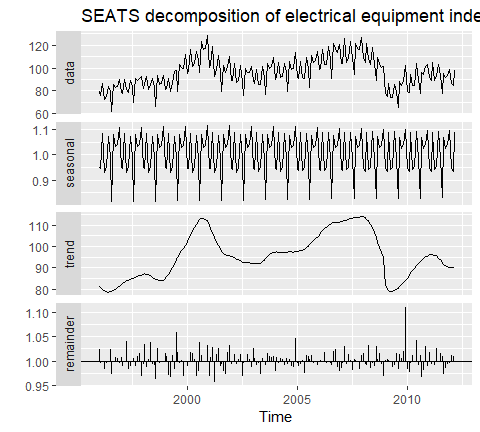
Seasonal sub-series plot of the seasonal component from the X11 decomposition shown in Figure @ref(fig:x11).

## SEATS decomposition

“SEATS” stands for **“Seasonal Extraction in ARIMA Time Series”** (ARIMA models are discussed in Chapter @ref(ch-arima)). This procedure was developed at the Bank of Spain, and is now widely used by government agencies around the world. The procedure works only with quarterly and monthly data. So seasonality of other kinds, such as daily data, or hourly data, or weekly data, require an alternative approach.

The details are beyond the scope of this book. However, a complete discussion of the method is available in @Dagum2016. Here we will only demonstrate how to use it via the seasonal package.

library(seasonal)  
  
elecequip %>% seas() %>%   
 autoplot()+  
 ggtitle("SEATS decomposition of electrical equipment index")



A SEATS decomposition of the new orders index for electrical equipment.

The result is quite similar to the X11 decomposition shown in Figure @ref(fig:x11).

As with the X11 method, we can use the seasonal(), trendcycle() and remainder() functions to extract the individual components, and seasadj() to compute the seasonally adjusted time series.

The seasonal package has many options for handling variations of X11 and SEATS. See [the package website](http://www.seasonal.website/seasonal.html) for a detailed introduction to the options and features available.

## STL decomposition

**STL** is a very versatile and robust method for decomposing time series. STL is an acronym for “Seasonal and Trend decomposition using Loess”, while Loess is a method for estimating nonlinear relationships. The STL method was developed by @Cleveland1990.

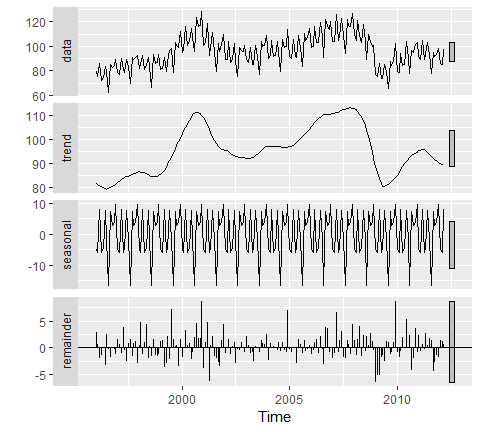
STL has several advantages over the classical, SEATS and X-11 decomposition methods: - Unlike SEATS and X-11, STL will handle any type of seasonality, not only monthly and quarterly data. - The **seasonal component** is allowed to change over time, and the rate of change can be controlled by the user. - The **smoothness** of the trend-cycle can also be controlled by the user. - It can be robust to **outliers** (i.e., the user can specify a robust decomposition), so that occasional unusual observations will not affect the estimates of the trend-cycle and seasonal components. They will, however, affect the remainder component.

On the other hand, STL has some disadvantages. In particular, it does not handle trading day or calendar variation automatically, and it only provides facilities for additive decompositions.

It is possible to obtain a multiplicative decomposition by first taking logs of the data, then back-transforming the components. Decompositions between additive and multiplicative can be obtained using a Box-Cox transformation of the data with . A value of corresponds to the multiplicative decomposition while is equivalent to an additive decomposition.

The best way to begin learning how to use STL is to see some examples and experiment with the settings. Figure @ref(fig:elecequip-stl) showed an example of STL applied to the electrical equipment orders data. Figure @ref(fig:elecequip-stl2) shows an alternative STL decomposition where the trend-cycle is more flexible, the seasonal component does not change over time, and the robust option has been used. Here, it is more obvious that there has been a down-turn at the end of the series, and that the orders in 2009 were unusually low (corresponding to some large negative values in the remainder component).

elecequip %>%   
 stl(t.window=13, s.window="periodic",robust = TRUE) %>%   
 autoplot()



The electrical equipment orders (top) and its three additive components obtained from a robust STL decomposition with flexible trend-cycle and fixed seasonality.

The two main parameters to be chosen when using STL are the trend-cycle window (t.window) and the seasonal window (s.window). These control how rapidly the trend-cycle and seasonal components can change. Smaller values allow for more rapid changes. Both t.window and s.window should be odd numbers and refer to the number of consecutive years to be used when estimating the trend-cycle and seasonal components respectively. The user must specify s.window as there is no default. Setting it to to be infinite is equivalent to forcing the seasonal component to be periodic (i.e., identical across years). Specifying t.window is optional, and a default value will be used if it is omitted.

As with the other decomposition methods discussed in this book, to obtain the separate components plotted in Figure @ref(fig:classical-elecequip), use the seasonal() function for the seasonal component, the trendcycle() function for trend-cycle component, and the remainder() function for the remainder component. The seasadj() function can be used to compute the seasonally adjusted series.

## Forecasting with decomposition

While decomposition is primarily useful for studying time series data, and exploring the historical changes over time, it can also be used in forecasting.

Assuming an additive decomposition, the decomposed time series can be written as

where is the seasonally adjusted component. Or, if a multiplicative decomposition has been used, we can write

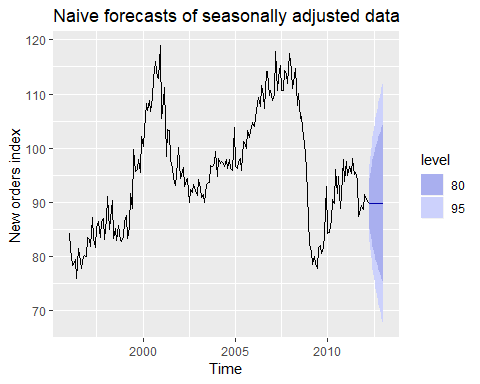
where .

To forecast a decomposed time series, we forecast the seasonal component, , and the seasonally adjusted component , separately. It is usually assumed that the seasonal component is unchanging, or changing extremely slowly, so it is forecast by simply taking the last year of the estimated component. In other words, a seasonal naïve method is used for the seasonal component.

To forecast the seasonally adjusted component, any non-seasonal forecasting method may be used. For example, a random walk with drift model, or Holt’s method (discussed in the next chapter), or a non-seasonal ARIMA model (discussed in Chapter @ref(ch-arima)), may be used.

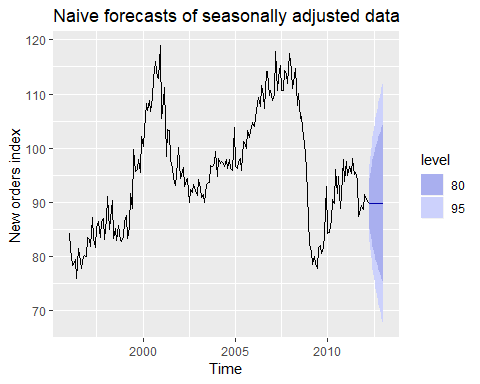
### Example: Electrical equipment manufacturing

fit <- stl(elecequip, t.window=13, s.window="periodic", robust=TRUE)  
  
fit %>% seasadj() %>% naive() %>% autoplot() + ylab("New orders index") +  
 ggtitle("Naive forecasts of seasonally adjusted data")



Naive forecasts of the seasonally adjusted data obtained from an STL decomposition of the electrical equipment orders data.

fit <- stl(elecequip, t.window=13, s.window="periodic", robust=TRUE)  
  
fit %>% seasadj() %>% naive() %>% autoplot() + ylab("New orders index") +  
 ggtitle("Naive forecasts of seasonally adjusted data")

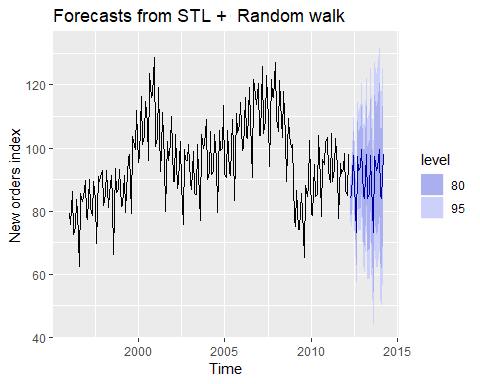


Naive forecasts of the seasonally adjusted data obtained from an STL decomposition of the electrical equipment orders data.

Figure @ref(fig:elecequip4) shows naïve forecasts of the seasonally adjusted electrical equipment orders data. These are then “reseasonalized” by adding in the seasonal naïve forecasts of the seasonal component.

This is made easy with the forecast function applied to the stl object. You need to specify the method being used on the seasonally adjusted data, and the function will do the re-seasonalizing for you. The resulting forecasts of the original data are shown in Figure @ref(fig:elecequip5).

fit %>% forecast(method="naive") %>% autoplot() + ylab("New orders index")



Forecasts of the electrical equipment orders data based on a naive forecast of the seasonally adjusted data and a seasonal naive forecast of the seasonal component, after an STL decomposition of the data.

The prediction intervals shown in this graph are constructed in the same way as the point forecasts. That is, the upper and lower limits of the prediction intervals on the seasonally adjusted data are “reseasonalized” by adding in the forecasts of the seasonal component. In this calculation, the uncertainty in the forecasts of the seasonal component has been ignored. The rationale for this choice is that the uncertainty in the seasonal component is much smaller than that for the seasonally adjusted data, and so it is a reasonable approximation to ignore it.

A short-cut approach is to use the stlf function. The following code will decompose the time series using STL, forecast the seasonally adjusted series, and return reseasonalize the forecasts.

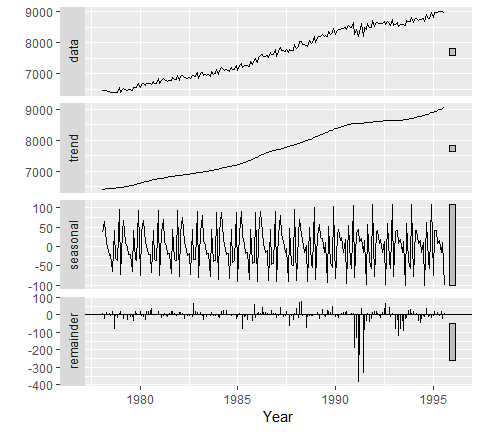
fcast <- stlf(eeadj, method='naive')

The stlf function uses default values for s.window and t.window.

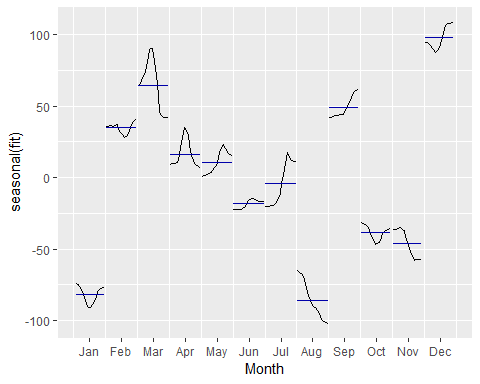
As well as the naïve method, several other possible forecasting methods are available with stlf, as described in the corresponding help file. If method is not specified, it will use the ETS approach (discussed in the next chapter) applied to the seasonally adjusted series. This usually produces quite good forecasts for seasonal time series, and some companies use it routinely for all their operational forecasts.

## Exercises

1. Show that a MA is equivalent to a 7-term weighted moving average with weights of 0.067, 0.133, 0.200, 0.200, 0.200, 0.133, and 0.067.
2. The plastics data set consists of the monthly sales (in thousands) of product A for a plastics manufacturer for five years.
   1. Plot the time series of sales of product A. Can you identify seasonal fluctuations and/or a trend-cycle?
   2. Use a classical multiplicative decomposition to calculate the trend-cycle and seasonal indices.
   3. Do the results support the graphical interpretation from part (a)?
   4. Compute and plot the seasonally adjusted data.
   5. Change one observation to be an outlier (e.g., add 500 to one observation), and recompute the seasonally adjusted data. What is the effect of the outlier?
   6. Does it make any difference if the outlier is near the end rather than in the middle of the time series?
3. Recall your retail time series data (from Exercise 3 in Section @ref(ex-graphics)). Decompose the series using X11. Does it reveal any outliers, or unusual features that you had not noticed previously?
4. Figures @ref(fig:labour) and @ref(fig:labour2) shows the result of decomposing the number of persons in the civilian labor force in Australia each month from February 1978 to August 1995.



Decomposition of the number of persons in the civilian labor force in Australia each month from February 1978 to August 1995.



Seasonal component from the decomposition shown in Figure @ref(fig:labour).

1. Write about 3–5 sentences describing the results of the seasonal adjustment. Pay particular attention to the scales of the graphs in making your interpretation.
2. Is the recession of 1991/1992 visible in the estimated components?
3. This exercise uses the cangas data (monthly Canadian gas production in billions of cubic metres, January 1960 – February 2005).
   1. Plot the data using autoplot, ggsubseriesplot and ggseasonplot to look at the effect of the changing seasonality over time. What do you think is causing it to change so much?
   2. Do an STL decomposition of the data. You will need to choose s.window to allow for the changing shape of the seasonal component.
   3. Compare the results with those obtained using SEATS and X11. How are they different?
4. We will use the bricksq data (Australian quarterly clay brick production. 1956–1994) for this exercise.
   1. Use an STL decomposition to calculate the trend-cycle and seasonal indices. (Experiment with having fixed or changing seasonality.)
   2. Compute and plot the seasonally adjusted data.
   3. Use a naïve method to produce forecasts of the seasonally adjusted data.
   4. Use stlf to reseasonalize the results, giving forecasts for the original data.
   5. Do the residuals look uncorrelated?
   6. Repeat with a robust STL decomposition. Does it make much difference?
   7. Compare forecasts from stlf with those from snaive, using a test set comprising the last 2 years of data. Which is better?
5. Use stlf to produce forecasts of the writing series with either method="naive" or method="rwdrift", whichever is most appropriate. Use the lambda argument if you think a Box-Cox transformation is required.
6. Use stlf to produce forecasts of the fancy series with either method="naive" or method="rwdrift", whichever is most appropriate. Use the lambda argument if you think a Box-Cox transformation is required.

## Further reading

A detailed modern discussion of SEATS and X11 decomposition methods is provided by @Dagum2016. @Cleveland1990 introduced STL, and still provides the best description of the algorithm. For a discussion of forecasting using STL, see @Theodosiou2011.

Dagum, E. B., & Bianconcini, S. (2016). Seasonal adjustment methods and real time trend-cycle estimation. Springer.

# Exponential smoothing

Exponential smoothing was proposed in the late 1950s (@Brown59, @Holt57, and @Winters60 are key pioneering works), and has motivated some of the most successful forecasting methods. Forecasts produced using exponential smoothing methods are weighted averages of past observations, with the weights decaying exponentially as the observations get older. In other words, the more recent the observation the higher the associated weight. This framework generates reliable forecasts quickly and for a wide range of time series, which is a great advantage and of major importance to applications in industry.

This chapter is divided into two parts. In the first part (Sections @ref(sec-7-1-SES)–@ref(sec-7-6-Taxonomy)) we present the mechanics of the most important **exponential smoothing methods**, and their application in forecasting time series with various characteristics. This helps us develop an intuition to how these methods work. In this setting, selecting and using a forecasting method may appear to be somewhat ad hoc. The selection of the method is generally based on recognising key components of the time series (trend and seasonal) and the way in which these enter the smoothing method (e.g., in an additive, damped or multiplicative manner).

In the second part of the chapter (Section @ref(sec-7-ETS)) we present the **statistical models** that underlie exponential smoothing methods. These models generate identical point forecasts to the methods discussed in the first part of the chapter, but also generate prediction intervals. Furthermore, this statistical framework allows for genuine model selection between competing models.

## Bibliography

Brown, R. G. (1959). Statistical forecasting for inventory control. McGraw/Hill.

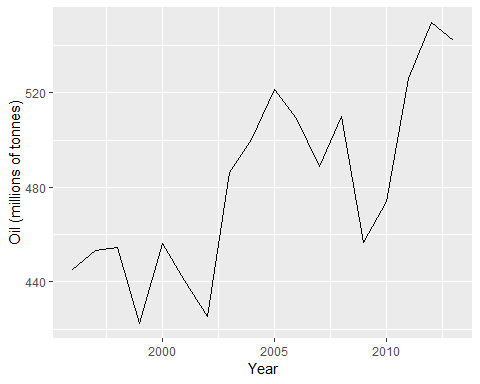
Holt, C. E. (1957). Forecasting seasonals and trends by exponentially weighted averages (O.N.R. Memorandum No. 52). Carnegie Institute of Technology, Pittsburgh USA. <https://doi.org/10.1016/j.ijforecast.2003.09.015>

Winters, P. R. (1960). Forecasting sales by exponentially weighted moving averages. Management Science, 6, 324–342. <https://doi.org/10.1287/mnsc.6.3.324>

## Simple exponential smoothing

The simplest of the exponentially smoothing methods is naturally called **“simple exponential smoothing” (SES)**. This method is suitable for forecasting data with **no** clear trend or seasonal pattern. For example, the data in Figure @ref(fig:7-oil) do not display any clear trending behaviour or any seasonality. (There is a rise in the last few years, which might suggest a trend. We will consider whether a trended method would be better for this series later in this chapter.) We have already considered the naïve and the average as possible methods for forecasting such data (Section @ref(sec-2-methods)).

oildata <- window(oil, start=1996)  
autoplot(oildata)+  
 ylab("Oil (millions of tonnes)")+xlab("Year")



Oil production in Saudi Arabia from 1980 to 2013

Using the **naive method**, all forecasts for the future are equal to the last observed value of the series.

for . Hence, the naïve method assumes that the most recent observation is the only important one, and all previous observations provide no information for the future. This can be thought of as a weighted average where all of the weight is given to the last observation.

Using the **average method**, all future forecasts are equal to a simple average of the observed data.

for . Hence, the average method assumes that all observations are of equal importance, and gives them equal weights when generating forecasts.

We often want something between these two extremes. For example, it may be sensible to attach larger weights to more recent observations than to observations from the distant past. This is exactly the concept behind simple exponential smoothing. Forecasts are calculated using weighted averages, where the weights decrease exponentially as observations come from further in the past — the smallest weights are associated with the oldest observations:

where is the smoothing parameter. The one-step-ahead forecast for time is a weighted average of all of the observations in the series . The rate at which the weights decrease is controlled by the parameter .

Table @ref(tab:alpha) shows the weights attached to observations for four different values of when forecasting using simple exponential smoothing. Note that the sum of the weights even for a small value of will be approximately one for any reasonable sample size.

Exponentially decaying weights attached to observations of time series when generating forecasts using simple exponential smoothing

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  |  |  |  |
|  | 0.2000 | 0.4000 | 0.6000 | 0.8000 |
|  | 0.1600 | 0.2400 | 0.2400 | 0.1600 |
|  | 0.1280 | 0.1440 | 0.0960 | 0.0320 |
|  | 0.1024 | 0.0864 | 0.0384 | 0.0064 |
|  | 0.0819 | 0.0518 | 0.0154 | 0.0013 |
|  | 0.0655 | 0.0311 | 0.0061 | 0.0003 |

For any between 0 and 1, the weights attached to the observations decrease exponentially as we go back in time, hence the name “exponential smoothing”. If is small (i.e., close to 0), more weight is given to observations from the more distant past. If is large (i.e., close to 1), more weight is given to the more recent observations. For the extreme case where , , and the forecasts are equal to the naïve forecasts.

We present two equivalent forms of simple exponential smoothing, each of which leads to the forecast equation @ref(eq:7-ses).

### Weighted average form

The forecast at time is equal to a weighted average between the most recent observation and the most recent forecast ,

for , where is the smoothing parameter.

The process has to start somewhere, so we let the first forecast of be denoted by (which we will have to estimate). Then

Substituting each equation into the following equation, we obtain

The last term becomes tiny for large . So, the weighted average form leads to the same forecast equation @ref(eq:7-ses).

### Component form

An alternative representation is the component form. For simple exponential smoothing, the only component included is the level, . (Other methods which are considered later in this chapter may also include a trend and a seasonal component .) Component form representations of exponential smoothing methods comprise a forecast equation and a smoothing equation for each of the components included in the method. The component form of simple exponential smoothing is given by:

where is the level (or the smoothed value) of the series at time . The forecast equation shows that the forecast value at time is the estimated level at time . The smoothing equation for the level (usually referred to as the level equation) gives the estimated level of the series at each period .

Applying the forecast equation for time gives , the most recent estimated level.

If we replace with and with in the smoothing equation, we will recover the weighted average form of simple exponential smoothing.

The component form of simple exponential smoothing is not particularly useful, but it will be the easiest form to use when we start adding other components.

### Flat forecast

o far, we have only given forecast equations for one step ahead. Simple exponential smoothing has a “flat” forecast function, and therefore for longer forecast horizons,

Remember that these forecasts will only be suitable if the time series has no trend or seasonal component.

### Optimization

The application of every *exponential smoothing method* requires the **smoothing parameters** and the initial values to be chosen. In particular, for simple exponential smoothing, we need to select the values of and . All forecasts can be computed from the data once we know those values. For the methods that follow there is usually more than one smoothing parameter and more than one initial component to be chosen.

In some cases, the smoothing parameters may be chosen in a subjective manner — the forecaster specifies the value of the smoothing parameters based on previous experience. However, a more reliable and objective way to obtain values for the unknown parameters is to estimate them from the observed data.

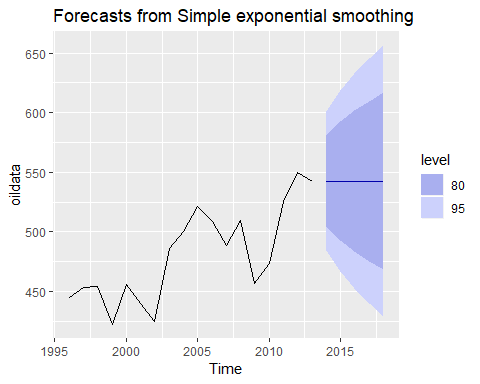
In Section @ref(Regr-LSprinciple), we estimated the coefficients of a regression model by minimizing the sum of the squared errors (SSE). Similarly, the unknown parameters and the initial values for any exponential smoothing method can be estimated by minimizing the SSE. The errors are specified as for (the one-step-ahead training errors). Hence, we find the values of the unknown parameters and the initial values that minimize

Unlike the regression case (where we have formulas which return the values of the regression coefficients that minimize the SSE), this involves a non-linear minimization problem, and we need to use an optimization tool to solve it.

### Example: Oil production

In this example, simple exponential smoothing is applied to forecast oil production in Saudi Arabia.

oildata <- window(oil, start=1996)  
# Estimate parameters  
fc <- ses(oildata, h=5)  
autoplot(fc)



# Accuracy of one-step-ahead training errors over period 1--12  
round(accuracy(fc),2) %>% knitr::kable()

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | ME | RMSE | MAE | MPE | MAPE | MASE | ACF1 |
| Training set | 6.4 | 28.12 | 22.26 | 1.1 | 4.61 | 0.93 | -0.03 |

#tmp <- accuracy(fc)  
#print(round(c(tmp[,c("MAE","RMSE","MAPE")],SSE=sum(residuals(fc)^2)),1))  
alpha <- fc$model$par[1]  
l0 <- fc$model$par[2]

This gives parameters and , obtained by minimizing SSE (equivalently RMSE) over periods , subject to the restriction that .

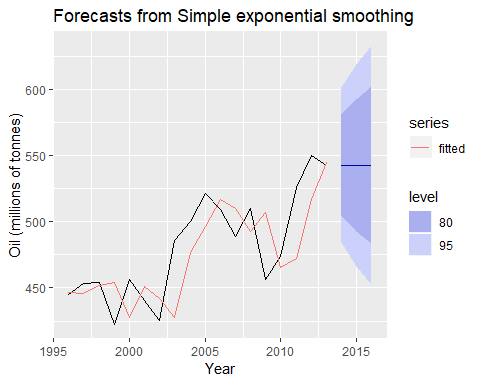
In Table @ref(tab:oilses) we demonstrate the calculation using these parameters. The second last column shows the estimated level for times to ; the last few rows of the last column show the forecasts for .

Forecasting the total oil production in millions of tonnes for Saudi Arabia using simple exponential smoothing.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Year | Time | Observation | Level | Forecast |
|  |  |  |  |  |
| 1995 | 0 |  | 446.59 |  |
| 1996 | 1 | 445.36 | 445.57 | 446.59 |
| 1997 | 2 | 453.20 | 451.93 | 445.57 |
| 1998 | 3 | 454.41 | 454.00 | 451.93 |
| 1999 | 4 | 422.38 | 427.63 | 454.00 |
| 2000 | 5 | 456.04 | 451.32 | 427.63 |
| 2001 | 6 | 440.39 | 442.20 | 451.32 |
| 2002 | 7 | 425.19 | 428.02 | 442.20 |
| 2003 | 8 | 486.21 | 476.54 | 428.02 |
| 2004 | 9 | 500.43 | 496.46 | 476.54 |
| 2005 | 10 | 521.28 | 517.15 | 496.46 |
| 2006 | 11 | 508.95 | 510.31 | 517.15 |
| 2007 | 12 | 488.89 | 492.45 | 510.31 |
| 2008 | 13 | 509.87 | 506.98 | 492.45 |
| 2009 | 14 | 456.72 | 465.07 | 506.98 |
| 2010 | 15 | 473.82 | 472.36 | 465.07 |
| 2011 | 16 | 525.95 | 517.05 | 472.36 |
| 2012 | 17 | 549.83 | 544.39 | 517.05 |
| 2013 | 18 | 542.34 | 542.68 | 544.39 |
|  |  |  |  |  |
| 2014 | 1 |  |  | 542.68 |
| 2015 | 2 |  |  | 542.68 |
| 2016 | 3 |  |  | 542.68 |

The black line in Figure @ref(fig:ses) is a plot of the data, which shows a changing level over time.

autoplot(fc)+  
 forecast::autolayer(fitted(fc),series="fitted")+  
 ylab("Oil (millions of tonnes)")+xlab("Year")



Simple exponential applied to oil production in Saudi Arabia (1996–2013)

The forecasts for the period 2014–2018 are plotted in Figure @ref(fig:ses). Also plotted are one-step-ahead fitted values alongside the data over the period 1996–2013. The large value of in this example is reflected in the large adjustment that takes place in the estimated level at each time. A smaller value of would lead to smaller changes over time, and so the series of fitted values would be smoother.

The prediction intervals shown here are calculated using the methods described in Section @ref(sec-7-ETS). The prediction intervals show that there is considerable uncertainty in the future values of oil production over the five-year forecast period. So interpreting the point forecasts without accounting for the large uncertainty can be very misleading.

## Trend methods

### Holt’s linear trend method

@Holt57 extended simple exponential smoothing to allow the forecasting of data with a trend. This method involves a forecast equation and two smoothing equations (one for the level and one for the trend):

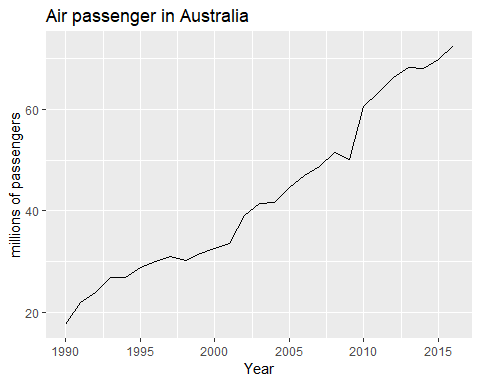
where denotes an estimate of the level of the series at time , denotes an estimate of the trend (slope) of the series at time , is the smoothing parameter for the level, , and is the smoothing parameter for the trend, . (We denote this as instead of for reasons that will be explained in Section @ref(sec-7-ETS).)

As with simple exponential smoothing, the level equation here shows that is a weighted average of observation and the one-step-ahead training forecast for time , here given by . The trend equation shows that is a weighted average of the estimated trend at time based on and , the previous estimate of the trend.

The forecast function is no longer flat but trending. The -step-ahead forecast is equal to the last estimated level plus times the last estimated trend value. Hence the forecasts are a linear function of .

### Example: Air passengers

air <- window(ausair, start=1990)  
  
air %>% autoplot()+  
 ggtitle("Air passenger in Australia")+  
 xlab("Year")+ylab("millions of passengers")



Total annual passengers of air carriers registed in Australia. 1990-2014

Figure @ref(fig:airholt0) shows annual passenger numbers for Australian airlines. In Table @ref(tab:airholt) we demonstrate the application of Holt’s method to these data. The smoothing parameters, and , and the initial values and are estimated by minimizing the SSE for the one-step training errors as in Section @ref(sec-7-1-SES).

fc <- holt(air,h=5)

Applying Holt’s linear method with and to Australian air passenger data (millions of passengers).

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Year | Time | Observation | Level | Slope | Forecast |
|  |  |  |  |  |  |
| 1989 | 0 |  | 15.57 | 2.102 |  |
| 1990 | 1 | 17.55 | 17.57 | 2.102 | 17.67 |
| 1991 | 2 | 21.86 | 21.49 | 2.102 | 19.68 |
| 1992 | 3 | 23.89 | 23.84 | 2.102 | 23.59 |
| 1993 | 4 | 26.93 | 26.76 | 2.102 | 25.94 |
| 1994 | 5 | 26.89 | 27.22 | 2.102 | 28.86 |
| 1995 | 6 | 28.83 | 28.92 | 2.102 | 29.33 |
| 1996 | 7 | 30.08 | 30.24 | 2.102 | 31.02 |
| 1997 | 8 | 30.95 | 31.19 | 2.102 | 32.34 |
| 1998 | 9 | 30.19 | 30.71 | 2.101 | 33.29 |
| 1999 | 10 | 31.58 | 31.79 | 2.101 | 32.81 |
| 2000 | 11 | 32.58 | 32.80 | 2.101 | 33.89 |
| 2001 | 12 | 33.48 | 33.72 | 2.101 | 34.90 |
| 2002 | 13 | 39.02 | 38.48 | 2.101 | 35.82 |
| 2003 | 14 | 41.39 | 41.25 | 2.101 | 40.58 |
| 2004 | 15 | 41.60 | 41.89 | 2.101 | 43.35 |
| 2005 | 16 | 44.66 | 44.54 | 2.101 | 44.00 |
| 2006 | 17 | 46.95 | 46.90 | 2.101 | 46.65 |
| 2007 | 18 | 48.73 | 48.78 | 2.101 | 49.00 |
| 2008 | 19 | 51.49 | 51.38 | 2.101 | 50.88 |
| 2009 | 20 | 50.03 | 50.61 | 2.101 | 53.49 |
| 2010 | 21 | 60.64 | 59.30 | 2.102 | 52.72 |
| 2011 | 22 | 63.36 | 63.03 | 2.102 | 61.40 |
| 2012 | 23 | 66.36 | 66.15 | 2.102 | 65.13 |
| 2013 | 24 | 68.20 | 68.21 | 2.102 | 68.25 |
| 2014 | 25 | 68.12 | 68.49 | 2.102 | 70.31 |
| 2015 | 26 | 69.78 | 69.92 | 2.102 | 70.60 |
| 2016 | 27 | 72.60 | 72.50 | 2.102 | 72.02 |
|  |  |  |  |  |  |
|  | 1 |  |  |  | 74.60 |
|  | 2 |  |  |  | 76.70 |
|  | 3 |  |  |  | 78.80 |
|  | 4 |  |  |  | 80.91 |
|  | 5 |  |  |  | 83.01 |

### Damped trend methods

The forecasts generated by Holt’s linear method display a constant trend (increasing or decreasing) indefinitely into the future. Empirical evidence indicates that these methods tend to over-forecast, especially for longer forecast horizons. Motivated by this observation, @GarMacK1985 introduced a parameter that “dampens” the trend to a flat line some time in the future. Methods that include a damped trend have proven to be very successful, and are arguably the most popular individual methods when forecasts are required automatically for many series.

In conjunction with the smoothing parameters and (with values between 0 and 1 as in Holt’s method), this method also includes a damping parameter :

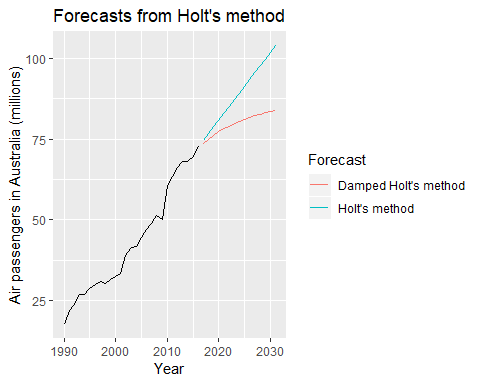
If , the method is identical to Holt’s linear method. For values between and , dampens the trend so that it approaches a constant some time in the future. In fact, the forecasts converge to as for any value . This means that short-run forecasts are trended while long-run forecasts are constant.

In practice, is rarely less than 0.8 as the damping has a very strong effect for smaller values. Values of close to 1 will mean that a damped model is not able to be distinguished from a non-damped model. For these reasons, we usually restrict to a minimum of 0.8 and a maximum of 0.98.

#### Example: Air passengers (continued)

Figure @ref(fig:dampedtrend) shows the forecasts for years 2014–2018 generated from Holt’s linear trend method and the damped trend method.

fc <- holt(air, h=15)  
fc2 <- holt(air, damped=TRUE, phi = 0.9, h=15)  
  
autoplot(air) +  
 forecast::autolayer(fc$mean, series="Holt's method") +  
 forecast::autolayer(fc2$mean, series="Damped Holt's method") +  
 ggtitle("Forecasts from Holt's method") +  
 xlab("Year") + ylab("Air passengers in Australia (millions)") +  
 guides(colour=guide\_legend(title="Forecast"))



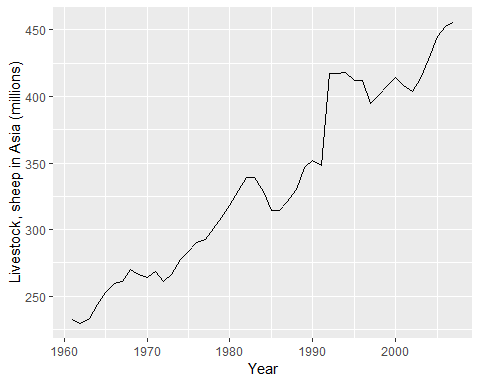
Forecasting Air Passengers in Australia (millions of passengers). For the damped trend method, .

We have set the damping parameter to a relatively low number to exaggerate the effect of damping for comparison. Usually, we would estimate along with the other parameters.

### Example: Sheep in Asia

In this example, we compare the forecasting performance of the three exponential smoothing methods that we have considered so far in forecasting the sheep livestock population in Asia. The data spans the period 1970–2007 and is shown in Figure 7.4.

library(fpp2)  
library(forecast)  
  
autoplot(livestock) +  
 xlab("Year") + ylab("Livestock, sheep in Asia (millions)")



Annual sheep livestock numbers in Asia (in million head)

We will use time series cross-validation to compare the one-step forecast accuracy of the three methods.

e1 <- tsCV(livestock,ses,h=1)  
e2 <- tsCV(livestock, holt, h=1)  
e3 <- tsCV(livestock, holt, damped=TRUE, h=1)  
  
# Compare MSE:  
mean(e1^2, na.rm=T)

## [1] 178.2531

mean(e2^2, na.rm=T)

## [1] 173.365

mean(e3^2, na.rm=T)

## [1] 162.6274

# Comare MAE:  
mean(abs(e1),na.rm=TRUE)

## [1] 8.53246

mean(abs(e2),na.rm=TRUE)

## [1] 8.803058

mean(abs(e3),na.rm=TRUE)

## [1] 8.024192

Based on MSE, Holt’s method is best. But based on MAE, simple exponential smoothing is best. Conflicts such as this are common in forecasting comparisons. As forecasting tasks can vary by many dimensions (length of forecast horizon, size of test set, forecast error measures, frequency of data, etc.), it is unlikely that one method will be better than all others for all forecasting scenarios. What we require from a forecasting method are consistently sensible forecasts, and these should be frequently evaluated against the task at hand. In this case, the data are clearly trended, so we will prefer Holt’s method, and apply it to the whole data set to get forecasts for future years.

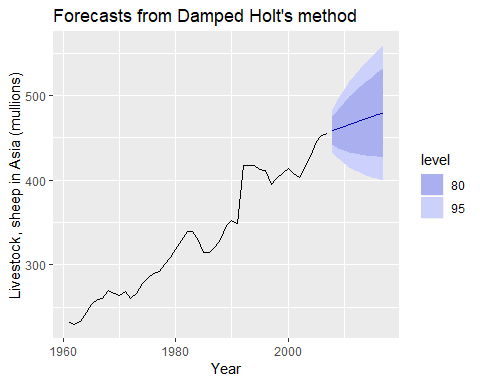
Damped Holt’s method is best whether you compare MAE or MSE values. So we will proceed with using the damped Holt’s method and apply it to the whole data set to get forecasts fo your futures.

fc <- holt(livestock, damped=TRUE)  
# Estimated parameters:  
fc[["model"]]

## Damped Holt's method   
##   
## Call:  
## holt(y = livestock, damped = TRUE)   
##   
## Smoothing parameters:  
## alpha = 0.9999   
## beta = 3e-04   
## phi = 0.9798   
##   
## Initial states:  
## l = 223.35   
## b = 6.9046   
##   
## sigma: 12.8435  
##   
## AIC AICc BIC   
## 427.6370 429.7370 438.7379

The smoothing parameter for the slope is estimated to be essentially zero, indicating that the trend is not changin over time. The value of is very close to one, showing the level reacts strongly to each new observation.

autoplot(fc)+  
 xlab("Year")+ylab("Livestock, sheep in Asia (mullions)")



The resulting forecasts look sensible with increasing trend, and relatively wide prediction intervals reflecting the variation in the historical data. The prediction intervals are calculated using the methods described in Section 7.5.

In this example, the process of selecting a method was relatively easy as both MSE and MAE comparisons suggested the same method (damped Holt’s). However, sometimes different accuracy measures will suggest different forecasting methods, and then a decision is required as to which forecasting method we prefer to use. As forecasting tasks can vary by many dimensions (length of forecast horizon, size of test set, forecast error measures, frequency of data, etc.), it is unlikely that one method will be better than all others for all forecasting scenarios. What we require from a forecasting method are consistently sensible forecasts, and these should be frequently evaluated against the task at hand.

### Bibliography

Holt, C. E. (1957). Forecasting seasonals and trends by exponentially weighted averages (O.N.R. Memorandum No. 52). Carnegie Institute of Technology, Pittsburgh USA. <https://doi.org/10.1016/j.ijforecast.2003.09.015>

Gardner, E. S., & McKenzie, E. (1985). Forecasting trends in time series. Management Science, 31(10), 1237–1246. <https://doi.org/10.1287/mnsc.31.10.1237>

## Holt-Winter’s seasonal method

@Holt57 and @Winters60 extended Holt’s method to capture seasonality. The Holt-Winters seasonal method comprises the forecast equation and three smoothing equations — one for the level , one for the trend , and one for the seasonal component , with corresponding smoothing parameters , and . We use to denote the frequency of the seasonality, i.e., the number of seasons in a year. For example, for quarterly data , and for monthly data .

There are two variations to this method that differ in the nature of the seasonal component. The additive method is preferred when the seasonal variations are roughly constant through the series, while the multiplicative method is preferred when the seasonal variations are changing proportional to the level of the series. With the additive method, the seasonal component is expressed in absolute terms in the scale of the observed series, and in the level equation the series is seasonally adjusted by subtracting the seasonal component. Within each year, the seasonal component will add up to approximately zero. With the multiplicative method, the seasonal component is expressed in relative terms (percentages), and the series is seasonally adjusted by dividing through by the seasonal component. Within each year, the seasonal component will sum up to approximately .

### Holt-Winter’s additive method

The component form for the additive method is:

where ,[[1]](#footnote-182) which ensures that the estimates of the seasonal indices used for forecasting come from the final year of the sample. The level equation shows a weighted average between the seasonally adjusted observation and the non-seasonal forecast for time . The trend equation is identical to Holt’s linear method. The seasonal equation shows a weighted average between the current seasonal index, , and the seasonal index of the same season last year (i.e., time periods ago).

The equation for the seasonal component is often expressed as

If we substitute from the smoothing equation for the level of the component form above, we get

which is identical to the smoothing equation for the seasonal component we specify here, with . The usual parameter restriction is , which translates to .

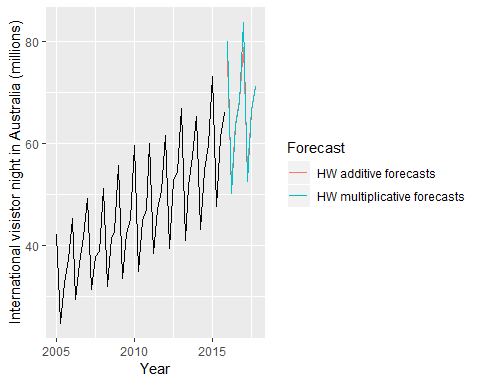
### Holt-Winters’ multiplicative method

The component form for the multiplicative method is:

### Example: International tourist visitor nights in Australia

In this example, we employ the Holt-Winters method with both additive and multiplicative seasonality to forecast quarterly visitor nights in Australia spent by international tourists. Figure @ref(fig:7-HW) shows the data from 2005, and the forecasts for 2016–2017. The data show an obvious seasonal pattern, with peaks observed in the March quarter of each year, corresponding to the Australian summer.

library(fpp2)  
library(forecast)  
aust <- window(austourists, start=2005)  
  
fit1 <- hw(aust,seasonal="additive")  
fit2 <- hw(aust,seasonal="multiplicative")  
  
autoplot(aust)+  
 forecast::autolayer(fit1$mean, series="HW additive forecasts")+  
 forecast::autolayer(fit2$mean, series="HW multiplicative forecasts")+  
 xlab("Year")+ylab("International visistor night in Australia (millions)")+  
 guides(color=guide\_legend(title="Forecast"))



Forecasting international visitor nights in Australia using the Holt-Winters method with both additive and multiplicative seasonality.

Applying Holt-Winters’ method with additive seasonality for forecasting international visitor nights in Australia. Notice that the additive seasonal component sums to approximately zero. The smoothing parameters and initial estimates for the components have been estimated by minimizing RMSE (, , and RMSE).

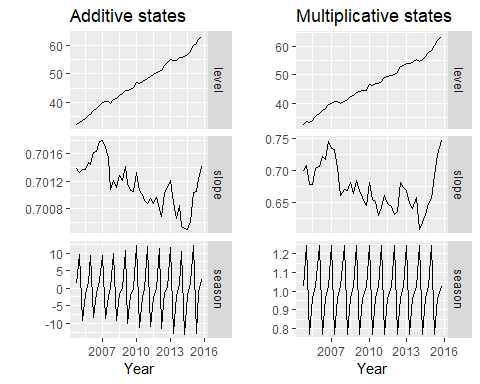
|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |
| 2004 Q1 | -3 |  |  |  | -1.69 |  |
| 2004 Q2 | -2 |  |  |  | -9.31 |  |
| 2004 Q3 | -1 |  |  |  | 9.70 |  |
| 2004 Q4 | 0 |  | 32.26 | 0.70 | 1.31 |  |
| 2005 Q1 | 1 | 42.21 | 32.82 | 0.70 | 9.50 | 42.66 |
| 2005 Q2 | 2 | 24.65 | 33.66 | 0.70 | -9.13 | 24.21 |
| 2005 Q3 | 3 | 32.67 | 34.36 | 0.70 | -1.69 | 32.67 |
| 2005 Q4 | 4 | 37.26 | 35.33 | 0.70 | 1.69 | 36.37 |
| 2006 Q1 | 5 | 45.24 | 35.94 | 0.70 | 9.38 | 45.54 |
| 2006 Q2 | 6 | 29.35 | 37.21 | 0.70 | -8.35 | 27.52 |
| 2006 Q3 | 7 | 36.34 | 37.95 | 0.70 | -1.64 | 36.21 |
| 2006 Q4 | 8 | 41.78 | 39.09 | 0.70 | 2.30 | 40.34 |
| 2007 Q1 | 9 | 49.28 | 39.83 | 0.70 | 9.42 | 49.17 |
| 2007 Q2 | 10 | 31.28 | 40.25 | 0.70 | -8.73 | 32.18 |
| 2007 Q3 | 11 | 37.85 | 40.50 | 0.70 | -2.26 | 39.31 |
| 2007 Q4 | 12 | 38.84 | 39.77 | 0.70 | 0.31 | 43.51 |
| 2008 Q1 | 13 | 51.24 | 40.89 | 0.70 | 9.99 | 49.90 |
| 2008 Q2 | 14 | 31.84 | 41.28 | 0.70 | -9.17 | 32.85 |
| 2008 Q3 | 15 | 41.32 | 42.47 | 0.70 | -1.58 | 39.71 |
| 2008 Q4 | 16 | 42.80 | 42.96 | 0.70 | 0.02 | 43.48 |
| 2009 Q1 | 17 | 55.71 | 44.29 | 0.70 | 10.87 | 53.66 |
| 2009 Q2 | 18 | 33.41 | 44.25 | 0.70 | -10.20 | 35.83 |
| 2009 Q3 | 19 | 42.32 | 44.63 | 0.70 | -2.03 | 43.38 |
| 2009 Q4 | 20 | 45.16 | 45.27 | 0.70 | -0.06 | 45.35 |
| 2010 Q1 | 21 | 59.58 | 46.81 | 0.70 | 12.04 | 56.84 |
| 2010 Q2 | 22 | 34.84 | 46.75 | 0.70 | -11.25 | 37.31 |
| 2010 Q3 | 23 | 44.84 | 47.27 | 0.70 | -2.28 | 45.43 |
| 2010 Q4 | 24 | 46.97 | 47.69 | 0.70 | -0.46 | 47.91 |
| 2011 Q1 | 25 | 60.02 | 48.26 | 0.70 | 11.86 | 60.42 |
| 2011 Q2 | 26 | 38.37 | 49.17 | 0.70 | -10.97 | 37.71 |
| 2011 Q3 | 27 | 46.98 | 49.68 | 0.70 | -2.54 | 47.59 |
| 2011 Q4 | 28 | 50.73 | 50.63 | 0.70 | -0.11 | 49.92 |
| 2012 Q1 | 29 | 61.65 | 50.86 | 0.70 | 11.20 | 63.19 |
| 2012 Q2 | 30 | 39.30 | 51.16 | 0.70 | -11.52 | 40.59 |
| 2012 Q3 | 31 | 52.67 | 52.89 | 0.70 | -1.11 | 49.33 |
| 2012 Q4 | 32 | 54.33 | 53.85 | 0.70 | 0.25 | 53.48 |
| 2013 Q1 | 33 | 66.83 | 54.88 | 0.70 | 11.66 | 65.76 |
| 2013 Q2 | 34 | 40.87 | 54.61 | 0.70 | -12.88 | 44.06 |
| 2013 Q3 | 35 | 51.83 | 54.58 | 0.70 | -2.12 | 54.19 |
| 2013 Q4 | 36 | 57.49 | 55.88 | 0.70 | 1.09 | 55.53 |
| 2014 Q1 | 37 | 65.25 | 55.67 | 0.70 | 10.39 | 68.25 |
| 2014 Q2 | 38 | 43.06 | 56.24 | 0.70 | -13.06 | 43.49 |
| 2014 Q3 | 39 | 54.76 | 56.92 | 0.70 | -2.14 | 54.82 |
| 2014 Q4 | 40 | 59.83 | 57.97 | 0.70 | 1.57 | 58.71 |
| 2015 Q1 | 41 | 73.26 | 59.96 | 0.70 | 12.18 | 69.05 |
| 2015 Q2 | 42 | 47.70 | 60.69 | 0.70 | -13.02 | 47.59 |
| 2015 Q3 | 43 | 61.10 | 61.96 | 0.70 | -1.35 | 59.24 |
| 2015 Q4 | 44 | 66.06 | 63.22 | 0.70 | 2.35 | 64.22 |
|  |  |  |  |  |  |  |
| 2016 Q1 | 1 |  |  |  |  | 76.10 |
| 2016 Q2 | 2 |  |  |  |  | 51.60 |
| 2016 Q3 | 3 |  |  |  |  | 63.97 |
| 2016 Q4 | 4 |  |  |  |  | 68.37 |
| 2017 Q1 | 5 |  |  |  |  | 78.90 |
| 2017 Q2 | 6 |  |  |  |  | 54.41 |
| 2017 Q3 | 7 |  |  |  |  | 66.77 |
| 2017 Q4 | 8 |  |  |  |  | 71.18 |

Applying Holt-Winters’ method with multiplicative seasonality for forecasting international visitor nights in Australia. Notice that the multiplicative seasonal component sums to approximately . The smoothing parameters and initial estimates for the components have been estimated by minimizing RMSE (, , and RMSE).

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |
| 2004 Q1 | -3 |  |  |  | 0.96 |  |
| 2004 Q2 | -2 |  |  |  | 0.77 |  |
| 2004 Q3 | -1 |  |  |  | 1.24 |  |
| 2004 Q4 | 0 |  | 32.49 | 0.70 | 1.02 |  |
| 2005 Q1 | 1 | 42.21 | 33.51 | 0.71 | 1.24 | 41.29 |
| 2005 Q2 | 2 | 24.65 | 33.24 | 0.68 | 0.77 | 26.36 |
| 2005 Q3 | 3 | 32.67 | 33.94 | 0.68 | 0.96 | 32.62 |
| 2005 Q4 | 4 | 37.26 | 35.40 | 0.70 | 1.02 | 35.44 |
| 2006 Q1 | 5 | 45.24 | 36.22 | 0.71 | 1.24 | 44.92 |
| 2006 Q2 | 6 | 29.35 | 37.44 | 0.72 | 0.77 | 28.44 |
| 2006 Q3 | 7 | 36.34 | 38.00 | 0.72 | 0.96 | 36.71 |
| 2006 Q4 | 8 | 41.78 | 39.64 | 0.74 | 1.02 | 39.64 |
| 2007 Q1 | 9 | 49.28 | 40.04 | 0.73 | 1.24 | 50.25 |
| 2007 Q2 | 10 | 31.28 | 40.70 | 0.73 | 0.77 | 31.41 |
| 2007 Q3 | 11 | 37.85 | 40.52 | 0.70 | 0.96 | 39.84 |
| 2007 Q4 | 12 | 38.84 | 39.77 | 0.66 | 1.02 | 42.21 |
| 2008 Q1 | 13 | 51.24 | 40.76 | 0.67 | 1.24 | 50.30 |
| 2008 Q2 | 14 | 31.84 | 41.39 | 0.67 | 0.77 | 31.91 |
| 2008 Q3 | 15 | 41.32 | 42.46 | 0.68 | 0.96 | 40.44 |
| 2008 Q4 | 16 | 42.80 | 42.55 | 0.66 | 1.02 | 44.16 |
| 2009 Q1 | 17 | 55.71 | 43.90 | 0.68 | 1.24 | 53.77 |
| 2009 Q2 | 18 | 33.41 | 44.05 | 0.67 | 0.77 | 34.34 |
| 2009 Q3 | 19 | 42.32 | 44.40 | 0.66 | 0.96 | 43.00 |
| 2009 Q4 | 20 | 45.16 | 44.64 | 0.65 | 1.02 | 46.13 |
| 2010 Q1 | 21 | 59.58 | 46.43 | 0.68 | 1.24 | 56.35 |
| 2010 Q2 | 22 | 34.84 | 46.28 | 0.65 | 0.77 | 36.29 |
| 2010 Q3 | 23 | 44.84 | 46.80 | 0.65 | 0.96 | 45.14 |
| 2010 Q4 | 24 | 46.97 | 46.76 | 0.63 | 1.02 | 48.57 |
| 2011 Q1 | 25 | 60.02 | 47.76 | 0.64 | 1.24 | 58.98 |
| 2011 Q2 | 26 | 38.37 | 49.03 | 0.66 | 0.77 | 37.28 |
| 2011 Q3 | 27 | 46.98 | 49.32 | 0.65 | 0.96 | 47.78 |
| 2011 Q4 | 28 | 50.73 | 49.79 | 0.64 | 1.02 | 51.14 |
| 2012 Q1 | 29 | 61.65 | 50.04 | 0.63 | 1.24 | 62.77 |
| 2012 Q2 | 30 | 39.30 | 50.82 | 0.64 | 0.77 | 39.03 |
| 2012 Q3 | 31 | 52.67 | 52.92 | 0.68 | 0.96 | 49.49 |
| 2012 Q4 | 32 | 54.33 | 53.37 | 0.67 | 1.02 | 54.86 |
| 2013 Q1 | 33 | 66.83 | 53.90 | 0.67 | 1.24 | 67.26 |
| 2013 Q2 | 34 | 40.87 | 53.90 | 0.65 | 0.77 | 42.03 |
| 2013 Q3 | 35 | 51.83 | 54.26 | 0.64 | 0.96 | 52.47 |
| 2013 Q4 | 36 | 57.49 | 55.46 | 0.66 | 1.02 | 56.19 |
| 2014 Q1 | 37 | 65.25 | 54.49 | 0.61 | 1.24 | 69.84 |
| 2014 Q2 | 38 | 43.06 | 55.46 | 0.62 | 0.77 | 42.44 |
| 2014 Q3 | 39 | 54.76 | 56.46 | 0.63 | 0.96 | 53.93 |
| 2014 Q4 | 40 | 59.83 | 57.69 | 0.65 | 1.02 | 58.43 |
| 2015 Q1 | 41 | 73.26 | 58.57 | 0.66 | 1.24 | 72.59 |
| 2015 Q2 | 42 | 47.70 | 60.42 | 0.69 | 0.77 | 45.62 |
| 2015 Q3 | 43 | 61.10 | 62.17 | 0.72 | 0.96 | 58.77 |
| 2015 Q4 | 44 | 66.06 | 63.62 | 0.75 | 1.02 | 64.38 |
|  |  |  |  |  |  |  |
| 2016 Q1 | 1 |  |  |  |  | 80.09 |
| 2016 Q2 | 2 |  |  |  |  | 50.15 |
| 2016 Q3 | 3 |  |  |  |  | 63.34 |
| 2016 Q4 | 4 |  |  |  |  | 68.18 |
| 2017 Q1 | 5 |  |  |  |  | 83.80 |
| 2017 Q2 | 6 |  |  |  |  | 52.45 |
| 2017 Q3 | 7 |  |  |  |  | 66.21 |
| 2017 Q4 | 8 |  |  |  |  | 71.23 |

The applications of both methods (with additive and multiplicative seasonality) are presented in Tables @ref(tab:tab75) and @ref(tab:tab76) respectively. Because both methods have exactly the same number of parameters to estimate, we can compare the training RMSE from both models. In this case, the method with multiplicative seasonality fits the data best. This was to be expected, as the time plot shows that the seasonal variation in the data increases as the level of the series increases. This is also reflected in the two sets of forecasts; the forecasts generated by the method with the multiplicative seasonality display larger and increasing seasonal variation as the level of the forecasts increases compared to the forecasts generated by the method with additive seasonality.

The estimated states for both models are plotted in Figure @ref(fig:fig-7-LevelTrendSeas). The small value of for the multiplicative model means that the seasonal component hardly changes over time. The small value of for the additive model means the slope component hardly changes over time (check the vertical scale).



Estimated components for the Holt-Winters method with additive and multiplicative seasonal components.

### Holt-Winter’s damped method

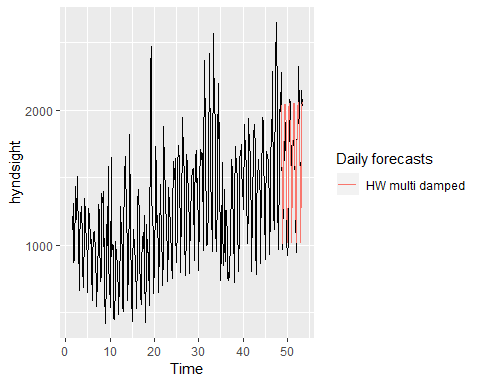
Damping is possible with both additive and multiplicative Holt-Winters’ methods. A method that often provides accurate and robust forecasts for seasonal data is the Holt-Winters method with a damped trend and multiplicative seasonality:

hw(x, damped=TRUE, seasonal="multiplicative")

### Example: Holt-Winters method with daily data

The Holt-Winters method can also be used for daily type of data, where the seasonal pattern is of length 7, and the appropriate unit of time for is in days. Here, we generate daily forecasts for the last five weeks for the hyndsight data, which contains the daily pageviews on the Hyndsight blog for one year starting April 30, 2014.

fc <- hw(subset(hyndsight,end=length(hyndsight)-35),  
 damped = TRUE, seasonal="multiplicative", h=35)  
  
autoplot(hyndsight) +  
 forecast::autolayer(fc$mean, series="HW multi damped")+  
 guides(colour=guide\_legend(title="Daily forecasts"))



Clearly the model has identified the weekly seasonal pattern and the increasing trend at the end of the data, and the forecasts are a close match to the test data.

### Bibliography

Holt, C. E. (1957). Forecasting seasonals and trends by exponentially weighted averages (O.N.R. Memorandum No. 52). Carnegie Institute of Technology, Pittsburgh USA. <https://doi.org/10.1016/j.ijforecast.2003.09.015>

Winters, P. R. (1960). Forecasting sales by exponentially weighted moving averages. Management Science, 6, 324–342. <https://doi.org/10.1287/mnsc.6.3.324>

## A taxonoy of exponential smooothing methods

Exponential smooothing methods are not restricted to those we have presented so far. By considering variations in the combinations of the trend and seasonal components, nine exponential smoothing methods are possible, listed in Table @ref(tab:taxonomy). Each method is labelled by a pair of letters (T,S) defining the type of ‘Trend’ and ‘Seasonal’ components. For example, (A,M) is the method with an additive trend and multiplicative seasonality; (Ad,N) is the method with damped trend and no seasonality; and so on.

(#tab:taxonomy) A two-way classification of exponential smoothing methods.

|  |  |  |  |
| --- | --- | --- | --- |
| Trend Component | Seasonal | Component |  |
|  | N (None) | A (Additive) | M (Multiplicative) |
| N (None) | (N,N) | (N,A) | (N,M) |
| A (Additive) | (A,N) | (A,A) | (A,M) |
| Ad (Additive damped) | (Ad,N) | (Ad,A) | (Ad) |

Some of these methods we have already seen:

|  |  |
| --- | --- |
| Short hand | Method |
| (N,N) | Simple exponential smoothing |
| (A,N) | Holt’s linear method |
| (Ad,N) | Additive damped trend method |
| (A,A) | Additive Holt-Winters’ method |
| (A,M) | Multiplicative Holt-Winters’ method |
| (Ad,M) | Holt-Winters’ damped method |

This type of classification was first proposed by @Pegels1969, who also included a method with a multiplicative trend. It was later extended by @Gar1985 to include methods with an additive damped trend and by @Taylor2003 to include methods with a multiplicative damped trend. We do not consider the multiplicative trend methods in this book as they tend to produce poor forecasts. See @expsmooth08 for a more thorough discussion of all exponential smoothing methods.

Table @ref(tab:pegels) gives the recursive formulae for applying the nine exponential smoothing methods in Table @ref(tab:taxonomy). Each cell includes the forecast equation for generating -step-ahead forecasts, and the smoothing equations for applying the method.

Table: (#tab:pegels) Formulae for recursive calculations and point forecasts. In each case, denotes the series level at time , denotes the slope at time , denotes the seasonal component of the series at time , and denotes the number of seasons in a year; , , and are smoothing parameters, and .

## Innovations state space models for exponential smoothing

In the rest of this chapter, we study the statistical models that underlie **the exponential smoothing methods** we have considered so far. The exponential smoothing methods presented in Table @ref(tab:pegels) are algorithms which generate point forecasts. The statistical models in this section generate the same point forecasts, but can also generate prediction (or forecast) intervals. A statistical model is a **stochastic (or random) data generating process** that can produce an entire forecast distribution. The general statistical framework we will introduce also provides a way of using the model selection criteria introduced in Chapter @ref(ch-regression), thus allowing the choice of model to be made in an objective manner.

Each model consists of a measurement equation that describes the observed data, and some transition equations that describe how the unobserved components or states (level, trend, seasonal) change over time. Hence, these are referred to as **“state space models”**.

For each method there exist two models: one with additive errors and one with multiplicative errors. The point forecasts produced by the models are identical if they use the same smoothing parameter values. They will, however, generate different prediction intervals.

To distinguish between a model with additive errors and one with multiplicative errors (and also to distinguish the models from the methods), we add a third letter to the classification of Table @ref(tab:taxonomy). We label each state space model as ETS() for (Error, Trend, Seasonal). This label can also be thought of as ExponenTial Smoothing. Using the same notation as in Table @ref(tab:taxonomy), the possibilities for each component are: Error A,M, Trend N,A,Ad and Seasonal N,A,M.

### ETS(A,N,N): simple exponential smoothing with additive errors

Recall the component form of simple exponential smooothing

If we re-arrange the smoothing equation for the level, we get the error correction form:

where for . That is, is the one-step error at time computed on the training data.

The training data errors lead to the adjustment of the estimated level throughout the smoothing process for . For example, if the error at time is negative, then and so the level at time has been over-estimated. The new level is then the previous level adjusted downwards. The closer is to one, the “rougher” the estimate of the level (large adjustments take place). The smaller the , the “smoother” the level (small adjustments take place).

We can also write , so that each observation is equal to the previous level plus an error. To make this into an innovations state space model, all we need to do is specify the probability distribution for . For a model with additive errors, we assume that one-step forecast errors are normally distributed white noise with mean 0 and variance . A short-hand notation for this is ; NID stands for “normally and independently distributed”.

Then the equations of the model can be written as

We refer to @ref(eq:ann-1a) as the *measurement* (or observation) equation and @ref(eq:ann-2a) as the *state* (or transition) equation. These two equations, together with the statistical distribution of the errors, form a fully specified statistical model. Specifically, these constitute an innovations state space model underlying simple exponential smoothing.

The term “innovations” comes from the fact that all equations in this type of specification use the same random error process, . For the same reason, this formulation is also referred to as a “single source of error” model, in contrast to alternative multiple source of error formulation swhich we do not present here.

The measurement equation shows the relationship between the observations and the unobserved states. In this case, observation is a linear function of the level , the predictable part of , and the random error , the unpredictable part of . For other innovations state space models, this relationship may be nonlinear.

The transition equation shows the evolution of the state through time. The influence of the smoothing parameter is the same as for the methods discussed earlier. For example, governs the degree of change in successive levels. The higher the value of , the more rapid the changes in the level; the lower the value of , the smoother the changes. At the lowest extreme, where , the level of the series does not change over time. At the other extreme, where , the model reduces to a random walk model, .

### ETS(M,N,N): simple exponential smoothing with multiplicative errors

In a similar fashion, we can specify models with multiplicative errors by writing the one-step random errors as relative errors:

where . Substituting gives and .

Then we can write the multiplicative form of the state space model as

### ETS(A,A,N): Holt’s linear method with additive errors

For this model, we assume that the one-step forecast errors are given by . Substituting this into the error correction equations for Holt’s linear method we obtain

where, for simplicity, we have set .

### ETS(M,A,N): Holt’s linear method with multiplicative errors

Specifying one-step forecast errors as relative errors such that

and following an approach similar to that used above, the innovations state space model underlying Holt’s linear method with multiplicative errors is specified as

where again and .

### Other ETS models

In a similar fashion, we can write an innovations state space model for each of the exponential smoothing methods of Table @ref(tab:pegels). Table @ref(tab:ssm) presents the equations for all of the models in the ETS framework.

Table: (#tab:ssm) State space equations for each of the models in the ETS framework.

## Estimation and model selection

### Estimating ETS models

An alternative to estimating the parameters by minimizing the sum of squared errors is to maximize the “likelihood”. The likelihood is the probability of the data arising from the specified model. Thus, a large likelihood is associated with a good model. For an additive error model, maximizing the likelihood gives the same results as minimizing the sum of squared errors. However, different results will be obtained for multiplicative error models. In this section, we will estimate the smoothing parameters , , and , and the initial states , , , by maximizing the likelihood.

The possible values that the smoothing parameters can take are restricted. Traditionally, the parameters have been constrained to lie between 0 and 1 so that the equations can be interpreted as weighted averages. That is, . For the state space models, we have set and . Therefore, the traditional restrictions translate to , and . In practice, the damping parameter is usually constrained further to prevent numerical difficulties in estimating the model. In R, it is restricted so that .

Another way to view the parameters is through a consideration of the mathematical properties of the state space models. The parameters are constrained in order to prevent observations in the distant past having a continuing effect on current forecasts. This leads to some *admissibility* constraints on the parameters, which are usually (but not always) less restrictive than the usual region. For example, for the ETS(A,N,N) model, the usual parameter region is but the admissible region is . For the ETS(A,A,N) model, the usual parameter region is and but the admissible region is and .

### Model selection

A great advantage of the ETS statistical framework is that information criteria can be used for model selection. The AIC, AIC and BIC, introduced in Section @ref(Regr-SelectingPredictors), can be used here to determine which of the ETS models is most appropriate for a given time series.

For ETS models, Akaike’s Information Criterion (AIC) is defined as

where is the likelihood of the model and is the total number of parameters and initial states that have been estimated (including the residual variance).

The AIC corrected for small sample bias (AIC) is defined as

and the Bayesian Information Criterion (BIC) is

Three of the combinations of (Error, Trend, Seasonal) can lead to numerical difficulties. Specifically, the models that can cause such instabilities are ETS(A,N,M), ETS(A,A,M), and ETS(A,Ad,M). We normally do not consider these particular combinations when selecting a model.

Models with multiplicative errors are useful when the data are strictly positive, but are not numerically stable when the data contain zeros or negative values. Therefore, multiplicative error models will not be considered if the time series is not strictly positive. In that case, only the six fully additive models will be applied.

### The ets() function in R

The models can be estimated in R using the ets() function in the forecast package. Unlike the ses, holt and hw functions, the ets function does not produce forecasts. Rather, it estimates the model parameters and returns information about the fitted model.

The R code below shows the most important arguments that this function can take, and their default values. If only the time series is specified, and all other arguments are left at their default values, then an appropriate model will be selected automatically. We explain the arguments below. See the help file for a more complete description.

ets(y, model="ZZZ", damped=NULL, alpha=NULL, beta=NULL, gamma=NULL,  
 phi=NULL, lambda=NULL, biasadj=FALSE, additive.only=FALSE,  
 restrict=TRUE, allow.multiplicative.trend=FALSE)

y

The time series to be forecast.

model

A three-letter code indicating the model to be estimated using the ETS classification and notation. The possible inputs are “N” for none, “A” for additive, “M” for multiplicative, or “Z” for automatic selection. If any of the inputs is left as “Z”, then this component is selected according to the information criterion chosen. The default value of ZZZ ensures that all components are selected using the information criterion.

damped

If damped=TRUE, then a damped trend will be used (either A or M). If damped=FALSE, then a non-damped trend will used. If damped=NULL (the default), then either a damped or a non-damped trend will be selected, depending on which model has the smallest value for the information criterion.

alpha, beta, gamma, phi

The values of the smoothing parameters can be specified using these arguments. If they are set to NULL (the default setting for each of them), the parameters are estimated.

lambda

Box-Cox transformation parameter. It will be ignored if lambda=NULL (the default value). Otherwise, the time series will be transformed before the model is estimated. When lambda is not NULL, additive.only is set to TRUE.

biasadj

If TRUE and lambda is not NULL, then the back-transformed fitted values and forecasts will be bias-adjusted.

additive.only

Only models with additive components will be considered if additive.only=TRUE. Otherwise, all models will be considered.

restrict

If restrict=TRUE (the default), the models that cause numerical difficulties are not considered in model selection.

allow.multiplicative.trend

Multiplicative trend models are also available, but not covered in this book. Set this argument to TRUE to allow these models to be considered.

### Working with ets objects

The ets() function will return an object of class ets. There are many R functions designed to make working with ets objects easy. A few of them are described below.

coef()

returns all fitted parameters.

accuracy()

returns accuracy measures computed on the training data.

summary()

prints some summary information about the fitted model.

autoplot() and plot()

produce time plots of the components.

residuals()

returns residuals from the estimated model.

fitted()

returns one-step forecasts for the training data.

simulate()

will simulate future sample paths from the fitted model.

forecast()

computes point forecasts and prediction intervals, as described in the next section.

### Example: International tourist visitor nights in Australia

We now employ the ETS statistical framework to forecast tourist visitor nights in Australia by international arrivals over the period 2016–2019. We let the ets() function select the model by minimizing the AICc.

aust <- window(austourists, start=2005)  
fit <- ets(aust)  
summary(fit)

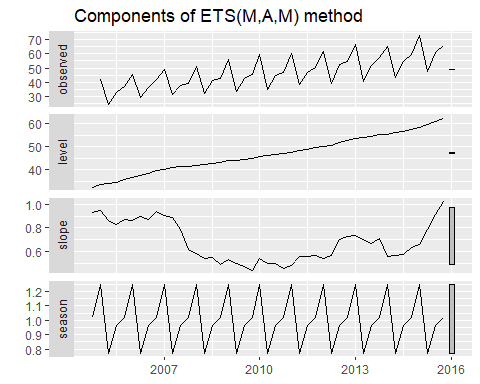
## ETS(M,A,M)   
##   
## Call:  
## ets(y = aust)   
##   
## Smoothing parameters:  
## alpha = 0.1908   
## beta = 0.0392   
## gamma = 2e-04   
##   
## Initial states:  
## l = 32.3679   
## b = 0.9281   
## s = 1.0218 0.9628 0.7683 1.2471  
##   
## sigma: 0.0383  
##   
## AIC AICc BIC   
## 224.8628 230.1569 240.9205   
##   
## Training set error measures:  
## ME RMSE MAE MPE MAPE MASE  
## Training set 0.04836907 1.670893 1.24954 -0.1845609 2.692849 0.409454  
## ACF1  
## Training set 0.2005962

The model selected is ETS(M,A,M):

The parameter estimates are , , and . The output also returns the estimates for the initial states , , , , and Compare these with the values obtained for the Holt-Winters method with multiplicative seasonality presented in Table @ref(tab:tab76). Although this model gives equivalent point forecasts to a multiplicative Holt-Winters’ method, the parameters have been estimated differently. With the ets() function, the default estimation method is maximum likelihood rather than minimum sum of squares.

Figure @ref(fig:MAMstates) shows the states over time, while Figure @ref(fig:MAMforecasts) shows point forecasts and prediction intervals generated from the model. The small values of and mean that the slope and seasonal components change very little over time. The narrow prediction intervals indicate that the series is relatively easy to forecast due to the strong trend and seasonality.

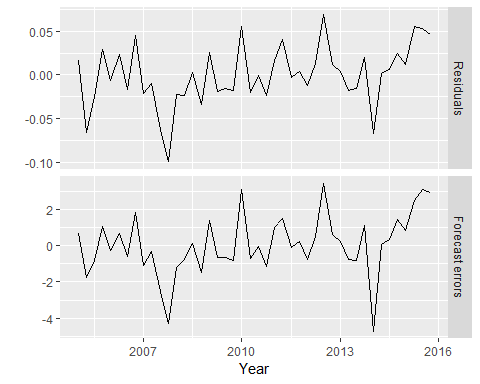
autoplot(fit)



Graphical representation of the estimated states over time.

Because this model has multiplicative errors, the residuals are not equivalent to the one-step forecast errors. The residuals are given by , while the one-step forecast errors are defined as . We can obtain both using the residuals function.

cbind('Residuals' = residuals(fit),  
 'Forecast errors' = residuals(fit, type='response')) %>%  
 autoplot(facet=TRUE) + xlab("Year") + ylab("")



Residuals and one-step forecast errors from the ETS(M,A,M) model.

The type argument is used in the residuals function to distinguish between residuals and forecast errors. The default is type='innovation' which gives regular residuals.

### Bibliography

Hyndman, R. J., Koehler, A. B., Ord, J. K., & Snyder, R. D. (2008). Forecasting with exponential smoothing: The state space approach. Berlin: Springer-Verlag. <http://www.exponentialsmoothing.net>

## Forecasting with ETS models

Point forecasts are obtained from the models by iterating the equations for and setting all for .

For example, for model ETS(M,A,N), Therefore Similarly,

Therefore, and so on. These forecasts are identical to the forecasts from Holt’s linear method, and also to those from model ETS(A,A,N). Thus, the point forecasts obtained from the method and from the two models that underlie the method are identical (assuming that the same parameter values are used).

ETS point forecasts are equal to the medians of the forecast distributions. For models with only additive components, the forecast distributions are normal, so the medians and means are equal. For ETS models with multiplicative errors, or with multiplicative seasonality, the point forecasts will not be equal to the means of the forecast distributions.

To obtain forecasts from an ETS model, we use the forecast function. The R code below shows the possible arguments that this function takes when applied to an ETS model. We explain each of the arguments in what follows.

forecast(object, h=ifelse(object$m>1, 2\*object$m, 10),  
level=c(80,95), fan=FALSE, simulate=FALSE, bootstrap=FALSE, npaths=5000,  
PI=TRUE, lambda=object$lambda, biasadj=NULL, ...)

object

The object returned by the ets() function.

h

The forecast horizon — the number of periods to be forecast.

level

The confidence level for the prediction intervals.

fan

If fan=TRUE, level=seq(50,99,by=1). This is suitable for fan plots.

simulate

If simulate=TRUE, prediction intervals are produced by simulation rather than using algebraic formulae. Simulation will also be used (even if simulate=FALSE) where there are no algebraic formulae available for the particular model.

bootstrap

If bootstrap=TRUE and simulate=TRUE, then the simulated prediction intervals use re-sampled errors rather than normally distributed errors.

npaths

The number of sample paths used in computing simulated prediction intervals.

PI

If PI=TRUE, then prediction intervals are produced; otherwise only point forecasts are calculated.

lambda

The Box-Cox transformation parameter. This is ignored if lambda=NULL. Otherwise, the forecasts are back-transformed via an inverse Box-Cox transformation.

biasadj

If lambda is not NULL, the backtransformed forecasts (and prediction intervals) are bias-adjusted.

fit %>% forecast(h=8) %>%   
 autoplot()

### Prediction intervals

A big advantage of the models is that prediction intervals can also be generated — something that cannot be done using the methods. The prediction intervals will differ between models with additive and multiplicative methods.

For most ETS models, a prediction interval can be written as

where depends on the coverage probability, and is the forecast variance. Values for were given in Table @ref(tab:pcmultipliers). For ETS models, formulas for can be complicated; the details are given in Chapter 6 of @expsmooth08. In the table below, we give the formulas for the additive ETS models, which are the simplest.

(#tab:pitable)Forecast variance expressions for each additive state space model, where is the residual variance, is the seasonal period, and denotes the integer part of .

|  |  |
| --- | --- |
| Model | Forecast variance: |
| (A,N,N) |  |
| (A,A,N) |  |
| (A,A,N) |  |
| (A,N,A) |  |
| (A,A,A) |  |
| (A,A,A) |  |
|  |  |

For a few ETS models, there are no known formulas for prediction intervals. In these cases, the forecast.ets function uses simulated future sample paths and computes prediction intervals from the percentiles of these simulated future paths.

## Bibliography

Hyndman, R. J., Koehler, A. B., Ord, J. K., & Snyder, R. D. (2008). Forecasting with exponential smoothing: The state space approach. Berlin: Springer-Verlag. <http://www.exponentialsmoothing.net>

## Exercises

1. Consider the pigs series — the number of pigs slaughtered in Victoria each month.
2. Use the ses function in R to find the optimal values of and , and generate forecasts for the next four months.
3. Compute a 95% prediction interval for the first forecast using where is the standard deviation of the residuals. Compare your interval with the interval produced by R.
4. Write your own function to implement simple exponential smoothing. The function should take arguments y (the time series), alpha (the smoothing parameter ) and level (the initial level ). It should return the forecast of the next observation in the series. Does it give the same forecast as ses?
5. Modify your function from the previous exercise to return the sum of squared errors rather than the forecast of the next observation. Then use the optim function to find the optimal values of and . Do you get the same values as the ses function?
6. Combine your previous two functions to produce a function which both finds the optimal values of and , and produces a forecast of the next observation in the series.
7. Data set books contains the daily sales of paperback and hardcover books at the same store. The task is to forecast the next four days’ sales for paperback and hardcover books.
8. Plot the series and discuss the main features of the data.
9. Use the ses function to forecast each series, and plot the forecasts.
10. Compute the RMSE values for the training data in each case.
11. Now apply Holt’s linear method to the paperback and hardback series and compute four-day forecasts in each case.
12. Compare the RMSE measures of Holt’s method for the two series to those of simple exponential smoothing in the previous question. (Remember that Holt’s method is using one more parameter than SES.) Discuss the merits of the two forecasting methods for these data sets.
13. Compare the forecasts for the two series using both methods. Which do you think is best?
14. Calculate a 95% prediction interval for the first forecast for each series, using the RMSE values and assuming normal errors. Compare your intervals with those produced using ses and holt.
15. For this exercise use data set eggs, the price of a dozen eggs in the United States from 1900–1993. Experiment with the various options in the holt() function to see how much the forecasts change with damped trend, or with a Box-Cox transformation. Try to develop an intuition of what each argument is doing to the forecasts.

[Hint: use h=100 when calling holt() so you can clearly see the differences between the various options when plotting the forecasts.] Which model gives the best RMSE?

1. Recall your retail time series data (from Exercise 3 in Section @ref(ex-graphics)).
2. Why is multiplicative seasonality necessary for this series?
3. Apply Holt-Winters’ multiplicative method to the data. Experiment with making the trend damped.
4. Compare the RMSE of the one-step forecasts from the two methods. Which do you prefer?
5. Check that the residuals from the best method look like white noise.
6. Now find the test set RMSE, while training the model to the end of 2010. Can you beat the seasonal naïve approach from Exercise 7 in Section @ref(ex-toolbox)?
7. For the same retail data, try an STL decomposition applied to the Box-Cox transformed series, followed by ETS on the seasonally adjusted data. How does that compare with your best previous forecasts on the test set?
8. For this exercise use data set ukcars, the quarterly UK passenger vehicle production data from 1977Q1–2005Q1.
9. Plot the data and describe the main features of the series.
10. Decompose the series using STL and obtain the seasonally adjusted data.
11. Forecast the next two years of the series using an additive damped trend method applied to the seasonally adjusted data. (This can be done in one step using stlf with arguments etsmodel="AAN", damped=TRUE.
12. Forecast the next two years of the series using Holt’s linear method applied to the seasonally adjusted data (as before but with damped=FALSE).
13. Now use ets() to choose a seasonal model for the data.
14. Compare the RMSE of the ETS model with the RMSE of the models you obtained using STL decompositions. Which gives the better in-sample fits?
15. Compare the forecasts from the three approaches? Which seems most reasonable?
16. Check the residuals of your preferred model.
17. For this exercise use data set visitors, the monthly Australian short-term overseas visitors data, May 1985–April 2005.
18. Make a time plot of your data and describe the main features of the series.
19. Split your data into a training set and a test set comprising the last two years of available data. Forecast the test set using Holt-Winters’ multiplicative method.
20. Why is multiplicative seasonality necessary here?
21. Forecast the two-year test set using each of the following methods:
22. an ETS model;
23. an additive ETS model applied to a Box-Cox transformed series;
24. a seasonal naive method;
25. an STL decomposition applied to the Box-Cox transformed data followed by an ETS model applied to the seasonally adjusted (transformed) data.
26. Which method gives the best forecasts? Does it pass the residual tests?
27. Compare the same five methods using time series cross-validation with the tsCV function instead of using a training and test set. Do you come to the same conclusions?
28. The fets function below returns ETS forecasts.

* fets <- function(y, h) {  
   forecast(ets(y), h = h)  
   }

1. Apply tsCV() for a forecast horizon of , for both ETS and seasonal naive methods to the cement data, XXX. (Hint: use the newly created fets and the existing snaive functions as your forecast function arguments.)
2. Compute the MSE of the resulting -steps-ahead errors. (Hint: make sure you remove missing values.) Why is there missing values? Comment on which forecasts are more accurate. Is this what you expected?
3. Compare ets, snaive and stlf on the following six time series. For stlf, you might need to use a Box-Cox transformation. Use a test set of three years to decide what gives the best forecasts.ausbeer, bricksq, dole, a10, h02, usmelec.
4. Use ets() on the following series:bicoal, chicken, dole, usdeaths, lynx, ibmclose, eggs.Does it always give good forecasts?
5. Find an example where it does not work well. Can you figure out why?

## Further reading

Two articles by Ev Gardner [@Gar1985;@Gar2006] provide a great overview of the history of exponential smoothing, and its many variations. A full book treatment of the subject providing the mathematical details is given by @expsmooth08.

# ARIMA models

1. The notation means the largest integer not greater than . [↑](#footnote-ref-182)