STATISTICS: AN INTRODUCTION USING R

By M.J. Crawley

Exercises

8. MULTIPLE REGRESSION & MODEL SIMPLIFICATION

Model Simplification

The *principle of parsimony* (Occam's razor) requires that the model should be as simple as possible. This means that the model should not contain any redundant parameters or factor levels. We achieve this by fitting a maximal model then simplifying it by following one or more of these steps:

- remove non-significant interaction terms;
- remove non-significant quadratic or other non-linear terms;
- remove non-significant explanatory variables;
- group together factor levels that do not differ from one another;
- amalgamate explanatory variables that have similar parameter values;
- set non-significant slopes to zero within ANCOVA

subject, of course, to the caveats that the simplifications make good scientific sense, and do not lead to significant reductions in explanatory power.

Model	**Interpretation**
Saturated model	One parameter for every data point Fit: perfect Degrees of freedom: none Explanatory power of the model: none
Maximal model	Contains all (p) factors, interactions and covariates that might be of any interest. Many of the model's terms are likely to be insignificant Degrees of freedom: $n - p - 1$ Explanatory power of the model: it depends
Minimal adequate model	A simplified model with $0 \le p' \le p$ parameters Fit: less than the maximal model, but not significantly so Degrees of freedom: $n - p' - 1$ Explanatory power of the model: r^2 = SSR/SST
Null model	Just 1 parameter, the overall mean $\bar{y}$ Fit: none; SSE = SST Degrees of freedom: $n - 1$ Explanatory power of the model: none

The steps involved in model simplification

There are no hard and fast rules, but the procedure laid out below works well in practice. With large numbers of explanatory variables, and many interactions and non-linear terms, the process of model simplification can take a very long time. But this is time well spent because it reduces the risk of overlooking an important aspect of the data. It is important to realise that there is no guaranteed way of finding all the important structures in a complex data frame.

Step	Procedure	Explanation
1	Fit the maximal model	Fit all the factors, interactions and covariates of interest. Note the residual deviance. If you are using Poisson or binomial errors, check for overdispersion and rescale if necessary
2	Begin model simplification	Inspect the parameter estimates using **summary**. Remove the least significant terms first, using **update -**, starting with the highest order interactions
3	If the deletion causes an insignificant increase in deviance	Leave that term out of the model Inspect the parameter values again Remove the least significant term remaining
4	If the deletion causes a significant increase in deviance	Put the term back in the model using **update** + . These are the statistically significant terms as assessed by deletion from the maximal model
5	Keep removing terms from the model	Repeat steps 3 or 4 until the model contains nothing but significant terms This is the minimal adequate model If none of the parameters is significant, then the minimal adequate model is the null model

Deletion

Deletion uses the **update** directive to remove terms from the model (you can automate this procedure, using **step**). Perhaps the simplest deletion is to remove the intercept from a regression study using the **~ . -1** directive (you need to be careful with the punctuation: the update formula contains "tilde dot minus" which means "fit the model (the tilde) with the last set of explanatory variables (the dot) but remove (the minus sign) the following"). This fits a new regression line with a single parameter Note that it does *not* rotate the regression line about the point ($\bar{x},\bar{y}$) until it passes through the origin (a common misconception). We can demonstrate this as follows. Take the following data

```
x<-c(0,1,2,3,4,5)
y<-c(2,1,1,5,6,8)
```

and fit a 2-parameter linear regression as model1.

```
model1<-lm(y~x)
model1
```

```
Coefficients:

 (Intercept)    x
   0.3333333 1.4
```

The intercept is 0.33333 and the slope is 1.4. This regression line is defined as passing through the point ($\bar{x},\bar{y}$) but we should check this:

```
mean(x); mean(y)
```

```
[1] 2.5
[1] 3.833333
```

We check that the model goes through the point (2.5,3.8333) using predict like this

```
predict(model1,list(x=2.5))
```

```
    1
 3.833333
```

So that's all right then. Now we remove the intercept (~ . -1) using update to fit a single parameter model that is forced to pass through the origin:

```
model2<-update(model1,~. -1)

model2
```

```
Coefficients:

        x
 1.490909
```

Note how the model formula has been altered by update. The slope of model2 is steeper (1.49 compared with 1.40), but does the line still pass through the point ($\bar{x},\bar{y}$)? We can test this using predict:

```
predict(model2,list(x=2.5))
```

```
        1
 3.727273
```

No, it doesn't. The single slope parameter is estimated from

$$y_i = \beta x_i + \varepsilon_i$$

in which the least squares estimate of β is b where

$$b = \frac{\sum xy}{\sum x^2}$$

instead of SSXY / SSX when 2 parameters are estimated from the data (see Practical 4). This graph does not pass through the point ($\bar{x}, \bar{y}$).

Because `update(model1,~. –1)` causes the regression line to be rotated away from its maximum likelihood position, this will inevitably cause an increase in the residual deviance. If the increase in deviance is significant, as judged by a likelihood ratio test, then the simplification is unwarranted, and the intercept should be added back to the model. Forcing the regression to pass through the origin may also cause problems with non-constancy of variance. Removing the intercept is generally not recommended unless we have confidence in the linearity of the relationship over the whole range of the explanatory variable(s) (i.e. all the way from the origin up the maximum value of x).

The practice of model simplification

In general we shall tend to begin the process of model simplification by removing high order interaction terms from a maximal model. Thoughts about forcing the line though the origin are most unlikely to arise. If removal of the high order interaction terms shows them to be non significant, we move on to test the low order interactions and then main effects. The justification for deletion is made with the current model *at the level in question*. Thus in considering whether or not to retain a given second order interaction, it is deleted with all other second order interactions (plus any significant higher-order terms that do not contain the variables of interest) included in the model. If deletion leads to a significant increase in deviance, the term must retained, and the interaction added back into the model. In considering a given first order interaction, all other first order interactions plus any significant higher-order interactions are included at each deletion. And so on.

Main effects which figure in significant interactions should not be deleted. If you do try to delete them there will be no change in deviance and no change in degrees of freedom, because the factor is aliased; the deviance and degrees of freedom will simply be transferred to a surviving interaction term. It is a moot point whether block effects should be removed during model simplification. In the unlikely event that block effects were *not* significant (bearing in mind that in science, everything varies, and so insignificant block effects are the exception rather than the rule), then you should compare your conclusions with and without the block terms in the model. If the conclusions differ, then you need to think very carefully about why, precisely, this has happened. The balance of opinion is that block effects are best left unaltered in the minimal adequate model.

Collapsing factor levels

It often happens that a categorical explanatory variable is significant, but not all of the factor levels are significantly different from one another. We may have a set of *a priori* contrasts that guide model selection. Often, however, we simply want to get rid of factor levels that are redundant. A frequent outcome during anova that, say, the 'low' and 'medium' levels of a treatment are not significantly different from one another, but both differ from the 'high' level treatment. Collapsing factor levels involves calculating a new factor that has the same value for 'low' and 'medium' levels, and another level for ‘high’. This new 2 level factor is then added to the model at the same time as the original 3 level factor is removed. The increase in deviance is noted. If the change is not significant, then the simplification is justified and the new 2-level factor is retained. If a significant increase in deviance occurs, then the original 3 level factor must be restored to the model.

```
yield.data<-read.table("c:\\temp\\levels.txt",header=T)
attach(yield.data)
names(yield.data)
```

```
[1] "yield" "level"
```

```
model<-aov(yield~level)
summary(model)
```

```
             Df  Sum Sq Mean Sq F value    Pr(>F)
level         2 28.1333 14.0667  13.188 0.000935 ***
Residuals    12 12.8000  1.0667
```

A highly significant effect of level is clear. We might leave it at this, but in the interests of parsimony, it would be sensible to ask whether we really need to retain all 3 levels of the factor. We should tabulate the treatment means:

```
tapply(yield,level,mean)
```

```
   A   B    C
 7.2 7.4 10.2
```

The means of levels A and B are very close to one another. Perhaps we can get almost the same explanatory power with a 2-level factor (level2) that has level 1 for A and for B and level 2 for C ?

```
level2<-factor(1+(level=="C"))
level2
```

```
 [1] 1 1 1 1 1 1 1 1 1 1 2 2 2 2 2
```

Note the use of logical arithmetic (see Practical 2) to get the second level of *level2*; level==”C” evaluates to 1 when true and to zero when false. Now we can remove *level* from *model* and replace it with *level2* using update like this:

```
model2<-update(model , ~ . - level + level2 )
```

To see if the model simplification was justified, we compare the simpler *model2* with the original *model* using **anova**:

```
anova(model,model2)
```

```
Analysis of Variance Table

Model 1: yield ~ level
Model 2: yield ~ level2

  Res.Df  RSS Df Sum of Sq      F Pr(>F)
1     12 12.8
2     13 12.9 -1      -0.1 0.0937 0.7647
```

The simplification is vindicated by the very high *p* value. Had *p* been less than 0.05 we would return the 3-level factor to the model, but there is no justification for keeping the 3-level factor in this case.

In multiple regression, two explanatory variables may have similar parameter values, and it may make sense to create a single variable that is the sum of the original two variables. For example, if yield increases by 2 kg for every unit of Maxigrow and 2.5 kg for every unit of Yieldmore we might try calculating a single variable (Fertilizer = Maxigrow + Yieldmore) and fitting this to the model instead. If there is no significant increase in residual deviance then we have saved another degree of freedom, and Occam would be proud of us.

Offsets in model simplification

Sometimes there may be a clear theoretical prediction about one or more of the parameter values. In a study of density dependent mortality in plant populations, for example, we might wish to test whether the self-thinning rule applied to a particular set of data relating mean plant weight (y) to mean plant density (x). Yoda's rule states that the relationship is allometric with $y = ax^{-3/2}$. We might wish to test whether our estimated exponent is significantly different from this. One way would be to do a t-test, comparing our estimate to –1.5 using the standard error from the summary table. An alternative is so specify the exponent as -3/2 in an offset and compare the fit of this model with our initial model in which the maximum likelihood estimate of the exponent was used.

```
Yoda<-read.table("c:\\temp\\Yoda.txt",header=T)
attach(Yoda)
names(Yoda)
[1] "density"  "meansize"
```

The theory is a power law relationship, so the appropriate linear model is log(y) against log(x). We fit the model as a **glm** rather than an **lm** simply because **glm** allows us to use offsets and **lm** does not.

```
model<-glm(log(meansize)~log(density))
```

```
summary(model)
```

```
Coefficients:
                    Value Std. Error   t value
 (Intercept)  12.213579 0.26622433  45.87702
log(density)  -1.529393 0.06442428 -23.73938
```

The slope is close to the predicted value of-3/2 but is it close enough ? A t-test suggests that it is definitely not significantly different from –1.5

```
(1.529393-1.5)/0.06442482
```

```
[1] 0.4562372
```

To find the probability of t = 0.456 with 19 d.f. we use the cumulative probability of Student’s t-distribution **pt** like this:

```
1-pt(0.4562372,19)
```

```
[1] 0.3266958
```

Here is the same test, but using offsets to specify the slope as exactly –3/2.

```
values<-  - 1.5*log(density)
```

The trick is that we specify the **offset** as part of the model formula like this (much as we specified the error term in a nested design). Now, the only parameter estimated by maximum likelihood is the intercept (coefficient ~1):

```
model2<-glm(log(meansize)~1+offset(values))
```

You get exactly the same model if you leave out the 1+ term; it is included just to emphasise that you are estimating only one parameter. Now we compare the two models using **anova**

```
anova(model,model2,test="F")
```

```
Analysis of Deviance Table
Response: log(meansize)

          Terms Resid. Df Resid. Dev Test Df    Deviance   F Value     Pr(F)
1      log(density)   19   8.781306
2 1 + offset(values)   20   8.877508      -1 -0.09620223 0.2081515 0.6533922
```

Our conclusion is the same in both cases, even though the p values differ somewhat (0.327 vs. 0.653). These data conform very precisely with the allometry predicted by Yoda’s rule.

There may be some virtue in simplifying parameter values in order to make the numbers easier to communicate. This should never be done, of course, if changing the parameter values causes a significant reduction in the explanatory power of the model. It is much more straightforward, for example, to say that yield increases by 2 kg per hectare for every extra unit of fertilizer, than to say that it increases by 1.947 kg. Similarly, it may be preferable to say that the odds of infection increase 10-fold under a given treatment, than to say that the logits increase by 2.321; without model simplification this is equivalent to saying that there is a 10.186-fold increase in the odds.

Caveats

Model simplification is an important process but it should not be taken to extremes. For example, the interpretation of deviances and standard errors produced with fixed parameters that have been estimated from the data, should be undertaken with caution. Again, the search for 'nice numbers' should not be pursued uncritically. Sometimes there are good scientific reasons for using a particular number (e.g. a power of 0.66 in an allometric relationship between respiration and body mass), but it would be absurd to fix on an estimate of 6 rather than 6.1 just because 6 is a whole number.

Summary

Remember that *order matters*. If your explanatory variables are correlated with each other, then the significance you attach to a given explanatory variable will depend upon whether you delete it from a maximal model or add it to the null model. Always test by model simplification and you won't fall into this trap.

The fact that we have laboured long and hard to include a particular experimental treatment does not justify the retention of that factor in the model if the analysis shows it to have no explanatory power. Anova tables are often published containing a mixture of significant and non-significant effects. This is not a problem in orthogonal designs, because sums of squares can be unequivocally attributed to each factor and interaction term. But as soon as there are missing values or unequal weights, then it is impossible to tell how the parameter estimates and standard errors of the significant terms would have been altered if the non-significant terms had been deleted. The best practice is this

- say whether your data are orthogonal or not
- present a minimal adequate model
- give a list of the non-significant terms that were omitted, and the deviance changes that resulted from their deletion

The reader can then judge for themselves the relative magnitude of the non-significant factors, and the importance of correlations between the explanatory variables.

The temptation to retain terms in the model that are 'close to significance' should be resisted. The best way to proceed is this. If a result would have been *important* if it had been statistically significant, then it is worth repeating the experiment with higher replication and/or more efficient blocking, in order to demonstrate the importance of the factor in a convincing and statistically acceptable way.

Multiple Regression

When there is more than one continuous explanatory variable there are lots of choices that need to be made. At the data inspection stage, there are many more kinds of **plots** we could do:

1) plot the response against each of the explanatory variables separately

2) plot the explanatory variables against one another (e.g. **pairs**)

3) plot the response against pairs of explanatory variables in 3-D plots

4) plot the response against explanatory variables for different combinations of other explanatory variables (e.g. conditioning plots, **coplot**; see p. 16).

At the modelling stage, we need to choose between multiple regression, non parametric surface-fitting (local regression or loess), additive models (with perhaps a mix of parametric and non parametric terms), tree models or multivariate techniques.

At the model checking stage, we need to be particularly concerned with the extent of correlations between the explanatory variables

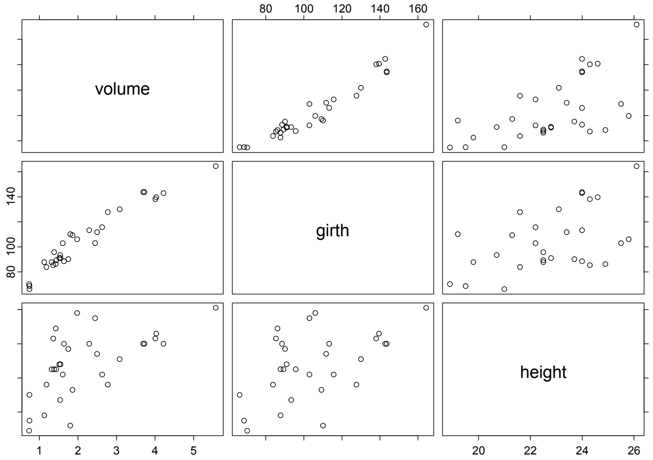
We shall begin with an extremely simple example with just two explanatory variables. The response variable is the volume of utilisable timber in harvested trunks of different lengths and diameters.

```
timber<-read.table("c:\\temp\\timber.txt",header=T)
attach(timber)
names(timber)
```

```
[1] "volume" "girth"  "height"
```

We begin by comparing different kinds of plots. It is a good idea to start by plotting the response against each of the explanatory variables in turn, and by looking at the extent to which the explanatory variables are correlated with each other. The multi-panel **pairs** function is excellent for this (see Practical 1).

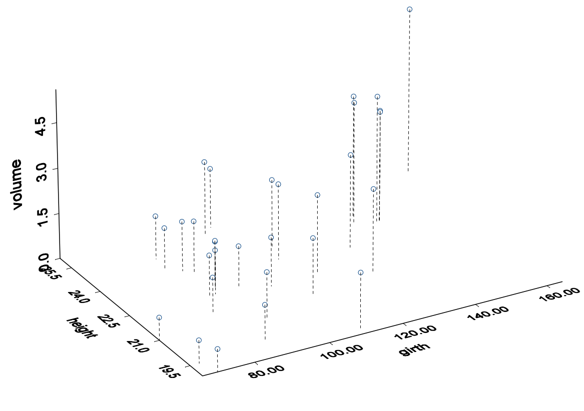
```
pairs(timber)
```



It is clear that girth is a much better predictor of the volume of usable timber than is height (top row; middle and right hand panels). This is what we might have expected, because it is very easy to imagine two tree trunks of exactly the same length that differ enormously in the volume of timber they continue. The long thin one might contain no useful timber at all, but you could build a whole house out of the stout one. The bottom-right quartet of panels shows that there is a positive correlation between the two explanatory variables, but trunks of the same girth appear to differ more in height than trunks of the same height differ in girth.

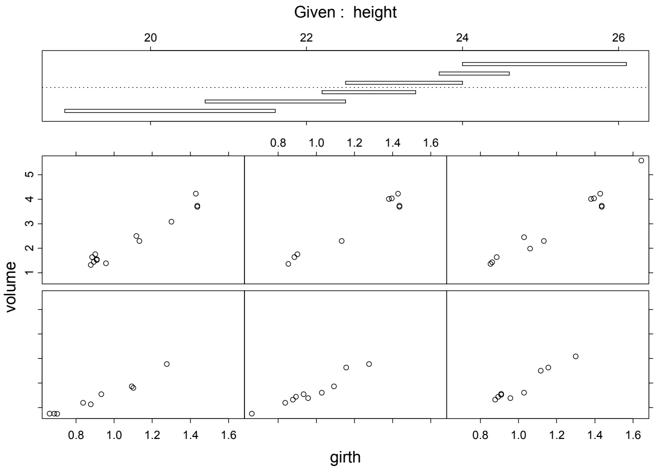
Although experts always advise against it, you might well be tempted to look at a **3D scatterplot** of the data. The reason experts advise against it is a good one: they argue that it is hard to interpret the data correctly from a perspective plot, and the whole idea of plotting is to aid interpretation. The problem with 3D scatterplots is that it is impossible to know where a point is located; you don't know whether it is in the foreground or the background. Plots showing 3D *surfaces* are another matter altogether; they can be very useful in interpreting the models that emerge from multiple regression analysis. For what it is worth, here is the 3D scatterplot of the timber data: gui stands for Graphics User Interface (**R doesn't do this**):

```
guiPlot("Drop Lines Plot",data,frame(girth,height,volume))
```



Perhaps the most useful kinds of plots for this purpose are **conditioning plots**. These allow you to see whether the response depends upon an explanatory variable in different ways at different levels of other explanatory variables. It takes 2 dimensional slices through a potentially multi-dimensional volume of parameter space, and these are often much more informative than the unconditioned scatterplots produced by **pairs**. Here is volume against girth, conditioning on height. Note the use of a model formula in the plotting directive:

```
coplot(volume~girth|height)
```



Much of the scatter that was such a feature of the **pairs** plot has gone. Within a height class, the relationship between volume and girth is relatively close. What seems to be happening is that the *slope* of the relationship depends upon the height, getting steeper for taller trees. This is a very informative plot. It tells us that both girth and height are likely to be required in the minimal model and that there may be an interaction between the two explanatory variables.

The multiple regression model

The assumptions are the same as with simple linear regression. The explanatory variables are assumed to be measured without error, the errors are normally distributed, the errors are confined to the response variable, and the variance is constant. The model for a multiple regression with 2 explanatory variables (x_1 and x_2) looks like this:

$$y_i = \beta_0 + \beta_1 x_{1,i} + \beta_2 x_{2,i} + \varepsilon_i$$

The *i*th data point, y_i, is determined by the levels of the 2 continuous explanatory variables $x_{1,i}$ and $x_{2,i}$ by the model's 3 parameters (the intercept β_0 and the two slopes β_1 and β_2), and by the residual ε_i of point *i* from the fitted surface. More generally, the model is presented like this:

$$y_i = \sum \beta_i x_i + \varepsilon_i$$

where the summation term is called the **linear predictor**, and can involve many explanatory variables, non linear terms and interactions. The question that immediately arises is this: how are the values of the parameters determined? In order to see this, we return to the simple linear regression case that we investigated earlier, and re-set that example in terms of matrix algebra. Once the matrix notation is reasonably clear, we can then extend it to deal with an arbitrarily large number of explanatory variables.

N.B. If you are not familiar with matrix algebra, then go directly to p. 230 and skip the next section.

Matrix representation of regression: working through a linear regression in matrix form

The best way to learn how matrix algebra can generalise our ideas about regression is to work through a simple example where the data are already familiar. The example involved weight gain of caterpillars fed diets with differing concentrations of tannin (see Practical 4). The famous five and sample size were:

$\sum x, \sum x^2, \sum y, \sum y^2, \sum xy, n$

```
36   204 62 536   175 9
```

The 3 steps involved are these:

1) Display the model in matrix form

$$\mathbf{Y = Xb + e}$$

2) Determine the least squares estimate of **b**

$$\mathbf{b=(X'X)^{-1}X'Y}$$

3) Carry out the analysis of variance

b’X’Y’

We look at each of these in turn. The response variable **Y, 1** and the errors **e** are simple n x 1 column vectors, **X** is a n x 2 matrix and β is a 2 x 1 vector of coefficients.

$$\mathbf{Y} = \begin{bmatrix} 12 \\ 10 \\ 8 \\ 11 \\ 6 \\ 7 \\ 2 \\ 3 \\ 3 \end{bmatrix} \quad \mathbf{X} = \begin{bmatrix} 1 & 0 \\ 1 & 1 \\ 1 & 2 \\ 1 & 3 \\ 1 & 4 \\ 1 & 5 \\ 1 & 6 \\ 1 & 7 \\ 1 & 8 \end{bmatrix} \quad \mathbf{e} = \begin{bmatrix} e_1 \\ e_2 \\ e_3 \\ e_4 \\ e_5 \\ e_6 \\ e_7 \\ e_8 \\ e_9 \end{bmatrix} \quad \mathbf{1} = \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \end{bmatrix} \quad \beta = \begin{bmatrix} \beta_0 \\ \beta_1 \end{bmatrix}$$

The y vector and the 1-vector are entered into the computer like this:

```
y<-c(12,10,8,11,6,7,2,3,3)
```

```
one<-rep(1,9)
```

The sample size is **1’1** (transpose of *one* times *one*):

```
t(one) %*% one
```

```
     [,1]
[1,]    9
```

The hard part concerns the matrix of the explanatory variables. This has one more column than there are explanatory variables. The extra column is a column of 1’s on the left hand side of the matrix. Because the x values are evenly spaced we can generate them rather than type them in (use lower case x)

```
x<-0:8
x
```

```
[1] 0 1 2 3 4 5 6 7 8
```

We manufacture the matrix **X** by using **cbind** to tack a column of 1’s in front of the vector of x’s. Note that the single number 1 is *coerced* into length n to match x (use upper case X):

```
X<-cbind(1,x)
```

```
X
        x
[1,] 1 0
[2,] 1 1
[3,] 1 2
[4,] 1 3
[5,] 1 4
[6,] 1 5
[7,] 1 6
[8,] 1 7
[9,] 1 8
```

Thus, all of the matrices are of length $n = 9$, except for β which is length k+1 (where k = number of explanatory variables; 1 in this example).

The transpose of a matrix (denoted by a prime superscript) is the matrix obtained by writing the rows as columns in the order in which they occur, so that the columns all become rows. So a 9x2 matrix **X** transposes (**t**) to a 2x9 matrix **X'** (called *Xp* for 'x prime') like this:

```
Xp <- t(X)
```

```
Xp
  [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9]
     1    1    1    1    1    1    1    1    1
x    0    1    2    3    4    5    6    7    8
```

This is useful for regression, because

$$\sum y^2 = y_1^2 + y_2^2 + ... + y_n^2 = \mathbf{Y'Y}$$

```
t(y) %*% y
```

```
     [,1]
[1,]  536
```

$$\sum y = n\bar{y} = y_1 + y_2 + ... + y_n = \mathbf{1'Y}$$

```
t(one) %*% y
```

```
   [,1]
[1,]   62
```

$$(\sum y)^2 = \mathbf{Y'11'Y}$$

```
t(y) %*% one %*% t(one) %*% y
```

```
     [,1]
[1,] 3844
```

For the matrix of explanatory variables, we see that **X'X** gives a 2x2 matrix containing *n*, $\sum x$ and $\sum x^2$. The numerical values are easy to find using matrix multiplication %*%

```
Xp %*% X
```

```
       x
   9  36
x 36  204
```

Note that **X'X** (a 2x2 matrix) is completely different from **XX'** (a 9x9 matrix). The matrix **X'Y** gives a 2x1 matrix containing $\sum y$, and the sum of products $\sum xy$

```
Xp %*% y
```

```
  [,1]
    62
x  175
```

Using the beautiful symmetry of the normal equations

$$b_0 n + b_1 \sum x = \sum y$$

$$b_0 \sum x + b_1 \sum x^2 = \sum xy$$

we can write the regression directly in matrix form as

$$\mathbf{X'Xb = X'Y}$$

because we already have the necessary matrices to form the left and right hand sides. To find the least squares parameter values **b** we need to divide both sides by **X'X.** This involves calculating the **inverse** of the **X'X** matrix. The inverse exists only when the matrix is square and when its determinant is non-singular. The inverse contains $-\bar{x}$ and $\sum x^2$ as its terms, with SSX as the denominator.

$$\mathbf{(X'X)^{-1}} = \begin{bmatrix} \dfrac{\sum x^2}{n\sum (x-\bar{x})^2} & \dfrac{-\bar{x}}{\sum (x-\bar{x})^2} \\ \dfrac{-\bar{x}}{\sum (x-\bar{x})^2} & \dfrac{1}{\sum (x-\bar{x})^2} \end{bmatrix}$$

When every element of a matrix has a common factor, it can be taken outside the matrix. Here, the term 1/(n.SSX) can be taken outside to give:

$$\mathbf{(X'X)^{-1}} = \frac{1}{n\sum (x-\bar{x})^2} \begin{bmatrix} \sum x^2 & -\sum x \\ -\sum x & n \end{bmatrix}$$

Computing the numerical value of this is easy using the matrix function **ginv** (this stands for generalized inverse):

```
XM<-Xp %*% X

library(MASS)

ginv(XM)
```

```
              [,1]        [,2]
[1,]  0.37777778 -0.06666667
[2,] -0.06666667  0.01666667
```

Now we can solve the normal equations

$$\mathbf{(X'X)^{-1}(X'X)b = (X'X)^{-1}X'Y}$$

using the fact that $\mathbf{(X'X)^{-1}(X'X)=I}$ to obtain the important general result:

$$\mathbf{b = (X'X)^{-1}X'Y}$$

In our example, we have $\mathbf{(X'X)^{-1}}$ as

```
              [,1]        [,2]
[1,]  0.37777778 -0.06666667
[2,] -0.06666667  0.01666667
```

and **X'Y** as

```
[,1]
    62
x   175
```

so **b** is found by the 3-matrix product

```
b<-ginv(XM) %*% Xp %*% y

b
             [,1]
[1,]  11.755556
[2,]  -1.216667
```

which you will recognise as the intercept and slope respectively.

The ANOVA computations are as follows. The correction factor, CF, = **Y'11'Y**/n

```
CF<-t(y) %*% one %*% t(one) %*% y / 9
CF
```

```
          [,1]
[1,] 427.1111
```

SST is **Y’Y** – CF

```
t(y) %*% y - CF
```

```
          [,1]
[1,] 108.8889
```

SSR is **b’X’Y** – CF

```
t(b) %*% t(X) %*% y - CF
```

```
          [,1]
[1,] 88.81667
```

and SSE is **Y’Y – b’X’Y**

```
t(y) %*% y  -  t(b) %*% t(X) %*% y
```

```
          [,1]
[1,] 20.07222
```

```
rm(b,y,CF)
```

Working through a multiple regression by hand

To show how multiple regression works, we shall go through a simple example long-hand. We have two explanatory variables (girth and height) and the response variable is the volume of usable timber logs of this girth and height. Here is the data set (it is loaded on p. 219):

```
timber
```

```
   volume  girth height
 1 0.7458  66.23   21.0
 2 0.7458  68.62   19.5
 3 0.7386  70.22   18.9
 4 1.1875  83.79   21.6
 5 1.3613  85.38   24.3
 6 1.4265  86.18   24.9
 7 1.1296  87.78   19.8
 8 1.3179  87.78   22.5
 9 1.6365  88.57   24.0
10 1.4410  89.37   22.5
11 1.7524  90.17   23.7
12 1.5206  90.97   22.8
13 1.5496  90.97   22.8
14 1.5424  93.36   20.7
15 1.3831  95.76   22.5
16 1.6075 102.94   22.2
17 2.4475 102.94   25.5
18 1.9841 106.13   25.8
19 1.8610 109.32   21.3
20 1.8030 110.12   19.2
21 2.4982 111.71   23.4
22 2.2954 113.31   24.0
23 2.6285 115.70   22.2
24 2.7734 127.67   21.6
25 3.0847 130.07   23.1
26 4.0116 138.05   24.3
```

```
27 4.0333 139.64   24.6
28 4.2216 142.84   24.0
29 3.7292 143.63   24.0
30 3.6930 143.63   24.0
31 5.5757 164.38   26.1
```

Note that in the data frame, girth is measured in cm and height in m. To keep things simple, we convert girth to m as well (volume is in m^3 already):

```
girth<-girth/100
```

The response variable (volume) is re-named *y* like this:

```
y<-volume
```

The vector of explanatory variables **X** is made by **cbind** like this

```
X<-cbind(1,girth,height)
```

and its transpose is Xp (X prime) **X'**

```
Xp<-t(X)
```

We obtain the sums of X and the sums of squares of X from **X'X**

```
Xp %*% X
```

```
                     girth     height
        31.0000  32.77230   706.8000
 girth  32.7723  36.52706   754.6654
height 706.8000 754.66542 16224.6600
```

This reads as follows. Top left is n = 31, the number of trees. The second row of the first column shows the sum of the girths = 32.7723, and the third row is the sum of the heights = 706.8. The second row of column 2 shows the sum of the squares of girth = 36.52706 and the third row the sum of the products girth*height = 754.66542 (the symmetry has this figure in the second row of the 3rd column as well). The last figure in the bottom right is the sum of the squares of the heights = 16224.66.

We get the sums of products from **X'Y**

```
Xp %*% y
              [,1]
          67.72630
 girth    80.24607
height  1584.99573
```

The top number is the sum of the timber volumes $\sum y$ = 67.7263. The second number is the sum of the products volume * girth = 80.24607 and the last number is the sum of the products volume * height = 1584.99573.

Now we need the inverse of **X’X**

```
ginv(Xp%*%X)
```

```
          [,1]        [,2]        [,3]
[1,]  4.9519523  0.35943351 -0.23244197
[2,]  0.3594335  0.72786995 -0.04951388
[3,] -0.2324420 -0.04951388  0.01249064
```

which shows all of the corrected sums of squares of the explanatory variables. For instance, the top left hand number 4.9519523 is

$$\frac{\sum g^2 \sum h^2 - (\sum g.h)^2}{c-d}$$

where *g* stands for girth and *h* for height, *c* is $n\sum g^2 \sum h^2 + 2\sum g \sum h \sum gh$ and *d* is $n(\sum gh)^2 + (\sum g)^2 \sum h^2 + (\sum h)^2 \sum g^2$ (see Draper & Smith (1981) for details of how to calculate determinants for matrices larger than 2x2).

Finally we compute the parameter values, **b**:

```
b<-ginv(Xp %*% X) %*% Xp %*% y
b
```

```
            [,1]
[1,] -4.19899732
[2,]  4.27251096
[3,]  0.08188343
```

and compare the vector **b** with the parameter estimates obtained by multiple regression

```
lm(volume~girth+height)
```

```
Coefficients:
 (Intercept)    girth     height
   -4.198997 4.272511 0.08188343
```

and are hugely relieved to find that they are the same. The first element of **b** is the intercept (-4.199), the second is the slope of the graph of volume against girth (4.27) and the third is the slope of the graph of volume against height (0.082).

To finish, we compute the sums of squares for the anova table, starting with the correction factor

```
CF<-t(y) %*% one %*% t(one) %*% y / length(y)
CF
        [,1]
[1,] 147.963
```

```
sst<-t(y) %*% y - CF
sst
         [,1]
[1,] 42.50408
```

```
ssr<-t(b) %*% Xp %*% y - CF
ssr
         [,1]
[1,] 40.29156
```

This is the sum of the girth sum of squares (39.75477) and the height (0.53679). Finally sse can be computed by difference:

```
sse<-sst-ssr
sse
         [,1]
[1,] 2.212518
```

These check out with sums of squares in the anova produced by the **aov** fit:

```
model<-aov(volume~girth+height)
summary(model)

          Df  Sum of Sq  Mean Sq  F Value       Pr(F)
    girth  1   39.75477 39.75477 503.1070 0.00000000
   height  1    0.53679  0.53679   6.7933 0.01449791
Residuals 28    2.21252  0.07902
```

Note that although the original scatter was greater on the graph of volume against girth, it is girth that explains more of the variation in volume in a model containing both explanatory variables. A more thorough analysis of these data, employing a more realistic scale of measurement, is described later.

Multiple Regression

In multiple regression we have a continuous response variable and two or more continuous explanatory variables (i.e. no categorical explanatory variables). There are several important issues involved in carrying out a multiple regression:

- which explanatory variables to include
- curvature in the response to the explanatory variables
- interactions between explanatory variables
- correlation between explanatory variables
- the risk of over-parameterization

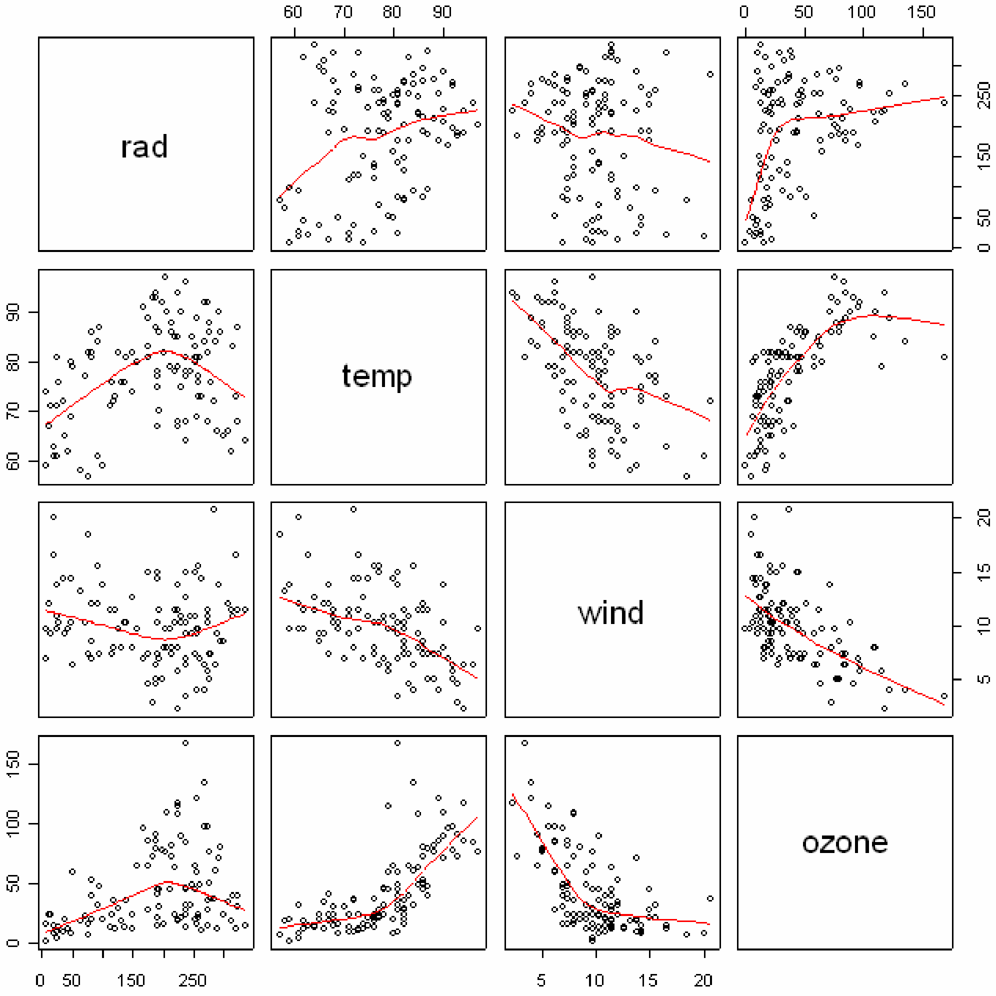
Let’s begin with an example from air pollution studies. How is ozone concentration related to wind speed, air temperature and the intensity of solar radiation?

```
ozone.pollution<-read.table("c:\\temp\\ozone.data.txt",header=T)
attach(ozone.pollution)
names(ozone.pollution)
```

```
[1] "rad"    "temp"   "wind"   "ozone"
```

In multiple regression, it is always a good idea to use pairs to look at all the correlations:

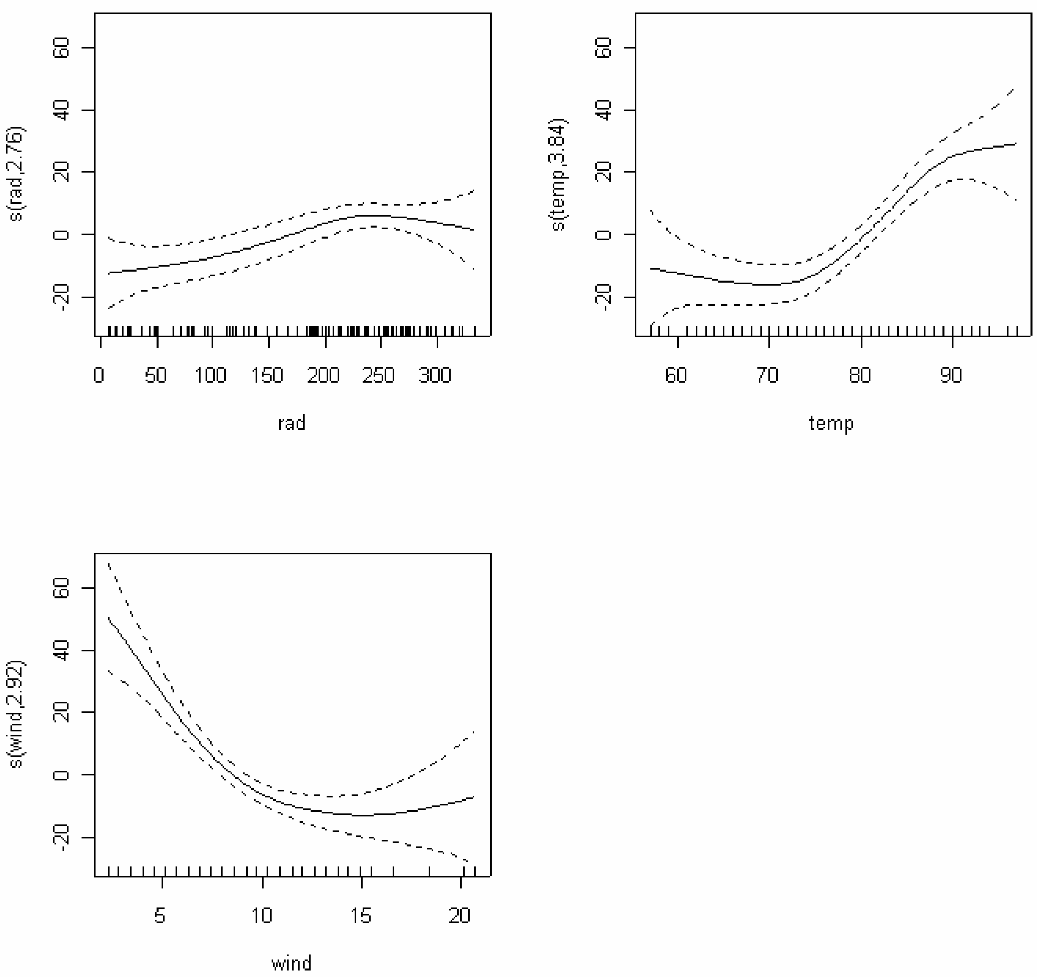
```
pairs(ozone.pollution,panel=panel.smooth)
```



The response variable, ozone concentration, is shown on the *y* axis of the bottom row of panels: there is a strong negative relationship with wind speed, a positive correlation with temperature and a rather unclear, humped relationship with radiation.

A good way to start a multiple regression problem is using non-parametric smoothers in a generalized additive model (gam) like this:

```
library(mgcv)
par(mfrow=c(2,2))
model<-gam(ozone~s(rad)+s(temp)+s(wind))
plot(model)
par(mfrow=c(1,1))
```

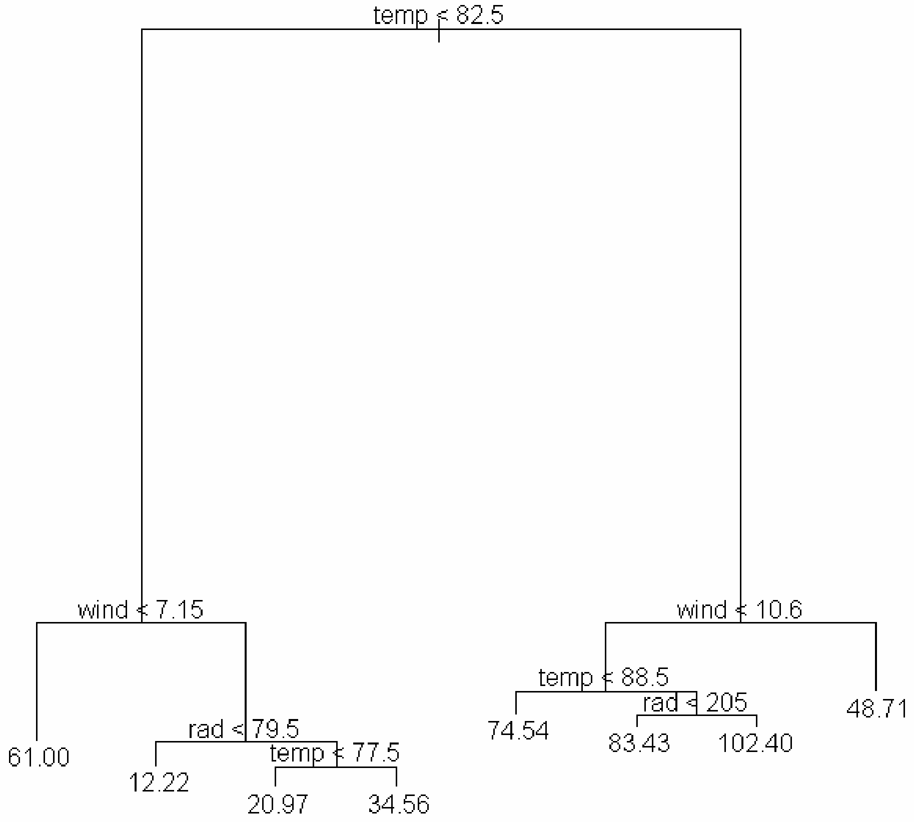


The confidence intervals are sufficiently narrow to suggest that the curvature in the relationship between ozone and temperature is real, but the curvature of the relationship with wind is questionable, and a linear model may well be all that is required for solar radiation.

The next step might be to fit a tree model to see whether complex interactions between the explanatory variables are indicated:

```
library(tree)
```

```
model<-tree(ozone~.,data=ozone.pollution)
plot(model)
text(model)
```



This shows that temperature is far and away the most important factor affecting ozone concentration (the longer the branches in the tree, the greater the deviance explained). Wind speed is important at both high and low temperatures, with still air being associated with higher mean ozone levels (the figures at the ends of the branches). Radiation shows an interesting, but subtle effect. At low temperatures, radiation matters at relatively high wind speeds (> 7.15), whereas at high temperatures, radiation matters at relatively low wind speeds (< 10.6); in both cases, however, higher radiation is associated with higher mean ozone concentration. The tree model therefore indicates that the interaction structure of the data is not complex (a reassuring finding).

Armed with this background information (likely curvature of the temperature response and an uncomplicated interaction structure) we can begin the linear modelling. We start with the most complicated model: this includes interactions between all 3 explanatory variables plus quadratic terms to test for curvature in response to each of the 3 explanatory variables. If you want to do calculations inside the model formula (e.g. produce a vector of squares for fitting quadratic terms), then you need to use the “as is” function, which is a capital I () like this (i.e. not a 1 or a lower case L):

```
model1<-lm(ozone~temp*wind*rad+I(rad^2)+I(temp^2)+I(wind^2))
summary(model1)
```

```
Coefficients:
                Estimate Std. Error t value Pr(>|t|)
(Intercept)    5.683e+02  2.073e+02   2.741  0.00725 **
temp          -1.076e+01  4.303e+00  -2.501  0.01401 *
wind          -3.237e+01  1.173e+01  -2.760  0.00687 **
rad           -3.117e-01  5.585e-01  -0.558  0.57799
I(rad^2)      -3.619e-04  2.573e-04  -1.407  0.16265
I(temp^2)      5.833e-02  2.396e-02   2.435  0.01668 *
I(wind^2)      6.106e-01  1.469e-01   4.157 6.81e-05 ***
temp:wind      2.377e-01  1.367e-01   1.739  0.08519 .
temp:rad       8.402e-03  7.512e-03   1.119  0.26602
wind:rad       2.054e-02  4.892e-02   0.420  0.67552
temp:wind:rad -4.324e-04  6.595e-04  -0.656  0.51358

Residual standard error: 17.82 on 100 degrees of freedom
Multiple R-Squared: 0.7394,     Adjusted R-squared: 0.7133
F-statistic: 28.37 on 10 and 100 DF,  p-value:     0
```

The 3-way interaction is clearly not significant, so we remove it to begin the process of model simplification:

```
model2<-update(model1,~. – temp:wind:rad)
summary(model2)
```

```
Coefficients:
              Estimate Std. Error t value Pr(>|t|)
(Intercept)  5.245e+02  1.957e+02   2.680   0.0086 **
temp        -1.021e+01  4.209e+00  -2.427   0.0170 *
wind        -2.802e+01  9.645e+00  -2.906   0.0045 **
rad          2.628e-02  2.142e-01   0.123   0.9026
I(rad^2)    -3.388e-04  2.541e-04  -1.333   0.1855
I(temp^2)    5.953e-02  2.382e-02   2.499   0.0141 *
I(wind^2)    6.173e-01  1.461e-01   4.225 5.25e-05 ***
temp:wind    1.734e-01  9.497e-02   1.825   0.0709 .
temp:rad     3.750e-03  2.459e-03   1.525   0.1303
wind:rad    -1.127e-02  6.277e-03  -1.795   0.0756 .
```

Start by removing the least significant interaction term. From model1 this looks like `wind:rad` (p = 0.67552), but the sequence has been altered by removing the 3-way term (it is now `temp:rad` with p = 0.1303):

```
model3<-update(model2,~. - temp:rad)
summary(model3)
```

```
Coefficients:
              Estimate Std. Error t value Pr(>|t|)
(Intercept)  5.488e+02  1.963e+02   2.796  0.00619 **
temp        -1.144e+01  4.158e+00  -2.752  0.00702 **
wind        -2.876e+01  9.695e+00  -2.967  0.00375 **
rad          3.061e-01  1.113e-01   2.751  0.00704 **
```

```
I(rad^2)    -2.690e-04  2.516e-04  -1.069  0.28755
I(temp^2)    7.145e-02  2.265e-02   3.154  0.00211 **
I(wind^2)    6.363e-01  1.465e-01   4.343 3.33e-05 ***
temp:wind    1.840e-01  9.533e-02   1.930  0.05644 .
wind:rad    -1.381e-02  6.090e-03  -2.268  0.02541 *
```

The `temp:wind` interaction is close to significance, but we are ruthless in our pruning, so we take it out:

```
model4<-update(model3,~. - temp:wind)
summary(model4)
```

```
Coefficients:
              Estimate Std. Error t value Pr(>|t|)
(Intercept)  2.310e+02  1.082e+02   2.135 0.035143 *
temp        -5.442e+00  2.797e+00  -1.946 0.054404 .
wind        -1.080e+01  2.742e+00  -3.938 0.000150 ***
rad          2.405e-01  1.073e-01   2.241 0.027195 *
I(rad^2)    -2.010e-04  2.524e-04  -0.796 0.427698
I(temp^2)    4.484e-02  1.821e-02   2.463 0.015432 *
I(wind^2)    4.308e-01  1.020e-01   4.225 5.16e-05 ***
wind:rad    -9.774e-03  5.794e-03  -1.687 0.094631 .
```

Now the `wind:rad` interaction, which looked so significant in model3 (p = 0.02541), is evidently not significant, so we take it out as well:

```
model5<-update(model4,~. - wind:rad)
summary(model5)
```

```
Coefficients:
              Estimate Std. Error t value Pr(>|t|)
(Intercept)  2.985e+02  1.014e+02   2.942  0.00402 **
temp        -6.584e+00  2.738e+00  -2.405  0.01794 *
wind        -1.337e+01  2.300e+00  -5.810 6.89e-08 ***
rad          1.349e-01  8.795e-02   1.533  0.12820
I(rad^2)    -2.052e-04  2.546e-04  -0.806  0.42213
I(temp^2)    5.221e-02  1.783e-02   2.928  0.00419 **
I(wind^2)    4.652e-01  1.008e-01   4.617 1.12e-05 ***
```

There is no evidence to support retaining any of the 2-way interactions. What about the quadratic terms: the term in `rad^2` looks insignificant, so we take it out:

```
model6<-update(model5,~. - I(rad^2))
summary(model6)
```

```
Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) 291.16758  100.87723   2.886  0.00473 **
temp         -6.33955    2.71627  -2.334  0.02150 *
wind        -13.39674    2.29623  -5.834 6.05e-08 ***
rad           0.06586    0.02005   3.285  0.00139 **
I(temp^2)     0.05102    0.01774   2.876  0.00488 **
I(wind^2)     0.46464    0.10060   4.619 1.10e-05 ***

Residual standard error: 18.25 on 105 degrees of freedom
Multiple R-Squared: 0.713,      Adjusted R-squared: 0.6994
F-statistic: 52.18 on 5 and 105 DF,  p-value: < 2.2e-16
```

Now we are making progress. All the terms in model6 are significant. We should check the assumptions, using plot(model6):

There is a clear pattern of variance increasing with the mean of the fitted values. This is bad news (heteroscedasticity). Also, the normality plot is distinctly curved; again, this is bad news. Let’s try transformation of the response variable. There are no zeros in the response, so a log transformation is worth trying:

```
model7<-lm(log(ozone) ~ temp + wind + rad + I(temp^2) + I(wind^2))
summary(model7)
```

```
Coefficients:
              Estimate Std. Error t value Pr(>|t|)
(Intercept)  2.5538486  2.7359735   0.933  0.35274
temp        -0.0041416  0.0736703  -0.056  0.95528
wind        -0.2087025  0.0622778  -3.351  0.00112 **
rad          0.0025617  0.0005437   4.711 7.58e-06 ***
I(temp^2)    0.0003313  0.0004811   0.689  0.49255
I(wind^2)    0.0067378  0.0027284   2.469  0.01514 *

Residual standard error: 0.4949 on 105 degrees of freedom
```

```
Multiple R-Squared: 0.6882,     Adjusted R-squared: 0.6734 
F-statistic: 46.36 on 5 and 105 DF,  p-value:     0 
```

On the log(ozone) scale, there is no evidence for a quadratic term in temperature, so let's remove that:

```
model8<-update(model7,~. - I(temp^2))
summary(model8)
```

```
Coefficients:
              Estimate Std. Error t value Pr(>|t|)    
(Intercept)  0.7231644  0.6457316   1.120  0.26528    
temp         0.0464240  0.0059918   7.748 5.94e-12 ***
wind        -0.2203843  0.0597744  -3.687  0.00036 ***
rad          0.0025295  0.0005404   4.681 8.49e-06 ***
I(wind^2)    0.0072233  0.0026292   2.747  0.00706 ** 

Residual standard error: 0.4936 on 106 degrees of freedom
Multiple R-Squared: 0.6868,     Adjusted R-squared: 0.675 
F-statistic: 58.11 on 4 and 106 DF,  p-value:     0 
```

```
plot(model8)
```

The heteroscedasticity and the non-normality have been cured, but there is now a highly influential data point (number 17 on the Cook's plot). We should refit the model with this point left out, to see if the parameter estimates or their standard errors are greatly affected:

```
model9<-lm(log(ozone) ~ temp + wind + rad + I(wind^2),subset=(1:length(ozone)!=17))
summary(model9)
```

```
Coefficients:
              Estimate Std. Error t value Pr(>|t|)
(Intercept)  1.1932358  0.5990022   1.992 0.048963 *
temp         0.0419157  0.0055635   7.534 1.81e-11 ***
wind        -0.2208189  0.0546589  -4.040 0.000102 ***
rad          0.0022097  0.0004989   4.429 2.33e-05 ***
I(wind^2)    0.0068982  0.0024052   2.868 0.004993 **

Residual standard error: 0.4514 on 105 degrees of freedom
Multiple R-Squared: 0.6974,     Adjusted R-squared: 0.6859
F-statistic:  60.5 on 4 and 105 DF,  p-value:     0
```

Finally, plot(model9) shows that the variance and normality are well behaved, so we can stop at this point. We have found the minimal adequate model. It is on a scale of log(ozone concentration), all the main effects are significant, but there are no interactions, and there is a single quadratic term for wind speed (5 parameters in all, with 105 d.f. for error).

A more realistically complex example of multiple regression

In the next example we introduce two new difficulties: more explanatory variables, and fewer data points. It is another air pollution data frame, but the response variable in this case is sulphur dioxide concentration. There are 6 continuous explanatory variables:

```
pollute<-read.table("c:\\temp\\sulphur.dioxide.txt",header=T)
attach(pollute)
names(pollute)
```

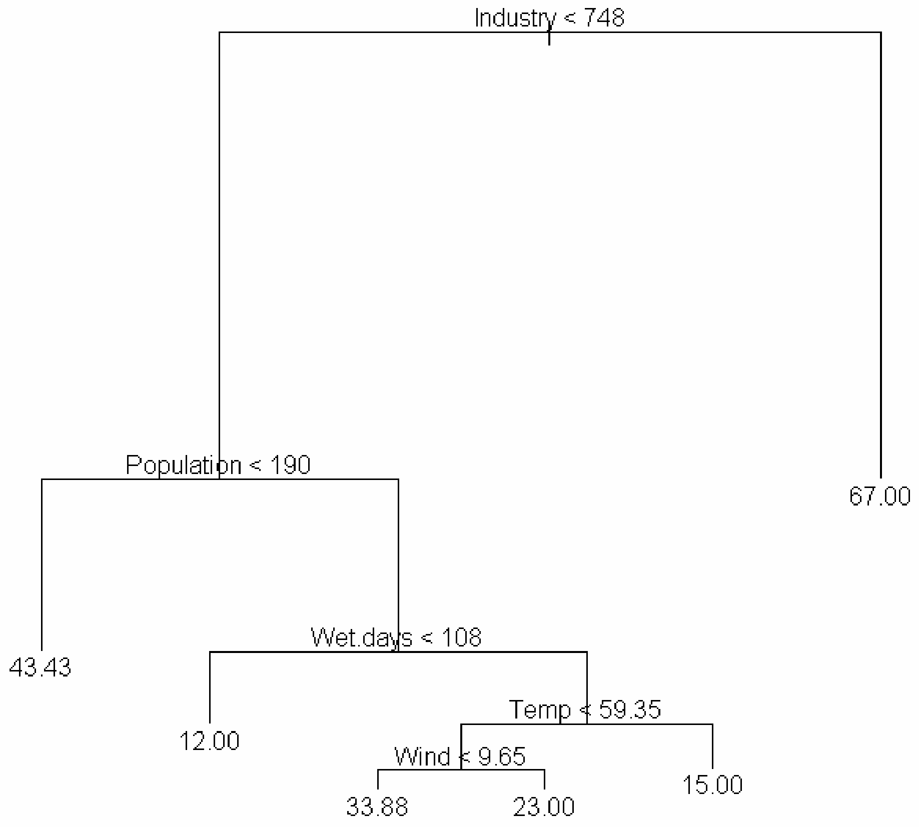
```
[1] "Pollution"  "Temp"       "Industry"   "Population" "Wind"
[6] "Rain"       "Wet.days"
```

Here are the 36 scatter plots:

```
pairs(pollute,panel=panel.smooth)
```

This time, let's begin with the tree model rather than the generalized additive model. A look at the pairs plots suggests that interactions may be more important than non-linearity in this case.

```
library(tree)
model<-tree(Pollution~.,data=pollute)
plot(model)
text(model)
```



This is interpreted as follows. The most important explanatory variable is Industry, and the threshold value separating low and high values of Industry is 748. The right hand branch of the tree indicates the mean value of air pollution for high levels of industry (67.00). The fact that this limb is unbranched means that no other variables explain a significant amount of the variation in pollution levels for high values of Industry. The left-hand limb does not show the mean values of pollution for low values of industry, because there are other significant explanatory variables. Mean values of pollution are only shown at the extreme ends of branches. For low values of Industry, the tree shows us that Population has a significant impact on air pollution. At low values of Population (<190) the mean level of air pollution was 43.43. For high values of Population, the number of Wet.days is significant. Low numbers of wet days (< 108) have mean pollution levels of 12.00 while Temperature has a significant impact on pollution for places where the number of wet days is large. At high temperatures (> 59.35 'F) the mean pollution level was 15.00 while at lower temperatures the run of Wind is important. For still air (Wind < 9.65) pollution was higher (33.88) than for higher wind speeds (23.00).

The virtues of tree-based models are numerous:

- they are easy to appreciated and to describe to other people
- the most important variables stand out
- interactions are clearly displayed
- non-linear effects are captured effectively
- the complexity of the behaviour of the explanatory variables is plain to see

We conclude that the interaction structure is highly complex. We shall need to carry out the linear modelling with considerable care.

Start with some elementary calculations. With 6 explanatory variables, how many interactions might we fit? Well, there are 5 + 4 + 3 + 2 + 1 = 15 two-way interactions for a start. Plus 20 three-way, 15 four-way and 6 five-way interactions, plus one six-way interaction for good luck. Then there are quadratic terms for each of the 6 explanatory variables. So we are looking at about 70 parameters that might be estimated from the data. But how many data points have we got?

```
length(Pollution)
```

```
[1] 41
```

Oops! We are planning to estimate almost twice as many parameters as there are data points. That's taking over-parameterization to new heights. We already know that you can't estimate more parameter values than there are data points (i.e. a maximum of 41 parameters). But we also know that when we fit a saturated model to the data, it has no explanatory power (there are no degrees of freedom, so the model, by explaining everything, ends up explaining nothing at all). There is a useful rule of thumb: *don't try to estimate more than n/3 parameters during a multiple regression*. In the present case *n* = 41 so the rule of thumb is suggesting that we restrict ourselves to estimating about 41/3 ≈ 13 parameters at any one time. We know from the tree model that the interaction structure is going to be complicated so we shall concentrate on that. We begin, therefore, by looking for curvature, to see if we can eliminate this:

```
model1<-lm(Pollution~Temp+I(Temp^2)+Industry+I(Industry^2)+Population+I(Population^2)+Wind+I(Wind^2)+Rain+I(Rain^2)+Wet.days+I(Wet.days^2))
summary(model1)
```

```
Coefficients:
                  Estimate Std. Error t value Pr(>|t|)
(Intercept)     -6.641e+01  2.234e+02  -0.297  0.76844
Temp             5.814e-01  6.295e+00   0.092  0.92708
I(Temp^2)       -1.297e-02  5.188e-02  -0.250  0.80445
Industry         8.123e-02  2.868e-02   2.832  0.00847 **
I(Industry^2)   -1.969e-05  1.899e-05  -1.037  0.30862
Population      -7.844e-02  3.573e-02  -2.195  0.03662 *
I(Population^2)  2.551e-05  2.158e-05   1.182  0.24714
Wind             3.172e+01  2.067e+01   1.535  0.13606
I(Wind^2)       -1.784e+00  1.078e+00  -1.655  0.10912
Rain             1.155e+00  1.636e+00   0.706  0.48575
I(Rain^2)       -9.714e-03  2.538e-02  -0.383  0.70476
```

```
Wet.days        -1.048e+00  1.049e+00  -0.999  0.32615
I(Wet.days^2)    4.555e-03  3.996e-03   1.140  0.26398

Residual standard error: 14.98 on 28 degrees of freedom
Multiple R-Squared: 0.7148,     Adjusted R-squared: 0.5925
F-statistic: 5.848 on 12 and 28 DF,  p-value: 5.868e-005
```

So that's our first bit of good news. There is no evidence of curvature for any of the 6 explanatory variables. Only the main effects of Industry and Population are significant in this (over-parameterized) model. Now we need to consider the interaction terms. We do not fit interaction terms without both the component main effects, so we can not fit all the two-way interaction terms at the same time (that would be 15 + 6 = 21 parameters; well above the rule of thumb value of 13). One approach is to fit the interaction terms in randomly selected pairs. With all 6 main effects, we can afford to try 13 – 6 = 7 interaction terms at a time. We'll try this. Make a vector containing the names of the 15 two-way interactions:

```
interactions<-c("ti","tp","tw","tr","td","ip","iw","ir","id","pw","pr","pd","wr","wd","rd")
```

Now shuffle the interactions into random order using sample without replacement:

```
sample(interactions)
```

```
[1] "wr" "wd" "id" "ir" "rd" "pr" "tp" "pw" "ti" "iw" "tw" "pd" "tr" "td" "ip"
```

It would be sensible and pragmatic to test the two-way interactions in 3 models, each containing 5 different two-way interaction terms:

```
model2<-
lm(Pollution~Temp+Industry+Population+Wind+Rain+Wet.days+Wind:Rain+
Wind:Wet.days+Industry:Wet.days+Industry:Rain+Rain:Wet.days)
model3<-
lm(Pollution~Temp+Industry+Population+Wind+Rain+Wet.days+Population:R
ain+Temp:Population+Population:Wind+Temp:Industry+Industry:Wind)
model4<-
lm(Pollution~Temp+Industry+Population+Wind+Rain+Wet.days+Temp:Wind+
Population:Wet.days+Temp:Rain+Temp:Wet.days+Industry:Population)
```

Extracting only the interaction terms from the 3 models, we see:

```
Industry:Rain     -1.616e-04  9.207e-04  -0.176 0.861891
Industry:Wet.days  2.311e-04  3.680e-04   0.628 0.534949
Wind:Rain          9.049e-01  2.383e-01   3.798 0.000690 ***
Wind:Wet.days     -1.662e-01  5.991e-02  -2.774 0.009593 **
Rain:Wet.days      1.814e-02  1.293e-02   1.403 0.171318

Temp:Industry   -1.643e-04  3.208e-03  -0.051   0.9595
Temp:Population  1.125e-03  2.382e-03   0.472   0.6402
Industry:Wind    2.668e-02  1.697e-02   1.572   0.1267
Population:Wind -2.753e-02  1.333e-02  -2.066   0.0479 *
Population:Rain  6.898e-04  1.063e-03   0.649   0.5214

Temp:Wind            1.261e-01  2.848e-01   0.443  0.66117
Temp:Rain           -7.819e-02  4.126e-02  -1.895  0.06811 .
```

```
Temp:Wet.days        1.934e-02  2.522e-02   0.767  0.44949
Industry:Population  1.441e-06  4.178e-06   0.345  0.73277
Population:Wet.days  1.979e-05  4.674e-04   0.042  0.96652
```

The next step might be to put all of the significant or close-to-significant interactions into the same model, and see which survive:

```
model5<-
lm(Pollution~Temp+Industry+Population+Wind+Rain+Wet.days+Wind:Rain+
Wind:Wet.days+Population:Wind+Temp:Rain)
summary(model5)
```

```
Coefficients:
                  Estimate Std. Error t value Pr(>|t|)
(Intercept)     323.054546 151.458618   2.133 0.041226 *
Temp             -2.792238   1.481312  -1.885 0.069153 .
Industry          0.073744   0.013646   5.404 7.44e-06 ***
Population        0.008314   0.056406   0.147 0.883810
Wind            -19.447031   8.670820  -2.243 0.032450 *
Rain             -9.162020   3.381100  -2.710 0.011022 *
Wet.days          1.290201   0.561599   2.297 0.028750 *
Temp:Rain         0.017644   0.027311   0.646 0.523171
Population:Wind  -0.005684   0.005845  -0.972 0.338660
Wind:Rain         0.997374   0.258447   3.859 0.000562 ***
Wind:Wet.days    -0.140606   0.053582  -2.624 0.013530 *
```

We certainly don’t need Temp:Rain

```
model6<-update(model5,~. – Temp:Rain)
```

or Population:Wind

```
model7<-update(model6,~. – Population:Wind)
```

All the terms in model7 are significant.   Time for a check on the behaviour of the model:

```
plot(model7)
```

That’s not bad at all. But what about the higher-order interactions? One way to proceed is to specify the interaction level using ^3 in the model formula, but if you do this, you will find that we run out of degrees of freedom straight away. A pragmatic option is to fit three way terms for the variables that already appear in 2-way interactions: in our case, that is just one term: Wind:Rain:Wet.days

```
model8<-update(model7,~. + Wind:Rain:Wet.days)
summary(model8)
```

```
Coefficients:
                     Estimate Std. Error t value Pr(>|t|)
(Intercept)        278.464474  68.041497   4.093 0.000282 ***
Temp                -2.710981   0.618472  -4.383 0.000125 ***
Industry             0.064988   0.012264   5.299  9.1e-06 ***
Population          -0.039430   0.011976  -3.293 0.002485 **
```

```
Wind                 -7.519344   8.151943  -0.922 0.363444
Rain                 -6.760530   1.792173  -3.772 0.000685 ***
Wet.days              1.266742   0.517850   2.446 0.020311 *
Wind:Rain             0.631457   0.243866   2.589 0.014516 *
Wind:Wet.days        -0.230452   0.069843  -3.300 0.002440 **
Wind:Rain:Wet.days    0.002497   0.001214   2.056 0.048247 *

Residual standard error: 11.2 on 31 degrees of freedom
Multiple R-Squared: 0.8236,     Adjusted R-squared: 0.7724
F-statistic: 16.09 on 9 and 31 DF,  p-value: 2.231e-009
```

That’s enough for now. I’m sure you get the idea. Multiple regression is difficult, time consuming, and always vulnerable to subjective decisions about what to include and what to leave out. The linear modelling confirms the early impression from the tree model: for low levels of industry, the SO_2 level depends in a simple way on population (people tend to want to live where the air is clean) and in a complicated way on daily weather (the 3-way interaction between wind, total rainfall and the number of wet days (i.e. on rainfall intensity)). Note that the relationship between pollution and population in the initial scatterplot suggested a positive correlation between these two variables, not the negative relationship we discovered by statistical modelling. This is one of the great advantages of multiple regression.

Common problems arising in multiple regression

- differences in the measurement scales of the explanatory variables, leading to large variation in the sums of squares and hence to an ill-conditioned matrix
- multicollinearity is which there is a near linear relation between two of the explanatory variables leading to unstable parameter estimates
- rounding errors during the fitting procedure
- non-independence of groups of measurments
- temporal or spatial correlation amongst the explanatory variables
- pseudoreplication and a host of others (see Wetherill et al. 1986 for a detailed discussion).