Computations for Linear and Generalized Additive Models

J.H. Maindonald*

These notes

review the theory of linear models, demonstrating the utility of Householder reflections, the QR decomposition and the Cholesky decomposition both for computation and for theory;
[c.f., Chapter 1 of Wood (2006)]

introduce computational, theoretical and practical aspects of regression splines, smoothing splines and penalized splines. This leads into an introduction to Generalized Additive models.
[Maindonald & Braun (2010, Section 7.5, pp. 231–238)]

demonstrate the use of resmampling and simulation methods for comparing models;
[See Subsection 5.2 for references]

give brief summary details of the theory of Generalized Linear Models, with logistic regression and Poisson regression as special cases. The extension into Generalized Additive models with an exponential error distribution follows naturally.
[Maindonald & Braun (2010, Sections 8.1 – 8.5); Wood (2006, Sections 2.1 – 2.3)]

*Centre for Mathematics & Its Applications, Australian National University, Canberra ACT 0200, Australia.

September 5, 2012: Reprinted with corrections to Section 2.6 & elsewhere.

Contents

1. Linear Models – Key Concepts

1.1. Terminology

Note the distinction between fixed and random effects. In the straight line model

$$y_i = \alpha + \beta x_i + \epsilon_i, \quad i = 1, 2, \ldots n \tag{1}$$

α and β are fixed effects, while the ϵ_i are random.

A common assumption is that the ϵ_i are distributed as N(0, σ^2), independently between observations. This is commonly known as the iid (independently and identically distributed) normal asssumption.

In a matrix formulation, the model can be written:

$$\mathbf{y} = \mathbf{X}\boldsymbol{\beta} + \boldsymbol{\epsilon},$$

For least squares estimation, $\boldsymbol{\beta}$ will be chosen to minimize $\|\boldsymbol{\epsilon}\|^2$.

In the straight line of Equation 1

$$\mathbf{X} = \begin{bmatrix} 1 & x_1 \\ 1 & x_2 \\ \ldots & \ldots \\ 1 & x_n \end{bmatrix}$$

More generally

$$\mathbf{X} = \begin{bmatrix} x_{11} & x_{12} & \ldots & x_{1p} \\ x_{21} & x_{22} & \ldots & x_{2p} \\ \ldots & \ldots & & \\ x_{n1} & x_{n2} & \ldots & x_{np} \end{bmatrix}$$

In most applications the first column is a column of 1's, i.e., $x_{i1} = 1 \;\; (i = 1, \ldots, n)$. Note that the columns of $\mathbf{X}$ can be arbitrary functions of the explanatory variables. There are several ways in which this will be exploited. The matrix $\mathbf{X}$ has the name "model matrix".

1.2. Why use least squares?

Least squares may be used because it seems intuitively sensible. Or it may be justified by maximum likelihood, assuming independently and identically distributed normal errors.

The maximum likelihood approach is more fundamental. It gives a methodology in contexts where least squares is clearly unsatisfactory or does not provide an answer. Such contexts include (these are not mutually exclusive):

- observations have unequal variances;
- data have a dependence structure. For example, maximum likelihood shows how to get estimates for time series models that assume sequential dependence;
- distributions are not normal.

1.3. Basis Functions

We begin by defining what we mean by a "linear model". Define basis functions

$$\phi_1(x_1, x_2, \ldots x_k), \phi_2(x_1, x_2, \ldots x_k), \ldots, \phi_p(x_1, x_2, \ldots x_k)$$

In the simplest case $p = k$ and

$$\phi_1(x_1, x_2, \ldots x_p) = x_1, \phi_2(x_1, x_2, \ldots x_p) = x_2, \ldots, \phi_p(x_1, x_2, \ldots x_p) = x_p$$

Then any function with values on the real line such that

$$f(x_1, x_2, \ldots x_k) = \beta_1\, \phi_1(x_1, x_2, \ldots x_k) + \beta_2\, \phi_2(x_1, x_2, \ldots x_k) + \ldots + \beta_p\, \phi_p(x_1, x_2, \ldots x_k)$$

where the elements of $\boldsymbol{\beta} = (\beta_1, \beta_2, \ldots \beta_p)$ are the only unknowns, specifies a linear model. In words, any model in which E[y] is a linear combination of the ϕ_i is a linear model.

Note three important non-trivial special cases:

- Several ϕ_i's are defined that together account for a factor term. See below for the definition of "term".
- Several ϕ_i's are defined that together account for a spline term.
- Interaction terms may be created by multiplying together columns that relate to other terms in the model.

The model is linear in the values that the ϕ's take on the sample data. It is not, in general, linear in the x_i's.

1.4. Terms – groups of basis functions

A factor with k levels requires, assuming that the model already has a constant term, $k - 1$ basis elements to represent it. These basis elements together constitute a "term", in this case a factor. The Wilkinson & Rogers notation makes it possible to specify the factor as a term in the model, leaving code that is called by R's `lm()` function to determine the basis functions that are needed.

Similarly a 4 degree of freedom spline term requires four basis elements. Again, code that is called by R's `bs()` (B-splines) or `ns()` (natural splines) function will determine the basis functions.

Quite generally, the basis functions $\phi_1, \phi_2, \ldots, \phi_p$ can be categorized into groups, with one group for each term the model, thus:

$$\underbrace{\phi_1, \ldots, \phi_{m_1}}_{\text{Term1}}, \underbrace{\phi_{m_1+1}, \ldots, \phi_{m_2}}_{\text{Term2}}, \ldots$$

1.4.1. Factor basis functions

A simple example will do for now. Consider the sugar data frame in the R package. There are three levels of trt – Control, A, B and C. Type into R:

```
> contrasts(sugar$trt)
        A B C
Control 0 0 0
A       1 0 0
B       0 1 0
C       0 0 1
```

The first factor level, ie Control, is taken as the baseline. The parameter associated with A is then its difference from the baseline, and similarly for B and C.

Now see how this works in practice:

```
> sugar.lm <- lm(weight ~ trt, data=sugar)
> # Now examine the model matrix (commonly denoted as X)
> model.matrix(sugar.lm)
   (Intercept) trtA trtB trtC
1            1    0    0    0
2            1    0    0    0
3            1    0    0    0
4            1    1    0    0
5            1    1    0    0
6            1    1    0    0
7            1    0    1    0
8            1    0    1    0
9            1    0    1    0
10           1    0    0    1
11           1    0    0    1
12           1    0    0    1
attr(,"assign")
[1] 0 1 1 1
attr(,"contrasts")
attr(,"contrasts")$trt
[1] "contr.treatment"
```

In the above, (Intercept) is really Control. The columns trtA, trtB and trtC measure differences from Control

Another possibility is

```
> sugar.lm0 <- lm(weight ~ -1 + trt, data=sugar)
> model.matrix(sugar.lm0)
   trtControl trtA trtB trtC
1           1    0    0    0
2           1    0    0    0
3           1    0    0    0
4           0    1    0    0
5           0    1    0    0
6           0    1    0    0
7           0    0    1    0
8           0    0    1    0
9           0    0    1    0
10          0    0    0    1
```

```
11          0    0    0    1
12          0    0    0    1
attr(,"assign")
[1] 1 1 1 1
attr(,"contrasts")
attr(,"contrasts")$trt
[1] "contr.treatment"
```

Omission of the "constant term" from the formula forces the use of one parameter for each different factor level. With this parameterization no restriction on the parameters is needed.

1.4.2. Spline basis functions

Section 4 will introduce spline basis functions, Spline basis functions are, in essence, a construction kit for constructing curves that have some predefined flexibility. See

2. Solving Least Squares Systems

The QR decomposition, usually achieved by a sequence of Householder reflections, is widely used in the practical computer solution of linear least squares systems. Additionally, the decomposition is a good starting point for deriving various theoretical results.

Certain special types of least squares problems are more efficiently handled by other methods, or by one or other adaptation of QR methods. Sparse linear systems are an important special case. For very large linear least squares systems, highly efficient computations and/or minimization of storage requirements can be crucial to completing calculations within reasonable time or even to getting calculations to run at all. See for example Bates (2006); Koenker and & Ng (2003).

An understanding of the computational details can assist greatly when efficient computation is important, e.g., a computation is repeated a large number of times. Such an understanding can also be important in order to take advantage of multiple processor systems. Algorithms that are highly efficient on single processor systems may require substantial adaptation to get maximum advantage from multiple processor systems. Or different algorithms may be required.

2.1. Properties of orthonormal matrices

Orthonormal matrices will be important for the discussion that follows. A matrix $\mathbf{H}$ is orthonormal if and only if $\mathbf{H}^\top\mathbf{H} = \mathbf{I}$.

Properties of orthonormal matrices

Two important properties are:

1. Pre-multiplication by an orthonormal matrix $\mathbf{H}$ leaves the length of any vector $\mathbf{x}$ unchanged. This follows because

2. Pre-multiplication of each of two vectors $\mathbf{x}_1$ and $\mathbf{x}_2$ by an orthonormal matrix $\mathbf{H}$ leaves their inner product unchanged.

The first of these results is a special case of the second, obtained by setting $\mathbf{x} = \mathbf{x}_1 = \mathbf{x}_2$.

These results are an immediate consequence of the orthonormal matrix property that $\mathbf{H}^\top\mathbf{H}$ is the identity matrix $\mathbf{I}$:

$$\begin{aligned}(\mathbf{H}\mathbf{x}_1)^\top \mathbf{H}\mathbf{x}_2 &= \mathbf{x}_1^\top \mathbf{H}^\top \mathbf{H}\,\mathbf{x}_2 \\ &= \mathbf{x}_1^\top (\mathbf{H}^\top\mathbf{H})\,\mathbf{x}_2 \\ &= \mathbf{x}_1^\top \mathbf{x}_2\end{aligned}$$

Geometrically, the statements are that orthonormal rotations preserve the lengths of vectors, and angles between vectors.

The QR decomposition

Given an $n \times p$ matrix $\mathbf{X}$, the QR decomposition derives an orthonormal matrix $\mathbf{Q}$ and an upper triangular matrix $\mathbf{R}$ such that

$$\mathbf{Q}^\top\mathbf{X} = \begin{bmatrix}\mathbf{R} \\ \mathbf{0}\end{bmatrix}$$

Then

$$\mathbf{X} = \mathbf{Q}\begin{bmatrix}\mathbf{R} \\ \mathbf{0}\end{bmatrix} \tag{2}$$

There are a number of different algorithms that can be used to derive this decomposition. Householder reflections are the most widely used. The algorithm is efficient and has good numerical properties.

Observe that the product of two orthonormal matrices is an orthonormal matrix. This follows because, given orthonormal matrices $\mathbf{Q}_1$ and $\mathbf{Q}_2$:

$$\begin{aligned}(\mathbf{Q}_2\mathbf{Q}_1)^\top(\mathbf{Q}_2\mathbf{Q}_1) &= (\mathbf{Q}_1^\top\mathbf{Q}_2^\top)\mathbf{Q}_2\mathbf{Q}_1 \\ &= \mathbf{Q}_1^\top(\mathbf{Q}_2^\top\mathbf{Q}_2)\mathbf{Q}_1 \\ &= \mathbf{Q}_1^\top\mathbf{Q}_1 \qquad \text{(this follows because } \mathbf{Q}_2^\top\mathbf{Q}_2 = \mathbf{I}\text{)} \\ &= \mathbf{I}\end{aligned}$$

2.2. Householder Reflections, and QR

A Householder reflection matrix has the form:

$$\mathbf{H} = \mathbf{I} - \gamma\,\mathbf{u}\mathbf{u}^\top, \quad \text{where} \quad \gamma = \frac{2}{||\mathbf{u}||^2}$$

It is a straightforward use of matrix algebra to show that Householder reflection matrices are orthonormal.

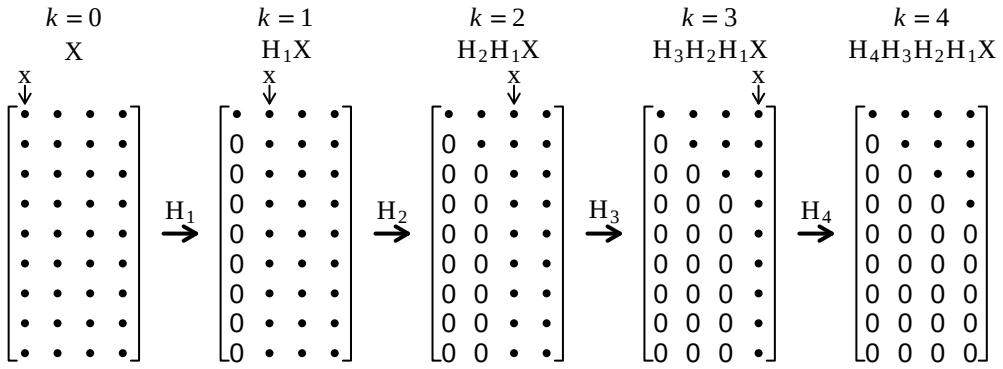

Figure 1: Diagrammatic representation of the sequence of steps that are applied to a 9×4 model matrix $\mathbf{X}$ when Householder reflections are used in least squares calculations. The vector $\mathbf{x}$ whose final $n - k$ elements are used in forming $\mathbf{u}_k$ and hence $\mathbf{H}_k$ is identified, for each of the successive steps, with an arrow (↓). Thus $\mathbf{H}_1$ is formed using the first column of $\mathbf{X}$ ($k = 0$), $\mathbf{H}_2$ is formed using the second and later elements in the second column of $\mathbf{H}_1\mathbf{X}$ ($k = 1$), $\mathbf{H}_3$ is formed using the third and later elements in the third column of $\mathbf{H}_2\mathbf{H}_1\mathbf{X}$ ($k = 2$), and so on. The same sequence of Householder reflections is applied to $\mathbf{y}$.

It will be shown that the QR decomposition can be achieved by constructing a sequence of Householder 'reflections', i.e., pre-multiplications by Householder matrices. The matrices $\mathbf{H}_1$, $\mathbf{H}_2$, ... are chosen so that they successively reduce to zero below diagonal elements in columns 1, 2, ... p of $\mathbf{X}$. The same sequence of reflections is applied also to $\mathbf{y}$.

The first reflection yields $\mathbf{H}_1\mathbf{X}$. The second reflection, which operates only on rows of $\mathbf{H}_1\mathbf{X}$ subsequent to the first, replaces below diagonal elements in the second column with zeros, yielding $\mathbf{H}_2\mathbf{H}_1\mathbf{X}$.

Figure 1 shows diagrammatically the sequence of Householder reflections, applied to a 9×4 model matrix $\mathbf{X}$, for reducing a 9×4 model matrix $\mathbf{X}$ to upper triangular form.

2.2.1. Properties of Householder reflection matrices

Given a vector $\mathbf{x}$ with n elements, then for any k ($k < n$) there is a Householder matrix $\mathbf{H}$ such that

$$\mathbf{Hx} = \begin{bmatrix} x_1 \\ x_2 \\ . \\ x_{k-1} \\ \eta_k \\ 0 \\ . \\ . \\ 0 \end{bmatrix}$$

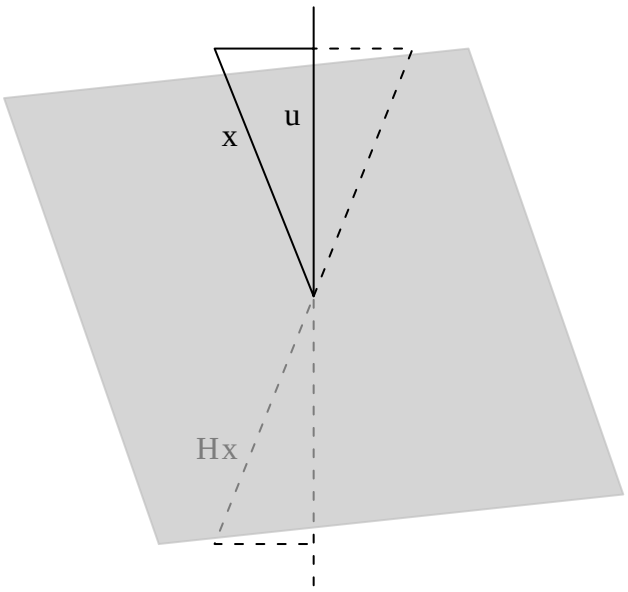

Figure 2: Geometrical representation of the reflection of $\mathbf{x}$ in the plane that is orthogonal to $\mathbf{u}$. Here $\mathbf{H} = I - \gamma\,\mathbf{u}\mathbf{u}^\top$, with $u_1 = x_1 + \text{sgn}(x_1)\|\mathbf{x}\|$, $u_i = x_i, i \geq 2$.

Observe that

$$(\mathbf{I} - \gamma\,\mathbf{u}\mathbf{u}^\top)\mathbf{x} = \mathbf{x} - \mathbf{u}\frac{2}{\|\mathbf{u}\|^2}(\mathbf{u}^\top\mathbf{x})$$

Consider the case $k = 1$. It is convenient to write $u_1 = x_1 + \alpha$. Set $u_i = x_i \quad (i = 2, \ldots n)$. Thus

$$\mathbf{u} = \begin{bmatrix} x_1 + \alpha \\ x_2 \\ . \\ . \\ x_n \end{bmatrix}$$

Now choose α so that $\dfrac{2}{\|\mathbf{u}\|^2}(\mathbf{u}^\top\mathbf{x}) = 1$. Then

$$\begin{aligned} 2\,(\|\mathbf{x}\|^2 + \alpha x_1) &= \|\mathbf{u}\|^2 \\ &= \|\mathbf{x}\|^2 + 2\alpha x_1 + \alpha^2 \end{aligned}$$

Thus

$$\alpha^2 = \|\mathbf{x}\|^2 \qquad \alpha = \pm\|\mathbf{x}\|$$

Figure 2 gives a geometrical representation of the reflection of $\mathbf{x}$ in the plane that is orthogonal to $\mathbf{u}$. Note that $\mathbf{Hx}$ is a positive or negative multiple $(1, 0, \ldots 0)$.

For numerical stability, we require $\alpha = \text{sgn}(x_1)\|\mathbf{x}\|$, thus ensuring that x_1 and α have the same sign. It follows that

$$\|\mathbf{u}\|^2 = 2\,\alpha(x_1 + \alpha)$$

$$\mathbf{u}^\top \mathbf{x} = \alpha(x_1 + \alpha)$$

$$\begin{aligned}\gamma &= \frac{2}{\|\mathbf{u}\|^2} \\ &= \frac{1}{\alpha(x_1 + \alpha)}\end{aligned}$$

More generally, choose

$$\mathbf{u}_{(k)} = \begin{bmatrix} 0 \\ . \\ 0 \\ x_k + \alpha_k \\ x_{k+1} \\ . \\ . \\ x_n \end{bmatrix}$$

where

$$\alpha_k = \operatorname{sgn}(x_k) \sqrt{\sum_{i=k}^{n} x_i^2}$$

Then

$$\mathbf{H}_k = \mathbf{I} - \gamma\, \mathbf{u}_{(k)} \mathbf{u}_{(k)}^\top, \quad \text{where} \quad \gamma = \frac{2}{\|\mathbf{u}_{(k)}\|^2} = \frac{1}{\alpha_k(x_k + \alpha_k)}$$

The sequence of reflections $\mathbf{H}_1$, $\mathbf{H}_2$, ..., with $\mathbf{x}$ chosen as indicated in Figure 1, then achieves the result that is required for the QR reduction.

2.2.2. Simplified calculations for row *k*

Let

$$\mathbf{X}^{(k)} = \mathbf{H}_{k-1} \ldots \mathbf{H}_2 \mathbf{H}_1 \mathbf{X}$$

Then, ignoring a possible change of sign of all elements in the row:

$$r_{kk} = |\alpha_k| = \|\mathbf{x}^{(k)}\| \tag{3}$$

$$r_{kj} = \frac{\mathbf{x}_k{}^{(k)\top} \mathbf{x}_j{}^{(k)}}{|\alpha_k|} \quad (j > k) \tag{4}$$

Equation 4 follows because pre-multiplication by an orthonormal matrix preserves inner products. As the transformation reduces to zero all elements in column k that are in rows subsequent to row k, the inner product following the transformation is $|\alpha_k|\, r_{kj}$. This has to equal the inner product prior to the transformation, i.e., $\mathbf{x}_k{}^{(k)\top} \mathbf{x}_j{}^{(k)}$.

Ignoring any change of sign in row k has the consequence that the transformation is no longer precisely a Householder reflection. We are still, however, premultiplying by an orthonormal matrix.

2.2.3. Diagonal elements that are zero

If following application of the first $k-1$ Householder transformations x_k is already zero, there is nothing more to do. The kth column of $\mathbf{X}$ is then a linear combination of the initial k columns.

2.3. Least Squares

Form

$$\mathbf{Q}^\top \mathbf{X} = \begin{bmatrix} \mathbf{R} \\ \mathbf{0} \end{bmatrix}, \quad \mathbf{Q}^\top \mathbf{y} = \begin{bmatrix} \mathbf{f} \\ \mathbf{r} \end{bmatrix} \tag{5}$$

where $\mathbf{R}$ $(p \times p)$ is upper triangular, $\mathbf{0}$ is an $(n-p) \times p$ array of zeros, $\mathbf{f}$ is $p \times 1$ and $\mathbf{r}$ is $(n-p) \times 1$. Then

$$\begin{aligned} \|\mathbf{y} - \mathbf{X}\boldsymbol{\beta}\|^2 &= \|\mathbf{Q}^\top \mathbf{y} - \mathbf{Q}^\top \mathbf{X}\boldsymbol{\beta}\|^2 \\ &= \left\| \begin{array}{l} \mathbf{f} - \mathbf{R}\boldsymbol{\beta} \\ \mathbf{r} \end{array} \right\|^2 \\ &= \|\mathbf{f} - \mathbf{R}\boldsymbol{\beta}\|^2 + \|\mathbf{r}\|^2 \end{aligned}$$

Hence $\|\mathbf{y} - \mathbf{X}\boldsymbol{\beta}\|^2$ is minimized by choosing $\boldsymbol{\beta} = \mathbf{b}$ such that

$$\mathbf{R}\mathbf{b} = \mathbf{f} \tag{6}$$

i.e.

$$\begin{aligned} r_{11}b_1 + r_{12}b_2 \ldots + r_{1p}b_p &= f_1 \\ r_{22}b_2 \ldots + r_{2p}b_p &= f_2 \\ r_{pp}b_p &= f_2 \end{aligned} \tag{7}$$

Solve first for b_p, then for b_{p-1}, $\ldots b_p$. The procedure is called *backward elimination*. The estimate $\mathbf{b}$ is uniquely determined, providing that all diagonal elements of $\mathbf{R}$ are non-zero. If r_{kk} is zero, then b_k can be chosen arbitrarily. The convention used in R's `lm()` function is to set $b_k = 0$ for purposes of completing the calculations, but then return a result that has $b_k = NA$. Irrespective of the action taken to resolve the indeterminacy, the residual sum of squares is $\|\mathbf{r}\|^2$.

Note that:

- Omission of the final column from $\mathbf{R}$ and the final element from $\mathbf{f}$ yields the system of equations that apply when the final column is omitted from $\mathbf{X}$.
- Consider the addition of a further $((p+1)^{th})$ variable $\mathbf{x}_{p+1}$. The reduced system of equations becomes:

$$\mathbf{Q}^\top \begin{bmatrix} \mathbf{X} & \mathbf{x}_{p+1} \end{bmatrix} = \begin{bmatrix} \mathbf{Q}_f^\top \mathbf{X} & \mathbf{Q}_f^\top \mathbf{x}_{p+1} \\ \mathbf{Q}_r^\top \mathbf{X} & \mathbf{Q}_r^\top \mathbf{x}_{p+1} \end{bmatrix} \tag{8}$$

$$= \begin{bmatrix} \mathbf{R} & \mathbf{Q}_f^\top \mathbf{x}_{p+1} \\ \mathbf{0} & \mathbf{Q}_r^\top \mathbf{x}_{p+1} \end{bmatrix} \tag{9}$$

We now apply a Householder operation that leaves rows 1 to p unchanged, and reduces to zero elements after the $(p+1)^{th}$ in $\mathbf{Q}_r^\top \mathbf{x}_{p+1}$. We obtain (possibly after a change of sign):

$$r_{p+1,p+1} = \alpha = \|\mathbf{Q}_r^\top \mathbf{x}_{p+1}\|$$

The element in the $(p+1)^{th}$ row of $\mathbf{Q}^\top \mathbf{y}$ is replaced by

$$\frac{(\mathbf{Q}_r^\top \mathbf{x}_{p+1})^\top \mathbf{r}}{\alpha}$$

2.3.1. Normal equations

Pre-multiplication of the two sides of $\mathbf{Rb} = \mathbf{f}$ (Equation 5) by $\mathbf{R}^\top$ yields respectively:

$$\mathbf{R}^\top \mathbf{Rb} == \mathbf{X}^\top \mathbf{Q}\mathbf{Q}^\top \mathbf{Xb} = \mathbf{X}^\top \mathbf{Xb} \qquad \text{(left side)}$$

and

$$\mathbf{R}^\top \mathbf{f} = \begin{bmatrix}\mathbf{R}^\top & \mathbf{0}\end{bmatrix} \begin{bmatrix}\mathbf{f} \\ \mathbf{r}\end{bmatrix} = \mathbf{X}^\top \mathbf{Q}\mathbf{Q}^\top \mathbf{y} = \mathbf{X}^\top \mathbf{y} \qquad \text{(right side)}$$

Hence the so-called normal equations

$$\mathbf{X}^\top \mathbf{Xb} = \mathbf{X}^\top \mathbf{y}$$

Note that there has not been any assumption of non-singularity.

Writing $\mathbf{Xb} = \boldsymbol{\mu}$ (the fitted values), the normal equations can then be written

$$\mathbf{X}^\top \boldsymbol{\mu} = \mathbf{X}^\top \mathbf{y}$$

This form of the least squares equations carries over to generalized linear models, in the special case where the canonical link is specified.

2.3.2. The hat matrix

Write

$$\mathbf{Q}^\top = \begin{bmatrix}\mathbf{Q}_f^\top \\ \mathbf{Q}_r^\top\end{bmatrix}, \quad \text{where} \quad \mathbf{Q}_f^\top \quad \text{is} \quad p \times n, \quad \text{and} \quad \mathbf{Q}_r^\top \quad \text{is} \quad n - p \times n \tag{10}$$

From Equation 2

$$\mathbf{X} = \begin{bmatrix}\mathbf{Q}_f & \mathbf{Q}_r\end{bmatrix} \begin{bmatrix}\mathbf{R} \\ \mathbf{0}\end{bmatrix} = \mathbf{Q}_f \mathbf{R} \tag{11}$$

From

$$\mathbf{Q}^\top \mathbf{y} = \begin{bmatrix}\mathbf{f} \\ \mathbf{r}\end{bmatrix}, \quad \text{i.e.} \begin{bmatrix}\mathbf{Q}_f^\top \\ \mathbf{Q}_r^\top\end{bmatrix} \mathbf{y} = \begin{bmatrix}\mathbf{f} \\ \mathbf{r}\end{bmatrix}, \quad \text{it follows that} \quad \mathbf{f} = \mathbf{Q}_f^\top \mathbf{y} \tag{12}$$

Then

$$\begin{aligned}\hat{\mathbf{y}} &= \mathbf{X}\mathbf{b} \\ &= \mathbf{Q}_f\mathbf{R}\mathbf{b} && \text{(from Equation 10)} \\ &= \mathbf{Q}_f\mathbf{f} \\ &= \mathbf{Q}_f\mathbf{Q}_f^\top\mathbf{y} && \text{(from Equation 12)}\end{aligned}$$

Hence:

$$\mathbf{y} - \widehat{\mathbf{y}} = (\mathbf{I}_n - \mathbf{Q}_f\mathbf{Q}_f^\top)\mathbf{y} \tag{13}$$

The matrix $\mathbf{A} = \mathbf{Q}_f\mathbf{Q}_f^\top$ is known as the hat matrix. If $\mathbf{X}^\top\mathbf{X}$ is non-singular, then it equals the more familiar expression

$$\mathbf{X}(\mathbf{X}^\top\mathbf{X})^{-1}\mathbf{X}^\top$$

2.3.3. Computational issues

1. Algebraically, one can write

   $$\mathbf{Q}^\top\mathbf{X} = \mathbf{H}_p\mathbf{H}_{p-1}\ldots\mathbf{H}_1\mathbf{X}$$

   Neither $\mathbf{Q}^\top$ nor any of the matrices $\mathbf{H}_i$ need be formed explicitly. For large matrices, this would be an impossibly computationally expensive way to do the computations. In

   $$\mathbf{H}\mathbf{X} = (\mathbf{I} - \gamma\,\mathbf{u}\mathbf{u}^\top)\mathbf{X}$$

   it is important to do the calculation as

   $$\mathbf{X} - \gamma\,\mathbf{u}(\mathbf{u}^\top\mathbf{X}) \tag{14}$$

   Thus for each column $\mathbf{x}_j$ of $\mathbf{X}$, first form $\gamma\,\mathbf{u}^\top\mathbf{x}_j$, then subtract this multiple of $\mathbf{u}$ from $\mathbf{x}_j$.

2. For calculation of row k of the array resulting from pre-multiplication by $\mathbf{H}_k$, formulae are available that are simplified and numerically more accurate than from direct use of equation 14. A further simplification is to form the diagonal element as $|\alpha_k|$, multiplying remaining elements in the ith row by $\text{sgn}(\alpha_k)$. In the examples shown below, sequences of such modified Householder reflections have been used. We no longer have Householder reflections, but the $\mathbf{H}_i$ are, just as before, orthonormal matrices. The difference is that when α_k is negative, all elements in row k change sign relative to the Householder matrix $\mathbf{H}_k$.

3. If the calculated diagonal element is zero to within machine precision, the easiest recourse is to set all elements in the row to zero. The vector $\mathbf{b}$ is no longer uniquely defined. The residual sum of squares is unique, independent of the action that may be taken to resolve the indeterminacy in the parameters.

4. Let $\mathbf{X}_k$ be the matrix that consists of the first k columns of $\mathbf{X}$. Then the information needed to minimize
$$\|\mathbf{y} - \mathbf{X}_k\mathbf{b}\|^2$$
is available once the first k Householder steps are complete. It is found in the $k \times k$ submatrix of $\mathbf{R}$ and in the first k elements of $\mathbf{Q}^\top\mathbf{y}$.

2.4. Demonstrations of the computational steps

Example 1

We will start with
$$\mathbf{X} = \begin{bmatrix} 1 & -6 & 0 \\ 1 & -3 & 6 \\ 1 & 6 & 15 \\ 1 & 21 & 9 \end{bmatrix} \qquad \mathbf{y} = \begin{bmatrix} -3 \\ 1 \\ 2 \\ 6 \end{bmatrix}$$
The aim is to minimize
$$[-3-(b_0-6b_1)]^2+[1-(b_0-3b_1+6b_2))]^2+[2-(b_0+6b_1+15b_2)]^2+[6-(b_0+21b_1+9b_2)]^2$$

The sequence of sweeps is
$$\begin{bmatrix} 1 & -6 & 0 & -3 \\ 1 & -3 & 6 & 1 \\ 1 & 6 & 15 & 2 \\ 1 & 21 & 9 & 6 \end{bmatrix} \rightarrow \begin{bmatrix} \mathbf{2} & \mathbf{9} & \mathbf{15} & \mathbf{3} \\ 1 & -4 & 1 & 1 \\ 1 & 5 & 10 & 2 \\ 1 & 20 & 4 & 6 \end{bmatrix} \rightarrow \begin{bmatrix} \mathbf{2} & \mathbf{9} & \mathbf{15} & \mathbf{3} \\ 1 & \mathbf{21} & \mathbf{6} & \mathbf{6} \\ 1 & 5 & 9 & 1 \\ 1 & 20 & 0 & 2 \end{bmatrix}$$

The first sweep is equivalent to pre-multiplication by $\mathbf{H}_1$, and the second to to pre-multiplication by $\mathbf{H}_2$. At this point, the element in the (4,3) position is zero, and use of $\mathbf{H}_3$ is not needed. ($\mathbf{H}_3$, if it were formed, would be the identity matrix.)

Notice that, rather than recording below diagonal elements in the ith column as zero once calculations for row i are complete, the values that were there previously have been left in place, albeit printed in small type. It is straightforward to ensure that any computations with the upper triangular matrix treat those elements as zeros. The values that are left in place can be used, when and if required, to reconstruct the sequence of Householder steps.

Use of R

```
X1df <- data.frame(X1=c(-6,-3,6,21), X2=c(0,6,15,9), y=c(-3,1,2,6))
lm(y ~ X1+X2, data=X1df)
summary(lm(y ~ X1+X2, data=X1df))  # Gives more information
anova(lm(y ~ X1+X2, data=X1df))    # Gives different information
lm(y ~ X1+X2, data=X1df)$qr        # qr decomposition
```

Observe that R stores somewhat different information in below diagonal positions.

Alternatively, R has a suite of functions that can be used for the underlying computations of least squares calculations. These include `qr()`, `qr.coef()` and `qr.solve()`. Because they work with matrices rather than data frames, they may be much faster than `lm()` for the handling of large least squares problems. Approaches to the calculations that use these will be demonstrated for the second slightly more substantial example that will now be presented.

Example 2

Table 1 shows the application of a sequence of 4 modified Householder reflections to a 9 × 5 array. The following shows the **X** matrix and column of observations **y** from which the calculations start, with the upper triangular system that is to be solved on the right:

$$(\mathbf{X}, y) = \left[\begin{array}{cccc|c} \text{X0} & \text{X1} & \text{X2} & \text{X3} & \text{y} \\ 1 & 1 & -1 & -14 & -7.7 \\ 1 & -1 & 6 & -2 & 8.5 \\ 1 & 1 & 9 & 8 & 19.6 \\ 1 & -2 & 8 & -3 & 11.4 \\ 1 & 0 & 5 & 1 & 11.4 \\ 1 & 0 & 3 & -1 & 6.8 \\ 1 & 4 & 2 & 9 & 2.5 \\ 1 & 7 & 0 & 8 & -1.6 \\ 1 & 8 & -5 & 3 & -13 \end{array}\right]$$

The R-matrix and the vector **f** for the equation **Rb** = **f** can be extracted from the first four rows of the final step of the reduction in Table 1. The least squares estimates are thus obtained by solving:

$$\left[\begin{array}{cccc} \mathbf{3} & \mathbf{6} & \mathbf{9} & \mathbf{3} \\ 0 & \mathbf{10} & \mathbf{-10} & \mathbf{10} \\ 0 & 0 & \mathbf{8} & \mathbf{16} \\ 0 & 0 & 0 & \mathbf{8} \end{array}\right] \mathbf{b} \quad = \quad \left[\begin{array}{c} \mathbf{12.63} \\ \mathbf{-20.04} \\ \mathbf{19.95} \\ \mathbf{4} \end{array}\right]$$

The numbers in the X-matrix were chosen so that the values in the R-matrix are integers. The least squares system is for a model in which there is a constant term X0 and explanatory variables X1, X2 and X3.

Observe the subsystems for which information is available

$$\left[\begin{array}{c|c} \mathbf{3} & \mathbf{12.63} \leftarrow \end{array}\right]$$

Solve $3b_0 = 12.63 \quad \Rightarrow \quad b_0 = 4.21$.
SS reduces by 12.63^2 (to CSS[a])

[a] SS = sum of squares; RSS = residual SS; CSS = corrected SS, i.e., RSS about mean

$$\left[\begin{array}{cc|c} \mathbf{3} & \mathbf{6} & \mathbf{12.63} \\ 0 & \mathbf{10} & \mathbf{-20.04} \leftarrow \end{array}\right]$$

$10b_1 = -20.04 \quad \Rightarrow \quad b_1 = -2.004$;
$3b_0 + 6b_1 = 12.63 \quad \Rightarrow \quad b_0 = 8.22$.
Additional reduction in RSS = 20.04^2.

$$\left[\begin{array}{ccc|c} \mathbf{3} & \mathbf{6} & \mathbf{9} & \mathbf{12.63} \\ 0 & \mathbf{10} & \mathbf{-10} & \mathbf{-20.04} \\ 0 & 0 & \mathbf{8} & \mathbf{19.95} \leftarrow \end{array}\right]$$

$8b_2 = 19.95$; $10b_1 - 10b_2 = -20.04$; $3b_0 + 6b_1 + 9b_2 = 12.63$.
Additional reduction in RSS = 19.95^2.

In the full model, the RSS reduces by a further 4^2. The residual sum of squares from the full model is

$$1.46^2 + 0.68^2 + 4.26^2 + 1.11^2 + 1.58^2 = 24.46$$

(Note that 24.46 is the result of rounding the RSS when it is calculated to full machine accuracy.)

Use of R

```
X2df <- data.frame(X1=c(1, -1, 1, -2, 0, 0, 4, 7, 8),
           X2=c(-1, 6, 9, 8, 5, 3, 2, 0, -5),
           X3=c(-14, -2, 8, -3, 1, -1, 9, 8, 3),
           y=c(-7.7, 8.5, 19.6, 11.4, 11.4, 6.8, 2.5, -1.6, -13))
lm(y ~ X1+X2+X3, data=X2df)
summary(lm(y ~ X1+X2+X3, data=X2df))  # Gives more information
anova(lm(y ~ X1+X2+X3, data=X2df))    # Gives different information
lm(y ~ X1+X2+X3, data=X2df)$qr        # qr decomposition
```

Alternatives to `lm()`

For computationally intensive least squares calculations, consider the use of `qr()` and associated functions. For use of this approach, the user must first extract or construct the design matrix. The following demonstrates the use of `qr()` and allied functions:

```
X <- with(X2df, model.matrix(~ X1+X2+X3))
## X <- cbind(rep(1,9), X2df[, 1:3])  # More efficient alternative
QR <- qr(X)
qr.coef(QR, X2df$y)
qr.resid(QR, X2df$y)
## etc, etc
```

Calculations may be computationally intensive because a major component of the calculation is repeated a large number of times, and/or because they involve one or more large design matrices.

2.4.1. Use of the Cholesky Decomposition to solve the normal equations

If $n >> p$ (substantially more rows than columns), then formation of $\mathbf{X}^\top\mathbf{X}$, followed by solution of the normal equations, will be faster than use of the `QR` decomposition. Having formed $\mathbf{X}^\top\mathbf{X}$, the Cholesky decomposition, implemented in the R function `chol()`, then yields an upper triangular matrix $\mathbf{R}$ such that:

$$\mathbf{R}^\top\mathbf{R} = \mathbf{X}^\top\mathbf{X}$$

The matrix $\mathbf{R}$ is, to within numerical error, the same as the matrix $\mathbf{R}$ that results from a version of `QR` in which all diagonal elements are positive (or, if $\mathbf{X}^\top\mathbf{X}$ is singular, 0).[1]

Having formed $\mathbf{R}$, two further steps are required to give the least squares estimates of the coefficients:

- Solve $\mathbf{R}^\top\mathbf{f} = \mathbf{X}^\top\mathbf{y}$ for $\mathbf{f}$.

[1] In practice, pivoting, i.e., rearrangement of the order of columns and rows of $\mathbf{X}^\top\mathbf{X}$, or columns of $\mathbf{X}$, may be used to ensure that highly dependent columns are taken last. The same pivoting scheme will then lead to the same matrix $\mathbf{R}$.

$$
\begin{bmatrix}
1 & 1 & -1 & -14 \\
1 & -1 & 6 & -2 \\
1 & 1 & 9 & 8 \\
1 & -2 & 8 & -3 \\
1 & 0 & 5 & 1 \\
1 & 0 & 3 & -1 \\
1 & 4 & 2 & 9 \\
1 & 7 & 0 & 8 \\
1 & 8 & -5 & 3
\end{bmatrix}
\begin{matrix} 7.7 \\ 8.5 \\ 19.6 \\ 11.4 \\ 11.4 \\ 6.8 \\ 2.5 \\ -1.6 \\ -13.0 \end{matrix}
\rightarrow
\begin{bmatrix}
\mathbf{3} & \mathbf{6} & \mathbf{9} & \mathbf{3} \\
1 & -2.75 & 4 & 0.75 \\
1 & -0.75 & 7 & 10.75 \\
1 & -3.75 & 6 & -0.25 \\
1 & -1.75 & 3 & 3.75 \\
1 & -1.75 & 1 & 1.75 \\
1 & 2.25 & 0 & 11.75 \\
1 & 5.25 & -2 & 10.75 \\
1 & 6.25 & -7 & 5.75
\end{bmatrix}
\begin{matrix} \mathbf{12.63} \\ 7.27 \\ 18.37 \\ 10.17 \\ 10.17 \\ 5.57 \\ 1.27 \\ -2.83 \\ -14.23 \end{matrix}
$$

$$
\rightarrow
\begin{bmatrix}
\mathbf{3} & \mathbf{6} & \mathbf{9} & \mathbf{3} \\
1 & \mathbf{10} & \mathbf{-10} & \mathbf{10} \\
1 & -0.75 & 6.18 & 11.29 \\
1 & -3.75 & 1.88 & 2.47 \\
1 & -1.75 & 1.08 & 5.02 \\
1 & -1.75 & -0.92 & 3.02 \\
1 & 2.25 & 2.47 & 10.12 \\
1 & 5.25 & 3.76 & 6.94 \\
1 & 6.25 & -0.14 & 1.22
\end{bmatrix}
\begin{matrix} \mathbf{12.63} \\ \mathbf{-20.04} \\ 16.76 \\ 2.14 \\ 6.42 \\ 1.82 \\ 6.09 \\ 8.41 \\ -0.85 \end{matrix}
\rightarrow
\begin{bmatrix}
\mathbf{3} & \mathbf{6} & \mathbf{9} & \mathbf{3} \\
1 & \mathbf{10} & \mathbf{-10} & \mathbf{10} \\
1 & -0.75 & \mathbf{8} & \mathbf{16} \\
1 & -3.75 & 1.88 & -1.15 \\
1 & -1.75 & 1.08 & 2.94 \\
1 & -1.75 & -0.92 & 4.79 \\
1 & 2.25 & 2.47 & 5.36 \\
1 & 5.25 & 3.76 & -0.31 \\
1 & 6.25 & -0.14 & 1.48
\end{bmatrix}
\begin{matrix} \mathbf{12.63} \\ \mathbf{-20.04} \\ \mathbf{19.95} \\ -2.74 \\ 3.63 \\ 4.21 \\ -0.31 \\ -1.34 \\ -0.49 \end{matrix}
$$

$$
\rightarrow
\begin{bmatrix}
\mathbf{3} & \mathbf{6} & \mathbf{9} & \mathbf{3} \\
1 & \mathbf{10} & \mathbf{-10} & \mathbf{10} \\
1 & -0.75 & \mathbf{8} & \mathbf{16} \\
1 & -3.75 & 1.88 & \mathbf{8} \\
1 & -1.75 & 1.08 & 2.94 \\
1 & -1.75 & -0.92 & 4.79 \\
1 & 2.25 & 2.47 & 5.36 \\
1 & 5.25 & 3.76 & -0.31 \\
1 & 6.25 & -0.14 & 1.48
\end{bmatrix}
\begin{matrix} \mathbf{12.63} \\ \mathbf{-20.04} \\ \mathbf{19.95} \\ \mathbf{4} \\ 1.46 \\ 0.68 \\ -4.26 \\ -1.11 \\ -1.58 \end{matrix}
$$

Table 1: Reduction of a 9×4 array to upper triangular form, with the same operations applied to the fifth column $\mathbf{y}$. Four modified Householder reflections are applied to the 9×5 array $(\mathbf{X}, \mathbf{y})$.

- Solve $\mathbf{R}\,\mathbf{b} = \mathbf{f}$ for $\mathbf{b}$.

A slightly more efficient way to organize the calculations will be noted below.

Timings

Here is a comparison of times:

```
> X <- matrix(rnorm(50000000), ncol=200)
> ## QR decomposition
> system.time(z <- qr(X))
   user  system elapsed
 28.459   0.512  29.036
> ## Cross-product, followed by Cholesky
> system.time(S <- crossprod(X))
   user  system elapsed
  9.289   0.007   9.294
> system.time(R <- chol(S))
   user  system elapsed
  0.003   0.000   0.003
> ## Totals for forming X'X, then applying Cholesky:
> ## 9.292  0.007  9.297  (cf, for qr: 28.459   0.512  29.036)
```

The comparison is not totally fair, because the QR decomposition preserves information that has to be recovered by other means when calculation of the cross-product is followed by the Cholesky decomposition. (These timings were on a 2.8GHz mid-2009 15" Macbook Pro with 4 GB memory. They vary somewhat from one execution of the code to another.)

Accuracy

If any off-diagonal elements of $\mathbf{X}^\top\mathbf{X}$ are a large fraction of the square root of the product of the diagonal elements for that row and column, there can be a serious loss of accuracy relative to QR. This will happen if one or more columns of $\mathbf{X}$ are close to a linear combination of earlier columns.

A first step is to centre columns of $\mathbf{X}$ (apart from any initial column of 1s), and to centre $\mathbf{y}$, by subtracting off means or approximate means. If remaining dependencies are on the first several columns of $\mathbf{X}$, or if columns can be rearranged so that this is the case, then a hybrid method is possible, i.e., first use Householder to form the first few rows of $\mathbf{R}$. If k rows have been formed, then drop rows and columns 1 to k from the matrix $\mathbf{X}_{(k)}$ that remains, and form $\mathbf{X}_{(k)}^\top\mathbf{X}_{(k)}$. The Cholesky decomposition of $\mathbf{X}_{(k)}^\top\mathbf{X}_{(k)}$ is then the remaining submatrix of $\mathbf{R}$.

Sparse matrix calculations

Calculation can be speeded up further by the use of sparse matrix methods.

2.5. The Analysis of Variance Table

We show the sequential analysis of variance table that is relevant to Example 2 in Subsection 2.4. From Table 1, we see that

$$\mathbf{Q}^\top \mathbf{y} = \begin{bmatrix} \mathbf{12.63} \\ \mathbf{-20.04} \\ \mathbf{19.95} \\ \mathbf{4} \\ 1.46 \\ 0.68 \\ -4.26 \\ -1.11 \\ -1.58 \end{bmatrix}$$

The usual form of sequential analysis of variance table, which partitions the sum of squares about the mean, is obtained by picking off the elements of $\mathbf{Q}^\top \mathbf{y}$ in turn:

Term	Sum of squares	DF
X1	20.04^2	1
X2	19.95^2	1
X3	4^2	1
Residual	24.46	$n - p$

Compare this with

```
X2df.lm <- lm(y ~ X1+X2+X3, data=X2df)
anova(X2df.lm)
```

Now observe how this table can be obtained from the output of `qr()` and friends.

```
## Now extract the table from the successive components of Qy
QR <- qr(cbind(rep(1,9), as.matrix(X2df[,1:3])))
Qty <- qr.qty(QR, X2df$y)
duetoSS <- Qty[2:4]^2   # Why is Qty[1] omitted?
rss <- sum(Qty[-(1:4)]^2)
aovtab <- data.frame(row.names=c("X1", "X2", "X3", "Residual"),
  Df=c(rep(1,3), length(Qty)-4), SS=c(duetoSS, rss))
aovtab
```

It is important to note that this is a sequential analysis of variance table. In general the contribution of each term depends on which (if any) term(s) precede it in the model.

2.6. Leave-one-out residuals

What is the residual, for the ith data observation, when the least squares estimate is calculated using remaining data points. These can be obtained without repeating the regression calculations.

For this, replace Equation 1.1, i.e.

$$\mathbf{y} = \mathbf{X}\boldsymbol{\beta} + \boldsymbol{\epsilon},$$

by

$$\mathbf{y} = \begin{bmatrix}\mathbf{X} & \boldsymbol{\delta}\end{bmatrix}\begin{bmatrix}\tilde{\boldsymbol{\beta}} \\ \zeta\end{bmatrix} + \tilde{\boldsymbol{\epsilon}} \tag{15}$$

where $\boldsymbol{\delta}$ has 1 in the jth row and is zero elsewhere. This has the effect of setting the residual associated with the jth observation to zero, while leaving other residuals unaffected. Here is how it works. The contribution of the jth observation to the residual sum of squares is:

$$(y_j - \beta_1 x_{j1} - \beta_2 x_{j2} \ldots - \beta_p x_{jp} - \zeta)^2$$

Here, for simplicity, the tildes (˜) have been left off.

Then the least squares estimate ζ is the value that reduces this contribution to zero. As this happens irrespective of the estimate of $\boldsymbol{\beta}$, the jth data point has no say in determining $\tilde{\boldsymbol{\beta}}$, and the estimate of ζ is the leave-one-out residual that we require.

Earlier discussion (refer back to Equation 5) derived an orthonormal matrix $\mathbf{Q}$ such that:

$$\mathbf{Q}^\top \mathbf{X} = \begin{bmatrix}\mathbf{R} \\ \mathbf{0}\end{bmatrix} \tag{16}$$

Write

$$\mathbf{Q}^\top = \begin{bmatrix}\mathbf{Q}_f^\top \\ \mathbf{Q}_r^\top\end{bmatrix} \text{ where } \mathbf{Q}_f^\top \text{ is } \quad p \times p \quad \text{and } \mathbf{Q}_r^\top \text{ is } n - p \times p$$

Premultiplication of the augmented matrix $\begin{bmatrix}\mathbf{X} & \boldsymbol{\delta}\end{bmatrix}$ from Equation 15 by $\mathbf{Q}^\top$ yields

$$\mathbf{Q}^\top \begin{bmatrix}\mathbf{X} & \boldsymbol{\delta}\end{bmatrix} = \begin{bmatrix}\mathbf{Q}_f^\top \\ \mathbf{Q}_r^\top\end{bmatrix}\begin{bmatrix}\mathbf{X} & \boldsymbol{\delta}\end{bmatrix} \tag{17}$$

$$= \begin{bmatrix}\mathbf{R} & \mathbf{q}_{j(f)} \\ \mathbf{0} & \mathbf{q}_{j(r)}\end{bmatrix} \tag{18}$$

Here, $\mathbf{q}_{j(f)}$ is the jth column of $\mathbf{Q}_f^\top$, while $\mathbf{q}_{j(r)}$ is the jth column of $\mathbf{Q}_r^\top$.

We can reduce elements after the $(p+1)^{th}$ in the final column to zero by premultiplication by the matrix that results from replacing the final $n - p$ row and column positions of an identity matrix by the Householder matrix:

$$\mathbf{H} = \mathbf{I} - \mathbf{q}_{j(r)}\mathbf{q}_{j(r)}^\top/\gamma$$

This leaves $r_{p+1,p+1} = \alpha = \|\mathbf{q}_{j(r)}\|$ (a possible change of sign is unimportant for present purposes), with elements after the $(p+1)^{th}$ in the final column reduced to zero.

Now apply the same premultiplication by $\mathbf{H}$ to

$$\begin{bmatrix}\mathbf{f} \\ \mathbf{r}\end{bmatrix} = \begin{bmatrix}\mathbf{Q}_f^\top \mathbf{y} \\ \mathbf{Q}_r^\top \mathbf{y}\end{bmatrix}$$

Elements 1 to p are unchanged, and the element in the $(p+1)^{th}$ position becomes

$$\frac{\mathbf{q}_{j(r)}^\top \mathbf{r}}{\alpha} = \frac{\mathbf{q}_{j(r)}^\top \mathbf{Q}_r^\top \mathbf{y}}{\alpha}$$

This is the element in the (j, j) position of

$$\frac{\mathbf{Q}_r \mathbf{Q}_r^\top \mathbf{y}}{\alpha} = \frac{\mathbf{I} - \mathbf{Q}_f \mathbf{Q}_f^\top \mathbf{y}}{\alpha}$$

i.e., it equals e_j/α, where e_j is the residual from the model that includes the j^{th} row. In place of $\mathbf{R}\boldsymbol{\beta} = \mathbf{f}$, the regression parameters are obtained by solving

$$\begin{bmatrix} \mathbf{R} & \mathbf{q}_{j(f)} \\ 0 \dots 0 & \alpha \end{bmatrix} \begin{bmatrix} \tilde{\boldsymbol{\beta}} \\ \zeta \end{bmatrix} = \begin{bmatrix} \mathbf{f} \\ \frac{e_j}{\alpha} \end{bmatrix}$$

Hence

$$e_{(j)} = \frac{e_j}{\alpha^2} \tag{19}$$

$$= \frac{e_j}{\|\mathbf{q}_{j(r)}\|^2} \tag{20}$$

$$= \frac{e_j}{1 - a_{jj}} \tag{21}$$

where a_{jj} is the j^{th} diagonal element of the matrix

$$\mathbf{A} = \mathbf{Q}_f \mathbf{Q}_f^\top = \mathbf{I} - \mathbf{Q}_r \mathbf{Q}_r^\top \tag{22}$$

Notice that we have not made any assumption of non-singularity.

2.6.1. The OCV mean square error

It follows from Equation 21 that the *ordinary cross-validation* (OCV) mean square error statistic:

$$OCV = \frac{1}{n} \sum_{j=1}^{n} e_{(j)}^2 = \frac{1}{n} \sum_{j=1}^{n} \frac{e_j^2}{(1 - a_{jj})^2} \tag{23}$$

This is a weighted sum of the squares of the ordinary residuals. High leverage points (large $1 - a_{ii}$ get the greatest weight.

In Subsection 2.6.2 that now follows, it will be argued that the *generalized cross-validation* (GCV) mean square prediction error estimate is preferable.

2.6.2. The GCV statistic

This continues the discussion above. The GCV statistic is based on an orthonormal rotation of the row space of $\mathbf{X}$, where the rotation is chosen to yield pseudo-observations that all have equal leverage.

If $\mathbf{P}$ is any orthonormal matrix

$$\begin{aligned} \|\mathbf{P}\mathbf{y} - \mathbf{P}\mathbf{X}\boldsymbol{\beta}\|^2 &= (\mathbf{y} - \mathbf{X}\boldsymbol{\beta})^\top P^\top P(\mathbf{y} - \mathbf{X}\boldsymbol{\beta}) \\ &= (\mathbf{y} - \mathbf{X}\boldsymbol{\beta})^\top (\mathbf{y} - \mathbf{X}\boldsymbol{\beta}) \\ &= \|\mathbf{y} - \mathbf{X}\boldsymbol{\beta}\|^2 \end{aligned}$$

Thus minimization of $\|\mathbf{y} - \mathbf{X}\boldsymbol{\beta}\|^2$ is equivalent to minimization of $\|\mathbf{Py} - \mathbf{PX}\boldsymbol{\beta}\|^2$. The effect is to replace columns of $\mathbf{X}$ by columns of $\mathbf{PX}$, and elemement of $\mathbf{y}$ by elements of $\mathbf{Py}$. Where the ordinary least squares residuals were $\boldsymbol{\epsilon}$ they are now $\mathbf{P}\boldsymbol{\epsilon}$. We have created a set of pseudo-observations and pseudo-residuals, iid normal if the elements of $\boldsymbol{\epsilon}$ are iid normal. If the observations are replaced by these pseudo-observations, the OCV statistic changes.

The GCV statistic is the OCV statistic that results from choosing $\mathbf{P}$ so that all pseudo-observations have equal leverage. It is not necessary to find such a $\mathbf{P}$ in order to calculate the GCV statistic; all that is necessary is to show that such a choice is possible.[2]

Now suppose that, as before, premultiplication by $\mathbf{Q}_f^\top$ reduces $\mathbf{X}$ to upper triangular form. Then

$$\mathbf{Q}_f^\top \mathbf{P}^\top \mathbf{PX} = \mathbf{Q}_f^\top \mathbf{P}^\top (\mathbf{PX})$$

i.e., premultiplication by $\mathbf{Q}_f^\top \mathbf{P}^\top$ reduces $\mathbf{PX}$ to upper triangular form. The leverages for these pseudo-observations are diagonal elements of

$$\mathbf{PQ}_r\mathbf{Q}_r^\top\mathbf{P}^\top = \mathbf{I}_n - \mathbf{PQ}_f\mathbf{Q}_f^\top\mathbf{P}^\top$$

Moreover the sum of the leverages is

$$\begin{aligned}
\operatorname{trace}(\mathbf{PQ}_r\mathbf{Q}_r^\top\mathbf{P}^\top) &= \operatorname{trace}(\mathbf{Q}_r\mathbf{Q}_r^\top\mathbf{P}^\top\mathbf{P}) \\
&= \operatorname{trace}(\mathbf{Q}_r\mathbf{Q}_r^\top) \\
&= \operatorname{trace}(\mathbf{I} - \mathbf{Q}_f\mathbf{Q}_f^\top) \\
&= n - \sum_{i=1}^{n} a_{ii}
\end{aligned}$$

The sum of the leverages is $n - \sum_{i=1}^n a_{ii}$, just as for the unrotated configuration. In the equal leverages configuration, the leverages are thus all equal to

$$\frac{1}{n}\left(n - \sum_{i=1}^{n} a_{ii}\right)$$

The sum of squares of the estimated pseudo-residuals is unchanged. Ordinary cross-validation gives, for these pseudo-observations, the *generalized cross-validation* (GCV) estimate

$$\frac{n\sum_{i=1}^n e_i^2}{(n - \sum_{i=1}^n a_{ii})^2}$$

Not only does this have better theoretical properties than OCV; it is simpler computationally. It is, as for the OLS estimate, derived from the residual sum of squares.

The divisor $n - \sum_{i=1}^n a_{ii}$ has the same role as the degrees of freedom in the classical theory.

[2] One way to show this is to note that if two rows have different L_2 norms, then we can find a Givens rotation that equalizes the lengths. This process can be repeated until all rows are, to within numerical error, of equal length.

2.7. Weighted Least Squares

If var[$\mathbf{y}$] = $\mathbf{V}$ is known to within the scale factor σ^2, then a linear transformation of the vector $\mathbf{y}$ can be determined such that elements of the linearly transformed vector are iid.

A suitable linear transformation is conveniently obtained from the Cholesky decomposition of $\mathbf{V}$, by which

$$\mathbf{V} = \mathbf{L}\mathbf{L}^\top$$

Then

$$\begin{aligned} \text{var}[\mathbf{L}^{-1}\mathbf{y}] &= \mathbf{L}^{-1}\ \text{var}[\mathbf{y}]\ \mathbf{L}^\top - 1 \\ &= \mathbf{L}^{-1}\mathbf{V}\mathbf{L}^\top - 1 \\ &= \mathbf{I}_n\sigma^2 \end{aligned}$$

Then we minimize

$$\|\mathbf{L}^{-1}(\mathbf{y} - \mathbf{X}\boldsymbol{\beta})\|^2$$

For this, replace $\mathbf{y}$ by $\mathbf{y}^* = \mathbf{L}^{-1}\mathbf{y}$, and $\mathbf{X}$ by $\mathbf{X}^* = \mathbf{L}^{-1}\mathbf{X}$, and proceed as before.

The normal equations become

$$\begin{aligned} \mathbf{X}^\top\mathbf{W}\mathbf{X}\mathbf{b} &= \mathbf{X}^\top\mathbf{W}\mathbf{y} \\ \text{i.e., } \mathbf{X}^{*\top}\mathbf{X}^*\mathbf{b} &= \mathbf{X}^{*\top}\mathbf{y}^* \end{aligned}$$

where $\mathbf{W} = \mathbf{V}^{-1}$

Then setting $\mathbf{X}\mathbf{b} = \widehat{\boldsymbol{\mu}}$ (the fitted values), this becomes

$$\mathbf{X}^\top\mathbf{W}\widehat{\boldsymbol{\mu}} = \mathbf{X}^\top\mathbf{W}\mathbf{y}$$

This is the form of the equation that carries across to Generalized Linear Models.

2.8. Unbalance – implications for addition or deletion of model terms

In data with suitable balance coefficients do not change when terms are added to, or dropped from, the model. This is reflected in the **R**-matrix. It is insightful to observe how the **R**-matrix (and regression coefficients) change when balanced data are modified, by use of unequal weights or by dropping observations.

```
> pain <- data.frame(vasScore=c(0.67, 0.05, 3.67, 3.13),
+                    trt=factor(rep(c("placebo", "baclofen"), 2),
+                               levels=c("placebo", "baclofen")),
+                    gender=factor(rep(c("male", "female"), c(2,2)),
+                                  levels=c("male", "female")),
+                    number=c(3, 9, 15, 7))
> pain
   vasScore      trt gender number
1      0.67  placebo   male      9
2      0.05 baclofen   male      3
3      3.67  placebo female      7
4      3.13 baclofen female     15
```

```
> pain.lm <- lm(vasScore ~ gender + trt, data=pain)
> round(coef(pain.lm), 2)
 (Intercept) genderfemale  trtbaclofen
        0.65         3.04        -0.58
```

Now omit consideration of the Gender effect:

```
> pain1.lm <- lm(vasScore ~ trt, data=pain)
> round(coef(pain1.lm), 2)
(Intercept) trtbaclofen
       2.17       -0.58
```

Observe that the estimate of the treatment effect is unchanged.

Weighted analysis

```
> pain.wlm <- lm(vasScore ~ gender + trt, weight=number, data=pain)
> round(coef(pain.wlm), 2)
 (Intercept) genderfemale  trtbaclofen
        0.66         3.03        -0.57
```

Observe that the estimate of the treatment effect has hardly changed. In general, the change may not be so small, but will not reverse an effect that goes in the same direction for the genders separarately.

Now omit consideration of the gender effect:

```
> pain.wlm1 <- lm(vasScore ~ trt, weight=number, data=pain)
> round(coef(pain.wlm1), 2)
(Intercept) trtbaclofen
       1.98        0.63
```

Observe that the treatment effect now goes in the other direction. The combination of unequal weights and omission of a relevant factor generates this misleading result.

Implications for the R-matrix

First, form the matrix on which the calculations will be performed.

```
> Xy <- cbind(model.matrix(~ gender + trt, data=pain),
              vasScore = pain$vasScore)
> Xy
  (Intercept) genderfemale trtbaclofen vasScore
1           1            0           0     0.67
2           1            0           1     0.05
3           1            1           0     3.67
4           1            1           1     3.13
```

Now use my function `house()` for the calculation. This modifies the Householder reflections so that diagonal elements are positive. Relative to the result from use of Householder reflections all elements in rows with a negative diagonal element are multiplied by -1.

```
> house(Xy, showzeros=3)
  (Intercept) genderfemale trtbaclofen vasScore
1           2            1           1     3.76
2           0            1           0     3.04
3           0            0           1    -0.58
4           0            0           0     0.04
```

To obtain estimates of the model parameters, solve:

$$\begin{bmatrix} 2 & 1 & 1 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \mathbf{b} = \begin{bmatrix} 3.76 \\ 3.04 \\ -0.58 \end{bmatrix}$$

Compare this with the calculations for the weighted analysis:

```
> house(Xy*sqrt(pain$number), showzeros=3)
  (Intercept) genderfemale trtbaclofen    vasScore
1    5.830952     3.772969    2.743977 13.62041763
2    0.000000     2.786522   -1.203271  9.17652419
3    0.000000     0.000000    2.650043 -1.49894659
4    0.000000     0.000000    0.000000  0.09892627
```

Now solve (values are rounded to three decimal places):

$$\begin{bmatrix} 5.831 & 3.773 & 2.744 \\ 0 & 2.787 & -1.203 \\ 0 & 0 & 2.650 \end{bmatrix} \mathbf{b} = \begin{bmatrix} 13.620 \\ 9.177 \\ -1.499 \end{bmatrix}$$

A key difference is that the R matrix no longer has a zero in the (2, 3) position. As a consequence, the estimate of the treatment effect changes (and changes in sign) if there is no allowance for Innings.

Further examples

An interesting experiment is to observe how dropping a few rows from a balanced design affects the R matrix and hence the parameter estimates. For example, try the following:

```
house(model.matrix(lm(headwt ~ Cult + Date, data=cabbages))
xtabs(~ Cult + Date, data=cabbages[11:14, ])
house(model.matrix(lm(HeadWt ~ Cult + Date,
      data=cabbages[-(11:14),])), showzeros=4)[1:4,]
  ## The (2,3) and (2,4) positions are not now occupied by zeros.
## Drop out row 11
house(model.matrix(lm(HeadWt ~ Cult + Date,
      data=cabbages[-(11),])), showzeros=4)[1:4,]
```

3. Model Assumptions and Model Choice

This section will discuss the the classical theory, and note implications for model choice. Finally, recourses will be noted that can be used when the classical theory fails or is in doubt.

The classical theory that will now be described assumes that conditional on $\mathbf{X}$

$$\mathbf{y} \sim N(\mathbf{X}\boldsymbol{\beta}, \mathbf{I}_n\sigma^2) \tag{24}$$

3.1. Limitations of the classical theory

According to the strict requirements of the theory, the model must be known in advance. Neither data based choice of transformation(s) nor variable selection are permitted. It is however permissible to divide the data set into two parts – a training set that is used to develop the model, and a test set that is used to derive the eventual model fit. Or three parts may be required – a training set, a holdout set on which which performance is optimized in order to tune the model, and a test set that is used for the eventual model fit. An objection is that this makes poor use of the data, which may be a serious issue for smallish data sets. Matters can be improved somewhat by repeating the process, with the roles of training, test and any holdout set interchanged.

With respect to Equation 24, note that:

- Depending on how the fitted model will be used, the normality assumption is commonly not of critical importance; approximate normality is enough. Independence is more crucial, and less open to checking. If the form of departure from independence is known or can be guessed, the dependence can be modelled. This leads into areas (time series modeling, multi-level models, repeated measures, etc.) whose details are beyond the scope of the present course.

- In practice limited use of plots or other mechanisms to determine appropriate transformations may be essential, to get a model fit that does not do violence to the data. Depending on the details of the specific model and associated data, this may not not seriously invalidate assumptions. Limited variable selection, e.g., choose two explanatory variables out of four, may be similarly acceptable. Again, much will depend on the details of the model and associated data. Selection effects are in general a more serious problem than effects from limited use of data-based transformations.

- Where one model emerges as very substantially superior to other models considered, selection effects are not an issue. Why?

When the model is fitted to the data used to select the model from a set of possible models, the effect is anti-conservative. Thus, standard errors will be smaller than indicated by the theory, and coefficients and t-statistics larger. Such anti-conservative estimates of standard errors and other statistics may, unless the bias is huge, nevertheless provide the useful guidance.

Almost inevitably, none of the models on offer will be strictly correct. Mis-specification of the fixed effects, and to a lesser extent of the random effects, is likely to bias model estimates, at the same time inflating the error variance or variances, i.e., it may to some extent work in the opposite direction to selection effects.

Recourses that are available when the classical theory fails will be discussed below.

3.1.1. Distributional Results – the classical theory

Observe that

$$\text{var}[\mathbf{Qy}] = \mathbf{Q}^\top \mathbf{I}_n \mathbf{Q}\sigma^2 = \mathbf{I}_n\sigma^2$$

Then

$$\text{E}\begin{bmatrix}\mathbf{f}\\ \mathbf{r}\end{bmatrix} = \text{E}[\mathbf{Q}^\top\mathbf{y}] = \mathbf{Q}^\top\mathbf{X}\boldsymbol{\beta} = \begin{bmatrix}\mathbf{R}\\ \mathbf{0}\end{bmatrix}\boldsymbol{\beta}$$

Thus,

$$\mathbf{f} \sim N(\mathbf{R}\boldsymbol{\beta}, \mathbf{I}_p\sigma^2),\ \mathbf{r} \sim N(\mathbf{0}, \mathbf{I}_{n-p}\sigma^2)$$

Moreover $\|\mathbf{f}\|^2$ and $\|\mathbf{r}\|^2$ are each sums of independent $\sigma^2\chi^2_1$ terms and therefore independent.

Finally $\mathbf{b} = \mathbf{R}^{-1}\mathbf{f}$ is distributed as

$$N(\boldsymbol{\beta}, \mathbf{R}^{-1}\mathbf{R}^{\top -1}\sigma^2), \quad \text{i.e., as } N(\boldsymbol{\beta}, (\mathbf{X}^\top\mathbf{X})^{-1}\sigma^2)$$

As we will use an estimate of σ^2 that has $n - p$ degrees of freedom, individual elements of $\mathbf{b}$ are distributed as t_{n-p}. Elements b_i of $\mathbf{b}$ are independent if and only if $\mathbf{X}^\top\mathbf{X}$ is diagonal.

3.2. Summary of the classical iid errors theory

Let $\mathbf{y}$ (n by 1) be a vector of observed values, $\mathbf{X}$ (n by p) the model matrix, and $\boldsymbol{\beta}$ (p by 1) the vector of coefficients. The model is

$$\mathbf{y} = \mathbf{X}\boldsymbol{\beta} + \boldsymbol{\epsilon} \tag{25}$$

i.e., $\mathbf{y}_i = \mathbf{X}_i\boldsymbol{\beta} + \boldsymbol{\epsilon}_i$, where the vector $\boldsymbol{\epsilon}$ of residuals is n by 1. Commonly, it will be assumed that $\text{E}[\boldsymbol{\epsilon}] = \mathbf{0}$.

The following summarizes important theoretical results:

- The least squares *normal* equations are

$$\mathbf{X}^\top\mathbf{X}\boldsymbol{\beta} = \mathbf{X}^\top\mathbf{y}$$

- Assuming $\boldsymbol{\epsilon}_i$ are iid normal, the least squares estimates are the maximum likelihood estimates.

- If variances are unequal, modify the *normal* equations to

$$\mathbf{X}^\top\mathbf{W}\mathbf{X}\boldsymbol{\beta} = \mathbf{X}^\top\mathbf{W}\mathbf{y}$$

  where $\mathbf{W}$ is a diagonal matrix with elements equal to the inverses of the variances (justification is from maximum likelihood, or argue that leverage should be independent of variance)

- More generally, if $\boldsymbol{\epsilon}$ is multivariate normal with known variance-covariance matrix $\boldsymbol{\Sigma}$, then ML theory gives the equation as above with $\mathbf{W} = \boldsymbol{\Sigma}^{-1}$.

General variance-covariance matrix – comments

Two values with a high positive correlation contain, jointly, less information than two independent values. The extreme case is where the correlation is 1 (or -1), they duplicate the same information.

More often. if observations are not iid, the form of the variance-covariance matrix $\mathbf{\Sigma}$ will not be known. Many different special methods are available for various special cases that occur in practice. Models that may be relevant include time-series models, spatial analysis and multi-level models.

Limitations

Standard errors, t- and F-statristics may be optimistic, because of model selection. Almost inevitably, none of the models on offer will be strictly correct. Mis-specification of the fixed effects, and to a lesser extent of the random effects, is likely to bias model estimates, at the same time inflating the error variance or variances.

4. Spline Terms, Smoothing Splines and GAM Models

See Maindonald & Braun (2010, Section 7.5, pp. 234–238). See Wood (2006) for a more complete account of regression splines.

Splines are polynomial curves that are joined at internal *knots*. A common requirement is that the slope and the second derivative must agree at the knots, and must be continuous over the whole range of values. Here attention will be limited to cubic splines, which are the most commonly used form of spline. Natural splines have the (often sensible) further constraint that the second derivative is zero at the boundary knots, or *boundary knots*. Beyond one or other boundary knot, the curve then extrapolates as a straight line. Boundary knots are commonly taken to coincide with the extremes of the data, but this is not necessary.

It can be shown that spline curves can be expressed as a linear combination of basis functions. The choice of basis functions is not unique. Each set of spline basis terms is a kitset for constructing a flexible variety of curves. As the number of knots increases, it becomes possible to accommodate an ever increasing variety of curves. Note that the basis functions depend on the x-values that are chosen.

For splines whose slopes are unconstrained at the boundary knots, B-splines as implemented in the *splines* package offer a choice of basis functions that works well. For natural splines, the N-splines that are implemented by the function `ns()` in the same package are, again, one of a number of possibilities. Here, the natural cubic splines will be the main focus.

4.1. Basis functions for spline curves

The columns A and B of Figure 3 show alternative choices of natural spline bases (alternative 'kitsets') for each of degrees 1, 2 and 3 (additional to the intercept), when x ranges from 0 to 1 in increments of 0.02.

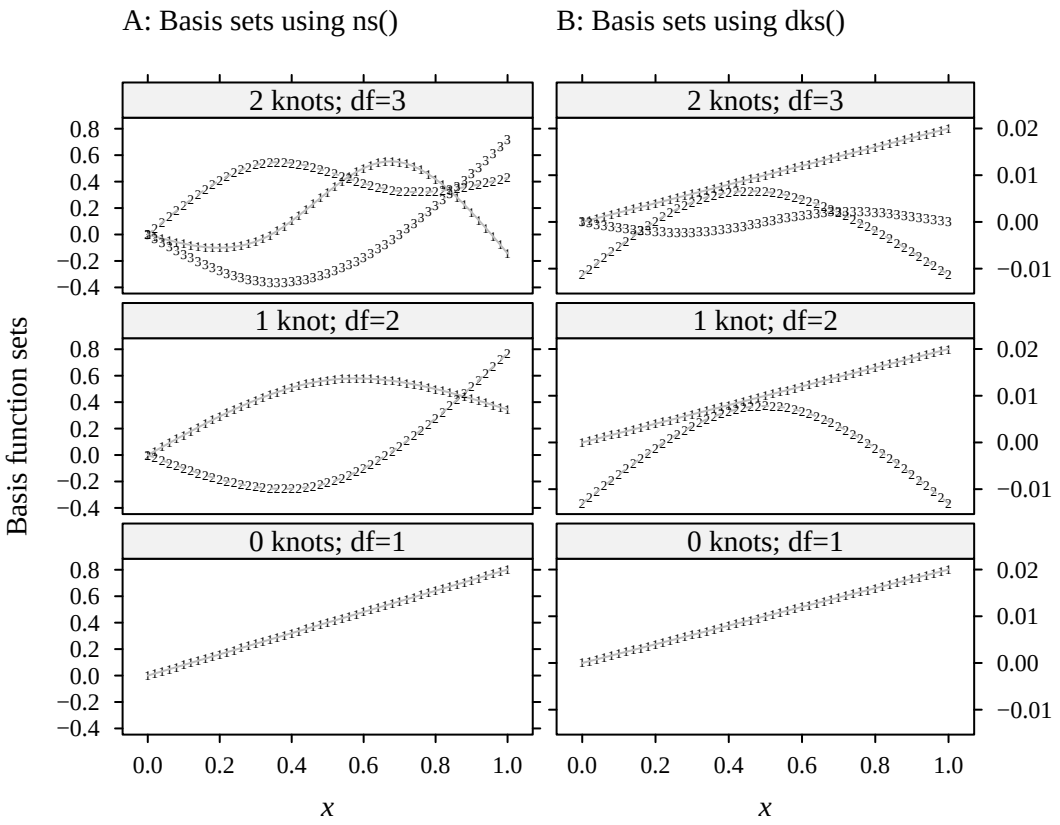

Figure 3: Sets of spline bases. Basis function 1 is plotted with 1's, and so on. The constant term, which would appear as a horizontal line, is not shown. The function `ns()` (*splines* package) generated the sets of bases shown in Column A, while the function `dks()` (see below) generated the sets of bases shown in Column B. In each case add 1 to the degrees of freedom to account for the intercept.

Observe the attributes `knots` and `Boundary.knots`

```
> x <- (0:50)/50
> Xns <- ns(x, df=3)
> round(attributes(Xns)$knots, 3)
33.33333% 66.66667%
    0.333     0.667
> attributes(Xns)$Boundary.knots
[1] 0 1
```

Code that will create the topmost plot in the left column (A) is:

```
## Use ns() to qenerate a matrix of values of the basis functions
library(splines)
library(lattice)
Xns <- data.frame(ns(x, df=3))
xyplot(X1+X2+X3 ~ x, data=Xns, outer=FALSE, type="b",
       xlab="x", ylab="Natural spline basis function set",
       par.settings=simpleTheme(pch=paste(1:3)))
```

Now consider an alternative choice of basis functions that was used to generate the panels in the Column B – a choice that has the sometimes useful property that the first spline basis term, after the intercept, is a straight line.[3]

The formula that follows assumes that x has been scaled so that values range from 0 to 1. The lower boundary knot is chosen to coincide with the lowest value, and the upper boundary knot to coincide with the largest value. Internal knots are placed at $z_1, z_2, \ldots, z_k$. Let $z_0 = 0$, and let

$$\begin{aligned} D(x, i, z) = (z[i] + z[i-1] - 1) * (z[i] - z[i-1]) * ((x - 0.5)^2 - 1/12)/4 + \\ ((|x - z[i]| - 0.5)^2 + (|x - z[i-1]| - 0.5)^2 - 0.5) * \\ (|x - z[i]| - |x - z[i-1]| + (z[i] - z[i-1]) * (2 * x - z[i] - z[i-1]))/24 \end{aligned}$$

Then the ith row of the model matrix $\mathbf{X}$ is

$$\mathbf{x}_i^\top = [1, x_i, D(x_i, 1, z), , D(x_i, 2, z), \ldots, D(x_i, k, z)]$$

The following function can be used to generate the model matrix $\mathbf{X}$:

```
dks <- function(x=engine$Size, knots=NULL, intercept=FALSE){
    dk <- function(x, i, z)
      (z[i+1]+z[i]-1)*(z[i+1]-z[i])*((x-0.5)^2-1/12)/4 +
      ((abs(x-z[i+1])-0.5)^2+(abs(x-z[i])-0.5)^2-0.5)*
      (abs(x-z[i+1])-abs(x-z[i])+(z[i+1]-z[i])*(2*x-z[i+1]-z[i]))/24
    xcol <- if(intercept) 2 else 1
    q <- length(knots)+xcol
    ## Form spline basis matrix
    n <- length(x)
    X <- matrix(1,n,q)
    X[,xcol] <- 0.02*x
    knots <- c(0,knots)
    if(q>xcol) X[, (xcol+1):q] <-
                  outer(x,1:(q-xcol),
                        FUN=function(x, i)dk(x, i, knots))
    X
  }
```

Cubic regression splines – examples

The following uses a cubic regression spline to explain the dependence of `loss` on `Fe`, by default for data in the data set corrosion. The following code can be used with any of the following:

- the function `ns()` (*splines*), which generates natural spline bases
- the function `dks()` which is defined above, which also generates natural spline bases

[3] Another useful feature of this choice of basis functions is that there is a reasonable simple formula for calculating $\int f''(x)^2 dx$. This will come in handy when we come to work with smoothing splines.

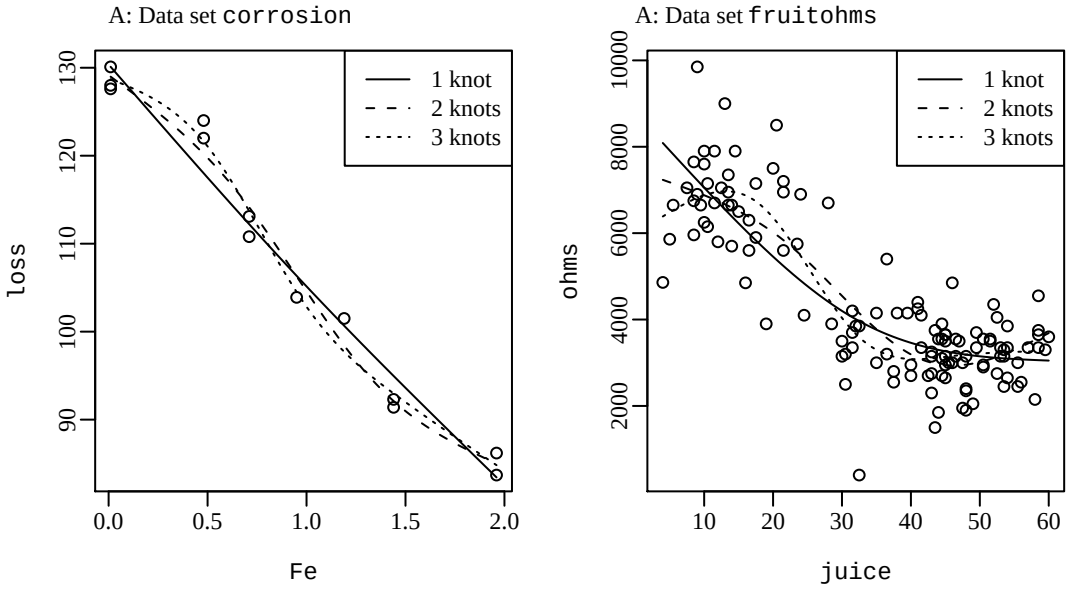

Figure 4: Regression natural spline fits to data (A) from the dataset `corrosion` and (B) from the dataset `fruitohms`.

- the function `bs()` (*splines*), which generates B-spline bases. (The second derivative is unconstrained at the boundary knots.)

```
rsfit
function(xk=(1:3)/4, form=loss~Fe, data=corrosion,
           xp=(0:500)/500, FUN=dks)
{
    vars <- all.vars(form)
    y <- data[,vars[1]]
    x <- data[,vars[2]]
    ran <- range(x)
    x <- (x-ran[1])/diff(ran)
    X <- FUN(x=x, knots=xk)
    b <- coef(lm(y ~ X))
    Xp <- model.matrix(~FUN(xp,knots=xk))
    data.frame(x=ran[1]+xp*diff(ran), y=Xp%*%b)
  }
```

Figure 4 demonstrates the use of this function. The left panel uses data from the data set `corrosion`, while the right panel uses data from the data set `fruitohms` (*DAAG*).

The following code may be used:

```
par(mfrow=c(1,2))
## Panel A
plot(loss ~ Fe, data=corrosion)
legend("topright", lty=1:3, legend=c("1 knot","2 knots","3 knots"))
for(k in 1:3){
  df <- rsfit(xk=(1:k)/(k+1), form=loss ~ Fe, data=corrosion,
```

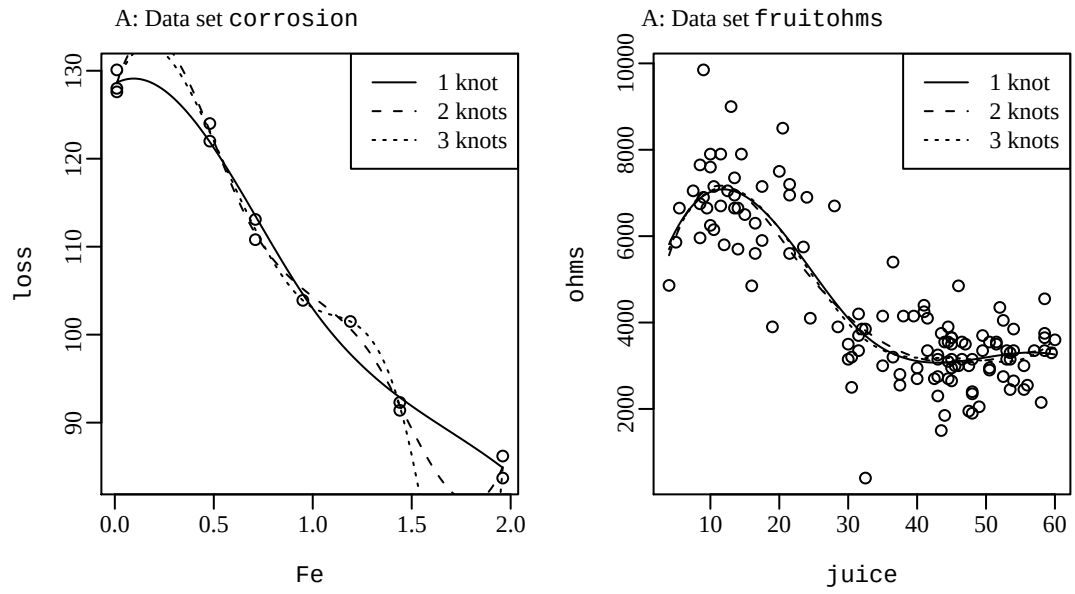

Figure 5: Regression B-spline fits to data (A) from the dataset `corrosion` and (B) from the dataset `fruitohms`. Each curve has two degrees of freedom more than the corresponding natural spline curve in Figure 4.

```
                FUN=FUN)
  with(df, lines(y~x, lty=k))
}
## Panel B
## Replace 'loss ~ Fe' by 'ohms ~ juice' and data=corrosion
## by data=fruitohms
par(mfrow=c(1,1))
```

Notice how fitting a curve 3 knots in Figure 4 has led to a curve that bends in an unsatisfactory manner at the boundaries. This is a typical consequence of over-fitting. With four knots, both curves exhibit such behaviour.

Alternatively, the function `rsfit()` can be called with the argument `FUN=ns`. For this, first attach the *splines* package. The function `ns()` generates an alternative natural spline basis, and exactly the same graphs result.

A further possibility is to call the function `rsfit()` with the argument `FUN=bs`. Again, this requires the *splines* package. Figure 5 shows the result. For these datasets, natural splines, as in Figure 4, are clearly preferable.

4.1.1. How many degrees of freedom?

A quadratic (hill or valley-shaped curve) has one degree of freedom for the intercept, plus one degree of freedom for the slope, plus one degree of freedom for the curvature. Every additional point where the second derivative is zero (the slope instantaneously ceases changing) accounts for one additional degree of freedom.

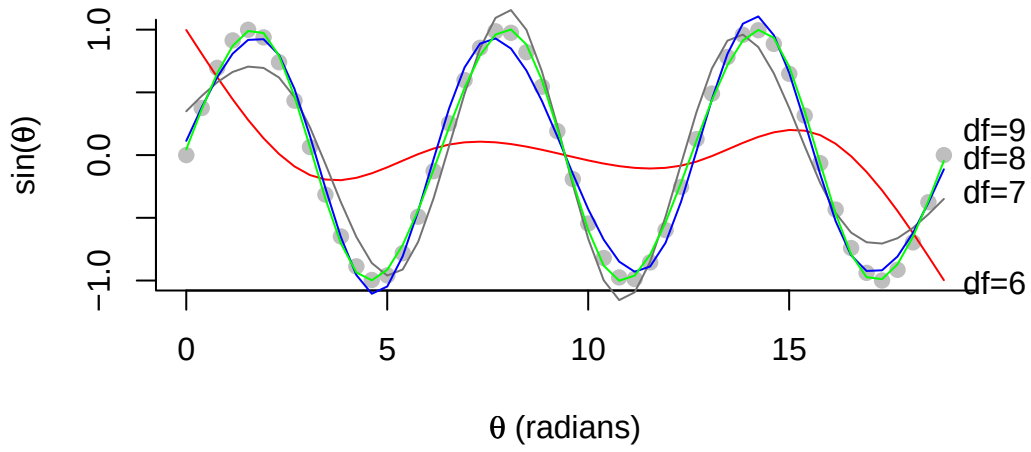

Figure 6: Regression spline fit to points that are determined by a sine curve, over the range 0 to 6π. A 6 d.f. natural spline curve is clearly grossly inadequate. A 7 d.f. curve gets the broad shape more or less correct. An 8 d.f. curve is clearly preferable. The improvement with a 9 d.f. curve is more marginal.

Consider the following sine curve

```
x <- seq(from=0, to = 6*pi, length=50)
y <- sin(x)
plot(x,y)
```

The first "hill" accounts for two degrees of freedom. There are five additional valleys/hills, accounting in total for seven degrees of freedom, additional to the intercept. Hence we try

```
lines(x, fitted(lm(y~ns(x,7))), col="red")
```

Any fewer degrees of freedom is grossly inadequate. The choice `df=6` shows how bad the spline fit can be when there are not enough degrees of freedom to capture the major features of the data. The choice `df=8` is an obviously worthwhile improvement on `df=7`. Figure 6 provides a comparison.

Note If B-splines are used (function `bs()`), two extra degrees of freedom might seem required to capture the broad shape of the curve. There is however a choice in whether the available degrees of freedom are used to give freedom to change shape near the boundary knots, or whether to fit the necessary number of hills and valleys. The least squares algorithm uses that discretion.

4.2. Smoothing splines

Smoothing splines place knots at each data point, thus avoiding arbitrariness in the choice of knots. A penalty, some multiple λ of the integral of the squared second derivative of y with respect to x, is however added to the residual sum of squares, penalizing any change in the slope. The smoother $f(x)$ has the form

$$\begin{aligned} f(x) &= \beta_0 + \beta_1\phi_1(x) + \ldots + \beta_k\phi_k(x) \\ &= \boldsymbol{\beta}^\top \boldsymbol{\phi}(x) \end{aligned}$$

where $\phi_1(x), \ldots, \phi_k(x)$ are the spline basis terms, and

$$\boldsymbol{\phi}(x) = (1, \phi_1(x), \ldots \phi_k(x)\ )$$

The penalty is then:

$$\begin{aligned} \lambda \int f''(x)^2 dx &= (\beta_1\phi_1''(x) + \ldots + \beta_k\phi_k''(x))^2\, dx \\ &= \lambda \int (\boldsymbol{\beta}^\top \boldsymbol{\phi}''(x))^2\, dx \\ &= \lambda \int (\boldsymbol{\beta}^\top \boldsymbol{\phi}''(x)\, \boldsymbol{\phi}''(x)^\top \boldsymbol{\beta})\, dx \\ &= \lambda\, \boldsymbol{\beta}^\top (\int \boldsymbol{\phi}''(x)\, \boldsymbol{\phi}''(x)^\top dx)\, \boldsymbol{\beta} \qquad \text{(matrix multiplication is associative)} \\ &= \boldsymbol{\beta}^\top \mathbf{S}\, \boldsymbol{\beta}, \text{ where } \mathbf{S} = \int \boldsymbol{\phi}(x)\, \boldsymbol{\phi}^\top(x) dx \end{aligned}$$

where the integral is over the range of x. The non-zero elements of $\mathbf{S}$ (rows and columns after the second) have the form

$$\int_0^1 D''(x, i, z)\, D''(x, j, z) dx$$

Let $\mathbf{X}$ be the matrix of spline basis elements, evaluated at $\mathbf{x}$. Instead of minimizing $||\mathbf{y} - \mathbf{X}\,\boldsymbol{\beta}||^2$, the model is fitted by minimizing:

$$\begin{aligned} ||\mathbf{y} - \mathbf{X}\boldsymbol{\beta}||^2 \ + \ \lambda \int f''(x)^2 dx &= ||\mathbf{y} - \mathbf{X}\boldsymbol{\beta}||^2 \ + \ \lambda \boldsymbol{\beta}^\top \mathbf{S}\, \boldsymbol{\beta} \\ &= ||\mathbf{y} - \mathbf{X}\boldsymbol{\beta}||^2 \ + \ \lambda\, ||\boldsymbol{\beta}\, \mathbf{C}||^2, \ \text{ where } \mathbf{C} \text{ satisfies } \mathbf{C}^\top \mathbf{C} = \mathbf{S} \end{aligned}$$

This is equal to the sum of squares of elements of

$$\begin{bmatrix} \mathbf{y} \\ \mathbf{0} \end{bmatrix} - \begin{bmatrix} \mathbf{X} \\ \sqrt{\lambda}\, \mathbf{C} \end{bmatrix} \boldsymbol{\beta}$$

Observe that the penalty is the integral of a squared term, so that $\boldsymbol{\beta}^\top \mathbf{S}\, \boldsymbol{\beta}$ is positive for all $\boldsymbol{\beta}$, implying that $\mathbf{S}$ is positive definite symmetric. Thus $\mathbf{C}$ can be chosen to be the Cholesky

decomposition of $\mathbf{S}$. This is then a least squares calculation. A criterion is required for choosing λ. Choice of λ will be described below.

The cubic spline basis that was introduced above makes it straightforward to compute $\int f''(x)^2 dx$. It may be shown, using some moderately intricate algebra, that the second derivative of the function $D(x, i, z)$ is:

$$\int_0^1 D''(x, i, z)\, D''(x, j, z) dx = D(z_{i+1}, j, z) - D(z_i, j, z)$$

4.2.1. Code that can be used for the calculations

The code now given does not claim to be particularly efficient. It does however demonstrate how relatively easy it is to implement these calculations, at least in this simple single explanatory variable case. The following function can be used to generate, first $\mathbf{S}$ and then a matrix $\mathbf{C}$ such $\mathbf{C}^\top\mathbf{C} = \mathbf{S}$.

```
dkp <- function(knots){
  dk <- function(x, i, z)
      (z[i+1]+z[i]-1)*(z[i+1]-z[i])*((x-0.5)^2-1/12)/4 +
      ((abs(x-z[i+1])-0.5)^2+(abs(x-z[i])-0.5)^2-0.5)*
      (abs(x-z[i+1])-abs(x-z[i])+(z[i+1]-z[i])*(2*x-z[i+1]-z[i]))/24
  q <- length(knots)+2
  ## Form penalty matrix
  S <- matrix(0,q,q)
  knots <- c(0, knots)
  S[3:q,3:q] <- outer(1:(q-2),1:(q-2),FUN=function(i,j)
                      dk(knots[i+1], j, knots) -
                      dk(knots[i], j, knots))
  d <- eigen(S, symmetric=TRUE)
  d$vectors%*%diag(d$values^0.5)%*%t(d$vectors)
  }
```

The following function `smfit()` fits a cubic spline. By default, knots are placed at the data points, i.e., it fits a cubic smoothing spline.

```
smfit <-
  function(x, y, knots=NULL, lambda){
    ran <- range(x)
    x <- (x-ran[1])/diff(ran)
    if(is.null(knots)){
      xk <- sort(unique(x))
      knots <- xk[-c(1,length(xk))]
    }
    q <- length(knots) + 2
    n <- length(x)
    X <- model.matrix(~dks(x, knots))
    S <- dkp(knots)
    y[(n+1):(n+q)] <- 0
    obj <- list(knots=knots, range=ran, lambda=lambda,
                coef=matrix(0, nrow=dim(X)[2], ncol=length(lambda)))
    i <- 0
    for(lam in lambda){
      i <- i+1
```

```
    Xa <- rbind(X, sqrt(lam)*S)
    obj$coef[,i] <- coef(lm(y ~ Xa-1))
}
  obj
}
```

Figure 7 shows smoothing spline fits to the same data as in Figure 5. The plot on the left can be obtained with:

```
y <- corrosion$loss
x <- corrosion$Fe
mod.2 <- smfit(x, y, knots=NULL,
               lambda=c(0.001, 0.0001, 0.00001))
b <- mod.2$coef
knots <- mod.2$knots
ran <- mod.2$range
lambda <- mod.2$lambda
xp <- (0:500)/500
Xp <- model.matrix(~dks(xp, knots))
hat <- Xp%*%b
dhat <- model.matrix(~dks((x-ran[1])/diff(ran), knots))%*%b
rss <- apply(dhat, 2, function(z)sum((y-z)^2))
plot(y ~ x)
for(i in 1:length(lambda)){
  lines(ran1]+xp*diff(ran), hat[,i], lty=i)
}
```

Details of the calculation of the GCV statistic were given in Subsection 2.6.2

4.3. Penalized splines

The placing of a knot at each data point may be overkill. Penalized splines compromise by choosing many more knots than would be appropriate for regression splines, but in cases where the number of points is large many fewer than the number of points. Points are spaced at regular intervals through the data.

For fitting penalized splines, the only change required to the code just given is that the knots need to be specified.

4.4. Exercises

- The `covsample` dataset that is in the *DAAGxtras* package gives forest cover type (one of eight types) as a function of various environmental attributes. There is no information on geographical coordinates. However, it might be expected that there would be an ordering in the data, reflecting some kind of geographical ordering. If so, this is likely to be reflected in a systematic variation in the environmental attributes. The first 10 of there are continuous variables. These are: Elevation, Aspect, Slope, Horizontal_Distance_To_Hydrology, Vertical_Distance_To_Hydrology, Horizontal_Distance_To_Roadways, Hillshade_9am, Hillshade_Noon, Hillshade_3pm, Horizontal_Distance_To_Fire_Points.

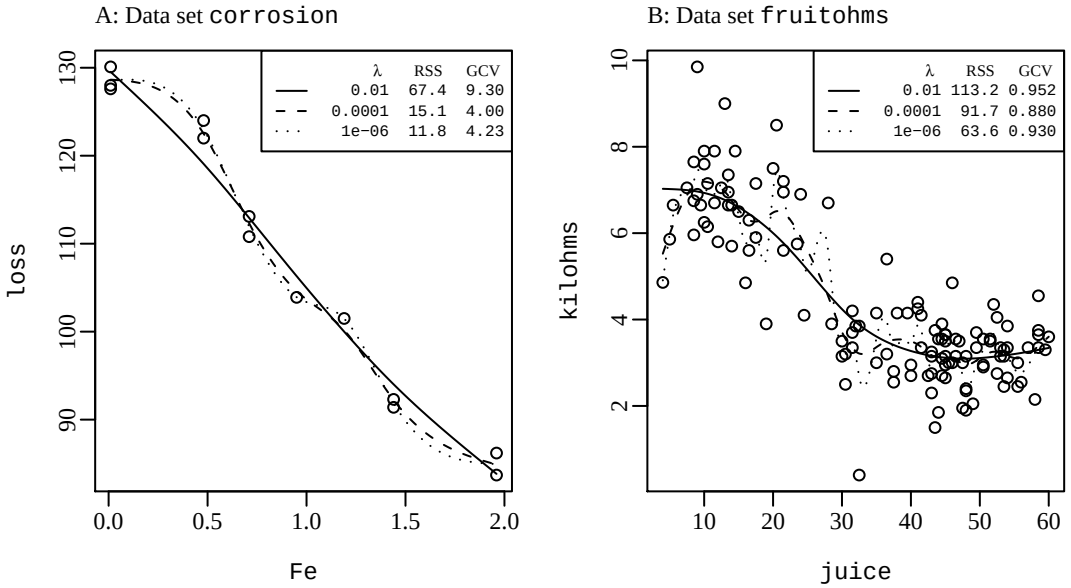

Figure 7: Smoothing spline fits to data (A) from the dataset `corrosion` and (B) from the dataset `fruitohms`. The rationale for the GCV statistic was described in Subsection 2.6.2.

Investigate whether the first of these (Elevation) seems to vary systematically through the data. We create a variable `dist` that measures distance through the data.

```
covdf <- cbind(covsample[, 1:10], dist=(1:11318-0.5)/11318)
```

- Define

$$r_k(x,z) = ((z-0.5)^2 - 1/12) * ((x-0.5)^2 - 1/12)/4 - \quad (26)$$
$$((|x-z|-0.5)^4 - (|x-z|-0.5)^2/2 + 7/240)/24 \quad (27)$$

The function `rk()` may be used as a basis for quartic splines. If used for smoothing splines, cubic splines are fitted. The contribution of the penalty function is minimized by taking only those linear combinations of the columns of $\mathbf{X}$ that result in cubic splines.

5. Comparing Models – Resampling Approaches

Here our interest will be in comparing residual sums of squares for models that are not necessarily nested. The methodology will be demonstrated using the `fruitohms` data in the *DAAG* package.

Resampling methods can be a useful recourse when theoretical results are unavailable, or when the accuracy of asymptotic results cannot be guaranteed. They allow a removal or

weakening of normality assumptions. While a recourse that is often useful, they are not the answer to every problem of failure of assumptions.

Three methods will be noted:

1. Permutation methods locate a fitted model statistic (e.g., the residual mean square) within the distribution that is obtained when observed values are randomly permuted. If the residual mean square for the fitted model is in the upper tail of this distribution, this is taken as evidence that the model has some predictive power. Another possibility is to work with permuted residuals, thus estimating a permutation distribution for model coefficients.

2. In cross-validation, observations are split into k parts. For each of $i = 1, 2, \ldots, k$, the ith part is left aside for use in testing, the model is fitted to the remaining $k - 1$ parts, and predictions made for the ith part. This yields predictions for all observations that have been derived independently of those observations. An accuracy measure can then be computed.

3. The idea of the bootstrap is to regard the sample as a microcosm of the population. Repeated with replacement resamples are taken, and the model fitted to each resample, yielding a sampling distribution of parameter estimates. There are many variations on this simple scheme.

5.1. Notation & Strategy

For this initial discussion, denote the models as M1 and M2. We first fit the model M1 to the original data, then obtaining fitted values and residuals. Also, we fit M2, and note the change as measured by a statistic $\tilde{F}_{12}$ that is calculated just as for the usual F-statistic.

If M2 is not giving an improvement, then $\tilde{F}_{12}$ will differ only by statistical error from the F_{12} observed when a random set of residuals are added. It will, in effect, be a random sample from the distribution obtained when repeated random sets of residuals are added on to the fitted values, and F_{12} is calculated for each such new set of "observations". We can then locate $\tilde{F}_{12}$ within this distribution. If $\tilde{F}_{12}$ falls above its 95th percentile then we will judge that, at the 5% significance level, M2 improves on M1. The change when the residuals are correctly attached is, on this criterion, more than statistical variation.

With slight modification, the argument applies if the resdiduals are obtained by permuting the original residuals. If model M2 is not improving on M1, then the change when new "observations" are derived by adding randomly permuted residuals back onto the fitted values will be attributable to statistical variation.

It also applies, again in slightly modified from, if repeated boostrap samples of the original residuals are used. Note that we are not, here, deriving bootstrap estimates of some parameter. While there are theoretical issues even with this use of the bootstrap, we do not have the issues of bias that commonly arise when the bootstrap is used to estimate variance-like statistics.

Note that the three distributions contemplated above are different. There will, accordingly, be differences of power between the three approaches.

The methodology can be varied or extended in various ways:

- We can choose M2 if it appears, on average, to improve on M1, i.e., choose M2 if the p-value is less than 0.5.
- This same style of approach can be used if M2 is selected from a wider class of models. The model selection step must then be repeated for each new bootstrap or permutation sample.

5.2. References to worked examples

For examples of the use of resampling methods, see Maindonald & Braun (2010) thus:

p.90: Brief general comments;

pp.128–132 (Section 4.7): Permutation methods & the bootstrap;

pp.152–158 (Section 5.4): Cross-validation & the bootstrap, in straight line regression;

pp.255–256 (Subsection 8.2.4): Cross-validation, in logistic regression;

pp.382–383 (Subsections 12.1.2): Use of the bootstrap to check the stability of a principal components plot;

pp.387–392 (Subsections 12.2.2 & 12.2.3): Cross-validation, in discriminant analysis.

pp.399–406 (Subsection 12.3.3): Use of cross-validation in variable selection for discriminant analysis.

6. Generalized Linear Models (GLMs)

The main case of interest will be logistic regression models, where the outcome is binary, i.e. 0 or 1. There will be some discussion of the general theory, but mainly for its relevance to this special case.

For examples of logistic regression models, see Maindonald & Braun (2010, Sections 8.1 & 8.2). Suitable texts for further reading and reference, in increasing order of technical demands, are Faraway (2006); Wood (2006); McCullagh and Nelder (1989).

In principle, models with a 0/1 outcome can be fitted using least squares. If unweighted least squares is used, and the variance really is that for a Binomial(n, p) distribution with n = 1, then estimates will be inefficient, though the effect will be of little consequence if fitted proportions lie between perhaps 0.25 and 0.75. Some fitted values may be less than 0 or greater than 1. Where a continuous explanatory variable has a non-zero coefficient, extrapolation to suitably small or suitably large values will always yield predicted proportions that are outside the range (0,1).

The efficiency of least squares estimates can be improved by taking the fitted proportions from an initial least squares fit, and using these to determine weights for a second fit. This is a step on the way to using a maximum likelihood fit for a generalized linear model. The remedy for preventing silly predicted values is to fit a linear model on the scale of a suitably transformed predicted value, which is exactly what GLMs are designed to do.

It will be interesting to compare logistic regression fits with least squares fits, for roughly equivalent models, to see what difference it makes.

6.1. Brief summary of theory

- As before, we have $\boldsymbol{\mu} = \mathrm{E}[\mathbf{y}]$ (n by 1), $\mathbf{X}$ (n by p), $\boldsymbol{\beta}$ (p by 1), and $\boldsymbol{\epsilon}$ (n by 1).
- The model is now
$$f(\boldsymbol{\mu}) = \mathbf{X}\boldsymbol{\beta}, \quad \text{where } \mathrm{E}[\mathbf{y}] = \boldsymbol{\mu}$$
Here, $f()$, which must be monotonic, has the name *link* function. For example,
$$f(\boldsymbol{\mu}_i) = \log(\frac{\boldsymbol{\mu}_i}{N_i - \boldsymbol{\mu}_i})$$
- The distribution of y_i is a function of the predicted value μ_i, independently for different observations. The distributions thus cannot be described as identical.
- An extension is to the quasi-exponential family, where the variance is a constant multiple of an exponential family variance. The multiplying constant is estimated as part of the analysis.
- Commonly used distributions are the normal, binomial and Poisson. Applications for models with quasibinomial and quasipoisson errors may be more extensive than for their exponential family counterparts.
- GLMs with binomial errors are formally equivalant to discriminant models where there are two categories. The GLM framework has advantages for some problems.
- Output is in much the same form as for the `lm` models. There are additional subtleties of interpretation – a `z value` is not a t-statistic, though for some GLMs that yield `z values` there are specific circumstances where it is reasonable to treat them `z values` as t-statistics.
[More technically, they are Wald statistics.]

GLMs – commentary on the theory

The exponential family of distributions

These are distributions such that the loglikelihood for a single observation y_i can be written

$$\log \ell_i = \phi^{-1}[\theta_i y_i - d(\theta_i)] + c(\phi, y_i)$$

where θ_i is a function of $\mu_i = \mathrm{E}[y_i]$. There are simple expressions for the mean and variance:

$$\mathrm{E}[y_i|\theta] = d'(\theta_i); \ \mathrm{var}[y_i] = \phi d''(\theta_i)$$

Then

$$\log \ell = \phi^{-1}[\sum_{i=1}^{n} \log \ell_i] \tag{28}$$

The three most common examples are (leaving off the subscript i.

$$\begin{aligned}
\text{normal:}\quad \ell &= (\sqrt{2\pi}\sigma)^{-1} e^{-\frac{(y-\mu)^2}{2\sigma^2}} && \phi = \sigma^2;\ \theta = \mu;\ d(\theta) = \tfrac{1}{2}\mu^2 \\
\log \ell &= \tfrac{1}{2}\sigma^{-2}(y-\mu)^2 + \log(\sqrt{2\pi}\sigma) \\
&= \sigma^{-2}(\mu y - \tfrac{1}{2}\mu^2) + c(\sigma^2, y) \\
\text{binomial:}\quad \ell &= \binom{m}{r} \pi^r (1-\pi)^{m-r} && \phi = 1;\ \theta = \log \tfrac{\pi}{1-\pi};\ d(\theta) = m\log(1+e^\theta) \\
\log \ell &= r(\log \pi - \log(1-\pi)) + m\log(1-\pi)) + \log \binom{m}{r} \\
&= \theta r + m\log(1+e^\theta) + c(n, y) \\
\text{poisson:}\quad \ell &= \tfrac{1}{x!}\, e^{-\lambda} \lambda^x && \phi = 1;\ \theta = \log \lambda;\ d(\theta) = e^\theta = \lambda \\
\log \ell &= -\lambda + x\log\lambda - \log(x!) \\
&= \theta y - e^\theta + c(y)
\end{aligned}$$

Maximum likelihood parameter estimates

The equations that maximize the log-likelihood take the same form for all distributions that are of exponential form. We have

$$\boldsymbol{\eta} = f(\boldsymbol{\mu}) = E(\mathbf{y}) = \mathbf{X}\boldsymbol{\beta}$$

where $\boldsymbol{\mu} = E[\mathbf{y}]$. For the ith observation

$$\eta_i = f(\mu_i) = \mathbf{x}_i^\top \boldsymbol{\beta}, \text{ where } \mathbf{x}_i \text{ is the } i^{\text{th}} \text{ row of } \mathbf{X}.$$

Now differentiate Equation 6.1 with respect to $\boldsymbol{\beta}$, sum over i, and equate the result to 0. This is equivalent to differentiating with respect to each of the elements of $\boldsymbol{\beta}$ in turn, then putting the results together into a vector.

Observe first that

$$\frac{d\log \ell_i}{d\theta_i} = \phi^{-1} \sum_{i=1}^{n} [y_i - d'(\theta_i)] \tag{29}$$

$$= \phi^{-1} \sum_{i=1}^{n} [y_i - \mu_i] \tag{30}$$

Then

$$\frac{d\log \ell_i}{d\boldsymbol{\beta}} = \frac{d\log \ell_i}{d\mu_i} \frac{d\theta_i}{d\mu_i} \frac{d\mu_i}{d\eta_i} \frac{d\eta_i}{d\boldsymbol{\beta}} \tag{31}$$

Summing over i and equating to 0 gives:

$$0 = \phi^{-1} \sum_{i=1}^{n} (y_i - \mu_i) \frac{d\theta_i}{d\mu_i} \frac{d\mu_i}{d\eta_i} \mathbf{x}_i^\top \tag{32}$$

$$= \mathbf{X}^\top \mathbf{W}(\mathbf{y} - \boldsymbol{\mu}), \text{ where } \mathbf{W} = \text{diag}(\frac{d\theta_i}{d\mu_i} \frac{d\mu_i}{d\eta_i}) \tag{33}$$

Thus, assuming a distribution from the exponential family, the maximum likelihood estimates of the parameters are given by

$$\mathbf{X}^\top \mathbf{W}\boldsymbol{\mu} = \mathbf{X}^\top \mathbf{W}\mathbf{y}, \text{ where } f(\boldsymbol{\mu}) = \mathbf{X}\boldsymbol{\beta}, \text{ and } \mathbf{W} \text{ is a diagonal matrix.}$$

Note that:

- The (diagonal) elements $\mathbf{W}_{ii}$ of $\mathbf{W}$ are in general functions both of var$[\mathbf{y}_i]$ and of μ_i
- For the canonical link function, $\mathbf{W}$ is the identity matrix, and the maximum likelihood equations simplify to
$$\mathbf{X}^\top \boldsymbol{\mu} = \mathbf{X}^\top \mathbf{y}$$

Adequacy of theoretical approximations

- Except in special cases, the statistical properties of parameters rely on asymptotic results. Standards errors and t-statistics rely on first-order Taylor series approximations that, in the worst case, can fail badly. This applies, especially, to binary logistic regression.
- For logistic regression models, and Poisson models with small expected values, assessments of predictive accuracy should be derived using a resampling approach, perhaps cross-validation.

6.2. Computation for GLMs

Above, it was noted that

$$\mathbf{X}^\top \mathbf{W}\boldsymbol{\mu} = \mathbf{X}^\top \mathbf{W}\mathbf{y}$$

where

$$f(\boldsymbol{\mu}) = E(\mathbf{y}) = \mathbf{X}\boldsymbol{\beta},$$

for a suitable monotonic "link" function $f()$. Here, $\boldsymbol{\mu} = g(\mathbf{X}\boldsymbol{\beta})$ where $g()$ is the inverse function to $f()$. For all the common link functions $f()$, other than the identity, $\boldsymbol{\mu}$ is a non-linear function of the regression parameters. Also, the matrix $\mathbf{W}$ is in general a function of the parameters that are to be estimated. Iteratively reweighted least squares is used, i.e. Newton-Raphson. Each step involves a standard least squares calculation with a diagonal matrix of weights. [To be expanded . . .]

7. References

References

Bates, D, 2007. Comparing least squares calculations. *Vignette "Comparisons" accompanying the package "Matrix" for R.*

Bishop, C. M, 2006. *Pattern Recognition and Machine Learning*. Springer.
[Chapters 3 and 4 offer an interesting and somewhat novel perspective on regression and discriminant methods. The theoretical framework is that of Bayesian Decision theory. This is a demanding text. There is helpful comparative commentary on the methods that are described.]

Cook, R D and Weisberg, S., 1999. *Applied Regression Including Computing and Graphics*. Wiley.
[This emphasizes geometric insights, linear predictors (transformation of predictors, if possible, so that pairwise regression relationships are linear), and dimension reduction.]

Davis, T. 2006. Direct Methods for Sparse Linear Systems. Society for Industrial and Applied Mathematics.

Faraway, J. J. 2006. *Extending the Linear Model with R.* Chapman & Hall/CRC.

Koenker, R and Ng, P, 2003. SparseM: A sparse matrix package for R. *Journal of Statistical Software* 8(6).

Maindonald, J. H. and Braun, W.J. 2010. *Data Analysis and Graphics Using R – An Example-Based Approach.* 3rd edition, Cambridge University Press.
URL:http://wwwmaths.anu.edu.au/~johnm/r-book.html

McCullagh, P. and Nelder, J. A., 1989. *Generalized Linear Models*. Chapman and Hall, 2nd edition.

Wood, S. N., 2006. *Generalized Additive Models*. An Introduction with R. Chapman & Hall/CRC.
[This has an elegant treatment of linear models and generalized linear models, as a lead-in to generalized additive models.]

A. R Code for QR Using Modified Householder

```
house <-
  function(X, irows=NULL){
    n <- dim(X)[1]
    p <- dim(X)[2]
    if(is.null(irows))irows <- 1:min(p, n-1)
    for(i in irows){
      tf <- rep(c(FALSE, TRUE), c(i-1, n-i+1))
      tfnoti <- tf
      tfnoti[i] <- FALSE
      xi <- X[tf, i]
      eta2 <- sum(xi^2)
      eta <- sqrt(eta2)
      gamma <- 1/(eta*(eta+abs(xi[1])))
      X[i, i] <- eta
      if (i<p){
        etawsgn <- eta*sign(xi[1])
        for (j in (i+1):p){
          sumijprod <- sum(xi*X[tf, j])
          X[tfnoti, j] <- X[tfnoti, j] -
                              (sumijprod+etawsgn*X[i,j])*gamma*xi[-1]
          X[i, j] <- sumijprod/eta
        }
      }
    }
    X
  }
```