

## Part 4, Programming: Exercises

Introduction to R  
McMaster  
Fall 2005

1. *A challenging problem:* Iterated weighted least squares (IWLS) is a standard method of fitting generalized linear models to data. As described in Section 5.5 of the text (Fox, 2002), the IWLS algorithm applied to binomial logistic regression proceeds as follows:

- (a) Set the regression coefficients to initial values, such as  $\boldsymbol{\beta}^{(0)} = \mathbf{0}$  (where the superscript 0 indicates start values).
- (b) At each iteration  $t$  calculate the current fitted probabilities  $\boldsymbol{\mu}$ , variance-function values  $\boldsymbol{\nu}$ , working-response values  $\mathbf{z}$ , and weights  $\mathbf{w}$ :

$$\begin{aligned}\mu_i^{(t)} &= [1 + \exp(-\eta_i^{(t)})]^{-1} \\ v_i^{(t)} &= \mu_i^{(t)}(1 - \mu_i^{(t)}) \\ z_i^{(t)} &= \eta_i^{(t)} + (y_i - \mu_i^{(t)})/v_i^{(t)} \\ w_i^{(t)} &= n_i v_i\end{aligned}$$

Here,  $n_i$  represents the binomial denominator for the  $i$ th observation; for binary data, all of the  $n_i$  are 1.

- (c) Regress the working response on the predictors by weighted least squares, minimizing the weighted residual sum of squares

$$\sum_{i=1}^n w_i^{(t)} (z_i^{(t)} - \mathbf{x}'_i \boldsymbol{\beta})^2$$

where  $\mathbf{x}'_i$  is the  $i$ th row of the model matrix.

- (d) Repeat steps 2 and 3 until the regression coefficients stabilize at the maximum-likelihood estimator  $\hat{\boldsymbol{\beta}}$ .
- (e) Calculate the estimated asymptotic covariance matrix of the coefficients as

$$\hat{\mathcal{V}}(\hat{\boldsymbol{\beta}}) = (\mathbf{X}'\mathbf{W}\mathbf{X})^{-1}$$

where  $\mathbf{W} = \text{diag}\{w_i\}$  is the diagonal matrix of weights from the last iteration and  $\mathbf{X}$  is the model matrix.

**Problem:** Program this method in R. The function that you define should take (at least) three arguments: The model matrix  $\mathbf{X}$ ; the response vector of observed proportions  $\mathbf{y}$ ; and the vector of binomial denominators  $\mathbf{n}$ . I suggest that you let  $\mathbf{n}$  default to a vector of 1s (i.e.,

for binary data, where  $y$  consists of 0s and 1s), and that you attach a column of 1s to the model matrix for the regression constant so that the user does not have to do this when the function is called.

**Programming hints:**

- Adapt the structure of the example developed on pages 273–274 of the text (but note that the example is for *binary* logistic regression, while the current exercise is to program the more general *binomial* logit model).
  - Use the `lsfit` function to get the weighted-least-squares fit, calling the function as `lsfit(X, z, w, intercept=FALSE)`, where  $X$  is the model matrix;  $z$  is the current working response; and  $w$  is the current weight vector. The argument `intercept=FALSE` is needed because the model matrix already has a column of 1s. The function `lsfit` returns a list, with element `$coef` containing the regression coefficients. See `help(lsfit)` for details.
  - One tricky point is that `lsfit` requires that the weights ( $w$ ) be a *vector*, while your calculation will probably produce a *one-column matrix* of weights. You can coerce the weights to a vector using the function `as.vector`.
  - Return a list with the maximum-likelihood estimates of the coefficients, the covariance matrix of the coefficients, and the number of iterations required.
  - You can test your function on the `Mroz` data in `car`, being careful to make all of the variables numeric (as on page 275). You might also try fitting a binomial (as opposed to binary) logit model.
2. *Another challenging problem (though perhaps somewhat less so):* A matrix is said to be in (*reduced*) *row-echelon form* when it satisfies the following criteria:
- (a) All of its nonzero rows (if any) precede all of its zero rows (if any).
  - (b) The first entry (from left to right) — called the *leading entry* — in each nonzero row is 1.
  - (c) The leading entry in each nonzero row after the first is to the right of the leading entry in the previous row.
  - (d) All other entries are 0 in a column containing a leading entry.

A matrix can be put into row echelon form by a sequence of *elementary row operations*, which are of three types:

- (a) Multiply each entry in a row by a nonzero constant.
- (b) Add a multiple of one row to another, replacing the other row.
- (c) Exchange two rows.

*Gaussian elimination* is a method for reducing a matrix to row-echelon form by elementary row operations. Starting at the first row and first column of the matrix, and proceeding down and to the right:

- (a) If there is a 0 in the current row and column (called the *pivot*), if possible exchange for a lower row to bring a nonzero element into the pivot position; if there is no nonzero pivot available, move to the right and repeat this step. If there are no nonzero elements anywhere to the right (and below), then stop.

- (b) Divide the current row by the pivot, putting a 1 in the pivot position.
- (c) Proceeding through the other rows of the matrix, multiply the pivot row by the element in the pivot column in another row, subtracting the result from the other row; this zeroes out the pivot column.

Consider the following example:

$$\begin{bmatrix} -2 & 0 & -1 & 2 \\ 4 & 0 & 1 & 0 \\ 6 & 0 & 1 & 2 \end{bmatrix}$$

Divide row 1 by -2:

$$\begin{bmatrix} 1 & 0 & 0.5 & -1 \\ 4 & 0 & 1 & 0 \\ 6 & 0 & 1 & 2 \end{bmatrix}$$

Subtract  $4 \times$  row 1 from row 2:

$$\begin{bmatrix} 1 & 0 & 0.5 & -1 \\ 0 & 0 & -1 & 4 \\ 6 & 0 & 1 & 2 \end{bmatrix}$$

Subtract  $6 \times$  row 1 from row 3:

$$\begin{bmatrix} 1 & 0 & 0.5 & -1 \\ 0 & 0 & -1 & 4 \\ 0 & 0 & -2 & 8 \end{bmatrix}$$

Multiply row 2 by -1:

$$\begin{bmatrix} 1 & 0 & 0.5 & -1 \\ 0 & 0 & 1 & -4 \\ 0 & 0 & -2 & 8 \end{bmatrix}$$

Subtract  $0.5 \times$  row 2 from row 1:

$$\begin{bmatrix} 1 & 0 & 0 & 1 \\ 0 & 0 & 1 & -4 \\ 0 & 0 & -2 & 8 \end{bmatrix}$$

Add  $2 \times$  row 2 to row 3:

$$\begin{bmatrix} 1 & 0 & 0 & 1 \\ 0 & 0 & 1 & -4 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

The matrix is now in row-echelon form. The rank of a matrix is the number of nonzero rows in its row-echelon form, and so the matrix in this example is of rank 2.

**Problem:** Write an R function to calculate the row-echelon form of a matrix by elimination.

**Programming hints:**

- When you do “floating-point” arithmetic on a computer, there are almost always rounding errors. One consequence is that you cannot rely on a number being exactly equal to a value such as 0. When you test that an element, say  $x$ , is 0, therefore, you should do so within a tolerance — e.g.,  $|x| < 1 \times 10^{-6}$ .

- The computations tend to be more accurate if the absolute values of the pivots are as large as possible. Consequently, you can exchange a row for a lower one to get a larger pivot even if the element in the pivot position is nonzero.
3. *A less difficult problem:* Write a function to compute *running medians*. Running medians are a simple smoothing method usually applied to time-series. For example, for the numbers 7, 5, 2, 8, 5, 5, 9, 4, 7, 8, the running medians of length 3 are 5, 5, 5, 5, 5, 5, 7, 7. The first running median is the median of the three numbers 7, 5, and 2; the second running median is the median of 5, 2, and 8; and so on. Your function should take two arguments: the data (say, `x`), and the number of observations for each median (say, `length`). Notice that there are fewer running medians than observations. How many fewer?