

Improving and Debugging R Programs: Exercises

1. In the Basic R Programming exercises you wrote one or more functions. If these functions don't work properly, attempt to debug them. Once they work properly, try to increase their efficiency in time and memory usage.
2. The file `bugged-functions.R` on the course web site contains "solutions" to the Basic R Programming exercises, but most of these functions have a bug (or two) that either causes them to fail or, possibly only in certain circumstances, to give wrong answers. In each case, find the bug(s) and fix the function.

In the case of `reducedRowEchelonForm()`, the bugged function returns the right answer for some problems, such as

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

whose reduced row-echelon form is

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

but gives the wrong answer for others, such as the matrix

$$\begin{bmatrix} 16 & 2 & 3 & 13 \\ 5 & 11 & 10 & 8 \\ 9 & 7 & 6 & 12 \\ 4 & 14 & 15 & 1 \end{bmatrix}$$

whose correct reduced row-echelon form is

$$\begin{bmatrix} 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 3 \\ 0 & 0 & 1 & -3 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

The functions `pom()` and `cumlogit()` fit ordered and general-cumulative logit and probit models. These functions work properly but are terribly inefficient (for the same reason – so just check one of them). Profile one of these functions to determine why it is so slow; if possible, improve the function. *Note:* The inefficiency here was in my first attempt to program these problems, and is the consequence of trying to be "clever" in avoiding a loop. I was able to locate and clear the bottleneck in the program by profiling it.