histoRicalg – Preserving and Transfering Algorithmic Knowledge

John C Nash, Telfer School of Management, University of Ottawa, nashjc@uottawa.ca

July 6, 2017

1-page summary

Many (most?) of the algorithms making up the numerical building-blocks of scientific software today were developed and rendered into published code in the decade and a half between 1965 and 1980. Codes were mostly in Fortran, with some in Algol and BASIC. Slightly after this period, some were published in Pascal. Today the proportion of workers in scientific and statistical computation who are fluent with these computer languages is small, and the original authors are retired or deceased. Some of the algorithms and the codes implementing them were historic in having a large impact on computing while others remain historical in simply being of their time. However, it may be important in the future to understand and be able to use these ideas.

Unfortunately, there may be deficiencies in the codes, some of which are still in use. In my own fields of work as they apply to the R Project for Statistical Computing, the provenance of the optim::L-BFGS-B and the nlm functions are unclear, and nlm() has recently been shown by Maria Boehmer to be buggy. Even if there are no outright bugs, there may be incomplete code sections (i.e., features mentioned but not fully implemented) or there may be programming constructs that were used to exploit or work around particular hardware and operating system constraints.

Many older algorithms were left by the wayside as they were found to be non-performant or awkward to use as computer power became more readily available. However, new computing platforms, such as quantum computers, may be able to better use some of these discarded methods.

To avoid the potential loss of algorithmic knowledge, I have started to code some algorithms which I have used and/or developed into plain R code. R is suitably high level for describing algorithms clearly but succinctly in most cases. However, the features of Rmarkdown and the knitr processor – including its ability to have Fortran and other language chunks – makes R a good vehicle for presenting and documenting old codes. Computational efficiency is not necessarily a goal.

While some greybeards like myself can do much by reimplementing old algorithms, the process of reimplementing algorithms, documenting, checking and comparing their results would gain a lot more value if done in partnership with young workers. The latter would be exposed to the older programming languages as well as the algorithms, and personal interaction with people who were participants in work using those methods could amplify the knowledge transfer.

The rest of this document adds some detail to the general proposal.

Existing infrastructure for "old" algorithms

The following initiatives are a partial list (I welcome suggestions of others) of resources related to numerical methods infrastructure in R:

- package lbfgsb3: the 2011 Fortran code for L-BFGS-B (Morales and Nocedal (2011))
- RQmin: a vignette about Rayleigh Quotient minimization (JN??)
- snewton: a developmental package for a safeguarded Newton method (JN??)
- pracma: a collection of numerical methods (Hans Werner Borchers??)

- geradin: an implementation of a special Rayleigh Quotient minimizer (JN?)
- Nash-Beleites and others: a vignette about various packages in R for PLS and svd (??ref)
- jacobi: Bill Venables work on the Jacobi eigenvalue algorithm and its derivatives

A possible process

The first need is to identify algorithms or families of algorithms for which better and documented understanding is worthwhile. I believe that some advertising of an effort such as that documented here to the community, and particularly mature workers, would generate a number of suggestions.

The second need is a way to link those who have at least some understanding – even an awareness of a method and some bibliographic or computational source – with younger workers, possibly even undergraduates. My experiences with working with students have been both positive and negative. The negative ones have been associated with the Google Summer of Code and similar activities, where the communication was always at a distance. I have found there needs to be a face-to-face interaction from time to time and especially when starting some effort. This applied not only to young workers, but it is, I think, more critical with them, and exacerbated by cultural differences if they exist. Therefore, I believe it is important to find resources so that those involved in the histoRicalg effort can meet and work together for a day or so at the beginning of an exercise. For a relatively extensive effort (multiple methods and codes), some update meeting every few months would be in order.

A third goal should be to report on the activity. Places like JSS or the R-Journal are an obvious choice, but sessions at conferences or independent workshops are a parallel vehicle. Again some resources will need to be found.

Resourcing

The activities proposed here are a form of good housekeeping for software. Such activities are noticed most when absent. However, they should not be a major part of an enterprise such as R. I see a level of effort of 4-5 mature workers somewhat like myself and 8-12 younger workers (most likely undergraduate, graduate and post-doc). I would envisage a stipend equivalent to 3-months pay at a level much like Google Summer of Code (base $6000). Travel and accommodation becomes expensive over distances, so it may be helpful to try to cluster younger and older participants geographically. I do not envisage any pay for the mature workers. Most of us are likely retired. However, for that very reason, their participation should not impose out-of-pocket costs or inconvenience.

A caveat to the "work for free" proposition is that requests by organizations within the R family for specific help should be paid at reasonable rates, including to the junior members. For example, work on older algorithms may reveal an opportunity for some useful code requiring packaging or documentation or standardization beyond the goals of histoRicalg.

I will eschew providing a specific total for resources for the project at this time, as I believe it certain there will be discussion and adjustment of the ideas before a specific formal proposal is finalized. I welcome commentary and exchange of views.

Outputs

To date, I have prinarily encapsulated my experiences with older codes as an Rmarkdown or enriched LaTEX (LaTEX with R or other code chunks) vignettes that are rendered to standard pdf documents but which in source form contain the codes we have investigated and tests run. These have been rather informal to date, but I believe they could be improved by including sections as follows:

- description of the problems solved
- some background (provenance) of existing codes
- if possible, existing code and example runs
- R implementation(s)
- documentation of the R code
- examples
- if possible, tests and comparisons
- pros and cons of the method, which could include performance issues if relevant
- possibly commentary or assessments about the use of the method(s) in R, particularly if there are deficiencies in existing R code or packages.
- indexing of the methods and codes in various ways

Deciding on methods

References

Morales, José Luis, and Jorge Nocedal. 2011. “Remark on "Algorithm 778: L-Bfgs-B: Fortran Subroutines for Large-Scale Bound Constrained Optimization".” *ACM Transactions on Mathematical Software* 38 (1). Association for Computing Machinery (ACM). doi:10.1145/2049662.2049669.