Ruby Association
SciRuby's Fellowship

Carlos Agarie Junior

March 12, 2013

This last report summarizes what was done during the period of the fellowship, showing the progress made and what can be done in the future. The social impacts of the whole work are analysed as well.

Review of the proposed objectives

There were 4 objectives for this fellowship: writing and preparing documentation for the NMatrix library with RDoc, writing guides and tutorials, implementing matrix inversion for rational and VALUE dtypes and implementing determinants when CLAPACK isn't available. Half of them were successfully completed and much progress was achieved with the rest.

Writing documentation and guides was the main focus of this work during december and january, resulting in a (bigger than it should be) pull request to NMatrix[1], another one to RDoc[2] with changes to `RDoc::Parser::C` and the addition of the "finished" API documentation to SciRuby's website[3]. Two guides showing how to use the shortcuts[4] and the diverse methods of creating a NMatrix[5] were published.

Regarding rational and VALUE matrices inversion, the author is using an algorithm similar to ATLAS' `getri` function, which, for a given matrix A with decomposition LU, solves $A * A^{-1} = 1 \Rightarrow U * A^{-1} = L^{-1}$, as inverting triangular matrices is reasonably fast and there's already the function `trtri`.

The original studied algorithm for matrix inversion calculates the LU decomposition of the matrix and uses it to solve 2 systems with backsubstitution, as shown in [6]. Given a matrix A and its LU decomposition:

$$A * A^{-1} = 1 \Rightarrow (LU) * A^{-1} = 1$$

And:

$$L * y = b \; and \; U * x = y$$

So it's only a matter of using this procedure for each column of A^{-1}, as backsubstitution is straightforward with triangular matrices (both L and U).

This minor difference in the algorithms led to some bugs in the implementation, as the author was trying to use the LAPACK's version (following the rest of NMatrix's algorithms) but understood correctly only the above procedure. At the time of this report, the code is compiling, but the calculated results are incorrect.

Also, determinants can be easily calculated with the LU decomposition of a matrix: it's only a matter of multiplying the diagonal elements of the upper triangular matrix U and adjusting the sign depending on the number of row interchanges executed. It wasn't completed because matrix inversion is a more important feature and was prioritized in this work.

Social impact

As said in the previous reports, documentation is very important for the developer community to use a library. Not only API documentation, but guides, tutorials and an updated wiki with information about how to *use* it.

Other important feature were shortcuts, which are aliases and simple methods to create common matrices or extract (`#row` and `#column`) pieces from them. They improve the readability of programs using NMatrix and are a very useful way for people to learn about it. Also, given that many times they are based on MATLAB (Octave, IDL, etc) function names, it also helps people with experience in other environments to get started.

Another improvement was the added support for C++ files in RDoc[2]. At the time of this writing, the pull request wasn't merged yet, but as soon as it is, other projects using C++ will be able to benefit from it.

And the features which weren't finished (rational/value matrices inversion and determinants without CLAPACK) would help people with problems to install LAPACK. These will be addressed after the end of the fellowship and should be included in the next release of NMatrix.

Future work

In the short-term, the inversion of rational and VALUE matrices will be completed as soon as the current implementation is correctly debugged. The implementation of determinants without CLAPACK should follow right after, as was already stated in the review section above.

For the medium and long-terms, corrections and improvements to the rdoc documentation are expected. Guides teaching how to use NMatrix for more complicated operations, e.g. principal component analysis, are also necessary.

Another point of interest for the SciRuby project as a whole is to finish fixing the rb-gsl gem so other libraries, e.g. statsample, will be able to use NMatrix for fast linear algebra operations.

Conclusion

Much progress was made regarding NMatrix's documentation. This is invaluable for the community at large before SciRuby can be adopted as the solution for scientific computing in Ruby, following SciPy and NumPy in Python.

This progress is already receiving feedback from some developers. A week ago, the author of this work received an e-mail about some possible improvements to the documentation (including RDoc, the wiki and SciRuby's website) that could help others understand and use NMatrix.

Unfortunately, the other objectives weren't finished on time due to problems implementing matrix inversion, as already stated.

This work was very important for the author and allowed him to learn a lot about numerical algorithms, linear algebra and how Ruby works internally

(parts of the MRI C API). He is happy to be able to help SciRuby (and the Ruby language) somehow and will continue to discuss ideas at the mailing list and improve the codebase whenever possible.

Bibliography

[1] Pull request with all the changes to documentation. https://github.com/SciRuby/nmatrix/pull/56

[2] Pull request to RDoc. *C functions with casts now accepted by the parser*. https://github.com/rdoc/rdoc/pull/184

[3] Pull request to add NMatrix's documentation to Sciruby's website. https://github.com/SciRuby/sciruby.com/pull/2

[4] SciRuby's blog. *http://sciruby.com/blog/2012/12/07/how-to-use-nmatrix-shortcuts/*

[5] Carlos Agarie's blog. *How to create a NMatrix*. http://onox.com.br/2012/12/27/creating-a-new-nmatrix.html

[6] PRESS, W. H.; FLANNERY, B. P.; TEUKOLSKY, S. A.; VETTERLING, W. T. Numerical Recipes in C. Cambridge University Press; 2 edition (October 30, 1992)