USING RCPP TO SPEED UP R - THE SPRING 2020 UPDATE

Matt Wand

27th August 2020

WAND "SPEEDING UP R" TIMELINE

* Early 1990s. Wand realised need for this. Had learnt Fortran 77 as an Undergraduate and rather than learning C or C++ just used Fortran 77 to speed up R.

* 1994 - mid 2019 Wand uses Fortran 77 Speed-ups in several papers and 3 R packages "published" on CRAN (kernSmooth, glmmEP, gammSlice).

* July 2019 Wand attends the Joint Statistical Meetings in Denver and browses some of the latest books in the book stalls. One is "Advanced R" by H. Wickham. It has a chapter on Rcpp. This sounds like a good topic for the upcoming Spring 2019 Frontiers+Tea.

* Early-August 2019 Wand finishes Fortran 77 programming for the Menictas, Di Credico & Wand paper - but gets tired of dealing with the quirks of Fortran and an argument number restriction (≤ 65) in the .Fortran() R function.

* September - November 2019

The Rcpp Package Discussion Group is run fortnightly during the Frontierst Tea Forum - with several toy problems such as the "Yak Milk" problem. Weichang Yu (Uni. Syd.) is a great help.

A web-page is established and is still there at:

http://matt-wand.utsacademics.info/RCPPDG.html

(link just added from Wand personal web-site).

By the end of the semester we figured out how do solve various toy problems and create R packages with Rcpp functions.

First half of 2020

Wand starts using Rcpp in papers and a new R package tied to one of these papers. The Nolan, Menictas & Wand paper just got through the "Journal of Machine Learning Research" review process thanks to Rcpp.

An R package is about to be submitted to CRAN (with Wand as sole author) with Rcpp code for speeding up a slice sampling scheme.

A LESS-TOY EXAMPLE: GIBBS SAMPLING FOR BAYESIAN INFERENCE IN A MULTIVARIATE NORMAL RANDOM SAMPLE MODEL

Consider the model:

$$X_1, \ldots, X_n \mid \mu, \Sigma \overset{\text{ind.}}{\sim} N(\mu, \Sigma),$$

$$\mu \sim N(0, \sigma_\mu^2 I)$$

$$\Sigma^{-1} \sim \text{Wishart}(A, B)$$

If G is the total number of samples including the warm-up (or burnin) we have the loop:

Initialise $(\Sigma^{-1})^{[0]}$

For $g = 1, \ldots, G$

$\mu^{[g]} \sim$ draw from a Multiv. Normal dist^{n} that depends on $(\Sigma^{-1})^{[g-1]}$

$(\Sigma^{-1})^{[g]} \sim$ draw from a Wishart dist^{n} that depends on $\mu^{[g]}$.

End For

DEMO HERE

Rcpp implementation benefited from:

[I] RcppArmadillo data structures such as cube for storing the $(\Sigma^{-1})^{[g]}$, $1 \leq g \leq G$, matrices. This is a proxy for the list structure commonly used in R for the same purpose.

II A web-page titled

"COMMON OPERATIONS WITH RCPPARMADILLO"

posted by a Univ. of Illinois doctoral student with pseudonym "The Coatless Professor". A link for this web-page has been added to the discuss^n group web-page.

III An R package named RcppDist which supports draws from Multiv. Normal and Wishart dist^ns

PARTIAL ASIDE:

Everything in "RcppLand"

involving

* Gamma distributions
* Chi-Square distributions
* Wishart distributions

Seems to use the

scale parameters

and never

rate parameters

[In the Gamma case $\text{scale} = \frac{1}{\text{rate}}$]

Gamma reminder:

$$p(x) \propto x^{\text{shape}-1} e^{-x/\text{scale}}$$

$$= x^{\text{shape}-1} e^{-(\text{rate})x}, \quad x > 0$$

PRINTING FROM INSIDE Rcpp

Figuring this out was crucial and Wand wouldn't have been able to implement fancier & fancies models with debugging via printing Rcpp objects.

To print the object "x" inside a *.cpp file insert the line:

```
Rcout << x << "\n";
```

SOME FINAL COMMENTS

(A) Wand now very unlikely to use Fortran 77 again - and reasonably confident of solving new "speed up" problems with Rcpp and its add-ons such as RcppArmadillo.

(B) Web searches have proven useful for figuring out new challenges

e.g. "how do I sample from the Wishart distribution in Rcpp?"

(C) PACKAGING

Almost every Rcpp thing that Wand has done involves having all code in an R package.

If you use UNIX then I can share my scripts for package construction and maintenance.

If you don't use UNIX and use RStudio then there are notes from November 2019 by James Yu on the discussion group web page on packaging.

(D) Pointers. Wand still knows nothing about "pointers" but has still managed to start using Rcpp for successful research outcomes. Discussion question (if time): can one get by without knowing about pointers?