RcppOctave: Seamless Interface to Octave – and Matlab

Renaud Gaujoux

RcppOctave version 0.6.5 as of January 26, 2012

The RcppOctave package provides a direct interface to Octave from R. It allows to call Octave functions in a similar way as calling C/C++/Fortran functions using the R core function .Call. Since Octave uses a language that is mostly compatible with Matlab, the RcppOctave package may be used to run Matlab m-files. As a matter of fact, it was originally developed to facilitates the port and comparison of R and Matlab code.

The package provides modules that port R core random number generators into Octave, that enable to reproduce and compare stochastic computations. These may also be used in any Octave session, independently of R, increasing the range of RNGs available in Octave.

1 Sample session

The RcppOctave package provides the function .CallOctave to call Octave functions:

```
> .CallOctave('eye', 3)
     [,1] [,2] [,3]
[1,]    1    0    0
[2,]    0    1    0
[3,]    0    0    1
> .CallOctave('svd', matrix(1:9,3))
              [,1]
[1,] 1.684810e+01
[2,] 1.068370e+00
[3,] 5.543107e-16
```

A convenient shortcut interface in defined by the object .O of class Octave, exported from RcppOctave namespace:

```
> .O$eye(3)
     [,1] [,2] [,3]
[1,]    1    0    0
[2,]    0    1    0
[3,]    0    0    1
> .O$svd(matrix(1:9,3))
              [,1]
[1,] 1.684810e+01
[2,] 1.068370e+00
[3,] 5.543107e-16
```

Comparing equivalent R and Octave functions is therefore as easy as comparing two R functions. For example, one can compare the respective functions svd with the following code:

```
> # define random data
> X <- matrix(runif(25), 5)
> # run SVD in R
> svd.R <- svd(X)
> # run SVD in Octave
> svd.O <- .O$svd(X)
> svd.O
           [,1]
[1,] 2.79024461
[2,] 1.00811191
[3,] 0.72941335
[4,] 0.28399920
[5,] 0.03464908
> all.equal(svd.R$d, as.numeric(svd.O))
[1] TRUE
> # but not exactly identical
> all.equal(svd.R$d, as.numeric(svd.O), tol=10^-16)
[1] "Mean relative difference: 2.763291e-16"
```

We notice here that Octave default `svd` returns only the eigen values as a column vector. This is documented in its documentation that is accessible via the function `o_help`, which will show it in a similar way as R documentation:

```
> # show Octave help for svd
> o_help(svd)
```

The documentation for – Octave – `svd` states that the complete decomposition is returned, if three output values are provided. This can be done using argument `argout`:

```
> # get full output from Octave svd
> .O$svd(X, argout=3)
[[1]]
           [,1]       [,2]       [,3]         [,4]       [,5]
[1,] -0.3106631  0.1954560 -0.1901423  0.799629818 -0.4355726
[2,] -0.4966938 -0.7504205 -0.4051016 -0.018521830  0.1603558
[3,] -0.4589905  0.4872312 -0.0528055  0.090653849  0.7354773
[4,] -0.4765634  0.3600661 -0.2457396 -0.593307742 -0.4804557
[5,] -0.4679730 -0.1778320  0.8582320  0.004110792 -0.1131281

[[2]]
         [,1]     [,2]      [,3]      [,4]       [,5]
[1,] 2.790245 0.000000 0.0000000 0.0000000 0.00000000
[2,] 0.000000 1.008112 0.0000000 0.0000000 0.00000000
[3,] 0.000000 0.000000 0.7294133 0.0000000 0.00000000
[4,] 0.000000 0.000000 0.0000000 0.2839992 0.00000000
[5,] 0.000000 0.000000 0.0000000 0.0000000 0.03464908

[[3]]
           [,1]       [,2]          [,3]       [,4]       [,5]
[1,] -0.5873724  0.2405447  3.371231e-01  0.5496070  0.4259248
[2,] -0.5098616  0.5831797 -8.830131e-05 -0.4986171 -0.3890034
[3,] -0.2605862 -0.2139173 -3.338201e-01  0.5515637 -0.6860582
[4,] -0.4043823 -0.2407562 -7.444146e-01 -0.2021471  0.4283630
[5,] -0.4044806 -0.7059147  4.698500e-01 -0.3228423 -0.1144277
```