STAT40780 Data Programming with C (online)

Lab Sheet 7

Dr Marie Galligan

Summer 2015

This week's lab requires you to write some C++ functions making use of the Rcpp and inline R packages. There are 3 sections in this lab sheet.

1 A simple function with Rcpp

Write a C++ function (inlined in an R script file) with Rcpp that accepts two integer scalar arguments passed from R, and returns the first value, divided by the second value. Compile, link, and load this function into R using `cxxfunction()`, then call the function from R.

2 Practice with `IntegerVector`

With the help of Rcpp, write a C++ version of the which.min() function in R, that accepts as input an integer vector from R. Note: you can assume for now that no missing values will be passed to this function.

If you are unsure what the `which.min()` function does, you can get help on this in R by running

```
?which.min
```

3 Bubblesort function

The C++ code inlined in the R script below is the implementation of the bubblesort algorithm from Lesson 4.2, which sorts an `IntegerVector` passed from R. Improve this function as follows:

1. On pass k over the vector, the top k-1 elements are already sorted. Modify the bubblesort function so that on pass k, it does not pass over the top k-1 elements.

2. A vector with n elements might be sorted in less than n-1 passes. Improve the bubblesort algorithm so that it does not continue to pass over a vector that is already sorted.

Benchmark your modified version against the original (shown here), to compare performance.

Note that it is possible to optimize the bubblesort function further - explain how.

Bubblesort algorithm

```
#body of the C++ bubblesort function
#stored in an R character string
body_bubblesort2 <- '
 IntegerVector xx = clone(x); //use of clone()
 int n = xx.size(); //no. of elements
 int temp; //temporary storage of swap value
 for( int k = 1; k <= n - 1; k++ ){ //for pass k
  //loop over pairs of elements
  for( int i = 0; i < n - 1; i ++ ){
    if( xx[ i ] > xx[ i+1 ] ){
      temp = xx[ i + 1 ];
      xx[ i + 1 ] = xx[ i ];
      xx[ i ] = temp;
    } //end of if
  } //end of loop over array pairs
 } //end of loop over passes
 return(wrap(xx));
'

#compile, link, load
bubblesort2 <- cxxfunction(signature( x = "integer" ),
                           body = body_bubblesort2,
                           plugin = "Rcpp")

#create an R integer vector to input to bubblesort
x2 <- as.integer( sample(1:100, size = 100, replace = FALSE) )

#call
bubblesort2(x2) #returns sorted x2
x2 #original x2 is not sorted
```