# Trabajo Computacion Estadistica III

Rudy Miranda

today

## Table Of Contents

## Ejercicio 1

***.C***

***.Call***

***.External***

## Ejercicio 2

Mediante los distintos metodos obtendremos la suma de los cuadrados al cuadrado mediante un ciclo *for* con la ayuda de una variable temporal llamada *aux*.

$$aux = \sum_{i=0}^{n} x_i^2$$

### Solo con *R*

```
squares_sum <- function(v) sum(v * v)
```

### Con *C*

Todas los metodos funcionan mediante 3 archivos (reducible a 2)

1. *fun.R*, funcion *R* que llama a uno de los tres metodos y recibe los argumentos que seran pasados a la funcion de *C*.
2. *fun.C*, Ejecuta el programa y puede retornar un valor.
3. *test.R*, ejecuta un ejemplo luego de haber cargado el script de *C* con la funcion *dyn.load()* y el de *R* con *source()*.

### Metodo 1: *.C*

#### fun.R

```
sumsquares <- function(x){
    .C("SumSquares",
    as.double(x),
    as.integer(length(x)),
    aux = as.double(0))$aux
}
```

#### fun.c

```
#include <R.h>

void SumSquares(double *x, int *nx, double *aux){
    for(int i = 0; i < *nx; i++)
       *aux += x[i] * x[i];
}
```

```
R CMD SHLIB fun.c
```

**test.R**

```
source("fun.R")
dyn.load("fun.so")

sumsquares(1:5)
```

### Metodo 2: *.Call*

**fun.R**

```
sumsquaresv2 <- function(x){
    .Call("SumSquares",
    as.double(x))
}
```

**fun.c**

```
#include <R.h>
#include <Rinternals.h>

SEXP SumSquares(SEXP x){
    SEXP aux;
    PROTECT(aux = allocVector(REALSXP, 1));
    REAL(aux)[0] = 0.0;
    for(int i = 0; i < length(x); i++)
        REAL(aux)[0] += REAL(x)[i] * REAL(x)[i];
    UNPROTECT(1);
    return aux;
}
```

```
R CMD SHLIB fun.c
```

**test.R**

```
source("fun.R")
dyn.load("fun.so")

sumsquaresv2(1:5)
```

### Metodo 3: *.External*

**fun.R**

```r
sumsquaresv3 <- function(...){
    .External("sumsquaresv3", ...)
}
```

**fun.c**

```c
#include <R.h>
#include <Rinternals.h>

SEXP sumsquaresv3(SEXP x){
    SEXP aux, now;
    PROTECT(aux = allocVector(REALSXP, 1));
    REAL(aux)[0] = 0.0;
    x = CDR(x);
    now = CAR(x);
    while(x != R_NilValue){
        x = CDR(x);
        REAL(aux)[0] += REAL(now)[0] * REAL(now)[0];
        now = CAR(x);
    }
    UNPROTECT(1);
    return aux;
}
```

```
R CMD SHLIB fun.c
```

**test.R**

```r
source("fun.R")
dyn.load("fun.so")

sumsquaresv3(1, 2, 3, 4, 5)
```

## Ejercicio 3

## Ejercicio 4