

# Práctica 3 de laboratorio de R

Curso: Introducción a la Estadística y Probabilidades CM-274

## Lecturas Importantes

1. Una guía de ciencia de datos con R.

<http://www.analyticsvidhya.com/blog/2016/02/complete-tutorial-learn-data-science-scratch/>.

---

## Preguntas

1. Sea una definición de la raíz cuadrada, definida por el método de Newton

```
> r =  
+   function(x, eps = 1e-10) {  
+     g = 1  
+     while(abs(1 - g^2/x) > eps)  
+       g = .5 * (g + x/g)  
+     g  
+   }
```

pero esta función trabaja para escalares, pero no cuando se pasa un vector

```
>r(c(1,2))  
[1] 1  
Warning message:  
In while (abs(1 - g^2/x) > eps) g = 0.5 * (g + x/g) :  
the condition has length > 1 and only the first element will be used
```

Una manera de resolver este problema es dividir el cálculo en dos partes; una para calcular la raíz cuadrada para valores escalares y la otra usa un bucle sobre un vector

```
> s.r =  
+   function(x, eps = 1e-10) {  
+     g = 1  
+     while(abs(1 - g^2/x) > eps)  
+       g = .5 * (g + x/g)  
+     g  
+   }
```

```
> r =  
+   function(x, eps = 1e-10) {  
+     ans = numeric(length(x))  
+     for(i in seq(along = x))  
+       ans[i] = s.r(x[i])  
+     ans  
+   }
```

Calculando ahora los valores

```
> r(c(1,2))
```

La estrategia de usar bucles funciona, pero tienden a ser ineficientes debido a los cálculos que se llevan a cabo elemento a elemento. Una estrategia alternativa es llevar a cabo el cálculo de vectores en lugar de escalares. En este caso particular, podemos cambiar el cálculo para que funcione con vectores como sigue

- Cambia la inicialización de  $g$  de forma que sea un vector y no un escalar

```
g = rep(1, length = length(x))
```

- Cambiar la prueba para que los cambios de  $g$  continúen hasta que todos los elementos de la respuesta se hayan calculado con una suficiente precisión.

```
while(any(abs(1 - g^2/x)) > eps))
```

Esto continúa mejorando las aproximaciones de la raíz cuadrada hasta que todos ellos han alcanzado el nivel de exactitud. Lleva esto a la práctica implementando los cambios en  $r$  y probando la función resultante.

La estrategia de la sección anterior conlleva el cálculo de la raíz cuadrada incluso después de que las raíces cuadradas se han determinado para elementos de  $x$ . Estos cálculos adicionales pueden evitarse manteniendo un registro de los elementos de  $x$  cuyas raíces no se calculan con la precisión suficiente y sólo realizando los cálculos para esos elementos.

```
n.d = abs(1 - g^2/x) > eps
```

Esto puede ser hecho como parte de la prueba

```
while(any((n.d = abs(1 - g^2/x))) > eps))
```

Dentro del bucle, los cambios pueden llevarse a cabo sólo en el subconjunto de  $g$  que necesita ser actualizado, es decir,  $g[n.d]$ . Las actualizaciones se llevan a cabo utilizando sólo los elementos correspondientes de  $g$  y  $x$ , es decir,  $g[n.d]$  y  $x[n.d]$ .

## 2. El siguiente programa que produce?

```
> f1 <- function(x ,k){  
+   n <- length(x)  
+   r <- NULL  
+   for(i in 1:(n -k)){  
+     if(all(x[i:i + k -1]==1))r <- c(r, i)  
+   }  
+   return(r)  
+ }
```

- (a) Si realizamos un test

```
> f1(c(1,0,0,1,1, 0, 1,1,1), 2)
```

y produce los valores 3467. Es correcto el resultado?.

- (b) Utiliza la función `debug()` para utilizar `browse` y mostrar

- Si el vector fue recibido correctamente
- cuando colocamos  $n$  dos veces en `browse`. Qué sucede cuando colocamos  $n$  tres veces en `browse`. Explica.
- Si  $k = 2$ , que significa y que produce lo siguiente  
`Browse[2] > x[i:i + k- 1]`

```

Browse[2] > i:i + k- 1
Browse[2] > i
Browse[2] > k

```

donde se encuentra el error, del código inicial, si es que existe?.

3. Escribimos dos funciones primero y ultimo, que extrae un número específico de elementos desde el inicio y el final de un vector (en el orden que aparecen en el vector). Las funciones deben ser llamadas como siguen

```

primero(x , k)
ultimo(x, k)

```

donde  $x$  es el vector de valores que son extraidos y  $k$  especifica el número de elementos a extraer. Si el argumento  $k$  es omitido en una de las llamadas, debe tomar por valor por defecto 1.

- (a) Asumiendo que  $k \leq \text{length}(x)$ , escribimos versiones (lo más simples) de las funciones dadas anteriormente.
  - (b) Modifica las funciones (a) de manera que si  $k > \text{length}(x)$  entonces estas funciones deberían retornar los valores en  $x$ .
  - (c) Modifica las funciones (a) de manera que si  $k > \text{length}(x)$  las funciones retornan los  $k$  valores, si no hay valores existentes estos deben ser NA.
4. Para  $n > 2$ , la densidad chi-cuadrado tiene un máximo valor. Escribe código R, que usa la función optimise para localizar el máximo de la densidad para un valor  $n > 2$ .
  5. El siguiente código define una función para la clase palette

```

> palette <- function(r, g, b, max=1) {
+   p <- list(colours=cbind(r, g, b), max=max)
+   class(p) <- "palette"
+   p
+ }

```

Ejemplos de la función son mostrados a continuación

```

> palette(1, 0, 0)

$colours
   r g b
[1,] 1 0 0

$max
[1] 1

attr(,"class")
[1] "palette"

```

```

> palette(0:3, 0, 0, max=3)

$colours
   r g b
[1,] 0 0 0
[2,] 1 0 0
[3,] 2 0 0

```

```
[4,] 3 0 0

$max
[1] 3

attr("class")
[1] "palette"
```

- (a) Escribe una función que imprima los colores usando la función `rgb()`. Por ejemplo

```
palette(1, 0, 0)
[1] "#FF0000"

palette(0:3, 0, 0, max=3)
[1] "#000000" "#550000" "#AA0000" "#FF0000"
```

- (b) Escribe una función que retorne un objeto conteniendo los colores seleccionados. Por ejemplo

```
palette(0:3, 0, 0, max=3)[1]
[1] "#000000"

palette(0:3, 0, 0, max=3)[2:3]
[1] "#550000" "#AA0000"
```

El resultado de esta función es un objeto de `palette` que está siendo impreso por la función anterior.

6. (a) ¿Qué produce los siguientes códigos y las propiedades que muestran

```
> f1 <- function(x = {y <- 1; 2}, y = 0) {
+   x + y
+ }
> f1()
```

- (b) ¿Por qué las siguientes dos invocaciones de `lapply` son equivalentes?

```
> trims <- c(0, 0.1, 0.2, 0.5)
> x <- rcauchy(100)
>
> lapply(trims, function(trim) mean(x, trim = trim))
> lapply(trims, mean, x = x)
```

- (c) Considera el siguiente problema : Dado una matriz numérica  $X$ , determina el índice de la primera fila de números positivos que no contiene  $NA$ . Resuelve el problema usando `for` y la función `apply()`.
- (d) ¿Cómo se determina el entorno desde el que se llama una función?