PRÁCTICA 4 : Definición de funciones y representación gráfica aplicando condicionales “if”, “while”

Este recurso es el resúmen de la cuarta práctica del curso, donde aprenderemos a representar las funciones de forma más definida y aplicando estructuras condicionales. Dicha práctica fue impartida por el profesor Arturo Hidalgo durante los días 18, 19 y 20 de octubre. Es muy recomendable que tengáis claro los conceptos básicos sobre representación de gráficas (práctica 3) antes de empezar con este recurso.

Tendréis todos los comandos que necesitaréis saber para representar una función o más de una en una misma gráfica, para cambiar el aspectos de la función o la escala numérica en caso de tener más de una función y para aplicar bucles “while” y “if”. También dejaremos el ejemplo de la práctica resuelto.

1. Comandos relevantes:

- Para generar un nº “x” de valores entre [a,b] usamos el comando: **=seq (a, b, paso[1])** o **=seq (a, b, length=x)**

Ej: Siendo x= 20, a=0 y b=10 y T el nombre de la lista:

```
> T=seq(0,10,length=20)

> T

 [1]  0.0000000  0.5263158  1.0526316  1.5789474  2.1052632
2.6315789

 [7]  3.1578947  3.6842105  4.2105263  4.7368421  5.2631579
5.7894737

[13]  6.3157895  6.8421053  7.3684211  7.8947368  8.4210526
8.9473684

[19]  9.4736842 10.0000000
```

2. Cómo definir funciones:

- Para nombrar la función que vamos a crear escribimos su nombre, comando de igualdad (=/ <-) y “function (-variable que queramos -)”. A continuación se escribe entre claudators la ecuación a la que es igual.

Ej: Siendo “f” el nombre de la función y “x” nuestra variable[2]:

---

[1] Se define como la variación entre cada uno de los elementos del conjunto [a,b], que es constante y se calcula como: paso=(b-a)/(x-1), siendo x el número de elementos total que queremos obtener

[2] En vez de llamar “x” a la variable se puede poner cualquier nombre como "pepín" o “peponcio”

```
f<-function(x){
 x*cos(x^2)
}
```

® Si a continuación escribimos “f”, en la consola, nos devuelve el texto de la imagen anterior, ya que la función solamente actúa cuando la llamamos por su nombre y le mandamos los datos de entrada, por ejemplo:

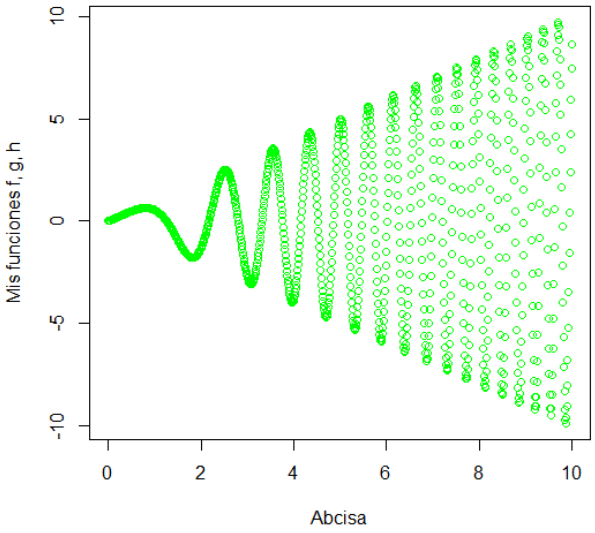
```
> f(pi/4)
[1] 0.6406528
```

® Para dibujar la gráfica de una función usamos el comando **plot(valor de x, valor de y, xlab= “Texto en el eje de las x”, ylab= “Texto en el eje de las y”, col= “color”) par(new= “true”)**[3]

Ej: Siendo “xx” el nombre de la lista que va a dar los valores a x, “f(xx)” la función[4] que va a dar los valores a y, “Abcisas” el texto en xlab=, “Mis funciones f, g, h” el texto en ylab= y “green” el texto en col=:

```
plot(xx,f(xx),xlab='Abcisa', ylab='Mis funciones f, g, h',
col='green')
par(new=TRUE)
```

Obtenemos el siguiente gráfico:



® Para combinar varias funciones en un mismo gráfico volvemos a escribir todo el comando **plot(** anterior pero escribiendo solo unas comillas en el texto de los ejes: **xlab= “”, ylab= “”** ,

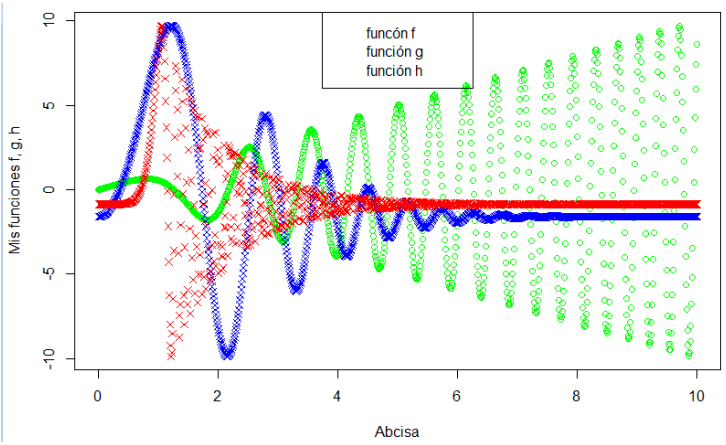
[3] Este operador sirve para que puedan haber varias funciones superpuestas en la misma gráfica
[4] Tanto la lista como la función deben estar previamente guardadas en el programa

y los comandos **axes=FALSE** para que no se solapen distintos textos de los ejes o los propios ejes:

Ej de varias funciones, añadiendo algunos operadores más como **legend(x= "ubicación"[5], c(valor 1, valor 2; ...)** :

```
plot(xx,f(xx),xlab='Abcisa', ylab='Mis funciones f, g, h',
col='green')

par(new=TRUE)

plot(xx,g(xx),xlab='', ylab='', col='blue', axes=FALSE, pch=4)

par(new=TRUE)

plot(xx,h(xx),xlab='', ylab='', col='red', axes=FALSE, pch=4)

legend(x='top',c('funcón f', 'función g', 'función h'))
```

Gráfico:



3. Aplicación de bucles "while" y "if"

® Una aplicación del bucle "while" es hacer una operación hasta que ocurra la condición, como puede ser que un número "X" sea igual al número de sumandos de la operación, por lo que se acaba el bucle al acabarse los sumandos.

Ej: Siendo "v" el vector (1,3.4,exp(3),log(6)), "length(v)" número de sumandos del vector "v", "i" el número que va a ir marcando los sumandos que ya se han operado, "sumaV" el sumatorio de las componentes de "v":

[5] Las posibles ubicaciones son: "bottomright", "bottom", "bottomleft", "left", "topright", "top", "topleft", "right", "left"

```
v=c(1,3.4,exp(3),log(6))
i=1; sumaV<-0
while (i<=length(v)){
 sumaV = sumaV+v[i]
 i=i+1
}
sumaV
```

"i" va aumentando en una unidad cada vez que se hace la operación de el valor de "sumaV", por lo que cuando se acaben los sumandos "i" valdrá el número de sumandos, "length(v)", y se acabará el bucle.

4. Ejercicio del medicamento

N(t) = 10exp(t/20) es la evolución de células infectadas dependiendo del tiempo. Dependiendo de N se pueden dar diferentes situaciones:

- Si N < 5000, enfermo recuperable sin medicación. EN VERDE
- Si N >= 5000 y N <= 10000, medicación moderada. EN NARANJA
- Si N > 20000, medicación agresiva. EN ROJO
- En otro caso se trata de una situación intermedia. EN AZUL

Se hace un estudio de 168h, calculando la evolución de células cada media hora (dt = 0.5).

En el vector "xd" se guardan los valores del tiempo, en el vector "yd" el número de células infectadas y en el vector "color" los colores de cada situación.

REPRESENTAR ESTA GRÁFICA CON R:

1. Introducimos los vectores vacíos y datos que sabemos.

```
N<-function(t){
 10*exp(t/20)
}
i=1; xd=c(0); yd=0; color=0; tfin=168; dt=0.5; t=0
```

2. Con un bucle "while" analizamos las situaciones posibles en los 168h.
    - lo que está al final quiere decir que "i" aumenta 1 cada vez que "t" aumenta 0.5 hasta llegar a 168.
    - lo del medio son las condicionales para asignar el color.

```
while(t<=tfin){
 xd[i]=t; yd[i]= N(t)
 if (N(t)<5000){
  color[i]='green'
 }else if (N(t)>=5000 & N(t)<=10000){
  color[i]='orange'
 }else if (N(t)>=20000){
  color[i]='red'
 }else{
  color[i]='blue'
 }
 t=t+dt; i=i+1
}
```

IMPORTANTE: Seguir la misma estructura que aparece de cerrar una condicional "}" en otra línea y seguir ahí con otra condicional. No es solo más ordenado visualmente sino que es esencial para que funcione.

3. Representamos la gráfica con plot. tipo "h" para el histograma. El vector N cogerá su color correspondiente ya que "yd" y "color" cambian a la vez a medida que avanza el tiempo "xd". (R no es capaz de representar tilde

```
plot(xd,yd,type='h', col=color, xlab='Tiempo (h)', ylab='Num c◆lulas')
```