Biotecnología Programación 2023-2024 Dificultad: fácil
1º curso Grupo M10

PRÁCTICAS 2 Y 3

En este documento hay una guía sobre las prácticas 2 y 3. Vamos a ir resolviendo apartado por apartado haciendo algunas aclaraciones.

REPRESENTACIÓN GRÁFICA DE FUNCIONES

Enunciado: Consideramos la producción de manzanas durante los 10 primeros meses del año.

1	2	3	4	5	6	7	8	9	10
100	90.50	45.23	68.14	30.60	25.55	12.01	16.48	32.00	87.10

APARTADO 1: **Construir un vector meses que contenga los números de 1 a 10 asignados a los meses. Emplear para ello la instrucción: seq(valor inicial,valor final,incremento).**
meses=seq(1,10,1)

APARTADO 2: **Construir un vector llamado manzanas que contenga la producción mensual de manzanas, a partir de la tabla dada.**
manzanas= c(100, 90.50, 45.23, 68.14, 30.60, 25.55, 12.01, 16.48, 32.00, 87.10)

como nos pide crear un VECTOR, no se nos puede olvidar poner la c

APARTADO 3: Representar gráficamente la producción mensual de manzanas. Usando línea continua y color azul:

Antes de resolver este apartado, es conveniente comentar esta nueva expresión que aparece aquí. “Plot”, que se usa cuando queremos representar puntos en un gráfico bidimensional.

```
plot(meses,manzanas,type='b', col='blue')
```

eje de abscisas | eje de ordenadas | tipo | color (escribirlo en inglés)

¡FIJARSE! Tanto después de type, como de col hay que poner la palabra (o letra) entre ‘ ‘ ,sino dará error.

Como **dato interesante**, para posibles ejercicios posteriores, si queréis investigar sobre otros tipos de líneas poner en el propio programa R ?plot y os saldrá mucha información al respecto.

APARTADO 4: Etiquetar los ejes como: 'Meses del año' (abscisas), 'Toneladas de manzanas' (ordenadas). Se usará xlab=' ' , ylab=' ' para tal fin.

Lo único que hay que hacer es añadir DENTRO DEL PARÉNTESIS de plot estas instrucciones para poner etiquetas a los ejes.

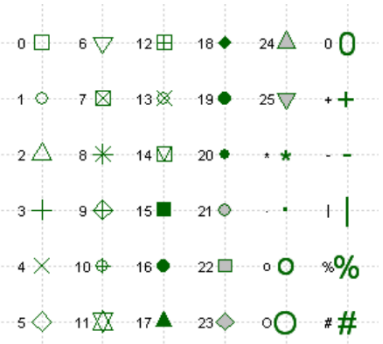
```
plot(meses,manzanas,type='b', col='blue', xlab='meses del año', ylab='toneladas de manzanas')
```

etiqueta eje de abscisas

etiqueta eje de ordenadas

APARTADO 5: Utilizar pch=número (entre 0 y 25) para cambiar símbolos.

Al igual que antes, tenemos que añadir esto nuevo dentro del paréntesis. Aquí abajo os dejo una tabla con los posibles símbolos que podéis poner.




```
plot(meses,manzanas,type='b', col='blue', xlab='meses del año', ylab='toneladas de manzanas', pch=4)
```

yo he utilizado este símbolo pero podéis usar el que más os guste

APARTADO 6: Poner título al gráfico. Se usará la instrucción main=' '.

Por último, tenemos que añadir dentro del paréntesis main=' ', para poner un título al gráfico (el que vosotros queráis).

```
plot(meses,manzanas,type='b', col='blue', xlab='meses del año', ylab='toneladas de manzanas', main='producción de manzanas')
```

Bucles

Para la elaboración de bucles en R es necesario seguir la siguiente estructura:

```
for (i in vinic:vfin)
        {Proceso de cálculo
}
```

Enunciado: Dados los vectores: v=(12,-3,5,18.7) y w=(12,0.25,77,exp(2))

ANTES DE EMPEZAR EL EJERCICIO:

Hay que tener en cuenta que es necesario escribir de la forma adecuada los vectores, es decir, colocar un c delante de los paréntesis:

```
v=c(12,-3,5,18.7)
w=c(12,0.25,77,exp(2))
```

APARTADO 1 Obtener la suma de los dos vectores mediante bucles y comprobar empleando v+w.

```
S=c(0)
for (i in 1:length(v)){
    S[i]=v [i]+ w [i]
}
S
```

Comprobación:

```
Suma= v+ w
Suma
```

APARTADO 2 Obtener la suma de las componentes del vector v y almacenarlos en SumaC.

```
SumaC= 0
for(i  in 1 length(v)){
    Sumac=SumaC+v[i]
Sumac
```

APARTADO 3 Realizar el producto escalar de ambos vectores mediante bucles y DESPUÉS comprobar empleando %*%.

```
p=0
for(i in 1:length(v)){
   p=p+v[i] * w[i]
```

```
}
p
```
Comprobación:
```
p=v%*%w
p
```

APARTADO 4 Multiplicar ambos vectores componente a componente mediante bucles

```
t=1
for(i in 1:length(v)){
    t[i]=c(v[i]*w[i])
}
t
```

APARTADO 5 Realizar la operación: zj=vj+2wj , j = 1,...,length(v) mediante bucles

```
z=c(0)
for(j in 1:length(v)){
    z[j]=v[j]+2*w[j]
}
z
```

APARTADO 6 Construir una tabla, data.frame que contenga:

En R, la estructura 'data.frame' se utiliza para organizar datos en filas y columnas, similar a una tabla en una base de datos o una hoja de cálculo.

Para la elaboración de tablas con data frame es necesario crear vectores con los nombres y datos que queramos poner en la tabla.

Después se escribe la siguiente estructura incluyendo los vectores creados:

`Suma´	Resultado del apartado 2
`Producto escalar´	Resultado del apartado 3

```
nombres=c('Suma','Producto escalar')
valores=c(Sumac, p)
data.frame(nombres, valores)
```

Resultado:

	nombres	valores
1	Suma	18.7000
2	Producto escalar	666.4253

BUCLES ANIDADOS

Para crear bucles anidados en R hay que tener perfectamente clara la estructura de los bucles, explicada en el ejercicio anterior. Deberemos seguir la siguiente estructura para elaborar bucles anidados en el problema:

```
for  (i  in  vinic1:vfin1) {
          for (j in  vinic2:vifin2){
               for (k in vinic3:vfin3){
                    Proceso de cálculo
               }
          }
}
```

Ejercicio:

APARTADO 1: Construir una matriz A1 de manera que los vectores v,w sean sus filas.

Para crear una matriz a partir de vectores, es importante definir anteriormente los vectores (v=c(x,y,z); w=c(k,l,m); por ejemplo). Posteriormente, podremos crear la matriz:

```
A1=rbind(v,w)
```

OJO! Ponemos 'rbind' porque nos piden que los vectores sean las filas. Si nos hubiesen pedido que fuesen las columnas, escribiríamos 'cbind'.

APARTADO 2: Construir una matriz A2 de manera que los vectores v,w sean sus columnas.

Como hemos explicado justo en el apartado anterior, definimos los vectores y después crearemos la matriz, utilizando 'cbind':

```
A2=cbind(v,w)
```

APARTADO 3: Multiplicar, empleando bucles, ambas matrices, obteniendo una matriz C.

Para multiplicar una matriz empleando bucles deberemos utilizar bucles anidados de la siguiente manera:

```
CC=matrix(c(0),nrow=nrow(A1),ncol=ncol(A2))
for(i in 1:nrow(A1)){
  for(j in 1:ncol(A2)){
      for(k in 1:ncol(A1)){
         CC[i,j]=CC[i,j]+ A1[i,k]*A2[k,j]
      }
  }
}
CC
```

APARTADO 4: Verificar el resultado obtenido previamente empleando %*%

Para multiplicar matrices podemos utilizar '%*%', como se nos ha proporcionado en el enunciado y solamente es poner el nombre de las dos matrices a un lado y otro del símbolo y darle un nombre a la matriz resultante de la multiplicación (C por el apartado anterior):

```
C=A1%*%A2
```

OJO! Hay que tener cuidado con las dimensiones de las matrices pues a veces no se pueden multiplicar (tener en cuenta en el apartado 3 también).

APARTADO 5: Inventar una matriz 2x2 y llamarla D

Tenemos dos opciones para crear una matriz. En el apartado 1 y 2 hemos utilizado vectores que han conformado sus filas y columnas. Otra manera de crearlas sería de la siguiente forma:

```
D=matrix(c(1,6,7,0.8),nrow=2,ncol=2)
```

Se utiliza la expresión 'matrix()' dentro de la cual añadiremos un vector con los diferentes componentes que queremos que la formen y el número de filas (nrow) y de columnas (ncol), que determinarán la dimensión de la matriz resultante.

FUNCIONES EN R

Para definir funciones en R, hemos de utilizar la siguiente expresión en el programa:

```
Nombre de la función <-
function(argumento1,argumento2,…){
      expresión de la función
}
```

Siendo argumento 1 o 2 las variables/incógnitas de tu función

EJERCICIO:

APARTADO 1: Define la función $f(x) = x^2 \cos(x^2)$ y obtenga el valor f(pi/4)

Para definir esta función en R debemos introducir la siguiente expresión:

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

La 'f' es el nombre que hemos puesto a la función según indica el enunciado

Para obtener el valor pedido en el enunciado, solamente tenemos que poner tal cual : f(pi/4)

APARTADO 2: Obtén un vector xx que tome 1001 valores en [0,10] (utilizando el comando seq con 'salto' 0.01)

Para obtener un vector es muy importante no olvidarnos de la 'c' al crearlo; pero en este caso, en el que se nos pide crear un vector formado por puntos de un intervalo, utilizamos el comando 'seq'.

```
xx= seq(0,10, 0.01)
```

El primer dígito que debemos colocar en el paréntesis es el primer número del intervalo y el segundo dígito el último de este mismo. El tercer dígito es el salto que se nos pide. (No necesitamos indicar el número de valores pues con saltos de 0.01 en el intervalo dado, obtenemos como justo los 1001 valores.

APARTADO 3: Representa gráficamente la función f, tomando como abscisas los valores xx. Etiqueta el eje de abscisas con 'Abscisa' y el eje de ordenadas con 'mis funciones f, g, h'. La gráfica iruña en color verde.

Para representar gráficamente una función utilizamos una expresión especial explicada en el documento' Representación gráfica de funciones', dentro de la cual podemos añadir diferentes comandos para cambiar el color de la curva o añadir etiquetas a los ejes, como se nos pide en este apartado.

```
plot(f,xx, xlab='Abscisa', ylab='mis funciones f,g,h', col='verde')
```

APARTADO 4: Define la función
Para definir la función, haremos como en el primer apartado:

```
g=function(x){
      sin(x^2)*exp(10)^(-x^2/10)
}
```

APARTADO 5: Utiliza la instrucción: par(new='true') para superponer las curvas.

APARTADO 6: Dibuja la función g en los puntos xx en color azul y empleando: xlab='', ylab='', axes=FALSE (para eliminar ejes), pch=4 (para símbolos).

APARTADO 7: Idem a 5-6 con la función $h(x) = \sin(x^8)e^{-x}$ **en rojo y usando pch=18.**

APARTADO 8: Añade a continuación la leyenda:

```
legend(x = "top", c("función f", "función g", "función h"), fill = c("green",
"blue", "red"))
```

— — — — — —

APARTADOS 5, 6, 7 y 8: Estos apartados están relacionados entre sí y por ello los resolveremos juntos.
En primer lugar deberemos representar las tres funciones, que deberemos haber definido anteriormente. La función f y g ya las hemos definido, pero tenemos aún que definir h:

```
h=function(x){
      sin(x^8)*exp(10)^-x
}
```

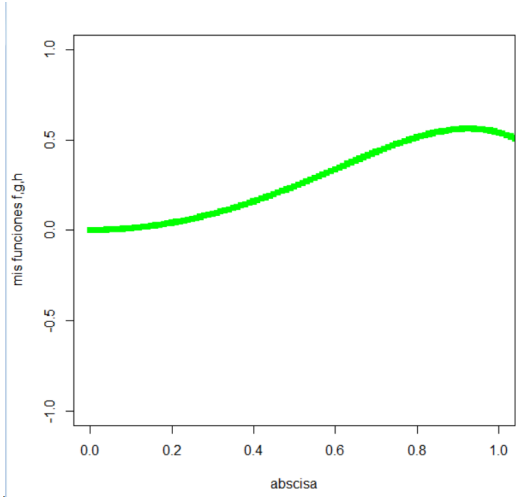
Después, representaremos las tres funciones, cada una con colores distintos indicados en la leyenda (apartado 8). Además, en dos de las funciones deberemos añadir xlab='', ylab='', axes='FALSE' para evitar que se representen varias veces los ejes y títulos de ejes.

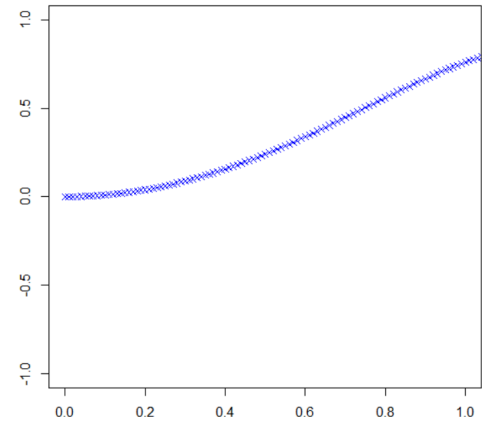
```
plot(xx,f(xx),pch=15,xlab='abscisa',ylab='mis funciones
f,g,h',col='green',type='b')
plot(xx,g(xx),pch=4,col='blue',xlab='',ylab='')
plot(xx,h(xx),pch=18,col='red',ylab='',type='b',,xlab='')
```

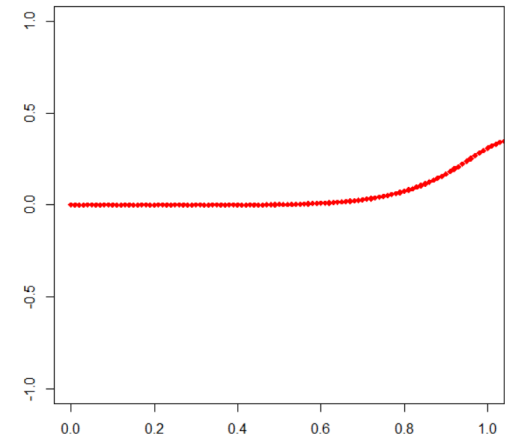
Sería recomendable introducir en las tres funciones, un límite de valores en el eje X e Y para evitar que la función nos salga extremadamente grande o pequeña, y para que las funciones se vean más claras en la superposición. Un valor recomendado sería:

xlim=c(0,1); ylim=c(-1,1)

Hasta ahora podríamos observar las gráficas por separado y quedarían tal que:







verde - f(x)
azul- g(x)
rojo- h(x)

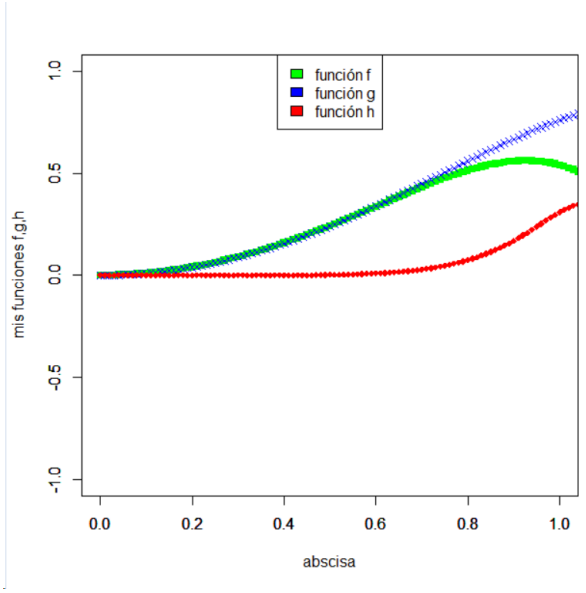
Ahora para superponer las gráficas utilizaremos la expresión proporcionada por el enunciado. Tenemos que poner esta expresión 'entre medias' de dos funciones, es decir:

```
plot(xx,f(xx),pch=15,xlab='abscisa',ylab='misfuncionesf,g,h',col='green',type='b',xlim=c(0,1),ylim=c(-1,1))
par(new='true')
plot(xx,g(xx),pch=4,col='blue',xlim=c(0,1),ylim=c(-1,1),ylab='')
par(new='true')
plot(xx,h(xx),pch=18,col='red',ylab='',type='b',xlim=c(0,1),ylim=c(-1,1),xlab='')
```

Ya conseguida la representación, vamos a elaborar la leyenda del apartado 8. Esta leyenda se forma siempre con la misma expresión, el la cual se introduce un vector con los nombres de cada función y otro vector con los colores de cada función. El orden de las funciones debe ser el mismo al de sus respectivos colores:

```
legend(x='top',c('función f','función g','función h'),fill=c('green','blue','red'))
```

La representación gráfica final quedaría así:



Al finalizar la práctica, para repasar todos los contenidos dados, deberemos realizar y entregar un ejercicio propuesto por el profesor. El ejercicio es el siguiente:

Ejercicio

APARTADO 1: Define un vector, llamado 'Nombres', con los nombres de 5 amigos o familiares.

Como en el resto de ejercicios que hemos realizado, pondremos una 'c' para definir al vector y *OJO* escribiremos '' pues vamos a introducir nombres:

```
Nombres=c('Sonia','Pepe','Guille','Rodrigo','Leire')
```

APARTADO 2: Define un vector, llamado 'Estatura', con la estatura, en metros, de esas 5 personas.

Tal y como hemos realizado el vector en el apartado anterior:

```
Estatura=c(1.63,1.80,1.85,1.75,1.70)
```

OJO los decimales se introducen con '.'

APARTADO 3: Define un vector, llamado 'Peso', con el peso en kg de esas 5 personas.

Al igual que en los apartados anteriores:

```
Peso=c(51,95,80,70,60)
```

Los datos en azul verdoso introducidos en los tres primeros apartados hace referencia a que se puede meter datos aleatorios que escoja la persona que vaya a realizar el ejercicio.

APARTADO 4: Calcula el *Índice de Masa Corporal =Peso/Estatura^2* y almacenarlo en el vector IMC.

Para calcular y almacenar el IMC en un vector, inicializamos el vector a 0 y abrimos un bucle que nos permitirá realizar la operación, de tal forma que:

```
IMC=0
for(i in 1:length(Peso)){
  IMC[i]=Peso[i]/Estatura[i]^2
}
IMC
```

APARTADO 5: Construye un data.frame que contenga los nombres, estaturas, peso e IMC de cada persona.

Este apartado nos pide crear una tabla empleando la expresión 'data.frame', dentro de la cual añadiremos los nombres de los vectores pedidos:

```
data.frame(Nombres,Estatura,Peso,IMC)
```

APARTADO 6: Genera un vector llamado 'pesos', que contenga una distribución aleatoria de pesos de 100 individuos en el intervalo [60,90]. Usa runif (100,60,90).

De nuevo, montaremos un vector que llamaremos 'pesos, pero esta vez lo crearemos con la función 'runif' que nos permite obtener un número determinado de valores dentro de un intervalo:

· Primer dígito = número datos
· Segundo dígito = apertura intervalo
· Tercer dígito = cierre intervalo

```
pesos=runif(100,60,90)
```

APARTADO 7: Genera un vector llamado 'estaturas', que contenga una distribución aleatoria de estatura de 100 individuos en el intervalo [1.60,1.90].

En este apartado volveremos a utilizar la expresión anterior, de tal manera que:

```
estaturas=runif(100,1.60,1.90)
```

APARTADO 8: Obtén un vector llamado 'imcs' que contenga el índice de masa corporal de cada individuo.

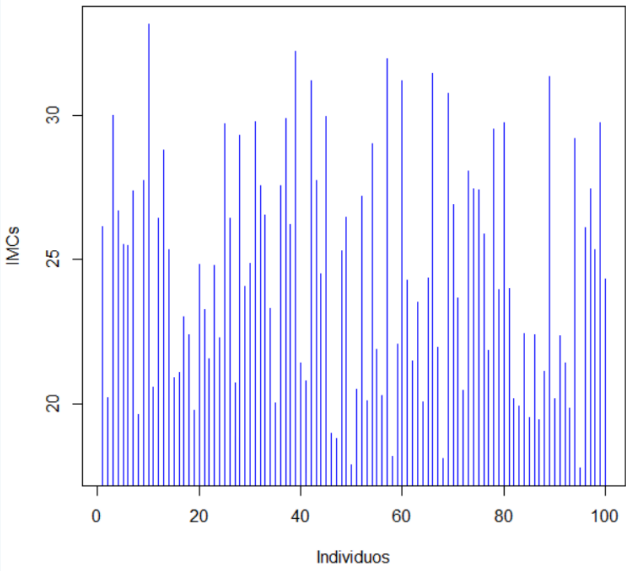
Este apartado es exactamente igual que el apartado 4. Sin embargo, en este caso no calcularemos un vector que contenga el IMC de 5 individuos, sino que de 100. Por ello, aumentaremos la 'n' del bucle:
(También inicializamos a 0)

```
imcs=0
for(i in 1:length(pesos)){
    imcs[i]=pesos[i]/estaturas[i]^2
}
imcs
```

APARTADO 9: Representa con una gráfica los imcs de cada individuo tipo histograma (type='h')

Al igual que en ejercicios anteriores, para representar una gráfica utilizaremos la siguiente expresión:

```
plot(imcs,type='h',xlab='Individuos',ylab='IMCs',col='blue',
     main='Índice Masa Corporal')
```



APARTADO 10: Calcula el imc medio de la población, señalando cuál es su situación según la tabla:

En este apartado calcularemos primero el IMC medio mediante el uso de un bucle y una operación. Con el bucle calcularemos la suma total de todos los valores de 'imcs'. Posteriormente, con una simple operación obtendremos la media.

```
K=0
for(i in 1:length(imcs)){
  K=K+imcs[i]
}
K
IMCmedio=K/length(imcs)
IMCmedio
```

En segundo lugar, según el valor obtenido en la media, deberemos indicar el rango en el que se encuentra ese dato. Para ello, tenemos que crear una estructura anidada bastante grande con una secuencia condicional por cada rango posible que podamos obtener.

Índice de Masa Corporal	Tu rango
15 o menos	Delgadez muy severa
15 – 15.9	Delgadez severa
16 – 18.4	Delgadez
18.5 – 24.9	Peso Saludable
25 – 29.9	Sobrepeso
30 – 34.9	Obesidad Moderada
35 – 39.9	Obesidad severa
40 o más	Obesidad muy severa (obesidad mórbida)

```
c=0
if(IMCmedio<=15){
  c='Delgadez muy severa'
}else if (IMCmedio<16){
  c='Delgadez severa'
}else if (IMCmedio<18.5){
  c='Delgadez'
}else if(IMCmedio<25){
  c= 'Peso saludable'
}else if(IMCmedio<30){
  c='Sobrepeso'
}else if(IMCmedio<35){
  c='Obesidad moderada'
}else if(IMCmedio<40){
  c='Obesidad severa'
}else if(40<=IMCmedio){
  c='Obesidad muy severa, obesidad
mórbida'
}
c
```