Pràctica 1

R com a eina de càlcul

1.1 Introducció

L'objectiu d'aquesta primera pràctica és conèixer el funcionamen bàsic del programa i veure, sense entrar en massa detalls tècnics, algunes de les seves possibilitats. Veurem primer com definir variables i funcions. Després ens centrarem en algunes estructures de dades i en el càlcul de funcions de densitat i distribució. Un cop presentats aquests elements, proposem un seguit d'execicis per agafar confiança en l'ús del programa.

1.2 Assignació de variables i operacions bàsiques

L'assignació de valors a una variable es fa:

```
> a <- 2
> b <- a + 2
> a
```

```
[1] 2
```

```
> b
```

```
[1] 4
```

```
> a + b
```

```
[1] 6
```

Els operadors bàsics segueixen les regles de preferència habituals

```
> 2 + 3 * 6
```

```
[1] 20
```

```
> (2 + 3) * 6
```

```
[1] 30
```

L'exponenciació té preferència:

```
> 2 + 3^6
```

```
[1] 731
```

```
> 2 + (3^6)
```

```
[1] 731
```

Quan una variable ja no interessa, pot se útil d'esborrar-la completament. Això es pot fer de la següent manera

```
> a <- 1
> a
```

```
[1] 1
```

```
> rm(a)
```

Si no s'esborra utilitzant aquesta opció, aleshores una variable està sempre activa. Evidentment, el seu valor es pot reassignar quan convingui:

```
> a <- 3
> a
```

```
[1] 3
```

```
> a <- 5
> a
```

```
[1] 5
```

1.3 Vectors i altres estructures de dades

En R és molt interessant utiltzar vectors, matrius, i llistes. Veurem alguns exemples introductoris d'aquestes estructures.

Un vector es pot definir com

```
> x <- c(12, 23, 34, 24, 56, 26)
> x
```

```
[1] 12 23 34 24 56 26
```

Ens podem referir als elements del vector fent

```
> x[1]
```

```
[1] 12
```

```
> x[4]
```

```
[1] 24
```

Podem operar amb vectors de la manera següent

```
> x[1] + x[4]
```

```
[1] 36
```

```
> 2 * x
```

```
[1]  24  46  68  48 112  52
```

```
> x + x
```

```
[1]  24  46  68  48 112  52
```

```
> x + x[3]
```

```
[1] 46 57 68 58 90 60
```

Els vectors poden contenir caracters

```
> noms <- c("Andreu", "Montse", "Jordi", "Rosa")
> edats <- c(18, 19, 21, 18)
```

Podem fer una matriu amb aquests vectors

```
> data <- cbind(noms, edats)
> data
```

```
     noms     edats
[1,] "Andreu" "18"
[2,] "Montse" "19"
[3,] "Jordi"  "21"
[4,] "Rosa"   "18"
```

En aquest cas, tots els elements de la matriu són de tiups catàcter. Això passa perqué tots els elements d'una matriu han de ser del mateix tipus. En R és més pràctic definir un *data.frame*

```
> data <- data.frame(noms, edats)
> data
```

```
    noms edats
1 Andreu    18
2 Montse    19
3  Jordi    21
4   Rosa    18
```

```
> data[1, 2]
```

```
[1] 18
```

```
> data[1, ]
```

```
    noms edats
1 Andreu    18
```

```
> data[1]
```

```
    noms
1 Andreu
2 Montse
3  Jordi
4   Rosa
```

Ens podem definir als vectors columna d'un *data.frame* utilitzant el seu nom

```
> data$noms
```

```
[1] Andreu Montse Jordi  Rosa
Levels: Andreu Jordi Montse Rosa
```

```
> data$edats
```

```
[1] 18 19 21 18
```

```
> data$edats[3]
```

```
[1] 21
```

```
> data$edats[3] + data$edats[4]
```

```
[1] 39
```

```
> length(data$edats)
```

```
[1] 4
```

Finalment, una llista és una estructura que pot contenir elements de diferent tipus. P.e:

```
> resultat <- list(Tipus = 1, nom = "Clinica Nova",
+     serveis = c("Urgencies", "Traumatologia",
+         "Interna"), llits = 43)
> resultat
```

```
$Tipus
[1] 1

$nom
[1] "Clinica Nova"

$serveis
[1] "Urgencies"     "Traumatologia" "Interna"

$llits
[1] 43
```

```
> resultat$nom
```

```
[1] "Clinica Nova"
```

```
> resultat$serveis
```

```
[1] "Urgencies"     "Traumatologia" "Interna"
```

```
> resultat$llits
```

```
[1] 43
```

```
> c(resultat$nom, resultat$llits)
```

```
[1] "Clinica Nova" "43"
```

Molts procediments estadístics en R produeixen resultats en forma de llista. Veurem més endevant com s'utilitzen les llistes i la interpretació dels seus valors en els resultats d'aquests anàlisis.

1.3.1 Omplir vectors amb repeticions

Molts cops és útil omplir vectors amb series de nombres o amb repeticions de nombres iguals. Això ho podem fer amb les instruccions

```
> c(1:10)
```

```
 [1]  1  2  3  4  5  6  7  8  9 10
```

```
> rep(1, 10)
```

```
 [1] 1 1 1 1 1 1 1 1 1 1
```

```
> c(rep(1, 10), rep(2, 10))
```

```
 [1] 1 1 1 1 1 1 1 1 1 1 2 2 2 2 2 2 2 2 2 2
```

```
> c(rep("A", 10), rep("B", 10))
```

```
 [1] "A" "A" "A" "A" "A" "A" "A" "A" "A" "A" "B" "B" "B" "B"
[15] "B" "B" "B" "B" "B" "B"
```

També podem obtenir vectors fent servir

```
> seq(1, 20, 2)
```

```
 [1]  1  3  5  7  9 11 13 15 17 19
```

```
> seq(1, 20, 3)
```

```
[1]  1  4  7 10 13 16 19
```

```
> seq(1, 20, 1)
```

```
 [1]  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18
[19] 19 20
```

1.4 Algunes funcions bàsiques

R porta incorporades moltes funcions, p.e. sinus, logaritme, exponencial i arrel quadrara:

```
> sin(pi/2)
```

```
[1] 1
```

```
> log(12)
```

```
[1] 2.484907
```

```
> exp(log(12))
```

```
[1] 12
```

```
> sqrt(16)
```

```
[1] 4
```

1.5 Definició de funcions en R

En R podem definir les nostres pròpies funcions per tal de realitzar els càlculs que necessitem. La sintaxi bàsica per a definir una funció que calculés l'arrel cúbica seria:

```
> Arrel.Cubica <- function(x) {
+     x^(1/3)
+ }
```

Ara podriem fer:

```
> Arrel.Cubica(27)
```

```
[1] 3
```

```
> Arrel.Cubica(189)
```

```
[1] 5.738794
```

```
> Arrel.Cubica(log(123))
```

```
[1] 1.688291
```

Una funció pot tenir diferents paràmetres. P.e. l'arrel n-èssima d'x es podria obtenir amb la funció:

```
> Arrel <- function(x, n) {
+     x^(1/n)
+ }
```

Aquesta funció s'utilitzaria introduint dos valors, un per x i un per a n:

```
> Arrel(16, 2)

[1] 4

> Arrel(16, 3)

[1] 2.519842

> Arrel(12^4, 4)

[1] 12

> Arrel(123, 4.3)

[1] 3.062136
```

1.5.1 Funcions amb valors de paràmetres per defecte

La funció que calcula l'arrel n-èssima d'x es podria definir com:

```
> Arrel <- function(x, n = 2) {
+     x^(1/n)
+ }
```

D'aquesta manera, si introduim només un paràmetre, obtindrem, per defecte, l'arrel quadrada:

```
> Arrel(16)
```

```
[1] 4
```

```
> Arrel(12, 2)
```

```
[1] 3.464102
```

```
> Arrel(12, 4)
```

```
[1] 1.861210
```

Podem definir tots els paràmetres per defecte:

```
> Arrel <- function(x = 4, n = 2) {
+     x^(1/n)
+ }
```

Aleshores, si no especifiquem cap valor, obtenim, per defecte, l'arrel quadrada de 4. Les següents instruccions són, ara, equivalents:

```
> Arrel()
```

```
[1] 2
```

```
> Arrel(4)
```

```
[1] 2
```

```
> Arrel(4, 2)
```

```
[1] 2
```

1.5.2 Gràfiques de funcions

L'obtenció de gràfiques de funcions és molt senzilla en R. Per exemple, definim una funció que representi la velocitat d'una reacció enzimàtica que segueixi un mecanisme de Michaelis-Menten:

```
> Rate.MM <- function(x, Vm = 10, Km = 2) {
+     Vm * x/(Km + x)
+ }
> Rate.MM(3)

[1] 6

> Rate.MM(3, Vm = 5, Km = 1)

[1] 3.75
```

Si indiquem els tres arguments, també podem indicar:

```
> Rate.MM(3, 5, 1)

[1] 3.75
```

El primer argument serà el valor d'x, el segon la V_m i el tercer K_m. La gràfica d'aquesta funció es pot obtenir fent (Fig.1.1)

```
> curve(Rate.MM(x, 5, 3), 0, 20)
```

Podem dibuixar diferents corbes en la mateixa gràfica fent (Fig.1.2)

```
> curve(Rate.MM(x, 5, 3), 0, 30)
> curve(Rate.MM(x, 5, 1), 0, 30, add = T, col = "red")
```

Podem definir les etiquetes de cada eix i introduir una legenda en la gràfica (Fig.1.3)

```
> curve(Rate.MM(x, 5, 3), 0, 30, xlab = "Concentració de substrat",
+     ylab = "Velocitat")
> legend(20, 1, c("Vm=5, Km=3", "Vm=5, Km=1"), text.col = c("black",
+     "red"))
> curve(Rate.MM(x, 5, 1), 0, 30, add = T, col = "red")
```

1.6 Control del flux d'execució, avaluacions lògiques i qüestions relacionades

1.6.1 Avaluació de condicions

En R, com es qualsevol llenguatge de programació es poden avaluar condicions. P.e.

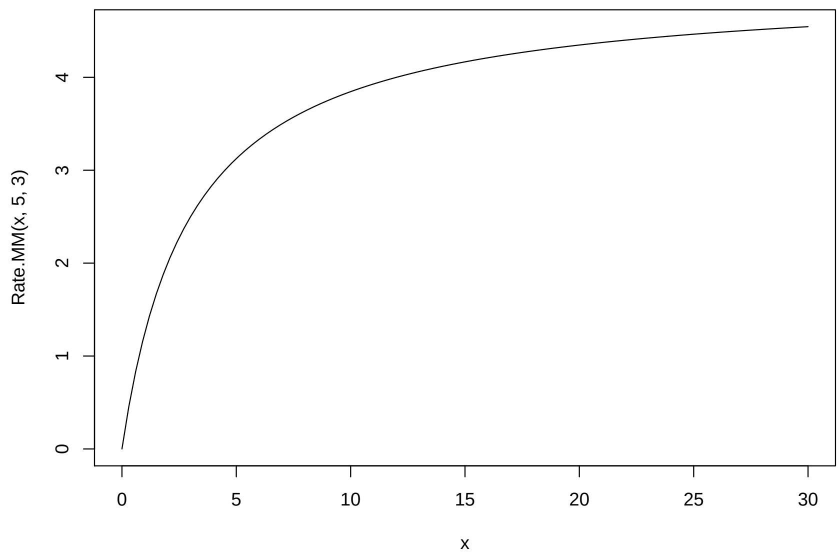

Figura 1.1: Velocitat d'una recció enzimàtica tipus Michaelis-Menten amb $V_m = 5, K_m = 3$

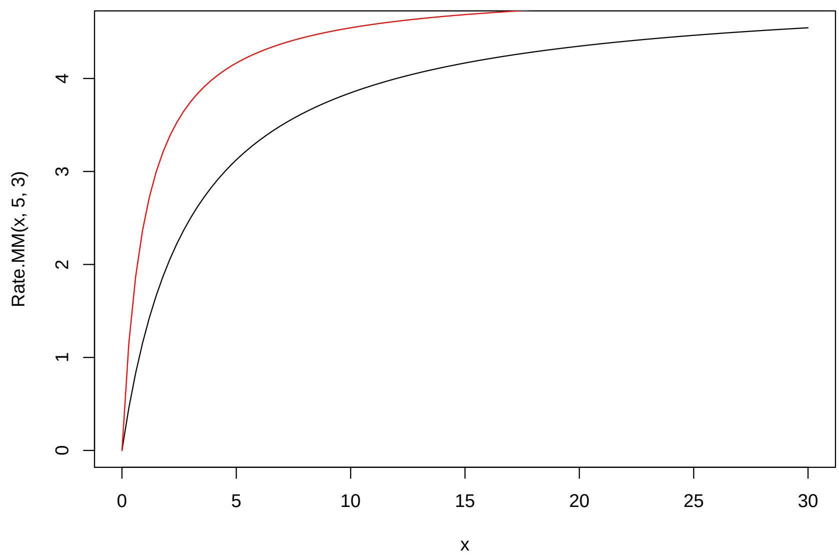

Figura 1.2: Comparació de la velocitat de dues reccions enzimàtiques tipus Michaelis-Menten (negre:$V_m = 5, K_m = 3$, vermell:$V_m = 5, K_m = 1$)

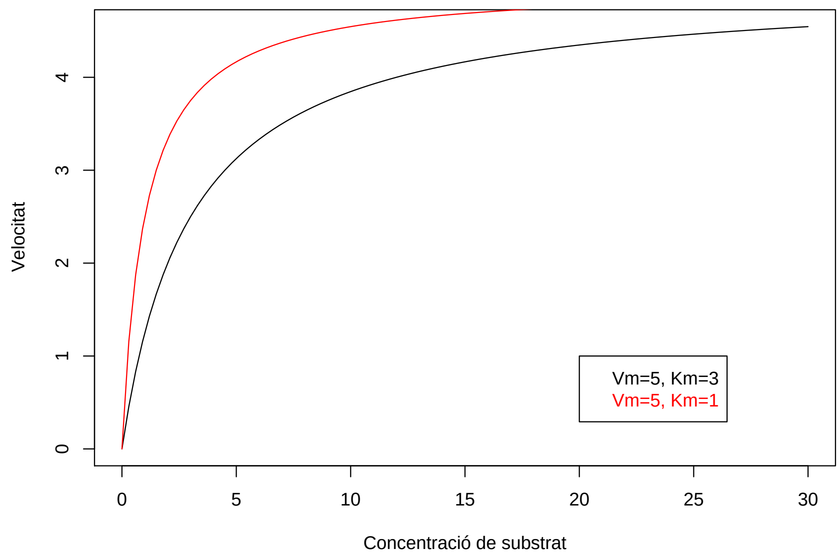

Figura 1.3: Comparació de la velocitat de dues reccions enzimàtiques tipus Michaelis-Menten

```
> 3 > 4
```

```
[1] FALSE
```

Podem comparar variables:

```
> x <- 4
> y <- 3
> x > 3
```

```
[1] TRUE
```

```
> x == y
```

```
[1] FALSE
```

També és posible provar diferents condicions. L'acompliment simultani de dues condicions es prova fent

```
> (x > y) & (3 < 4)
```

```
[1] TRUE
```

Si volem provar si una o altre condició es compleix, farem

```
> (x > y) | (x > 4)

[1] TRUE
```

Fent servir avaluacions lògiques podem fer funcions que prenguin diferent valors segons els valors de l'argument. La instrucció sl ifelse(condicio,valor si veritable,valor si fals) produeix una avaluació de la condició, de manera que retorna un o altre valor segons que sigui certa o falsa. Fent servir aquest evaluador, podem construir una funció com ara:

```
> f <- function(x, punt = 0, step = 1) {
+     r <- ifelse(x <= punt, 0, step)
+     r
+ }
> f(-3)

[1] 0

> f(2)

[1] 1
```

Si canviem el punt, aleshores tidrem

```
> f(2, punt = 1)

[1] 1

> f(2, punt = 3)

[1] 0
```

Finalment, si canviem *step*, aleshores el valor de la funció per a valors per sobre del *punt* canvia

```
> f(2, punt = 1, step = 0.5)

[1] 0.5
```

1.6.2 Iteracions

Podem iterar (repetir) un càlcul fent

```
> suma <- 0
> for (i in 1:3) suma <- suma + i
> suma
```

```
[1] 6
```

Amb iteracions d'aquesta mena podem programar funcions com el factorial (tot i que esta definit en R ho fem a tall d'exemple):

```
> f <- function(n) {
+     r <- 1
+     for (i in 1:n) r <- r * i
+     r
+ }
```

Ara podem calcular el factorial fent servir aquesta funció i comparar el resultat amb la funció d'R

```
> f(7)
```

```
[1] 5040
```

```
> factorial(7)
```

```
[1] 5040
```

També pot ser útil fer algun càlcul mentre es compleixi una condició. P.e.

```
> x <- 16
> while (x > 4) {
+     print(x)
+     x <- x - 1
+ }
```

```
[1] 16
[1] 15
```

```
[1] 14
[1] 13
[1] 12
[1] 11
[1] 10
[1] 9
[1] 8
[1] 7
[1] 6
[1] 5
```

1.7 Funcions de densitat i distribució de variables aleatòries

R porta incorporades les funcions de densitat i distribució de variables aleatòries de les distribucions més comuns. Per ejemple, en el cas d'una v.a. amb distribució Binomial de paràmetres $n = 20, p = 0.3$, la $P(X = 3)$ s'obté amb la funció:

```
> dbinom(3, 20, 0.3)
```

```
[1] 0.07160367
```

Podem comprovar aquest resultat calculant $P(X = 3) = \binom{20}{3} 0.3^3 0.7^{17} = 20!/(3!17!)0.3^3 0.7^{17}$

```
> factorial(20)/(factorial(3) * factorial(17)) *
+     0.3^3 * 0.7^17
```

```
[1] 0.07160367
```

La funció de distribució $P(X \leq 3)$ es calcularia fent

```
> pbinom(3, 20, 0.3)
```

```
[1] 0.1070868
```

Podem generar una gràfica amb tots els valors de la funció de densitat per a la binomial $B(20, 0.3)$ mitjançant el procediment (Fig.1.4)

```
> x <- c(1:21)
> d <- c(1:21)
> for (i in 0:20) d[i + 1] <- dbinom(i, 20, 0.3)
> barplot(d, names.arg = c(0:20), xlab = "Valors d'X",
+     ylab = "P(X=x)")
```

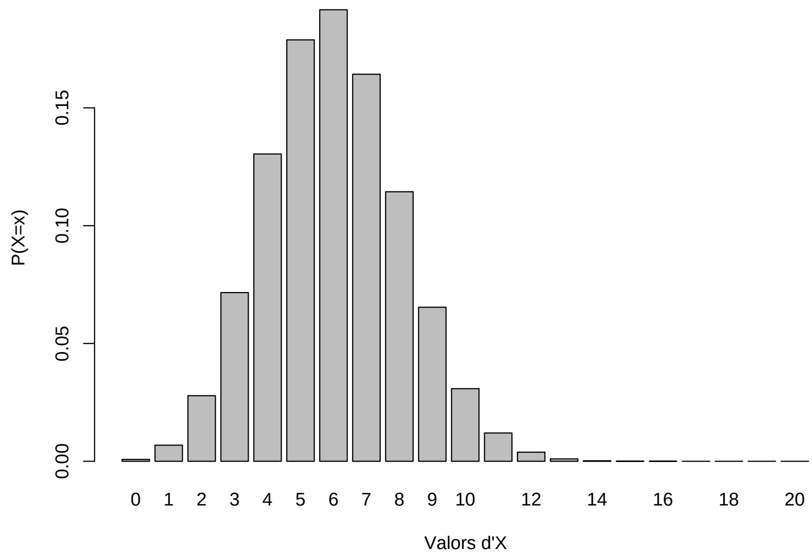

Figura 1.4: Funció de densitat de la $B(20, 0.3)$

1.7.1 Exemples de càlcul amb una $B(n, p)$

Exemple 1

Suposem una v.a. $B(100, 0.4)$. Veiem com podem calcular alguns resultats

- $P(X < 25) = P(X \leq 24)$:

  ```
  > pbinom(24, 100, 0.4)
  [1] 0.0005615352
  ```

- Una manera pràctica de calcular$P(24 < X < 67)$ és:

  ```
  > sum(dbinom(25:66, 100, 0.4))
  [1] 0.9994384
  ```

Exemple 2

Un tractament és efectiu en un 70% dels casos. Si s'administra a 25 pacients, calcula:

- Probabilitat que sigui efectiu en més d'un 70% dels casos: $P(X > 25 * 0.7) = 1 - P(X \leq 25 * 0.7)$:

  ```
  > pbinom(25 * 0.7, 25, 0.7)
  ```

  ```
  [1] 0.4881515
  ```

- Probabilitat que sigui efectiu entre un 70% i un 80% dels casos: Primer calcularem:

  ```
  > 25 * 0.7
  ```

  ```
  [1] 17.5
  ```

  ```
  > 25 * 0.8
  ```

  ```
  [1] 20
  ```

  Necessitem calcular, per tant

  $$P(17.5 \leq X \leq 20) = P(X = 18) + P(X = 19) + P(X = 20)$$

  Això ho farem amb la instrucció:

  ```
  > sum(dbinom(18:20, 25, 0.7))
  ```

  ```
  [1] 0.4213766
  ```

1.7.2 Altres variables aleatòries

Distribució multinomial

La densitat multinomial correspon a l'expressió:

$$f(x_1, .., x_k) = \frac{n!}{x_1! \cdots x_n!} p_1^{x_1} \cdots p_k^{x_k}$$

Com un exemeple, suposem el cas dels descendents d'un creuament Aa x Aa. Els possibles descendents tenen probabilitats:

$$P(AA) = 1/4, P(Aa) = 1/2), P(aa) = 1/4$$

La probabilitat d'obtenir 1 descendent AA, 2 Aa i 1 aa, en un total de 4 descendents, es calcula:

```
> dmultinom(c(1, 2, 1), prob = c(1/4, 1/2, 1/4))
```

```
[1] 0.1875
```

Altres probabilitats, en aquest cas, serien:

```
> dmultinom(c(2, 2, 0), prob = c(1/4, 1/2, 1/4))

[1] 0.09375

> dmultinom(c(0, 4, 0), prob = c(1/4, 1/2, 1/4))

[1] 0.0625

> dmultinom(c(2, 0, 2), prob = c(1/4, 1/2, 1/4))

[1] 0.0234375

> dmultinom(c(4, 0, 0), prob = c(1/4, 1/2, 1/4))

[1] 0.00390625
```

Distribució de Poisson

Una v.a. amb distribució de Poisson té funció de densitat:

$$f(x) = e^{-\lambda} \frac{\lambda^x}{x!}$$

En el cas $\lambda = 3.3$, la $P(X = 2)$ i la $P(X \leq 5)$ es calculen:

```
> dpois(2, 3.3)

[1] 0.2008288

> ppois(5, 3.3)

[1] 0.8828768
```

Per a calcular $P(1 \leq X \leq 8)$ farem

```
> sum(dpois(1:8, 3.3))
```

```
[1] 0.956205
```

Com a exemple, suposem que en una empresa hi ha, de mitjana, 5 aturades tècniques al dia degudes a errades en la maquinària. Quina és la probabilitat que en un dia determinat hi hagin més de 5 aturades?. Utilitzant la distribució de Poisson de paràmetre $\lambda = 5$, es tracta de calcular $P(X > 5) = 1 - P(X \leq 5)$. La probabilitat buscada és:

```
> 1 - ppois(5, 5)
```

```
[1] 0.3840393
```

En quin percentage de dies hi haurà 8 o més aturades?. Es tracta de calcular

$$P(X \geq 8) = 1 - P(X < 8) = 1 - P(X \leq 7)$$

Podem obtenir aquesta probabiliat fent:

```
> 1 - ppois(7, 5)
```

```
[1] 0.1333717
```

1.8 Algunes distribucions continues

1.8.1 Distribució uniforme

La distribució uniforme te una funció de densitat:

$$\begin{aligned} f(x) &= \frac{1}{b-a} \text{ si } x \in (a, b) \\ f(x) &= 0 \text{ si } x \in (a, b) \end{aligned}$$

Com en la resta de variables, la densitat i la ditribució s'ontenen com

```
> dunif(0.3)
```

```
[1] 1
```

```
> punif(0.3)
```

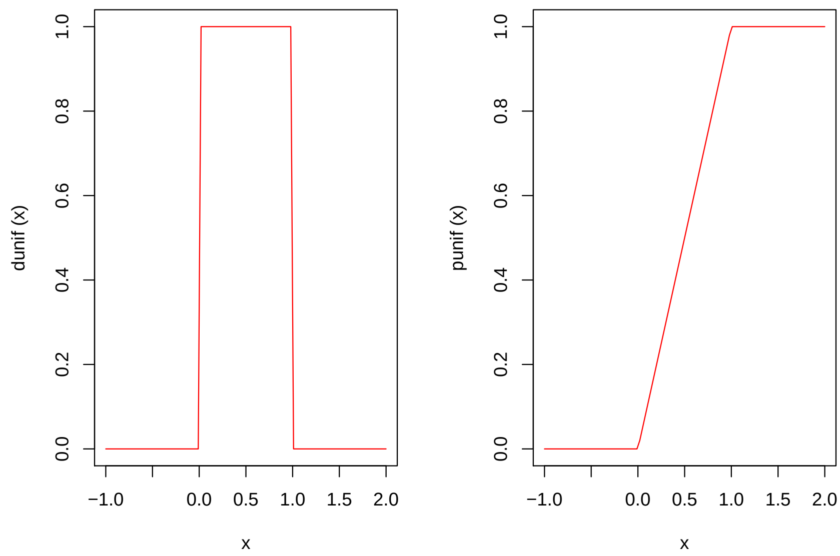

Figura 1.5: Funció de densitat i distribució de la Uniforme(0,1)

```
[1] 0.3
```

Com veieu, per defecte $a = 0$ i $b = 1$. Si volem altres valors podem indicar-ho com:

```
> dunif(0.7, min = 0.5, max = 1.3)
```

```
[1] 1.25
```

Podem fer la gràfica de la funció de densitat i distribució de la uniforme entre $(0, 1)$ (Fig.1.5)

```
> par(mfrow = c(1, 2))
> curve(dunif, -1, 2, xlabel = "X", ylabel = "f(x)",
+     col = "red")
> curve(punif, -1, 2, xlabel = "X", ylabel = "F(x)",
+     col = "red")
> par(mfrow = c(1, 1))
```

1.8.2 Distribució exponencial

Una v.a. amb distribució exponencial té com a densitat

$$f(x) = \alpha e^{-\alpha\, x}$$

Aquesta distribució serveix per a representar situacions com ara el temps que passa fins a que es produeixi un esdeveniment. Veiem-ne un exemple. Considerem que el temps mig de vida útil d'un determinat compost és de 5 anys. En aquest cas, podem considerar que

$$E(X) = 1/\alpha = 5 \Rightarrow \alpha = 0.2$$

Podem calcular la probabilitat que el compost duri més de 7 anys com $P(X > 7) = 1 - P(X \leq 7)$:

```
> pexp(7, 0.2)
```

```
[1] 0.753403
```

La gràfica de la funció de distribució s'obté amb la instrucció (Fig.1.6)

```
> curve(pexp(x, 0.2), 0, 30, xlab = "X", ylab = "F(x)",
+     col = "red")
```

Quin és el temps que, d'acord amb aquesta distribució, superaran el 90% dels components?. En aquest cas, es tracta de solucionar l'eqüació

$$P(X > a) = 0.9 \Rightarrow P(X \leq a) = 0.1$$

És a dir, hem de trobar el valor del quantil 0.1:

```
> xp <- qexp(0.1, 0.2)
> xp
```

```
[1] 0.5268026
```

Podem comprovar que el resultat és correcte

```
> 1 - pexp(xp, 0.2)
```

```
[1] 0.9
```

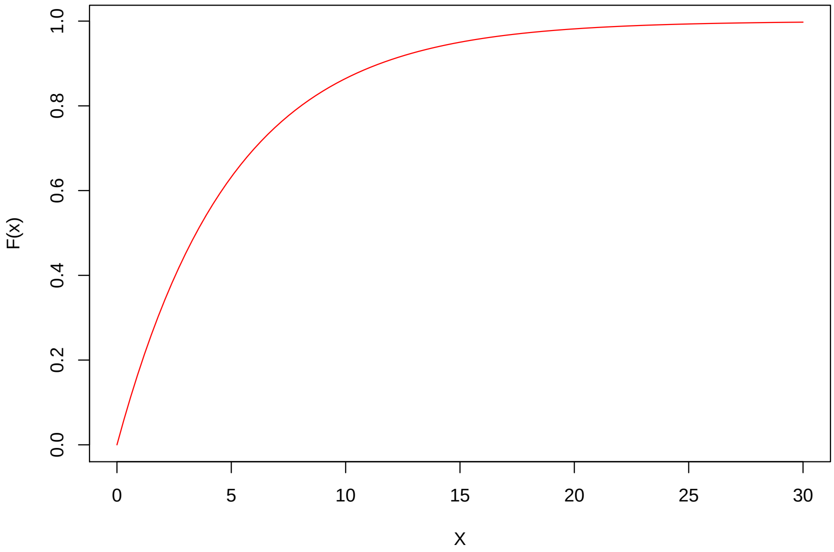

Figura 1.6: Funció de distribució de la Exponencial(0.2)

1.8.3 Càlculs amb variables aleatòris no definides en R

Considereu, per exemple, el cas d'una variable aleatòria que té com a funció de densitat

$$\begin{aligned} f(x) &= \frac{1}{3-\pi/\sqrt{3}} \times \frac{x^2}{3+x^2} \text{ si } x \in (0,3) \qquad (1.1) \\ f(x) &= 0 \text{ si } x \in (0,3) \end{aligned}$$

Per calcular la seva funció de distribució hem de fer:

$$F(x) = \int_0^x \frac{1}{3-\pi/\sqrt{3}} \times \frac{x^2}{3+x^2} dx$$

En R podem fer-ho numèricament, de manera que podem definir una funció que ho calculi

```
> dva <- function(x) {
+     1/(3 - pi/sqrt(3)) * x^2/(3 + x^2)
+ }
> pva <- function(x) {
+     res <- integrate(dva, 0, x)[1]
+     res
+ }
```

Calculem ara $P(X \leq 2)$

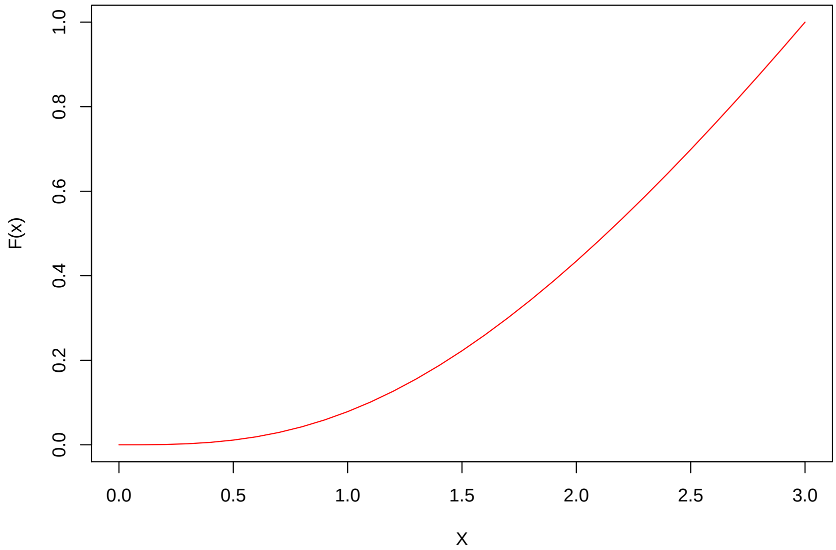

Figura 1.7: Funció de distribució de la v.a. definida en l'eq.1.1

```
> pva(2)

$value
[1] 0.4345874
```

Per a graficar aquesta funció, hem de generar dos vectors, un amb els valors de les x i l'altre amb les $F(x)$

```
> X <- seq(0, 3, 0.1)
> F <- sapply(seq(0, 3, 0.1), pva)
```

Ara podem obtenir la gràfica fent (Fig.1.7)

```
> plot(X, F, type = "l")
```

1.9 Exercicis pràctics

- Definiu una funció que calculi automàticament la probabilitat $P(a < X < b)$ en el cas d'un v.a. Binomial. La funció ha de funcionar entrant els valors de a i b directament. També s'han de poder especificar els valors dels paràmetres de la binomial.

- Troba com definir automàticament, és a dir sense entrar-los manualment, els vectors següents:
  - (1,1,1,1,1,1,1,2,2,2,2,2,2,2)
  - ((1,1,1),(2,2,2),(3,3,3))
  - (1,3,5,7,9)
  - (A,B,A,B,A,B,A,B)
  - (7,6,5,4,3,2,1)

- Considera una v.a. discreta amb funció de densitat:

$$\begin{aligned} f(1) &= 1/8 \\ f(2) &= 1/4 \\ f(3) &= 1/16 \\ f(4) &= 3/16 \\ f(5) &= 3/8 \end{aligned}$$

  - Dibuixa la seva funció de densitat, indicant etiquetes apropiades per als eixos.
  - Defineix una funció que calculi la $f(x)$ d'aquesta v.a. (la funció *switch* d'R et por ser útil).
  - Defineix una funció que calculi la $F(x)$ d'aquesta v.a.

- Repasa els càlculs de la segona sessió de problemes fent servir les possibilitats de càlcul d'R

Índex