

Introductie in R

R is een programmeer taal met een groot aantal voorgeprogrammeerde statistische functies. Het is de open source versie van S-plus. Wij gebruiken R dan ook omdat het gratis is. Documentatie voor R (S-plus) vind je op het web:

```
http://www.math.montana.edu/stat/tutorials/R-intro.pdf
http://www.math.montana.edu/stat/docs/sguide.ps      door Brian Ripley
http://www.math.montana.edu/stat/docs/Splus_notes.ps  door Bill Venables
http://www.stat.berkeley.edu/users/spector/intro_s.pdf door Phil Spector
```

Andere, bekendere software voor statistische analyse is Excel, SPSS, SAS, Minitab, etc. Wij maken geen gebruik van Excel, omdat het de statistische functies teveel achter knoppen verborgen houdt. Wij willen juist zien hoe alles in z'n werk gaat. R kan wel eenvoudig data vanuit Excel importeren.

Hieronder staan een aantal voorbeelden van de belangrijkste R commando's. Het is de bedoeling dat je ze zelf intypt, of met copy-paste overbrengt, en kijkt wat er gebeurt. Wees nieuwsgierig en probeer zelf varianten uit. Wat zou er gebeuren als ik $1/0$ uitreken... Hoe kan ik de sinus functie plotten...

1 beginnen en ophouden

Start R op. Je krijgt een window met een prompt `>`. Hier kun je commando's typen. Om R af te sluiten, typ je

```
> q()
Save workspace image? [y/n/c]:
```

en typ n.

2 R als rekenmachine

```
> 2+4
[1] 6
> 3*5
[1] 15
> 2^3
[1] 8
> sqrt(9)
[1] 3
> sin(pi)          # R kent pi
[1] 1.224606e-16   # nou ja, eigenlijk 0!
```

Soms is het antwoord van een berekening TRUE of FALSE

```
> 2>4             # is 2 groter dan 4?
[1] FALSE

> 3==3           # let op de dubbele ==
[1] TRUE
```

```
> 3!=3          # != staat voor ongelijkheid
[1] FALSE

> (3>2)&(3>4)   # & staat voor het logische "EN"
[1] FALSE

> (3>2)|(3>4)   # | staat voor het logische "OF"
[1] TRUE
```

opgave 1 Bepaal uit je hoofd

```
((3>2)|(3>4))|((3>2)&(3>4))
```

en controleer je antwoord met R Zijn de haakjes belangrijk? Wat is

```
((3>2)|(3>4)|(3>2))&(3>4)
```

3 variabelen

Met behulp van een pijltje `<-` kunnen we een waarde toekennen aan een variabele. Het volgende commando maakt een variabele `a` met waarde 3.162.

```
> a <- sqrt(10)
```

Kijk maar:

```
> a
[1] 3.162278
```

```
> b <- a+5
```

maakt een variabele `b` met waarde 8.162.

```
> rm(a)
```

verwijdert de variabele `a`.

Een zogeheten Boolese variabele (naar de engelsman George Boole) neemt de waarde `TRUE` of `FALSE` aan.

```
> c<-(3<4)
> c
[1] TRUE
```

Je kunt ook met Boolese variabelen rekenen. Dan geldt: `TRUE=1` en `FALSE=0`.

4 vectoren

Vectoren zijn het belangrijkste onderdeel van R. Probeer de volgende voorbeelden uit

```
> x<-c(1,3,2,6,0)
> x<-rep(3,10)
> x<-seq(1, 20, by = 3)
> x<-seq(10, 1, by = -2)
> x<-1:10
> x
[1] 1 2 3 4 5 6 7 8 9 10
```

We kunnen ook rekenen met vectoren

```
> y<-3*x - 12
> y
[1] -9 -6 -3 0 3 6 9 12 15 18
> z<-(y<0)
> z
[1] TRUE TRUE TRUE FALSE FALSE FALSE FALSE FALSE FALSE
```

Rekenen gaat meestal elementsgewijs, maar soms niet. Vergelijk

```
> x*y
[1] -9 -12 -9 0 15 36 63 96 135 180
> x%*%y
      [,1]
[1,] 495
```

De notatie `x%*%y` geeft het inproduct van de vectoren `x` en `y`

$$xy = \sum_{i=1}^n x_i y_i.$$

5 subscripting

Subscripting is het selecteren van elementen van een vector. We gebruiken rechte haken [en]

```
> y[3]
[1] -3
> y[3:5]
[1] -3 0 3
> y[c(1,3,5)]
[1] -9 -3 3
```

Negatieve selectie kan ook

```
> y[-3] # laat het derde element weg
[1] -9 -6 0 3 6 9 12 15 18
> y[-c(1,3,5)] # laat elementen 1,3 en 5 weg
[1] -6 0 6 9 12 15 18
```

We kunnen ook selecteren met behulp van een Boolese vector. We selecteren alle elementen van `y` die groter zijn dan 0 door

```
> y[y>0]
[1] 3 6 9 12 15 18
```

6 matrices

```
> A<-matrix(1:12, nrow = 4, ncol = 3) # de getallen 1 t/m 12 in 4 rijen
# en 3 kolommen.
```

```
> A
      [,1] [,2] [,3]
[1,]    1    5    9
[2,]    2    6   10
[3,]    3    7   11
[4,]    4    8   12
```

We kunnen natuurlijk elementen van de matrix selecteren

```
> A[1,2]
[1] 5
```

```
> A[1:2,2:3]
      [,1] [,2]
[1,]    5    9
[2,]    6   10
```

```
> A[2,]          #selecteer de tweede rij
[1] 2 6 10
```

7 functies

We hebben al een aantal functies gebruikt: `q()`, `sqrt()`, `sin()`, `c()`, `rep()`, `seq()`, `matrix()`. R heeft bijzonder veel ingebouwde functies. De uitleg van een functie is de vinden door

```
> help(sqrt)
```

Hoe weet je nou of Splus een ingebouwde functie heeft voor (bijvoorbeeld) de cosinus? De beste methode is proberen te raden

```
> cosinus(pi)
Error: couldn't find function "cosinus"
> cos(pi)
[1] -1      # Aha!
```

Je kunt ook altijd Google of de documentatie proberen.

8 plotten

Je kunt de functies `plot()`, `points()` en `lines()` gebruiken om plaatjes te maken.

```
> x<-seq(0,2*pi,by=0.1)
> plot(x,sin(x))
```

of

```
> plot(x,sin(x),type='l')
```

Bekijk `help(plot)` voor alle mogelijkheden. De functie `plot()` maakt het bestaande plaatje eerst leeg. `lines()` en `points()` plotten over het bestaande plaatje heen.

opgave 2 Plot de functie $f(x) = x^2$ op het interval $[-5, 5]$. Breng vervolgens de functie $g(x) = |5x|$ aan in je grafiek.

opgave 3 Bedenk zelf een functie met een verticale asymptoot, en plot deze.

opgave 4 Maak een plaatje van de letter F.

9 scripting

Een script is een programmaatje. Kies je favoriete editor, en schrijf een lijstje met commando's dat je wilt laten uitvoeren. Met copy-paste kun je deze commando's dan naar het R window kopiëren. Je kunt ook de functie `source()` gebruiken

```
> source("myscript.r")
```

Hier is een voorbeeld van een scriptje dat de kwadraten van de getallen 1 t/m 10 uitrekent. We gebruiken een for-lus.

```
for (i in 1:10) {  
  cat(i,i^2,"\n")  
}
```

De functie `cat()` schrijft naar het scherm. De toevoeging `"\n"` betekent "nieuwe regel". Laat de toevoeging weg, en kijk wat er gebeurt.

Hier is een script dat de if-constructie demonstreert. Let op alle accolades! Merk ook op hoe je door in te springen de structuur van het programma duidelijk kunt maken.

```
for (i in 1:10) {  
  cat(i,i^2,"\n")  
  if (i == 5) {  
    cat("halverwege!\n")  
  }  
} # deze accolade sluit de if-constructie af  
# deze accolade sluit de for-lus af
```

opgave 5 Een for-lus is soms niet nodig, en kost dan nodeloos veel rekentijd. Bepaal de kwadraten van de getallen 1 t/m 10 met behulp van een enkel commando.

opgave 6 In de Fibonacci rij is ieder getal gelijk aan de som van de twee voorafgaande getallen: 1, 1, 2, 3, 5, 8, 13, 21, ... Gebruik R om de eerste 20 Fibonacci getallen te bepalen.

opgave 7 Zij $x(1) = 1, x(2) = 1, x(3) = 2, \dots$ de Fibonacci getallen. Maak een plot van de getallen $y(1) = x(2)/x(1), y(2) = x(3)/x(2), \dots, y(19) = x(20)/x(19)$. De rij $y(i)$ convergeert naar de zogeheten Gouden verhouding.

opgave 8 De methode van Newton wordt gebruikt om het nulpunt van een functie f te bepalen. Begin met een start-waarde $x(1)$ die redelijk dicht in de buurt van het nulpunt is. Herhaal de volgende procedure

$$x(n+1) = x(n) - \frac{f(x(n))}{f'(x(n))}, \quad n = 1, 2, \dots$$

totdat het verschil tussen de opeenvolgende waarden kleiner is dan, zeg, 10^{-9} .

Implementeer de methode van Newton om het nulpunt te bepalen van de functie $f(x) = \cos(x)$ op het interval $[0, \pi]$. Begin met startwaarde $x(1) = 1$.

opgave 9 Het commando `x<-sample(1:10,10000,replace=T)` produceert een rij van tienduizend toevallig gekozen getallen tussen 1 en 10. Voer het commando uit, en tel hoeveel vijven er in de rij voorkomen.

opgave 10 Stel, x is een vector. Wat is het verschil tussen `rank(x)`, `sort(x)` en `order(x)`? Probeer ze allemaal uit!

opgave 11 Het volgende antieke Chinese lineaire probleem bestaat uit drie vergelijkingen voor drie onbekenden (x_1, x_2 en x_3).

$$3x_1 + 2x_2 + x_3 = 39$$

$$2x_1 + 3x_2 + x_3 = 34$$

$$x_1 + 2x_2 + 3x_3 = 26$$

In matrix notatie is dit probleem van de vorm $Ax = b$ met

$$A = \begin{pmatrix} 3 & 2 & 1 \\ 2 & 3 & 1 \\ 1 & 2 & 3 \end{pmatrix} \quad x = \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} \quad \text{en } b = \begin{pmatrix} 39 \\ 34 \\ 26 \end{pmatrix}$$

De matrix A is inverteerbaar, en dus is de oplossing van het probleem $A^{-1}b$. Bepaal deze oplossing met R. De functie `solve()` berekent de inverse van een matrix.