

Bachelor in de biologie, chemie
Biochemie en biotechnologie
Bio-ingenieurswetenschappen
Bachelor in de fysica, informatica, wiskunde
Ingenieurswetenschappen
Bachelor in de toegepaste economische wetenschappen
Handelsingenieur

Statistiek in R

Handleiding bij de oefeningen van de vakken

X0A14B – Statistiek

X0A17A – Statistiek & data-analyse

D0W48A – Bedrijfsstatistiek (HIR)

Stijn Rebry

`stijn.rebry@kuleuven-kulak.be`

Inleiding

R is een computerpakket en programmeertaal, speciaal ontwikkeld voor het maken van numerieke berekeningen, het manipuleren van data en het grafisch voorstellen van gegevens. Dit alles maakt R erg geschikt voor statistische berekeningen. Bovendien is het programma open source en is de programmeertaal sterk objectgeoriënteerd, waardoor enorme bibliotheken vol gespecialiseerde commando's vrij beschikbaar zijn. De instructies kunnen één voor één via een commandolijn worden ingevoerd, maar meestal zullen ze per onderzoeksvraag worden gebundeld in een script.

Het programma kan worden gedownload via de home-pagina van R op het volgende adres

www.r-project.org.

De kern van het programma is de console: het scherm dat de opdrachtlijn bevat, waar alle commando's moeten worden ingevoerd en waar de output verschijnt. In de meeste besturingssystemen heeft R ook een eenvoudige grafische gebruikersinterface, maar de mogelijkheden daarvan zijn zeer beperkt. Het gebruiksgemak kan aanzienlijk worden verhoogd door het werken met een extra programma zoals **RStudio**. De specifieke functionaliteit van dit afzonderlijke programma zal steeds in de marge worden toegelicht.

Deze handleiding geeft eerst een korte neerslag van “An Introduction to R” [4], met een overzicht van de algemene werking, syntax en programmeerstructuren (hoofdstuk 1) en omstandige uitleg voor het correct importeren en manipuleren van gegevensmatrices (hoofdstuk 2). Daarna volgt de tekst de globale structuur van het boek “Statistiek en Wetenschap” [1] met achtereenvolgens beschrijvende statistiek (hoofdstuk 3), hypothesetesten (hoofdstuk 4) en regressie (hoofdstuk 5). Waar in dit laatste hoofdstuk achtergrondinformatie ontbreekt is “Practical Regression and Anova using R” [2] een zeer verhelderende bron. Algemeen zal intensief gebruik van de ingebouwde help-functie [3] onontbeerlijk zijn.

De hoofdttekst zal enkel de functionaliteit van het programma R behandelen, in de marge worden specifieke mogelijkheden van **RStudio** toegelicht. Dit is geen stand-alone programma maar integreert de R-console in een overzichtelijke grafische omgeving. Het werken in deze zogenaamde front-end verandert niets aan de werking van R maar maakt een aantal taken intuïtiever.

Inhoudsopgave

1	Aan de slag	7
1.1	Basisbewerkingen	7
1.2	Vectoren en matrices	8
1.3	Kansen en kwantielen berekenen	12
1.4	Programmeerstructuren	12
1.5	De programma-omgeving	16
2	Statistische datasets	19
2.1	De gegevensmatrix	19
2.2	Soorten variabelen	19
2.3	Gegevens ingeven	20
2.4	Gegevens inlezen	20
2.5	Gegevens selecteren en sorteren	22
2.6	Veranderlijken aanpassen	23
2.7	Kwalitatieve veranderlijken	23
2.8	Gegevens klaar voor gebruik	25
2.9	Classificatie van variabelen	25
3	Beschrijvende statistiek	27
3.1	Frequenties van discrete veranderlijken	27
3.2	Frequenties van continue veranderlijken	28
3.3	Steekproefstatistieken	30
3.4	Boxplots	30
3.5	Het verband tussen veranderlijken	31
3.6	De klasse-afhankelijke methoden	33
3.7	Meer over grafieken	34
4	Hypothesetesten	39
4.1	Test voor normaliteit	39
4.2	Eén gemiddelde of gepaarde groepen	42
4.3	Twee gemiddelden, ongepaarde groepen	45
4.4	Test voor afhankelijkheid	49
4.5	Test voor proporties	54
5	Regressie-analyse	60
5.1	Model opstellen	60
5.2	Modelveronderstellingen	63
5.3	Opsporen van uitschieters	64

5.4	Voorspellingen en voorspellingsintervallen	66
5.5	Meervoudige regressie	67
5.6	Transformaties van veranderlijken	69
5.7	Indicatorvariabelen	70
5.8	Kwaliteit van een regressiemodel	72
6	Variantie-analyse	77
6.1	One-way ANOVA	77
6.2	Paarsgewijze vergelijking tussen groepen	81
6.3	Geneste modellen	82
6.4	Partiële F -test	84
6.5	ANCOVA	85
6.6	Two-way ANOVA	86
6.7	Goodness-of-fit	88
6.8	ANOVA versus regressie	92

Lijst van figuren

1.1	Voorbeelden van verdelingen	13
3.1	Staaf- of Pareto diagram van een kwalitatieve veranderlijke . . .	28
3.2	Histogrammen met verschillende klassebreedten	29
3.3	Dichtheidshistogram	29
3.4	Empirische cumulatieve distributiefunctie	31
3.5	Boxplots en vergelijkende boxplots	32
3.6	Puntenplot van twee continue veranderlijken	34
3.7	Een tikz-figuur	37
4.1	Dichtheidshistogrammen en kwantielplots	40
4.2	Vergelijkende boxplots van het mediane inkomen per regio	49
5.1	Voorspellingen en residuen bij een eenvoudig regressiemodel. . . .	61
5.2	Regressiemodel en modelveronderstellingen	65
5.3	Opsporen van outliers	67
5.4	Regressierechte met betrouwbaarheids- en predictieband	68
5.5	Residuplots bij meervoudige regressie	71
6.1	Voorspellingen en residuen bij een one-way ANOVA model	78
6.2	Voorstelling en veronderstellingen bij one-way ANOVA	80
6.3	Voorstelling van het ANCOVA-model	87
6.4	ANOVA modellen bij twee discrete regressoren	89
6.5	Voorstelling en veronderstellingen bij two-way ANOVA	90
6.6	Vergelijking van regressie- en one-way ANOVA-model.	92


Lijst van tabellen

1.1	Aan de slag in R	18
2.1	Gegevens geordend als gegevensmatrix en als kruistabel	19
2.2	Statistische datasets in R	26
3.1	Beschrijvende statistieken en grafieken in R	38
4.1	Stappenplan voor het uitvoeren van hypothesetesten	56
4.2	Statistieken bij de belangrijkste hypothesetesten	57
4.3	Overzicht van de belangrijkste hypothesetesten	58
4.4	Hypothesetesten in R	59
5.1	Belangrijkste statistieken in verband met lineaire regressie	74
5.2	Stappenplan voor achterwaartse regressie	75
5.3	Regressie in R	76
6.1	Stappenplan bij two-way ANOVA	94
6.2	Variantie-analyse in R	95

Hoofdstuk 1

Aan de slag

1.1 Basisbewerkingen

Voor de basisbewerkingen gebruikt R dezelfde syntax en voorrangsregels als de meeste rekenmachines. Een overzicht van de belangrijkste functies en constanten is te vinden in Tabel 1.1. Voorbeeldcommando's kunnen gewoon worden ingetypt in de console , verderop in de tekst wordt uitgelegd hoe de commando's automatisch kunnen worden ingelezen middels het samen met deze handleiding verspreide script.

```
1 > 1 + 1
  [1] 2
3 > 2^3/4 - 2
  [1] 0
5 > cos(pi)
  [1] -1
7 > exp(1)
  [1] 2.718282
9 > log(10)
  [1] 2.302585
11 > log10(10)
  [1] 1
13 > log(10, base=10)
  [1] 1
```

 Binnen RGui en RStudio is het belangrijkste onderdeel de **Console**. Alle invoer en uitvoer komt hier terecht. Basiscommando's in deze sectie kunnen gewoon in dat venster na de prompt > worden ingetypt. De rest van de grafische interface speelt hier nog geen rol.

Enkele tips die het werken met de commandolijn aangenamer kunnen maken:

- Met de pijltjestoetsen, ↑ en ↓, kunnen eerder ingevoerde commando's terug worden opgeroepen. De invoer `history()` geeft een overzicht van alle gebruikte commando's.
- Met de **Tab**-toets wordt een gedeeltelijk ingevoerd commando automatisch aangevuld, of worden alle mogelijke commando's met deze beginletters getoond.
- R is hoofdlettergevoelig, `abc`, `Abc` en `ABC` zijn drie verschillende namen.
- R negeert spaties en regeleindes, dus deze kunnen naar wens worden tussengevoegd om de invoer leesbaarder te maken.

- Al wat achter een `#`-symbool komt, wordt beschouwd als commentaar.

Uitkomsten van berekeningen kunnen worden toegekend aan variabelen met een gelijkheidsteken (`=`) of pijltjes opgebouwd uit een minteken en een kleiner (`<-`) of groter dan teken (`->`). Toekenningen worden opgeheven met `rm()` en een overzicht van alle actieve veranderlijken wordt verkregen met `ls()`.

```

> a = 1
> a <- 1
> 1 -> a
> b = 2
> a + b
[1] 3
> rm(a)
> a + b
Error: object "a" not found
> ls()
[1] "b"

```

1.2 Vectoren en matrices

Getallen kunnen worden opgeslaan in een geordende lijst, een vector, die kan worden gemaakt met het commando `c()`. Logische rijen kunnen worden gegenereerd met `1:n`, `seq()` of `rep()`. De objecten in deze en voorgaande voorbeelden behoren tot de klasse `numeric`, wat kan opgevraagd worden met het commando `class()`. In wat volgt komen nog andere klassen aan bod en zal het belang van deze indeling in klassen blijken.

```

> c(1, 2, 3)
[1] 1 2 3
> class(c(1, 2, 3))
[1] "numeric"
> length(c(1, 2, 3))
[1] 3
> 1:10
[1] 1 2 3 4 5 6 7 8 9 10
> seq(0, 1, by=.1)
[1] 0.0 0.1 0.2 0.3 0.4 0.5 0.6 0.7 0.8 0.9 1.0
> rep(c(1, 2, 3), times=5)
[1] 1 2 3 1 2 3 1 2 3 1 2 3 1 2 3
> rep(c(1, 2, 3), each=5)
[1] 1 1 1 1 1 2 2 2 2 2 3 3 3 3 3

```

Op vectoren werken `+` en `*` als de gewone optelling van vectoren en de componentgewijze vermenigvuldiging. Wanneer de dimensies van twee vectoren in deze bewerkingen niet overeenstemmen, wordt de kortste vector herhaald tot de vereiste lengte.

```

> x = c(1, 2, 3)
> y = c(3, 4, 5)
> 2 * x
[1] 2 4 6
> x + y
[1] 4 6 8

```



```

46 > x * y
[1] 3 8 15
48 > c(1,1,1,1,1) + c(2,3)
[1] 3 4 3 4 3
50 Warning message:
In c(1, 1, 1, 1, 1) + c(2, 3) :
52 longer object is not a multiple of shorter object

```

De meeste functies (zie Tabel 1.1) in R werken componentgewijs op matrices en vectoren. Bewerkingen als de scalaire vermenigvuldiging en norm zijn niet standaard geïmplementeerd maar kunnen gemakkelijk worden berekend mits samenstelling van commando's. Met `t()` wordt een vector of matrix getransponeerd. Het valt op dat R vectoren beschouwt als kolomvectoren maar niet altijd zo weergeeft.

```

> sin(1:10)
54 [1] 0.8414710 0.9092974 0.1411200 -0.7568025 -0.9589243
[6] -0.2794155 0.6569866 0.9893582 0.4121185 -0.5440211
56 > sum(x*y)
[1] 26
58 > sqrt(sum(x*x))
[1] 3.741657
60 > t(x)
      [,1] [,2] [,3]
62 [1,]    1    2    3
> t(t(x))
64      [,1]
[1,]    1
66 [2,]    2
[3,]    3

```

Hier blijkt dat de aanduiding `[1]` eigenlijk betekent dat het eerstvolgende getal de eerste component van de uitvoer is, `[6]` op de tweede lijn betekent dat de uitvoer daar wordt verder gezet vanaf de zesde component. Deze notatie laat ook toe om één of meerdere elementen uit vectoren te selecteren. Om elementen te elimineren kunnen negatieve indices worden gebruikt.

```

68 > v = (1:10)^2
> v[3]
70 [1] 9
> v[-2]
72 [1] 1 9 16 25 36 49 64 81 100
> v[c(1,1,3,5)]
74 [1] 1 1 9 25
> v[-(2:7)]
76 [1] 1 64 81 100

```

Behalve functies kunnen ook (on)gelijkheden (`==`, `!=`, `>`, `<`, `>=`, `<=`) elementsgewijs worden getest op elke component van een vector. De logische connectieven *en* (`&`), *of* (`|`) en *niet* (`!`) kunnen gebruikt worden om logische tests samen te stellen. De resulterende vector van klasse `logical` kan worden gebruikt om precies die elementen te selecteren die voldoen aan de voorwaarde. De functie `which()` vertelt welke elementen aan de gestelde voorwaarde voldoen.

```

> v == 9
78 [1] FALSE FALSE TRUE FALSE FALSE FALSE FALSE FALSE FALSE

```

```

> v != 9
80 [1] TRUE TRUE FALSE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
> v > 20
82 [1] FALSE FALSE FALSE FALSE TRUE TRUE TRUE TRUE TRUE TRUE
> v > 20 & v < 80
84 [1] FALSE FALSE FALSE FALSE TRUE TRUE TRUE TRUE FALSE FALSE
> v[v > 20 & v < 80]
86 [1] 25 36 49 64
> which(v>20 & v < 80)
88 [1] 5 6 7 8

```

Het commando `array()` zet een getal of een vector van lengte $m \cdot n$ om in een matrix (klasse `matrix`) met dimensies $m \times n$, `cbind()` en `rbind()` vormen matrices door vectoren als kolommen respectievelijk rijen aan elkaar te hangen, `diag()` maakt een diagonaalmatrix van een vector of geeft de diagonaal van een matrix.

```

> array(0, dim=c(2,2))
90      [,1] [,2]
[1,]    0    0
[2,]    0    0
> array(1:6, dim=c(2,3))
94      [,1] [,2] [,3]
[1,]    1    3    5
[2,]    2    4    6
> cbind(x,y)
98      x y
[1,]  1 3
100 [2,]  2 4
     [3,] 3 5
102 > rbind(x, y)
     [,1] [,2] [,3]
x      1    2    3
y      3    4    5
106 > A = array(1:6, dim=c(2,3))
> class(A)
108 [1] "matrix"
> dim(A)
110 [1] 2 3
> diag(A)
112 [1] 1 4
> diag(y)
114      [,1] [,2] [,3]
[1,]    3    0    0
[2,]    0    4    0
116 [3,]    0    0    5

```

📘 Zodra een veranderlijke wordt geïntroduceerd, verschijnt de naam en belangrijkste specificatie in het **Workspace**-venster, zoals bijvoorbeeld

b	2
v	numeric[10]
A	2x3 double matrix

Bij een getal wordt de waarde getoond, bij een vector of matrix het datatype en de lengte of dimensie. Bij aanklikken wordt de corresponderende code getoond.

De manier om elementen uit een object te selecteren hangt af van het datatype 📘. Uit een matrix kunnen met `A[i,]`, `A[,j]` en `A[i,j]` respectievelijk rij i , kolom j en het element $A_{i,j}$ worden geselecteerd. Met dergelijke aanduidingen kunnen zeer flexibel delen van matrices en vectoren worden geselecteerd en aangepast. Logische uitdrukkingen vormen een andere manier om elementen uit een matrix te selecteren.

```

118 > A[2, 3]

```

```

[1] 6
120 > A[2,]
[1] 2 4 6
122 > A[2, c(1,3)]
[1] 2 6
124 > A[A>3] = 0
> A
126      [,1] [,2] [,3]
[1,]     1     3     0
128 [2,]     2     0     0

```

Rekenen met matrices is analoog aan rekenen met vectoren. Speciale aandacht is wel vereist voor de matrixvermenigvuldiging. De asterisk staat opnieuw voor componentgewijze vermenigvuldiging, matrixvermenigvuldiging kan met het minder intuïtieve `%*%`. Andere belangrijke concepten uit de lineaire algebra zijn het berekenen van determinant, eigenstructuur en inverse van een matrix, in R met respectievelijk `det()`, `eigen()` en `solve()`. Dit laatste commando is ook geschikt voor het oplossen van lineaire stelsels.

```

> B = array(c(1,4,2,3,5,2,7,3,1), dim=c(3,3))
130 > B %*% x
      [,1]
132 [1,] 28
    [2,] 23
134 [3,] 9
> A %*% B
136      [,1] [,2] [,3]
[1,]    13    18    16
138 [2,]     2     6    14
> det(B)
140 [1] -9
> eigen(B)
142 $values
[1] 9.1806405 -2.5631142 0.3824737
144 $vectors
      [,1] [,2] [,3]
146 [1,] -0.5570981 -0.8870363 0.6440541
    [2,] -0.7649654 0.3494436 -0.7220589
148 [3,] -0.3232176 0.3017544 0.2526366
> B %*% eigen(B)$vectors[,1]
150 [1] -5.114518 -7.022872 -2.967345
> eigen(B)$values[1] * eigen(B)$vectors[,1]
152 [1] -5.114518 -7.022872 -2.967345
> solve(B)
154      [,1] [,2] [,3]
[1,] 0.1111111 -1.2222222 2.8888889
156 [2,] -0.2222222 1.4444444 -2.7777778
    [3,] 0.2222222 -0.4444444 0.7777778
158 > solve(B,x)
[1] 6.333333 -5.666667 1.666667

```

1.3 Kansen en kwantielen berekenen

Is de verdeling van een stochastische veranderlijke X bekend, dan kunnen kansen en kwantielen worden berekend. Dit gebeurt middels de dichtheids-, verdelings-, en kwantielfunctie.

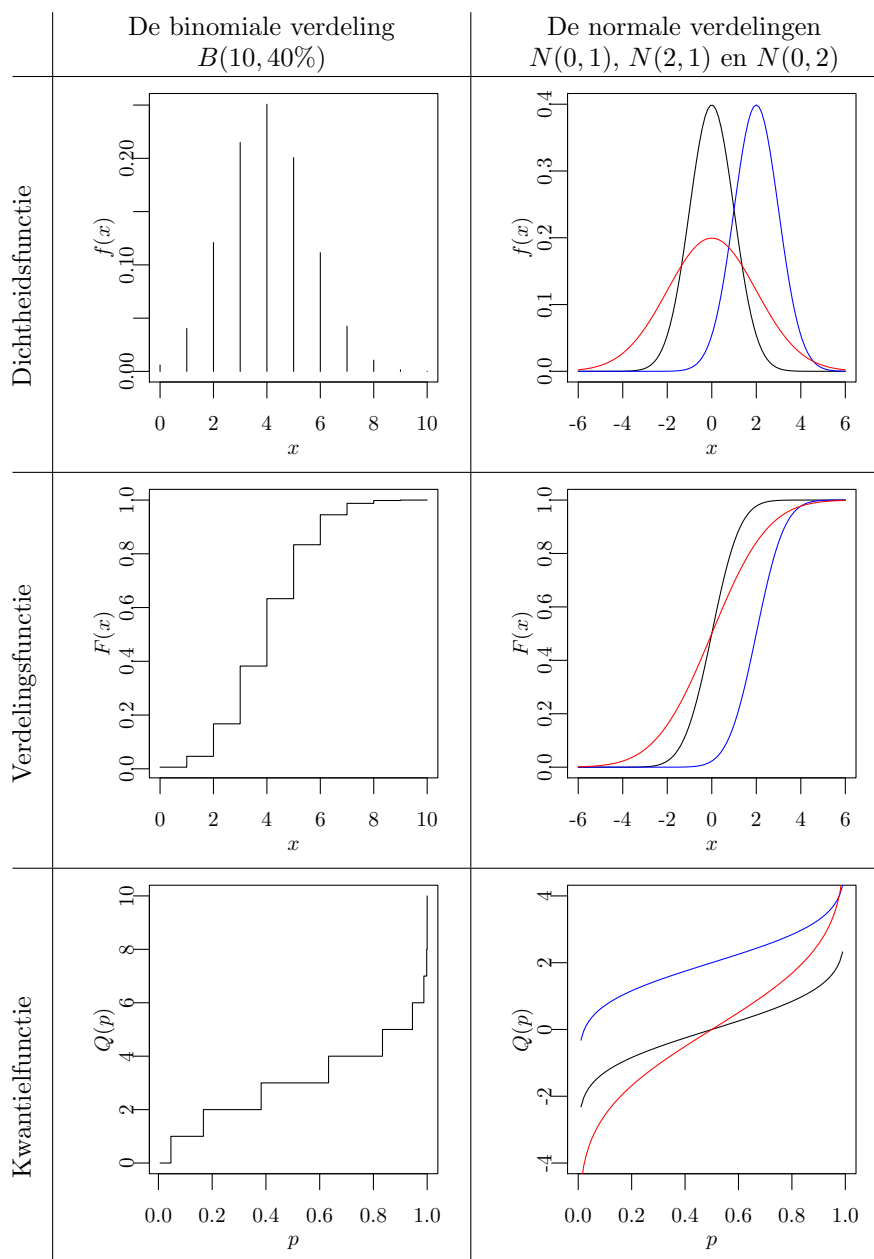
Deze drie karakteriserende functies zijn in R voor de meest voorkomende verdelingen geïmplementeerd. De dichtheidsfunctie begint steeds met de letter **d**, de verdelingsfunctie met een **p** en de naam van de kwantielfunctie wordt voorafgegaan door een **q**. Deze beginletter wordt telkens gevolgd door de naam van de verdeling, zoals te zien in Tabel 1.1. Ter illustratie staan hieronder grafieken van een paar zo'n functies. Details over het maken daarvan volgen in sectie 3.

```
160 > f = dbinom(0:10, 10, .4)
    > F = pbinom(0:10, 10, .4)
162 > rbind(f, F)
      [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9] [,10] [,11]
164 f 0.01 0.04 0.12 0.21 0.25 0.20 0.11 0.04 0.01 0.00 0.00
    F 0.01 0.05 0.17 0.38 0.63 0.83 0.95 0.99 1.00 1.00 1.00
166 > qbinom(F, 10, .4)
      [1] 0 1 2 3 4 5 6 7 8 9 10
168 > plot(0:10, f, type="h", xlab="x")
    > plot(0:10, F, type="s", xlab="x")
170 > plot(F, 0:10, type="s", xlab="p")
    > x = seq(-6, 6, length=100)
172 > plot(c(-6,6), c(0,.4), type="n")
    > lines(x, dnorm(x, 0, 1))
174 > lines(x, dnorm(x, 2, 1), col="blue")
    > lines(x, dnorm(x, 0, 2), col="red")
176 > plot(c(-6,6), c(0,1), type="n")
    > lines(x, pnorm(x, 0, 1))
178 > lines(x, pnorm(x, 2, 1), col="blue")
    > lines(x, pnorm(x, 0, 2), col="red")
180 > p=seq(0, 1, length=100)
    > plot(c(0,1), c(-4,4), type="n")
182 > lines(p, qnorm(p, 0, 1))
    > lines(p, qnorm(p, 2, 1), col="blue")
184 > lines(p, qnorm(p, 0, 2), col="red") # zie Figuur 1.1
```

1.4 Programmeerstructuren

Het is eenvoudig om in R zelf functies te schrijven met `function()`. Dit kunnen eenvoudige ‘wiskundige’ functies zijn, met één of meerdere argumenten, maar even goed meer complexe structuren. De uitkomst is een functie van de klasse `function`. Volgende functie f berekent de standaardnormale dichtheid, de functie g doet hetzelfde, maar heeft optionele argumenten μ en σ , die standaard zijn ingesteld op 0 respectievelijk 1.

```
    > f = function(x) {exp(-x^2/2)/sqrt(2*pi)}
186 > f(1)
      [1] 0.2419707
188 > f(0:3)
```



Figuur 1.1: Voorbeelden van verdelingen

```

[1] 0.398942280 0.241970725 0.053990967 0.004431848
190 > g = function(x, mu=0, sigma=1)
+   {exp(-(x-mu)^2/2/sigma^2)/sqrt(2*pi*sigma^2)}
192 > g(0:3)
[1] 0.398942280 0.241970725 0.053990967 0.004431848
194 > g(0:3, mu=1, sigma=2)
[1] 0.1760327 0.1994711 0.1760327 0.1209854

```

Omdat een functie zelf gebruik kan maken van andere functies en het onmogelijk is om daarbij het gebruik van alle mogelijke opties en argumenten te voorzien, kan in de functiedefinitie met “...” worden aangegeven dat eventuele extra argumenten gewoon moeten worden doorgegeven.

```

196 > smartplot = function(f, a=c(0,10), ...)
+   {x = seq(a[1], a[2], length=100)
198 +   y = f(x)
+   plot(x, f(x), type='l', ...)}
200 > smartplot(sin, col='red')

```

Het is mogelijk om een binaire operator te definiëren. De naam van deze operator staat steeds tussen twee percent-tekens en in de definitie hoort het geheel tussen aanhalingstekens. Volgend voorbeeld definieert het kruisproduct $\vec{a} \times \vec{b} \in \mathbb{R}^3$ van twee vectoren.

```

> "%x%" = function(X,Y)
202 +   {c(X[2]*Y[3] - X[3]*Y[2],
+       X[3]*Y[1] - X[1]*Y[3],
204 +       X[1]*Y[2] - X[2]*Y[1])}
> c(1,0,0) "%x%" c(0,1,0)
206 [1] 0 0 1

```

Om repetitieve taken uit te voeren zijn `for()`- en `while()`-lussen erg geschikt. Volgende lussen zijn louter illustratieve voorbeelden, die eenvoudiger op een andere manier kunnen worden uitgevoerd: het berekenen van de cumulatieve som van een vector (ingebouwd als `cumsum()`) en het zoeken van de eerste index waarvoor de waarde groter is dan 20 (als matrixuitdrukking: `(1:10)[a>20][1]`).

```

> a = 1:10
208 > for (x in 1:10) {a[x] = sum(1:x)}
> a
210 [1] 1 3 6 10 15 21 28 36 45 55
> i = 1
212 > while (a[i]<20) {i = i+1}
> i
214 [1] 6

```

Echter, alleen al de zoekoperaties bij het overlopen van grote datastructuren kunnen zeer tijdsintensief zijn. Het is daarom aanbevolen om bij het overlopen van elementen in grote vectoren of matrices `for()`- en `while()`-lussen zo veel mogelijk te vermijden en in de mate van het mogelijke te vervangen door matrixoperaties. Vergelijk ter illustratie de rekentijd voor grote invoer $n = 10^5, 10^6, \dots$ van volgende functies f en g , die essentieel hetzelfde doen. Voor functies die standaard niet componentsgewijs op een matrixargument werken, kan `sapply()` mogelijk het gebruik van een `for()`-lus omzeilen. Om een functie per rij of per kolom toe te passen is er `apply()`.

```

> f = function(n) {x=1:n; for (i in x) {y[i]=sin(x[i])}}
216 > g = function(n) {x=1:n; y=sin(x)}
> sapply(1:5, sin)
218 [1] 0.8414710 0.9092974 0.1411200 -0.7568025 -0.9589243
> B = array(c(1,4,2,3,5,2,7,3,1), dim=c(3,3))
220 > B
      [,1] [,2] [,3]
222 [1,]    1    3    7
     [2,]    4    5    3
224 [3,]    2    2    1
> apply(B,1,max)
226 [1] 7 5 2
> apply(B,2,max)
228 [1] 4 5 7

```

Voor het uitvoeren van voorwaardelijke opdrachten, ten slotte, bestaat een analoge syntax. Let er op dat de accolades links en rechts van **else** op dezelfde lijn staan. Deze functie `if()` laat echter geen gebruik van vectoren in de conditie toe, het commando `ifelse()` kan daarom soms flexibeler zijn. Volgend voorbeeld geeft een implementatie van de Heaviside-functie voor beide constructies,

$$H : \mathbb{R} \rightarrow \{0,1\} : x \mapsto \begin{cases} 0 & \text{als } x < 0 \\ 1 & \text{als } x \geq 0. \end{cases}$$

```

> H = function(x) { if (x<0) {0} else {1} }
230 > H(3)
     [1] 1
232 > H(-3:3)
     [1] 0
234 Warning message:
In if (x < 0) : [...] only the first element will be used
236 > H = function(x) { ifelse(x<0, 0, 1) }
> H(-3:3)
238 [1] 0 0 0 1 1 1 1

```

Volgend voorbeeld combineert voorgaande technieken voor het berekenen van de grootste gemene deler. Merk het gebruik van `return()` op, dat de uitvoer van een functie specificeert in het geval deze niet in de laatste regel wordt berekend.

```

> gcd = function(a,b) {
240 +   if (a==0) {
+     return(b)
242 +   } else {
+     while (b!=0) {
244 +       if (a>b) {
+         a = a - b
246 +       } else {
+         b = b - a } }
+     return(a) } }
248 > gcd(12,30)
250 [1] 6

```

❷ In RStudio kan de werkmapij worden ingesteld door rechts in het venster **File** op de knop ... te drukken, de gewenste map te selecteren en daarna via **More** te klikken op **Set As Working Directory**.

❸ RStudio heeft een ingebouwde editor die speciaal bedoeld is voor het schrijven van .R-scripts. Deze vergemakkelijkt het werken door functies en parameters in een afzonderlijke kleur te zetten, overeenkomstige haakjes aan te duiden, gekende commando's aan te vullen en pop-up's met help-informatie te tonen door op **Tab** te drukken. Een nieuw script kan worden gestart via het menu **File** of door de toetscombinatie **Ctrl+N** en wordt opgeslaan met **Ctrl+S**. Het huidige commando kan vanuit het script worden uitgevoerd door de toetsencombinatie **Ctrl+Enter**. Omgekeerd kunnen commando's die via de commandolijst zijn ingevoerd naar het script worden gekopieerd door deze in het **History**-venster te selecteren en op de knop **To Source** te drukken.

❹ In het venster **Workspace** kunnen alle bestaande veranderlijken worden weggeschreven naar een .RData-bestand met de knop **Save**. Zo een bestand inladen kan met de knop **Load**.

1.5 De programma-omgeving

Het programma R kan commando's en gegevens in een bestand inlezen of wegschrijven. Daartoe is het belangrijk in te stellen in welke map deze bestanden staan: de zogenaamde werkmapij. Met het commando `getwd()` kan de huidige werkmapij worden opgevraagd, met `setwd()` kan een andere locatie worden ingesteld ❶. Hieronder wordt als werkmapij `H:\statistiek` ingesteld in de veronderstelling dat deze al een bestand `script.R` bevat. De inhoud van de werkmapij kan worden opgevraagd met `dir()`. Let er op dat Windows-gebruikers elke backslash in een map-adres moeten vervangen door een gewone slash.

```
> getwd()
252 [1] "C:/Users/u0041551/Documents"
> setwd("H:/statistiek")
254 > dir()
[1] "script.R"
```

Het programma houdt permanent een lijst bij van de laatst uitgevoerde commando's. Deze kunnen met het commando `history()` worden opgeroepen. De lijst van alle commando's die tijdens het experimenteren worden ingetypt is meestal niet relevant. Voor een systematisch onderzoek is het aangewezen om een overzichtelijke reeks commando's met bijhorende commentaar te bundelen en op te slaan in een afzonderlijk bestand, een zogenaamd script: een tekstbestand waarvan de bestandsnaam als extensie `.R` heeft ❷. Zo een script kan in zijn geheel worden uitgevoerd vanaf de commandolijst met `source()`.

```
256 > source("script.R", echo=TRUE)
```

Een script bevat dus enkel een lijst commando's. Het is ook mogelijk om waarden uit het werkgeheugen op te slaan, door de corresponderende variabelen naar een bestand weg te schrijven met `save()/save.image()`, en terug in te laden, met `load()` ❸. Data uit R wordt opgeslaan met extensie `.RData`.

```
> ls()
258 [1] "a" "A" "b" "x" "y"
> save("A", "A.RData")
260 > save.image("alles.RData")
> rm(list=ls())
262 > load("A.RData")
> ls()
264 [1] "A"
> load("alles.RData")
266 > ls()
[1] "a" "A" "b" "x" "y"
```

Het allerbelangrijkste commando is ongetwijfeld `help()`, kortweg `?`, waar elk commando uitgebreid wordt toegelicht met al zijn opties en enkele voorbeelden. Het commando `help.search()` geeft alle help-secties die een bepaalde term behandelen.

```
268 > help("log")
> ?log
270 > help.search("logarithm")
```

Er zijn talloze extra commando's beschikbaar die niet automatisch vanuit R aan te spreken zijn. Om deze toch te kunnen gebruiken moet de juiste bibliotheek worden ingeladen met het commando `library()`. Soms zal het nodig zijn

deze extra bibliotheek eerst op de computer te installeren met het commando `install.packages()` ④ Het is aangewezen de optie `dependencies=TRUE` te gebruiken, zodat eventuele andere vereiste pakketten ook automatisch worden geïnstalleerd. De gebruiker moet daarna een *mirror* naar keuze kiezen, best de dichtstbijzijnde. Op sommige systemen is het enkel als administrator mogelijk extra pakketten te installeren.

```
|> install.packages('car', dependencies=TRUE)
```

Samenvattend, de bestandstypes ④ eigen aan R zijn de volgende:

- **.R-bestanden:** scripts. Maken en bewerken in een editor. Als geheel uitvoeren met `source()`.
- **.RData-bestanden:** gegevens. Wegschrijven en inlezen van gegevens met `save()` respectievelijk `load()`.
- **.RHistory-bestanden:** eerder uitgevoerde commando's. Tonen, wegschrijven en inlezen met `history()`, `loadhistory()` en `savehistory()`.
- **.tar.gz-, .zip- of .tgz-bestanden:** pakketten. Installeren en inladen met `install.packages()` en `library()`.

④ Pakketten installeren in de grafische interface kan via **Tools, Install Packages** of door de gewenste pakketten aan te vinken in het tabblad **Packages** en te klikken op **Install**.

④ Er zijn drie **load-knoppen** in **RStudio**:

- **Scripts:** links bovenaan in de knoppenbalk.
- **Data:** rechts bovenaan in het tabblad **Environment**.
- **History:** rechts bovenaan in het tabblad **History**.

Verwar deze niet, want dat leidt tot onverwachte resultaten.

Tabel 1.1: Aan de slag in R

```

## basisbewerkingen
2 abs(), sign(), sqrt(),      # basisfuncties
  exp(), log(), log10()
4 cos(), sin(), tan(), acos(), asin(), atan()
  cosh(), sinh(), tanh(), acosh(), asinh(), atanh()
6 factorial(), choose(),     # combinatoriek
  combn()
8 +, -, *, /, ^             # basisbewerkingen
  &, |, !, ==, <, <=, >=, > # logische operatoren
10 Arg, Conj, Im, Mod, Re    # complexe getallen
  %%, %/%                   # mod en div
12 ## vectoren en matrices
  :, c(), seq(), rep()      # vectoren en matrices
14 length(), sum(), prod(), sort()
  array(), cbind(), rbind(), diag()
16 t(), dim(), det(), eigen(), solve()
## verdelingsrekenen
18 d<dist>(...)              # dichtheidsfunctie
  p<dist>(...)              # verdelingsfunctie
20 q<dist>(...)              # kwantielfunctie
  dbinom(x,n,p),           pbinom(x,n,p),       qbinom(q,n,p)
22 dpois(x,lambda),         ppois(x,lambda),     qpois(q,lambda)
  dgeom(x,p),              pgeom(x,p),          qgeom(q,p)
24 dnorm(x,mu,sigma),       pnorm(x,mu,sigma),   qnorm(q,mu,sigma)
  dexp(x,lambda),          pexp(x,lambda),       qexp(q,lambda)
26 dt(x,df),               pt(x,df),             qt(q,df)
  df(x,df1,df2),           pf(x,df1,df2),       qf(q,df1,df2)
28 dchisq(x,df),           pchisq(x,df),        qchisq(q,df)
## programmeerstructuren
30 function(X,Y=y0,...) {...} # functie schrijven
  "%x%" = function(X,Y){...} # binaire operator
32 for (i in 1:n) {...}      # for-lus
  while (x != 0) {...}      # while-lus
34 ifelse (x != 0, ..., ...) # if-then-else
  if (x != 0) {...} else {...}
36 ## de programma-omgeving
  getwd(), setwd(), dir(), ls(), rm()
38 load(), save(), save.image(), source()
  history(), loadhistory(), savehistory()
40 help(), help.search()
  library(), install.packages()

```

Hoofdstuk 2

Statistische datasets

2.1 De gegevensmatrix

Verzamelde steekproefgegevens bevatten informatie over verschillende eigenschappen (*variables*) van een verzameling onderzochte individuen (*cases*) uit de populatie. Deze moeten voor statistische verwerking altijd in een gegevensmatrix (*case variable dataset*) worden gestructureerd, met in elke rij één case en elke kolom één veranderlijke, en zeker niet als kruistabel, zoals te zien in Tabel 2.1.

2.2 Soorten variabelen

Naargelang het type veranderlijke zullen andere statistische methoden van toepassing zijn. Voor het vervolg is het dus belangrijk een duidelijk onderscheid te voeren tussen die verschillende types.

Kwantitatief of numeriek, de uitkomsten zijn getallen:

Continu, tussen twee uitkomsten is steeds nog een derde denkbaar;

Discreet, het aantal mogelijke uitkomsten is aftelbaar.

Tabel 2.1: Gegevens geordend als gegevensmatrix en als kruistabel
(a) Gegevensmatrix (b) Kruistabel

Studies	Geslacht	Aantal		Man	Vrouw
Wiskunde	Man	17	Wiskunde	17	4
Informatica	Man	9	Informatica	9	0
Fysica	Man	26	Fysica	26	1
Chemie	Man	17	Chemie	17	17
Biologie	Man	12	Biologie	12	1
Wiskunde	Vrouw	4			
Informatica	Vrouw	0			
Fysica	Vrouw	1			
Chemie	Vrouw	17			
Biologie	Vrouw	1			

Kwalitatief of categorisch, de uitkomsten zijn geen getallen:

Ordinaal, geordende klassen;

Nominaal, ongeordende klassen.

In sectie 2.9 blijkt dat met elke soort veranderlijke een klasse in R is geassocieerd.

2.3 Gegevens ingeven

Een dataset aanmaken gebeurt door veranderlijke per veranderlijke een vector te maken met daarin de meetwaarden. Vervolgens brengt het commando `data.frame()` deze vectoren samen tot een gegevensmatrix van de gelijknamige klasse `data.frame`. De namen van de vectoren worden gebruikt als namen voor de veranderlijken. Als voorbeeld worden de gegevens uit Tabel 2.1 in R ingegeven.

```
2 > studie = rep(c("Wis","Inf","Fys","Che","Bio"), times=2)
2 > geslacht = rep(c("M","V"), each=5)
> aantal = c(17,9,26,17,12,4,0,1,17,1)
4 > gegevens = data.frame(studie, geslacht, aantal)
> gegevens
6   studie geslacht aantal
8 1     Wis         M     17
9 2     Inf         M      9
10 3     Fys         M     26
11 4     Che         M     17
12 5     Bio         M     12
13 6     Wis         V      4
14 7     Inf         V      0
15 8     Fys         V      1
16 9     Che         V     17
17 10    Bio         V      1
> class(gegevens)
18 [1] "data.frame"
```

2.4 Gegevens inlezen

Doorgaans zullen gegevens niet binnen R worden ingetypt maar in een of ander bestandsformaat worden aangeboden. Deze handleiding beschrijft de nodige stappen om vertrekkend van een tekstbestand datasets in te lezen met het commando `read.table()`. Met aangepaste `read`-commando's is het mogelijk om diverse andere bestandsformaten in te voeren, Tabel 2.2 en de helpbestanden van R geven meer details.

Eerst moet met `setwd()` de werkmapij, *working directory*, worden ingesteld, de locatie waar de gegevens zich bevinden. Vervolgens moet in het tekstbestand nauwkeurig worden nagegaan hoe de gegevens zijn gestructureerd om volgende opties van het `read.table()`-commando correct te kunnen instellen.

header=TRUE|FALSE: Bevat de eerste rij in het tekstbestand de namen van de variabelen?

col.names=VEC: Indien de eerste rij geen namen bevat, kan een vector **VEC** met namen voor de veranderlijken worden meegegeven.

row.names=VAR: Duidt aan dat de veranderlijke **VAR** de namen van de cases bevat.

sep=",": Bepaalt welk symbool in het tekstbestand als scheidingsteken tussen de kolommen wordt gebruikt, "\t" voor een tabulator, " " voor een spatie, ",", voor een komma, ...

na.strings="NA": Bepaalt welke markering in het tekstbestand wordt gebruikt voor ontbrekende waarden.


dec=".": Bepaalt welk symbool in het tekstbestand wordt gebruikt als decimaalteken.

In de rest van deze tekst zal de pollutie-dataset uit het handboek [1] gebruikt worden. Deze is terug te vinden op

lib.stat.cmu.edu/DASL/Datafiles/SMSA.html.

en dient te worden opgeslaan in de werkmap, bijvoorbeeld als volgt,

H:\statistiek\pollutie.txt.

Vooraleer de data in te laden , wordt de structuur van `pollutie.txt` in detail bekeken:

- de eerste rij bevat de namen van de variabelen,
- een vector met kolomnamen is dus overbodig,
- de variabele `city` bevat de naam van elke case,
- de kolommen worden gescheiden door een tabulator,
- de ontbrekende waarden zijn blanco cellen,
- het decimaalteken is een punt.

 In RStudio volstaat het om in het scherm **Workspace** op de knop **Import Dataset** te klikken om een bestand in te lezen. Een dialoogschermbijlage laat dan toe om het effect van de verschillende opties onmiddellijk te zien. Het resulterende commando kan dan uit het **History**-scherm naar het script worden verplaatst.

```
> rm(list=ls())
20 > getwd()
[1] "C:/Windows/system32"
22 > setwd("H:/statistiek")
> pollutie = read.table("pollutie.txt",
24 + header=TRUE, row.names="city",
+ sep="\t", na.strings="", dec=".")
26 > pollutie
              JanTemp JulyTemp RelHum
28 Akron, OH          27         71     59
Albany-Schenectady-Troy, NY      23         72     57
30 Allentown, Bethlehem, PA-NJ    29         74     54
> names(pollutie)
32 [1] "JanTemp" "JulyTemp" "RelHum" ...
> row.names(pollutie)
34 [1] "Akron, OH" "Albany-Schenectady-Troy, NY" ...
```

2.5 Gegevens selecteren en sorteren

Cases, variables en individuele meetwaarden kunnen worden aangesproken net als rijen, kolommen en cellen in een matrix. Bovendien is het mogelijk om een variabele `VAR` uit een gegevensmatrix `gegevens` te selecteren als `gegevens$VAR`. Met behulp van de gepaste logische test kan een selectie gemaakt worden binnen een gegevensmatrix.

```
> pollutie[1,]
36      JanTemp JulyTemp RelHum ...
Akron, OH      27      71      59 ...
38 > pollutie[1,3]
[1] 59
40 > pollutie$RelHum
[1] 59 57 54 56 ...
42 > pollutie$RelHum[1]
[1] 59
44 > pollutie[pollutie$JanTemp>50,]
      JanTemp JulyTemp RelHum ...
46 Houston, TX      55      84      59 ...
Los Angeles, Long Beach, CA      53      68      47 ...
48 Miami-Hialeah, FL      67      82      60 ...
New Orleans, LA      54      81      62 ...
50 San Diego, CA      55      70      61 ...
> max(pollutie$JulyTemp-pollutie$JanTemp)
52 [1] 61
> pollutie[pollutie$JulyTemp-pollutie$JanTemp==61,]
54      JanTemp JulyTemp RelHum ...
Minneapolis-St. Paul, MN-WI      12      73      58 ...
```

Een vector sorteren kan met het commando `sort()`, maar om een gegevensmatrix volgens een welbepaalde veranderlijke te sorteren, is `order()` aangewezen, welke per index i de rang geeft waar de i -de kleinste waarde zich bevindt. Zodoende zal de uitdrukking `x[order(x)]` een vector genereren met dezelfde lengte als `x`, en als i -de component telkens de i -de kleinste waarde van `x`, formeel: `x[order(x)] \equiv sort(x)`.

```
56 > sort(pollutie$JanTemp)
[1] 12 20 23 23 24 24 24 24 24 25 26 26 27 27 27 ...
58 > order(pollutie$JanTemp)
[1] 34 33 2 54 9 20 22 52 58 45 12 53 1 10 19 ...
60 > row.names(pollutie)[34]
[1] "Minneapolis-St. Paul, MN-WI"
62 > pollutie[order(pollutie$JanTemp),]
      JanTemp JulyTemp RelHum ...
64 Minneapolis-St. Paul, MN-WI      12      73      58 ...
Milwaukee, WI      20      69      64 ...
66 Albany-Schenectady-Troy, NY      23      72      57 ...
Utica-Rome, NY      23      71      60 ...
68 Buffalo, NY      24      70      61 ...
...
```

In het voorbeeld is via `sort()` te zien dat de kleinste meetwaarde 12 Fahrenheit is en, via `order()`, dat deze in rij 34 van de gegevensmatrix voorkomt. De


stad met de laagste wintertemperatuur is dus blijkbaar *Minneapolis -St. Paul*, de eerste stad in de volgens `JanTemp` gesorteerde lijst.

2.6 Veranderlijken aanpassen

Een veranderlijke aanpassen kan door ze te overschrijven, hieronder geïllustreerd door de veranderlijken naar SI-eenheden te converteren. Een extra veranderlijke maken kan door toekenning aan een nieuwe naam `gegevens$nieuw`. Door een toekenning van `NULL` kan een veranderlijke worden verwijderd.

```
70 > pollutie$JanTemp = 5/9*(pollutie$JanTemp-32)
71 > pollutie$JulyTemp = 5/9*(pollutie$JulyTemp-32)
72 > pollutie$Rain = pollutie$Rain*2.54
73 > pollutie$PopDensity = pollutie$PopDensity/2.58998811
74 > pollutie$Temp = (pollutie$JanTemp + pollutie$JulyTemp)/2
75 > pollutie$JanTemp
76 [1] -2.77777778 -5.0000000 -1.6666667 7.2222222 ...
77 > pollutie$Temp
78 [1] 9.444444 8.611111 10.833333 16.666667 13.333333 ...
79 > pollutie$Temp = NULL
80 > pollutie$Temp
NULL
```

2.7 Kwalitatieve veranderlijken

Het bestand `pollutie-regiocode.RData` bevat een vector met de geografische ligging van de steden in de dataset `pollutie`. Als dit bestand in de werkmapp staat, kan het worden ingelezen met `load()` . Dit resulteert in een nieuwe veranderlijke `regiocode`s die echter nog een verkeerde waarde `M` bevat en van de klasse `character` is, bedoeld voor tekstvelden waarop geen statistiek wordt toegepast. Daarom wordt de foutieve waarde eerst overschreven, vooraleer `regiocode`s als een nieuwe, categorische veranderlijke aan de dataset `pollutie` toe te voegen. Door de veranderlijke met `as.factor()` te converteren naar het type `factor`, beschouwt R deze als een nominale variabele waarvoor de gepaste statistische methodes van toepassing zullen zijn. De mogelijke uitkomsten vormen een attribuut, `$levels`, van de categorische veranderlijke. Aangezien een nominale veranderlijke geen natuurlijke ordening heeft, zal R de labels alfabetisch ordenen.

```
82 > load("pollutie-regiocode.RData")
83 > ls()
84 [1] "pollutie" "regiocode"
85 > class(regiocode)
86 [1] "character"
87 > table(regiocode)
88 regiocode
89  C  M  N NO  W ZO
90  9  1 15 24  6  5
91 > regiocode[regiocode=='M'] = 'C'
92 > pollutie$regio = as.factor(regiocode)
93 > table(pollutie$regio)
```

 Een `.RData`-bestand kan in RStudio rechtstreeks ingeladen worden door er op te klikken in het **Workspace**-venster. Er wordt bevestiging van de gebruiker gevraagd om overschrijven van bestaande gegevens te vermijden. Het corresponderende `load()`-commando verschijnt automatisch in de console.

```

94 |   C   N  NO   W  ZO
    | 10 15 24   6   5
96 | > attributes(pollutie$regio)
    | $levels
98 | [1] "C"  "M"  "N"  "NO" "W"  "ZO"
    | $class
100 | [1] "factor"

```

Soms is het ook nuttig een continue veranderlijke te categoriseren, te verdelen in een beperkt aantal klassen. De veranderlijke `X.WC`, die het percentage bedienden in een stad voorstelt, kan bijvoorbeeld worden omgezet en gediscrètiseerd tot een nieuwe variabele `arbeid` met volgende drie categorieën:

- hoog – meer dan 58% arbeiders;
- gemiddeld – tussen 48% en 58% arbeiders;
- laag – minder dan 48% arbeiders.

Met `cut(VAR, c(a_0, ..., a_n))` kan een continue veranderlijke in halfopen intervallen $]a_{i-1}, a_i]$, $i \in \{1, \dots, n\}$, worden opgedeeld. Waarden buiten het interval $]a_0, a_n]$ worden als ontbrekend gemarkeerd (NA). Nuttige opties hier zijn `labels` om de klassen namen te geven, `ordered_result` om een ordinale veranderlijke (klasse `ordered factor`) te bekomen, en `include.lowest` om het meest linkse interval te sluiten.

```

102 | > pollutie$arbeid = cut(100-pollutie$X.WC,
    |   c(0,48,58,100),
    |   + labels=c("laag","gemiddeld","hoog"),
104 |   + ordered_result=TRUE, include.lowest=TRUE)
    | > pollutie$arbeid
106 | [1] gemiddeld gemiddeld hoog gemiddeld gemiddeld ...
    | Levels: laag < gemiddeld < hoog
108 | > class(pollutie$arbeid)
    | [1] "ordered" "factor"
110 | > sort(pollutie$arbeid)
    | [1] laag laag laag laag gemiddeld gemiddeld ...
112 | Levels: laag < gemiddeld < hoog

```

Als tweede voorbeeld wordt `income` gecategoriseerd tot de nieuwe veranderlijke `inkomen`:

- 1 – mediaan inkomen < 30000 dollar;
- 2 – 30000 dollar < mediaan inkomen < 35000 dollar;
- 3 – 35000 dollar < mediaan inkomen < 40000 dollar;
- 4 – 40000 dollar < mediaan inkomen.

De uitkomstenverzameling van de resulterende veranderlijke telt dan wel enkel de getallen 1 tot 4, het resultaat is een categorische veranderlijke. Aangezien deze getallen slechts labels zijn, weigert R dan ook om het gemiddelde te berekenen.


```

> pollutie$inkomen = cut(pollutie$income,
114 + c(0,30000,35000,40000,Inf), labels=1:4,
+ ordered_result=TRUE, include.lowest=TRUE)
116 > pollutie$inkomen
[1] 2 3 3 3 3 2 4 <NA> 3 2 ...
118 Levels: 1 < 2 < 3 < 4
> mean(inkomen)
120 [1] NA
Warning message:
122 In mean.default(inkomen) : argument is not numeric ...

```

2.8 Gegevens klaar voor gebruik

Eens een dataset volledig op punt staat, is het mogelijk om er met het commando `attach()` voor te zorgen dat elke veranderlijke `gegevens$VAR` rechtstreeks via `VAR` kan worden aangeroepen. Dit commando maakt eigenlijk een kopie van de veranderlijke `gegevens$VAR` naar een nieuw object `VAR`, wat voor verwarring kan zorgen als de dataset daarna alsnog wordt aangepast of als er verschillende datasets worden gebruikt waarin de zelfde namen voorkomen. Is het toch nodig om nog aanpassingen te doen, dan is er het commando `detach()` om de verkorte versie `VAR` van de variabelen te verwijderen.

De gegevensverzameling wordt ten slotte weggeschreven naar een `.RData`-bestand voor later gebruik.

```

> RelHum
124 Error: object 'RelHum' not found
> attach(pollutie)
126 > RelHum
[1] 59 57 54 56 55 ...
128 > save(pollutie, file="pollutie-gegevens.RData")

```

2.9 Classificatie van variabelen

De soorten veranderlijken corresponderen met klassen in R als volgt.

numeric – continue veranderlijke.

integer – discrete veranderlijke. Met `as.integer()` kan een numerieke veranderlijke expliciet als discreet worden ingesteld.

factor – nominale veranderlijke. De verschillende waarden, *levels*, worden aan het einde van de lijst in alfabetische volgorde opgesomd:

Levels: C M N NO W ZO.

ordered factor – ordinale veranderlijke. De verschillende waarden worden telkens in oplopende volgorde opgesomd:

Levels: laag < gemiddeld < hoog.

character – tekstvelden. Dit zijn bijvoorbeeld *case names* of opmerkingen en bevatten gegevens die niet bedoeld zijn voor statistische bewerkingen.

Het is steeds noodzakelijk na te gaan of de klasse van een veranderlijke klopt, zodat de gebruikte methoden correct werken.

Tabel 2.2: Statistische datasets in R

```

## gegevens inlezen
2 gegevens = read.table(      # naam toekennen aan dataset
  "gegevens.txt",           # bestand inladen
4   header=FALSE,            # eerste rij bevat rijkoppen
  sep="\t",                  # kolomscheidingsteken
6   na.strings="NA",         # ontbrekende waarden
  dec=".")                   # decimaalteken
8 names(gegevens)            # namen van variabelen
row.names(gegevens)         # case names toekennen
10 gegevens$NAME             # veranderlijke selecteren
attach(gegevens)            # NAME = gegevens$NAME
12 detach(gegevens)          # gegevensmatrix ontkoppelen
## cases manipuleren
14 gegevens[52,]             # case oproepen
gegevens[row.names(gegevens)=="ABC",]
16 gegevens[52,] = NULL      # case wissen
rbind(gegevens,c(...))      # case toevoegen
18 gegevens[gegevens$CTU>3,] # selectie maken
sort(gegevens$ORD)           # veranderlijke sorteren
20 gegevens =                # data sorteren
  gegevens[order(gegevens$VAR1,gegevens$VAR2),]
22 ## veranderlijke manipuleren
gegevens[,6]                 # veranderlijke oproepen
24 gegevens$NAME             # veranderlijke oproepen
gegevens$NAME = NULL         # veranderlijke verwijderen
26 gegevens$TOT =            # nieuwe veranderlijke maken
  gegevens$CTU1 + gegevens$CTU2
28 factor(gegevens$NOM,      # nominale var maken
  levels=c(1,2,3), labels=c('rood','geel','blauw'))
30 factor(gegevens$ORD,      # ordinale var maken
  levels = c(1,2,3), labels=c('small','medium','large'),
32   ordered=TRUE)
as.integer(gegevens$INT)     # discrete var maken
34 rank(gegevens$ORD)         # rangveranderlijke maken
cut(gegevens$CTU,            # veranderlijke categoriseren
36   seq(min(gegevens$CTU),max(gegevens$CTU),length.out=3),
  labels=c('rood','geel','blauw'))
38 ## gegevens controleren
class(gegevens)              # "data.frame"
40 length(gegevens)           # aantal veranderlijken
dim(gegevens)                # afmetingen van de matrix
42 class(NOM)                 # "factor"
class(ORD)                   # "ordered" "factor"
44 class(INT)                 # "integer"
class(CTU)                   # "numeric"

```


Hoofdstuk 3

Beschrijvende statistiek

De lijst met commando's in Tabel 3.1 aangevuld met de ingebouwde `?-help`, is voldoende om de meest gangbare statistieken te berekenen en grafieken te genereren. In de praktijk zullen de klasse-afhankelijke methodes `summary()` en `plot()` zelfs meestal volstaan. Hieronder toch nog een bondige bespreking.

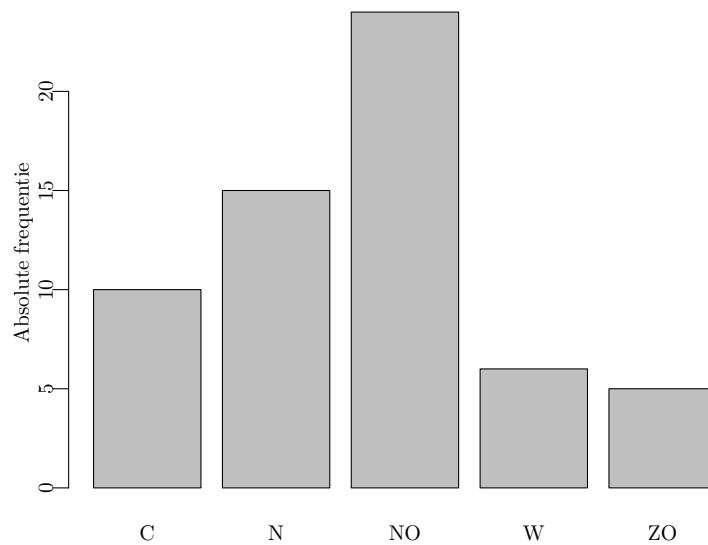
3.1 Frequenties van discrete veranderlijken

Het commando `table()` berekent een absolute frequentietabel of kruistabel naargelang de vorm van het argument. Er is geen rechtstreeks commando voor een tabel met relatieve frequenties, maar deze kan uiteraard eenvoudig worden berekend uit het aantal meetwaarden met `length()`. Het resultaat van `table()` is een object van de klasse `table`, waarop nog andere methoden van toepassing zijn, zoals `cumsum()` voor cumulatieve frequenties en `barplot()` voor een staaf- of Pareto diagram. Hoewel ze in de praktijk vaak worden gebruikt, zijn taartdiagrammen (`pie()`) minder geschikt om gegevens mee voor te stellen.

Om de commando's in onderstaande voorbeelden niet te overbelasten, zijn de gebruikte opties in de afgedrukte code tot een minimum beperkt. De figuren die in de handleiding zijn opgenomen, zijn daarentegen zo goed mogelijk afgewerkt. Hierdoor kan de output in R verschillen van die in deze tekst. Details over het aanpassen van de lay-out, bereik, astitels, ... en het wegschrijven van grafieken via de commandolijn is te vinden in sectie 3.7 .

```
> table(regio)
2 regio
  C  N NO  W ZO
4 10 15 24  6  5
> class(table(regio))
6 [1] "table"
> table(regio)/length(regio)
8 regio
      C      N      NO      W      ZO
10 0.16667 0.25000 0.40000 0.10000 0.08333
> barplot(table(regio)) # zie Figuur 3.1
```

 Figuren kunnen via de grafische interface worden opgeslaan in diverse formaten door op de knop **Export** te klikken. Het verdient aanbeveling te kiezen voor een vectorafbeelding (`.pdf`) en *niet* voor een rasterafbeelding (`.jpg`).



Figuur 3.1: Staafdiagram van de kwalitatieve veranderlijke **regio**

3.2 Frequenties van continue veranderlijken

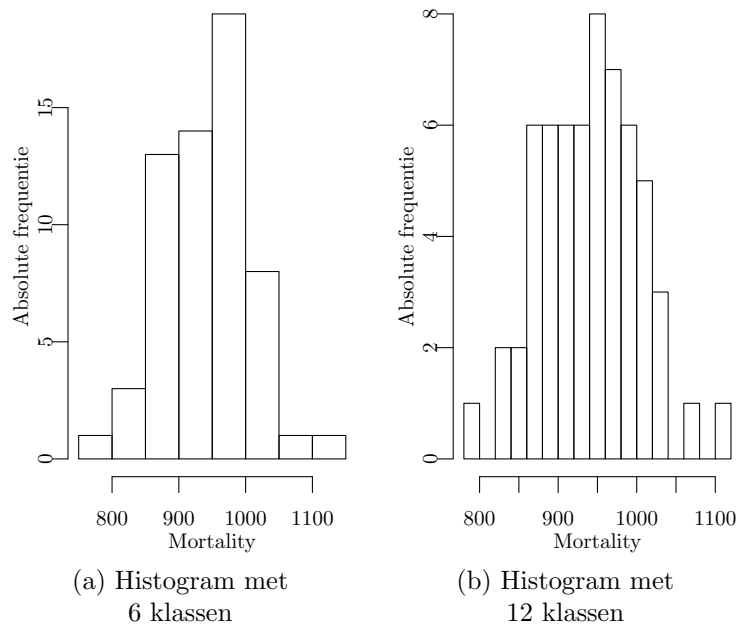
Om een frequentietabel voor een kwantitatieve veranderlijke op te stellen dient deze eerst te worden gecategoriseerd. Daarna zijn dezelfde methodes als eerder toepasbaar. Het commando `hist()` maakt een histogram van een continue veranderlijke, waarvoor het deze zelf in een aantal klassen verdeelt.

```
12 > table(cut(Mortality, 6))
      (790,844]      (844,898]      (898,952]
14           4           11           17
      (952,1.01e+03] (1.01e+03,1.06e+03] (1.06e+03,1.11e+03]
16           20           6           2
18 > table(cut(Mortality, 6))/length(Mortality)
      (790,844]      (844,898]      (898,952]
20 0.06666667 0.18333333 0.28333333
      (952,1.01e+03] (1.01e+03,1.06e+03] (1.06e+03,1.11e+03]
22 0.33333333 0.10000000 0.03333333
22 > hist(Mortality, breaks=6)
23 > hist(Mortality, breaks=12) # zie Figuur 3.2
```

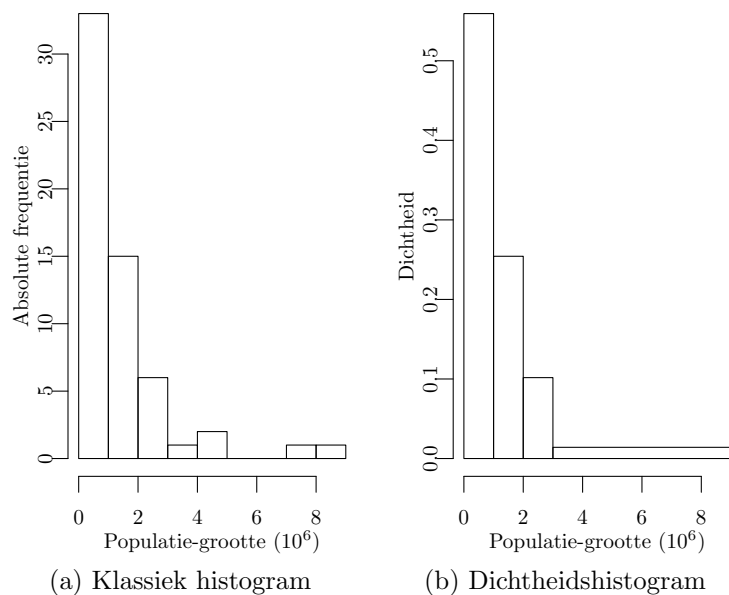
Bovendien definieert men voor kwantitatieve veranderlijken het klassiek histogram en het dichtheidshistogram. Van zodra de klassebreedten ongelijk zijn, kiest R automatisch voor een dichtheidshistogram, enkel dan stellen de oppervlakten in de grafiek immers kansen voor.

```
24 > hist(pop/10^6)
25 > hist(pop/10^6, breaks=c(0,1,2,3,9)) # zie Figuur 3.3
26 > hist(pop/10^6, breaks=c(0,1,2,3,9), freq=TRUE)
Warning message: ... AREAS in the plot are wrong ...
```

Waar de histogrammen een beeld geven van de dichtheidsfunctie, kan met `ecdf()` de empirische cumulatieve dichtheidsfunctie gegenereerd worden. Het



Figuur 3.2: Histogrammen met verschillende klassebreedte van **Mortality**



Figuur 3.3: Klassiek en dichtheidshistogram van de populatie-grootte

resultaat hiervan is van de klasse `function` die kan worden getoond met `plot()`.

```
28 > ecdf(Mortality)
    Empirical CDF
30 Call: ecdf(Mortality)
    x[1:60] = 790.73, 823.76, 839.71, ..., 1071.3, 1113.2
32 > plot(ecdf(Mortality)) # zie Figuur 3.4
33 > class(ecdf(Mortality))
34 [1] "ecdf" "stepfun" "function"
```

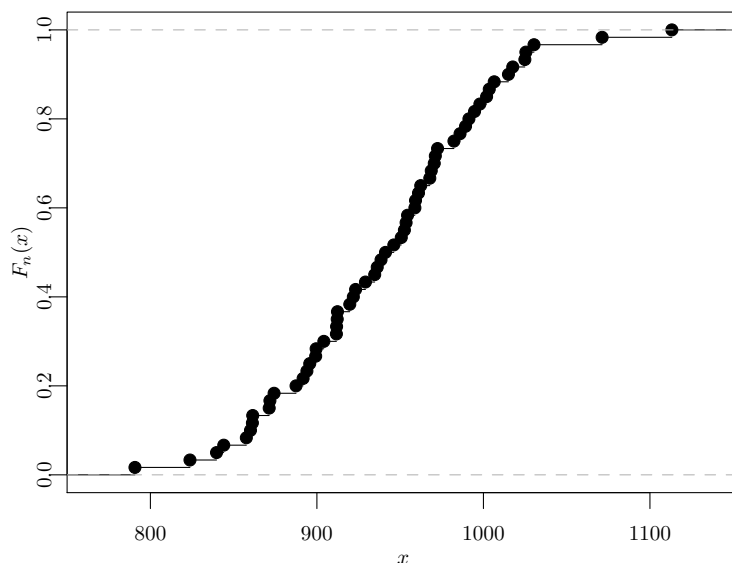
3.3 Steekproefstatistieken

De meeste steekproefstatistieken hebben voor de hand liggende commando's, maar steeds is voorzichtigheid geboden. Indien bepaalde waarden ontbreken (NA), dienen deze immers expliciet te worden uitgesloten met de optie `na.rm`.

```
35 > mean(income)
36 [1] NA
37 > mean(income, na.rm=TRUE)
38 [1] 33246.66
39 > mean(income, trim=0.05, na.rm=TRUE)
40 [1] 32968.04
41 > weighted.mean(income, pollutie$pop, na.rm=TRUE)
42 [1] 34748.11
43 > median(income, na.rm=TRUE)
44 [1] 32452
45 > var(income, na.rm=TRUE)
46 [1] 20008579
47 > sd(income, na.rm=TRUE)
48 [1] 4473.095
49 > range(income, na.rm=TRUE)
50 [1] 25782 47966
51 > diff(range(income, na.rm=TRUE))
52 [1] 22184
53 > quantile(income, 0:4/4, na.rm=TRUE)
54      0%      25%      50%      75%     100%
25782.0 30004.5 32452.0 35496.0 47966.0
55 > fivenum(income)
56 [1] 25782.0 30004.5 32452.0 35496.0 47966.0
57 > mad(income, na.rm=TRUE)
58 [1] 3989.677
59 > scale(income)
60      [,1]
61      [1,] -0.82418572
62      [2,] -0.39987101
63      [3,] -0.31089458
64      [...]
```

3.4 Boxplots

Een snelle aanzet om verdelingen te bestuderen of veranderlijken te vergelijken is het bekijken van boxplots. Met de naam van één veranderlijke als argument



Figuur 3.4: Empirische cumulatieve distributiefunctie van **Mortality**

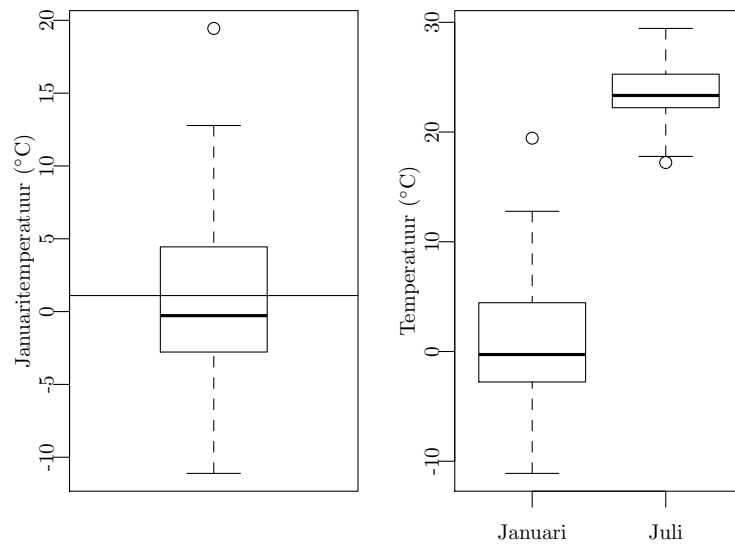
geeft `boxplot()` uiteraard de boxplot van die ene veranderlijke. Interessanter is om meerdere veranderlijken mee te geven, waarna R de verschillende boxplots ter vergelijking naast elkaar zet. Om de invloed van een factor als **regio** na te gaan, kan het interessant zijn om de boxplots van bijvoorbeeld **JanTemp** binnen elke groep van **regio** (C, N, NO, ...) naast elkaar te zetten. Notaties als **VAR1** ~ **VAR2** + **VAR3** zijn van de klasse **formula** en komen bij regressie nog uitgebreid terug aan bod.

```
66 > boxplot(JanTemp)
    > abline(h=mean(JanTemp))
68 > boxplot(JanTemp, JulyTemp)
    > boxplot(JanTemp ~ regio)                                     # zie Figuur 3.5
```

3.5 Het verband tussen veranderlijken

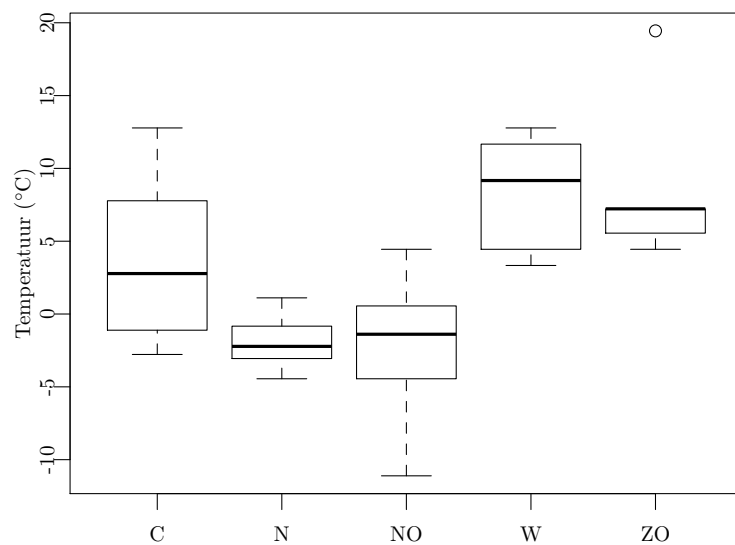
Afhankelijkheid tussen continue veranderlijken wordt gemeten met correlatie of covariantie en het best gevisualiseerd met een puntenplot. Om de correlatiematrix op te stellen, moeten de veranderlijken eerst worden gebundeld in een **data.frame()**. Correlatie tussen ordinale veranderlijken zal niet kunnen worden getest met de gewone Pearsoncorrelatie, maar zal gebruik maken van de rangveranderlijken – de Spearmancorrelatie. Omdat het bij het vergelijken van verschillende veranderlijken niet voldoende is om enkel de ontbrekende waarden te negeren, omdat daardoor ook beschikbare gegevens ongebruikt blijven, moet bij deze commando's de optie **use** worden ingesteld in plaats van **na.rm**.

Een spreidingsdiagram (*scatterplot*) is de aangewezen manier om het verband tussen twee continue veranderlijken grafisch voor te stellen. Het commando `plot()` met twee argumenten van het type **numeric** genereert dat soort figuren.



(a) Boxplot en gemiddelde

(b) Vergelijkende boxplots



(c) Gecategoriseerde boxplots van `JanTemp` ten opzichte van `regio`

Figuur 3.5: Boxplots en vergelijkende boxplots

Nominale veranderlijken worden vergeleken in een kruistabel. Statistieken van continue veranderlijken berekenen per groep van een nominale veranderlijke kan met `tapply()`.

```

70 > cov(JanTemp, JulyTemp)
[1] 5.176728
72 > cor(JanTemp, JulyTemp)
[1] 0.3462819
74 > cor(data.frame(JanTemp, JulyTemp, RelHum))
      JanTemp JulyTemp RelHum
76 JanTemp 1.00000000 0.3462819 0.06787158
   JulyTemp 0.34628186 1.0000000 -0.45280915
78 RelHum 0.06787158 -0.4528092 1.00000000
> cor(JanTemp, JulyTemp, method="spearman")
80 [1] 0.4371762
> cor(rank(JanTemp), rank(JulyTemp))
82 [1] 0.4371762
> plot(JanTemp, JulyTemp) # zie Figuur 3.6
84 > cor(pop, income, use="complete.obs")
[1] 0.3184839
86 > table(arbeid, inkomen)
      inkomen
arbeid      1  2  3  4
88 laag      1  0  2  1
90 gemiddeld  9 24 10  2
   hoog      5  4  0  1
92 > tapply(income, regio, mean, na.rm=TRUE)
      C      N      NO      W      ZO
94 36893.22 31626.80 32430.62 38209.83 29503.60
> tapply(income, regio, sd, na.rm=TRUE)
96      C      N      NO      W      ZO
5389.124 2439.165 3251.236 5863.121 3041.000

```

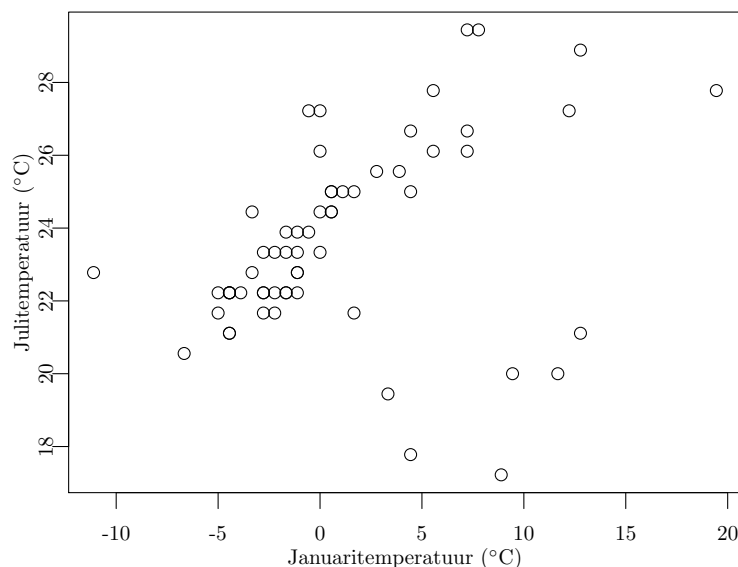
3.6 De klasse-afhankelijke methoden

Op basis van de klasse van een veranderlijke kan R bepalen welke grafieken en statistieken het meest relevant zijn. In de praktijk zal het dus vaak volstaan `summary()` en `plot()` te gebruiken. Hieronder staat het resultaat van het `summary()`-commando toegepast op de datatypes `integer`, `factor` en `data.frame`. Naargelang de invoer maakt het `plot()`-commando een tijdreeksgrafiek, staafdiagram, vergelijkende boxplots of een spreidingsdiagram.

```

98 > summary(income)
      Min. 1st Qu.  Median    Mean 3rd Qu.    Max.    NAs
100 25780   30000   32450   33250   35500   47970      1
> summary(regio)
102 C  N NO  W  ZO
   10 15 24  6  5
104 > summary(pollutie)
      JanTemp      JulyTemp      ...      inkomen
106 Min.      : -11.1111   Min.      :17.22      1      :15
   1st Qu.:  -2.7778   1st Qu.:22.22      2      :28
108 Median :  -0.2778   Median :23.33      3      :12

```



Figuur 3.6: Puntenplot van JulyTemp ten opzichte van JanTemp

	Mean	:	1.1019	Mean	:	23.66	4	:	4
110	3rd Qu.	:	4.4444	3rd Qu.	:	25.14	NA's	:	1
	Max.	:	19.4444	Max.	:	29.44			

3.7 Meer over grafieken

Omdat de standaardoutput vaak nog aanpassingen vergt om een publiceerbare grafiek te bekomen, volgen hier nog meer mogelijkheden van het `plot`-commando. Een algemene opmerking over de manier waarop grafieken in een rekenpakket worden gemaakt is misschien nuttig:

Een symbolisch rekenpakket beschikt typisch over een zogenaamd *smart-plot*-commando, de gebruiker dient enkel een vergelijking en grenzen voor de veranderlijken in te geven en het pakket bepaalt zelf welke punten worden uitgerekend. Voor het tekenen van de grafiek van een functie bijvoorbeeld, zal het pakket dan de functiewaarde in een voldoende groot aantal punten berekenen en de bekomen punten met lijnstukken verbinden. In numerieke pakketten zoals **R** zullen de meeste situaties echter vereisen dat de gebruiker zelf de punten bepaalt die moeten worden getekend, dat deze de corresponderende functiewaarden zelf berekent en zelf een lijngrafiek van de corresponderende punten maakt.

Een heel aantal mogelijkheden worden hieronder geïllustreerd bij het maken van een functie-onderzoek van de standaardnormale dichtheidsfunctie.

1. Functies en puntenrijen definiëren.

```

112 > f = function(x) { exp(-x^2/2)/sqrt(2*pi) }
> g = function(x) { -x*exp(-x^2/2)/sqrt(2*pi) }
114 > h = function(x) { (x^2-1)*exp(-x^2/2)/sqrt(2*pi) }
```

2. Een leeg grafiekvenster starten met passende assen en titels.

```

> plot(c(-4,4), c(-.4,.4), type='n',
116 + xlab='$x$', ylab='$y$',
+ main='De standaardnormale dichtheid')
118 > x = seq(-1, 1, length=100)
> polygon( c(x,rev(x)), c(f(x),rep(0,length=100)),
120 + col="gray", border=NA)

```

3. Punten en lijnen in de gewenste stijl toevoegen aan het grafiekvenster.

```

> x = seq(-4, 4, length=100)
122 > lines(x, f(x), lwd=3, col='blue')
> lines(x, g(x), lwd=2, col='green', lty='dashed')
124 > lines(x, h(x), lwd=2, col='red', lty='dashed')
> abline(h=0); abline(v=0)
126 > points(c(-1,0,1), c(0,0,0),
+ col=c('red','green','red'), pch='o')
128 > points(c(-1,0,1), f(c(-1,0,1)),
+ col=c('red','green','red'), pch='o')
130 > lines(c(-1,-1), c(0,f(-1)), col='red', lty='dashed')
> lines(c(0,0), c(0,f(0)), col='green', lty='dashed')
132 > lines(c(1,1), c(0,f(1)), col='red', lty='dashed')

```

4. Tekst en legende toevoegen.

```

> text(1.75, f(1), "buigpunt", col='red')
134 > text(-1.75, f(1), "buigpunt", col='red')
> text(1.75, f(0),
136 + paste("maximum $f(0) \\\approx ",
+ round(f(0),2),"$", sep=''), col='green')
138 > text(0.5, f(0.5)/2,
+ paste(round(100*(1-2*pnorm(-1))), '$\\%$'),
140 + col='white')
> legend(-4, y=-.2, col=c('blue','green','red'),
142 + lwd=c(3,2,2),
+ lty=c('solid','dashed','dashed'), # zie Figuur 3.7
144 + legend=c("$y=f(x)$", "$y=f\\'(x)$", "$y=f\\''(x)$"))

```

De bekomen grafiek kan dan worden opgeslaan zoals eerder beschreven.

L^AT_EX-gebruikers kunnen de grafiek ook als TikZ-code exporteren, welke dan rechtstreeks in een T_EX-bron kan worden ingevoegd. Dit garandeert een leesbare grafiek met lettertypes conform het document. Hiervoor moet het pakket **tikzDevice** worden geladen in R en met het commando **tikz()** wordt de bestandsnaam gespecificeerd. Vervolgens de plot-commando's uitvoeren waarvan de output dan niet te zien is op het scherm. Het is mogelijk om L^AT_EX-code te gebruiken in alle tekstlabels. Om technische redenen moet elke backslash in L^AT_EX-annotaties dubbel worden gezet. Met **dev.off()** wordt de grafiek ten slotte weggeschreven naar het bestand.

```

> library(tikzDevice)
146 > tikz(file = "grafiek.tex")
> plot(...)
148 [...]
> legend(...)
150 > dev.off()
null device
152 1

```

In het `.tex`-document moet het pakket `tikz` worden geladen en kan de grafiek met `\input{grafiek}` worden ingevoegd. De code uit de vijf bovenliggende blokken R-code resulteert zo in figuur 3.7 door onderstaande L^AT_EX-commando's.

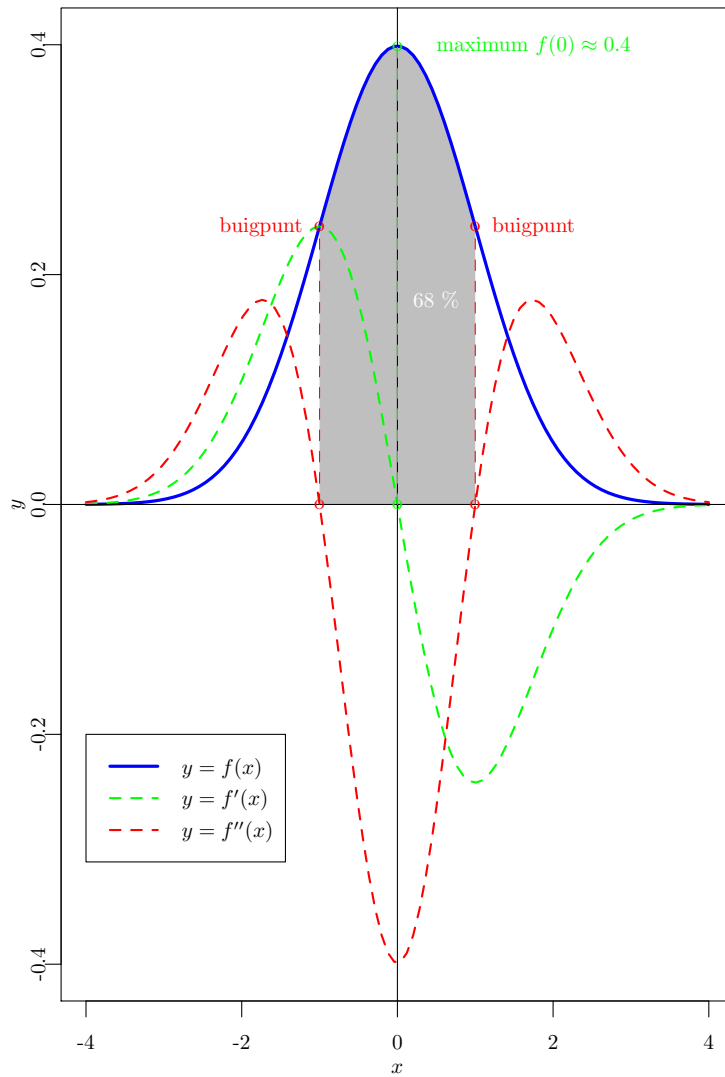
```
\documentclass{article}
\usepackage{tikz}

\begin{document}

\input{grafiek}

\end{document}
```

De standaardnormale dichtheid



Figuur 3.7: Een tikz-figuur.

Tabel 3.1: Beschrijvende statistieken en grafieken in R

```

## steekproefstatistieken voor 1 veranderlijke
2 summary(X) # samenvatting ngl. klasse X:
# frequentietabel, kwantielen en gemiddelde, ...
4 table(CAT) # frequentietabel
mean(NUM) # gemiddelde
6 mean(NUM, trim=0.05) # getrimd gemiddelde
median(NUM) # mediaan
8 sd(NUM) # standaarddeviatie
var(NUM) # variantie
10 mad(income) # mediaan abs deviatie
min(NUM) # minimum
12 max(NUM) # maximum
quantile(NUM, .25) # kwantiel (1e kwartiel)
14 fivenum(income) # kwartielen
## vergelijkende steekproefstatistieken
16 cor(gegevens) # correlatiematrix
cor(ORD1, ORD2, # correlatie
18 method = "spearman") # "pearson", "kendall"
cov(NUM1, NUM2) # covariantie
20 table(CAT1, CAT2) # kruistabel
tapply(NUM, CAT, mean) # statistieken per groep
22 ## nieuwe grafiek - high-level
plot(CAT) # staafdiagram
24 plot(NUM) # tijdreeksgrafiek
plot(NUM1, NUM2) # spreidingsdiagram
26 plot(CAT1, CAT2) # vergelijkende staafdiagrammen
plot(CAT, NUM) # vergelijkende boxplots
28 plot(ecdf(VAR)) # emp. verdelingsfunctie
boxplot(NUM) # boxplot
30 boxplot(NUM ~ CAT1 + CAT2) # vergelijkende boxplots
hist(NUM) # histogram
32 barplot(table(CAT)) # staafdiagram
qqnorm(CTU); qqline(CTU) # kwantielplot
34 ## opties binnen plot-commando's
main|sub|xlabel|ylabel = "..." # (as)titels toevoegen
36 xlim|ylim = c(a,b) # bereik van de assen
asp = 1 # assen schalen
38 type = "p|l|b|h|s|n" # grafiektype
col = "red|blue|..." # kleur van de grafiek
40 pch = 0:25 # plot-symbool
lwd = 2 # lijndikte
42 ## elementen toevoegen aan grafieken - low-level
points|lines(NUM3, NUM4) # punten|lijnplot toevoegen
44 abline(intercept, slope) # rechte toevoegen
abline(h=1|v=1) # hor.|vert. rechte toevoegen
46 title(main=..., sub=...) # titels toevoegen
text(X, Y, "tekst") # tekstvak toevoegen
48 legend(X, Y, c(...)) # legende toevoegen

```

Hoofdstuk 4

Hypothesetesten

Dit hoofdstuk illustreert aan de hand van een aantal voorbeelden hoe hypothesetests met R kunnen worden uitgevoerd. Een lijst van de belangrijkste commando's en attributen van testen is te vinden in Tabel 4.4.

Per soort test wordt de samenhang tussen de belangrijkste testmethodes schematisch voorgesteld in Tabel 4.3. Deze schema's zijn uiteraard slechts een geheugensteuntje, in de praktijk dienen altijd de veronderstellingen te worden nagegaan. In het algemeen hebben de tests in de gegeven volgorde telkens aflopende kracht.

Het uitvoeren van een test verloopt steeds volgens het schema in Tabel 4.1. De eerste 3 stappen kunnen grotendeels worden beantwoord vooraleer een steekproef is genomen en bevatten in principe geen steekproefgegevens. Deze stappen vormen het design van de test en laten bijvoorbeeld toe om voorafgaand aan metingen de kracht van een test na te gaan of de vereiste steekproefgrootte te schatten.

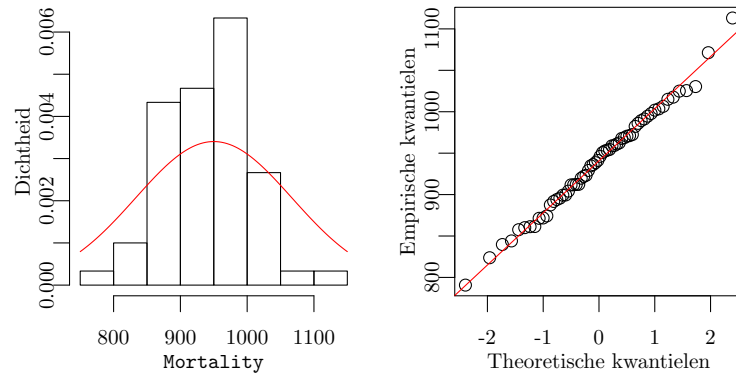
4.1 Test voor normaliteit

Veel hypothesetesten vereisen normaliteit van de veranderlijke X of van het steekproefgemiddelde \bar{X} , maar meestal is de verdeling van een veranderlijke niet gekend. Daarom starten de meeste analyses met het nagaan van de normaliteit.

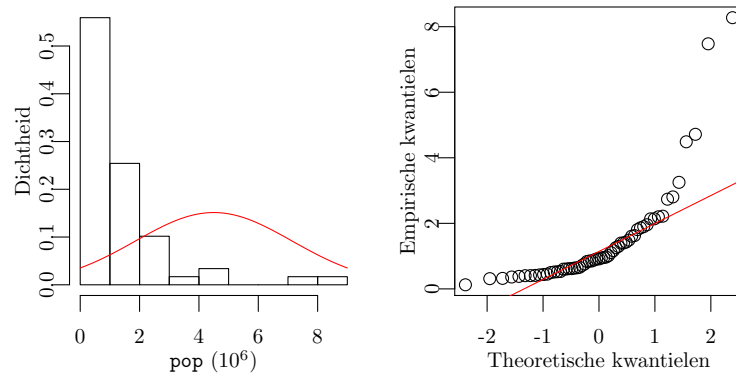
Een eerste mogelijkheid om de verdeling van een veranderlijke te achterhalen is naar de vorm van het histogram te kijken, aangevuld met een passende normale kromme. Hoewel deze voorstelling relevante informatie kan geven over symmetrie en de globale vorm van de verdeling, is het onmogelijk uitspraken te doen over de exacte verdeling, aangezien door het categoriseren van de data veel informatie verloren gaat.

```
> x = Mortality
2 > hist(x, freq=FALSE)
  > curve(dnorm(x, mean(x), sd(x)), col='red', add=TRUE)
4 > x = pop/10^6
  > hist(x, freq=FALSE, main="") # zie Figuur 4.1
6 > curve(dnorm(x, mean(x), sd(x)), col='red', add=TRUE)
```

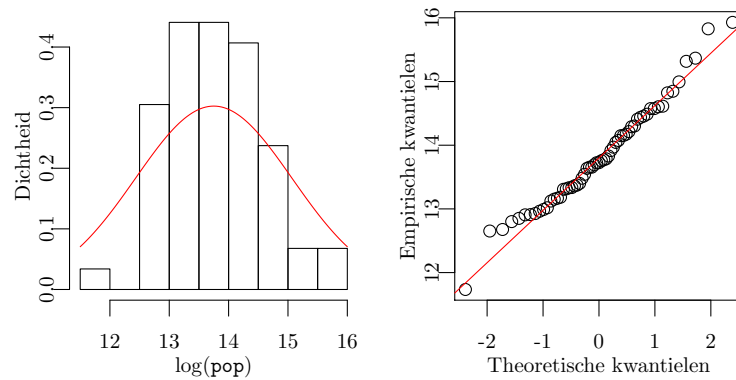
Een normale kwantielplot levert een veel duidelijkere manier om te bepalen of een veranderlijke X normaal verdeeld is op basis van een steekproef x_i ,



(a) Mortaliteit



(b) Populatiegrootte



(c) Populatiegrootte na logaritmische transformatie

Figuur 4.1: Dichtheidshistogrammen en kwantielplots

$i = 1, \dots, n$. Bovendien kunnen hierop objectieve statistische tests worden uitgevoerd.

Voor het maken van een normale kwantielplot wordt uit de gesorteerde steekproef $x_{(i)}$ de empirische kwantiel functie \hat{Q} behaald,

$$\hat{Q}\left(\frac{i}{n}\right) = x_{(i)}.$$

Binnen de steekproef is immers i/n van de data kleiner of gelijk aan $x_{(i)}$.

Bij de empirische kansen i/n horen dus de empirische kwantielen $x_{(i)}$, maar kunnen even goed theoretische kwantielen $\Phi^{-1}\left(\frac{i}{n}\right)$ worden berekend, de kwantielen die bij deze kansen horen onder een standaardnormale verdeling. Omdat $\Phi^{-1}(1)$ niet bestaat, dringt zich een zogenaamde continuïteitscorrectie op, die $\frac{i}{n}$ vervangt door $\frac{i-0.5}{n}$. De theoretische kwantielen worden

$$z_i = \Phi^{-1}\left(\frac{i-0.5}{n}\right)$$

en omdat voor een normale verdeelde veranderlijke $X \sim N(\mu, \sigma^2)$ geldt dat

$$Q(p) = \mu + \sigma \cdot \Phi^{-1}(p)$$

vormt de puntenverzameling

$$(z_i, x_i) = \left(\Phi^{-1}\left(\frac{i-0.5}{n}\right), x_i\right).$$

idealiter een perfecte rechte met intercept μ en slope σ .

In de praktijk liggen de punten in een normale kwantielplot natuurlijk nooit perfect op een rechte, en is ook deze methode dus niet objectief. De Shapiro-Wilk test (`shapiro.test()`) laat toe een test uit te voeren op basis van de correlatiecoëfficiënt van de normale kwantielplot.

```

> qqnorm(Mortality)
8 > qqline(Mortality, col='red')
> shapiro.test(Mortality)
10      Shapiro-Wilk normality test
data:  Mortality
12 W = 1, p-value = 0.9897
> qqnorm(pop/10^6)
14 > qqline(pop/10^6, col='red') # zie Figuur 4.1
> shapiro.test(pop/10^6)
16      Shapiro-Wilk normality test
data:  pop/10^6
18 W = 0.66, p-value = 2.327e-10

```

In het vervolg van deze tekst zullen er naast de `shapiro.test()` nog heel wat andere testen aan bod komen. R toont telkens een overzicht van de test-procedure, maar de uitvoer is eigenlijk een object van de klasse `htest`, met een aantal attributen zoals teststatistiek, p -waarde en – waar van toepassing – betrouwbaarheidsinterval `$conf.int`.

```

> TEST = shapiro.test(pop/10^6)
20 > attributes(TEST)

```

```

$names
22 [1] "statistic" "p.value"      "method"      "data.name"
$class
24 [1] "htest"
> TEST$statistic
26      W
0.663387
28 > TEST$p.value
[1] 2.326859e-10

```

Blijkt een veranderlijke X de normale verdeling niet te volgen, dan is het misschien mogelijk een transformatie f te vinden zodat $f(X)$ wel normaal is.

Een kandidaat functie f kan afgelezen worden van de normale kwantielplot. Is namelijk $f(X)$ normaal, dan zullen de getallen $f(x_i)$ een rechte normale kwantielplot $z_i = f(x_i)$ opleveren, dus toont de normale kwantielplot de inverse transformatie $x_i = f^{-1}(z_i)$.

In het bijzonder zal het vaak mogelijk zijn om met behulp van een logaritme een scheve verdeling te transformeren naar een (bij benadering) normale verdeling.

```

30 > x = log(pop)
> hist(x, freq=FALSE)
32 > curve(dnorm(x, mean(x), sd(x)), col='red', add=TRUE)
> qqnorm(x)
34 > qqline(x, col='red')
> shapiro.test(x)
36      Shapiro-Wilk normality test
data:  x
38 W = 0.98, p-value = 0.3888

```

In Tabel 4.3 staat een stappenplan dat toelaat te beslissen of X of \bar{X} normaal verdeeld zijn. Samenvattend spelen volgende drie factoren daarin mee.

Steekproefgrootte Is de steekproefgrootte echt klein, dan is er te weinig informatie om een normaliteitstest te doen en kan over de verdeling van geen van beide veranderlijken X of \bar{X} een geldige uitspraak gedaan worden.

C.L.S. Is de steekproef voldoende groot, minstens een twintigtal meetwaarden voor een symmetrische verdeling of 40 voor een eerder scheve verdeling, dan is de centrale limietstelling (C.L.S.) geldig en is het steekproefgemiddelde \bar{X} bij benadering normaal (zie [1] pagina 194).

S.W.T. Voor de steekproefwaarden zelf kan bij een steekproefgrootte vanaf 10 à 15 de Shapiro-Wilk test (S.W.T.) worden uitgevoerd. Deze vertelt of de veranderlijke X *zelf* normaal verdeeld is (zie [1] pagina 244).

4.2 Eén gemiddelde of gepaarde groepen

Om een test voor één gemiddelde EX of gelijkheid $E(X - Y) = 0$ van het gemiddelde van twee gepaarde groepen X en Y uit te voeren, zijn vier verschillende tests opgesteld in Tabel 4.3. Gerangschikt naar aflopende kracht:

z-test Indien het steekproefgemiddelde \bar{X} normaal verdeeld is en de populatievariantie σ^2 gekend, kan de normale test gebruikt worden. In principe is

dat nooit het geval. Daarom is deze in R dan ook niet geïmplementeerd (zie [1] pagina's 244 en 272).

t-test In de meeste gevallen is de Student's *t*-test van toepassing. Deze vereist enkel normaliteit van het steekproefgemiddelde (zie [1] pagina 232).

W.R.S. Indien de centrale limietstelling niet geldig is, kan Wilcoxon's rangsomtest (W.R.S.) worden gebruikt. Deze test is aangewezen als beide verdelingen ongeveer dezelfde vorm hebben (zie [1] pagina 261).

Mediaan In extremis kan de mediaantest of tekentest worden gebruikt. Deze is geldig ongeacht verdeling van de veranderlijke of het gemiddelde, maar is in het typische geval van kleine steekproefgrootte zeer zwak. Bovendien is het een test voor de mediaan en niet voor het gemiddelde (zie [1] pagina 247).

Als voorbeeld zijn hier een aantal testen in detail uitgeschreven. Waar elders enkel de R-code staat afgedrukt, blijft de interpretatie van de test aan de lezer.

Voorbeeld 1. Verschilt de gemiddelde januaritemperatuur in de U.S.A. significant van het vriespunt?

```

40 > mean(JanTemp)
[1] 1.101852
42 > sd(JanTemp)
[1] 5.649388
44 > length(JanTemp)
[1] 60
46 > t = mean(JanTemp)/sd(JanTemp)*sqrt(length(JanTemp)); t
[1] 1.510767
48 > qt(.975, 59)
[1] 2.000995
50 > 2*pt(-abs(t), length(JanTemp)-1)
[1] 0.1361856
52 > t.test(JanTemp, mu=0, alternative="two.sided")
      One Sample t-test
data:  JanTemp
t = 1.5108, df = 59, p-value = 0.1362
alternative hypothesis: true mean is not equal to 0
95 percent confidence interval:
-0.3575398  2.5612435
sample estimates:
mean of x
60 1.101852

```

1. TEST VOOR ÉÉN GEMIDDELDE:

$$\begin{aligned}
 &X = \text{JanTemp}, \\
 &\begin{cases} H_0 : \mu_X = 0 \\ H_1 : \mu_X \neq 0 \end{cases} \quad \text{met significantie } \alpha = 5\%.
 \end{aligned}$$

2. Student's *t*-test voor één gemiddelde:

$$T = \frac{\bar{X}_n - 0}{S/\sqrt{n}} \stackrel{\text{C.L.S.}}{\sim} t_{n-1},$$

C.L.S. is geldig wegens $n = 60$

3. Tweezijdige test:
 H_0 verwerpen als $|t| > t_{n-1, \alpha/2}$.
4. Geobserveerde statistieken:
 $\bar{x}_{60} = 1.10$, $t = \frac{1.10 - 0}{5.65/\sqrt{60}} = 1.51 < 2.00 = t_{59, 2.5\%}$
 $p = P(|T| > 1.51) = 13.62\% > 5\%$.
5. Besluit:
De test bevestigt wat op basis van figuur 3.5 (a) al kon worden vermoed:
op basis van deze steekproef kan niet worden beslist dat de gemiddelde
januaritemperatuur in de U.S.A. verschilt van het vriespunt. Het geob-
serveerde verschil van 1.10°C is niet significant.

Voorbeeld 2. Is de gemiddelde julitemperatuur in de U.S.A. significant hoger dan de gemiddelde januaritemperatuur?

```

> mean(JanTemp)
62 [1] 1.101852
> mean(JulyTemp)
64 [1] 23.65741
> mean(JanTemp - JulyTemp)
66 [1] -22.55556
> sd(JanTemp - JulyTemp)
68 [1] 5.344582
> length(JanTemp - JulyTemp)
70 [1] 60
> qt(.05, 59)
72 [1] -1.671093
> t.test(JanTemp, y=JulyTemp, alternative="less",
74 +      paired=TRUE)
      Paired t-test
data:  JanTemp and JulyTemp
t = -32.69, df = 59, p-value < 2.2e-16
78 alternative hypothesis: true difference is less than 0
95 percent confidence interval:
80      -Inf -21.40253
sample estimates:
82 mean of the differences
      -22.55556
84 > t.test(JanTemp-JulyTemp, mu=0, alternative="less")
      One Sample t-test
data:  JanTemp - JulyTemp
t = -32.69, df = 59, p-value < 2.2e-16
88 alternative hypothesis: true mean is less than 0
95 percent confidence interval:
90      -Inf -21.40253
sample estimates:
92 mean of x
      -22.55556

```

1. TEST VOOR TWEE GEMIDDELDEN, GEPAARDE GROEPEN:
 $X = \text{JanTemp}$ en $Y = \text{JulyTemp}$,

$$\begin{cases} H_0 : \mu_X \geq \mu_Y \\ H_1 : \mu_X < \mu_Y \end{cases} \quad \text{of} \quad \begin{cases} H_0 : \mu_X - \mu_Y \geq 0 \\ H_1 : \mu_X - \mu_Y < 0 \end{cases} \quad \text{met significantie } \alpha = 5\%.$$

2. Student's t -test voor twee gepaarde gemiddelden,
equivalent met t -test voor één verschilveranderlijke:

$$T = \frac{\overline{X-Y}}{S/\sqrt{n}} \stackrel{\text{C.L.S.}}{\sim} t_{n-1},$$
 C.L.S. is geldig wegens $n = 60$.
3. Eenzijdige test:
 H_0 verwerpen als $t < -t_{n-1,\alpha}$.
4. Geobserveerde statistieken:

$$\overline{x-y} = -22.56, t = \frac{1.10-23.66}{5.34/\sqrt{60}} = -32.69 < -1.67 = -t_{59,5\%}.$$

$$p = P(T < -32.69) \approx 0\% \ll 5\%.$$
5. Besluit:
 Rekening houdend met de spreiding, ziet het verschil van 22.56° tussen januari- en julitemperatuur er op figuur 3.5 (b) zeer groot uit. Dit verschil blijkt inderdaad significant: op basis van deze steekproef kan worden beslist dat de gemiddelde julitemperatuur in de U.S.A. hoger is dan de gemiddelde januaritemperatuur.

4.3 Twee gemiddelden, ongepaarde groepen

Het is uiteraard niet mogelijk om de verschilveranderlijke $X - Y$ te gebruiken in het geval van ongepaarde groepen. Er kan wel met de veranderlijke $\overline{X} - \overline{Y}$ worden gewerkt, maar daarvan is de verdeling afhankelijk van de onderliggende varianties σ_X^2 en σ_Y^2 . Daarom zijn in dit geval andere tests van toepassing, te vinden in tabel 4.3 en hieronder opnieuw gerangschikt naar aflopende kracht.

Tests voor gemiddelden

z -test De normale test, in de praktijk nooit geldig wegens onbekende varianties (zie [1] pagina's 250 en 276).

$t_{\sigma_1^2 \neq \sigma_2^2}$ -test Welch test of Student's t -test voor verschillende varianties (zie [1] pagina 252).

$t_{\sigma_1^2 = \sigma_2^2}$ -test Student's t -test voor gelijke varianties (zie [1] pagina 254).

W.R.S. De Wilcoxon rangsomtest of Mann-Whitney test, een veralgemening van het geval voor één gemiddelde, is aangewezen indien de centrale li-mietstelling in een van beide groepen niet van toepassing is (zie [1] pagina 261).

Test voor varianties

F -test Om op basis van twee steekproeven te bepalen of X en Y dezelfde dan wel verschillende variantie hebben, is de Fisher test voor twee varianties aangewezen. Deze is echter enkel geldig indien beide populaties normaal zijn (zie [1] pagina 254).

Voorbeeld 3. Is er een significant verschil tussen de gemiddelde temperatuur in januari in de noordelijke en noordoostelijke regio van de U.S.A.?

```

94 > table(regio)
regio
96  C  N NO  W ZO
10 15 24  6  5
98 > tapply(JanTemp, regio, mean)
          C          N          NO          W          ZO
100 3.944444 -2.074074 -1.527778  8.425926  8.777778
> tapply(JanTemp, regio, sd)
          C          N          NO          W          ZO
102 5.887899 1.605583 3.612504 3.807346 6.078194
104 > shapiro.test(JanTemp[regio=="N"])
      Shapiro-Wilk normality test
106 data:  JanTemp[regio == "N"]
      W = 0.9622, p-value = 0.7314
108 > shapiro.test(JanTemp[regio=="NO"])
      Shapiro-Wilk normality test
110 data:  JanTemp[regio == "NO"]
      W = 0.9588, p-value = 0.4143
112 > var.test(JanTemp[regio=="N"], JanTemp[regio=="NO"])
      F test to compare two variances
114 data:  JanTemp[regio == "N"] and JanTemp[regio == "NO"]
      F = 0.1975, num df = 14, denom df = 23, p-value = 0.002938
116 alternative hypothesis: true ratio is not equal to 1
      95 percent confidence interval:
118  0.0791222 0.5532753
      sample estimates:
120 ratio of variances
          0.1975370
122 > t.test(JanTemp[regio=="N"], y=JanTemp[regio=="NO"],
+   paired=FALSE, var.equal=FALSE)
124      Welch Two Sample t-test
      data:  JanTemp[regio == "N"] and JanTemp[regio == "NO"]
126 t = -0.6458, df = 34.22, p-value = 0.5227
      alternative hypothesis: true difference is not equal to 0
128 95 percent confidence interval:
      -2.265049  1.172457
130 sample estimates:
      mean of x mean of y
132 -2.074074 -1.527778

```

1. TEST VOOR TWEE GEMIDDELDEN, ONGEPAARDE GROEPEN:

$X = \text{JanTemp}_N$ en $Y = \text{JanTemp}_{NO}$,

$$\begin{cases} H_0 : \mu_X = \mu_Y \\ H_1 : \mu_X \neq \mu_Y \end{cases} \quad \text{met significantie } \alpha = 5\%.$$

2. Student's t -test, twee ongepaarde groepen, verschillende varianties:

$$T = \frac{\bar{X} - \bar{Y}}{\sqrt{S_X^2/n_X + S_Y^2/n_Y}} \stackrel{\text{C.L.S.}}{\sim} t_r,$$

C.L.S. geldig omdat beide veranderlijken normaal verdeeld zijn, hierdoor kan de Fisher test worden gebruikt om varianties te vergelijken.

(a) TEST VOOR NORMALITEIT:

$$\begin{cases} H_0 : X \sim N \\ H_1 : X \not\sim N \end{cases} \quad \text{en} \quad \begin{cases} H_0 : Y \sim N \\ H_1 : Y \not\sim N \end{cases} \quad \text{met significantie } \alpha = 5\%.$$

- (b) Shapiro-Wilk test:
geldig voor $n_X = 15$ en $n_Y = 24$.
- (c) Verwerpingsregel:
 H_0 verwerpen als $p < \alpha$.
- (d) Geobserveerde statistieken:
voor X : $p = 73\% > 5\%$,
voor Y : $p = 41\% > 5\%$.
- (e) Besluit:
Beide veranderlijken zijn normaal verdeeld.

- (a) TEST VOOR TWEE VARIANTIES:

$$\begin{cases} H_0 : \frac{\sigma_X^2}{\sigma_Y^2} = 1 \\ H_1 : \frac{\sigma_X^2}{\sigma_Y^2} \neq 1 \end{cases} \quad \text{met significantie } \alpha = 5\%.$$

- (b) Fisher test voor twee varianties:
geldig wegens $X \sim N$ en $Y \sim N$.
- (c) Verwerpingsregel:
 H_0 verwerpen als $p < \alpha$.
- (d) Geobserveerde statistieken:
 $\frac{s_X^2}{s_Y^2} = 0.20$ en $p = 0.29\% \ll 5\%$.
- (e) Besluit:
De varianties zijn verschillend.

3. Tweezijdige test:
 H_0 verwerpen als $|t| > t_{r,\alpha/2}$.

4. Geobserveerde statistieken:
 $\bar{x} = -2.07$, $\bar{y} = -1.53$, $|t| = |-0.65| < 2.03 = t_{34.22, 2.5\%}$.
 $p = P(|T| > 0.65) = 52.27\% > 5\%$.

5. Besluit:
Het minieme geobserveerde verschil in figuur 3.5 (c) is niet significant.
Op basis van deze steekproef kan niet worden beslist dat de gemiddelde januaritemperatuur in het noorden en het noordoosten van de U.S.A. verschillend is.

Voorbeeld 4. Is er een significant verschil tussen het mediane inkomen in steden van de noordelijke en noordoostelijke regio van de U.S.A.?

```

134 > tapply(income, regio, mean, na.rm=TRUE)
      C      N      NO      W      ZO
36893.22 31626.80 32430.62 38209.83 29503.60
136 > tapply(income, regio, sd, na.rm=TRUE)
      C      N      NO      W      ZO
138 5389.124 2439.165 3251.236 5863.121 3041.000
    > shapiro.test(income[regio=="N"])
140      Shapiro-Wilk normality test
data:  income[regio == "N"]
142 W = 0.8812, p-value = 0.04948
    > shapiro.test(income[regio=="NO"])

```

```

144         Shapiro-Wilk normality test
data:  income[regio == "NO"]
146 W = 0.9337, p-value = 0.1181
> t.test(income[regio=="N"], y=income[regio=="NO"],
148 + paired = FALSE, var.equal=FALSE)
        Welch Two Sample t-test
150 data:  income[regio == "N"] and income[regio == "NO"]
t = -0.8786, df = 35.62, p-value = 0.3855
152 alt. hyp.: true difference in means is not equal to 0
95 percent confidence interval:
154 -2660.051  1052.401
sample estimates:
156 mean of x mean of y
      31626.80   32430.62

```

1. TEST VOOR TWEE GEMIDDELDEN, ONGEPAARDE GROEPEN:

$X = \text{income}_N$ en $Y = \text{income}_{NO}$,

$$\begin{cases} H_0 : \mu_X = \mu_Y \\ H_1 : \mu_X \neq \mu_Y \end{cases} \quad \text{met significantie } \alpha = 5\%.$$

2. Student's t -test, twee ongepaarde groepen, verschillende varianties:

$$T = \frac{\bar{X} - \bar{Y}}{\sqrt{S_X^2/n_X + S_Y^2/n_Y}} \stackrel{\text{C.L.S.}}{\sim} t_r,$$

S.W.T. verwerpt normaliteit in één groep dus is de Fisher test niet geldig, maar omdat de afwijking van normaliteit niet te groot is ($p \approx 5\%$), is de C.L.S. wel bij benadering geldig voor $n_1 = 15$ en $n_2 = 24$: gebruik een t -test voor 2 onafhankelijke groepen met verschillende variantieschattingen.

3. Tweezijdige test:

H_0 verwerpen als $|t| > t_{r,\alpha/2}$.

4. Geobserveerde statistieken:

$\bar{x} = 31626.80$, $\bar{y} = 32430.62$, $|t| = |-0.8786| < 2.03 = t_{35.62, 2.5\%}$.

$p = P(|T| > 0.8786) = 38.55\% > 5\%$.

5. Besluit:

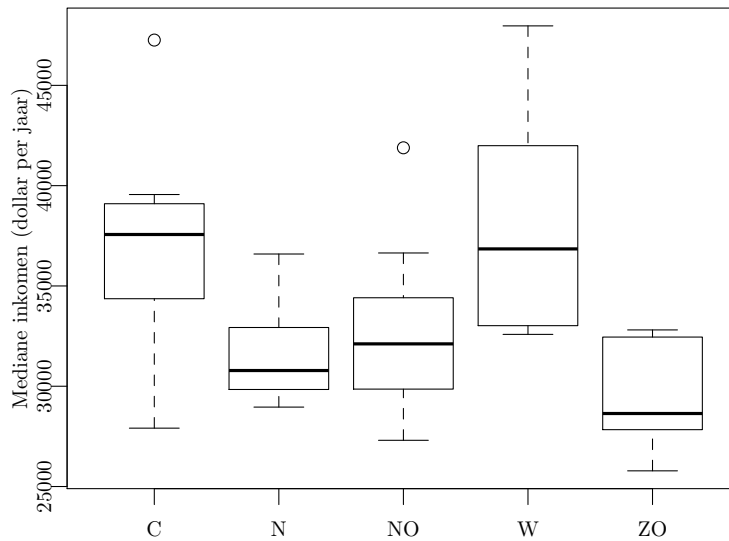
Zoals figuur 4.2 al suggereert is het geobserveerde verschil niet significant. Op basis van deze steekproef kan niet worden beslist dat het mediane loon in steden van de noordelijke en de noordoostelijke regio van de U.S.A. verschillend is.

Voorbeeld 5. Is er een significant verschil tussen de gemiddelde temperatuur in januari in de noordelijke en de centrale regio van de U.S.A.?

```

158 > wilcox.test(JanTemp[regio=="N"], y=JanTemp[regio=="C"],
+             alternative="two.sided", paired=FALSE)
160         Wilcoxon rank sum test with continuity correction
data:  JanTemp[regio == "N"] and JanTemp[regio == "C"]
162 W = 24.5, p-value = 0.005321
alternative hypothesis: true location shift not equal to 0
164 Warning message: ...
      cannot compute exact p-value with ties

```

Figuur 4.2: Vergelijkende boxplots van het mediane inkomen per regio

1. TEST VOOR TWEE GEMIDDELDEN, ONGEPAARDE GROEPEN:

$X = \text{JanTemp}_N$ en $Y = \text{JanTemp}_C$,

$$\begin{cases} H_0 : \mu_X = \mu_Y \\ H_1 : \mu_X \neq \mu_Y \end{cases} \text{ met significantie } \alpha = 5\%.$$

2. Wilcoxon rangsomtest:

W = rangsom van waarden in een groep,

steekproefgrootte $n_Y = 10$ ontoereikend voor zowel C.L.S. als S.W.T.

3. Tweezijdige test:

H_0 verwerpen als $p < \alpha$.

4. Geobserveerde statistieken:

$w = 24.5$, $p = 0.53\% < 5\%$.

5. Besluit:

Het verschil tussen januaritemperaturen in het noorden en het centrum van de U.S.A. ziet er op figuur 3.5 (c) eerder uitgesproken uit en is inderdaad significant. Op basis van deze steekproef kan worden beslist dat het tijdens de maand januari in het noorden van de U.S.A. gemiddeld kouder is dan in het centrum van het land.

4.4 Test voor afhankelijkheid

Voorgaande testprocedures vergeleken enkel populatieparameters. Vaak is het ook interessant om het verband tussen verschillende veranderlijken te bestuderen. Hiertoe bestaan verschillende tests die voornamelijk afhangen van het type veranderlijke, zoals te zien in tabel 4.3.

P.C.T. Samenhang tussen continue veranderlijken kan bestudeerd worden middels de correlatiecoëfficiënt. De Pearson correlatietest (P.C.T.) is echter enkel geldig voor normale populaties (zie [1] pagina 278).

S.C.T. Voor niet-normale populaties is er de Spearman correlatietest. Deze gebruikt de rangen in plaats van de meetwaarden en kan in bepaalde gevallen dus ook gebruikt worden voor ordinale veranderlijken, namelijk als er niet overmatig veel samenvallende waarden (*ties*) zijn.

χ^2 -Test Voor kwalitatieve veranderlijken kan een kruistabel worden opgesteld, waarin dan verwachte met geobserveerde aantallen kunnen worden vergeleken. Zijn de celfrequenties voldoende groot (Cochran-regel), dan is de Pearson χ^2 -test voor afhankelijkheid in een kruistabel van toepassing (zie [1] pagina 279).

F.E.T. In het geval van een 2×2 -tabel waarin de Cochran-regel niet geldig is, kan de Fisher exacte test worden gebruikt. De precieze werking valt buiten bestek van de cursus, maar de hypothesen zijn identiek als bij de χ^2 -test voor afhankelijkheid en de door R berekende *p*-waarde kan dus probleemloos worden geïnterpreteerd.

Voor kwalitatieve veranderlijken zijn er tests voor algemene afhankelijkheid, gebaseerd op kruistabellen. Onderscheid is nodig naargelang het aantal verwachte observaties in elke cel.

Voorbeeld 6. Is er significante samenhang tussen de hoeveelheid arbeiders en het inkomen?

```

166 > chisq.test(arbeid, inkomen)$observed
      inkomen
arbeid      1  2  3  4
  laag      1  0  2  1
170 gemiddeld  9 24 10  2
      hoog      5  4  0  1
172 Warning message: ... approximation may be incorrect
> chisq.test(arbeid, inkomen)$expected
      inkomen
arbeid      1      2      3      4
176 laag      1.016949  1.898305  0.8135593  0.2711864
      gemiddeld 11.440678 21.355932  9.1525424  3.0508475
178 hoog      2.542373  4.745763  2.0338983  0.6779661
Warning message: ... approximation may be incorrect
180 > arbeid2 = cut(100-X.WC, c(0,55,100),
+   labels=c("laag2","hoog2"), ordered_result=TRUE)
182 > inkomen2 = cut(income, c(0,30000,35000,Inf),
+   labels=c("1","2","3-4"), ordered_result=TRUE)
184 > chisq.test(arbeid2, inkomen2)$observed
      inkomen2
arbeid2      1  2 3-4
  laag2      5 18 12
188 hoog2     10 10  4
> chisq.test(arbeid2, inkomen2)$expected
      inkomen2
arbeid2      1      2      3-4

```

```

192 | laag2 8.898305 16.61017 9.491525
    | hoog2 6.101695 11.38983 6.508475
194 | > chisq.test(arbeid2, inkomen2)
    |      Pearson's Chi-squared test
196 | data:  arbeid2 and inkomen2
    | X-squared = 6.1141, df = 2, p-value = 0.04703
198 | > chisq.test(arbeid2, inkomen2)$residuals
    |      inkomen2
200 | arbeid2      1      2      3-4
    | laag2 -1.3068393  0.3410160  0.8142199
202 | hoog2  1.5781584 -0.4118160 -0.9832639

```

1. TEST VOOR AFHANKELIJKHEID:

$X = \text{arbeid}$,

$Y = \text{inkomen}$,

$\begin{cases} H_0 : X \text{ en } Y \text{ zijn onafhankelijk} \\ H_1 : X \text{ en } Y \text{ zijn afhankelijk} \end{cases}$ met significantie $\alpha = 5\%$.

2. χ^2 -test voor afhankelijkheid:

Heel wat verwachte aantallen zijn kleiner dan 5, dus wordt een nieuwe veranderlijke `arbeid2` gemaakt als volgt,

`hoog2` – meer dan 55% arbeiders;

`laag2` – minder dan 55% arbeiders.

Analoog voor `inkomen2`,

1 – mediaan inkomen < 30000 dollar;

2 – 30000 dollar < mediaan inkomen < 35000 dollar;

3-4 – 35000 dollar < mediaan inkomen.

Hiermee worden de vereiste aantallen wel gehaald en zal de χ^2 -benadering voldoende accuraat zijn.

3. Eenzijdige test:

H_0 verwerpen als $\chi^2 > \chi^2_{df,\alpha}$ of $p < \alpha$.

4. Geobserveerde statistieken:

$\chi^2 = 6.11 > 5.99 = \chi^2_{2,5\%}$ en $p = 4.70\% < 5\%$.

5. Besluit:

Er is een indicatie dat het mediaan inkomen en het aantal arbeiders in een stad afhankelijk zijn van elkaar. In de tabel met residuen is te zien dat de grootste afwijking zich voordoet in cel (`hoog2`, 1) met geobserveerde frequentie 10 en verwachte frequentie 6. In de dataset zijn dus meer steden met veel arbeiders en lage lonen dan verwacht in het geval er geen afhankelijkheid zou zijn. Dit effect wordt bevestigd in de andere cellen: de gegevens suggereren dus dat de lonen lager liggen als er meer arbeiders zijn en hoger als er meer bedienden zijn.

Ter illustratie worden hieronder de verschillende statistieken bij de χ^2 -test manueel nagerekend.

```

> obs = table(arbeid2, inkomen2); obs
      inkomen2
arbeid2  1   2  3-4
204      laag2   5 18  12
206      hoog2  10 10   4
208 > exp = rowSums(obs)%*%t(colSums(obs))/sum(obs); exp
      1   2   3-4
210 [1,] 8.898305 16.61017 9.491525
      [2,] 6.101695 11.38983 6.508475
212 > res = sign(obs-exp)*sqrt((obs-exp)^2/exp); res
      inkomen2
214 arbeid2      1      2      3-4
      laag2 -1.3068393 0.3410160 0.8142199
216      hoog2 1.5781584 -0.4118160 -0.9832639
> chisq = sum(res^2); chisq
218 [1] 6.114059
> df = prod(dim(res)-1); df
220 [1] 2
> 1-pchisq(chisq,df)
222 [1] 0.04702718

```

Voorbeeld 7. Is er ook in de noordoostelijke regio significante samenhang tussen de hoeveelheid arbeiders en het inkomen?

```

> x = arbeid[regio=="NO"]; y = inkomen[regio=="NO"]
224 > chisq.test(x,y)$expected
      y
226 x      1      2      3      4
      laag      0.50 1.083333 0.3333333 0.08333333
228      gemiddeld 4.25 9.208333 2.8333333 0.70833333
      hoog      1.25 2.708333 0.8333333 0.20833333
230 Warning message: ... approximation may be incorrect
> fisher.test(x,y)
232      Fisher's Exact Test for Count Data
data:  x and y
234 p-value = 0.09498
alternative hypothesis: two.sided

```

1. TEST VOOR AFHANKELIJKHEID:
 $X = \text{arbeid}_{\text{NO}},$
 $Y = \text{inkomen}_{\text{NO}},$
 $\begin{cases} H_0 : X \text{ en } Y \text{ zijn onafhankelijk} \\ H_0 : X \text{ en } Y \text{ zijn afhankelijk} \end{cases}$ met significantie $\alpha = 5\%$.
2. Fisher exacte test voor afhankelijkheid:
Zelfs na samenvoegen blijven heel wat celfrequenties kleiner dan 5, de χ^2 -benadering is niet betrouwbaar.
3. Tweezijdige test:
 H_0 verwerpen als $p < \alpha$.
4. Geobserveerde statistieken:
 $p = 9.50\% > 5\%$.

5. Besluit:

De Fisher exacte test vindt geen significant verband tussen het mediane inkomen en het aantal arbeiders in noordoostelijke steden. Dit kan betekenen dat er in die regio inderdaad geen verband is, maar mogelijk is het aantal observaties hier te laag om de nulhypothese te verwerpen.

Voor kwantitatieve veranderlijken zijn er correlatietests, welke enkel *lineaire* afhankelijkheid nagaan. Het zal soms nuttig zijn om kwantitatieve veranderlijken door hercodering om te zetten in kwalitatieve veranderlijken namelijk om andere dan lineaire verbanden te vergelijken, of om afhankelijkheid tussen een continue en een categorische veranderlijke na te gaan. Voor een ordinale veranderlijke met veel categorieën en relatief weinig knopen kan een Spearman-correlatietest aangewezen zijn, bijvoorbeeld als hercoderen te veel informatieverlies betekent. In dat geval worden de ordinale gegevens in essentie eerst met `rank()` omgezet in rangnummers vooraleer de steekproefcorrelatie te berekenen.

Voorbeeld 8. Is er significante samenhang tussen januari- en julitemperatuur?

```
236 > cor(JanTemp, JulyTemp)
      [1] 0.3462819
238 > cor(JanTemp, JulyTemp, method = c("spearman"))
      [1] 0.4371762
240 > shapiro.test(JanTemp)
      Shapiro-Wilk normality test
242 data:  JanTemp
      W = 0.9278, p-value = 0.001606
244 > shapiro.test(JulyTemp)
      Shapiro-Wilk normality test
246 data:  JulyTemp
      W = 0.9778, p-value = 0.3443
248 > cor.test(JanTemp, JulyTemp, method = c("spearman"))
      Spearman's rank correlation rho
250 data:  JanTemp and JulyTemp
      S = 20256.03, p-value = 0.0004783
252 alt. hyp.: true rho is not equal to 0
      sample estimates:
254      rho
      0.4371762
256 Warning message: [] Cannot compute exact p-values with ties
```

1. TEST VOOR AFHANKELIJKHEID:

$X = \text{JanTemp}$,
 $Y = \text{JulyTemp}$,
$$\begin{cases} H_0 : \rho(X, Y) = 0 \\ H_0 : \rho(X, Y) \neq 0 \end{cases} \quad \text{met significantie } \alpha = 5\%.$$

2. Spearman correlatietest:

De Shapiro-Wilk test ($p = 0.16\%$) toont dat `JanTemp` niet normaal verdeeld is, waardoor de Pearson correlatietest niet is aangewezen.

3. Eenzijdige test:

H_0 verwerpen als $p < \alpha$.

4. Geobserveerde statistieken:

$p = 0.05\% \ll 5\%$.

5. Besluit:

De steekproefcorrelatie $r = 0.35$ van januari- en julitemperatuur is significant verschillend van nul. De test bevestigt dus de trend in figuur 3.6: de gegevens tonen afhankelijkheid tussen beide veranderlijken, steden met een hogere januaritemperatuur hebben gemiddeld ook een hogere julitemperatuur.

4.5 Test voor proporties

In het geval van één of twee proporties is een normale benadering vaak te verantwoorden, maar in R zijn andere testen geïmplementeerd. Voor het testen van één proportie, is er immers de exacte, binomiale test. In het geval van twee proporties is het mogelijk een test voor afhankelijkheid uit te voeren op de kruistabel van successen en mislukkingen versus beide groepen.

Voorbeeld 9. Verschilt de proportie steden met een gemiddelde hoeveelheid arbeiders significant van 50%?

```

> table(arbeid)
arbeid
      laag gemiddeld      hoog
      4      46      10
> length(arbeid)
[1] 60
> binom.test(46, 60, p=0.50, alternative="two.sided")
Exact binomial test
data: 46 and 60
number of successes = 46, number of trials = 60, p-value =
4.224e-05
alt. hyp.: true prob. of success not equal to 0.5
95 percent confidence interval:
0.6396172 0.8661627
sample estimates:
probability of success
0.7666667
> 2*(1-pbinom(45, 60, .5))
[1] 4.223705e-05

```

1. TEST VOOR ÉÉN PROPORTIE:

X = Een stad heeft een laag aantal arbeiders,
 $\begin{cases} H_0 : \mu_X = 50\% \\ H_1 : \mu_X \neq 50\% \end{cases}$ met significantie $\alpha = 5\%$.

2. Exacte test voor één proportie:

$X_i \sim_{H_0} B(50\%)$ en $Y = \sum_{i=1}^{60} X_i \sim B(60, 50\%)$.

3. Tweezijdige test:

H_0 verwerpen als $2P(Y \geq y) < \alpha$.

4. Geobserveerde statistieken:

$n = 60$ en $y = 46$, $p = 2P(Y \geq 46) \approx 0\% \ll 5\%$.

5. Besluit:

De geobserveerde proportie, 77% steden met een gemiddeld aantal arbeiders, verschilt significant van 50%. De steekproef wijst op een grotere proportie steden met een gemiddeld aantal arbeiders.

Voorbeeld 10. Is er een significant verschil tussen de proportie steden met een gemiddeld aantal arbeiders in de noordoostelijke regio en de rest van de U.S.A.?

```
276 > table(arbeid=="gemiddeld", regio=="NO")
      FALSE TRUE
278 FALSE     7   7
      TRUE  29  17
280 > prop.test(c(17,29), c(24,36), alternative="two.sided")
      2-sample test for equality of proportions
282      with continuity correction
data:  c(17, 29) out of c(24, 36)
284 X-squared = 0.3144, df = 1, p-value = 0.575
      alternative hypothesis: two.sided
286 95 percent confidence interval:
      -0.3550639  0.1606194
288 sample estimates:
      prop 1      prop 2
290 0.7083333 0.8055556
```

1. TEST VOOR TWEE PROPORTIES:

X = Een stad uit het NO heeft een gemiddeld aantal arbeiders,

Y = Een stad uit een andere regio heeft een gemiddeld aantal arbeiders,

$$\begin{cases} H_0 : \mu_X = \mu_Y \\ H_1 : \mu_X \neq \mu_Y \end{cases} \text{ met significantie } \alpha = 5\%.$$

2. Test omtrent twee onbekende proporties.

3. Tweezijdige test:

H_0 verwerpen als $p < \alpha$.

4. Geobserveerde statistieken:

$$p = 57.5\% > 5\%.$$

5. Besluit:

De geobserveerde proporties steden met een gemiddeld aantal arbeiders in het noordoosten (71%) en in de rest van de U.S.A. (81%) zijn niet significant verschillend. Op basis van deze test kan niet worden beslist dat er een verschillende proportie arbeiders in het noorden of het noordoosten is.

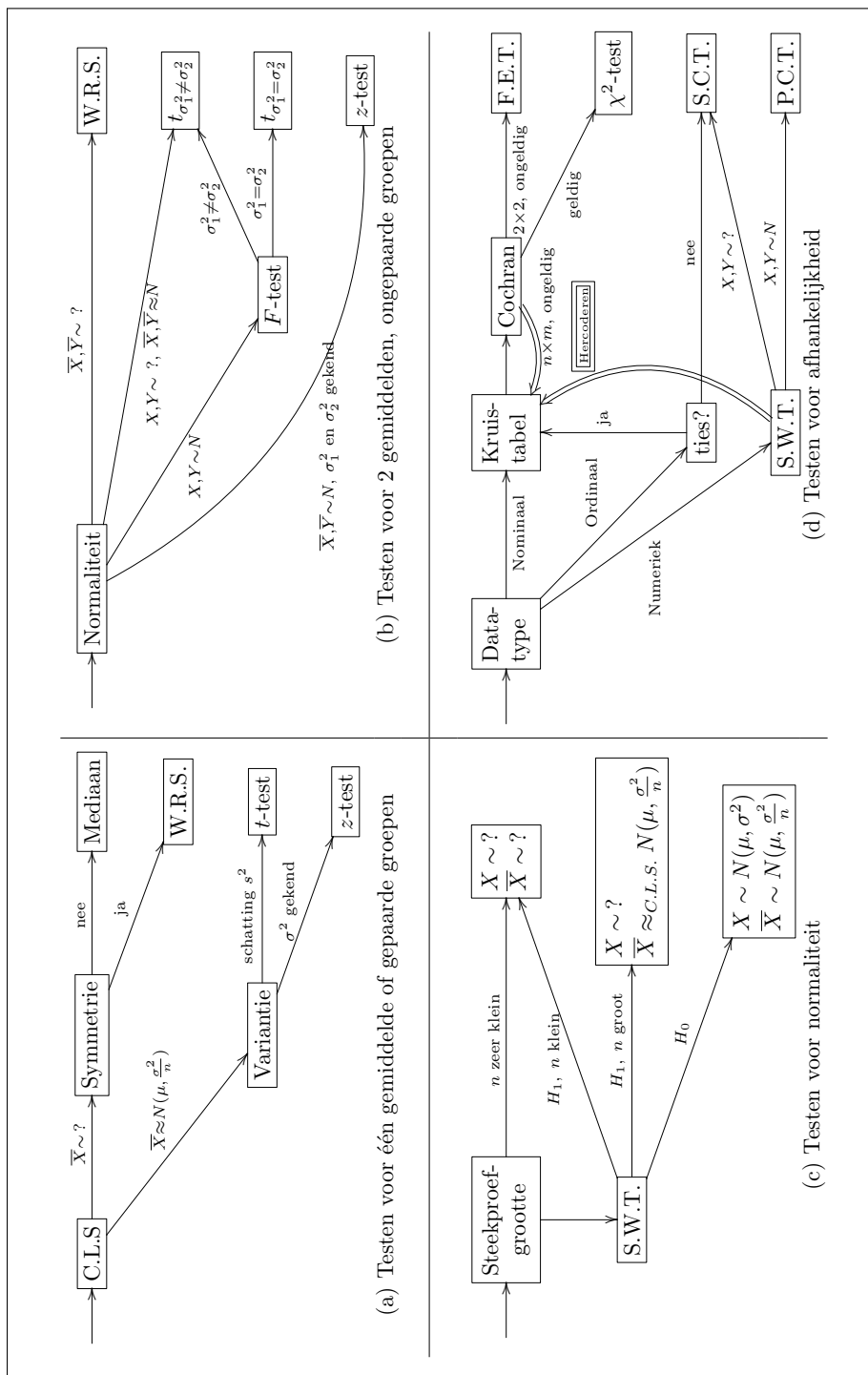
Tabel 4.1: Stappenplan voor het uitvoeren van hypothesetesten

1. Testprobleem:	Voorbeeld:
<ul style="list-style-type: none"> • Soort test • Nulhypothese en alternatieve hypothese • Significantieniveau 	<p>Test voor één gemiddelde</p> $\begin{cases} H_0 : \mu = \mu_0 \\ H_1 : \mu \neq \mu_0 \end{cases}$ <p>$\alpha = 5\%$</p>
2. Teststatistiek:	
<ul style="list-style-type: none"> • Veranderlijke • Verdeling • Voorwaarden 	$T_n = \frac{\bar{X}_n - \mu_0}{S_n / \sqrt{n}}$ $T_n \sim t_{n-1}$ <p>C.L.S.</p>
3. Testcriterium:	
<ul style="list-style-type: none"> • Aanvaardingsgebied • Definieer p-waarde • Schets van de verdeling en het aanvaardingsgebied 	$[-t_{n-1, \alpha/2}, t_{n-1, \alpha/2}]$ $p = P(T_n > t_{obs})$
4. Observaties:	
<ul style="list-style-type: none"> • Steekproefstatistiek • Bereken p-waarde • Vul figuur aan met statistiek en p-waarde 	$t_{obs} = \frac{\bar{x}_n - \mu_0}{s_n / \sqrt{n}}$ $p = P(T_n > t_{obs})$
5. Besluit:	
<ul style="list-style-type: none"> • Schrijf of H_0 wordt aanvaard of verworpen • Formuleer een conclusie: <i>Op basis van deze steekproef...</i> • Schrijf <i>niet</i> dat de nulhypothese waar is of dat de alternatieve wordt verworpen. 	

Tabel 4.2: Statistieken bij de belangrijkste hypothesetesten

z-test voor één gemiddelde	$Z = \frac{\bar{X} - \mu_0}{\sigma/\sqrt{n}} \sim N(0, 1)$	z-test
t-test voor één gemiddelde	$T = \frac{\bar{X} - \mu_0}{S_X/\sqrt{n}} \sim t_{n-1}$	t-test
Twee gemiddelden, gepaarde gegevens	$T = \frac{\bar{X} - \bar{Y}}{S_{X-Y}/n} \sim t_{n-1}$	
Twee gemiddelden, ongepaard, gelijke varianties		$t_{\sigma_1^2 = \sigma_2^2}$
	$T = \frac{(\bar{X} - \bar{Y})}{S_p \sqrt{1/n_1 + 1/n_2}} \sim t_{n_1+n_2-2} \text{ met } S_p^2 = \frac{(n_1-1)S_1^2 + (n_2-1)S_2^2}{n_1+n_2-2}$	
Twee gemiddelden, ongepaard, verschillende varianties		$t_{\sigma_1^2 \neq \sigma_2^2}$
	$T = \frac{\bar{X} - \bar{Y}}{\sqrt{S_1^2/n_1 + S_2^2/n_2}} \approx t_r \text{ met } r = \left(\frac{s_1^2}{n_1} + \frac{s_2^2}{n_2}\right)^2 / \left(\frac{(s_1^2/n_1)^2}{n_1-1} + \frac{(s_2^2/n_2)^2}{n_2-1}\right)$	
Twee varianties	$F = \frac{S_1^2}{S_2^2} \sim F_{n_1-1, n_2-1}$	F-test
Wilcoxon-rangsom of Mann-Whitney test		W.R.S.
	$\frac{W - n_1 \cdot (n_1 + n_2 + 1)/2}{\sqrt{n_1 \cdot n_2 \cdot (1 + n_1 + n_2)/12}} \approx N(0, 1)$	
z-test voor één propoortie	$Z = \frac{\hat{p} - p_0}{\sqrt{p_0(1-p_0)/n}} \sim N(0, 1)$	
z-test voor twee propoorties	$Z = \frac{(\hat{p}_1 - \hat{p}_2)}{\sqrt{p_1(1-p_1)/n_1 + p_2(1-p_2)/n_2}} \sim N(0, 1)$	
Pearson correlatietest	$T = \frac{R\sqrt{n-2}}{\sqrt{1-R^2}} \sim t_{n-2}$	P.C.T.
χ^2-test voor afhankelijkheid van kwalitatieve variabelen		χ^2 -test
	$\chi^2 = \sum_{i=1}^m \sum_{j=1}^n \frac{(O_{ij} - E_{ij})^2}{E_{ij}} \sim \chi_{(m-1)(n-1)}^2$	

Tabel 4.3: Overzicht van de belangrijkste hypothesetesten



Tabel 4.4: Hypothesetesten in R

```

## hypothesetest uitvoeren
2 shapiro.test(CTU) # Shapiro-Wilk-test
t.test(CTU, # t-test voor 1 gemiddelde
4 mu = mu0,
alternative="two.sided" # "less", "greater"
6 conf.level=0.95)
t.test(CTU1, # t-test voor 2 gemiddelden
8 y = CTU2,
paired = FALSE,
10 var.equal = FALSE)
binom.test(x, n, p = 0.5) # binomiale test 1 proportie
12 prop.test(x, n) # chi^2-test voor 2 proporties
wilcox.test(ORD, mu = 0, # Wilcoxon-tekenrangsomtest
14 y = NULL,
paired = FALSE)
16 var.test(CTU1, CTU2) # F-test voor 2 varianties
cor.test(VAR1, VAR2, # Correlatietest
18 method = "pearson") # "kendall", "spearman"
chisq.test(CAT1, CAT2) # chi^2-test afhankelijkheid
20 fisher.test(CAT1, CAT2) # Fisher exacte test
## resultaten van een testprocedure
22 TEST = <naam>.test(...) # testresultaten toekennen
class(TEST) # "htest"
24 attributes(TEST) # alle attributen bij test
TEST$statistic # teststatistiek
26 TEST$parameter # vrijheidsgraden
TEST$p.value # p-waarde
28 TEST$conf.int # betrouwbaarheidsinterval
TEST$estimate # schatter
30 TEST$null.value # H0-waarde
TEST$alternative # H1
32 TEST$method # testmethode
TEST$data.name # naam van de dataset
34 ?<naam>() # meer info onder 'value'

```

Hoofdstuk 5

Regressie-analyse

Waar een correlatietest enkel kan vertellen *of* er een verband is tussen verschillende veranderlijken, zal het met regressie mogelijk zijn om dat verband expliciet op te stellen. Dit vereist echter een aantal weldoordachte stappen zoals schematisch weergegeven in tabel 5.2. Het is immers niet moeilijk om de best passende rechte door een puntenwolk te trekken, wel om na te gaan of deze effectief het onderliggende verband tussen de veranderlijken beschrijft.

Eerst zal een eenvoudig regressiemodel worden opgesteld, omdat het resultaat, residu-analyse en outlier-onderzoek in geval van één enkele veranderlijke gemakkelijk grafisch voor te stellen zijn. Vanaf sectie 5.5 wordt het algemene geval van meerdere kandidaat regressoren bestudeerd.

5.1 Model opstellen

Op basis van theorie, voorkennis of verkennende grafieken, wordt een model voorgesteld, met $p - 1$ verklarende veranderlijken en p onbekende, te schatten coëfficiënten β_i ,

$$E(Y \mid X = x_i) = \beta_0 + \sum_{i=1}^{p-1} \beta_i \cdot x_i.$$

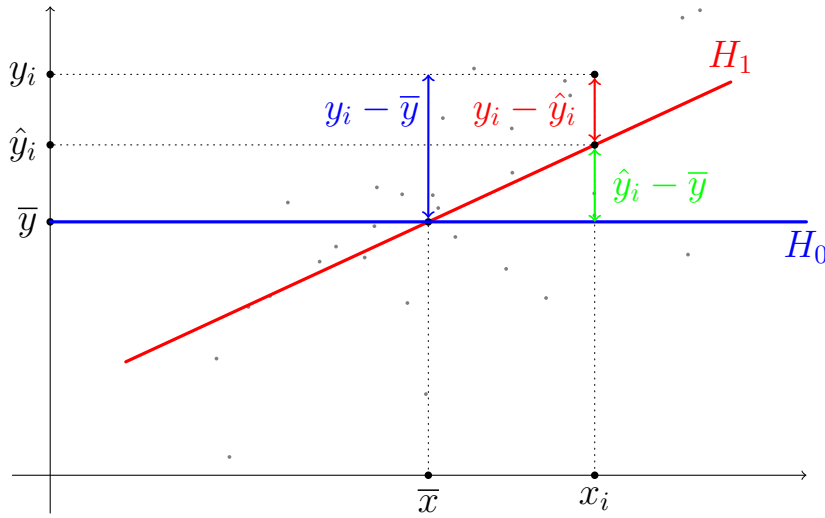
De kleinste kwadratenmethode levert schattingen $b_i = \hat{\beta}_i$ voor deze coëfficiënten. Algemeen zullen de meetpunten (x_i, y_i) zoals geïllustreerd in Figuur 5.1 niet gelijk zijn aan de voorspellingen (x_i, \hat{y}_i) van het model

$$\hat{y}_i = b_0 + \sum_{i=1}^{p-1} b_i \cdot x_i,$$

maar is er een afwijking r_i , het residu,

$$y_i = b_0 + \sum_{i=1}^{p-1} b_i \cdot x_i + r_i = \hat{y}_i + r_i.$$

Deze residuen zullen de sleutel zijn om de kwaliteit van een model te bepalen: ze laten toe problemen met modelveronderstellingen te ontdekken, hypothesetesten uit te voeren en om modellen te vergelijken.



Figuur 5.1: Voorspellingen en residuen bij een eenvoudig regressiemodel.

Als een regressiemodel de variatie in de data ten opzichte van het evenwichtspunt \bar{y} goed verklaart, zullen de voorspellingen \hat{y}_i dicht aansluiten bij de meetwaarden y_i en zullen de residuen klein zijn. Om kwantitatieve uitspraken te doen over alle residuen, wordt gekeken naar de kwadratensommen:

- $SST = \sum_{i=1}^n (y_i - \bar{y})^2 = \text{Sum of Squares Total}$
- $SSM = \sum_{i=1}^n (\hat{y}_i - \bar{y})^2 = \text{Sum of Squares for Model}$
- $SSE = \sum_{i=1}^n (y_i - \hat{y}_i)^2 = \text{Sum of Squares for Error}$

Een eerste maat die op basis van deze kwadratensommen wordt gedefinieerd is de determinatiecoëfficiënt $R^2 = \frac{\sum_{i=1}^n (\hat{y}_i - \bar{y})^2}{\sum_{i=1}^n (y_i - \bar{y})^2} = \frac{SSM}{SST} = 1 - \frac{SSE}{SST}$, maat voor het percentage van de variabiliteit in de dataset dat door het model wordt verklaard. Voor een perfect model, waarin alle residuen gelijk zijn aan nul, geldt $SSM = SST$ en is $R^2 = 1$, voor een model waar alle coëfficiënten nul zijn en waarin de regressoren dus niets verklaren geldt $R^2 = 0$.

In de praktijk wordt doorgaans de aangepaste determinatiecoëfficiënt gebruikt,

$$R_{\text{adj}}^2 = 1 - \frac{SSE/(n-p)}{SST/(n-1)},$$

die dezelfde interpretatie heeft maar wordt gepenaliseerd voor extra termen in het regressiemodel: wordt een term toegevoegd die weinig of geen bijdrage aan het model levert, dan zal R_{adj}^2 dalen.

Als de modelveronderstellingen geldig zijn (volgende sectie), kan worden nagegaan of het gevonden model wel degelijk enige variatie in de data verklaart,

$$H_0 : \beta_1 = \beta_2 = \dots = \beta_{p-1} = 0.$$

Indien het model absoluut niets verklaart, zal SSM klein zijn in vergelijking met SSE , in het andere geval wordt SSM relatief groot. De test bij deze hypothese

op basis van volgende teststatistiek heet de globale F -test,

$$F = \frac{SSM/(p-1)}{SSE/(n-p)} \sim F_{p-1, n-p}.$$

Van elke coëfficiëntschatte $\hat{\beta}_i$ kan de standaardfout $s(\hat{\beta}_i)$ worden berekend en, als de modelveronderstellingen geldig zijn, de verdeling opgesteld,

$$\frac{\hat{\beta}_i - \beta_i}{s_{\beta_i}} \sim t_{n-p-1}.$$

Deze statistieken laten toe te testen of een individuele coëfficiënt β_i al dan niet significant van nul verschilt.

Een regressiemodel wordt gemaakt met het commando `lm()` dat een argument van de klasse `formula` heeft. Het resultaat is een object van klasse `lm` dat heel wat attributen bevat. Verder geven de methoden `summary()` en `plot()` een overzicht van de belangrijkste aspecten van het regressiemodel.

```

> model1 = lm(income~Education)
2 > plot(income~Education)
> abline(model1, col='red') # zie Figuur 5.2
4 > summary(model1)
Call:
lm(formula = income ~ Education)
Residuals:
8      Min       1Q   Median       3Q      Max
-5977.7 -3070.4 -210.7  1918.5 14988.3
10 Coefficients:
              Estimate Std. Error t value Pr(>|t|)
12 (Intercept)   3984.1     6600.5   0.604   0.548
   Education    2668.5       600.1   4.446 4.09e-05 ***
14 ---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1
16 Residual standard error: 3888 on 57 degrees of freedom
(1 observation deleted due to missingness)
18 Multiple R-squared:  0.2575,    Adjusted R-squared:  0.2445
F-statistic: 19.77 on 1 and 57 DF,  p-value: 4.090e-05
20 > class(model1)
[1] "lm"
22 > attributes(model1)
$names
24 [1] "coefficients" "residuals" "effects" "rank"
   [5] "fitted.values" "assign" "qr" "df.residual"
26 [9] "na.action" "xlevels" "call" "terms"
[13] "model"

```

Het voorgestelde eenvoudige lineaire model $\text{income} = \beta_0 + \beta_1 \cdot \text{Education}$ levert als kleinste kwadratschatting

$$\text{income} = 3984 + 2669 \cdot \text{Education}. \quad (5.1)$$

Er wordt slechts een klein deel van de variabiliteit van `income` verklaard door `Education` ($R^2 = 0.26$) maar het waargenomen effect is zeker geen toevalseffect, aangezien de F -test extreem significant is ($p \approx 0\%$).

Deze vergelijking stelt dat het jaarinkomen in een stad met gemiddeld 2669 dollar stijgt per jaar opleiding dat de inwoners van die stad gemiddeld genieten en in principe is de betekenis van de intercept $b_0 = 3984$ dat het gemiddelde inkomen in een stad waar geen scholing is per jaar 3984 dollar bedraagt.

De intercept is echter niet te interpreteren omdat deze ver buiten het bereik $[9.0, 12.3]$ van **Education** ligt en dus een enorm grote fout kent. Het heeft geen zin zich af te vragen wat het loon is in een stad waar geen scholing is, omdat in de steekproef scholing in elke stad minstens negen jaar is. Enkel indien de intercept een fysische betekenis heeft *en* indien het bereik van de regressor rond of in de buurt van de nul ligt, kan deze waarde worden geïnterpreteerd en heeft de corresponderende *t*-test (zie verder) zin. Maar zelfs in dat geval is het expliciet weglaten van de intercept doorgaans enkel aangewezen als het theoretisch model dit voorschrijft. De intercept onderdrukken kan met een commando van de vorm `lm(Y~X-1)`.

5.2 Modelveronderstellingen

Zelfs al is een model significant, dan nog is het niet noodzakelijk juist of betrouwbaar. Of dit het geval is, kan nagegaan worden met de residuen, die zich zouden moeten gedragen als een zuiver lukrake steekproef uit een normaalverdeling met gemiddelde nul en een constante variantie ten opzichte van de regressoren, de zogenaamde Gauss-Markov voorwaarden:

$$\varepsilon_i \sim N(0, \sigma^2).$$

Deze modelveronderstellingen kunnen worden gecontroleerd op de normale kwantielplot en residuplots (\hat{y}_i, e_i) . Zijn de residuen niet normaal verdeeld, of is er sprake van heteroscedasticiteit, dan is het niet mogelijk om betrouwbare intervalschattingen rond de voorspellingen te maken. Een transformatie (sectie 5.6) kan eventueel wel soelaas bieden wanneer de residuen geen parallelle band rond nul vormen.

De specifieke deelset waarop de analyse is gebaseerd, is op te vragen met `$model`. Deze bestaat enkel uit de veranderlijken die bij de regressie betrokken zijn en die cases waarvan geen waarden ontbreken voor deze veranderlijken. Er zijn attributen voor de residuen (`$residuals`) en de geschatte respons (`$fitted.values`).

```

28 > model1$model
      income Education
30 ...
19  33858      10.8
32 20  32000      10.8
22 22  29915      10.9
34 23  29450      10.4
   ...
36 > residuen = model1$residuals
> y_hat = model1$fitted.values
38 > x_i = model1$model[,2]
> y_i = model1$model[,1]
40 > qqnorm(residuen)
> qqline(residuen, col='red')
```

```

42 > plot(x_i, residuen)
    > abline(h=0, col='red') # zie Figuur 5.2

```

De residuplot toont een mooie parallelle band rond nul, dus de residuen hebben een constant gemiddelde nul en een constante variantie. De residuen zijn echter niet normaal verdeeld, dus intervalschattingen met dit model zijn niet noodzakelijk betrouwbaar. Er zijn op het zicht mogelijks twee belangrijke uitschieters, die nader kunnen worden onderzocht.

5.3 Opsporen van uitschieters

Tot slot is het mogelijk dat individuele waarden, uitschieters, de parameters van het model sterk beïnvloeden. Worden dergelijke waarden opgemerkt, dan moeten deze zeker worden gerapporteerd als invloedrijk, maar *absoluut niet* zonder meer uit het regressiemodel verwijderd. Dit mag enkel indien

- een waarde zo onwaarschijnlijk is dat het zeker een meetfout moet betreffen, bijvoorbeeld een julitemperatuur van 75°C;
- een case om duidelijke redenen niet binnen de onderzoekspopulatie past, bijvoorbeeld een klein dorpje in een steekproef met voor de rest enkel metropolen.

In elk van beide gevallen moet de keuze om een meetwaarde uit een regressiemodel te weren goed worden gemotiveerd. Punten kunnen op verschillende manieren invloed uitoefenen:

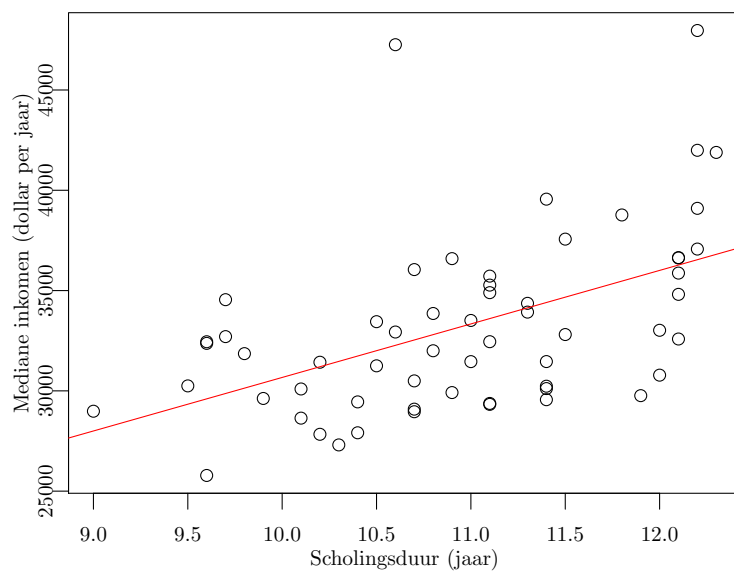
- als verticale outlier,
- als horizontale outlier,
- als invloedrijk punt.

De eerste soort outliers kan worden geïdentificeerd als die punten met de grootste waarden voor het gestandaardiseerde residu. Het is zeker niet de bedoeling om hier een bepaald percentage van de meest extreme punten weg te gooien. Enkel indien punten zich in de grafiek duidelijk van de andere distantiëren, komen ze eventueel in aanmerking als outlier, maar ze mogen pas definitief worden verwijderd uit het model indien ze een duidelijke en negatieve invloed hebben op het model, en indien het te verantwoorden is dat zij niet door dat model kunnen worden verklaard. Het opsporen van andere soorten outliers valt buiten bestek van deze handleiding maar wordt bijvoorbeeld behandeld in sectie 7.4 van [2].

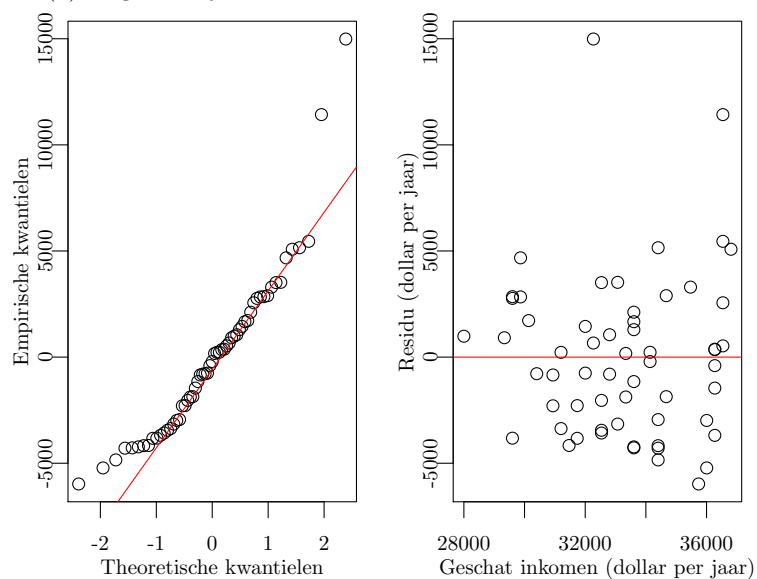
```

44 > resstand = rstandard(model1)
    > plot(sort(resstand))
46 > which(resstand>3)
    8 48
    8 47
48 > row.names(pollutie)[c(8,48)]
50 [1] "Bridgeport-Milford, CT" "San Francisco, CA"
    > points(c(59,58),resstand[c(8,47)], pch=16, col='red')
52 > model2 = lm(income[-c(8,48)]~Education[-c(8,48)])
    > summary(model2)

```

(a) Regressielijn voor inkomen in functie van studieduur



(b) Normale kwantielplot

(c) Residuplot

Figuur 5.2: Regressiemodel en modelveronderstellingen

```

54 ...
Coefficients:
56             Estimate Std. Err. t value Pr(>|t|)
(Intercept)      5970.1   5175.2   1.154   0.254
58 Education[-c(8, 48)] 2444.8   471.2   5.188 3.15e-06***
...
60 Multiple R-squared: 0.3286,      Adjusted R-squared: 0.3164
F-statistic: 26.92 on 1 and 55 DF,  p-value: 3.153e-06
62 > plot(income~Education)
> points(income[c(8,48)]~Education[c(8,48)],
64 +   pch=16, col='red')
> abline(model1, col='red')
66 > abline(model2, col='blue') # zie Figuur 5.3

```

De steden Bridgeport-Milford en San Francisco blijken een voor het regressiemodel onverklaarbaar hoog mediaan inkomen te hebben. Een model waarbij deze outliers zijn weggelaten,

$$\text{income} = 5970 + 2445 \cdot \text{Education}, \quad (5.2)$$

levert echter geen winst in de determinatiecoëfficiënt en verandert het model nauwelijks. Er is geen reden te bedenken waarom die twee steden zich van de andere onderscheiden, er is evenmin reden om aan te nemen dat de waarden voor `income` voor deze twee steden foutief zouden zijn. Het oorspronkelijke model (5.1) wordt behouden.

5.4 Voorspellingen en voorspellingsintervallen

Indien uiteindelijk een bevredigend model is gevonden, kan dit worden gebruikt om schattingen te doen:

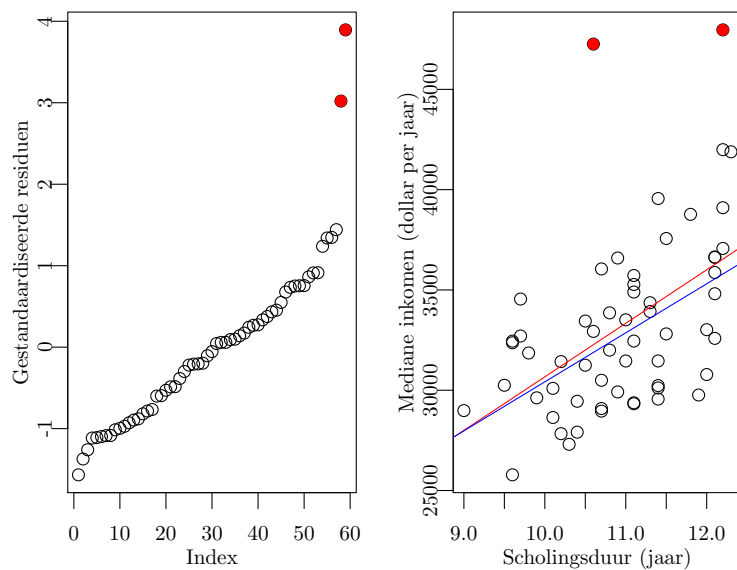
1. Een puntschatting $\hat{y}_0 = b_0 + \sum_{i=1}^{p-1} b_i x_{i,0}$;
2. Een betrouwbaarheidsinterval dat met kans $1 - \alpha$ de gemiddelde respons bij deze x_0 bevat;
3. Een voorspellingsinterval dat met dezelfde kans een individuele respons bij x_0 bevat.

Voorspellingen en voorspellingsintervallen worden berekend met `predict()`. Kansuitspraken betreffende de intervalschattingen zijn enkel geldig indien alle modelveronderstellingen zijn voldaan.

```

> predict(model1, data.frame(Education=10),
68 +   interval = "confidence", level = 0.95)
      fit      lwr      upr
70 1 30668.66 29127.47 32209.85
> predict(model1, data.frame(Education=10),
72 +   interval = "prediction", level = 0.95)
      fit      lwr      upr
74 1 30668.66 22732.07 38605.26
> betrouw = predict(model1,
76 +   interval = "confidence", level = 0.95)
> predictie = predict(model1,

```



(a) Gestandaardiseerde residuen (b) Model met en zonder outliers

Figuur 5.3: Opsporen van outliers

```

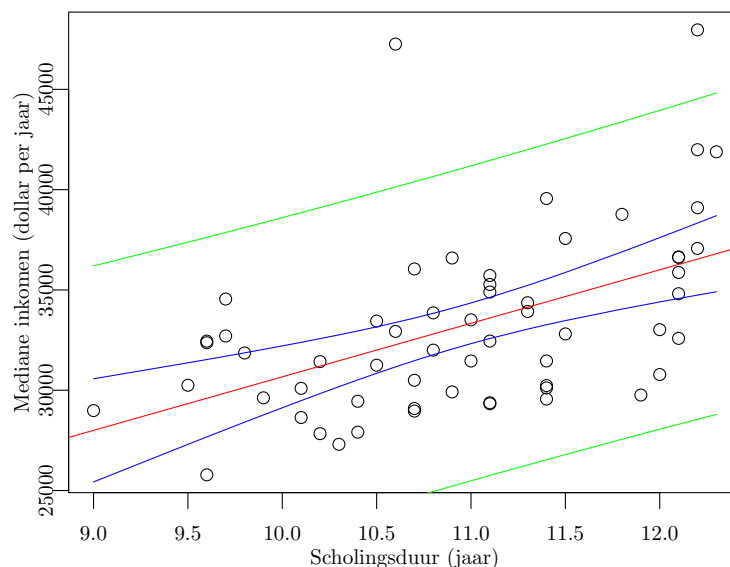
78 + interval = "prediction", level = 0.95)
> plot(income ~ Education)
80 > abline(model1, col='red')
> lines(sort(x_i), betrouwbh[order(x_i),2], col='blue')
82 > lines(sort(x_i), betrouwbh[order(x_i),3], col='blue')
> lines(sort(x_i), predictie[order(x_i),2], col='green')
84 > lines(sort(x_i), predictie[order(x_i),3], col='green') #
    zie Figuur 5.4

```

Het geschatte inkomen in steden met 10 jaar scholing bedraagt 30669 dollar. Het gemiddelde inkomen in steden met 10 jaar scholing ligt met 95% zekerheid tussen 29127 en 32210 dollar. Steden waar het mediaan aantal jaar opleiding 10 jaar bedraagt hebben 95% kans een mediaan inkomen tussen 22732 en 38605 dollar te hebben. Deze intervallen zijn niet zeer betrouwbaar gezien de residuen van een relatief kleine dataset niet normaal lijken te zijn. Betrouwbaarheids- en predictiebanden rond de regressierechte zijn weergegeven in figuur 5.4.

5.5 Meervoudige regressie

Indien meerdere verklarende veranderlijken worden gebruikt voor het opstellen van een regressiemodel, blijft de globale structuur van de analyse geldig, zij het met een aantal belangrijke verschillen. Ten eerste zal de determinatiecoëfficiënt doorgaans stijgen indien er meer veranderlijken in het model worden meegenomen. Daarom wordt er in de context van meervoudige regressie uitsluitend met de R^2_{adj} gewerkt. Ten tweede kan een model met meerdere veranderlijken dan wel significant zijn, daarom is iedere individuele regressor dat nog niet – om dit na te gaan, wordt per veranderlijke een bijkomende t -test uitgevoerd. Ten



Figuur 5.4: Regressierechte met betrouwbaarheids- en predictieband

derde geeft de residuplot (\hat{y}_i, e_i) niet noodzakelijk nog alle gewenste informatie en kan het ook nuttig zijn om per regressor ook naar de residuen ten opzichte van elke regressor afzonderlijk (x_i, e_i) te kijken.

In deze cursus wordt achterwaartse regressie gebruikt. De zoektocht naar een model start met een zo ruim mogelijke verzameling regressoren, waaruit dan de niet significante regressoren één voor één worden verwijderd. De veranderlijke met de grootste p -waarde, die niet significant van nul verschilt kan doorgaans immers uit het model worden geweerd zonder dat dit de determinatiecoëfficiënt of de significantie van de F -test schaadt.

Het aanpassen van een lopend regressiemodel kan met `update()`. De subset `$model` wordt hierdoor aangepast en kan door ontbrekende waarden telkens een ander aantal cases bevatten.

```

> loon = lm(income~Education+Mortality+pop+pop.house)
86 > summary(loon)
...
88 Coefficients:
      Estimate Std. Error t value Pr(>|t|)
(Intercept)  1.658e+04  1.739e+04   0.953   0.3448
Education    2.155e+03   7.248e+02   2.973   0.0044 **
Mortality    -5.414e+00   9.945e+00  -0.544   0.5884
pop          6.703e-04   3.581e-04   1.872   0.0667 .
94 pop.house  -8.706e+02   3.231e+03  -0.269   0.7886
...
96 Multiple R-squared:  0.3134,    Adjusted R-squared:  0.2626
> loon = update(loon, .~-pop.house)
98 > summary(loon)
...
100 Coefficients:
      Estimate Std. Error t value Pr(>|t|)

```

```

102 (Intercept)  1.404e+04  1.449e+04    0.969  0.33695
      Education  2.189e+03  7.076e+02    3.093  0.00311 **
104 Mortality   -6.161e+00  9.470e+00   -0.651  0.51805
      pop        7.009e-04  3.369e-04    2.081  0.04214 *
106 ...
      Multiple R-squared:  0.3125,      Adjusted R-squared:  0.275
108 > loon = update(loon, .~-Mortality)
      > summary(loon)
110 ...
      Coefficients:
112             Estimate Std. Error t value Pr(>|t|)
      (Intercept)  5.617e+03  6.484e+03    0.866  0.390011
114 Education      2.433e+03  5.965e+02    4.079  0.000145 ***
      pop          6.597e-04  3.291e-04    2.004  0.049887 *
116 ...
      Multiple R-squared:  0.3072,      Adjusted R-squared:  0.2825
118 > loon$model
      income Education      pop
120 1    29560      11.4  660328
      2    31458      11.0  835880
122 3    31856       9.8  635481
      ...

```

Door het verwijderen van de minst significante veranderlijke `pop.house` wordt het aanvankelijk randsignificante `pop` plots significant. Deze modelaanpassing gaat bovendien gepaard met een lichte stijging van de determinatiecoëfficiënt. Analooch blijkt ook de veranderlijke `Mortality` niet in het model thuis te horen. Uiteindelijk bevat het regressiemodel de twee verklarende veranderlijken `Education` en `pop`,

$$\text{income} = 5617 + 2433 \cdot \text{Education} + 6.597 \cdot 10^{-4} \cdot \text{pop}. \quad (5.3)$$

Door het toevoegen van de populatiegrootte wordt het inkomen meer verklaard dan door formule (5.1). Volgens dit nieuwe model stijgt het jaarinkomen in een stad met gemiddeld 2433 euro per extra jaar opleiding en met 659.7 dollar per miljoen inwoners. Om dit model te weerhouden moeten eerst de modelveronderstellingen worden nagegaan.

5.6 Transformaties van veranderlijken

Is het gemiddelde van de residuen niet constant nul ten opzichte van de verklarende veranderlijken, maar vertoont deze een gedrag $e_i = f(x_i)$, dan kan dit euvel mogelijks verholpen worden door de getransformeerde veranderlijke $f(X)$ aan het regressiemodel toe te voegen met `update(model, .~.-X+f(X))`.

Let op bij gebruik van polynomiale transformaties. Om technische redenen moeten termen die aritmetische bewerkingen bevatten worden toegevoegd met behulp van de operator `I()`, bijvoorbeeld in het geval van een kwadratische term met `update(model, .~.+I(X^2))`. Bovendien moeten in het algemeen alle lagere orde termen in het model blijven. Is de coëfficiënt b_{p-1} bij X^{p-1} significant verschillend van nul, dan moeten alle termen $b_i \cdot X^i$ van lagere orde $i < p - 1$ ook in het model blijven zelfs al verschillen deze b_i zelf niet significant van de nul.

De erg scheef verdeelde veranderlijke `pop` resulteert in een aberrante residuplot, zoals te zien in figuur 5.5 (a). In dit geval lijkt een logaritmische transformatie aangewezen.

```

124 > residuen = loon$residuals
    > plot(loon$model[,3], residuen)
126 > abline(h=0, col='red')
    > loon = update(loon, .~.-pop+log10(pop))
128 > summary(loon)
    ...
130 Coefficients:
            Estimate Std. Error t value Pr(>|t|)
132 (Intercept) -13896.6      9233.3  -1.505  0.137930
      Education   2246.0       592.9   3.788  0.000373 ***
134 log10(pop)   3751.5      1420.2   2.641  0.010679 *
    ...
136 Multiple R-squared: 0.3398,      Adjusted R-squared: 0.3162
      F-statistic: 14.41 on 2 and 56 DF,  p-value: 8.933e-06
138 > residuen = loon$residuals
    > plot(loon$model[,3], residuen)
140 > abline(h=0, col='red')                                     # zie Figuur 5.5

```

Door het vervangen van de veranderlijke `pop` door zijn tiendeling logaritme $\log_{10}(\text{pop})$ is de residuplot nu wel zeer aanvaardbaar. Bovendien is de determinatiecoëfficiënt gestegen en is de nieuwe coëfficiënt duidelijk signifikanter. Het weerhouden model is

$$\text{income} = -13897 + 2246 \cdot \text{Education} + 3752 \cdot \log_{10}(\text{pop}). \quad (5.4)$$

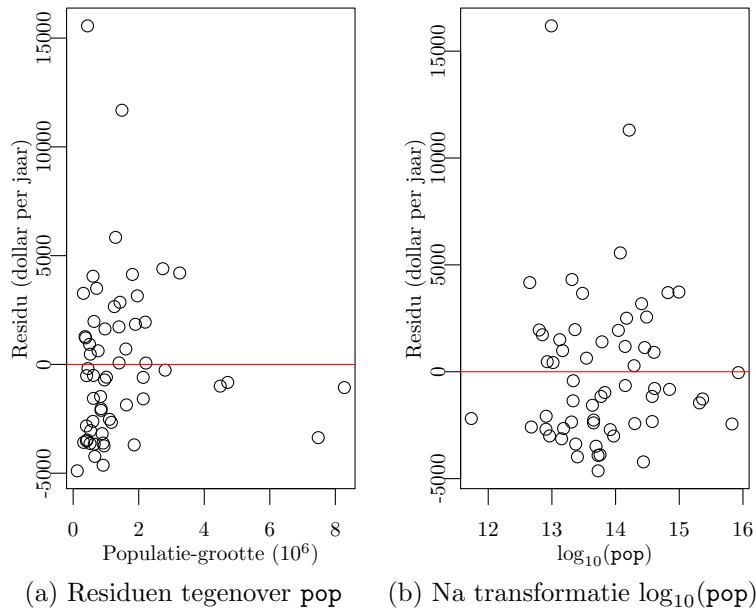
Het jaarinkomen neemt hier met gemiddeld 2246 dollar toe per extra jaar opleiding, zeer vergelijkbaar met modellen 5.1 en 5.3. Doordat de modelveronderstellingen hier beter voldaan zijn dan bij het vorige model, is een logaritmisch verband tussen inkomen en populatie-grootte meer plausibel dan een lineair. Het gemiddelde inkomen neemt toe met gemiddeld 3752 dollar *per vertienvoudiging* van het inwonersaantal. De intercept heeft hier opnieuw geen enkele betekenis, want deze ligt ver buiten de puntenwolk en het is niet relevant te spreken over een stad zonder inwoners of scholing.

In principe moet nu opnieuw worden nagegaan of er geen outliers zijn, maar deze analyse wijkt niet af van wat er in sectie 5.3 werd gedaan en wordt overgelaten aan de lezer.

5.7 Indicatorvariabelen

Regressie $Y = \beta_0 + \sum_{i=1}^{p-1} \beta_i \cdot X_i$ werd tot nu toe enkel besproken voor continue veranderlijken. Het is echter ook in het geval van een binaire veranderlijke B mogelijk om deze als verklarende veranderlijke toe te voegen. De groepen die door zo een veranderlijke worden bepaald, kunnen op twee manieren een rol spelen in het model.

Hoofdeffect: De term $\delta_0 \cdot B$ impliceert verschillende intercepten en beschrijft dus een systematisch verschil tussen beide groepen (dat echter meestal niet verder geïnterpreteerd kan worden als het interactie-effect significant is).



Figuur 5.5: Residuplots bij meervoudige regressie

Interactie-effect: De termen $\delta_i \cdot B \cdot X_i$ impliceren verschillende richtingscoëfficiënten voor alle afzonderlijke regressoren. Indien significant verschillend van nul betekent deze term dus een andere gemiddelde verandering van Y per eenheid toename in X_i voor beide groepen.

Vertrekkend van de vergelijking

$$Y = \beta_0 + \sum_{i=1}^{p-1} \beta_i \cdot X_i + \delta_0 \cdot B + \sum_{i=1}^{p-1} \delta_i \cdot B \cdot X_i$$

ontstaan dus twee modellen, deze waarvoor $B = 0$, waaruit de geïntroduceerde termen wegvallen, en deze waarvoor $B = 1$, wat aanleiding geeft tot coëfficiënten die het verschil tussen beide groepen beschrijven,

$$\begin{cases} \hat{y}_{B=0} &= b_0 + \sum_{i=1}^{p-1} b_i \cdot x_i \\ \hat{y}_{B=1} &= (b_0 + d_0) + \sum_{i=1}^{p-1} (b_i + d_i) \cdot x_i \end{cases}$$

Merk op dat het zonder meer toevoegen van een categorische veranderlijke met meer dan twee klassen aan een regressiemodel zinloos is aangezien de regressiecoëfficiënten automatisch een lineair verband tussen deze klassen impliceren. Het is wel degelijk mogelijk om regressie te doen met categorische veranderlijken met $n > 2$ niveaus, namelijk door $n - 1$ binaire dummy-veranderlijken te maken, maar deze techniek valt buiten het bestek van deze tekst.

Een indicatorvariabele in R aan een model toevoegen, gebeurt met een term $B \cdot X$. Let er op dat bij het gebruik van interacties de afzonderlijke termen X en B steeds in het model blijven, ook al zijn de corresponderende coëfficiënten eventueel niet significant verschillend van 0. Deze termen worden dus analoog behandeld als de intercept of lagere orde termen bij polynomiale transformaties.

```

> binair = arbeid=="laag"
142 > loon1 = update(loon,
+       ~.+binair*Education+binair*log10(pop))
144 > summary(loon1)
...
146 Coefficients:
              Estimate Std.Err. t value Pr(>|t|)
148 (Intercept)      -11341.5   10290.9   -1.102   0.27540
      Education         2102.6     661.3    3.180   0.00246 **
150 log10(pop)         3561.7    1460.6    2.438   0.01814 *
      binairTRUE      -25450.8   55667.8   -0.457   0.64940
152 Education:binairTRUE  -1139.8    3587.0   -0.318   0.75191
      log10(pop):binairTRUE  6670.1   14755.6    0.452   0.65308
154 ...
Residual standard error: 3747 on 53 degrees of freedom
156 > x = data.frame(Education=10, pop=500000, binair=TRUE)
> predict(loon1, x, interval="confidence", level=0.95)
158      fit      lwr      upr
1 31146.07 25058.02 37234.12

```

Er worden twee verschillende modellen gevonden:

$$\begin{cases} \text{income}_{\text{midden-hoog}} = -11342 + 2103 \cdot \text{Education} + 3562 \cdot \log_{10}(\text{pop}) \\ \text{income}_{\text{laag}} = -36792 + 963 \cdot \text{Education} + 10232 \cdot \log_{10}(\text{pop}) \end{cases} \quad (5.5)$$

De kleinste kwadratenmethode geeft aanleiding tot twee ogenschijnlijk compleet verschillende modellen: in steden met een laag aantal arbeiders is het effect van scholing een pak minder (-1140) maar weegt de populatiegrootte veel zwaarder door (6670). Geen van de nieuwe veranderlijken is echter significant en deze kunnen één voor één weer uit het model worden gehaald. Er blijkt dat het mediaan inkomen niet afhangt van het feit of er een lage dan wel hogere proportie arbeiders in een stad is. Het eerder gevonden model (5.4) is dus meer aangewezen om voorspellingen te doen.

5.8 Kwaliteit van een regressiemodel

Blijkbaar geeft vergelijking (5.4) het meest geschikte model om het mediane inkomen in een stad te verklaren uit de beschikbare veranderlijken. Ondanks de uitgebreide analyse, is dit model toch niet ideaal. Volgende aspecten bepalen immers de kwaliteit van een regressiemodel.

Significantie. Het loon in een stad stijgt naarmate de opleidingsgraad en (het logaritme van) het inwonersaantal hoger is. Dit is een systematisch effect, niet te wijten aan toeval want zowel de F -test als beide t -testen zijn zeer significant.

Juistheid. De vorm van de vergelijking lijkt juist te zijn, want de residuplot in figuur 5.5 (b) lijkt aan de modelveronderstellingen te voldoen: inkomen stijgt lineair met opleidingsgraad en logaritmisch met het bevolkingsaantal.

Betrouwbaarheid. De betrouwbaarheid om aan voorspellingen te doen is beperkt omdat de residuen niet normaal verdeeld zijn. Hierdoor zullen kans-

uitspraken met betrekking tot betrouwbaarheids- en voorspellingsintervallen slechts bij benadering geldig zijn.

Volledigheid. De determinatiecoëfficiënt is laag, wat er op wijst dat het inkomen niet enkel door het opleidingsniveau en de populatiegrootte kan worden verklaard. Vermoedelijk zijn er nog andere factoren die impact hebben op het loon maar die niet in de dataset zijn opgenomen. Dit resulteert in brede voorspellingsintervallen.

Tabel 5.1: Belangrijkste statistieken in verband met lineaire regressie

Meervoudig regressiemodel $Y_i = a + \sum_{j=1}^{p-1} b_j \cdot x_{i,j} + \epsilon_i$ met $\epsilon_i \sim N(0, \sigma^2)$

Kleinste kwadratenschatting voor $p = 2$

$$y_i = \hat{a} + \hat{b} \cdot x_i + r_i = \hat{y}_i + r_i \text{ met } \begin{cases} \hat{a} &= \bar{y} - \hat{b} \cdot \bar{x} \\ \hat{b} &= \frac{\text{cov}(x, y)}{s_x^2} \end{cases}$$

t-test voor individuele coëfficiënten voor $p = 2$

$$T_a = \frac{\hat{a} - a}{s \sqrt{\frac{1}{n} + \frac{\bar{x}^2}{(n-1)s_x^2}}} \sim t_{n-2} \text{ en } T_b = \frac{\hat{b} - b}{s \sqrt{\frac{1}{(n-1)s_x^2}}} \sim t_{n-2}$$

Determinatiecoëfficiënt $R^2 = \frac{\sum_{i=1}^n (\hat{y}_i - \bar{y})^2}{\sum_{i=1}^n (y_i - \bar{y})^2} = \frac{SSM}{SST} = 1 - \frac{SSE}{SST}$

Aangepaste determinatiecoëfficiënt $R_{\text{adj}}^2 = 1 - \frac{SSE/(n-p)}{SST/(n-1)}$

F-test voor totaal regressiemodel $F = \frac{SSM/(p-1)}{SSE/(n-p)} \sim F_{p-1, n-p}$

Gemiddelde kwadratische fout $MSE = \hat{\sigma}^2 = s^2 = \frac{1}{n-p} \sum_{i=1}^n r_i^2$

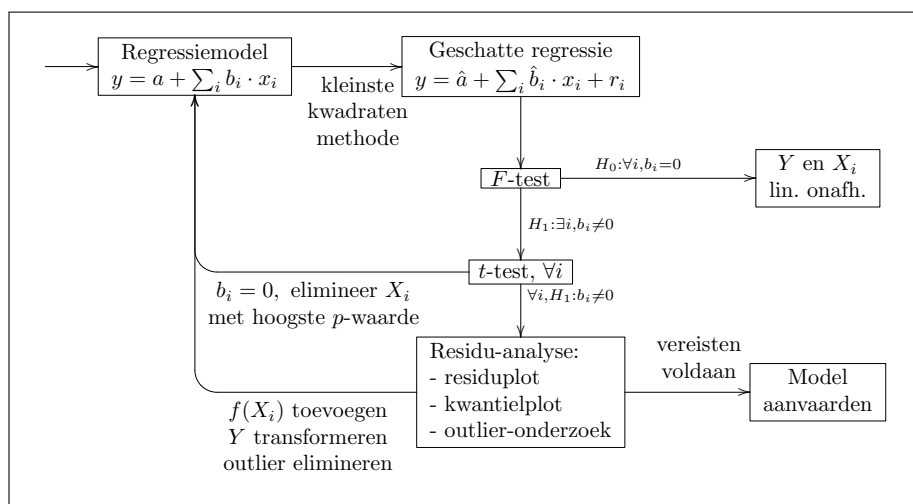
Betrouwbaarheidsinterval voor $p = 2$, $P(Y_0 \in [A, B]) = 1 - \alpha$ met

$$[A, B] = [\hat{a} + \hat{b}x_0 - t_{n-2, \frac{\alpha}{2}} s \sqrt{\frac{1}{n} + \frac{(x_0 - \bar{x})^2}{(n-1)s_x^2}}, \hat{a} + \hat{b}x_0 + t_{n-2, \frac{\alpha}{2}} s \sqrt{\frac{1}{n} + \frac{(x_0 - \bar{x})^2}{(n-1)s_x^2}}]$$

Predictie-interval voor $p = 2$, $P(Y_0 \in [A, B]) = 1 - \alpha$ met

$$[A, B] = [\hat{a} + \hat{b}x_0 - t_{n-2, \frac{\alpha}{2}} s \sqrt{\frac{n+1}{n} + \frac{(x_0 - \bar{x})^2}{(n-1)s_x^2}}, \hat{a} + \hat{b}x_0 + t_{n-2, \frac{\alpha}{2}} s \sqrt{\frac{n+1}{n} + \frac{(x_0 - \bar{x})^2}{(n-1)s_x^2}}]$$

Tabel 5.2: Stappenplan voor achterwaartse regressie



Tabel 5.3: Regressie in R

	<code>FIT = lm(Y~X)</code>	# eenvoudig lineair model
2	<code>plot(Y~X), abline(FIT)</code>	# voorstelling van het model
	<code>FIT = lm(Y~X1+X2)</code>	# meervoudig lineair model
4	<code>FIT = update(FIT, .~.+X3)</code>	# regressor toevoegen
	<code>FIT = update(FIT, .~-X2)</code>	# regressor verwijderen
6	<code>FIT = update(FIT, .~-1)</code>	# intercept verwijderen
	<code>FIT = update(FIT, .~.+f(X))</code>	# getransformeerde toevoegen
8	<code>FIT = update(FIT, .~.+I(X^2))</code>	# polynomiale transformatie
	<code>class(FIT)</code>	# "lm"
10	<code>attributes(FIT)</code>	# alle attributen bij model
	<code>summary(FIT)</code>	# belangrijkste statistieken
12	<code>FIT\$model</code>	# de gebruikte data
	<code>FIT\$fitted.values</code>	# de modelschattingen
14	<code>FIT\$residuals</code>	# residuen
	<code>rstandard(FIT)</code>	# gestandaardiseerde residuen
16	<code>predict(FIT,</code>	# voorspellingen
	<code>data.frame(X=x,Y=y),</code>	# waarden van de regressoren
18	<code>interval = "confidence",</code>	# "none" of "prediction"
	<code>level = 0.95)</code>	# betrouwbaarheidsniveau

Hoofdstuk 6

Variantie-analyse

Twee vragen zijn in de vorige hoofdstukken nog niet beantwoord:

- Hoe meer dan twee gemiddelden tegelijk vergelijken?
- Hoe categorische data gebruiken om een veranderlijke te verklaren?

Variantie-analyse (ANOVA) zal in beide gevallen een antwoord bieden. De naam is enigszins misleidend: ANOVA bestudeert weliswaar varianties, maar doet in het algemeen uitspraken over gemiddelden.

6.1 One-way ANOVA

Het eenvoudigste model waarop ANOVA wordt toegepast is het geval waarin de gemiddelden tussen p verschillende groepen worden vergeleken, bepaald door één categorische veranderlijke X met uitkomsten $\Omega_X = \{x_0, x_1, \dots, x_{p-1}\}$. De nulhypothese stelt dat de gemiddelden in alle groepen gelijk zijn aan elkaar,

$$H_0 : \mu_0 = \mu_1 = \dots = \mu_{p-1}$$

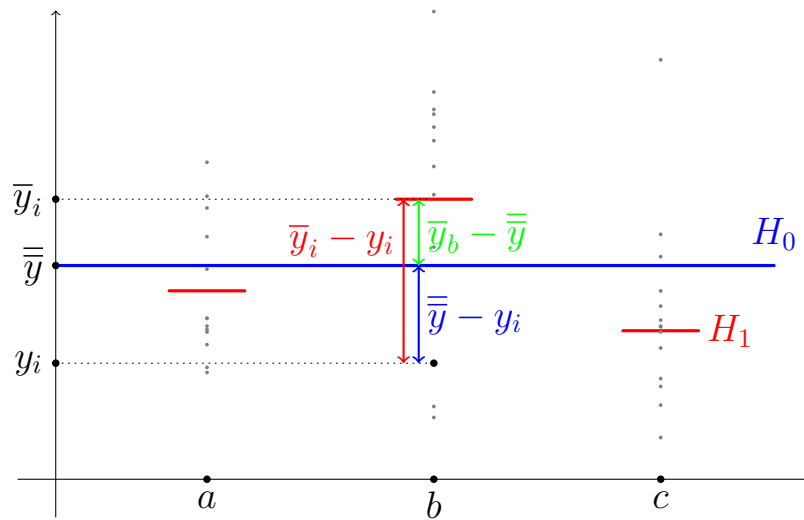
en dus ook gelijk aan het gemiddelde μ over alle groepen heen.

Staat de notatie \bar{y}_i voor het gemiddelde van de meetwaarden in de groep waartoe y_i behoort en is $\bar{\bar{y}}$ het gemiddelde over alle groepen heen, dan moeten als de nulhypothese geldig is de groepsgemiddelden \bar{y}_i in een steekproef op een toevalsfout na gelijk zijn aan het globale gemiddelde $\bar{\bar{y}}$ (zie Figuur 6.1). Dit leidt tot analoge kwadratensommen als bij regressie, die in deze context vaak anders benoemd worden:

- $SST = \sum_{i=1}^n (y_i - \bar{\bar{y}})^2 = \text{Sum of Squares Total}$
- $SSM = \sum_{i=1}^n (\bar{y}_i - \bar{\bar{y}})^2 = \text{Sum of Squares Between Groups}$
- $SSE = \sum_{i=1}^n (y_i - \bar{y}_i)^2 = \text{Sum of Squares Within Groups}$

Dezelfde statistiek, onder dezelfde voorwaarden, laat toe om de nulhypothese te testen:

$$F = \frac{SSM/(p-1)}{SSE/(n-p)}.$$



Aangezien er geen kleinstekwadratenschattingen moeten worden berekend, maar enkel gemiddelden, zijn de schattingen bij one-way ANOVA zeer eenvoudig te berekenen. Toch is dit in wezen een regressiemodel, namelijk

$$y = \beta_0 + \sum_{i=1}^{p-1} \beta_i B_i,$$

met indicatorvariabelen B_i , $i = 1, \dots, p - 1$ die voldoen aan

$$B_i = \begin{cases} 1 & \text{als de } i\text{-e observatie tot groep } i \text{ behoort} \\ 0 & \text{als de } i\text{-e observatie } \textit{niet} \text{ tot groep } i \text{ behoort} \end{cases}$$

In dit model is de parameter β_0 het gemiddelde μ_0 in de referentiegroep en meten de andere coëfficiënten $\beta_i = \mu_i - \mu_0$ het effect binnen elke groep ten opzichte van de referentiegroep. De nulhypothese van de globale F -test en die bij one-way ANOVA zijn dus equivalent,

$$H_0 : \beta_1 = \beta_2 = \dots = \beta_{p-1} = 0 \Leftrightarrow H_0 : \mu_0 = \mu_1 = \dots = \mu_{p-1}.$$

Dit schetst hoe one-way ANOVA toelaat om meerdere groepsgemiddelden te vergelijken, hoe categorische veranderlijken in een regressiemodel te gebruiken en meteen ook dat beide equivalent zijn. Alle statistieken, interpretaties en voorwaarden uit het vorige hoofdstuk zijn hier ook van toepassing. In het bijzonder moet voldaan zijn aan de Gauss-Markov-voorwaarden,

$$\varepsilon_i \sim N(0, \sigma^2).$$

De residuen moeten zich gedragen als een zuiver lukrake steekproef uit een normaalverdeling met een constante variantie. Aangezien bij een one-way ANOVA-model het gemiddelde in elke groep afzonderlijk wordt geschat, is automatisch voldaan aan de voorwaarde dat de residuen gemiddelde nul moeten hebben.

Voorbeeld 11. Verschilt het gemiddelde inkomen naargelang de regio waarin een stad zich bevindt?

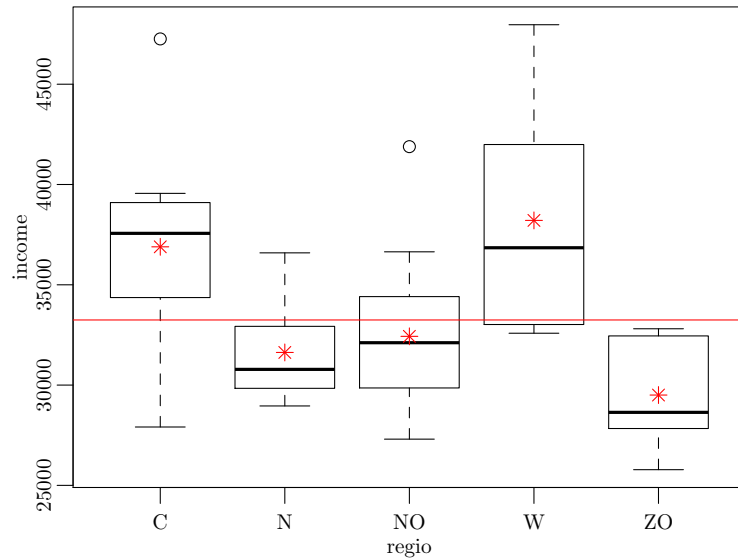
```

160 > xibar = tapply(income, regio, mean, na.rm=TRUE); xibar
      C      N      NO      W      ZO
162 36893 31626 32430 38209 29503
> tapply(income, regio, sd, na.rm=TRUE)
      C      N      NO      W      ZO
164 5389 2439 3251 5863 3041
166 > xbar = mean(income, na.rm=TRUE); xbar
[1] 33246.66
168 > income.aov = aov(income~regio)
> summary(income.aov)
170 Df      Sum Sq   Mean Sq F value    Pr(>F)
    regio         4 392868884 98217221  6.909 0.000144 ***
172 Residuals    54 767628698 14215346
    [...]
174 > plot(income~regio)
> abline(h=xbar, col='red')
176 > points(1:5, xibar, pch=8, col='red') # zie Figuur 6.2 (b)
> plot(income.aov$residuals~income.aov$fitted.values)
178 > abline(h=0, col='red') # zie Figuur 6.2 (c)
> qqnorm(income.aov$residuals)
180 > qqline(income.aov$residuals, col='red') # zie Figuur 6.2 (b)
> model.tables(income.aov, type='means')
182 Tables of means
Grand mean
184      33246.66
regio
186      C      N      NO      W      ZO
36893 31627 32431 38210 29504
188 rep    9    15    24    6    5
> model.tables(income.aov, type='effects')
190 Tables of effects
regio
192      C      N      NO      W      ZO
3647 -1620 -816 4963 -3743
194 rep    9    15    24    6    5

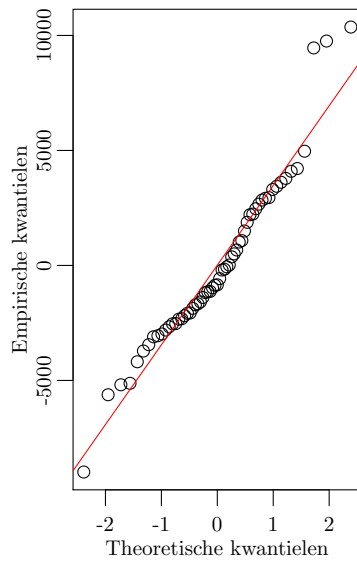
```

Het gemiddelde inkomen in de verschillende regio's loopt van 29503 dollar in het Zuidoosten tot 38209 dollar in het Westen, met 33247 dollar als globaal gemiddelde. Om na te gaan of deze verschillen significant zijn, wordt een one-way ANOVA model opgesteld dat zeer significant is ($p \approx 0.0001$), de gegevens wijzen dus op verschillende regionale gemiddelden. Het commando `model.tables()` geeft een overzicht van gemiddelden en/of effecten.

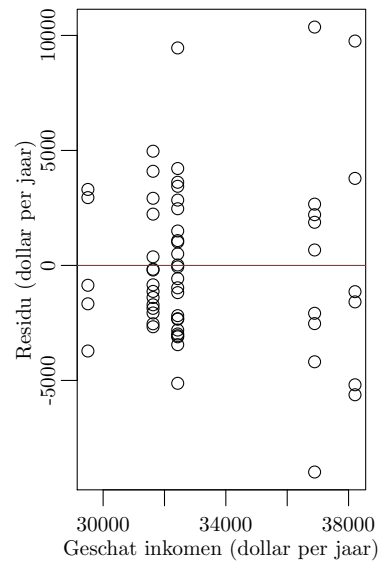
De uitspraken in dit voorbeeld zijn betrouwbaar omdat de residuen behoorlijk goed aan de Gauss-Markov voorwaarden voldoen (zie Figuur 6.2). De varianties in elke groep hebben dezelfde grootte-orde en de residuen wijken niet al te sterk af van de normale verdeling: de verdeling van de teststatistiek is robuust voor dit soort afwijkingen.



(a) Gecategoriseerde boxplots van **income** ten opzichte van **regio** met globaal gemiddelde (horizontale lijn) en groepsgemiddelden (asterisken)



(b) Normale kwantielplot



(c) Residuplot

Figuur 6.2: Voorstelling van en modelveronderstellingen bij het one-way ANOVA model

6.2 Paarsgewijze vergelijking tussen groepen

Als kan worden besloten dat de groepsgemiddelden onderling verschillen, is een volgende evidente vraag tussen welke groepen het verschil significant is. Een eerste naïeve manier om dit te beantwoorden, zou kunnen zijn om paarsgewijze *t*-testen te doen. Bij p groepen zijn er echter $p \cdot (p - 1)/2$ vergelijkingen: deze simultaan uitvoeren verlaagt drastisch de betrouwbaarheid.

De aangewezen manier om dit soort *post-hoc* testen te doen, is om het significantieniveau bij de individuele testen te verlagen om de globale betrouwbaarheid op peil te houden. Een zeer conservatieve methode die algemeen toepasbaar is, is de Bonferroni-correctie. Deze schrijft voor om bij m afzonderlijke hypothesetesten het significantieniveau α/m te gebruiken opdat de globale betrouwbaarheid minstens α zou blijven. In het specifieke geval van one-way ANOVA is de Tukey-test een krachtiger alternatief.

Met het commando `pairwise.t.test()` kan het verschil tussen gemiddelden van verschillende groepen twee aan twee systematisch worden vergeleken. De optie `p.adjust.method` bepaalt of en hoe de p -waarden worden aangepast om de globale betrouwbaarheid te garanderen. Om te vergelijken met afzonderlijke *t*-tests mag niet met een gepoolde variantieschatting (`pool.sd=FALSE`) worden gewerkt. Voor one-way ANOVA modellen is er de afzonderlijke methode `TukeyHSD()`. Aangezien ANOVA-modellen homogeniteit van varianties als assumptie hebben, gebruikt de Tukey-test een gepoolde variantieschatting. Om een faire vergelijking met de Bonferroni-correctie te maken is een gepoolde variantieschatting dus nodig (`pool.sd=FALSE`). Met het commando `p.adjust()` kan de globale significantie bij het uitvoeren van herhaalde individuele testen worden nagerekend.

Voorbeeld 12. Tussen welke regio's verschilt het gemiddelde inkomen?

```
> t.test(income[regio=="N"], income[regio=="C"])
196 Welch Two Sample t-test
    t = -2.7666, df = 10.001, p-value = 0.0199
198 [...]
> pairwise.t.test(income, regio,
200 +   p.adjust.method='none', pool.sd=FALSE)
Pairwise comparisons using t tests with non-pooled SD
202      C      N      NO      W
N  0.0199      -      -      -
204 NO  0.0414  0.3855      -      -
W   0.6692  0.0394  0.0604      -
206 ZO  0.0066  0.2078  0.1007  0.0139
> pairwise.t.test(income, regio,
208 +   p.adjust.method='bonferroni', pool.sd=FALSE)
Pairwise comparisons using t tests with non-pooled SD
210      C      N      NO      W
N   0.199      -      -      -
212 NO  0.414  1.000      -      -
W   1.000  0.394  0.604      -
214 ZO  0.066  1.000  1.000  0.139
> p.adjust(c(.0199, .0414, .3855, .6692, .0394,
216 +   .0604, .0066, .2078, .1007, .0139),
+   method='bonferroni')
218 [1] 0.199 0.414 1.000 1.000 0.394 0.604 0.066
```

```

[8] 1.000 1.000 0.139
220 > pairwise.t.test(income, regio,
+       p.adjust.method='bonferroni', pool.sd=TRUE)
222 Pairwise comparisons using t tests with pooled SD
      C      N      NO      W
224 N  0.0165 -      -      -
      NO 0.0377 1.0000 -      -
226 W  1.0000 0.0066 0.0144 -
      ZO 0.0090 1.0000 1.0000 0.0035
228 > TukeyHSD(income.aov)
      Tukey multiple comparisons of means
      diff      lwr      upr      p adj
230 N-C    -5266.422 -9752.7062 -780.1382 0.0137153
232 NO-C   -4462.597 -8621.4938 -303.7006 0.0296530
      W-C    1316.611 -4291.2439 6924.4661 0.9634925
234 ZO-C   -7389.622 -13324.4181 -1454.8263 0.0077131
      NO-N     803.825 -2698.2793 4305.9293 0.9663394
236 W-N     6583.033  1443.3493 11722.7173 0.0057303
      ZO-N   -2123.200 -7617.7533 3371.3533 0.8106343
238 W-NO    5779.208   922.6634 10635.7532 0.0120691
      ZO-NO -2927.025 -8157.6839 2303.6339 0.5169225
240 ZO-W   -8706.233 -15149.1682 -2263.2985 0.0031370

```

Zonder correctie van de p -waarden zijn vijf verschillen significant en een zesde randsignificant. Bij het simultaan uitvoeren van tien paarsgewijze vergelijkingen, is een aanpassing van het verwerpingsniveau (of dus van de p -waarden) echter noodzakelijk. Bonferroni-correctie garandeert in alle omstandigheden dat de globale betrouwbaarheid bewaard blijft en toont hier (zonder gepoolde variantieschatting!) geen enkel significant verschil meer. Het gebruik van de gepoolde variantieschatting compenseert in dit voorbeeld blijkbaar ruimschoots correctie voor simultaan testen: zowel met Bonferroni-correctie als via de Tukey-methode zijn er zes significant verschillende paren. Verifieer dat de p -waarde bij de Tukey-methode steeds lager ligt dan door Bonferroni-correctie.

Conclusie is dat er twee duidelijk afgeleiden groepen regio's zijn, C-W en N-NO-ZO. Verschillen binnen deze groepen zijn niet significant, verschillen tussen de groepen zijn dat wel. Vaak zullen resultaten niet zo scherp afgelijnd zijn als hier en is een conclusie minder eenduidig.

6.3 Geneste modellen

Neem aan dat er twee geneste modellen overwogen worden: elke groep uit het eerste model is verdeeld in één of meerdere groepen in het tweede model. De ANOVA-modellen hebben dezelfde totale som van kwadraten SST, samengesteld als volgt,

$$\left. \begin{array}{l} \text{SST} = \text{SSM}_1 + \text{SSE}_1 \\ \text{SST} = \text{SSM}_2 + \text{SSE}_2 \end{array} \right\} \Rightarrow \text{SST} = \underbrace{\text{SSM}_1}_{\text{model 1}} + \underbrace{(\text{SSE}_1 - \text{SSE}_2)}_{\text{verdere opsplitsing}} + \text{SSE}_2.$$

Is dan model 1 te verkiezen, dat minder parameters bevat ($p_1 \leq p_2$) en dus eenvoudiger is? Of is model 2 beter, dat kleinere residuen heeft ($\text{SSE}_2 \leq \text{SSE}_1$) en dus meer verklaart? Een formele test bepaalt of de extra som van kwadraten

$SSE_1 - SSE_2$ significant groter is dan nul gegeven het aantal extra parameters $p_2 - p_1$,

$$\frac{(SSE_1 - SSE_2)/(p_2 - p_1)}{SSE_2/(n - p_2)} \sim F_{p_2 - p_1, n - p_2}. \quad (6.1)$$

Voorbeeld 13. Is het zinvol om vijf afzonderlijke regionale gemiddelden te werken, of volstaat het om de ruwere geografische indeling in twee groepen te gebruiken: centrum en westelijke deel C-W enerzijds, oostelijke en noordelijke deel N-NO-ZO anderzijds?

```

> regioCW = regio=="C"|regio=="W"
242 > tapply(income, regioCW, mean, na.rm=TRUE)
FALSE TRUE
244 31823 37419
> income.bis = aov(income~regioCW)
246 > summary(income.bis)
              Df    Sum Sq   Mean Sq F value    Pr(>F)
248 regioCW      1 350291959 350291959    24.64 6.59e-06 ***
   Residuals    57 810205623 14214134
250 > anova(income.bis, income.aov)
Analysis of Variance Table
252 Model 1: income ~ regioCW
   Model 2: income ~ regio
254   Res.Df    RSS Df Sum of Sq    F Pr(>F)
      1     57 810205623
256      2     54 767628698    3 42576925 0.9984 0.4007
> income.null = aov(income~1)
258 > summary(income.null)
              Df    Sum Sq   Mean Sq F value    Pr(>F)
260 Residuals    58 1.16e+09 20008579
> anova(income.null, income.bis)
262 Analysis of Variance Table
   Model 1: income ~ 1
264   Model 2: income ~ regioCW
      Res.Df    RSS Df Sum of Sq    F    Pr(>F)
266      1     58 1160497581
      2     57 810205623    1 350291959 24.644 6.594e-06 ***

```

Model 1 is het model met twee groepen, model 2 is het fijnere model met vijf groepen uit voorbeeld 11. Net zoals model 2 ($p \approx 10^{-4}$) toont model 1 ($p \approx 10^{-6}$) dat een opdeling in groepen leidt tot significant verschillende gemiddelden. De afgeronde groepsgemiddelden (in duizenden dollars) en kwadratensommen van beide modellen en het null model (zie verder) zijn samengevat in de tabel hieronder.

	p	C	W	N	NO	ZO	SSE	$n - p$
Model 0	1	33					$116 \cdot 10^7$	58
Model 1	2	37		32			$81 \cdot 10^7$	57
Model 2	5	37	38	32	32	30	$77 \cdot 10^7$	54

Het fijnere model verklaart niet significant meer ($p \approx 40\%$). Een model met slechts twee groepen is dus eenvoudiger, verklaart essentieel evenveel en is dus te verkiezen.

De laatste drie commando's tonen dat de globale F -test voor one-way ANOVA hetzelfde is als het vergelijken van dat model het zogenaamde *null model*, het

model dat slechts één groep bevat en dus als enige schatter het globale gemiddelde heeft. De residuele som van kwadraten van het null model is gelijk aan de totale som van kwadraten (RSS in R komt overeen met SSE in de cursus).

6.4 Partiële F -test

Formule (6.1) uit de vorige sectie, is algemener toepasbaar. Dezelfde techniek kan gebruikt worden om simultaan te testen of een aantal q veranderlijken in een model significant zijn.

Voorbeeld 14. Verklaren de meteorologische parameters vochtigheid, regenval, Januari- en Juli-temperatuur simultaan een deel van het inkomen in een stad?

```

268 > model1 = lm(income~log10(pop)+Education)
269 > model2 = lm(income~log10(pop)+Education
270 + JanTemp+JulyTemp+Rain+RelHum)
271 > summary(model2)
272 Coefficients:
273 Estimate Std. Error t value Pr(>|t|)
274 (Intercept) -2128.344 15842.674 -0.134 0.89365
275 log10(pop) 3320.473 1578.750 2.103 0.04030 *
276 Education 1893.587 683.250 2.771 0.00772 **
277 JanTemp 89.667 103.006 0.871 0.38803
278 JulyTemp -185.572 271.282 -0.684 0.49698
279 Rain -11.310 21.948 -0.515 0.60850
280 RelHum 1.238 108.468 0.011 0.99094
281 [...]
282 > anova(model1,model2)
283 Analysis of Variance Table
284 Model 1: income ~ log10(pop) + Education
285 Model 2: income ~ log10(pop) + Education + JanTemp +
286 JulyTemp + Rain + RelHum
287 Res.Df RSS Df Sum of Sq F Pr(>F)
288 1 56 766166243
289 2 52 741111963 4 25054280 0.4395 0.7795

```

De partiële F -test vergelijkt twee modellen met elk een verschillende som van kwadraten van de residuen SSE (genoteerd als RSS in het programma R):

	SSE	p
Model 1	$0.77 \cdot 10^9$	3
Model 2	$0.74 \cdot 10^9$	7

De partiële F -test bepaalt opnieuw of de extra som van kwadraten $SSE_1 - SSE_2$ significant groter is dan nul gegeven het aantal extra parameters $p_2 - p_1$. Dit resulteert in de statistiek $F = 0.44$ met p -waarde 78%.

Elk afzonderlijk zijn de veranderlijken **JanTemp**, **JulyTemp**, **RelHum** en **Rain** niet significant in het regressiemodel (dit volgt uit de t -testen), maar ook simultaan verklaren ze geen significant deel van de variatie in het inkomen in de verschillende steden.

In het algemeen is het niet onmogelijk dat coëfficiënten die afzonderlijk niet significant zijn, simultaan toch een niet verwaarloosbaar deel van de variatie in de respons verklaren, en wel om twee redenen:

- Simultane uitspraken op basis van meerdere p -waarden vereist correctie van het significantieniveau, om de betrouwbaarheid van de globale uitspraak te garanderen (zie ook eerder).
- Waar een t -test enkel rekening houdt met de standaarddeviatie van de individuele coëfficiënt, gebruikt de partiële F -test ook informatie over de correlatie tussen de verschillende coëfficiënten.

6.5 ANCOVA

Worden categorische en continue voorspellers door elkaar in een regressiemodel gebruikt, dan spreekt men van ANCOVA. Net zoals bij one-way ANOVA wordt een categorische veranderlijke met p uitkomsten in zo een model voorgesteld door $p - 1$ indicator- of dummyveranderlijken. De vraag die hier kan worden gesteld is of de regressieformule gelijk is over alle verschillende groepen, dan wel of er verschillen zijn tussen de groepen onderling. Omdat het vaak relevant is te weten of de coëfficiënt bij een continue regressor verschilt tussen twee groepen, worden vaak ook interactietermen tussen de continue veranderlijken en de indicatorvariabelen in het model opgenomen (zie ook sectie 5.7).

Voorbeeld 15. Uit voorgaande is geweten dat het gemiddelde inkomen in een stad afhangt van de scholingsduur. Verschilt deze afhankelijkheid naargelang de regio, of kan er één formule worden gehanteerd voor alle steden in de U.S.?

```

290 > income.lm = lm(income~Education)
      > income.regio = lm(income~Education*regio)
292 > anova(income.regio)
      Df      Sum Sq      Mean Sq F value      Pr(>F)
294 Education      1 298869383 298869383 28.1335 2.705e-06 ***
      regio      4 192937739 48234435 4.5405 0.003369 **
296 Education:regio 4 148151095 37037774 3.4865 0.013905 *
      Residuals    49 520539364 10623252
298 > summary(income.regio)
      Estimate Std. Error t value Pr(>|t|)
300 (Intercept)      18326.9      15556.3    1.178 0.24445
      Education      1654.4      1382.8    1.196 0.23729
302 regioN          26816.7      22340.3    1.200 0.23577
      regioNO       -11231.8      17705.1   -0.634 0.52878
304 regioW          -62036.8      21716.5   -2.857 0.00627 **
      regioZO       -29037.9      26983.2   -1.076 0.28713
306 Education:regioN  -2886.2      2010.3   -1.436 0.15744
      Education:regioNO  719.9      1592.5    0.452 0.65324
308 Education:regioW  51113.6      17905.6    2.855 0.00630 **
      Education:regioZO  2175.5      2510.4    0.867 0.39038
310 > betas = income.regio$coefficients; betas
      (Intercept)      Education      regioN      regioNO      regioW
312 18326      1654      26816      -11231      -620368
      regioZO Edu:regioN Edu:regioNO Edu:regioW Edu:regioZO
314 -29037      -2886      719      51113      2175
      > intercept = c(betas[1], betas[1] + betas[3:6]); intercept
316 (Intercept)      regioN      regioNO      regioW      regioZO
      18326.886 45143.574 7095.055 -602041.900 -10710.966
318 > slope = c(betas[2], betas[2] + betas[7:10]); slope

```

```

320 Education      Edu:regionN Edu:regionO Edu:regionW Edu:regionZ
      1654          -1231          2374          52768          3829
> plot(income~Education,col=region)
322 > for (k in 1:5) { abline(intercept[k],slope[k],col=k) }
> legend(9,48000,levels(region),col=1:5,lty=1) # zie Figuur 6.3

```

De interactie tussen `Education` en `region` is significant $p = 0.01$ en wijst er op dat het effect van scholing op het loon in de verschillende regio's anders is. Dit resulteert in een vergelijking per regio als volgt

$$\left\{ \begin{array}{llll} \text{income}_C & = & 18327 & + & 1654 \cdot \text{Education} \\ \text{income}_N & = & 45144 & - & 1232 \cdot \text{Education} \\ \text{income}_O & = & 7095 & + & 2374 \cdot \text{Education} \\ \text{income}_W & = & -602042 & + & 52768 \cdot \text{Education} \\ \text{income}_Z & = & -10711 & + & 3830 \cdot \text{Education} \end{array} \right.$$

De vergelijkingen op zich zijn in dit model niet zo betrouwbaar omdat er veel te weinig gegevens in elke groep zijn om een model op te bouwen (amper 5 steden in het Zuidoosten en 6 in het Westen), maar de gegevens volstaan om te beslissen dat er verschillen zijn tussen de regio's.

Testen welke coëfficiënten precies verschillend zijn, kan met de t -testen (suboptimaal omdat dit hercodering en p -waarde-correctie vereist), meer specifieke methoden vallen buiten bestek van deze cursus.

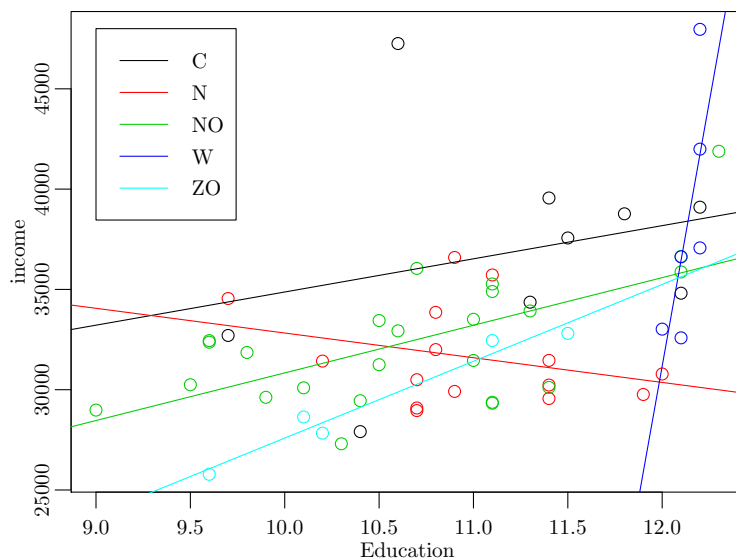
6.6 Two-way ANOVA

In het geval van twee discrete verklarende veranderlijken zijn vijf verschillende modellen mogelijk en spreekt men van two-way ANOVA:

- Het null model dat één globaal gemiddelde berekent;
- Twee one-way ANOVA modellen die telkens groepsgemiddelden schatten volgens één van beide factoren;
- Het additieve model, dat voor beide veranderlijken groepseffecten berekent, die onafhankelijk van elkaar worden toegepast;
- Het multiplicatieve model of *full model*, dat een interactieterm toevoegt, wat er op neer komt dat er voor elke cel een afzonderlijk gemiddelde wordt berekend.

Deze modellen zijn genest zoals te zien in figuur 6.4. De residuen moeten zich gedragen als een zuiver lukrake steekproef uit een normaalverdeling met constant gemiddelde en variantie. In een multiplicatief model wordt per cel het gemiddelde geschat en is dus net als bij one-way ANOVA automatisch voldaan aan de voorwaarde dat de residuen gemiddelde nul moeten hebben. In een additief model is dat niet zo en moet dit bijkomend worden nagegaan.

Voorbeeld 16. Verschilt het gemiddelde inkomen in een stad naargelang regio en grootte van die stad? Ten behoeve van dit voorbeeld wordt eerst de continue veranderlijke `pop` gediscretiseerd. Dit is een onlogische stap die enkel wordt gezet wegens gebrek aan geschikte categorische veranderlijken in de dataset `pollutie`. In de praktijk zou een ANCOVA-model (zie sectie 6.5) worden gemaakt met `region` en `pop`. De veranderlijke `size` wordt bepaald als volgt:



Figuur 6.3: Voorstelling van het ANCOVA-model

small – minder dan 500000 inwoners;

average – tussen 500000 en 1000000 inwoners;

large – tussen 1000000 en 2000000 inwoners;

huge – meer dan 2000000 inwoners.

```

324 > size = cut(pop,c(0,500000,1000000,2000000,Inf),
              c('small','average','large','huge'))
326 > income.1 = aov(income~1); summary(income.1)
      Df Sum Sq Mean Sq F value Pr(>F)
328 Residuals  58 1.16e+09 20008579
      > income.R = aov(income~regio); summary(income.R)
      Df Sum Sq Mean Sq F value Pr(>F)
330 regio      4 0.39e+09 98217221  6.909 0.000144 ***
332 Residuals  54 0.77e+09 14215346
      > income.S = aov(income~size); summary(income.S)
      Df Sum Sq Mean Sq F value Pr(>F)
334 size       3 0.27e+09 91157170  5.652 0.00189 **
336 Residuals  55 0.89e+09 16127747
      > income.RS = aov(income~regio+size); summary(income.RS)
      Df Sum Sq Mean Sq F value Pr(>F)
338 regio      4 0.39e+09 98217221  8.116 3.83e-05 ***
340 size       3 0.15e+09 50144183  4.144 0.0105 *
      Residuals  51 0.62e+09 12101885
342 > income.SR = aov(income~size+regio); summary(income.SR)
      Df Sum Sq Mean Sq F value Pr(>F)
344 size       3 0.27e+09 91157170  7.532 0.000289 ***
      regio      4 0.27e+09 67457481  5.574 0.000843 ***
346 Residuals  51 0.62e+09 12101885

```

```

> income.full = aov(income~size*regio); summary(income.full)
348 Df      Sum Sq   Mean Sq F value    Pr(>F)
    size          3  0.27e+09  91157170     7.435 0.000436 ***
350 regio          4  0.27e+09  67457481     5.502 0.001220 **
    size:regio    10  0.11e+09  11448105     0.934 0.513230
352 Residuals     41  0.50e+09  12261344
> interaction.plot(regio2,size,income)           # zie Figuur 6.5

```

Uit bovenstaande (zeer redundante) code valt het volgende af te leiden

- Beide one-way modellen verklaren een significant deel van de respons, zowel de grootte als de ligging van een stad zijn bepalend voor het inkomen.
- Vergelijking van deze modellen met het additieve model model leert dat beide effecten significant zijn, ook al wordt er al rekening gehouden met het andere.
- Het multiplicatieve model is niet significant, het effect van **size** is niet regio-afhankelijk. Grafisch blijkt dit uit figuur 6.5 waar de curves voor verschillende waarden van **size** min of meer parallel lopen (de afwijkingen doen zich voor in kleine groepen en wegen blijkbaar niet door op het resultaat).

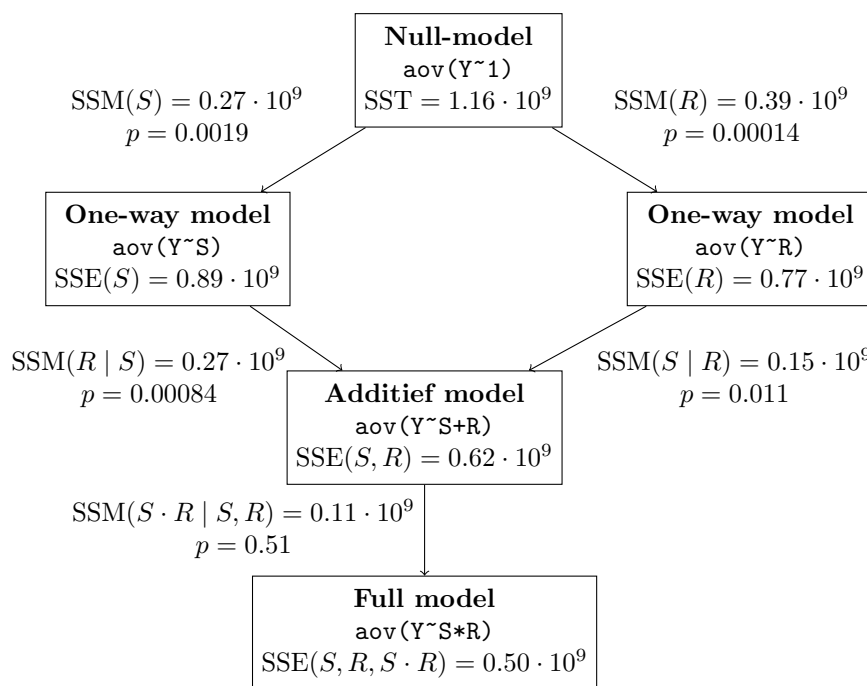
Er lijkt wat heteroscedasticiteit in de residuplot van het additieve model aanwezig, dit two-way ANOVA is niet geschikt voor het maken van betrouwbare intervalschattingen.

Merk op dat in de praktijk doorgaans niet al deze bevindingen relevant zijn, maar dat er naargelang de onderzoeksvraag specifiek naar één of enkele waarden wordt gekeken.

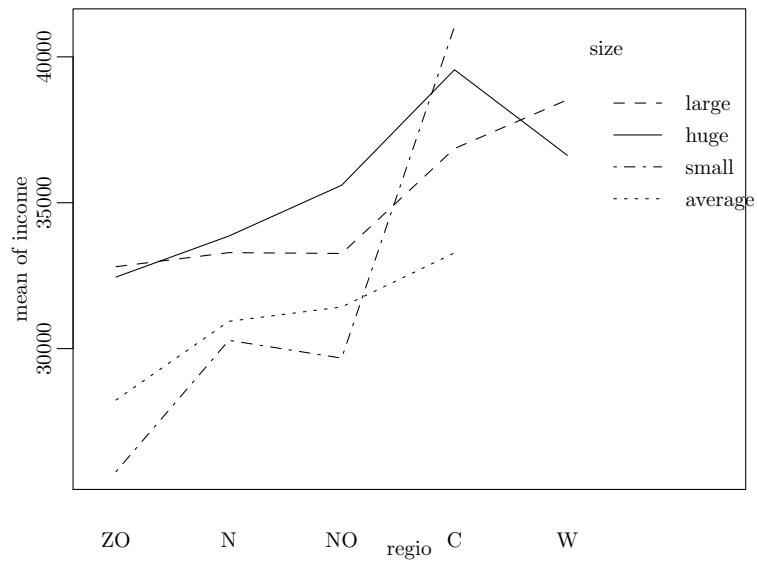
- Is het de bedoeling de respons zo goed mogelijk te verklaren, dan gaat men op zoek naar het eenvoudigste model waaraan geen significante termen meer kunnen worden toegevoegd. In dit geval is het additief model aangewezen want alle termen geven daarin een significante bijdrage, terwijl de interactietermen dat niet doen.
- Is de vraag of de ligging van een stad, los van de grootte-orde, invloed heeft op het inkomen, dan zijn vooral $SSM(R | S)$ en $SSM(S \cdot R | S, R)$ relevant. Hier blijkt dan dat er inderdaad nog een significant regionaal effect is bovenop een schaaffect, maar dat er geen aanwijzingen zijn dat het schaaffect regio-afhankelijk is.

6.7 Goodness-of-fit

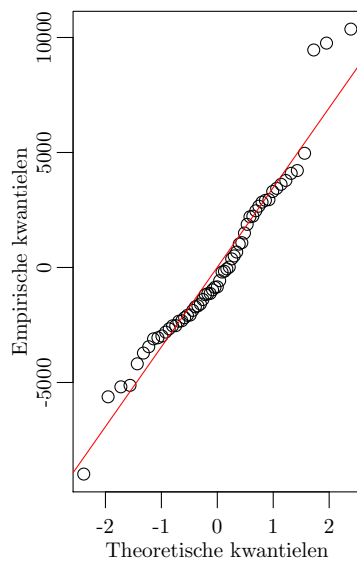
Veel datasets bevatten metingen bij gecontroleerde omstandigheden, waardoor de gemeten veranderlijken in wezen wel continu zijn (bijvoorbeeld temperatuur) maar de verschillende waarden in de dataset eerder discreet (bijvoorbeeld enkel metingen bij $-10^{\circ}C$, $0^{\circ}C$, $20^{\circ}C$ of $100^{\circ}C$). De vraag is dan of deze veranderlijke als continu kan worden beschouwd en gebruikt in een regressiemodel, dan wel of ze beter als categorisch geldt om er een one-way ANOVA mee uit te voeren. Beide benaderingen zijn mogelijk en leiden telkens tot andere residuen zoals voorgesteld in figuur 6.6. Zodoende is een partiële F -test volgens formule 6.1



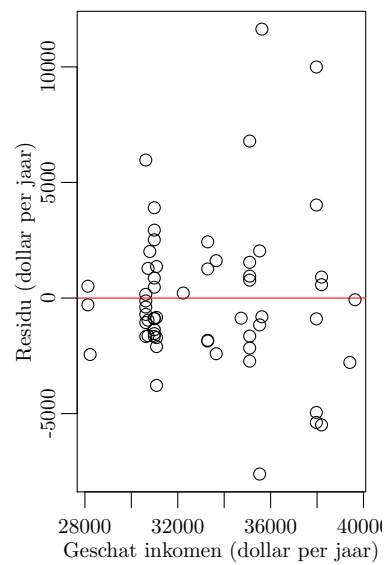
Figuur 6.4: Mogelijke ANOVA modellen bij twee discrete regressoren R en S met continue respons Y, telkens met som van kwadraten van de residuen, extra som van kwadraten en p -waarde van de bijhorende partiële F -test.



(a) Interactieplot van **income** in functie van **regio** en **size**



(b) Normale kwantielplot



(c) Residuplot

Figuur 6.5: Voorstelling van en modelveronderstellingen bij het two-way ANOVA model

mogelijk. Is het verschil $SSE_1 - SSE_2$ niet significant, dan beschrijft de lineaire vergelijking de groepsgemiddelden niet significant minder goed dan afzonderlijke schattingen in elke groep. Het regressiemodel levert dan een eenvoudiger model dat in essentie even veel verklaart. In het andere geval levert het gebruik van afzonderlijke schattingen wél extra informatie en is het gebruik van een lineair regressiemodel te zeer vereenvoudigend.

Voorbeeld 17. Is het verband tussen het aantal jaar opleiding in een stad en het inkomen lineair, of vereenvoudigt dat het one-way ANOVA model te zeer? Ten behoeve van dit voorbeeld wordt eerst de continue veranderlijke **Education** gediscrètiseerd. Dit is een onlogische stap die enkel wordt gezet wegens gebrek aan geschikte veranderlijken in de dataset **pollutie**. In de praktijk is er bij **Education** geen discussie: een regressiemodel is de aangewezen methode, one-way ANOVA zou leiden tot haast even veel categorieën als meetwaarden. De numerieke veranderlijke **Enum** en categorische equivalent **Ecat** worden berekend als volgt, in dit voorbeeld wordt verder gedaan alsof de bekomen waarden 10, 11 en 12 exacte meetwaarden zijn.

10 – minder dan 10.5 jaar scholing;

11 – tussen 10.5 en 11.5 jaar scholing;

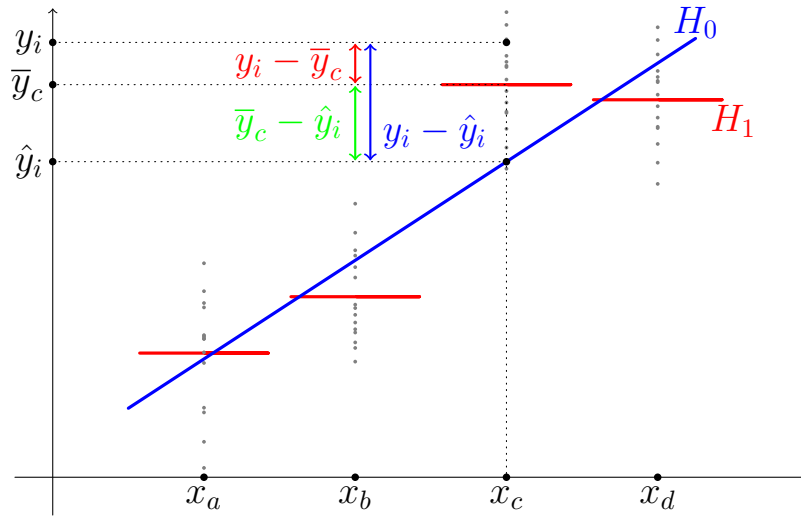
12 – meer dan 11.5 jaar scholing.

```

354 > Ecat = cut(Education, c(0, 10.5, 11.5, Inf), 10:12)
> income.Ecat = aov(income ~ Ecat)
356 > summary(income.Ecat)
      Df      Sum Sq      Mean Sq F value    Pr(>F)
358 Ecat      2 342248655 171124327  11.71 5.63e-05 ***
Residuals    56 818248927 14611588
360 > model.tables(income.Ecat, type='means')
Tables of means
362 10      11      12
30329 33287 36920
364 > Enum = as.numeric(as.character(Ecat))
> income.Enum = lm(income ~ Enum)
366 > summary(income.Enum)
Coefficients:
368 Estimate Std. Error t value Pr(>|t|)
(Intercept) -2571.0      7377.3  -0.349    0.729
370 Enum      3276.3      673.3    4.866 9.38e-06 ***
> predict(income.Enum, data.frame(Enum=c(10, 11, 12)))
372 1      2      3
30192 33469 36745
374 > anova(income.Enum, income.Ecat)
Analysis of Variance Table
376 Model 1: income ~ Enum
Model 2: income ~ Ecat
378 Res.Df      RSS Df Sum of Sq      F Pr(>F)
1      57 819905961
380 2      56 818248927  1 1657034 0.1134 0.7376

```

Het one-way ANOVA model vertelt dat er significante ($p = 6 \cdot 10^{-5}$) verschillen zijn tussen de gemiddelde inkomens binnen de groepen bepaald door **Ecat**. De voorspellingen zijn terug te vinden in onderstaande tabel.



Figuur 6.6: Vergelijking van regressie- en one-way ANOVA-model.

Wordt de veranderlijke beschouwd als continu en gebruikt om een regressiemodel te schatten, dan levert dat een significant model ($p = 9 \cdot 10^{-6}$) met vergelijking

$$\text{income} = -2571 + 3276 \cdot \text{Enum}.$$

Voor de waarden 10, 11 en 12 uit de dataset leidt dit tot voorspellingen die licht verschillen van deze uit het one-way ANOVA-model:

	10	11	12	SSE	p
income.Ecat	30329	33287	36920	$0.820 \cdot 10^9$	3
income.Enum	30192	33469	36745	$0.818 \cdot 10^9$	2

Dit laatste model is eenvoudiger omdat het bepaald wordt door slechts twee parameters (intercept en slope), maar het verklaart dus ook minder. Uit de partiële F -test ($F = 0.11$ volgens formule 6.1) blijkt dat dit verschil niet significant ($p = 74\%$) is. Dit levert allerm minst een bewijs dat het verband tussen beide grootheden lineair is, maar het eenvoudigere lineaire model verklaart virtueel even veel als en is dus te verkiezen boven het one-way ANOVA-model.

6.8 ANOVA versus regressie

Uit het voorgaande blijkt dat ANOVA en regressie, voor zo ver niet identiek, sterk verweven zijn. Terwijl het eigenlijk weinig zin heeft dit als afzonderlijke concepten te gebruiken, gebeurt dat in de praktijk toch.

Het doel van regressie is vaak, maar niet uitsluitend, om de respons zo goed mogelijk te verklaren en om voorspellingen te maken, idealiter resulteert een regressiemodel in een kwantitatief model voor de respons. De term ANOVA wordt gebruikt voor alle situaties waarin naar kwadratensommen wordt gekeken, dus ook in regressiemodellen zonder categorische veranderlijken. De specifieke modelnamen ANCOVA, one- en two-way ANOVA worden gebruikt in specifieke

situaties (meestal labo-experimenten) waar het aantal veranderlijken strikt onder controle wordt gehouden en waar het niet per se de bedoeling is de respons te voorspellen, maar enkel om aan te tonen dat de respons afhangt van (de interactie van) de voorspellers: ANOVA-modellen beperken zich vaak tot kwalitatieve uitspraken.

Aangezien de rekentechnieken, hypothesetesten en modelveronderstellingen bij al deze modellen identiek zijn, kunnen regressie- en ANOVA-modellen beide voor zowel kwantitatieve als kwalitatieve uitspraken gebruikt worden: de eerste vraag die moet worden gesteld voor men op zoek gaat naar een geschikt model, is welke onderzoeksvraag men beantwoord wil zien.

Tabel 6.1: Stappenplan bij two-way ANOVA

1. Bekijk eerst de interactie-term. Indien significant:
 - Factoren beïnvloeden elkaar: gemiddelde per cel
 - Tracht het effect af te lezen van een interactieplot
 - (Significantie van) individuele effecten niet meer relevant
2. Interactie niet significant, bekijk het additief model:
 - Is tweede term significant: significante bijdrage bovenop de eerste
 - Groepswaarde bepalen uit effect per cel en kolom
 - Volgorde termen belangrijk voor individuele testen
 - Globale significantie wel symmetrisch (vaak niet relevant)
 - Eenvoudige interpretatie van coëfficiënten:
twee effecten afzonderlijk te berekenen
3. Tweede term in additief model niet significant:
 - Bekijk de afzonderlijke one-way modellen
 - Indien beide significant: confounding factor

Tabel 6.2: Variantie-analyse in R

	<code>FIT = aov(CTU~CAT1*CATT2)</code>	<code># ANOVA-model opstellen</code>
2	<code>model.tables(FIT,</code>	<code># groepsgemiddelden/-effecten</code>
	<code>type="means")</code>	<code># "effects"</code>
4	<code>class(FIT)</code>	<code># "aov" "lm"</code>
	<code>pairwise.t.test(X,CAT1)</code>	<code># post-hoc testen</code>
6	<code>p.adjust(PVALS,</code>	<code># p-waarden corrigeren</code>
	<code>method="bonferroni")</code>	<code># "none", ...</code>
8	<code>TukeyHSD(FIT)</code>	<code># Tukey post-hoc testen</code>
	<code>interaction.plot()</code>	<code># interactieplot bij two-way ANOVA</code>

Bibliografie

- [1] J. Beirlant, M. Dierckx, and M. Hubert. *Statistiek en Wetenschap*. Acco, Leuven, 2009.
- [2] J.J. Faraway. *Practical regression and Anova using R*. 2002.
- [3] The R Development Core Team. *The R Reference Index*. 2010.
- [4] W.N. Venables and D.M. Smith. *An Introduction to R*. 2010.

KU Leuven Kulak
Wetenschap & Technologie
Etienne Sabbelaan 53, 8500 Kortrijk
Tel. +32 56 24 60 20
Fax +32 56 24 69 99

