4 – steekproefonderzoek

De normale verdeling

In R een functie plotten kan je niet rechtstreeks. R kan aan de hand van een lijst met x- en y-coördinaten wel punten tekenen en lijnen ertussen. Die moeten we dus eerst berekenen.

Plot van de standaardnormale verdeling

Specifiek kan je voor de standaardnormale verdeling $Z \sim Nor(\mu=0, \sigma=1)$ als volgt te werk gaan:

```
# Maak een lijst van 200 getallen tussen het interval [-4, 4]
# die gelijk verdeeld zijn.
x <- seq(from = -4,
         to = 4,
         length.out = 200)
# Bereken voor elke x-waarde het punt op de Gauss-curve
y <- dnorm(x)

# Maak hier een lijngrafiek van
plot(x, y, type = 'l')
```

Plot van een normale verdeling

We kunnen deze werkwijze veralgemenen voor een normale verdeling. Als voorbeeld nemen we $X \sim Nor(\mu=5, \sigma=1.5)$. Alle “interessante” punten op de Gausscurve liggen op max. 4 standaardafwijkingen links of rechts van het gemiddelde. We gaan onze x-waarden dan ook zo berekenen. De `dnorm`-functie laat toe om gemiddelde en standaardafwijking ook op te geven.

```
m <- 5    # gemiddelde
s <- 1.5  # standaardafwijking
# Bepaal de "interessante" x-waarden voor de plot
x <- seq(from = m-4*s,
         to = m+4*s,
         length.out = 200)
# Bereken de dichtheidsfunctie
y <- dnorm(x, m, s)
```

```
# Maak hier een lijngrafiek van
plot(x, y, type = 'l')
```

De vorm van deze grafiek is volledig dezelfde als die van de standaardnormaalverdeling, enkel de schaal (zowel op de x- als de y-as) is anders.

Histogram met dichtheidsfunctie

Het volgende voorbeeldje toont hoe je een histogram kan tekenen van normaal verdeelde data en de plot van de “theoretische” dichtheidsfunctie.

```
# Genereer 10000 willekeurige normaal verdeelde getallen:
n <- 10000
observaties <- rnorm(n, m, s)

# Teken een histogram (zonder titel)
histogram <- hist(observaties, main="")
```

Als je op bovenstaand histogram een Gauss-curve zou plotten, dan zal je niet veel zien. De schaal op de y-as voor de Gauss-curve is immers veel kleiner dan die van het histogram (zie bovenstaande plot van een normale verdeling).

We kunnen ook een histogram tekenen met densiteiten. In dit geval is de oppervlakte van een staaf gelijk aan de relatieve frequentie van die klasse. (De densiteit of dichtheid is de relatieve frequentie gedeeld door de breedte van de klasse.) De som van alle oppervlaktes van alle balken is één. Dit komt overeen met de oppervlakte onder de Gauss-curve die is ook gelijk aan één. Het histogram en de Gausscurve kunnen dan dezelfde y-as gebruiken.

```
# Genereer 10000 willekeurige normaal verdeelde getallen:
n <- 10000
observations <- rnorm(n, m, s)

# Teken een histogram van de relatieve ipv de absolute
# frequenties.
# Speel met de waarde van "breaks" om een "fijner" of "grover"
# histogram te bekomen.
hist(observations, freq = FALSE, breaks = 50)

# Bereken y-waarden voor de dichtheidsfunctie
y <- dnorm(x, m, s)

# Voeg deze functie toe aan het histogram
```

```
lines(x, y, col = 'blue')
```

De oproep van de functie `lines` maakt dat de nieuwe plot bovenop het reeds bestaande histogram wordt getekend. Als je hier de `plot`-functie gebruikt, dan wordt de tekening gewist en begin je opnieuw.

Kansverdeling in de normale verdeling

Stel, Superman heeft een reactiesnelheid die normaal verdeeld is met gemiddelde 5 ms en standaardafwijking 1.5 ms.

```
m <- 5
s <- 1.5
```

Wat is de kans dat zijn reactiesnelheid groter is dan 6.5 ms? Wiskundige notatie: $P(X>6.5)$P(X>6.5)

```
# Bereken eerst de z-score
x <- 6.5
z <- (x - m) / s
z
## [1] 1
```

Met de `pnorm` kunnen we enkel de linkerstaartkans $P(X<6.5)$P(X<6.5) of $P(Z<1)$P(Z<1) berekenen. Er wordt gevraagd naar de kans dat de reactiesnelheid *groter* is dan 6.5 ms, dus we berekenen $1-P(X<6.5)$1−P(X<6.5) of $1-P(Z<1)$1−P(Z<1):

```
1-pnorm(z)
## [1] 0.1586553
1-pnorm(x, m, s)
## [1] 0.1586553
```

Grafische voorstelling van deze situatie:

```
# interval van de plot (x-waarden)
x_interval <- seq(m - 4 * s, m + 4 * s, length=200)
# punten op de Gauss-curve
norm_dist <- dnorm(x_interval, m, s)
plot(x_interval, norm_dist,
     type = 'l', xlab = '', ylab = '')

# Het gebied links van x inkleuren
i <- x_interval <= x
```

```
polygon(
  c(x_interval[i], x,              x),
  c(norm_dist[i],  dnorm(x, m, s), 0),
  col = 'lightgreen')
text(x, .01, x)

# Toon het gemiddelde ahv een rode vertikale lijn
abline(v = m, col='red')
text(m, .01, m)
```

Andere voorbeelden van kansberekening:

1. Hoe groot is de kans dat de reactiesnelheid van Superman minder dan 4 ms is?

```
pnorm(4, m, s)
```
```
## [1] 0.2524925
```

2. Hoe groot is de kans dat hij in `m` eerdan 7 ms reageert?

```
1 - pnorm(7, m, s)
```
```
## [1] 0.09121122
```

3. Hoe groot is de kans dat Superman in minder dan 3 ms reageert?

```
pnorm(3, m, s)
```
```
## [1] 0.09121122
```

4. Hoe groot is de kans dat hij reageert tussen de 2 en de 6,5 ms?

```
pnorm(6.5, m, s) - pnorm(2, m, s)
```
```
## [1] 0.8185946
```

5. Onder welke tijd ligt 80% van zijn reactiesnelheid?

```
qnorm(.8, m, s)
```
```
## [1] 6.262432
```

Betrouwbaarheidsinterval

Een betrouwbaarheidsinterval is een schatting aan de hand van een gebied waarbinnen je met een gekozen betrouwbaarheid kan veronderstellen dat het onbekende populatiegemiddelde erbinnen zal liggen.

Stel, we hebben n = 100 metingen gedaan van de reactiesnelheid van Superman en we bekomen een steekproefgemiddelde van 5.2 ms. We veronderstellen dat we de werkelijke standaardafwijking van zijn reactiesnelheid kennen en dat die 1.5 ms is.

Om een betrouwbaarheidsinterval te bepalen gaan we als volgt te werk:

1. We nemen als initiële schatting het steekproefgemiddelde en kiezen een zekerheidsniveau, bv. $1-\alpha=0.95$ (of 95%).
2. We zoeken vervolgens de z-score waartussen 95% van alle waarden liggen bij een standaardnormale verdeling.
3. Die gebruiken we om de waarden links en rechts van het steekproefgemiddelde te bepalen waartussen we verwachten dat 95% van de waarden terecht komen voor de kansverdeling die we uit de centrale limietstelling halen.

```
# Stap 1
m <- 5.2       # steekproefgemiddelde
s <- 1.5       # standaardafwijking van de populatie
n <- 100       # steekproefgrootte
alpha <- 0.05 # 1 - alpha is het zekerheidsniveau
# Stap 2
z <- qnorm(1-alpha/2) # waarom delen door 2?
z
```

```
## [1] 1.959964
```

```
# Stap 3: het betrouwbaarheidsinterval
low  <- m - z * s / sqrt(n)
high <- m + z * s / sqrt(n)
c(low, high)
```

```
## [1] 4.906005 5.493995
```

We stellen dus met een betrouwbaarheidsniveau van 95% dat de reactiesnelheid van de superhelden ligt tussen de 4.91 en 5.49 ms ligt.

Betrouwbaarheidsinterval bij kleine steekproeven

Wanneer we een kleine steekproef hebben (kleiner dan 30) valt de veronderstelling die we in de centrale limietstelling gedaan hebben weg. We kunnen in dat geval ook de normale verdeling niet gebruiken.

Via de zgn. Student-t-verdeling is er echter toch een manier om een betrouwbaarheidsinterval te construeren. Deze verdeling lijkt op de normale verdeling in die zin dat ze ook op een Gauss-curve lijkt. De Student-t-verdeling houdt echter rekening met de steekproefgrootte n en moet je ook mee opgeven. De dichtheidsfunctie krijgt dan een extra parameter die het aantal *vrijheidsgraden* genoemd wordt (Eng. *degrees of freedom*, afgekort `df`) en gelijk is aan $n-1$.

Hoe kleiner het aantal vrijheidsgraden, hoe “platter” de curve en hoe breder de bekomen betrouwbaarheidsintervallen zullen zijn. Dit modelleert de grotere onzekerheid die we krijgen omwille van de kleine steekproef. Hoe groter n, hoe dichter de curve die van de normaalverdeling zal benaderen.
In de grafiek hieronder vind je de dichtheidsfunctie voor de Student-t-verdeling voor verschillende vrijheidsgraden:

```
# Bron: https://www.statmethods.net/advgraphs/probability.html
x <- seq(-4, 4, length=100) # x-waarden
std_norm_dist <- dnorm(x)   # standaardnormaalverdeling, ter vgl
degf <- c(1, 3, 8, 30)      # te plotten vrijheidsgraden

# afwerking van de grafiek (kleur, legende)
colors <- c("red", "blue", "darkgreen", "gold", "black")
labels <- c("df=1", "df=3", "df=8", "df=30", "normaal")

# plot van de standaardnormaalverdeling
plot(x, std_norm_dist,
     type="l", lty=2,
     xlab="x-waarde", ylab="dichtheid",
     main="Vergelijking van Student-t verdelingen")

# plot van de vier Student-t verdelingen
for (i in 1:4){
  lines(x, dt(x,degf[i]), lwd=2, col=colors[i])
}

legend("topright", inset=.05, title="Verdelingen",
       labels, lwd=2, lty=c(1, 1, 1, 1, 2), col=colors,
       cex = .5)
```

Om dit te illustreren nemen we hetzelfde voorbeeld van hierboven, maar veronderstellen dat de steekproefgrootte slechts 15 was.

```
# Stap 1
m <- 5.2      # steekproefgemiddelde
s <- 1.5      # standaardafwijking van de populatie
n <- 15       # steekproefgrootte
alpha <- 0.05 # 1 - alpha is het zekerheidsniveau
# Stap 2, maar nu gebruiken we de Student-t-verdeling!
```

```
t <- qt(1-alpha/2, df = n - 1)
t
```

```
## [1] 2.144787
```

```
# Stap 3: het betrouwbaarheidsinterval
low  <- m - t * s / sqrt(n)
high <- m + t * s / sqrt(n)
c(low, high)
```

```
## [1] 4.369328 6.030672
```

We stellen dus met een betrouwbaarheidsniveau van 95% dat de reactiesnelheid van de superhelden ligt tussen de 4.37 en 6.03 ms ligt.

Dit interval is een stuk breder dan wat we verkregen bij een grotere steekproef. We zijn dus minder zeker van de positie van het populatiegemiddelde.

Betrouwbaarheidsinterval bij fracties

Sommige variabele hebben slechts twee mogelijke waarden, bv. ja/nee, waar/vals, gelukt/mislukt, succes/falen, 1/0, enz. Aan de hand van de steekproef willen we schatten wat in werkelijkheid de verhouding is tussen deze twee uitkomsten over de gehele populatie.

Stel dat we willen weten welk percentage van de superhelden (op eigen kracht) kan vliegen. Er werden 100 superhelden ondervraagd, 6 daarvan konden demonstreren dat ze inderdaad kunnen vliegen. Construeer een 95%-betrouwbaarheidsinterval voor het verwachte percentage van vliegende superhelden in de gehele pupulatie.

```
# Stap 1.
n <- 100   # steekproefgrootte
k <- 6     # aantal "successen" in de steekproef
a <- 0.05  # betrouwbaarheidsniveau 95%

p <- k / n # schatting voor het percentage successen
q <- 1 - p # schatting voor het percentage falingen
c(p, q)    # toon de waarden van p en q
```

```
## [1] 0.06 0.94
```

```
# Stap 2. We gebruiken opnieuw de normale verdeling
z <- qnorm(1-a/2)

# Stap 3.
low <- p - z * sqrt(p*q/n)
high <- p + z * sqrt(p*q/n)
c(low, high)
```

```
## [1] 0.01345343 0.10654657
```

We stellen dus met een betrouwbaarheidsniveau van 95% dat tussen de 1.3% en 10.6% van de superhelden kan vliegen.