R-anvisningar till *Grundläggande statistik*

2021-10-04

Innehåll

Kapitel 1

Introduktion

Detta dokument är en kort introduktion till R för en kurs i grundläggande statistik.

Kapitel 2

Installation

2.1 Inledning

För att köra R-kod på sin dator krävs en installation av programspråket R. För att effektivt arbeta i R används ofta en utvecklingsmiljö (ett tilläggsprogram som på flera sätt förenklar arbetet) och här ges anvisningar till den vanligaste utvecklingsmiljön för R, som är RStudio. För att komma ingång måste man alltså installera R och RStudio.

2.2 Installation av R

Programspråket R kan laddas ner från https://www.r-project.org/ med följande steg:

1. Klicka på *CRAN* längst upp till vänster.
2. Klicka på den översta länken under 0-Cloud.
3. Välj en nedladdning beroende på operativsystem.
4. För Windows, välj *base*. För macOS, välj den senaste tillgängliga versionen.
5. Installera R från den nedladdade filen. Installation sker som för andra nedladdade program.

2.3 Installation av RStudio

RStudio kan laddas ner från https://www.rstudio.com/ med följande steg:

1. Klicka på *Download* uppe till höger.
2. Scrolla nedåt och välj *Download* under *RStudio Desktop*.
3. Klicka på nedladdningsknappen.
4. Installera RStudio från den nedladdade filen. Installation sker som för andra nedladdade program.

2.4 Gränssnittet i RStudio

När man nu öppnar RStudio ser man att fönstret är uppdelat i fyra delar och att varje del består av en eller flera flikar. De viktigaste är i nuläget

- *Console* där kod körs och resultat skrivs ut,
- *Environment* där man ser skapade objekt,
- *History* där man ser tidigare körd kod,
- *Plots* där man ser skapade grafer, och
- *Help* där man ser hjälpsidor för funktioner.

Ofta skriver man inte sin kod direkt i konsollen, utan i ett separat *skript* - en vanlig textfil som innehåller den kod man vill köra. Genom att organisera sin kod i ett skript kan man lätt strukturera och dokumentera sitt arbete. I RStudio kan man öppna ett nytt skript genom att gå till *File > New File > R Script* eller genom att klicka *Ctrl + Shift + N*. Ett tomt skript öppnar sig då i det övre vänstra delfönstret. Om man skriver

```
a <- 5
```

i skriptet och trycker *Ctrl + Enter* bör man se att koden i skriptet körs i konsollen. Om man tittar i fliken *Environment* ska man också se att det nu skapats ett objekt *a*.

2.5 Paket i R

En av de stora styrkorna med R är att språket kan byggas ut av dess användare. De här tilläggen kan sedan samlas i paket (*packages*) och delas med andra. Rs officiella bibliotek för paket kallas för *CRAN* (*Comprehensive R Archive Network*) och består av mer än 18 000 uppladdade paket som innehåller allt från fritt tillgänglig data till avancerade statistiska modeller.

För att använda ett specifikt paket måste det först installeras. Om man vet namnet på paketet man vill installera kan man köra

```
install.packages("tidyverse")
```

I det här fallet installeras paketet `tidyverse`, vilket innehåller funktioner för hantering av data.

I RStudio kan man också installera paket från *Packages*-fliken.

Paket måste också laddas för varje ny session. Innan man kan använda innehållet i ett paket måste man därför köra

```
library(tidyverse)
```

Kapitel 3

Objekt och funktioner

Ett *objekt* i R är en namngiven informationsmängd. Objekt kan se ut på många olika sätt - under kursens gång används objekt som består av insamlad data (konstruerade som vektorer eller tabeller), objekt som är statistiska modeller, och flera andra former. I ett tidigare exempel fanns raden

```
a <- 5
```

Här skapas ett objekt med namnet `a` som innehåller informationen `5`. Ett lite mer komplicerat exempel på ett objekt ges av

```
b <- c(3, 1, 4, 1, 5, 9)
```

Här skapas ett objekt `b` som innehåller en *serie* numeriska värden (en *vektor*).

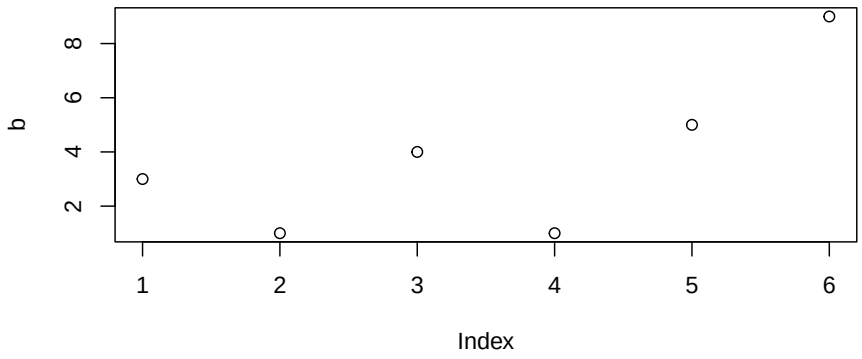
Objekt kan sedan manipuleras genom att tillämpa *funktioner*. En funktion tar någon ingående data och ger något utgående resultat. Funktioner anges genom att skriva funktionens namn följt av ingående data inom parenteser, och resultatet kan antingen skrivas ut i konsollen eller sparas som ett nytt objekt. En grundinstallation av R innehåller en mängd färdiga funktioner, t.ex.

```
sum(b)
```

```
## [1] 23
```

vilket ger summan av värdena i vektorn `b`,

```
plot(b)
```



som ger en simpel graf, och

```
sqrt(b)
```

```
## [1] 1.732051 1.000000 2.000000 1.000000 2.236068 3.000000
```

som beräknar kvadratroten för varje element i vektorn.

Vid konstruktionen av vektorn användes också en grundläggande funktion - funktionen `c` som tar en serie värden och skapar en sammanhängande vektor av värden.

Alla R-funktioner har en tillhörande hjälpfil som kan plockas fram genom att skriva frågetecken följt av funktionsnamnet, t.ex. `?sum`. Från hjälpfilen får man att `sum` tar numeriska vektorer som ingående värde och beräknar summan. Man kan styra funktionens beteende genom att sätta ett argument `na.rm` (vilket här styr hur funktionen hanterar saknade värden). Som illustration kan man titta på

```
b <- c(3, 1, 4, 1, 5, 9, NA)  # Lägger till ett saknat värde
sum(b)                        # na.rm = FALSE är grundinställning
```

```
## [1] NA
```

```
sum(b, na.rm = TRUE)          # na.rm sätts till TRUE
```

```
## [1] 23
```

Det första försöket `sum(b)` ger utfallet `NA`, men om man sätter `na.rm = TRUE` beräknas summan efter att det saknade värdet plockats bort. Notera också att skript kan kommenteras med `#`.

3.1 Sekvenser av funktioner

Ofta vill man genomföra flera operationer på ett objekt. Man behöver då genomföra en sekvens av funktionssteg. Säg till exempel att man har värdena

$$(-4, -2, -1, 1, 2, 4)$$

och vill ta absolutvärde (vilket gör negativa tal till motsvarande positiva tal) och sedan summera. Den typen av sekvenser kan genomföras på ett par olika sätt. Ett första sätt är att spara resultatet i varje steg och sedan använda utfallet i nästa steg:

```
c <- c(-4, -2, -1, 1, 2, 4)
c <- abs(c)
sum(c)
```

```
## [1] 14
```

Här skapas ett objekt `c` som innehåller en vektor där några tal är negativa. I nästa rad används `abs` för att skapa absolutvärden. Slutligen summeras absolutvärdena med `sum`. Notera för övrigt att det är möjligt att skapa ett objekt med namnet `c` trots att det redan är namnet på en funktion - R förstår ur sammanhanget vilket objekt som behövs.

Ett alternativ är att skriva en senare funktion *runt* en tidigare funktion. Det fungerar för att R utvärderar funktioner inifrån-ut. Med samma exempel som tidigare får man

```
sum(abs(c(-4, -2, -1, 1, 2, 4)))
```

```
## [1] 14
```

Den här typen av skrivning kan spara plats men blir snabbt svårläst.

Ett sista alternativ är att använda en så kallad *pipe* (namnet kommer från att en sekvens funktioner kallas en *pipeline*). En pipe skrivs `%>%` och tar utfallet av en funktion till vänster och sänder till en funktion till höger. Språkligt kan pipen utläsas *och sen*. Funktionen kan laddas genom att ladda paketet `tidyverse`. Med samma exempel som tidigare kan vi skriva

```
library(tidyverse)

c(-4, -2, -1, 1, 2, 4) %>%  # Skapa en datamängd och sen
  abs() %>%                 # ta absolutvärden, och sen
  sum()                     # beräkna summan.
```

```
## [1] 14
```

Kapitel 4

Dataimport

Det första praktiska steget i en statistisk analys är att importera data. I R kan det göras genom att direkt skriva in sin data och spara som ett nytt objekt, men ett bättre och vanligare sätt är att importera sin data från en extern fil eller databas. Om man arbetar med små datamängder har man ofta sin data i en excelfil.

Som exempel används här följande data från ett försök på purjolök.

Notera att det finns saknade värden i kolumnen N.

4.1 Direkt inskrivning av data

I ett tidigare exempel användes funktionen `c` för att skapa en vektor av data. En datatabell (en `data.frame` i R) skapas genom funktionen `data.frame` följt av namngivna vektorer. Exempeldata kan skrivas in genom föjande.

```
dat <- data.frame(Vecka = c(7, 7, 7, 7, 7, 7,
                            11, 11, 11, 11, 11, 11),
                  Behandling = c(0,0,0,1,1,1,0,0,0,1,1,1),
                  Vikt = c(232, 161, 148, 368, 218, 257,
                           1633, 2213, 972, 2560, 2430, 855),
                  N = c(2.63, 2.90, 2.99, 3.54, 3.30,
                        2.85, 1.53, 1.90, NA, 2.58, NA, NA))

dat
```

```
##    Vecka Behandling Vikt    N
## 1      7          0  232 2.63
## 2      7          0  161 2.90
## 3      7          0  148 2.99
## 4      7          1  368 3.54
```

Vecka	Behandling	Vikt	N
7	0	232	2.63
7	0	161	2.90
7	0	148	2.99
7	1	368	3.54
7	1	218	3.30
7	1	257	2.85
11	0	1633	1.53
11	0	2213	1.90
11	0	972	NA
11	1	2560	2.58
11	1	2430	NA
11	1	855	NA

```
## 5      7          1  218 3.30
## 6      7          1  257 2.85
## 7     11          0 1633 1.53
## 8     11          0 2213 1.90
## 9     11          0  972   NA
## 10    11          1 2560 2.58
## 11    11          1 2430   NA
## 12    11          1  855   NA
```

Radbrytningar och blanksteg är oviktiga i R, och används bara för läsbarhet här. Saknade värden skrivs in som `NA` för *not available*.

4.2 Import från en extern fil

Inskrivning av värden är ofta tidskrävande och kan lätt leda till misstag. Det är därför mycket vanligare att data läses in från en extern fil. Det finns en mängd funktioner för dataimport och det exakta valet av funktion beror på vilken typ av fil datan är sparad i. Det vanligaste filformatet för mindre datamängder är Excel. En excelfil kan läsas med `read_excel` från paketet `readxl`.

```
library(readxl)
dat <- read_excel("Data/Purjolök.xlsx")
dat
```

```
## # A tibble: 12 x 4
##    Vecka Behandling  Vikt     N
##    <dbl>      <dbl> <dbl> <dbl>
##  1     7          0   232  2.63
##  2     7          0   161  2.9
##  3     7          0   148  2.99
```

```
##  4     7          1   368  3.54
##  5     7          1   218  3.3
##  6     7          1   257  2.85
##  7    11          0  1633  1.53
##  8    11          0  2213  1.9
##  9    11          0   972 NA
## 10    11          1  2560  2.58
## 11    11          1  2430 NA
## 12    11          1   855 NA
```

För att identifiera en fil behövs filens namn och placering. När man arbetar i R finns ett *working directory*, en mapp på datorn som R-session för tillfället är kopplad till. Man kan se sitt *working directory* genom att titta längst upp i konsollfönstret eller genom att köra `getwd()`. Ett filnamn kan anges relativt sessionens *working directory*. Om man till exempel vill importera en fil som ligger i *working directory* kan man ange filens namn direkt. I exemplet ovan låg excelfilen i en mapp *Data* - filen måste därför anges som `Data/Purjolök.xlsx`.

Notera också att utskriften av den importerade datan inte ser likadan ut som utskriften av den inskrivna datan. Det beror på att `read_excel` importerat datan som en *tibble* - ett dataformat som har lite mer avancerade egenskaper är det ursprungliga dataformatet `data.frame`, till exempel ger en utskriven *tibble* information om tabellens storlek och kolumnernas datatyper.

Kapitel 5

Beskrivande statistik

R och dess tilläggspaket innehåller funktioner för att sammanfatta och illustrera en datamängd. Detta avsnitt behandlar funktioner för att filtrera ut intressanta observationer, välja ut intressanta variabler, beräkna sammanfattande mått (som medelvärde, median och standardavvikelse), och konstruera och tolka grafer.

5.1 Datamängd

Som exempel används data från *Gapminder* - en stiftelse med mål att sprida information om ekonomisk utveckling och hälsa, grundad av Hans Rosling (1948-2017). Datamängden finns tillgänglig i R-paketet `gapminder`.

```
install.packages("gapminder") # Behöver bara köras första gången
library(gapminder)
```

Efter att paketet laddas med `library(gapminder)` är datan tillgänglig under namnet `gapminder`. Man kan skriva ut de första raderna genom

```
gapminder
```

```
## # A tibble: 1,704 x 6
##    country     continent  year lifeExp      pop gdpPercap
##    <fct>       <fct>     <int>   <dbl>    <int>     <dbl>
##  1 Afghanistan Asia       1952    28.8  8425333      779.
##  2 Afghanistan Asia       1957    30.3  9240934      821.
##  3 Afghanistan Asia       1962    32.0 10267083      853.
##  4 Afghanistan Asia       1967    34.0 11537966      836.
##  5 Afghanistan Asia       1972    36.1 13079460      740.
##  6 Afghanistan Asia       1977    38.4 14880372      786.
##  7 Afghanistan Asia       1982    39.9 12881816      978.
```

```
##  8 Afghanistan Asia       1987    40.8 13867957      852.
##  9 Afghanistan Asia       1992    41.7 16317921      649.
## 10 Afghanistan Asia       1997    41.8 22227415      635.
## # ... with 1,694 more rows
```

Datan anger förväntad medellivslängd, populationsstorlek och bnp per capita, per land och år (vart femte år från 1952 till 2007).

5.2 Filtrering av rader och selektion av kolumner

En vanlig operation på en tabell är att göra ett urval - antingen ett urval av rader (t.ex. ett visst land eller år), vilket kallas *filtrering* eller ett urval av variabler (t.ex. år och population), vilket kallas *selektion*. Det finns flera olika sätt att göra ett urval i R. Det traditionella sättet är att använda index inom hakparenteser (t.ex. `gapminder[4, 2]` för fjärde raden, andra kolumnen) eller dollartecken för specifika kolumner (t.ex. `gapminder$pop` för befolkningskolumnen). Här fokuseras dock på hur det kan göras med funktionerna `filter` och `select` från paketet `tidyverse`.

För att filtrera på ett givet land kan använda pipe-funktionen från datan till en filter-funktion, t.ex.

```
gapminder %>%                  # Ta gapminder-datan och sen
  filter(country == "Sweden") # filtrera för specifikt land
```

```
## # A tibble: 12 x 6
##    country continent  year lifeExp     pop gdpPercap
##    <fct>   <fct>     <int>   <dbl>   <int>     <dbl>
##  1 Sweden  Europe     1952    71.9 7124673     8528.
##  2 Sweden  Europe     1957    72.5 7363802     9912.
##  3 Sweden  Europe     1962    73.4 7561588    12329.
##  4 Sweden  Europe     1967    74.2 7867931    15258.
##  5 Sweden  Europe     1972    74.7 8122293    17832.
##  6 Sweden  Europe     1977    75.4 8251648    18856.
##  7 Sweden  Europe     1982    76.4 8325260    20667.
##  8 Sweden  Europe     1987    77.2 8421403    23587.
##  9 Sweden  Europe     1992    78.2 8718867    23880.
## 10 Sweden  Europe     1997    79.4 8897619    25267.
## 11 Sweden  Europe     2002    80.0 8954175    29342.
## 12 Sweden  Europe     2007    80.9 9031088    33860.
```

Inom filter-funktionen anges ett logisk villkor `country == Sweden"` och utfallet är de rader där villkoret är sant. Notera de dubbla likhetstecknen - de måste användas för ett logisk villkor eftersom enkelt likhetstecken används för att skapa objekt och sätta funktionsargument. Om man vill välja flera länder kan man använda funktionen `%in%` på ett liknande sätt.

```
gapminder %>%
  filter(country %in% c("Sweden", "Denmark"))
```

```
## # A tibble: 24 x 6
##    country continent  year lifeExp     pop gdpPercap
##    <fct>   <fct>     <int>   <dbl>   <int>     <dbl>
##  1 Denmark Europe     1952    70.8 4334000     9692.
##  2 Denmark Europe     1957    71.8 4487831    11100.
##  3 Denmark Europe     1962    72.4 4646899    13583.
##  4 Denmark Europe     1967    73.0 4838800    15937.
##  5 Denmark Europe     1972    73.5 4991596    18866.
##  6 Denmark Europe     1977    74.7 5088419    20423.
##  7 Denmark Europe     1982    74.6 5117810    21688.
##  8 Denmark Europe     1987    74.8 5127024    25116.
##  9 Denmark Europe     1992    75.3 5171393    26407.
## 10 Denmark Europe     1997    76.1 5283663    29804.
## # ... with 14 more rows
```

och om man vill ha mer än ett villkor kan man rada dem i filter-funktionen eller ha flera filter-steg:

```
gapminder %>%
  filter(country %in% c("Sweden", "Denmark"),
         year == 1987)
```

```
## # A tibble: 2 x 6
##   country continent  year lifeExp     pop gdpPercap
##   <fct>   <fct>     <int>   <dbl>   <int>     <dbl>
## 1 Denmark Europe     1987    74.8 5127024    25116.
## 2 Sweden  Europe     1987    77.2 8421403    23587.
```

alternativt

```
gapminder %>%
  filter(country %in% c("Sweden", "Denmark")) %>%
  filter(year == 1987)
```

```
## # A tibble: 2 x 6
##   country continent  year lifeExp     pop gdpPercap
##   <fct>   <fct>     <int>   <dbl>   <int>     <dbl>
## 1 Denmark Europe     1987    74.8 5127024    25116.
## 2 Sweden  Europe     1987    77.2 8421403    23587.
```

Om man istället vill göra ett urval av kolumner kan man använda `select`. Som argument anges de kolumner man vill välja, t.ex.

```
gapminder %>%
  select(country, pop)
```

```
## # A tibble: 1,704 x 2
##    country          pop
##    <fct>          <int>
##  1 Afghanistan  8425333
##  2 Afghanistan  9240934
##  3 Afghanistan 10267083
##  4 Afghanistan 11537966
##  5 Afghanistan 13079460
##  6 Afghanistan 14880372
##  7 Afghanistan 12881816
##  8 Afghanistan 13867957
##  9 Afghanistan 16317921
## 10 Afghanistan 22227415
## # ... with 1,694 more rows
```

Som avslutning ges ett lite mer komplicerat exempel på ett urval av land, kontinent och befolkning för länder utanför Europa som 2002 hade en befolkning över 100 miljoner

```
gapminder %>%                      # Ta datan och sen
  filter(continent != "Europe",    # filtrera på kontinent ej lika med (!=) Europa,
         year == 2002,             # år lika med 2002,
         pop > 100000000) %>%      # befolkning över 100 mil, och sen
  select(country, continent, pop) # selektera på land, kontinent och befolkning
```

```
## # A tibble: 10 x 3
##    country       continent        pop
##    <fct>         <fct>          <int>
##  1 Bangladesh    Asia       135656790
##  2 Brazil        Americas   179914212
##  3 China         Asia      1280400000
##  4 India         Asia      1034172547
##  5 Indonesia     Asia       211060000
##  6 Japan         Asia       127065841
##  7 Mexico        Americas   102479927
##  8 Nigeria       Africa     119901274
##  9 Pakistan      Asia       153403524
## 10 United States Americas   287675526
```

5.3 Transformationer av variabler

Variabler kan omräknas och nya variabler kan skapas med `mutate`-funktionen. I gapminder-datan finns befolkning och bnp per capita, så det är naturligt att beräkna total bnp som produkten av de två variablerna genom multiplikation.

```
gapminder <- gapminder %>%
  mutate(gdptotal = gdpPercap * pop)
```

Den inledande delen med `gapminder <-` gör så att utfallet av beräkningen sparas i gapminder-datan. Vi kan skriva ut objektet och se resultatet av beräkningen:

```
gapminder
```

```
## # A tibble: 1,704 x 7
##    country     continent  year lifeExp      pop gdpPercap     gdptotal
##    <fct>       <fct>     <int>   <dbl>    <int>     <dbl>        <dbl>
##  1 Afghanistan Asia       1952    28.8  8425333      779.  6567086330.
##  2 Afghanistan Asia       1957    30.3  9240934      821.  7585448670.
##  3 Afghanistan Asia       1962    32.0 10267083      853.  8758855797.
##  4 Afghanistan Asia       1967    34.0 11537966      836.  9648014150.
##  5 Afghanistan Asia       1972    36.1 13079460      740.  9678553274.
##  6 Afghanistan Asia       1977    38.4 14880372      786. 11697659231.
##  7 Afghanistan Asia       1982    39.9 12881816      978. 12598563401.
##  8 Afghanistan Asia       1987    40.8 13867957      852. 11820990309.
##  9 Afghanistan Asia       1992    41.7 16317921      649. 10595901589.
## 10 Afghanistan Asia       1997    41.8 22227415      635. 14121995875.
## # ... with 1,694 more rows
```

5.4 Sammanfattande statistik

För att presentera insamlad data på ett tolkningsbart sätt används sammanfattande mått såsom summor, medelvärden, medianer och standardavvikelser. Den typen av beräkningar kan göras som ett nytt steg i en pipe med hjälp av funktionen `summarise`. Om man kombinerar `summarise` med funktionen `group_by` kan man dessutom summera efter en indelning given av en annan variabel. En beräkning av total befolkningsmängd per år kan till exempel ges av

```
gapminder %>%                              # Ta datan och sen
  group_by(year) %>%                       # gruppera efter år och sen
  summarise(Totalbefolkning = sum(pop) / 1e9) # summera per grupp
```

```
## # A tibble: 12 x 2
##     year Totalbefolkning
##    <int>           <dbl>
##  1  1952            2.41
##  2  1957            2.66
##  3  1962            2.90
##  4  1967            3.22
##  5  1972            3.58
##  6  1977            3.93
##  7  1982            4.29
##  8  1987            4.69
```

```
##  9  1992           5.11
## 10  1997           5.52
## 11  2002           5.89
## 12  2007           6.25
```

I det sista steget skapas en variabel *Totalbefolkning* som ges av summan av den ursprungliga variabeln *pop*.

Funktionerna `summarise_at` och `summarise_all` kan användas för att summera flera variabler i ett steg. Man kan också ange mer än en funktion, om man vill beräkna flera olika mått.

```
gapminder %>%
  filter(year == 2007) %>%
  group_by(continent) %>%
  summarise_at(c("lifeExp", "pop"), c(mean, sd))
```

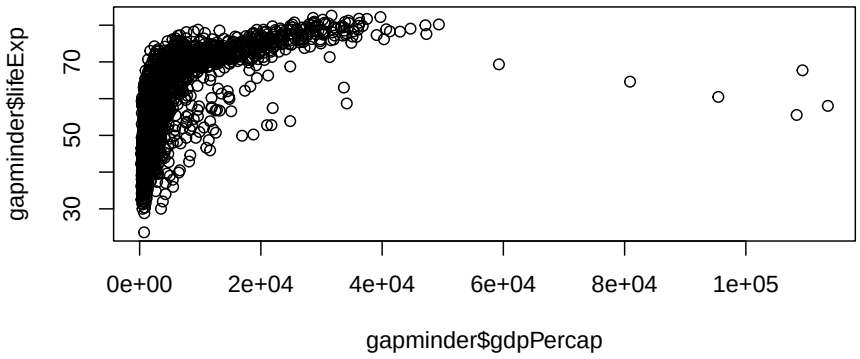
```
## # A tibble: 5 x 5
##   continent lifeExp_fn1    pop_fn1 lifeExp_fn2    pop_fn2
##   <fct>           <dbl>      <dbl>       <dbl>      <dbl>
## 1 Africa           54.8  17875763.       9.63   24917726.
## 2 Americas         73.6  35954847.       4.44   68833781.
## 3 Asia             70.7 115513752.       7.96  289673399.
## 4 Europe           77.6  19536618.       2.98   23624744.
## 5 Oceania          80.7  12274974.       0.729  11538855.
```

Kolumnerna för förväntat medelllivslängd och befolkning sammanfattas med medelvärde och standardavvikelse för observationer från 2007.

5.5 Grafer

R har en mängd grundläggande funktioner för grafer. Ett enkelt spridningsdiagram kan till exempel skapas med

```
plot(gapminder$gdpPercap, gapminder$lifeExp)
```



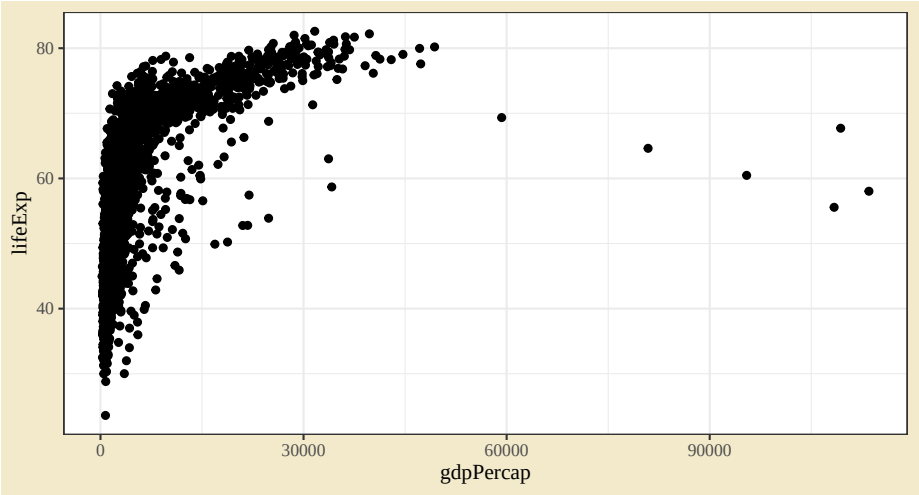
Tecknet `$` används här för att välja en kolumn i en tabell.

För mer avancerade grafer används dock ofta funktioner ur Rs paketbibliotek. Här illustreras det mest populära - `ggplot2`. I `ggplot2` byggs grafer upp med tre grundläggande byggstenar:

- *data*, informationen man vill visualisera,
- *aestethics*, en koppling mellan data och visuella element såsom grafens axlar, objekts storlek och färg,
- *geometries*, de geometriska former som visas i grafen.

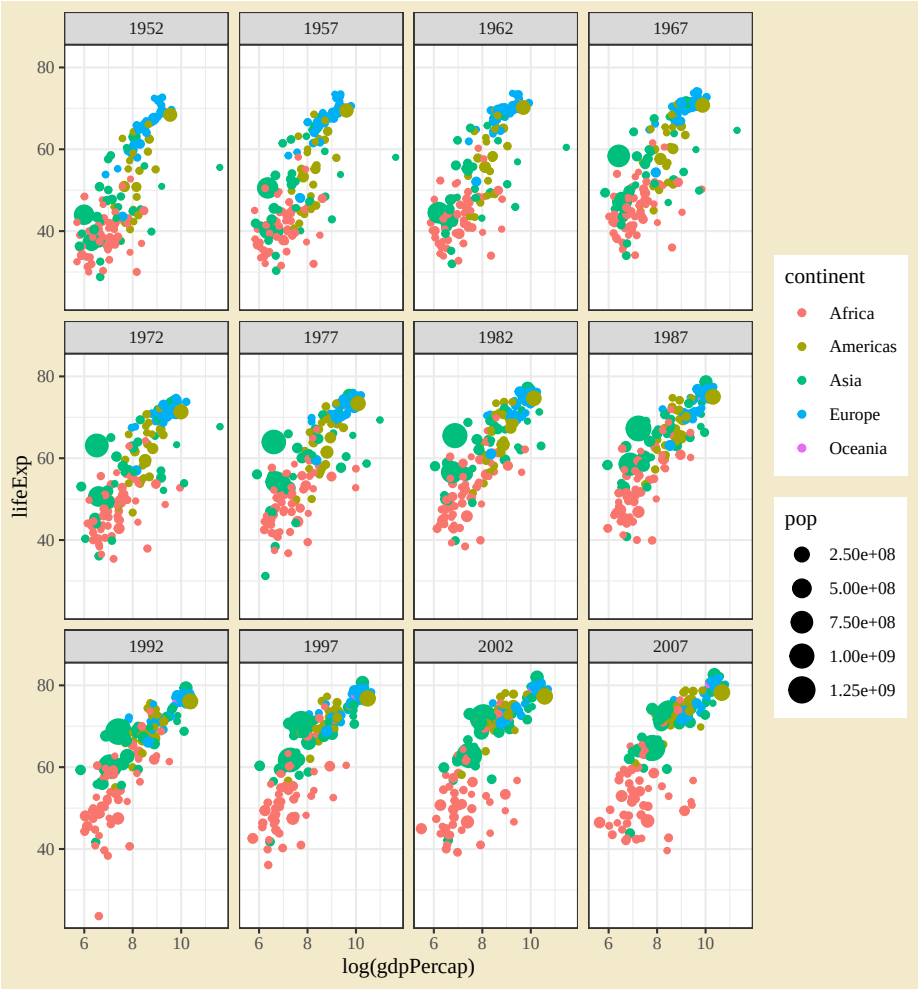
En graf skrivs med en startfunktion `ggplot` som anger namnet på datan och grafens *aestethics*, och därefter sätts geometriska element genom funktioner som börjar med `geom_`. Ett spridningsdiagram kan t.ex. skapas med `geom_point`.

```
ggplot(gapminder, aes(x = gdpPercap, y = lifeExp)) +
  geom_point()
```



Grafen kan byggas ut genom att sätta *aestethics* för färg och storlek. Man kan också dela en graf i småfönster med `facet_wrap` och styra grafens utseende genom att sätta ett tema såsom `theme_bw`.

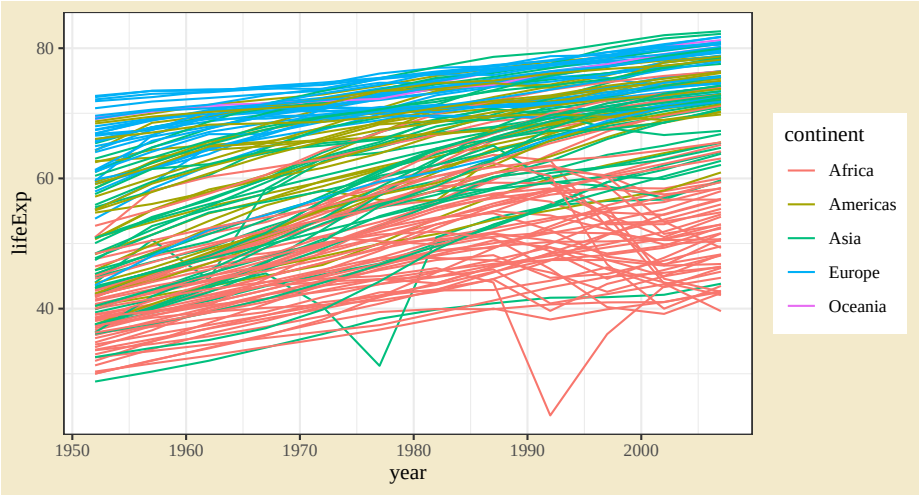
```
ggplot(gapminder, aes(x = log(gdpPercap), y = lifeExp, color = continent, size = pop))
  geom_point() +
  facet_wrap(~ year)
```



Här används dessutom log-transformerad bnp per capita för att få en jämnare fördelning i x-axeln.

Andra graftyper kan skapas med andra `geom_`-funktioner. För ett linjediagram används `geom_line`. De observationer som ska ge en specifik linje anges med `group` i `aes`-funktionen.

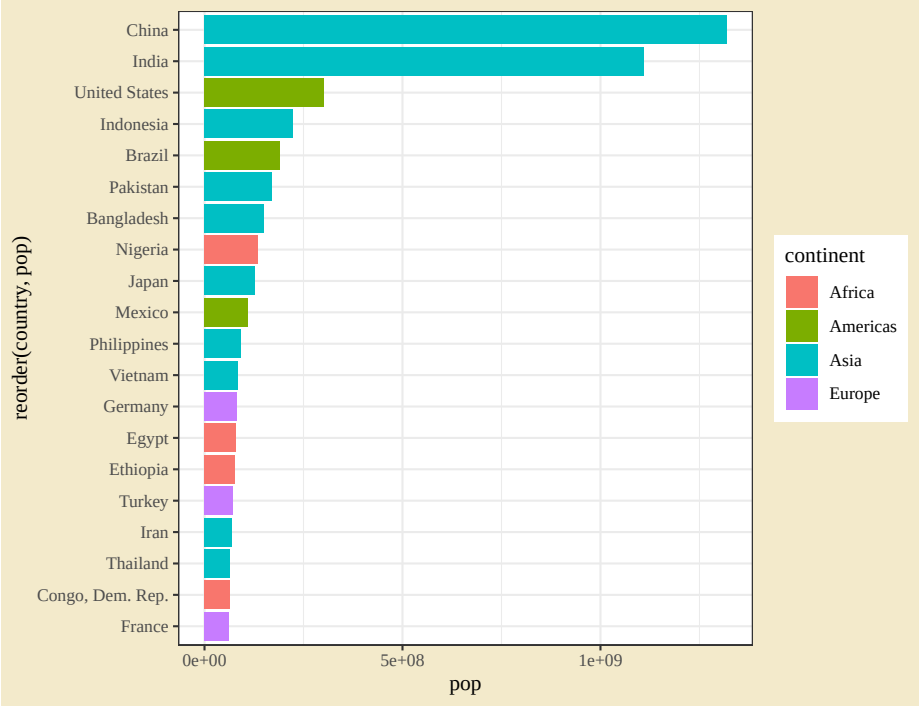
```
ggplot(gapminder, aes(x = year, y = lifeExp, color = continent, group = country)) +
  geom_line()
```



Stapeldiagram ges av `geom_bar`. Om diagrammet ska visa ett urval av data kan man skriva grafen som sista steget i en längre pipe, t.ex.

```
gapminder %>%
  filter(year == 2007) %>%
  filter(rank(-pop) <= 20) %>%
  ggplot(aes(x = pop, y = reorder(country, pop), fill = continent)) +
  geom_bar(stat = "identity")
```

4	3	0	0	0	1	5	2	2	8
3	1	0	1	2	1	1	4	1	0
2	2	3	0	4	1	5	1	1	3



Här ger `rank(-pop) <= 20` att landets rang, dess värde i storleksordning, är minst 20, `reorder(country, pop)` att länder skrivs ut i grafen i storleksordning, och `stat = "identity"` att värdet för pop ska skrivas ut i grafen som det är (utan att transformeras).

5.6 Övningar

Övning 5.1 (Antal barn)**.** Antal barn per familj för 30 familjer är

a. Sammanfatta materialet i en frekvenstabell och illustrera materialet med ett lämpligt diagram.

b. Beräkna genomsnittligt antal barn per familj.

c. Beräkna variansen.

d. Pröva att ändra sista värdet till 30 och se vad som händer med resultaten.

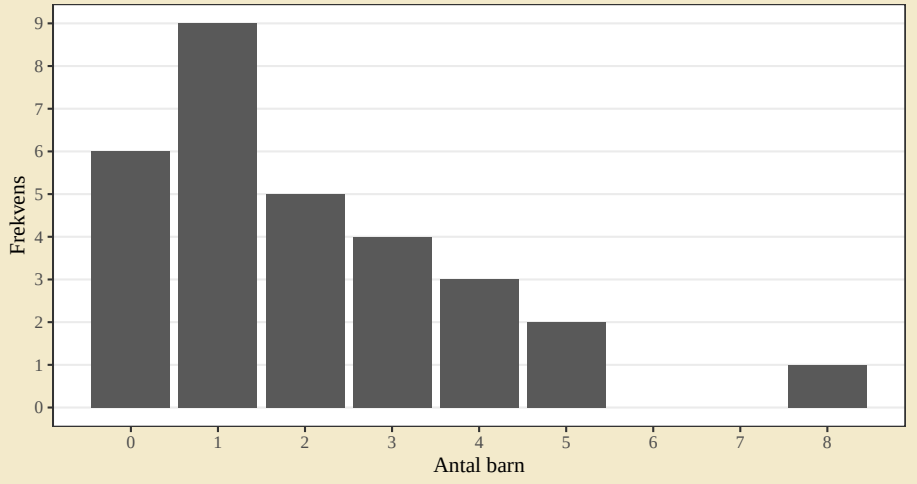
Lösningsförslag 5.1 (Antal barn)**.** Datan kan illustreras med ett stapeldiagram och beskrivande statistik kan beräknas med lämpliga funktioner.

a.

```
dat <- c(4,3,0,0,0,1,5,2,2,8,3,1,0,1,2,1,1,4,1,0,2,2,3,0,4,1,5,1,1,3)
table(dat)
```

```
## dat
## 0 1 2 3 4 5 8
## 6 9 5 4 3 2 1
```

```
dat <- tibble(`Antal barn` = dat)
ggplot(dat, aes(`Antal barn`)) +
  geom_bar() +
  scale_x_continuous(breaks = 0:8) +
  scale_y_continuous(breaks = 0:10) +
  ylab("Frekvens") +
  theme(panel.grid.minor = element_blank(),
        panel.grid.major.x = element_blank())
```



b.

```
dat %>% summarise_all(c(mean, median, var))
```

```
## # A tibble: 1 x 3
##     fn1   fn2   fn3
##   <dbl> <dbl> <dbl>
## 1  2.03   1.5  3.55
```

c.

```
var(dat$`Antal barn`)
```

```
## [1] 3.550575
```

d.

17.4	20.4	20.0	20.0	18.4
18.6	18.6	15.3	16.5	18.0
16.3	18.0	12.8	15.5	18.0

```
dat$`Antal barn`[30] <- 30 # Ändra trettionde värde till 30
dat %>% summarise_all(c(mean, median, var))
```

```
## # A tibble: 1 x 3
##     fn1   fn2   fn3
##   <dbl> <dbl> <dbl>
## 1  2.93   1.5  29.7
```

Medelvärde ökar med ungefär 1, medianen är densamma (varför?) och variansen ökar kraftigt.

Övning 5.2 (Darwindata)**.** Följande data gäller längden (i tum) hos 15 stycken plantor som fortplantats genom självbefruktning:

a. Beräkna medelvärdet.

b. Beräkna medianen.

c. Beräkna variansen.

d. Beskriv fördelningen med ett histogram.

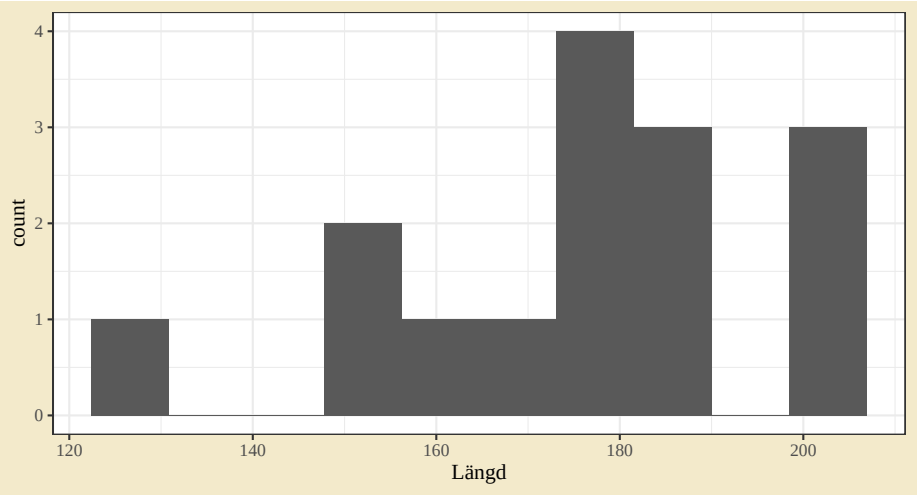
e. Beskriv fördelningen med ett lådagram.

Lösningsförslag 5.2 (Darwindata)**.** Funktionen `summarise_all` kan användas för att beräkna medelvärde, median och varians. Grafer kan konstrueras med `geom_histogram` och `geom_boxplot`.

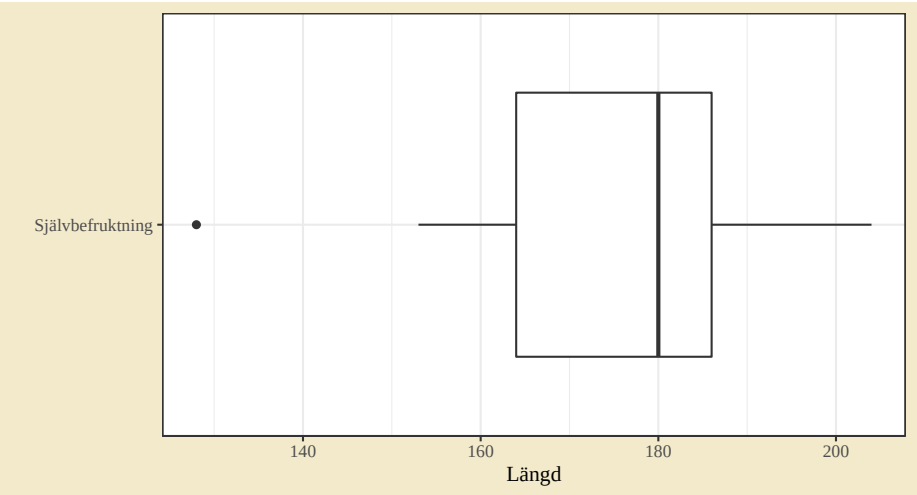
```
dat <- tibble(Längd = c(174,204,200,200,184,186,186,
                        153,165,180,163,180,128,155,180))
dat %>%
  summarise_all(c(mean, median, var))
```

```
## # A tibble: 1 x 3
##     fn1   fn2   fn3
##   <dbl> <dbl> <dbl>
## 1  176.   180  415.
```

```
ggplot(dat, aes(Längd)) +
  geom_histogram(bins = 10)
```




```
ggplot(dat, aes(Längd, "Självbefruktning")) +
  geom_boxplot() +
  ylab("")
```



Storleken på intervallen i histogrammets x-axel kan styras med `binwidth`. Med så få observationer är histogram sällan informativt.

Övning 5.3 (Stapeldiagram med felstaplar)**.** Darwins studie gav följande data:

Konstruera ett stapeldiagram med felstaplar där stapelns höjd ges av medelvärdet inom gruppen och felstapelns längd av standardavvikelsen inom gruppen.

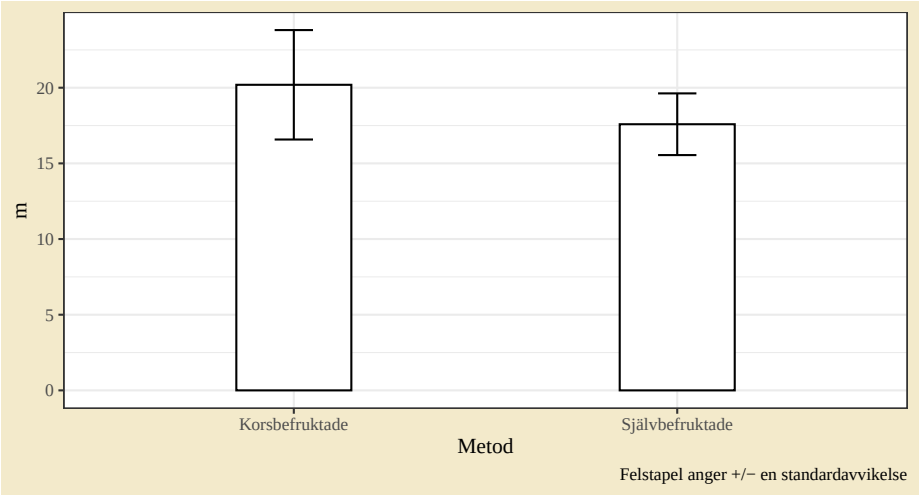
Datan finns tillgänglig i arket *Darwin* i excelfilen *Uppgiftsdata.xlsx*. Exceldata kan läsas in med funktionen `read_excel` från paketet `readxl`.

Lösningsförslag 5.3 (Stapeldiagram med felstaplar)**.** Felstaplar kan kon-

Självbefruktade	Korsbefruktade
17.4	23.5
20.4	12.0
20.0	21.0
20.0	22.0
18.4	19.1
18.6	21.5
18.6	22.1
15.3	20.4
16.5	18.3
18.0	21.6
16.3	23.3
18.0	21.0
12.8	22.1
15.5	23.0
18.0	12.0

strueras genom att beräkna medelvärde och standardavvikelse, och sedan *pipa* (`%>%`) in i en plot där staplar konstrueras med `geom_bar` och felstaplar med `geom_errorbar`.

```
dat <- readxl::read_excel("Data/Uppgiftsdata.xlsx", sheet = "Darwin")
dat %>%
  group_by(Metod) %>%
  summarise(m = mean(Utfall), s = sd(Utfall)) %>%
  ggplot(aes(Metod, m)) +
  geom_bar(stat = "identity", width = 0.3, col = "black", fill = "white") +
  geom_errorbar(aes(ymin = m - s, ymax = m + s), width = 0.1) +
  labs(caption = "Felstapel anger +/- en standardavvikelse")
```



Det är inte alltid klart vilket spridningsmått felstaplarna illusterar (vanliga alternativ är standardavvikelsen, medelfelet (standardavvikelsen delat på roten ur stickprovsstorleken) och konfidensintervallet). Det är därför god praxis att skriva ut vad felstaplarna anger.

Övning 5.4 (Animal Crossing)**.** TidyTuesday (https://github.com/rfordatascience/tidytuesday) är ett R-kopplat projekt som varje vecka släpper ett nytt dataset, med tanken att vemsomhelst kan analysera datan och publicera informativa grafer. Ta gärna titt på Twitter under *#tidytuesday* (helst på en tisdag förstås). Ett dataset från TidyTuesday täcker bybor från spelserien *Animal Crossing*. Datan kan läsas in med

```
villagers <- readr::read_csv('https://raw.githubusercontent.com/rfordatascience/tidytu
```

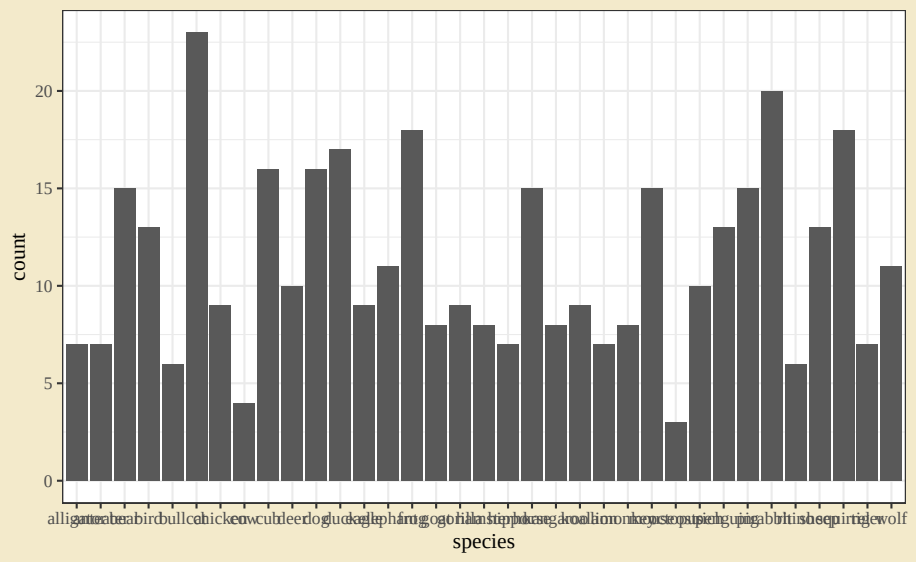
a. Illustrera fördelning mellan arter (kolumnen *species*) med ett lämpligt diagram. Försök att med hjälp av en internetsökning lösa de problem som kan uppstå med överlappande etiketter och ordning på staplar.

b. Utveckla diagrammet från (a) genom att på lämpligt sätt ange personlighet (kolumnen *personality*).

c. Vilken är spelens mest populära låt?

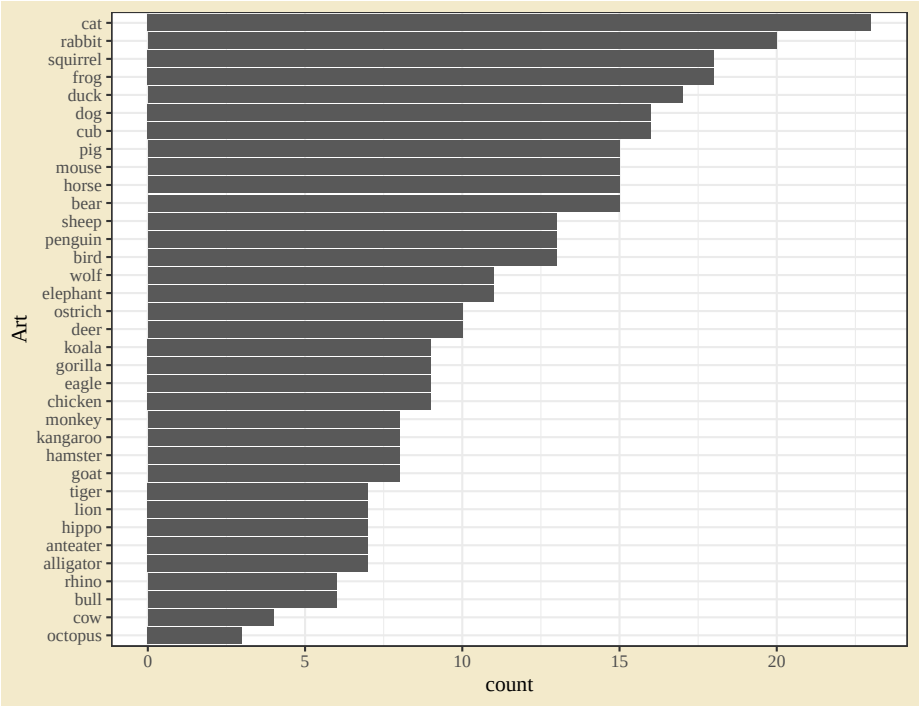
Lösningsförslag 5.4 (Animal Crossing)**.** a.

```
ggplot(villagers, aes(species)) +
  geom_bar()
```



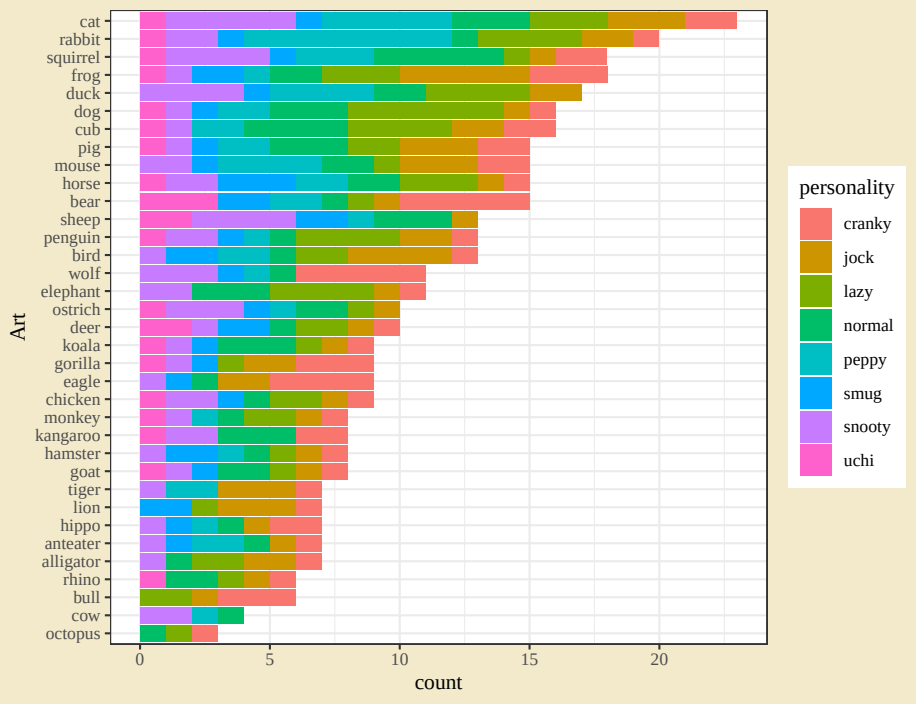
En internetsökning på exempelvis 'ggplot2 bar order' kommer ge flera möjliga lösningar - en är att använda `fct_reorder` från paketet `forcats`, vilket ordnar en faktor efter antalet förekomster (genom att sätta funktionen till `length`). Överlappande etiketter kan också lösas på ett par olika sätt (lutande etiketter, etiketter på olika nivåer). En enkel lösning är att vrida staplarna och få etiketterna till vänster med `coord_flip`.

```
ggplot(villagers, aes(forcats::fct_reorder(species, species, .fun = length))) +
  geom_bar() +
  xlab("Art") +
  coord_flip()
```



b.

```
ggplot(villagers, aes(forcats::fct_reorder(species, species, .fun = length), fill = pe
  geom_bar() +
  xlab("Art") +
  coord_flip()
```



c.

```
villagers %>%
  count(song, sort = T)
```

```
## # A tibble: 93 x 2
##    song              n
##    <chr>         <int>
##  1 <NA>             11
##  2 K.K. Country     10
##  3 Forest Life       9
##  4 I Love You        7
##  5 Imperial K.K.     7
##  6 K.K. Lament       7
##  7 K.K. Lullaby      7
##  8 K.K. Ragtime      7
##  9 K.K. Soul         7
## 10 Cafe K.K.         6
## # ... with 83 more rows
```

Det vanligaste alternativet är att sång saknas, vilket då anges som `NA`. Den mest populära faktiska sången är *KK Country*.

Övning 5.5 (Väderstation Falsterbo)**.** SMHI publicerar historisk väderdata

från en stor mängd väderstationer (https://www.smhi.se/data/meteorologi/ladda-ner-meteorologiska-observationer). Ett exempel på en sådan fil, från väderstationen i Falsterbo, finns bland kursdatan. Notera att flera inledande rader innehåller metadata och inte ska läsas in. Filen är dessutom inte kommaseparerad utan semikolon-separerad. Den kan läsas in med funktionen `read_csv2` från `readr`. Vid inläsning kan man också ändra datatyp för temperaturkolumnen, som annars felaktigt tolkats som en textkolumn.

```
dat <- read_csv2("Data/smhi-opendata_1_52230_20210912_114534.csv", skip = 9) %>%
  mutate(Lufttemperatur = as.numeric(Lufttemperatur))
```

a. Vilken är den högst uppmätta temperaturen?

b. Beräkna max-temperaturen för varje år. Plotta ett linjediagram med år på x-axeln och maxtemperatur på y-axeln.

c. Skapa en variabel som anger observationens årtionde. Beräkna medeltemperaturen per årtionde och illustrera med lämplig graf.

Lösningsförslag 5.5 (Väderstation Falsterbo)**.** a. Ordna data efter lufttemperatur och skriv ut de översta raderna.

```
dat %>% arrange(desc(Lufttemperatur)) %>% print(n = 10)
```

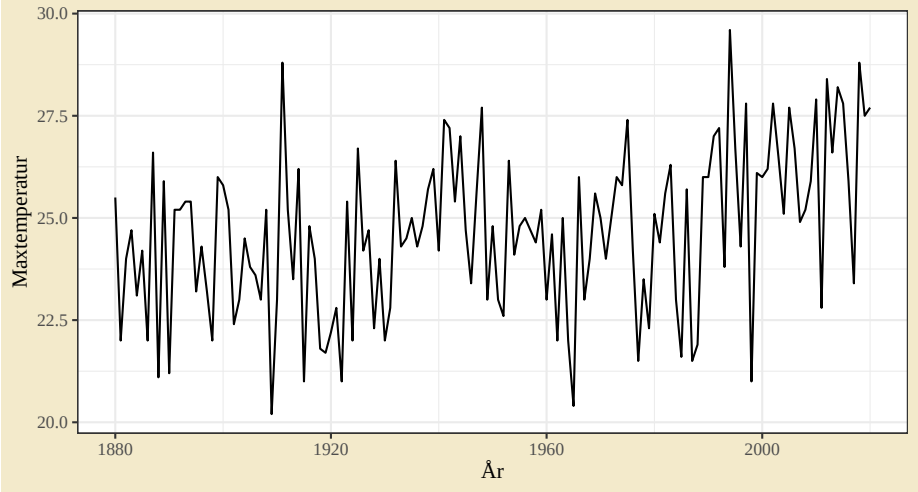
```
## # A tibble: 245,869 x 6
##    Datum      `Tid (UTC)` Lufttemperatur Kvalitet ...5  `Tidsutsnitt:`
##    <date>     <time>                <dbl> <chr>    <lgl> <chr>
##  1 1994-07-31 12:00                  29.6 G        NA    <NA>
##  2 1911-08-01 13:00                  28.8 G        NA    <NA>
##  3 2018-08-03 15:00                  28.8 G        NA    <NA>
##  4 2018-07-26 15:00                  28.7 G        NA    <NA>
##  5 1994-07-28 12:00                  28.4 G        NA    <NA>
##  6 2012-08-19 15:00                  28.4 G        NA    <NA>
##  7 1994-07-28 15:00                  28.3 G        NA    <NA>
##  8 2014-07-09 12:00                  28.2 G        NA    <NA>
##  9 2018-08-02 15:00                  28.2 G        NA    <NA>
## 10 2018-07-26 12:00                  28.1 G        NA    <NA>
## # ... with 245,859 more rows
```

Fem av de tio högsta mätningarna inträffade mellan 26 juli och 3 augusti 2018. Två samma dag.

b. Beräkningen kan göras genom att skapa en årsvariabel med `year` från `lubridate`-paketet. Därefter gruppera per år och summera med `max`-funktionen.

```
dat %>%
  mutate(År = lubridate::year(Datum)) %>%
  group_by(År) %>%
  summarise(Maxtemperatur = max(Lufttemperatur)) %>%
  ggplot(aes(År, Maxtemperatur)) +
```

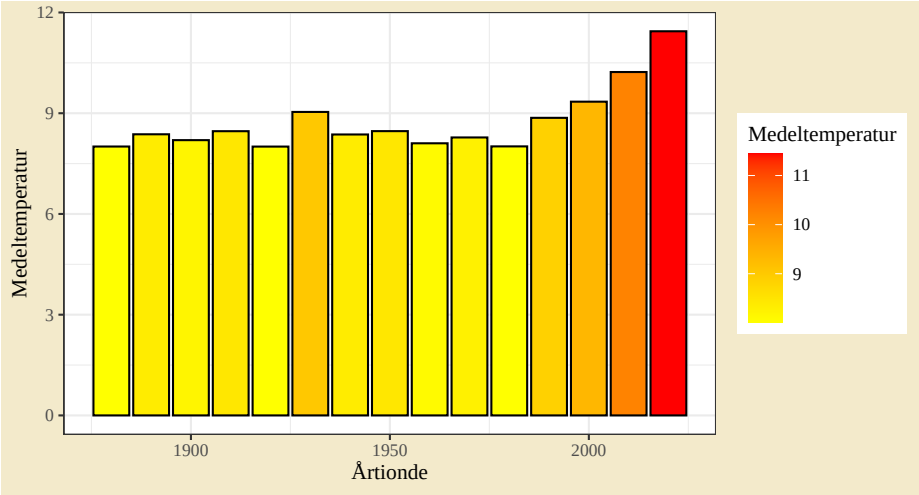
```
  geom_line()
```



Datan visar på stor variation mellan år.

c. Likt (b) kan man skapa en årsvariabel och från den beräkna årtionde genom att dela på tio, använda `floor` för att ta bort decimalen, och sedan multiplicera med tio. Därefter kan man gruppera efter årtionde och summera till medelvärdet.

```
dat %>%
  mutate(Årtionde = floor(lubridate::year(Datum) / 10) * 10) %>%
  group_by(Årtionde) %>%
  summarise(Medeltemperatur = mean(Lufttemperatur)) %>%
  ggplot(aes(Årtionde, Medeltemperatur, fill = Medeltemperatur)) +
  geom_bar(stat = "identity", col = "black") +
  scale_fill_gradient(low = "yellow", high = "red")
```



Temperaturen verkar öka i Falsterbo. Märk att stapeln för 2020-talet bara innefattar ett år.

Övning 5.6 (Allsvenskan för herrar)**.** Bland kursdatan finns en fil med matchresultat från herrarnas fotbollsallsvenska, 1924 - 2019. Läs in datan och undersök följande.

a. Hur många gånger har Malmö FF mött Mjällby? Vilket datum inföll Mjällbys enda seger?

b. Fyra säsonger har lag gått rent på hemmaplan (segrar i samtliga matcher). Vilka är de två lagen och vilka är de fyra säsongerna?

c. Vilka är de målrikaste matcherna?

d. Producera ett spridningsdiagram med hemmamål på x-axeln och bortamål på y-axeln. Hur kan man hantera överlappande punkter?

Lösningsförslag 5.6 (Allsvenskan för herrar)**.** Filen kan läsas in med `read_csv` från `readr`.

```
dat <- read_csv("Data/Allsvenskan, herrar, 1924-2019.csv")
```

a. Matcher kan filtreras ut genom filter-funktionen. Två skilda rader kan användas för att få både hemma- och borta-matcher.

```
dat %>% filter(hemma == "Malmö FF", borta == "Mjällby")
```

```
## # A tibble: 8 x 6
##   datum      sasong hemma    borta   hemmamal bortamal
##   <date>     <chr>  <chr>    <chr>      <dbl>    <dbl>
## 1 1980-05-03 1980   Malmö FF Mjällby        1        0
## 2 1983-04-17 1983   Malmö FF Mjällby        1        1
## 3 1985-06-13 1985   Malmö FF Mjällby        2        0
```

```
## 4 2010-11-07 2010   Malmö FF Mjällby        2        0
## 5 2011-04-20 2011   Malmö FF Mjällby        1        0
## 6 2012-09-01 2012   Malmö FF Mjällby        1        1
## 7 2013-09-30 2013   Malmö FF Mjällby        1        0
## 8 2014-09-27 2014   Malmö FF Mjällby        4        1
```

```
dat %>% filter(hemma == "Mjällby", borta == "Malmö FF")
```

```
## # A tibble: 8 x 6
##   datum      sasong hemma   borta    hemmamal bortamal
##   <date>     <chr>  <chr>   <chr>       <dbl>    <dbl>
## 1 1980-07-30 1980   Mjällby Malmö FF        1        1
## 2 1983-10-02 1983   Mjällby Malmö FF        1        1
## 3 1985-08-11 1985   Mjällby Malmö FF        1        1
## 4 2010-05-15 2010   Mjällby Malmö FF        4        2
## 5 2011-08-06 2011   Mjällby Malmö FF        1        1
## 6 2012-05-03 2012   Mjällby Malmö FF        2        2
## 7 2013-05-17 2013   Mjällby Malmö FF        2        2
## 8 2014-05-22 2014   Mjällby Malmö FF        0        1
```

Lagen har mötts sexton gånger. Den enda malmöförlusten inträffade 2010 på bortaplan.

b. Andelen vinster på hemmaplan kan tas fram genom att gruppera på lag och säsong, beräkna antalet segrar och antalet matcher, och slutligen beräkna andelen segrar. För att beräkna antalet vinster kan man använda `sum(hemmamal > bortamal)` - summan av antal gånger hemmamålen överstiger bortamålen. För det totala antalet matcher en säsong kan man använda funktionen `n()` - antalet rader i en viss gruppering.

```
dat %>%
  group_by(hemma, sasong) %>%
  summarise(Vinster = sum(hemmamal > bortamal), Total = n()) %>%
  mutate(Proportion = Vinster / Total) %>%
  filter(Proportion == 1)
```

```
## # A tibble: 4 x 5
## # Groups:   hemma [2]
##   hemma        sasong    Vinster Total Proportion
##   <chr>        <chr>       <int> <int>      <dbl>
## 1 IFK Göteborg 1934_1935      11    11          1
## 2 IFK Göteborg 1941_1942      11    11          1
## 3 Malmö FF     1949_1950      11    11          1
## 4 Malmö FF     1950_1951      11    11          1
```

Två lag (IFK Göteborg och Malmö FF) med två gånger var (1934-35 och 1941-42 respektive 1949-50 och 1950-51).

c. Skapa en ny variabel för totalt antal mål och sortera efter den kolumnen.

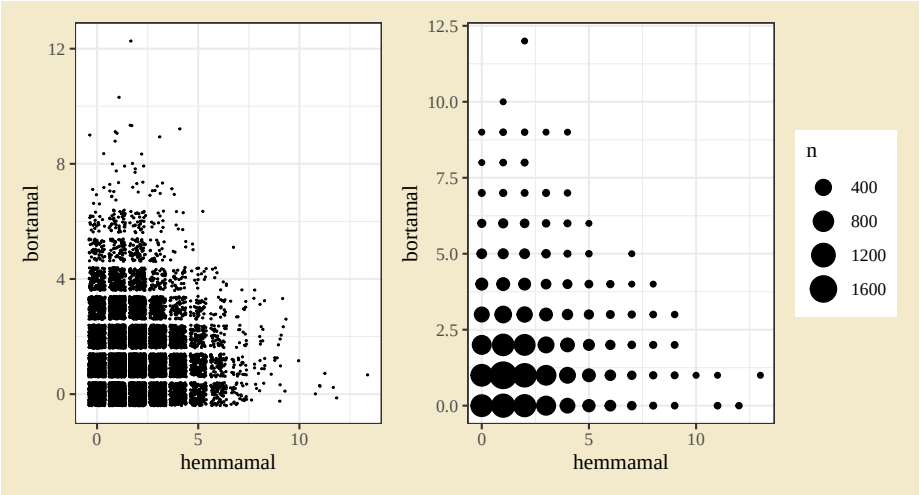
```
dat %>%
  mutate(Mål = hemmamal + bortamal) %>%
  arrange(-Mål) %>%
  print(n = 5)
```

```
## # A tibble: 14,996 x 7
##   datum      sasong    hemma         borta      hemmamal bortamal   Mål
##   <date>     <chr>     <chr>         <chr>         <dbl>    <dbl> <dbl>
## 1 1928-10-21 1928_1929 Helsingborg   Eskilstuna       13        1    14
## 2 1936-04-19 1935_1936 Eskilstuna    Elfsborg          2       12    14
## 3 1954-11-06 1954_1955 Sandviken AIK AIK               4        9    13
## 4 1927-10-30 1927_1928 AIK           Eskilstuna        8        4    12
## 5 1928-11-04 1928_1929 Sleipner      Eskilstuna        7        5    12
## # ... with 14,991 more rows
```

Två gånger har det blivit 14 mål. Bägge gångerna förluster för Eskilstuna.

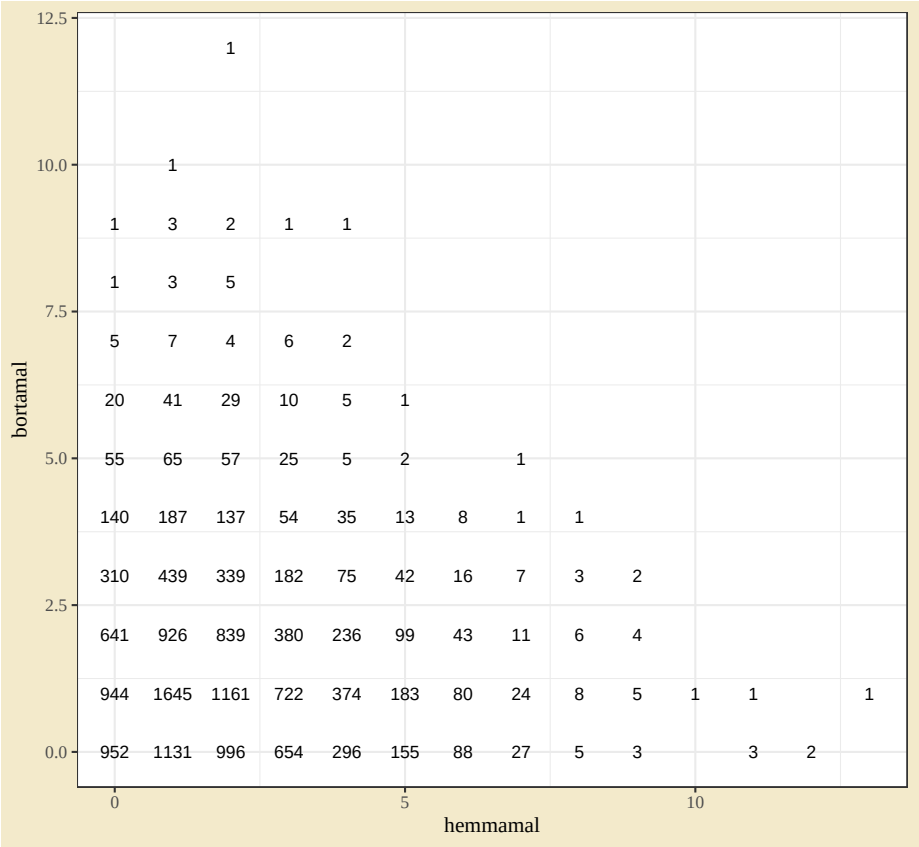
d. Ett spridningsdiagram kan skapas med `ggplot`. Ett sätt att hantera överlappande punkter är att ge punkterna ett *jitter* (ett slumpmässig justering så att de inte längre överlappar). Man kan också använda `geom_count`, vilket gör att punktens storlek beror på antal överlappande observationer, eller skriva ut antalet överlappande fall.

```
g1 <- ggplot(dat, aes(hemmamal, bortamal)) +
  geom_jitter(size = 0.1)

g2 <- ggplot(dat, aes(hemmamal, bortamal)) +
  geom_count()

library(patchwork)
g1 + g2
```



Det verkar som att 1-1 är det vanligaste resultatet. För att få en siffra kan man räkna hemma- och bortamål med `count`, för att sedan plotta med `geom_text`.

```
dat %>%
  count(hemmamal, bortamal) %>%
  ggplot(aes(hemmamal, bortamal, label = n)) +
  geom_text(size = 3)
```



Det verkar som att 1-1, följt av 2-1 och 1-0 är de vanligaste resultaten.

Övning 5.7 (Egenskapad standardavvikelsefunktion)**.** En styrka med R är hur enkelt nya funktioner kan skapas. Många standardfunktioner är skrivna i R. Ta som exempel funktionen för standardavvikelse:

```
sd
```

```
## function (x, na.rm = FALSE)
## sqrt(var(if (is.vector(x) || is.factor(x)) x else as.double(x),
##     na.rm = na.rm))
## <bytecode: 0x0000000025f69d90>
## <environment: namespace:stats>
```

Standardavvikelsen beräknas alltså som kvadratroten (`sqrt`) av variansen (`var`). Sedan finns också lite mer avancerad kod för att hantera olika typer av ingångsdata.

Standardavvikelsen av n datapunkter ges av

$$s = \sqrt{\frac{1}{n-1}\sum_{i=1}^{n}(x_i - \bar{x})^2},$$

vilket kan brytas upp i steg som (1) beräkna medelvärdet, (2) dra ifrån medelvärdet från varje datavärde, (3) kvadrera differenserna, (4) summera kvadraterna, (5) dela summan med $n-1$, och (6) ta kvadratroten ur kvoten.

Skapa en egen standardavvikelsefunktion baserat på de stegen. Funktionen ska ta en godtycklig mängd värden som ingångsvärden och ge standardavvikelsen som output.

Lösningsförslag 5.7 (Egenskapad standardavvikelsefunktion)**.** Funktionen nedan går igenom stegen för att beräkna standardavvikelsen.

```
egen_sd <- function(x){
  n <- length(x)           # Spara datans storlek som n
  medel <- mean(x)         # 1
  diffs <- x - medel       # 2
  kvadrater <- diffs^2     # 3
  summa <- sum(kvadrater) # 4
  kvot <- summa / (n - 1) # 5
  s <- sqrt(kvot)          # 6
  s                        # Ange output
}

dat <- c(3,1,4,5,9)
sd(dat)
```

```
## [1] 2.966479
```

```
egen_sd(dat)
```

```
## [1] 2.966479
```

Den egenskapade funktionen ger samma utfall som R funktionen `sd`.

Kapitel 6

Sannolikhetsfördelningar och slumptal

6.1 Fördelningar

R kommer med en stor mängd funktioner för att beräkna sannolikheter ur kända sannolikhetsfördelningar såsom binomial- och normalfördelningen. För en lista på fördelningar täckta av grundpaketen i R kan man köra `?distributions`.

För en *likformig fördelning* gäller att alla utfall mellan 0 och 1 är lika sannolika. Som exempel kan man tänka sig att man stoppar ett tidtagarur vid ett slumpmässigt tillfälle och tittar på utfallets decimaler - de kommer ge ett värde mellan 0 och 1 och det finns ingen anledning att tro att vissa värden är mer sannolika än andra. Sannolikhetsfunktionen $f(x)$ kan beräknas genom $dunif$, där `d` står för *density* (täthet) och `unif` anger en *uniform* fördelning. Fördelningens sannolikhetsfunktion är 0 för värden på x under 0 eller över 1, och däremellan är sannolikhetsfunktionen 1.

```
dunif(-0.1)
```

```
## [1] 0
```

```
dunif(0.1)
```

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

```
dunif(1.1)
```

```
## [1] 0
```

Fördelningsfunktionen anger sannolikheten för ett värde mindre än x, $F(x) = P(X \leq x)$ och kan i R beräknas genom funktionen $punif$, där `p` står för *probability*.

```
punif(-0.1)
```

```
## [1] 0
```

```
punif(0.1)
```

```
## [1] 0.1
```

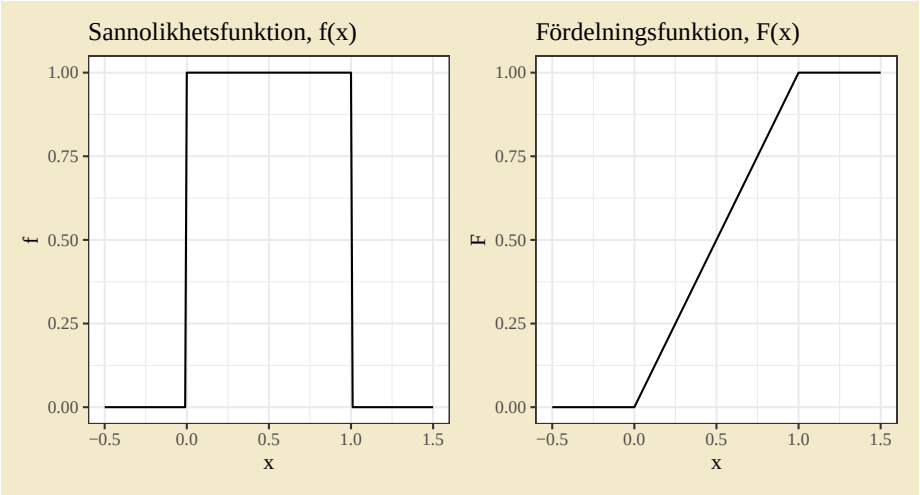
```
punif(1.1)
```

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

Fördelningsfunktionen för en likformig fördelning är för värden på x under 0, 1 för värden på x över 1, och däremellan lika med x.

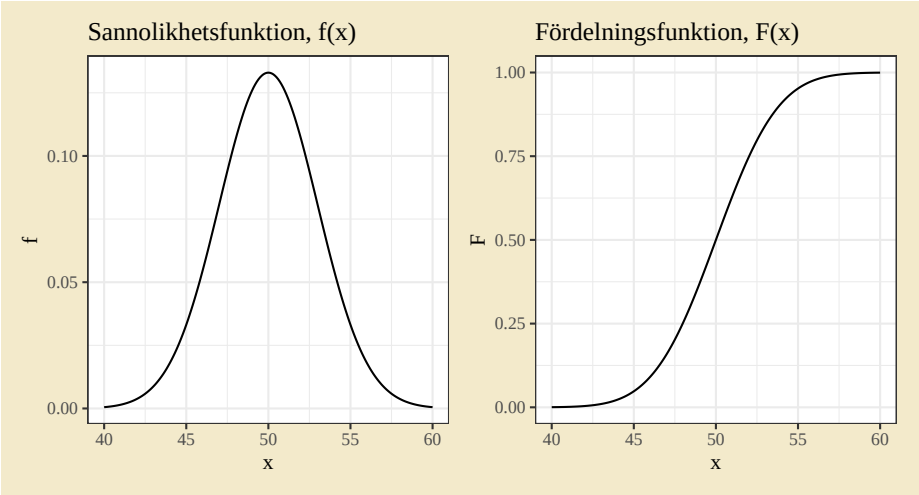
Sannolikhetsfunktion och fördelningsfunktion kan illustreras med grafer. Funktionen `seq` används för att skapa en sekvens från ett värde till ett annat värde. Paketet `patchwork` används för att kombinera två grafer. Funktionen `tibble` används för att skapa ett `tibble`-objekt - funktionen fungerar likt `data.frame` i att kolumner anges som `namn = kolumnvärden`.

```
g1 <- tibble(x = seq(from = -0.5, to = 1.5, by = 0.01),
             f = dunif(x)) %>%
  ggplot(aes(x, f)) +
  geom_line() +
  labs(title = "Sannolikhetsfunktion, f(x)")

g2 <- tibble(x = seq(from = -0.5, to = 1.5, by = 0.01),
             F = punif(x)) %>%
  ggplot(aes(x, F)) +
  geom_line() +
  labs(title = "Fördelningsfunktion, F(x)")

library(patchwork)
g1 + g2
```



Motsvarande funktioner för en normalfördelning ges av `dnorm` och `pnorm`. Medelvärde och standardavvikelse kan sättas genom argumenten `mean` och `sd`. Här ges ett exempel på en normalfördelning med medelvärdet 50 och standardavvikelse 3.

```
g1 <- tibble(x = seq(from = 40, to = 60, by = 0.1),
             f = dnorm(x, mean = 50, sd = 3)) %>%
  ggplot(aes(x, f)) +
  geom_line() +
  labs(title = "Sannolikhetsfunktion, f(x)")

g2 <- tibble(x = seq(from = 40, to = 60, by = 0.1),
             F = pnorm(x, mean = 50, sd = 3)) %>%
  ggplot(aes(x, F)) +
  geom_line() +
  labs(title = "Fördelningsfunktion, F(x)")

g1 + g2
```



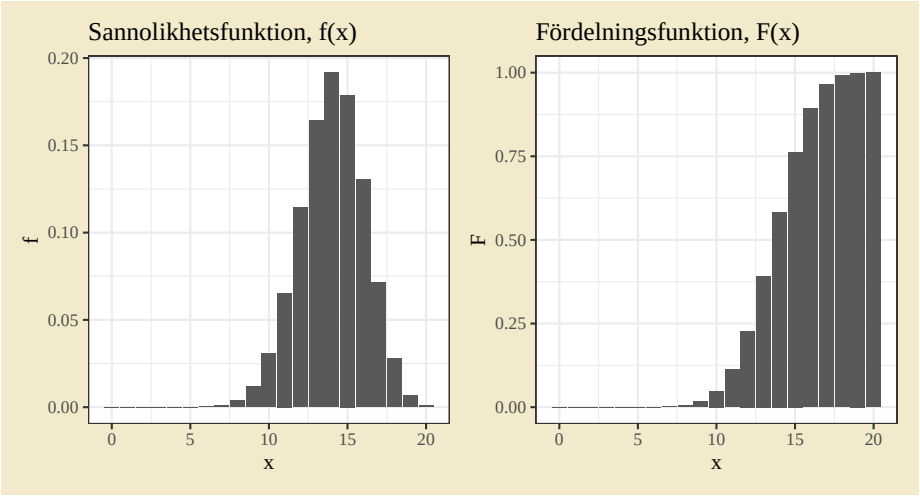
Fördelningsfunktionen anger sannolikheten att ett utfall ligger under värdet x. Om man vill beräkna det omvända fallet - ett x-värde sådant att sannolikheten att ligga under det värdet är en viss sannolikhet p - använder man *kvantilfunktionen*. Som exempel beräknas ett värde på x-axeln sådant att en fjärdedel ligger under det värdet i en normalfördelning med $\mu = 50$ och $\sigma = 3$.

```
qnorm(0.25, mean = 50, sd = 3)
```

```
## [1] 47.97653
```

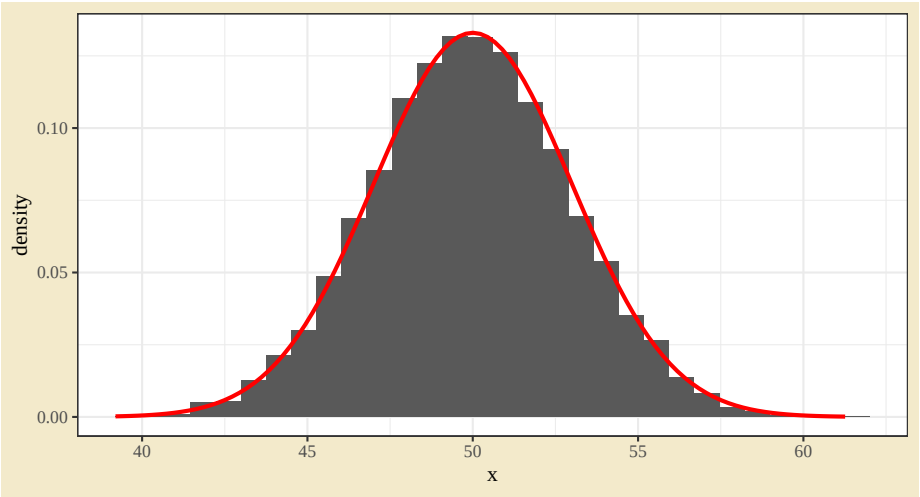
Sannolikhetsfunktion och fördelningsfunktion kan även beräknas för diskreta fördelningar. Binomialfördelningen ges till exempel av `dbinom` och `pbinom` med argumenten `size` för parametern n och `prob` för parametern p. Här ges ett exempel på en binomialfördelning med $n = 20$ och $p = 0.7$. Eftersom en diskret fördelning oftast illustreras med stapeldiagram ersätts `geom_line()` med `geom_bar(stat = "identity")`.

```
g1 <- tibble(x = 0:20,
             f = dbinom(x, size = 20, prob = 0.7)) %>%
  ggplot(aes(x, f)) +
  geom_bar(stat = "identity") +
  labs(title = "Sannolikhetsfunktion, f(x)")

g2 <- tibble(x = 0:20,
             F = pbinom(x, size = 20, prob = 0.7)) %>%
  ggplot(aes(x, F)) +
  geom_bar(stat = "identity") +
  labs(title = "Fördelningsfunktion, F(x)")

g1 + g2
```



Utöver sannolikhetsberäkningar har R funktioner för att skapa slumptal från en angiven fördelning. Dessa anges genom bokstaven `r` (för *random*) följt av fördelningens namn, t.ex. `rnorm` för normalfördelningen och `rbinom` för binomialfördelningen. I exemplet nedan dras tiotusen observationer från en normalfördelning. Histogrammet visar att slumptalen ungefär följer den teoretiska fördelningen (här utritad med funktionen `stat_function`).

```
dat <- tibble(x = rnorm(10000, mean = 50, sd = 3))
ggplot(dat, aes(x)) +
  geom_histogram(aes(y =..density..)) +
  stat_function(fun = dnorm, args = list(mean = 50, sd = 3),
                col = "red", size = 1)
```



Argumentet `y = ..density..` anger att y-axeln ska vara i andelar, istället för

antal.

6.2 Egna funktioner

Det är möjligt att definera egna funktioner genom konstruktionsfunktionen `function`. Ett enkelt exempel på en funktion som tar ett värde och ger det värdet plus 4:

```
add_four <- function(x){
  y <- x + 4
  y
}

add_four(5)
```

```
## [1] 9
```

Här är `add_four` namnet på den funktion som skapas och `function(x)` anger att man skapar en funktion med ett ingångsvärde `x`. Stycket inom `{...}` är själva funktionsberäkning. I det här fallet skapar funktionen ett objekt `y` som ges av x plus 4, och därefter skrivs resultatet ut. Det som skrivs ut i funktionens sista rad blir funktionens output.

Funktioner kan ha mer än ett ingående värde, t.ex.

```
add_two_numbers <- function(x, y){
  res <- x + y
  res
}

add_two_numbers(15, 3)
```

```
## [1] 18
```

Ett ingångsvärde kan ges ett grundläge genom att ange det som ett argument i funktionen, t.ex.

```
add_two_numbers <- function(x, y = 3){
  res <- x + y
  res
}

add_two_numbers(15)
```

```
## [1] 18
```

```
add_two_numbers(15, 12)
```

```
## [1] 27
```

Om inget värde anges för det andra ingåendevärdet (y) sätts det värdet till 3, eftersom det anges i definitionen av funktionen.

6.3 Simuleringar

Med hjälp av egna funktioner och slumptal kan man utforska många grundläggande statistiska resultat. Teoretiska resultat säger till exempel att om en slumpvariabel X har standardavvikelsen σ, så har ett medelvärde av n observationer av X standardavvikelsen $\sigma/\sqrt{n}$. För att undersöka detta skapas en funktion som för ett angivet värde på n ger ett medelvärde av n slumptal.

```
mean_of_n_obs <- function(n = 1){
  x <- rnorm(n, mean = 0, sd = 1)
  mean(x)
}
```

Funktionen tar ingångsvärdet n, simulerar n stycken slumptal från en normalfördelning med medelvärde 0 och standardavvikelse 1, och ger ut medelvärdet av de slumptalen.

Funktionen `replicate` kan användas för att köra en funktion upprepade gånger - `replicate(100, mean_of_n_obs(n = 10))` upprepar den definerade funktionen 100 gånger och ger alltså 100 stycken medelvärden där varje medelvärde beräknas från 10 observationer.

Eftersom standardavvikelsen i den ursprungliga dragningen var 1 ($\sigma = 1$) bör standardavvikelsen i ett medelvärde av 16 observationer vara 0.25 ($\sigma/\sqrt{16} = 1/4 = 0.25$). Det testas genom att beräkna tiotusen medelvärden (genom `replicate`) och beräkna standardavvikelsen i den serien av medelvärden.

```
means <- replicate(10000, mean_of_n_obs(n = 16))
sd(means)
```

```
## [1] 0.2481614
```

Detta kan uppepas för andra stickprovsstorlekar - ett stickprov om hundra observationer bör ge ett värde kring 0.1 (eftersom $\sigma/\sqrt{100} = 1/10 = 0.1$).

```
means <- replicate(10000, mean_of_n_obs(n = 100))
sd(means)
```

```
## [1] 0.09975045
```

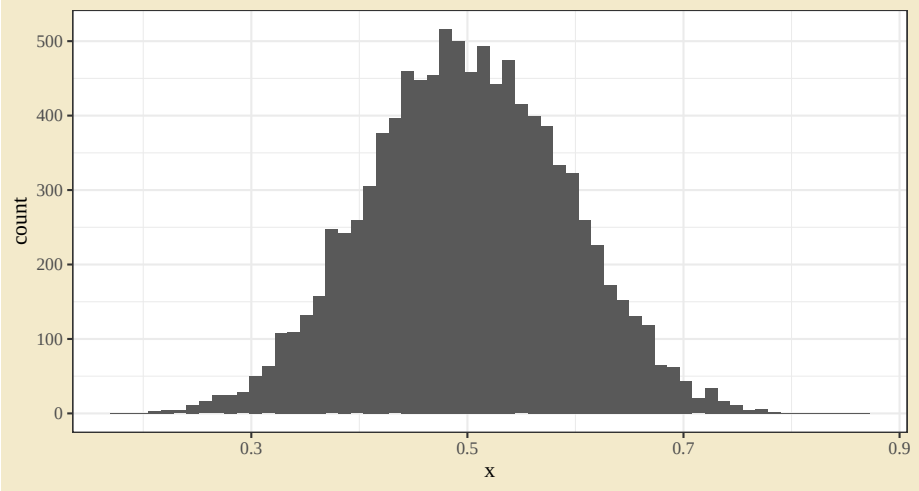
Sannolikhetsteorin viktigaste resultat är *centrala gränsvärdesatsen*, som säger att medelvärden av flera lika slumpvariabler är ungefärligt normalfördelade även om den ursprungliga slumpvariabeln inte är det. Detta kan illustreras genom att dra slumptal från valfri fördelning, beräkna medelvärden av de slumptalen, och sedan titta på fördelningen för de medelvärdena. Ett första steg kan vara att skriva en funktion som drar slumptal och beräknar ett medelvärde.

```
draw_random_calculate_mean <- function(){
  x <- runif(10)
  mean(x)
}
```

Den ursprungliga fördelningen är här en likformig fördelning och stickprovsstorleken är 10.

Funktionen `replicate` används för att dra tiotusen medelvärden och `ggplot` används för att skapa ett histogram över medelvärdena.

```
means <- tibble(x = replicate(10000, draw_random_calculate_mean()))
ggplot(means, aes(x)) +
  geom_histogram(bins = 60)
```



Medelvärdena följer en ungefärlig normalfördelning trots att den ursprungliga variabeln följer en likformig fördelning.

6.4 Övningar

Övning 6.1 (Binomial grobarhet)**.** I ett försöks sätts 10 frön med en grobarhetssannolikhet om 60 procent. Antal frön som gror följer då en binomialfördelning med $N = 10$ och $p = 0.6$, vilket kan skrivas $X \sim Bin(10, 0.6)$.

a. Vad är sannolikheten att få exakt 6 groende frön, $P(X = 6)$?

b. Vad är sannolikheten att få högst 6 groende frön, $P(X \leq 6)$?

c. Beräkna slumpvariabelns *fördelningsfunktion*.

d. Illustrera sannolikheterna från (c).

Ledning: `dbinom` och `pbinom` kan beräkna sannolikheter från en binomialfördelning.

Lösningsförslag 6.1 (Binomial grobarhet)**.** Uppgiften kan lösas genom att ta fram sannolikheter med `dbinom` och `pbinom` med argumenten `size = 10` för tio frön och `prob = 0.6` för en grobarhet på 0.6

a.

```
dbinom(x = 6, size = 10, prob = 0.6)
```

```
## [1] 0.2508227
```

b.

```
pbinom(6, size = 10, prob = 0.6)
```

```
## [1] 0.6177194
```

c.

```
tibble(x = 0:10,
       Fördelningsfunktion = pbinom(x, size = 10, prob = 0.6))
```

```
## # A tibble: 11 x 2
##        x Fördelningsfunktion
##    <int>               <dbl>
##  1     0            0.000105
##  2     1            0.00168
##  3     2            0.0123
##  4     3            0.0548
##  5     4            0.166
##  6     5            0.367
##  7     6            0.618
##  8     7            0.833
##  9     8            0.954
## 10     9            0.994
## 11    10            1
```

d.

```
dat <- tibble(x = 0:10,
       Fördelningsfunktion = pbinom(x, size = 10, prob = 0.6),
       Sannolikhetsfunktion = dbinom(x, size = 10, prob = 0.6))

ggplot(dat, aes(x)) +
  geom_bar(aes(y = Fördelningsfunktion), stat = "identity",
           fill = "pink", col = "hotpink", width = 0.5) +
  geom_segment(aes(x = x, xend = x,
                   y = Sannolikhetsfunktion, yend = 0)) +
  scale_x_continuous(breaks = 0:10) +
```

```
  theme(panel.background = element_rect(fill = "purple"),
        panel.grid = element_blank())
```

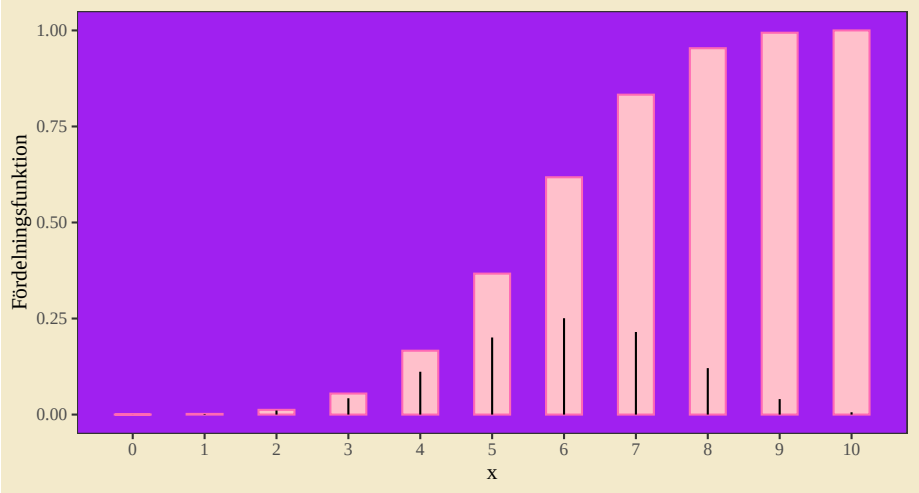


Illustration av fördelningsfunktionen (som breda staplar) med sannolikhetsfunktionen inritad som smala streck. Exempel på argument för val av färger för geom och tema. Sannolikheten i (a) (exakt 6 groende frön) ges av det svarta strecket vid 6 och sannolikheten i (b) (högst 6 groende frön) ges av den breda stapeln vid 6.

Övning 6.2 (Poissonfördelad klöver)**.** Antalet fyrklöver på en slumpmässigt vald kvadratmeter från en gräsmatta är poissonfördelad med väntevärde 1 per m^2.

a. Hur stor är sannolikheten att en slumpmässigt vald m^2 innehåller minst en fyrklöver, $P(X \geq 1)$? Exakt en fyrklöver, $P(X = 1)$?

b. Hur stor är sannolikheten att en slumpmässigt vald yta om 10 kvadratmeter innehåller exakt 10 fyrklöver?

Ledning: Summan av n stycken likadana poissonfördelade variabler är en poissonfördelad variabel med väntevärde givet av n gånger väntevärdet för den enskilda variabeln.

(Från Olsson, *Biometri*.)

Lösningsförslag 6.2 (Poissonfördelad klöver)**.** Sannolikheter för en poissonfördelning kan tas fram med `dpois` för sannolikhetsfunktionen eller `ppois` för fördelningsfunktionen.

a. Antal klöver på en kvadratmeter följer en poissonfördelning med $\lambda = 1$. Sannolikheten för minst en fyrklöver kan beräknas genom ett minus sannolikheten för noll fyrklöver (utfallet noll fyrklöver är ett *komplement* till utfallet en eller fler fyrklöver).

```
# P(X >= 1) = 1 - P(X <= 0)
1 - ppois(0, lambda = 1)
```

```
## [1] 0.6321206
```

```
# P(X = 1)
dpois(1, lambda = 1)
```

```
## [1] 0.3678794
```

b. Summan av poissonfördelade variabler är poissonfördelad med parametern λ given av summan av de ursprungliga variablernas parametervärden. Tio kvadratmeter kan ses som summan av tio stycken observationer av en kvadratmeter. Antal klöver på tio kvadratmeter bör därmed följa en poissonfördelning med $\lambda = 10$.

```
# Y ~ Po(lambda = 10)
dpois(10, lambda = 10)
```

```
## [1] 0.12511
```

Ungefär 12.5 procent.

Övning 6.3 (Sannolikheter från en normalfördelning)**.** Slumpvariabeln X är normalfördelad med medelvärde 2 och varians 9. Beräkna följande

a. $P(X > 2.75)$

b. $P(X \leq 2.75)$

c. $P(X > 2.50)$

d. $P(2.30 < X < 2.45)$

e. $P(X > -0.02)$

(Från Olsson, *Biometri*.)

Lösningsförslag 6.3 (Sannolikheter från en normalfördelning)**.** Sannolikheter från normalfördelningen kan tas fram med `pnorm`.

a. Notera att *variansen* σ^2 är 9 och att standardavvikelsen σ därmed är 3. Funktionen `pnorm` ger sannolikheten att ligga *under* ett givet x-värde. För att beräkna $P(X > 2.75)$ kan ta ett minus $P(X < 2.75)$.

```
1 - pnorm(2.75, mean = 2, sd = 3)
```

```
## [1] 0.4012937
```

b. Direkt tillämpning av `pnorm`.

```
pnorm(2.75, 2, 3)
```

```
## [1] 0.5987063
```

c. Likt (a).

```
1 - pnorm(2.5, 2, 3)
```

```
## [1] 0.4338162
```

d. För att beräkna sannoliketen att ligga mellan två värden kan man ta skillnaden mellan två värden framräknade med `pnorm`.

```
pnorm(2.45, 2, 3) - pnorm(2.30, 2, 3)
```

```
## [1] 0.01978986
```

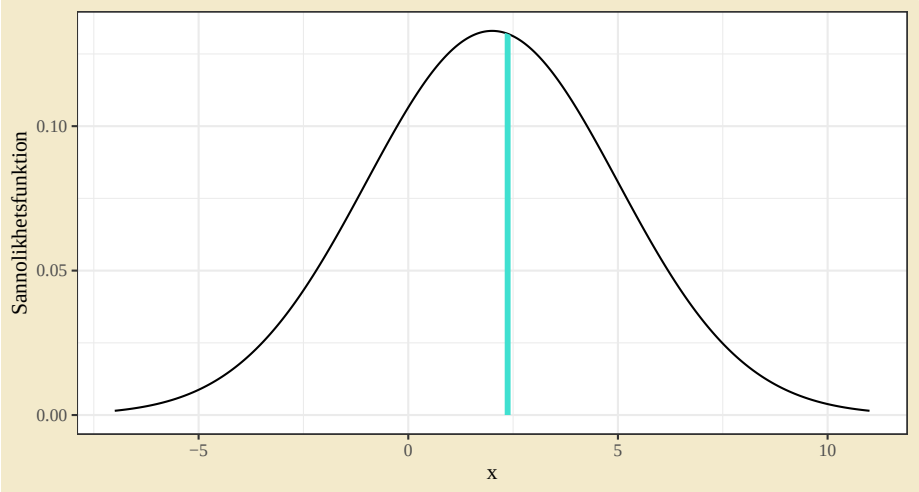
e.

```
1 - pnorm(-0.02, 2, 3)
```

```
## [1] 0.7496324
```

Om man vill illustrera en sannolikhet från en normalfördelning kan man beräkna normalfördelningskurvan med `dnorm` och sedan färglägga en sektion genom ett `geom_ribbon` på filtrerad data. Exempel för (d).

```
dat <- tibble(x = seq(-7, 11, 0.01),
              Sannolikhetsfunktion = dnorm(x, 2, 3))

ggplot(dat, aes(x, Sannolikhetsfunktion)) +
  geom_line() +
  geom_ribbon(aes(ymax = Sannolikhetsfunktion, ymin = 0),
              data = dat %>% filter(x > 2.30 & x < 2.45),
              fill = "turquoise")
```



Den turkosa ytan motsvarar sannolikheten att X ger ett utfall mellan 2.30 och 2.45, sannolikheten uträknad i (d).

Övning 6.4 (Relativ frekvens)**.** Säg att man kastar ett häftstift och att sannolikheten att stiftet stannar med spetsen upp är 0.66. Simulera tusen häftstiftskast och beräkna den relativa frekvensen häftstiften hamnat med spetsen uppåt. Den relativa frekvensen för en serie värden ges av antalet positiva utfall delat på antalet kast för varje kast. Den relativa frekvensen efter 35 kast är till exempel antalet positiva utfall bland de 35, delat på 35. Illustrera med ett linjediagram.

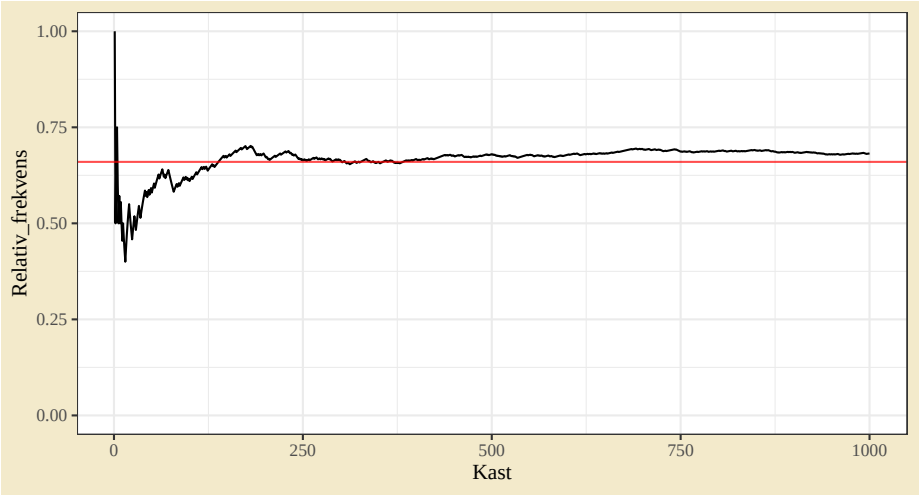
Ledning: tusen kast kan simuleras med `rbin(1000, 1, 0.66)`, vilket skapar tusen slumptal från en binomialfördelning där $n = 1$ och $p = 0.67$.

Lösningsförslag 6.4 (Relativ frekvens)**.** För att beräkna och illustrera den relativa frekvensen skapas en `tibble` där den första kolumnen anger antalet kast vid varje steg, det vill säga en kolumn som anger ett till tusen. Därefter dras tusen slumptal enligt ledningen ovan. Därefter beräknas den *kumulativa summan* - antalet positiva utfall upp till det kast som anges för raden. Och slutligen beräknas den relativa frekvensen genom att ta den kumulativa summan och dela med antalet kast.

```
dat <- tibble(Kast = 1:1000,
              Utfall = rbinom(1000, 1, 0.67),
              Antal_spets_upp = cumsum(Utfall),
              Relativ_frekvens = Antal_spets_upp / Kast)

dat
```

```
## # A tibble: 1,000 x 4
##     Kast Utfall Antal_spets_upp Relativ_frekvens
##    <int>  <int>           <int>            <dbl>
##  1     1      1               1            1
##  2     2      0               1            0.5
##  3     3      1               2            0.667
##  4     4      1               3            0.75
##  5     5      0               3            0.6
##  6     6      0               3            0.5
##  7     7      1               4            0.571
##  8     8      0               4            0.5
##  9     9      1               5            0.556
## 10    10      0               5            0.5
## # ... with 990 more rows
```

```
ggplot(dat, aes(Kast, Relativ_frekvens)) +
  geom_line() +
  geom_hline(yintercept = 0.66, col = "red", alpha = 0.7) +
  ylim(0,1)
```



Den relativa frekvensen stabiliseras efter ett par hundra kast.

Övning 6.5 (Jämförelse mellan binomial och poisson)**.** Poissonfördelning kan ses som en approximation av en binomialfördelning när n är stort och p är litet. Poissonfördelningens parameter λ sätts vid approximation till binomialfördelningens populationsmedelvärde np.

Ta som exempel en binomialfördelning men $n = 10$ och $p = 0.1$; dess väntevärde är $10 \cdot 0.1 = 1$. Beräkna sannolikheter från binomialfördelningen och motsvarande poissonfördelning. Jämför utfallen och illustrera med en passande graf.

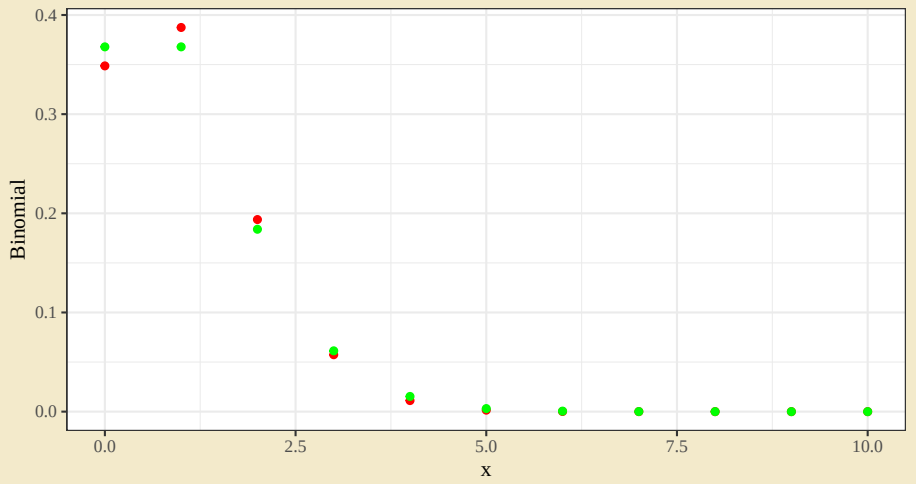
Lösningsförslag 6.5 (Jämförelse mellan binomial och poisson)**.** Fördelningen för binomial och poisson kan tas fram med `dbinom` respektive `dpois`.

```
dat <- tibble(x = 0:10,
              Binomial = dbinom(x, 10, 0.1),
              Poisson = dpois(x, 1),
              Differens = Binomial - Poisson)
dat %>% round(3)
```

```
## # A tibble: 11 x 4
##        x Binomial Poisson Differens
##    <dbl>    <dbl>   <dbl>     <dbl>
##  1     0    0.349   0.368    -0.019
##  2     1    0.387   0.368     0.02
##  3     2    0.194   0.184     0.01
##  4     3    0.057   0.061    -0.004
##  5     4    0.011   0.015    -0.004
##  6     5    0.001   0.003    -0.002
##  7     6    0       0.001     0
##  8     7    0       0         0
##  9     8    0       0         0
```

```
## 10     9    0       0         0
## 11    10    0       0         0
```

```
ggplot(dat, aes(x)) +
  geom_point(aes(y = Binomial), col = "red") +
  geom_point(aes(y = Poisson), col = "green")
```



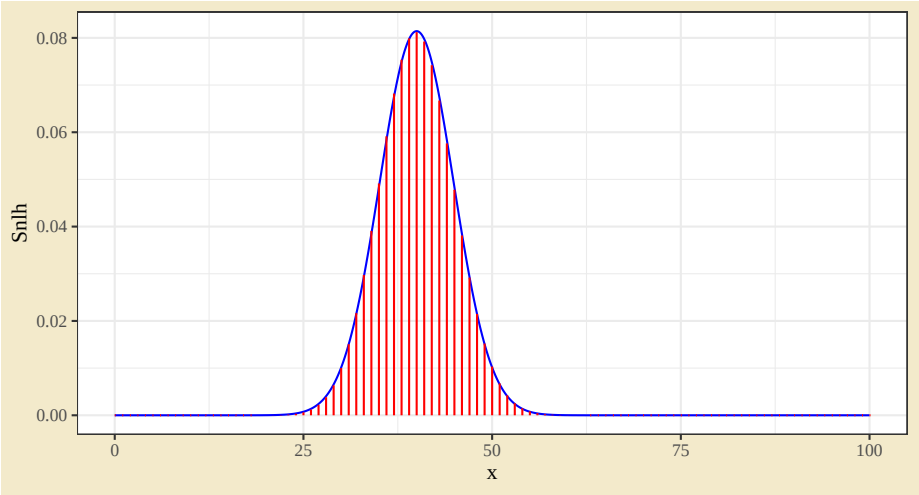
Fördelningarna är ganska lika. Relativt binomialen överskattar poissonfördelningen sannolikheten att få exakt 0 och överskattar sannolikheten att få exakt 1.

Övning 6.6 (Jämförelse mellan binomial och normal)**.** En av många orsaker till att normalfördelningen förekommer i tillämpningar är att både binomialfördelningen och poissonfördelningen kan approximeras med en normalfördelning om populationsmedelvärdet är stort. För att se exempel på detta kan man jämföra en binomialfördelning med en normalfördelning. Skapa en graf för en binomialfördelning med n $= 100$ och p $= 0.4$ med en överliggande normalfördelning med samma populationsmedelvärde och -varians, dvs. populationsmedelvärdet $\mu = np = 100 \cdot 0.4 = 40$ och populationsvariansen $\sigma^2 = n \cdot p \cdot (1 - p) = 100 \cdot 0.4 \cdot 0.6 = 24$.

Lösningsförslag 6.6 (Jämförelse mellan binomial och normal)**.** Funktionerna `dbinom` och `dnorm` kan användas för att ta fram funktionsvärden från binomial- och normalfördelning. Dessa kan sedan plottas i en ggplot genom `geom_segment` för binomialen och `geom_line` för den kontinuerliga normalfördelningen.

```
dat_bin <- tibble(x = 0:100,
                  Snlh = dbinom(x, 100, 0.4))
dat_norm <- tibble(x = seq(0, 100, by = 0.1),
                   Snlh = dnorm(x, mean = 40, sd = sqrt(24)))

ggplot() +
```

```
  geom_line(aes(x, Snlh), data = dat_norm, col = "blue") +
  geom_segment(aes(x = x, xend = x, y = 0, yend =Snlh),
               data = dat_bin, col = "red")
```



Normalkurvan (blå kontinuerlig linje) följer binomalfördelningen (röda staplar). Man kan också jämföra specifika sannolikheter, till exempel

```
# Exakt 35 i binomialen
dbinom(35, 100, 0.4) # 0.0491
```

```
## [1] 0.04913282
```

```
# Mellan 34.5 och 35.5 i normalen
pnorm(35.5, 40, sqrt(24)) - pnorm(34.5, 40, sqrt(24)) # 0.0483
```

```
## [1] 0.04837712
```

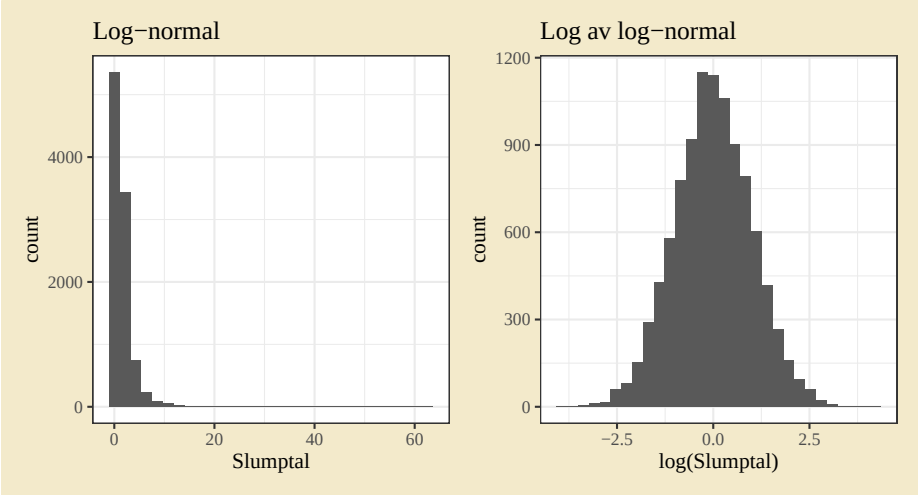
Sannolikheten för exakt 35 i binomialen ligger nära sannolikheten för utfall mellan 34.5 och 35.5 i normalfördelningen.

Övning 6.7 (Log-normal fördelning)**.** Ett exempel på en fördelning som är kontinuerlig, men inte normal, är en log-normal fördelning. En log-normal fördelning defineras av att den ger en normalfördelning efter att den logaritmeras - den är exponentialen av normalfördelningen. Dra 10000 slumptal från en log-normal fördelning (funktionen `rlnorm`) och illustrera med ett histogram. Ta logaritmen av datan och skapa histogrammet på nytt.

Lösningsförslag 6.7 (Log-normal fördelning)**.** Funktionen `rlnorm` skapar slumptal från en log-normal fördelning. Två separata grafer plottar histogram över slumptalen och slumptalen efter log-transform.

```
dat <- tibble(Slumptal = rlnorm(10000))
```

```
g1 <- ggplot(dat, aes(Slumptal)) +
  geom_histogram() +
  ggtitle("Log-normal")

g2 <- ggplot(dat, aes(log(Slumptal))) +
  geom_histogram() +
  ggtitle("Log av log-normal")

library(patchwork)
g1 + g2
```



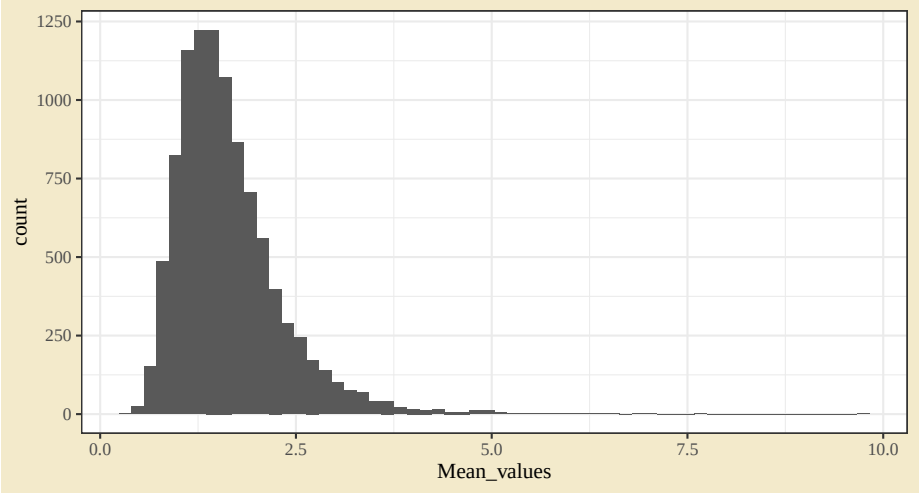
Övning 6.8 (Centrala gränsvärdesatsen)**.** Centrala gränsvärdesatsen ger att summor (och medelvärden) av flera likadana slumpvariabler följer en ungefärlig normalfördelning. Skriv en funktion som drar 10 observation från en log-normal fördelning och beräknar ett medelvärde. Dra 10000 upprepningar från den fördelningen och se om medelvärdena följer en normalfördelning. Gör samma sak men skriv funktionen så att den drar 1000 observationer och tar ett medelvärde.

Se anvisningarna för ett liknande exempel. Observera att detta är en svårare uppgift.

Lösningsförslag 6.8 (Centrala gränsvärdesatsen)**.** En funktion som tar medelvärdet av tio observation skapas. Tiotusen sådana medelvärden beräknas och dessa illustreras med ett histogram.

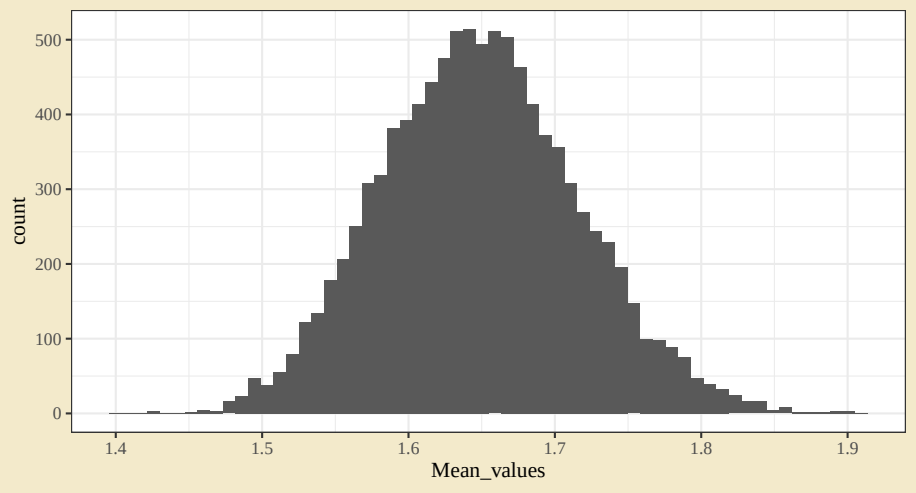
```
mean_of_ten <- function(){
  x <- rlnorm(10)
  mean(x)
}

means <- replicate(10000, mean_of_ten())
```

```
dat <- tibble(Mean_values = means)

ggplot(dat, aes(Mean_values)) +
  geom_histogram(bins = 60)
```



Histogrammet visar en klar skevhet.

```
mean_of_thousand <- function(){
  x <- rlnorm(1000)
  mean(x)
}

means <- replicate(10000, mean_of_thousand())

dat <- tibble(Mean_values = means)

ggplot(dat, aes(Mean_values)) +
  geom_histogram(bins = 60)
```



Histogrammet ligger närmare en typisk normalfördelning.

Övning 6.9 (Multiplicera med två eller addera två)**.** I en övningsuppsgift väcks frågan om det finns någon skillnad mellan att ta utfallet av en slumpvariabel och multiplicera med två, och att addera två slumpvariabler. Ta som exempel tre slumpvariabler X_1, X_2, X_3 och låt dem alla följa en binomialfördelning med $n = 100$ och $p = 0.8$. Dra 10000 slumptal från respektive fördelning, beräkna $2 \cdot X_1$ och $X_2 + X_3$ och illustrera dessa beräknade variabler. Beräkna också väntevärde och standardavvikelse från de beräknade variablerna.

Lösningsförslag 6.9 (Multiplicera med två eller addera två)**.** Tre variabler med slumptal konstrueras med `rbinom`. Därefter skapas två nya variabler, en genom summan av två av slumpvariablerna och en som den tredje slumpvariabeln multiplicerad med 2.

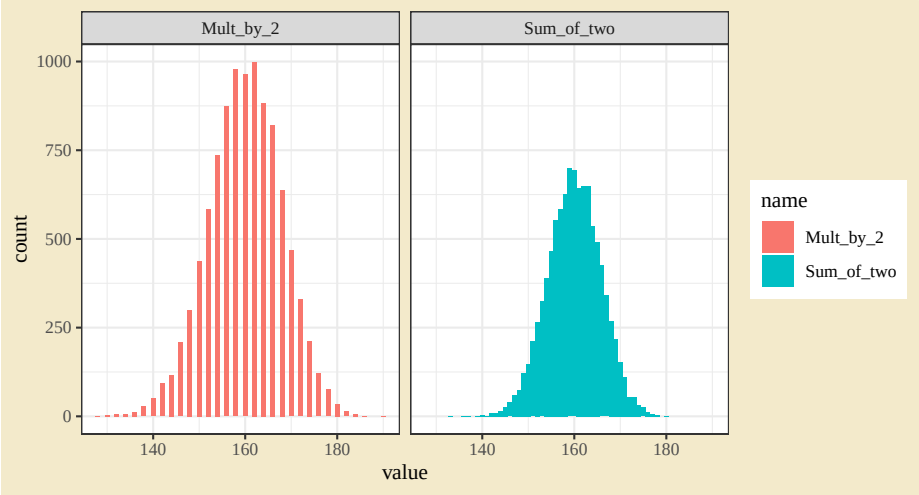
```
dat <- tibble(X1 = rbinom(10000, size = 100, prob = 0.8),
              X2 = rbinom(10000, size = 100, prob = 0.8),
              X3 = rbinom(10000, size = 100, prob = 0.8)) %>%
  mutate(Mult_by_2 = 2 * X1,
         Sum_of_two = X2 + X3)

dat %>%
  summarise_all(mean)
```

```
## # A tibble: 1 x 5
##      X1    X2    X3 Mult_by_2 Sum_of_two
##   <dbl> <dbl> <dbl>     <dbl>      <dbl>
## 1  80.0  80.0  80.0      160.       160.
```

```
dat %>%
  summarise_all(sd)
```

```
## # A tibble: 1 x 5
##      X1    X2    X3 Mult_by_2 Sum_of_two
##   <dbl> <dbl> <dbl>     <dbl>      <dbl>
## 1  3.97  4.04  4.05      7.93       5.76
```

```
dat %>%
  select(Mult_by_2, Sum_of_two) %>%
  pivot_longer(1:2) %>%
  ggplot(aes(value, fill = name)) +
  geom_bar() +
  facet_wrap(~ name)
```



Stapeldiagrammet för fallet med en variabel multiplicerad med två ger högre staplar och större spridning än stapeldiagrammet med två summerade variabler. Diagrammet visar också hur vissa utfall inte kan inträffa i fallet med multiplikation, eftersom bara jämna utfall kan inträffa. Intuitivt kan man förstå den mindre spridningen för två summerade variabler med att ovanligt låga eller höga värden ofta kommer kvittas ut mot ett *vanligt* värde.

Skillnaden märks också i beräkning av standardavvikelse. Den ursprungliga variabeln har en varians på $np(1-p) = 100 \cdot 0.8 \cdot 0.2 = 16$ och därmed en standardavvikelse på $\sqrt{16} = 4$. Multiplikation med två ger en slumpvariabel med den dubbla standardavvikelsen, alltså 8, medan addition av två likadana slumpvariabler ger en lägre standardavvikelse. För summor gäller att *variansen av summan är summan av varianserna* (detta antar dock oberoende slumpvariabler) - här fås alltså att variansen av summan är $16 + 16 = 32$ och att standardavvikelsen är $\sqrt{32} = 5.6569$. Värdet i datan kommer förstås avvika något från den teoretiska beräkningen, eftersom det är slumptal.

Kapitel 7

Ett stickprov

7.1 Normalfördelad data (eller stora stickprov)

Om man har normalfördelad data och vill testa om datans medelvärde är skilt från något hypotetiskt värde μ_0 kan man använda ett t-test för ett stickprov. Testets hypoteser ges i det tvåsidiga fallet av

$$H_0 : \mu = \mu_0 \qquad H_1 : \mu \neq \mu_0.$$

Testet kan beräknas genom att beräkna testvärdet

$$t = \frac{\bar{x} - \mu_0}{s/\sqrt{n}}$$

och beräkna ett p-värde som sannolikheten i svansarna bortom det t-värdet, i en t-fördelning med $n - 1$ frihetsgrader.

Som exempel ges följande data på 8 observationer av havreskörd.

Man vill testa om skörden är skild från 50, så hypoteser ges av

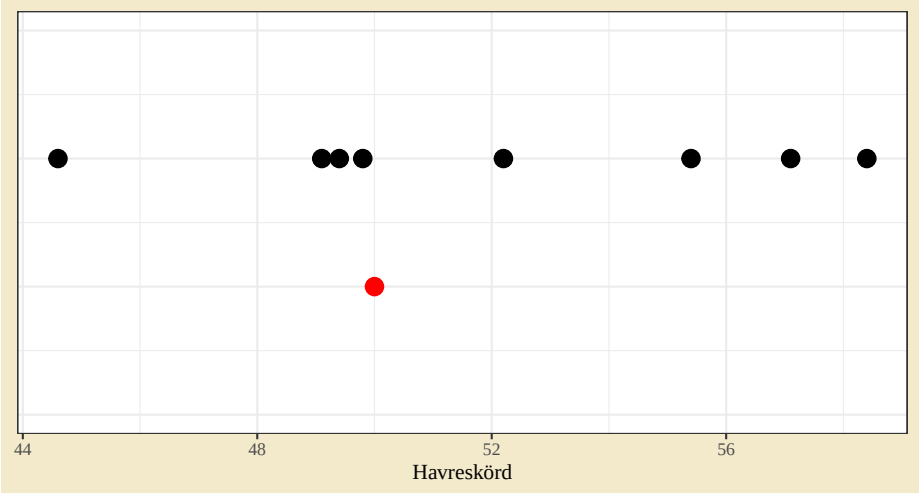
$$H_0 : \mu = 50 \qquad H_1 : \mu \neq 50.$$

Innan man utför testet kan det vara bra att ta en titt på datan och bilda en första uppfattning om nollhypotesens rimlighet. Man kan rita varje observation som en punkt i ett punktdiagram. Om det kan finnas överlappande punkter kan

49.8	58.4	49.4	57.1
52.2	49.1	44.6	55.4

`geom_dotplot` eller `geom_count` användas för att separera eller notera överlappningar.

```
dat <- c(49.8, 58.4, 49.4, 57.1, 52.2, 49.1, 44.6, 55.4)
dat_t <- tibble(x = dat)

ggplot(dat_t, aes(x, y = 0)) +
  geom_point(size = 4) +
  annotate("point", x = 50, y = -1, col = "red", size = 4) +
  ylim(-2,1) + xlab("Havreskörd") +
  theme(axis.text.y = element_blank(),
        axis.ticks.y = element_blank(),
        axis.title.y = element_blank())
```



Det skattade medelvärdet är $\bar{x} = 52.0$ och standardavvikelsen $s = 4.68$. Teststorheten kan beräknas till

$$t = \frac{52 - 50}{4.68/\sqrt{8}} = 1.209$$

Testen kan genomföras i R med funktionen `t.test`.

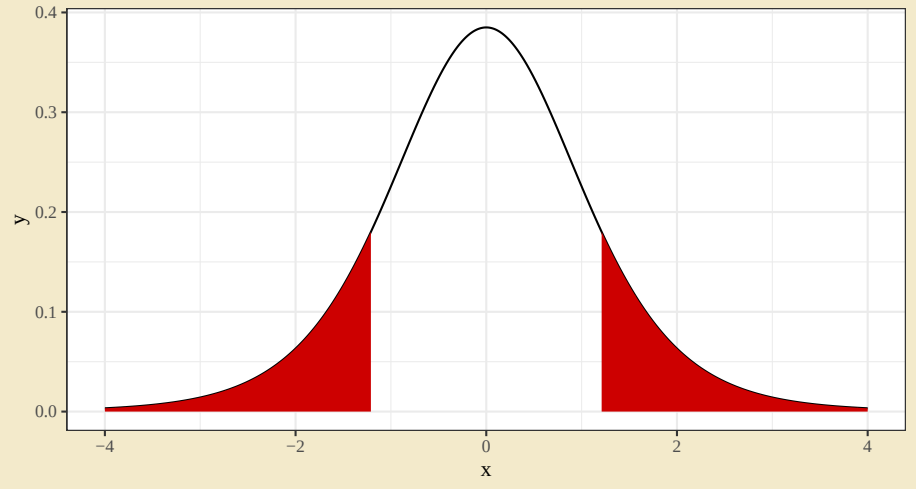
```
dat <- c(49.8, 58.4, 49.4, 57.1, 52.2, 49.1, 44.6, 55.4)
t.test(dat, mu = 50)
```

```
##
##  One Sample t-test
##
## data:  dat
## t = 1.2086, df = 7, p-value = 0.266
## alternative hypothesis: true mean is not equal to 50
```

```
## 95 percent confidence interval:
##  48.08713 55.91287
## sample estimates:
## mean of x
##        52
```

I funktionen anges datan `dat` och nollhypotesens värde `mu = 50`. Utskriften ger t-värdet till 1.2086 och p-värdet till 0.266. Det höga p-värdet ger att man inte förkastar nollhypotesen. Beräkningen av p-värdet kan illustreras med en t-fördelning.

```
test_results <- t.test(dat, mu = 50)

dat_t <- tibble(x = seq(-4, 4, 0.01),
                y = dt(x, 7))

ggplot(dat_t, aes(x, y)) +
  geom_line() +
  geom_ribbon(aes(ymax = y, ymin = 0), fill = "red3",
              data = dat_t %>% filter(x < -test_results$statistic)) +
  geom_ribbon(aes(ymax = y, ymin = 0), fill = "red3",
              data = dat_t %>% filter(x > test_results$statistic))
```



Notera hur man kan spara utfallet av `t.test` och sedan använda resultat direkt ur det objektet (här genom `test_restults$statistic` som ger teststorheten, men även `test_results$p.value` kan vara användbart). Testobjektets namn kan förstås vara något helt annat än `test_results`.

Den röda arean i svansarna motsvarar p-värdet 0.266. Sannolikheten att få den observerade skillnaden mellan skattat medelvärde och nollhypotesens värde (52 respektive 50) även om det inte finns någon verklig skillnad är alltså ungefär en

på fyra.

Utskriften från `t.test` ger också ett 95-procentigt konfidensintervall: $(48.09, 55.91)$. Tolkningen är att det sanna populationsmedelvärdet ligger i intervallet med 95 procents konfidens. Notera att nollhypotesens värde 50 ligger i intervallet.

7.2 Binär data. Proportioner

En skattad proportion kan ställas mot en nollhypotes genom ett *binomialtest* eller *z-test*. Ta som exempel att man samlar data om antal infekterade plantor i ett slumpmässigt urval av 50 och finner att 17 är infekterade. Man kan då testa nollhypotesen att den sanna populationsproportionen är 0.5,

$$H_0 : p = 0.5 \qquad H_1 : p \neq 0.5.$$

I R kan ett binomialtest utföras med `binom.test` och ett z-test (som bygger på en normalapproximation av binomialfördelningen) med `prop.test`. Det senare kan antingen utföras med eller utan en *kontinuitetskorrektion*. Det test som beräknas för hand är oftast utan korrektionen.

```
binom.test(17, 50, p = 0.5)
```

```
##
##  Exact binomial test
##
## data:  17 and 50
## number of successes = 17, number of trials = 50, p-value = 0.03284
## alternative hypothesis: true probability of success is not equal to 0.5
## 95 percent confidence interval:
##  0.2120547 0.4876525
## sample estimates:
## probability of success
##                   0.34
```

```
prop.test(17, 50, p = 0.5, correct = F)
```

```
##
##  1-sample proportions test without continuity correction
##
## data:  17 out of 50, null probability 0.5
## X-squared = 5.12, df = 1, p-value = 0.02365
## alternative hypothesis: true p is not equal to 0.5
## 95 percent confidence interval:
##  0.2243695 0.4784617
## sample estimates:
##    p
```

```
## 0.34
```

```
prop.test(17, 50, p = 0.5, correct = T)
```

```
##
##  1-sample proportions test with continuity correction
##
## data:  17 out of 50, null probability 0.5
## X-squared = 4.5, df = 1, p-value = 0.03389
## alternative hypothesis: true p is not equal to 0.5
## 95 percent confidence interval:
##  0.2159464 0.4885541
## sample estimates:
##    p
## 0.34
```

Funktionernas argument är antal postiva utfall, det totala antalet utfall, och nollhypotesens värde. För `prop.test` anger argumentet `correct` om en korrektion ska utföras eller ej. De tre testen ger liknande resultat, med p-värden kring 3 procent. Ett test på femprocentsnivån skulle alltså ge att nollhypotesen förkastas.

Utskrifterna ger också konfidensintervall, och precis som p-värden beror intervallet på fördelning och kontinuitetskorrektion. Tolkningen av intervallet är detsamma som i fallet med kontinuerlig data - den sanna populationsproportionen ligger i intervallet med 95 procents konfidens - men den exakta konstruktionen varierar beroende på detaljer i antaganden. För ett konfidensintervall som motsvarar det man vanligen beräknar för hand kan man använda `binomial`-paketet och funktionen `binom.confint`.

```
library(binom)
binom.confint(17, 50, methods = "asymp")
```

```
##       method  x  n mean    lower    upper
## 1 asymptotic 17 50 0.34 0.208697 0.471303
```

Det interval som ges är kan beräknas genom att sätta $\hat{p} = 17/50$ i uttrycket

$$\hat{p} \pm 1.96\sqrt{\frac{\hat{p}(1-\hat{p})}{50}},$$

med ett litet avrundningsfel eftersom 1.96 är avrundat.

7.3 Nominal eller ordinal data. Goodness-of-fit

Goodness-of-fit-test testar fördelningen i insamlad data med en hypotetiskt fördelning. Teststorheten ges av

Art	Antal
Ladusvala	237
Hussvala	220
Backsvala	143

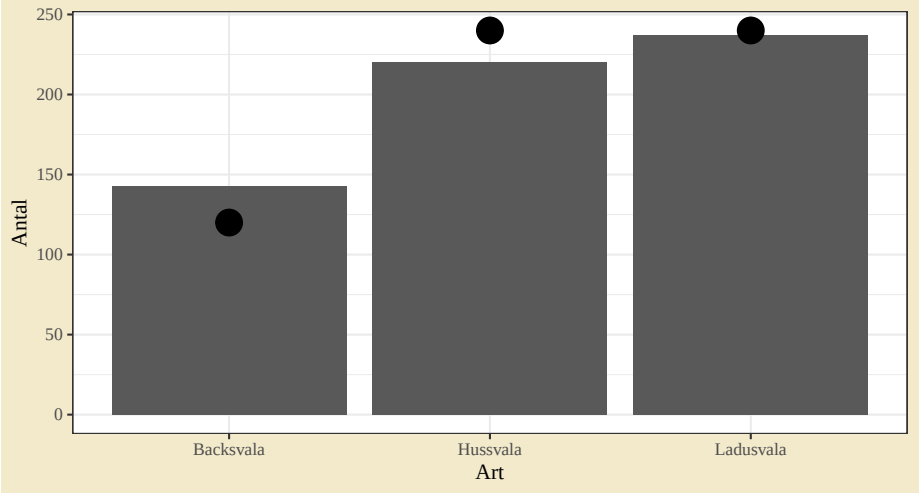
$$\chi^2 = \sum \frac{(O-E)^2}{E}$$

där summan går över alla utfall i fördelning, O är det observerade antalet i klassen och E är det förväntade antalet enligt den hypotetiska fördelningen. Under nollhypotesen följer teststorheten en χ^2-fördelning.

Som exempel ges följande data på fågelobservationer.

Totalt har man observerat 600 fåglar. Från tidigare års studier tror man att Ladusvalor och Hussvalor är lika vanliga, medan Backsvalar förekommer hälften så ofta - proportionerna skulle alltså vara 0.4, 0.4 respektive 0.2. Detta kan illustreras med ett stapeldiagram, med den hypotetiska fördelningen som punkter.

```
dat <- tibble(Art = c("Ladusvala", "Hussvala", "Backsvala"),
              Antal = c(237, 220, 143))

ggplot(dat, aes(Art, Antal)) +
  geom_bar(stat = "identity") +
  geom_point(aes(y = c(0.4, 0.4, 0.2) * 600), size = 6)
```



Grafen visar att hussvalor förekommer något mer sällan än väntat, medan backsvalor förekommer någon oftare.

Utfall	Frekvens
0	21
1	76
2	125
3	107
4	79
5	54
6	23
7	15

För att genomföra testet i R används `chisq.test`. Funktionens ingångsvärden är den observerade datan och de teoretiska andelarna.

```
chisq.test(dat$Antal, p = c(0.4, 0.4, 0.2))
```

```
##
##  Chi-squared test for given probabilities
##
## data:  dat$Antal
## X-squared = 6.1125, df = 2, p-value = 0.04706
```

Utskriften ger teststorheten $\chi^2 = 6.1125$ och p-värdet 0.04706. I detta fall är p-värdet strax under fem procent - man skulle alltså förkasta nollhypotesen på signifikansnivån fem procent.

Utskriften ger också antalet frihetsgrader till två. I det här fallet ges den hypotetiska fördelningen av antagna sannolikheter, antalet frihetsgrader ges därför av antalet klasser minus ett. Men man kan också använda goodness-of-fit-test för att testa om data följer en viss typfördelning vars parameter beror på värden i data. Ett typiskt exempel på detta är test om data följer en poissonfördelning där man använder datan för att skatta λ-parametern i fördelningen. Antal frihetsgrader blir i sådana fall antalet klasser, minus antalet skattade parametrar, minus ett. Funktionen `chisq.test` kommer då ge fel antal frihetsgrader - p-värdet kan istället beräknas genom `pchisq`.

Ta som exempel följande frekvenstabell, beräknad från 500 observationer av en diskret variabel.

```
dat <- tibble(Utfall = 0:7,
              Frekvens = c(21,76,125,107,79,54,23,15))
kable(dat)
```

Målet är att testa om datan följer en poissonfördelning. För att beräkna sannolikheter från en poissonfördelning behövs en skattning av parametern λ. Detta ges av medelvärdet av de 600 observationerna, vilket kan beräknas genom att multiplicera utfallet med frekvensen, summera, och dela på 600.

```
lambda <- sum(dat$Utfall * dat$Frekvens) / 500
lambda
```

```
## [1] 2.952
```

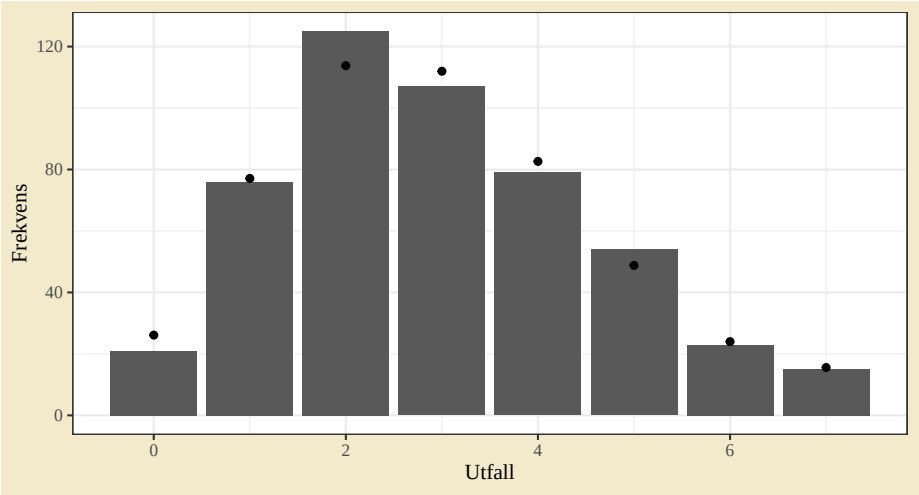
Skattningen på λ är 2.952. Nästa steg är att beräkna sannolikheter från en poissonfördelning med det parametervärdet. För att sannolikheterna ska summera till ett justeras sannolikheten för det avslutande utfallet 7.

```
dat$Sannolikheter <- dpois(0:7, lambda)
dat$Sannolikheter[8] <- 1 - ppois(6, lambda)
sum(dat$Sannolikheter)
```

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

Data och nollhypotes kan illustreras med ett stapeldiagram där punkter anger förväntade värden från nollhypotesen.

```
ggplot(dat, aes(Utfall, Frekvens)) +
  geom_bar(stat = "identity") +
  geom_point(aes(y = Sannolikheter * 500))
```



Data och nollhypotes verkar stämma ganska väl. Funktionen `chisq.test` kan användas för att beräkna teststorheten, men observera att antalet frihetsgrader nu blir fel - datan har åtta klasser, och då en parameter skattas i beräkningen av förväntade antal bör testet genomföras med $8 - 1 - 1 = 6$ frihetsgrader. För att korrigera kan man ta fram teststorheten och beräkna p-värdet ur den korrekta fördelningen.

```
chisq.test(dat$Frekvens, p = dat$Sannolikheter)
```

```
##
##  Chi-squared test for given probabilities
```

1054.8	1052.9	1051.0	1049.8	1051.6
1047.9	1051.8	1048.5	1040.2	1050.7

```
##
## data:  dat$Frekvens
## X-squared = 3.122, df = 7, p-value = 0.8735
```

```
test_result <- chisq.test(dat$Frekvens, p = dat$Sannolikheter)
1 - pchisq(test_result$statistic, df = 6)
```

```
## X-squared
## 0.7933799
```

I ett χ^2-test är p-värdet sannolikheten i svansen till höger om teststorheten. Eftersom `pchisq`-funktionen beräknar sannolikheten till vänster, kan man ta komplementet genom att dra ifrån sannolikheten från ett.

7.4 Övningar

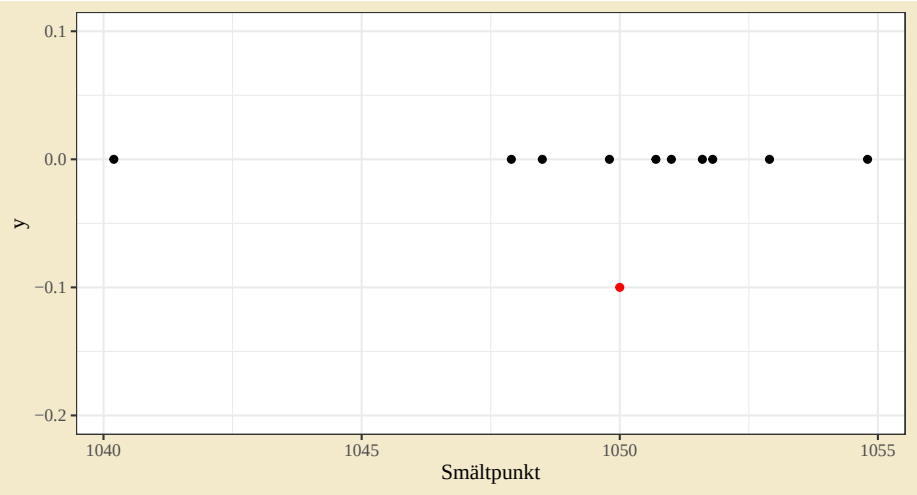
Övning 7.1 (Smältpunkt)**.** Någon mäter smältpunkter för legeringar av metall och får följande värden.

a. Beräkna ett 95-procentigt konfidensintervall för medelvärdet.

b. Genomför ett hypotestest för att testa om medelsmältpunkten är 1050 grader.

Lösningsförslag 7.1 (Smältpunkt)**.** Datan kan importeras från excelfilen med uppgiftsdata.

```
dat <- readxl::read_excel("Data/Uppgiftsdata.xlsx", sheet = "Smältpunkt")

ggplot(dat, aes(Smältpunkt, 0)) +
  geom_point() +
  annotate("point", x = 1050, y = -0.1, col = "red") +
  ylim(-0.2,0.1)
```




```
t.test(dat$Smältpunkt, mu = 1050)
```

```
##
##  One Sample t-test
##
## data:  dat$Smältpunkt
## t = -0.063821, df = 9, p-value = 0.9505
## alternative hypothesis: true mean is not equal to 1050
## 95 percent confidence interval:
##  1047.084 1052.756
## sample estimates:
## mean of x
##   1049.92
```

I bilden är varje svart punkt en observation och den röda punkten är nollhypotesens värde.

Testet ger ett högt p-värde, vilket tyder på att nollhypotesen inte bör förkastar. Ett 95-procentigt konfidensintervall ges av $(1047.08, 1052.76)$. Notera att det täcker nollhypotesens värde på 1050.

Övning 7.2 (Väderstation Falsterbo. Test och konfidensintervall)**.** Bland kursdatan finns en fil med temperaturmätningar från Falsterbo. Datan kan läsas in med följande kod.

```
dat <- read_csv2("Data/smhi-opendata_1_52230_20210912_114534.csv", skip = 9) %>%
  mutate(Lufttemperatur = as.numeric(Lufttemperatur))
```

Medeltemperatur 2000-2009 ges av följande tabell.

```
dat_temp <- dat %>%
  mutate(År = lubridate::year(Datum)) %>%
```

År	Medeltemperatur
2000	9.72
2001	8.98
2002	9.51
2003	8.89
2004	8.92
2005	9.01
2006	9.63
2007	9.80
2008	9.81
2009	9.17

```
  filter(År %in% 2000:2009) %>%
  group_by(År) %>%
  summarise(Medeltemperatur = mean(Lufttemperatur)) %>%
  mutate(Medeltemperatur = round(Medeltemperatur, 2))

kable(dat_temp)
```

a. Konstruera och tolka ett 95-procentigt konfidensintervall för medeltemperaturen under perioden.

b. Mätningarna från 1900-1949 ger en medeltemperatur på 8.42. Genomför ett passande t-test för att se om mätningarna från 2000-talet skiljer sig i medelvärde.

c. Titta på antalet mätningar per dag. Diskutera möjlig påverkan på testet i (b).

Lösningsförslag 7.2 (Väderstation Falsterbo. Test och konfidensintervall)**.** a. Konfidensvallet kan tas fram med `t.test`. Den aggregerade datan med årsmedelvärden sparades tidigare som `dat_temp`.

```
t.test(dat_temp$Medeltemperatur)
```

```
##
##  One Sample t-test
##
## data:  dat_temp$Medeltemperatur
## t = 76.691, df = 9, p-value = 5.514e-14
## alternative hypothesis: true mean is not equal to 0
## 95 percent confidence interval:
##  9.06838 9.61962
## sample estimates:
## mean of x
##     9.344
```

Intervallet ges av $(9.06, 9.62)$. Populationsmedelvärdet för 2000-2009 ligger med 95-procents konfidens i det intervallet.

b. Funktionen `t.test` kan användas med nollhypotesens värde angivet med argumentet `mu`. Nollhypotesen är att populationsmedelvärdet μ är 8.42.

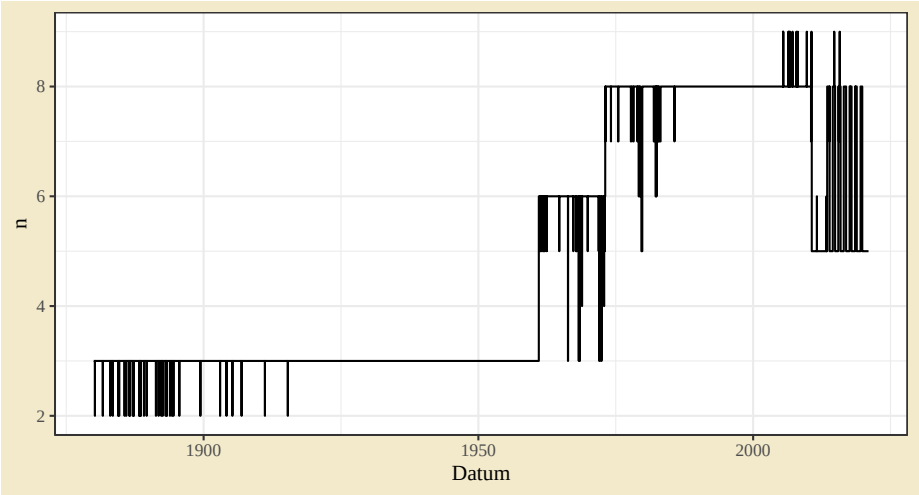
```
t.test(dat_temp$Medeltemperatur, mu = 8.42)
```

```
##
##  One Sample t-test
##
## data:  dat_temp$Medeltemperatur
## t = 7.5837, df = 9, p-value = 3.383e-05
## alternative hypothesis: true mean is not equal to 8.42
## 95 percent confidence interval:
##  9.06838 9.61962
## sample estimates:
## mean of x
##     9.344
```

Det låga p-värdet ger att nollhypotesen förkastas, vilket tyder på att medeltemperaturen 2000-2009 är skild från 8.42.

c. Man kan räkna antal observationer per dag med `count` och göra en linjegraf med `ggplot` och `geom_line`.

```
dat %>% count(Datum) %>% ggplot(aes(Datum, n)) + geom_line()
```



Antal observationer per dag är inte konstant. Om vissa dagar är uppmätta vid tider på dygnet då det är kallare eller varmare kan det ge en skevhet i mätningarna, vilket gör att jämförelsen mellan olika tider inte är rättvis. En lösning skulle kunna vara att ta värdet en viss tid på dygnet, en tid då det

Träd	Före	Efter	Diff
A	271.1	311.2	40.1
B	307.9	285.4	-22.5
C	305.9	320.9	15.0
D	302.6	353.5	50.9
E	263.4	358.0	94.6
F	277.7	286.1	8.4

finns observationer för samtliga dagar. Det finns förstås också andra möjliga felkällor som påverkar en jämförelse över tid, t.ex. ändrade mätinstrument, små positionsförändringar, förändringar i kringliggande bebyggelse).

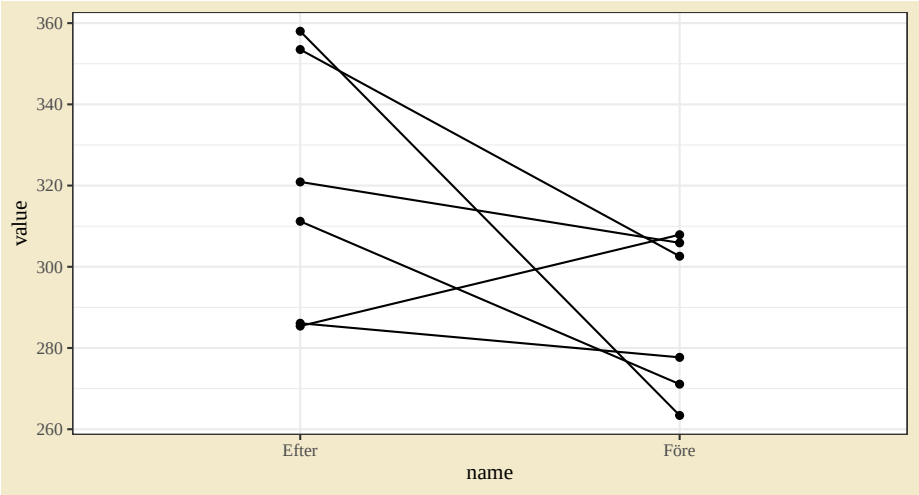
Övning 7.3 (Äppelträd)**.** I en undersökning av insektsskador på äppelträd har 6 plantor blivit vägda före och efter insektsdödande behandling. Syftet är att undersöka om behandlingen leder till förändrad vikt. Resultatet ges av följande tabell.

a. Gör ett test för att se om medelvärdet för behandling är skilt från 310.

b. Beräkna ett konfidensintervall för skillnaden i populationsmedelvärden.

Lösningsförslag 7.3 (Äppelträd)**.** Datan importeras från excelfilen med uppgiftsdata. Datan ändras till långform genom `pivot_longer` och plottas med `ggplot`.

```
dat <- readxl::read_excel("Data/Uppgiftsdata.xlsx", sheet = "Äppelträd")

dat %>%
  pivot_longer(-Träd) %>%
  ggplot(aes(name, value, group = Träd)) +
  geom_point() +
  geom_line()
```



a.

```
t.test(dat$Före, mu = 310)
```

```
##
##  One Sample t-test
##
## data:  dat$Före
## t = -2.733, df = 5, p-value = 0.04113
## alternative hypothesis: true mean is not equal to 310
## 95 percent confidence interval:
##  267.5012 308.6988
## sample estimates:
## mean of x
##     288.1
```

Testet ger ett p-värde på 0.041. Nollhypotesen förkastas på femprocentsnivån men ej på enprocentsnivån.

b.

```
t.test(dat$Före, dat$Efter, paired = T)
```

```
##
##  Paired t-test
##
## data:  dat$Före and dat$Efter
## t = -1.8855, df = 5, p-value = 0.118
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -73.45961  11.29295
## sample estimates:
```

```
## mean of the differences
##               -31.08333
```

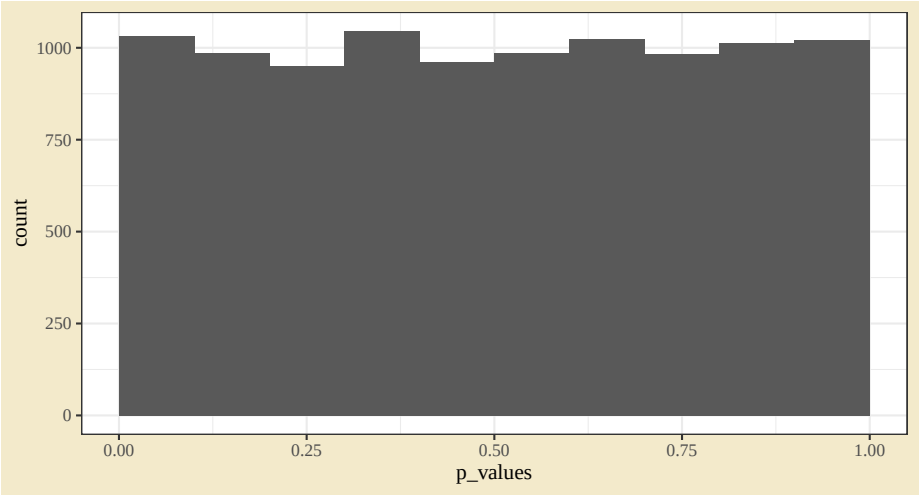
Konfidensintervallet ges av $(-73.46, 11.29)$. Den sanna behandlingsskillnaden ligger med 95 procents konfidens i det intervallet.

Övning 7.4 (Simulering, hypotestest)**.** Slumptal kan användas för att undersöka hypotestestens egenskaper. Skriv en funktion som genererar 100 slumptal från en normalfördelning med populationsmedelvärdet 0, testar mot nollhypotesen $\mu_0 = 0$ i ett t-test, och ger ut p-värdet från det testet. Generera 10000 körningar av funktionen och illustrera p-värdena med ett histogram.

Upprepa samma procedur men dra denna gång slumptal från en normalfördelning med populationsmedelvärdet 0.1 (använs samma nollhypotes som tidigare $\mu_0 = 0$). Hur stor andel av de 10000 simuleringarna resulterar i ett p-värde under 0.05?

Lösningsförslag 7.4 (Simulering hypotestest)**.** Här konstrueras en funktion som drar hundra slumpvärden, testar om medelvärdet är skilt från noll, och ger ut p-värdet från testet. Funktionen replikeras 10000 gånger och de resulterande p-värdena illustreras med ett histogram.

```
p_values_when_mean_0 <- function(){
  x <- rnorm(100, mean = 0)
  test <- t.test(x, mu = 0)
  test$p.value
}

p_values <- replicate(10000, p_values_when_mean_0())

dat <- tibble(p_values)
ggplot(dat, aes(p_values)) +
  geom_histogram(breaks = seq(0, 1, 0.1))
```

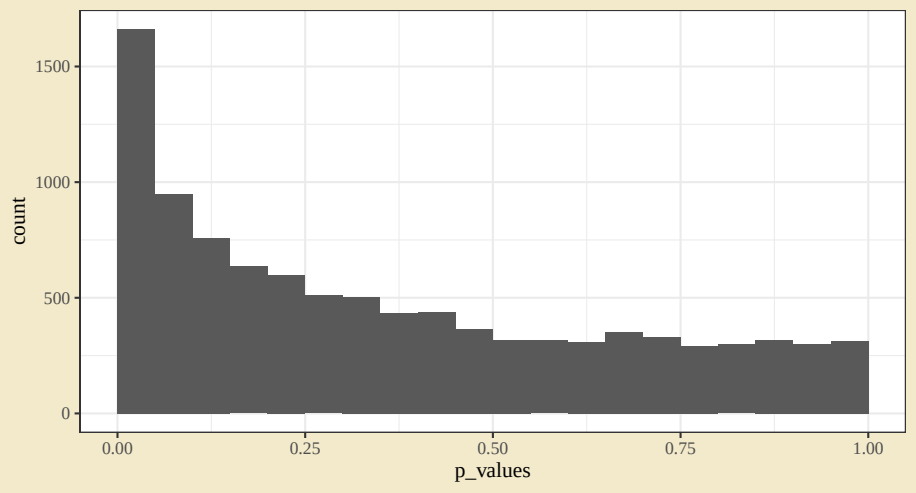



```
mean(p_values < 0.05)
```

```
## [1] 0.0475
```

När slumptalen genereras från en normalfördelning med medelvärde 0 följer p-värdena en likformig fördelning. Ganska exakt 5 procent av simuleringarna ger ett p-värde under 5 procent. Detta är simuleringar i situationen att nollhypotesen är sann - sannolikheten att felaktigt förkasta på femprocentsnivån när nollhypotesen stämmer är alltså fem procent. Detta är alltså risken för falska positiva resultat.

```
p_values_when_mean_0 <- function(){
  x <- rnorm(100, mean = 0.1)
  test <- t.test(x, mu = 0)
  test$p.value
}

p_values <- replicate(10000, p_values_when_mean_0())

dat <- tibble(p_values)
ggplot(dat, aes(p_values)) +
  geom_histogram(breaks = seq(0, 1, 0.05))
```




```
mean(p_values < 0.05)
```

```
## [1] 0.1661
```

När slumptalen istället genereras från en normalfördelning med medelvärde 0.1 följer p-värdena en avtagande kurva men högst sannolikheter kring noll. Runt 16 procent av simuleringarna har givit p-värden under 0.05 - man har alltså en större chans att korrekt förkasta nollhypotesen när den inte stämmer. Notera att man ändå har ganska låg sannolikhet att förkasta nollhypotesen.

Övning 7.5 (Guldfiskgenetik)**.** En teori inom genetik förutsäger att tre fjärdedelar i en grupp guldfiskar ska ha genomskinliga fjäll. Observationer ger att nittio av hundra har genomskinliga fjäll. Genomför ett test för att se om den faktiska proportionen skiljer sig från 0.75.

Lösningsförslag 7.5 (Guldfiskgenetik)**.** Frågan kan hanteras med ett χ^2-test, ett z-test eller ett binomialtest. Här ges exempel på det första. Nollhypotesen är att sannolikheten för genomskinliga fjäll är tre fjärdedelar. Notera att funktionen anges både med antalet (och andelen under nollhypotesen) med genomskinliga fjäll och antalet med färglagda fjäll.

```
chisq.test(c(90, 10), p = c(0.75, 0.25))
```

```
##
##  Chi-squared test for given probabilities
##
## data:  c(90, 10)
## X-squared = 12, df = 1, p-value = 0.000532
```

Det låga p-värdet tyder på att den sanna sannolikheten för genomskinliga fjäll inte är 0.75.

Övning 7.6 (Mer guldfiskgenetik)**.** En konkurrerande teori inom genetik förut-

säger att femton sextondelar (proportionen 0.9375) ska ha genomskinliga fjäll. Observationer ger att nittio av hundra har genomskinliga fjäll. Genomför ett test för att se om proportionen skiljer sig från 0.9375.

Lösningsförslag 7.6 (Mer guldfiskgenetik)**.** Frågan kan hanteras med ett χ^2-test. Nollhypotesen är att sannolikheten är genomskinliga fjäll är femton sextondelar.

```
chisq.test(c(90, 10), p = c(15/16, 1/16))
```

```
##
##  Chi-squared test for given probabilities
##
## data:  c(90, 10)
## X-squared = 2.4, df = 1, p-value = 0.1213
```

Nollhypotesen att sannolikheten för genomskinliga fjäll är femton sextondelar, $p = 15/16$, kan ej förkastas, eftersom testet ger ett p-värde över de vanliga signifikansnivåerna.

Övning 7.7 (Simulering konfidensintervall)**.** Konfidensinterval konstrueras så att det sanna parametervärdet ingår i intervallet med en viss konfidensgrad (oftast 95 procent). Skriv en funktion som drar 10 slumptal från en normalfördelning med medelvärdet 9 och beräknar konfidensintervall. Upprepa funktionen 10000 gånger. Hur ofta täcker intervallet 9?

Notera att detta är en svårare uppgift.

Lösningsförslag 7.7 (Simulering konfidensintervall)**.** Här konstrueras en funktion som drar 10 slumptal och beräknar ett konfidensintervall. Eftersom funktionen ger två värden (intervallets lägre och övre gräns) behövs lite mer hantering än i tidigare exempel (där de funktioner som använts givit ett utfallsvärde).

```
ci_from_ten <- function(){
  x <- rnorm(10, mean = 9)
  test <- t.test(x)
  test$conf.int
}

intervals <- replicate(10000, ci_from_ten())
intervals <- t(intervals) # Transponera till kolumner
intervals <- as.data.frame(intervals)
names(intervals) <- c("Undre", "Övre")

mean(intervals$Undre > 9)
```

```
## [1] 0.029
```

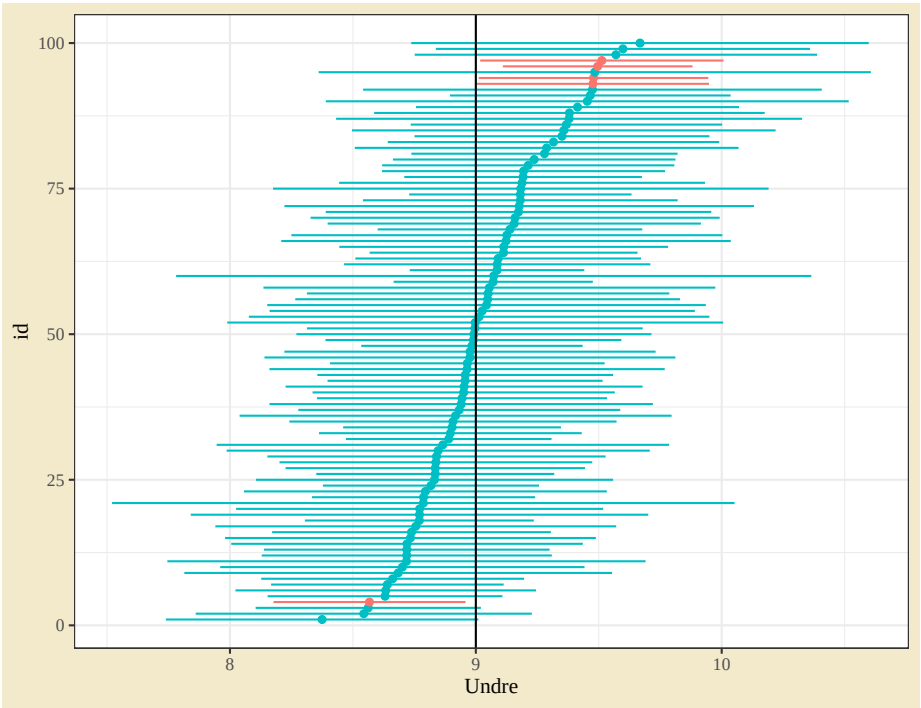
```
mean(intervals$Övre < 9)
```

```
## [1] 0.023
```

Värdet 9 ligger under den övre gränsen ungefär 2.5 procent av gångerna och över den undre gränsen ungefär 2.5 gångerna. Intervallet täcker alltså 9 ungefär 95 procent av gångerna.

En illustration med 100 simulerade fall.

```
intervals <- replicate(100, ci_from_ten())
intervals <- t(intervals) # Transponera till kolumner
intervals <- as.data.frame(intervals)
names(intervals) <- c("Undre", "Övre")

intervals %>%
  arrange(Undre + Övre) %>%
  mutate(id = 1:n(),
         Täcker_9 = Undre < 9 & Övre > 9) %>%
  ggplot(aes(x = Undre, xend = Övre, y = id, yend = id, col = Täcker_9)) +
  geom_segment() +
  geom_point(aes(x = (Övre + Undre) / 2, id)) +
  geom_vline(xintercept = 9) +
  theme(legend.position = "none")
```



Varje streck motsvarar konfidensintervallet från en simulering. Intervallet väntas täcka 9 i 95 av hundra fall, men det kan förstås variera eftersom det beror på

slumptalen.

Kapitel 8

Två stickprov

8.1 Normalfördelad data (eller stora stickprov)

t-test för två stickprov används för att jämföra två grupper och se om de har samma populationsmedelvärde i någon insamlad utfallsvariabel. Det finns två specifika fall: t-test för matchade stickprov, där det finns en parvis koppling mellan de två stickproven, t.ex. att de är mätningar på två syskon; och t-test för oberoende stickprov, där det saknas en sådan koppling mellan stickproven.

8.1.1 t-test för två matchade stickprov

Vid matchade stickprov kan varje observation i en behandlingsgrupp paras med en observation i den andra gruppen. Själva testet är ett t-test för *ett* stickprov på differensserien beräknat från varje par. I R kan man antingen beräkna den differensserien eller använda `t.test`-funktionen med två dataserier och argumentet för parvisa observationer satt till sant, `paired = T`. Som exempel ges följande data från en studie på äpple, där trädhöjd mätts före och efter en näringsbehandling.

```
dat <- tibble(TrädID = 1:4,
              `Tidpunkt 1` = c(48, 43, 30, 47),
              `Tidpunkt 2` = c(51, 44, 42, 54))
dat
```

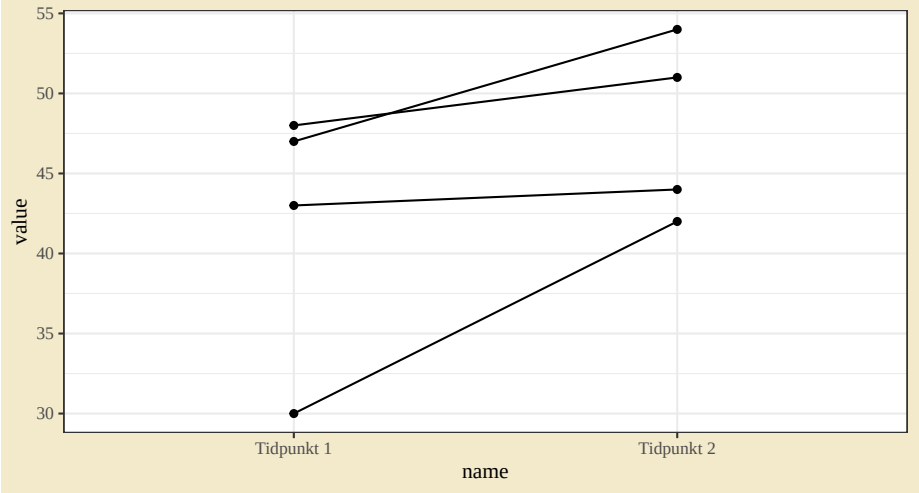
```
## # A tibble: 4 x 3
##   TrädID `Tidpunkt 1` `Tidpunkt 2`
##    <int>        <dbl>        <dbl>
## 1      1           48           51
## 2      2           43           44
## 3      3           30           42
## 4      4           47           54
```

Datan kan illustreras med ett punktdiagram där en linje binder samman paret. För att enkelt skapa grafen i `ggplot2` kan man först omstrukturera datan till lång form genom `pivot_longer`.

```
dat_long <- dat %>% pivot_longer(-TrädID)
dat_long
```

```
## # A tibble: 8 x 3
##   TrädID name       value
##    <int> <chr>      <dbl>
## 1      1 Tidpunkt 1    48
## 2      1 Tidpunkt 2    51
## 3      2 Tidpunkt 1    43
## 4      2 Tidpunkt 2    44
## 5      3 Tidpunkt 1    30
## 6      3 Tidpunkt 2    42
## 7      4 Tidpunkt 1    47
## 8      4 Tidpunkt 2    54
```

```
ggplot(dat_long, aes(name, value, group = TrädID)) +
  geom_point() +
  geom_line()
```



Testet för parade stickprov kan antingen utföras som ett enkelt t-test på differensserien

```
t.test(dat$`Tidpunkt 2` - dat$`Tidpunkt 1`)
```

```
##
##  One Sample t-test
##
## data:  dat$`Tidpunkt 2` - dat$`Tidpunkt 1`
```

```
## t = 2.3681, df = 3, p-value = 0.09868
## alternative hypothesis: true mean is not equal to 0
## 95 percent confidence interval:
##  -1.977405 13.477405
## sample estimates:
## mean of x
##      5.75
```

eller som ett t-test för två stickprov där man särskilt anger att datan är parad

```
t.test(dat$`Tidpunkt 1`, dat$`Tidpunkt 2`, paired = T)
```

```
##
##  Paired t-test
##
## data:  dat$`Tidpunkt 1` and dat$`Tidpunkt 2`
## t = -2.3681, df = 3, p-value = 0.09868
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -13.477405   1.977405
## sample estimates:
## mean of the differences
##                   -5.75
```

För bägge alternativen måste datan vara ordnad så att de två vektorerna matchar varandra parvis. Notera att ordningen på vektorerna påverkar konfidensintervall men inte p-värdet (i fallet med en tvåsidig mothypotes). Här är det naturligt att ta den andra mätningen först eftersom konfidensintervallet då blir ett intervall för medelvärdesökningen efter behandling. Ett p-värde på 0.0987 ger att man inte förkastar vid en signifikansnivå på fem procent.

8.1.2 t-test för två oberoende stickprov

Ett t-test för två oberoende stickprov testar om två populationsmedelvärden är lika. Ta som exempel följande data på jordgubbsskörd vid två olika näringsbehandlingar (A och B). Här är stickprov inte matchade - det finns ingen direkt koppling mellan en observation i den ena behandlingsgruppen till någon observation i den andra.

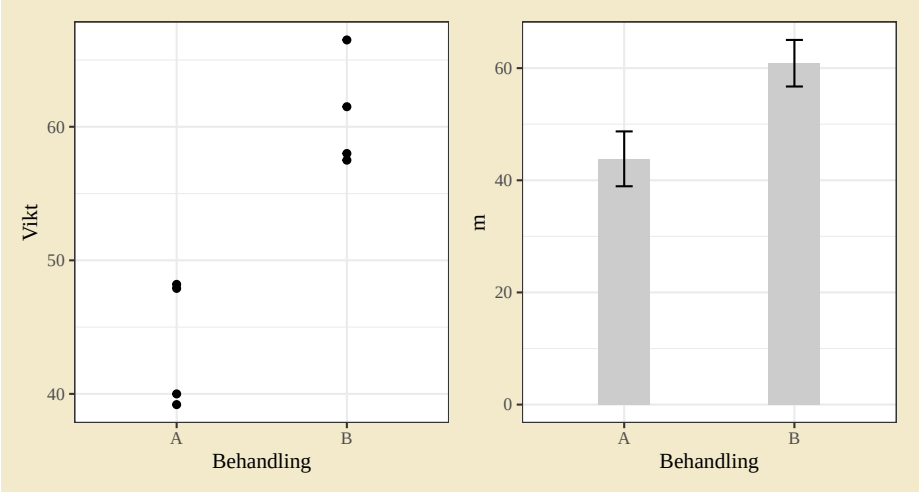
```
dat <- tibble(Behandling = c("A", "A", "A", "A", "B", "B", "B", "B"),
              Vikt = c(40, 48.2, 39.2, 47.9, 57.5, 61.5, 58, 66.5))
kable(dat)
```

Datan kan illustreras med ett enkelt punktdiagram. I ett publiceringssammanhang hade det kanske presenterats med ett stapeldiagram med felstaplar.

```
g1 <- ggplot(dat, aes(Behandling, Vikt)) +
  geom_point()
```

Behandling	Vikt
A	40.0
A	48.2
A	39.2
A	47.9
B	57.5
B	61.5
B	58.0
B	66.5

```
g2 <- dat %>%
  group_by(Behandling) %>%
  summarise(m = mean(Vikt), s = sd(Vikt)) %>%
  ggplot(aes(Behandling, m)) +
  geom_bar(stat = "identity", fill = "grey80", width = 0.3) +
  geom_errorbar(aes(ymin = m - s, ymax = m + s), width = 0.1)

library(patchwork)
g1 + g2
```



Ett t-test för två oberoende stickprov har nollhypotesen att grupperna har samma populationsmedelvärde och alternativhypotesen att populationsmedelvärdena är skilda (för det tvåsidiga fallet):

$$\mu_1 = \mu_2 \qquad \mu_1 \neq \mu_2.$$

Testet kan utföras i R genom funktionen `t.test`. Data kan antingen anges som

en formel med dess data `Vikt ~ Behandling, data = dat` (vilket man kan läsa som *vikt uppdelat efter behandling*) eller som två skilda vektorer. Det förra alternativet är oftast enklare om man har datan på lång form - med en kolumn som anger grupp (i exemplet *Behandling*) och en kolumn som anger utfallsvärdet (i exemplet *Vikt*).

```
# Formelskrivning
t.test(Vikt ~ Behandling, data = dat, var.equal = T)
```

```
##
##  Two Sample t-test
##
## data:  Vikt by Behandling
## t = -5.3157, df = 6, p-value = 0.001803
## alternative hypothesis: true difference in means between group A and group B is not equal to 0
## 95 percent confidence interval:
##  -24.898417  -9.201583
## sample estimates:
## mean in group A mean in group B
##          43.825          60.875
```

```
# Två separata vektorer
## Filtrera ut data där behandling är A
Vikt_A <- dat$Vikt[dat$Behandling == "A"]

## Filtrera ut data där behandling är B
Vikt_B <- dat$Vikt[dat$Behandling == "B"]

t.test(Vikt_A, Vikt_B, var.equal = T)
```

```
##
##  Two Sample t-test
##
## data:  Vikt_A and Vikt_B
## t = -5.3157, df = 6, p-value = 0.001803
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -24.898417  -9.201583
## sample estimates:
## mean of x mean of y
##    43.825    60.875
```

Argumentet `var.equal = T` används för att beräkna testet där gruppernas varianser antas vara lika. Grundinställningen är testet där varianser inte antas vara lika, så `t.test(Vikt ~ Behandling, data = dat)` ger ett lite annat resultat. Testet ger ett p-värde på 0.0018, vilket leder till att nollhypotesen förkastas på enprocentsnivån. Detta tyder på att det finns en viktskillnad mellan behandlingarna. Utskriften ger också ett 95-procentigt konfidensintervall på

$(-24.898, -9.202)$. Tolkningen är att skillnaden mellan populationsmedelvärden ligger i intervallet med 95 procents konfidens. Notera att värdet noll inte ligger i intervallet.

8.2 Binär data. Proportioner

Om man vill jämföra två proportioner kan man använda z-testet för två stickprov. Säg till exempel att man utvecklar den tidigare studien, som gav 17 av 50 infekterade plantor, till att undersöka ytterligare en sort och att den sorten har 26 infekterade plantor av en total på 60. Testets hypotesen är i det tvåsidiga fallet

$$H_0 : p_1 = p_2 \qquad H_1 : p_1 \neq p_2.$$

I R kan testet genomföras med `prop.test`-funktionen. Funktionens första argument är antalen infekterade, som en vektor med två värden, och dess andra argument är totalerna. Likt testet med ett stickprov finns en möjlighet att göra en kontinuitetskorrektion med `correct`-argumentet.

```
prop.test(c(17, 26), c(50, 60), correct = F)
```

```
##
##  2-sample test for equality of proportions without continuity
##  correction
##
## data:  c(17, 26) out of c(50, 60)
## X-squared = 0.9978, df = 1, p-value = 0.3178
## alternative hypothesis: two.sided
## 95 percent confidence interval:
##  -0.27488771  0.08822105
## sample estimates:
##    prop 1    prop 2
## 0.3400000 0.4333333
```

Notera att funktionen inte ger ett z-värde utan ett χ^2-värde (utskrivet `X-squared`). Det beror på att funktionen beräknar z-testet som ett likvärdigt χ^2-test. Det z-värde man får om man genomför testet som ett z-test är detsamma som roten ur utskriftens χ^2-värde. Testet ger ett högt p-värde vilket innebär att nollhypotesen inte förkastas.

8.3 Nominal eller ordinal data. Korstabeller

Data med två insamlade variabler per observerad enhet kan presenteras med en korstabell. Ta som (ett något deppigt) exempel överlevandsdata från Titanic. Datan finns tillgänglig i R som `Titanic` och man kan konstruera en korstabell

Class	Sex	Age	Survived	n
1st	Male	Adult	No	118
2nd	Male	Adult	No	154
3rd	Male	Adult	No	387
Crew	Male	Adult	No	670
1st	Male	Adult	Yes	57
2nd	Male	Adult	Yes	14
3rd	Male	Adult	Yes	75
Crew	Male	Adult	Yes	192

med `pivot_wider`. I detta fall ges överlevnad filtrerad på vuxna män, uppdelat efter klass.

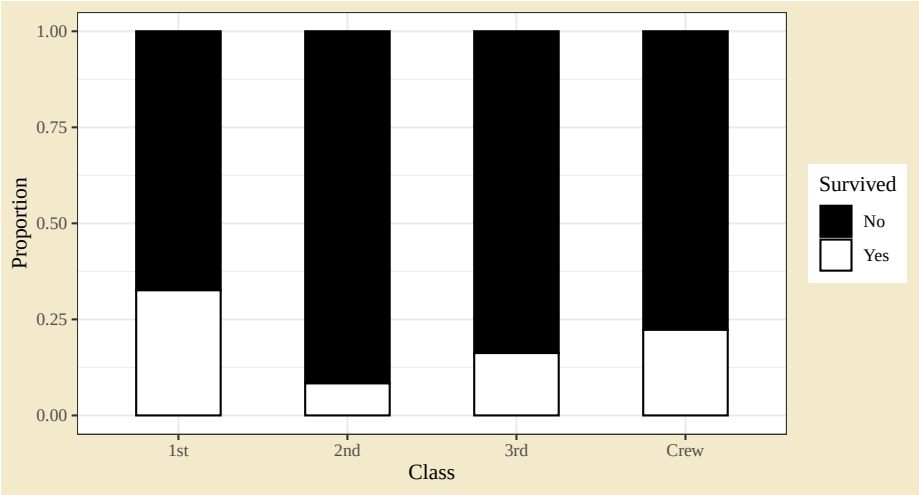
```
dat <- Titanic %>%
  as_tibble() %>%
  filter(Age == "Adult", Sex == "Male")
kable(dat)
```

En korstabell kan konstrueras med `pivot_wider`.

```
dat_wide <- dat %>%
  pivot_wider(names_from = Survived, values_from = n)
```

Datan tyder på att överlevnad är beroende av klass - en tredjedel av förstaklass överlever, men en sjättedel av tredjeklass överlever. En illustration kan göras genom ett stapeldiagram.

```
ggplot(dat, aes(Class, n, fill = Survived)) +
  geom_bar(stat = "identity", position = position_fill(),
           color = "black", width = 0.5) +
  scale_fill_manual(values = c("black", "white")) +
  ylab("Proportion")
```



Argumentet `position` i `geom_bar` används för att skapa proportionella staplar.

Ett χ^2-test på en korstabell har nollhypotesen att det inte finns något samband mellan variabeln för rader och variabeln för kolumner. Antal frihetsgrader ges av antal rader minus ett gånger antal kolumner minus ett. Testet kan enkelt göras med `chisq.test`. Som ingångsvärde kan man plocka ut kolumnerna med numeriska värden genom hakparenteser.

```
dat_wide[, 4:5] # De två numeriska kolumnerna
```

```
## # A tibble: 4 x 2
##      No   Yes
##   <dbl> <dbl>
## 1   118    57
## 2   154    14
## 3   387    75
## 4   670   192
```

```
chisq.test(dat_wide[, 4:5])
```

```
##
##  Pearson's Chi-squared test
##
## data:  dat_wide[, 4:5]
## X-squared = 37.988, df = 3, p-value = 2.843e-08
```

Utskriften ger teststorheten, antal frihetsgrader, och p-värdet. I det här fallet är p-värdet mycket litet (skrivning med `e` ska läsas som $2.843e-8 = 2.843 \cdot 10^{-8} = 0.00000002843$) och slutsatsen blir att nollhypotesen förkastas - det finns ett samband mellan klass och överlevnad. Antalet frihetsgrader ges av antalet rader minus ett gånger antalet kolumner minus ett (här $(4-1) \cdot (2-1) = 3$).

Odlingstyp	Monoglukosid
Ekologisk	135.7
Ekologisk	186.7
Ekologisk	182.0
Ekologisk	180.5
Konventionell	138.9
Konventionell	155.1
Konventionell	126.3
Konventionell	120.1

χ^2-test är ett asymptotiskt test - dess egenskaper är beroende av *stora* stickprov. Som gräns för storleken används ofta att samtliga förväntade antal ska vara större än 5. Funktionen ger en varning om förväntade värden är små. En möjlig lösning i sådana fall är att slå ihop klasser.

```
test_result <- chisq.test(dat_wide[, 4:5])
test_result$expected # Samtliga förväntade värden över 5
```

```
##            No       Yes
## [1,] 139.5171  35.48290
## [2,] 133.9364  34.06359
## [3,] 368.3251  93.67487
## [4,] 687.2214 174.77864
```

Om detta krav inte är uppfyllt skriver funktionen ut en varning.

```
dat <- matrix(c(4,2,5,1), 2)
test_result <- chisq.test(dat)
test_result$expected
```

```
##      [,1] [,2]
## [1,]  4.5  4.5
## [2,]  1.5  1.5
```

8.4 Övningar

Övning 8.1 (Lök)**.** Åtta monoglukosidmätningar på lök samlas in från fyra konventionella och fyra ekologiska ordlare.

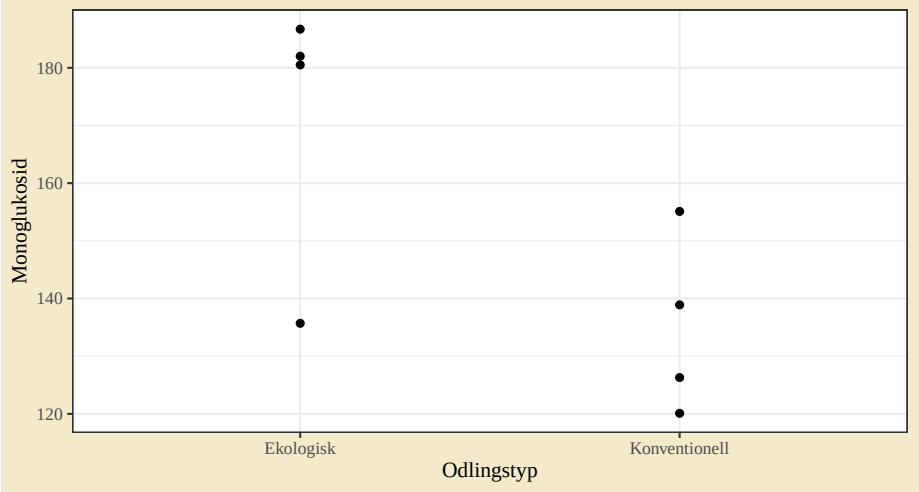
a. Genomför ett hypotestest för att se om det finns en medelvärdesskillnad mellan odlingstyperna.

b. Beräkna ett 95-procentigt konfidensintervall för skillnaden i medelvärde.

Lösningsförslag 8.1 (Lök)**.** Datan kan importeras från excelfilen med uppgiftsdata. En lämplig graf kan skapas genom att pivotera till långform och plotta

med ggplot.

```
dat <- readxl::read_excel("Data/Uppgiftsdata.xlsx", sheet = "Lökfärg")

dat %>%
  pivot_longer(-Odlare, names_to = "Odlingstyp",
               values_to = "Monoglukosid") %>%
  ggplot(aes(Odlingstyp, Monoglukosid)) +
  geom_point()
```



a. Eftersom det inte finns någon koppling mellan odlare är ett t-test för två oberoende stickprov ett lämpligt test. Testet kan genomföras med eller utan ett antagande om lika varianser inom grupperna.

```
t.test(dat$Konventionell, dat$Ekologisk, var.equal = T)
```

```
##
##  Two Sample t-test
##
## data:  dat$Konventionell and dat$Ekologisk
## t = -2.5436, df = 6, p-value = 0.04387
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -70.877089  -1.372911
## sample estimates:
## mean of x mean of y
##   135.100   171.225
```

```
t.test(dat$Konventionell, dat$Ekologisk, var.equal = F)
```

```
##
##  Welch Two Sample t-test
```

```
##
## data:  dat$Konventionell and dat$Ekologisk
## t = -2.5436, df = 5.145, p-value = 0.05033
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -72.32625466   0.07625466
## sample estimates:
## mean of x mean of y
##   135.100   171.225
```

Hypotestestet (ett t-test för två oberoende stickprov) ger ett p-värde på 0.0439 eller 0.0503, beroende på om varianser inom grupperna antas lika eller ej.

b. Om varianser antas lika ges ett 95-procentigt konfidensintervall av $(-70.9, -1.4)$. Notera att ett 95-procentigt intervall inte täcker noll, vilket är i linje med att p-värdet är mindre än 5 procent.

Övning 8.2 (Allsvenskan. Genomsnittligt antal mål)**.** Bland kursdata finns datafiler med allsvenska matcher för damer (2000-2020) och herrar (1924-2019). Det genomsnittliga antalet mål i respektive serie ges för åren 2000-2009 av följande tabell.

```
dat_dam <- read_csv("Data/Allsvenskan, damer, 2000-2020.csv")
dat_herr <- read_csv("Data/Allsvenskan, herrar, 1924-2019.csv")

dat_dam_medel <- dat_dam %>%
  group_by(sasong) %>%
  summarise(Mål = mean(hemmamal + bortamal)) %>%
  filter(sasong %in% 2000:2009)

dat_herr_medel <- dat_herr %>%
  group_by(sasong) %>%
  summarise(Mål = mean(hemmamal + bortamal)) %>%
  filter(sasong %in% 2000:2009) %>%
  mutate(sasong = as.numeric(sasong))

dat_total <- dat_dam_medel %>%
  mutate(Allsvenska = "Dam") %>%
  bind_rows(dat_herr_medel %>% mutate(Allsvenska = "Herr")) %>%
  rename(Säsong = sasong)

dat_total %>%
  pivot_wider(values_from = Mål, names_from = Allsvenska) %>%
  kable(digits = 2)
```

a. Jämför serierna med en lämplig graf.

b. Beräkna ett konfidensintervall för det genomsnittliga antalet mål i allsvenskan

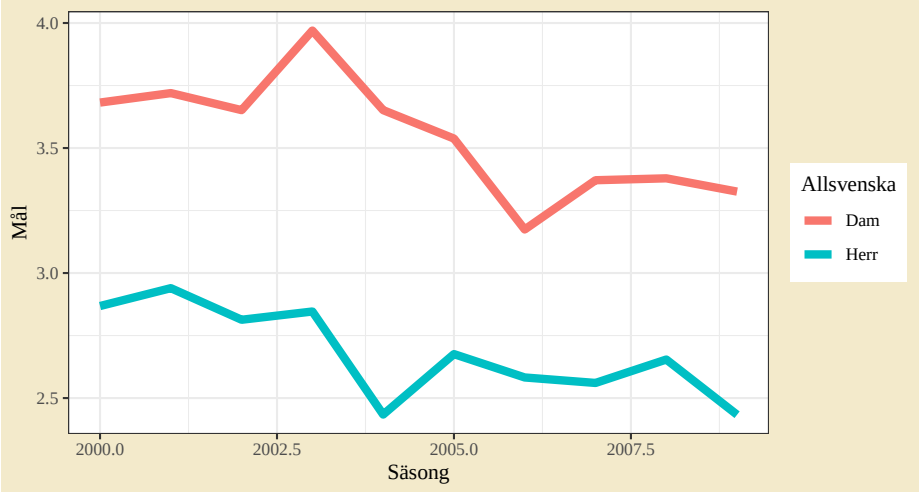
Säsong	Dam	Herr
2000	3.68	2.87
2001	3.72	2.94
2002	3.65	2.81
2003	3.97	2.85
2004	3.65	2.43
2005	3.54	2.68
2006	3.17	2.58
2007	3.37	2.56
2008	3.38	2.65
2009	3.33	2.43

för damer.

c. Genomför ett lämpligt test för att se om serierna har samma populationsmedelvärde för antal mål. Diskutera om datan bör ses som parad (två observationer per år).

Lösningsförslag 8.2 (Allsvenskan. Genomsnittligt antal mål)**.** a. Datan kan illustreras med ett linjediagram med två separata linjer.

```
# Fortsättning från inläsningen ovan
ggplot(dat_total, aes(Säsong, Mål, col = Allsvenska)) +
  geom_line(size = 2)
```



Allsvenskan för damer har ett högre genomsnittligt antal mål än allsvenskan för herrar.

b. Konfidensintervall kan beräknas med `t.test` efter att observationer för damer filtrerats ut.

```
dat <- dat_total %>% filter(Allsvenska == "Dam")
t.test(dat$Mål)
```

```
##
##  One Sample t-test
##
## data:  dat$Mål
## t = 47.784, df = 9, p-value = 3.858e-12
## alternative hypothesis: true mean is not equal to 0
## 95 percent confidence interval:
##  3.378329 3.714095
## sample estimates:
## mean of x
##  3.546212
```

Konfidensintervallet ges av $(3.37, 3.71)$. Det sanna populationsmedelvärdet ligger med 95-procents konfidens i det intervallet.

c. Ett lämpligt test kan genomföras med `t.test`. Nollhypotesen är att serierna har samma populationsmedelvärde och att den skillnad man kan se därmed enbart är slumpmässig variation.

```
t.test(Mål ~ Allsvenska, dat_total)
```

```
##
##  Welch Two Sample t-test
##
## data:  Mål by Allsvenska
## t = 9.24, df = 16.897, p-value = 5.129e-08
## alternative hypothesis: true difference in means between group Dam and group Herr is not equal
## 95 percent confidence interval:
##  0.6677709 1.0631973
## sample estimates:
##  mean in group Dam mean in group Herr
##           3.546212           2.680728
```

Det låga p-värdet ger att nollhypotesen bör förkastas - man kan dra slutsatsen att allsvenskan för damer har högre genomsnittligt antal mål än allsvenskan för herrar.

I det här fallet kan man diskutera om det inte är mer lämpligt att använda ett t-test för parade observation, eftersom det är två observationer per år. Ett argument mot det är att det inte finns någon egentlig anledning att tro att det skulle finnas något samband mellan antal mål för damer respektive för herrar ett visst år.

Övning 8.3 (Nature's mosquito)**.** En viktig del i arbetet mot spridning av malaria är att förstå myggors reaktion på dofter. Ett vanligt sätt att undersöka detta är att följa flyktmönster i en sluten vindtunnel. I ett försök har man en

vindtunnel med måtten 60 x 30 x 100 cm (bredd, höjd, djup). I den ena änden finns två doftkällor placerade i jämn höjd, med 30 centimeters mellanrum. I den andra änden släpper man en mygga i taget och genom videoinspelning kan man beräkna dess position varje tiondels sekund. Försöket omfattar 60 individer av två arter. Datan finns tillgänglig bland kursdatan i filen *Mosquitos.csv*.

a. Illustrera en eller flera myggors flykt med ett passande diagram. En möjlighet är att ha z-koordinaten på den horisontella axeln och x-koordinaten på den vertikala axeln och göra en spårlinje med `geom_path`. Den linjen motsvarar då flykten sedd från ovan. Eftersom flykten anges med x-, y- och z-koordinater kan man även illustrera det som en tre-dimensionell graf, till exempel med `rgl`-paketet.

b. Det är möjligt att ta fram myggornas landningspunkt på bortre änden genom att filtera på värden där `Proportional_time` är 1. Man får då följande värden på x-koordinaten.

Konstruera ett lådagram över x-koordinaterna för de 60 individerna vid landningstillfället. Dela grafen så att arterna visas i skilda lådagram. Finns det tecken på skillnader i landningspunkt?

c. Genomför ett lämpligt t-test för att se om det finns skillnader mellan arterna, med avseende på x-koordinat vid landningstillfället.

d. Konstruera ett konfidensintervall för medelvärdet av x-koordinaten vid landning för *Aedes aegypti*. Använd intervallet för att se om medelvärdet är statistiskt signifikant skilt från 0.

e. Om man bara sorterar individer efter om de landar på den högra eller vänstra halvan av den borde änden kan man ställa upp följande korstabell.

Genomför ett lämpligt z-test eller χ^2-test för att se om det finns en statistiskt säkerställd skillnad i landningsposition mellan arterna.

Frågan är baserad på Hinze et al (2021) *Mosquito Host Seeking in 3D Using a Versatile Climate-Controlled Wind Tunnel System*.

Lösningsförslag 8.3 (Nature's mosquito)**.** Datan importas med `read_csv`.

```
dat <- read_csv("Data/Mosquitos.csv")
```

a. En passande graf kan konstrueras med `ggplot2` och `geom_path`. Med två grafer kan man illustrera flykten från sidan och från ovan. Färger kan separera arterna. Den specifika individen måste sättas med `group = id` för att den enskilda flykten ska bli en egen linje.

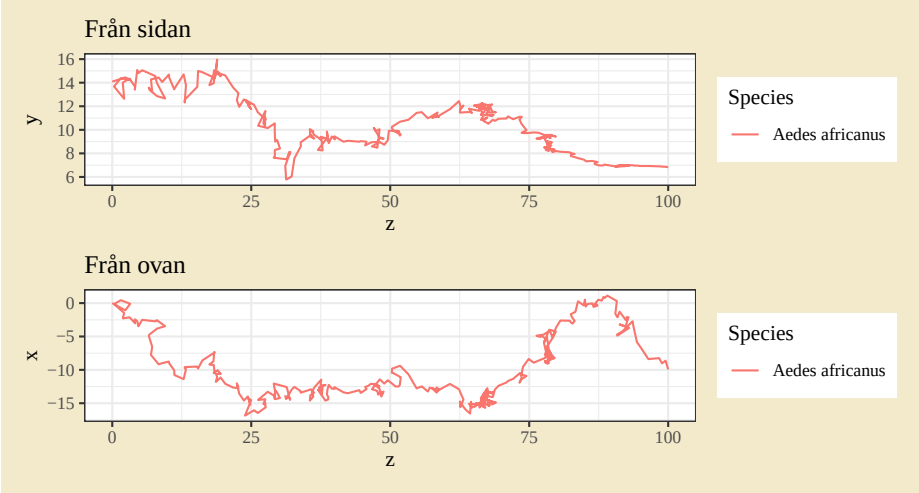
Ett exempel på en enskild individ.

```
dat_ind1 <- dat %>% filter(id == 1)

g1 <- ggplot(dat_ind1, aes(z, y, group = id, col = Species)) +
  geom_path() +
```

Aedes africanus	Aedes aegypti
-9.94	8.54
-10.75	1.18
-4.97	-1.55
-3.50	-1.65
-3.63	7.64
-11.39	4.99
-15.98	-3.77
2.01	4.53
-13.19	3.18
-4.51	9.63
-6.17	5.83
-11.56	-0.26
-5.38	3.71
-3.88	3.07
-1.83	5.15
-4.04	0.02
-5.80	-0.45
2.12	-5.83
-13.67	7.14
-14.61	12.28
-3.46	-3.50
-8.02	5.83
0.20	1.43
-14.54	-8.64
-17.62	5.14
-10.41	-0.11
1.03	13.40
-6.06	14.80
-3.41	-6.91
-0.48	8.38

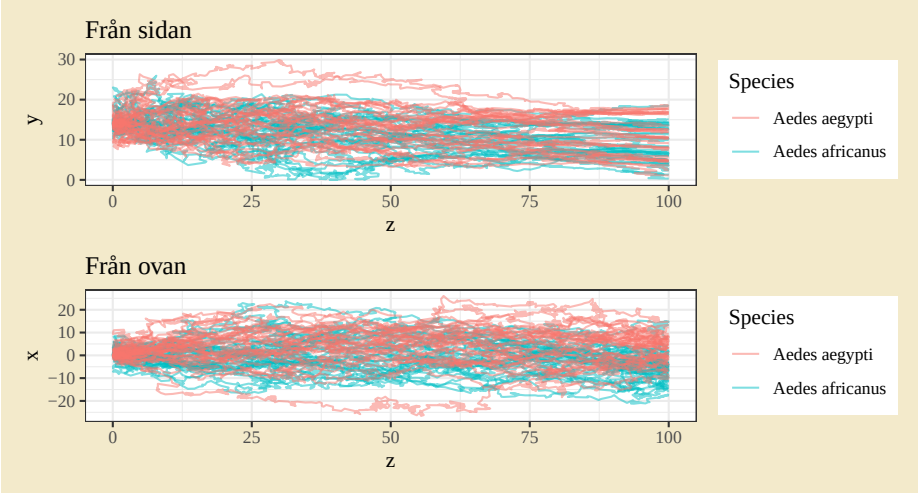
Species	Left	Right
Aedes aegypti	10	20
Aedes africanus	26	4

```
  labs(title = "Från sidan")

g2 <- ggplot(dat_ind1, aes(z, x, group = id, col = Species)) +
  geom_path() +
  labs(title = "Från ovan")

library(patchwork)
g1 / g2
```



Ett exempel med samtliga individer i en (rörig) illustration.

```
g1 <- ggplot(dat, aes(z, y, group = id, col = Species)) +
  geom_path(alpha = 0.5) +
  labs(title = "Från sidan")

g2 <- ggplot(dat, aes(z, x, group = id, col = Species)) +
  geom_path(alpha = 0.5) +
  labs(title = "Från ovan")

library(patchwork)
g1 / g2
```

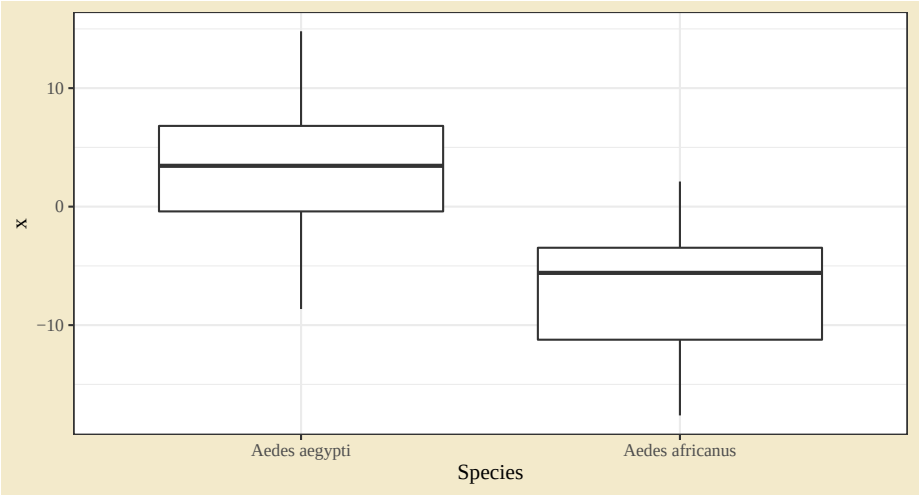


R har paket med funktioner för tre-dimensionella grafer. Ett exempel är genom paketet `rgl` och funktionerna `plot3d` och `lines3d`.

```
# install.packages(rgl)
library(rgl)
# En enskild mygga plottad med plot3d
# Det första argumentet är de tre koordinatvariablerna efter filter på första individen
plot3d(dat %>% filter(id == 1) %>% select(z, x, y), type = "l", asp = F, col = "blue", alpha = 0.

# Övriga flykter tillagda med lines3d i en for-loop
# Loopen går inom samtliga individer och lägger till linjen till 3d-grafen
for(i in 1:60){
  lines3d(dat %>% filter(id == i) %>% select(z, x, y),
          col = ifelse(dat %>% filter(id == i) %>% pull(Species) == "Aedes africanus", "blue", "r
          alpha = 0.15)
}

# Skapa punkter för landningsplatsen
points3d(dat %>% filter(Proportional_time == 1) %>% select(z, x, y), size = 6,
         col = ifelse(dat %>% filter(Proportional_time == 1) %>% pull(Species) == "Aedes africanu
```

b. För att titta på landningsplatser filtreras på de observationer där den proportionella observationstiden är 1 (den sista observationen). Därefter kan en lämplig graf skapas med `geom_boxplot` med art på x-axeln och x-koordinaten på grafens y-axel.

```
dat_endtime <- dat %>% filter(Proportional_time == 1)

ggplot(dat_endtime, aes(Species, x)) + geom_boxplot()
```



Aedes aegypti uppvisar högre x-koordinater, vilket tyder på att den arten varit mer lockad av doftämnet till höger.

c. Ett t-test för oberoende stickprov kan genomföras genom att ta x-koordinaten som utfallsvariabel och arterna som de två grupperna. Testet kan genomföras med eller utan antaganden om lika varianser inom grupperna. Nollhypotesen är arterna har samma populationsmedelvärde i x-koordinaten vid landningstillfället.

```
t.test(x ~ Species, dat_endtime)
```

```
##
##  Welch Two Sample t-test
##
## data:  x by Species
## t = 6.7001, df = 57.845, p-value = 9.436e-09
## alternative hypothesis: true difference in means between group Aedes aegypti and gr
## 95 percent confidence interval:
##   6.93371 12.84229
## sample estimates:
##   mean in group Aedes aegypti mean in group Aedes africanus
##                      3.106667                      -6.781333
```

```
t.test(x ~ Species, dat_endtime, var.equal = T)
```

```
##
##  Two Sample t-test
##
## data:  x by Species
## t = 6.7001, df = 58, p-value = 9.331e-09
## alternative hypothesis: true difference in means between group Aedes aegypti and gr
```

```
## 95 percent confidence interval:
##   6.933879 12.842121
## sample estimates:
##   mean in group Aedes aegypti mean in group Aedes africanus
##                      3.106667                      -6.781333
```

Bägge testet ger stark signifikans. Man drar slutsatsen att det finns en skillnad mellan arterna.

d. Ett konfidensintervall kan tas fram med `t.test` efter filtrering på art så att enbart *aegypti* förekommer i datan.

```
dat_endtime_aegypti <- dat_endtime %>% filter(Species == "Aedes aegypti")
t.test(dat_endtime_aegypti$x)
```

```
##
##  One Sample t-test
##
## data:  dat_endtime_aegypti$x
## t = 2.9029, df = 29, p-value = 0.006996
## alternative hypothesis: true mean is not equal to 0
## 95 percent confidence interval:
##  0.9178614 5.2954719
## sample estimates:
## mean of x
##  3.106667
```

Intervallet ges av $(0.92, 5.30)$. Eftersom 0 inte ingår i det 95-procentiga konfidensintervallet är det statistiskt signifikant (på 5-procentsnivån) att populationsmedelvärdet skiljer sig från 0.

e. Antalet kan antingen skrivas in från siffrorna i uppgiften eller beräknas från data genom att koda om x-koordinaten till höger respektive vänster beroende på om x är större eller mindre än 0. Därefter kan ett test genomföras med `chisq.test`. Nollhypotesen är att populationsproportionen högergående (eller vänstergående) är lika stor för de två arterna.

```
dat_prop <- dat_endtime %>%
  count(Species, Direction = ifelse(x < 0, "Left", "Right")) %>%
  pivot_wider(names_from = Direction, values_from = n)

chisq.test(dat_prop %>% select(Left, Right))
```

```
##
##  Pearson's Chi-squared test with Yates' continuity correction
##
## data:  dat_prop %>% select(Left, Right)
## X-squared = 15.625, df = 1, p-value = 7.723e-05
```

	Lungcancer	Ej lungcancer
Burfågel	98	101
Ej burfågel	141	328

Färg	a1b1	a1b2	a2b1	a2b2
A	56	64	36	38
B	45	36	44	48
C	18	13	27	20
D	6	12	18	19

Testet pekar på en skillnad i proportionen höger- respektive vänstergående mellan arterna.

Övning 8.4 (Burfågel)**.** I en undersökning av lungcancerpatienter finner man följande antal.

Genomför ett test för att se om andelen burfågelägare än densamma i de två grupperna.

Lösningsförslag 8.4 (Burfågel)**.** Frågan är om proportionen burfågelägare bland patienter är densamma som proportionen burfågelägare bland icke-drabbade. Man ska alltså ställa proportionen 98 av 239 mot 101 av 429.

```
prop.test(c(98, 101), c(98 + 141, 101 + 328), correct = F)
```

```
##
##  2-sample test for equality of proportions without continuity
##  correction
##
## data:  c(98, 101) out of c(98 + 141, 101 + 328)
## X-squared = 22.374, df = 1, p-value = 2.244e-06
## alternative hypothesis: two.sided
## 95 percent confidence interval:
##  0.1004485 0.2487727
## sample estimates:
##    prop 1    prop 2
## 0.4100418 0.2354312
```

Det låga p-värdet tyder på att det finns skillnader mellan grupperna.

Övning 8.5 (Po-ta-toes)**.** I en undersökning på potatis används fyra behandlingar (a1b1, a1b2, a2b1 och a2b2). 125 potatisar från varje behandling sorteras in i fyra olika färggrupper (A, B, C och D). Frekvenstabellen ges av följande.

Genomför ett lämpligt test för att se om det finns färgskillnader mellan behandlingarna.

Lösningsförslag 8.5 (Po-ta-toes)**.** Funktionen `matrix` kan användas för att skapa korstabellen. Tabellen kan sedan tas som ingångsvärde till `chisq.test`.

```
dat <- matrix(c(56,64,36,38,
                45,36,44,48,
                18,13,27,20,
                6,12,18,19),
              nrow = 4, byrow = T)

test <- chisq.test(dat)
test
```

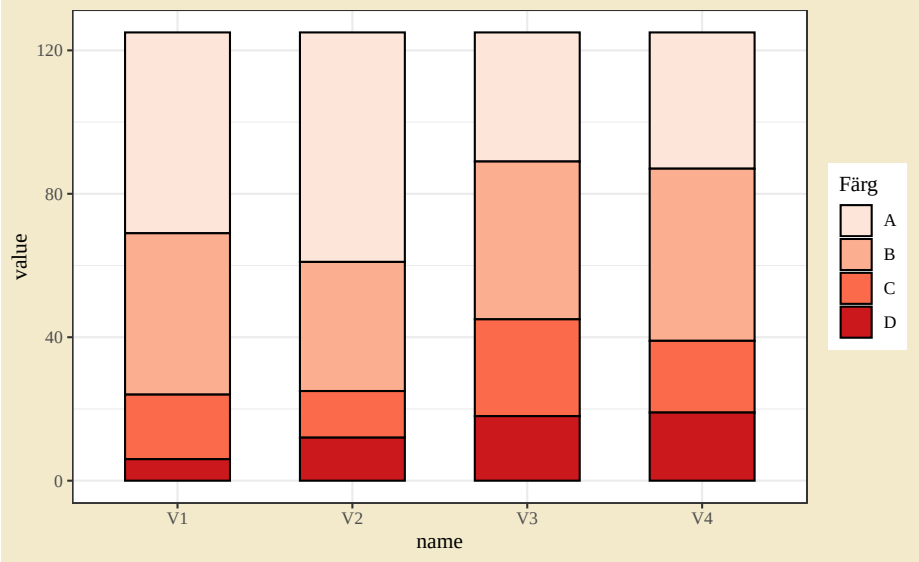
```
##
##  Pearson's Chi-squared test
##
## data:  dat
## X-squared = 26.518, df = 9, p-value = 0.00168
```

I ett χ^2-test är nollhypotesen att kolumner och rader är oberoende. Det låga p-värdet tyder på att nollhypotesen bör förkastas, vilket tyder på att det finns ett samband mellan behandling och färg.

Datan kan illustreras med ett stapeldiagram.

```
dat_long <- dat %>%
  as_tibble() %>%
  mutate(Färg = c("A", "B", "C", "D")) %>%
  pivot_longer(-Färg)

ggplot(dat_long, aes(name, value, fill = Färg)) +
  geom_bar(stat = "identity", col = "black", width = 0.6) +
  scale_fill_brewer(palette = "Reds")
```

	Lungcancer	Ej lungcancer
Burfågel	98	101
Ej burfågel	141	328



Övning 8.6 (Mer burfågel)**.** En tidigare uppgift gav följande data kring en eventuell koppling mellan fågelägande och lungcancer.

Genomför ett χ^2-test för att se om det finns något signifikant samband mellan variablerna.

Lösningsförslag 8.6 (Mer burfågel)**.** För att genomföra χ^2-testet kan man skriva in data som en 2-gånger-2-matris med funktionen `matrix`.

```
dat <- matrix(c(98, 141, 101, 328), nrow = 2)
chisq.test(dat, correct = F)
```

```
##
##  Pearson's Chi-squared test
##
## data:  dat
## X-squared = 22.374, df = 1, p-value = 2.244e-06
```

Samma resultat som i den tidigare beräkningen på samma data.

Övning 8.7 (Röda rummet, Gösta Berlings saga, och okänd)**.** Bland kursdatan finns text till Selma Lagerlöfs *Gösta Berlings saga* och August Strindbergs *Röda rummet*, i kolumnform. Det finns också en text med okänd författare (filen *Den tredje boken*). En ytlig textanalys kan baseras på frekvensen för olika ord.

Ord	Gösta Berlings saga	Okänd	Röda rummet
och	109	143	166
han	116	115	52
att	67	130	74
det	46	91	63
i	52	58	83

Närliggande tabell ger antalet förekomster för texternas fem vanligaste ord i första kapitlet av varje text.

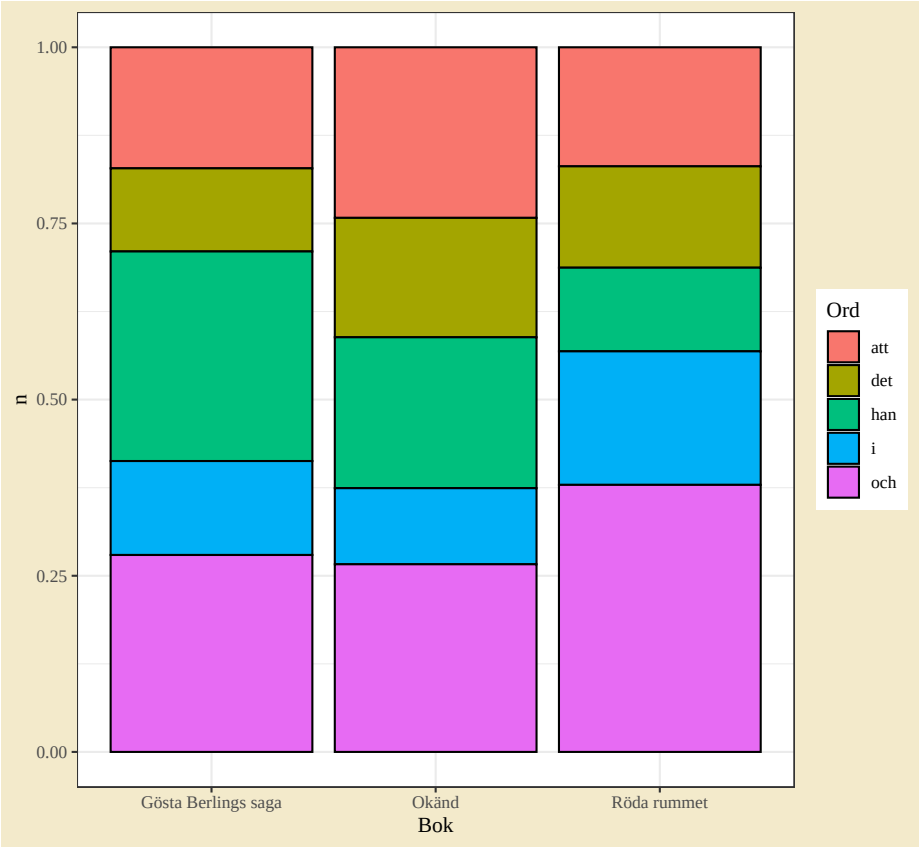
```
dat_rr <- read_csv("Data/Röda rummet.csv") %>%
  mutate(Bok = "Röda rummet")
dat_gbs <- read_csv("Data/Gösta Berlings saga.csv") %>%
  mutate(Bok = "Gösta Berlings saga")
dat_tredje <- read_csv("Data/Den tredje boken.csv") %>%
  mutate(Bok = "Okänd")

dat_full <- bind_rows(dat_rr, dat_gbs, dat_tredje)
vanligaste_ord <- dat_full %>%
  count(Ord, sort = T) %>%
  slice(1:5) %>%
  pull(Ord)

dat_ord <- dat_full %>%
  filter(Ord %in% vanligaste_ord, Kapitel == 1) %>%
  count(Bok, Ord)

dat_ord_wide <- dat_ord %>%
  pivot_wider(names_from = Bok, values_from = n) %>%
  arrange(desc(`Gösta Berlings saga` + Okänd + `Röda rummet`))
kable(dat_ord_wide)
```

a Illustrera frekvenserna med en lämplig graf.

b. Genomför två χ^2-test för att se vilken av Gösta Berlings saga och Röda rummet den okända texten är mest lik (sett till de fem vanligaste orden).

Lösningsförslag 8.7 (Röda rummet, Gösta Berlings saga, och okänd)**.** a. En möjlig illustration är ett stapeldiagram för proportion av varje ord inom respektive text.

```
dat_ord %>%
  ggplot(aes(Bok, n, fill = Ord)) +
  geom_bar(stat = "identity", position = position_fill(), col = "black")
```



Röda rummet har större andel förekomster av *och* och *i*.

b. Två χ^2-test kan användas för att ställa de kända verken mot det okända verket. Detta kan göras genom att välja kolumner från `dat_ord_wide` med hakparenteser.

```
# Gösta Berlings saga (kolumn 2) mot okänd text (kolumn 3)
chisq.test(dat_ord_wide[, 2:3])
```

```
##
##  Pearson's Chi-squared test
##
## data:  dat_ord_wide[, 2:3]
## X-squared = 16.963, df = 4, p-value = 0.001965
```

```
# Röda rummet (kolumn 4) mot okänd text (kolumn 3)
chisq.test(dat_ord_wide[, 3:4])
```

```
##
##  Pearson's Chi-squared test
```

```
##
## data:  dat_ord_wide[, 3:4]
## X-squared = 40.742, df = 4, p-value = 3.039e-08
```

Bägge testen förkastar nollhypotesen (som är att det inte finns några skillnader mellan verken i ordens relativa frekvenser), men det högre p-värdet i testet mot Gösta Berlings saga tyder på att det finns ett starkare samband mellan de texterna än mellan Röda rummet och den okända texten.

Kapitel 9

Variansanalys

Variansanalys (eller *anova-modellen*) är en statistisk modell där medelvärdet varierar beroende på en behandling och ett normalfördelat slumpfel. Från en anova-modell kan man beräkna ett F-test, som testar om det finns någon övergripande gruppskillnad, och post-hoc-test, som jämför specifika grupper med varandra.

Den specifika modellen beror på försöksupplägget. Här ges exempel på variansanalys med en faktor, en faktor med block, och två faktorer.

9.1 Variansanalys. En faktor

Vid variansanalys med en faktor har man ett upplägg där varje observation av en kontinuerlig utfallsvariabel är kopplad till en specifik grupp. Som exempel används en datamängd från ett odlingsförsök på havre. Datan finns tillgänglig i paketet `MASS` som `oats`. Försöket är ett två-faktoriellt försök med block. Faktorerna ges av havresort och kvävetillsättning; utfallsvariabeln är skördvikt. Som första exempel ignoreras sortvariabeln genom att beräkna medelvärde per block och kvävenivå.

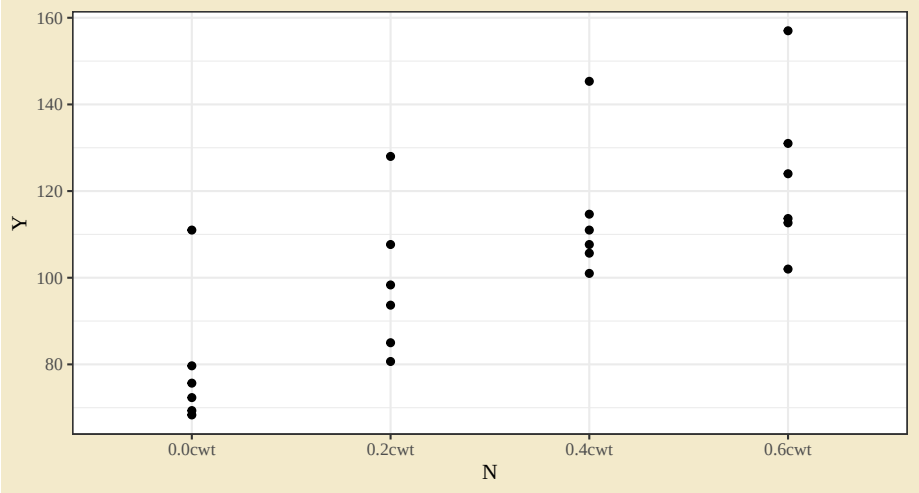
```
dat <- MASS::oats
dat <- dat %>%
  group_by(B, N) %>%
  summarise(Y = mean(Y)) %>%
  ungroup() %>%
  as_tibble()

kable(dat %>% pivot_wider(names_from = N, values_from = Y),
      digits = 1)
```

Datan har 18 observationer av skördvikt och varje observation tillhör någon

B	0.0cwt	0.2cwt	0.4cwt	0.6cwt
I	111.0	128.0	145.3	157.0
II	75.7	107.7	114.7	131.0
III	72.3	98.3	111.0	102.0
IV	69.3	93.7	105.7	124.0
V	68.3	80.7	101.0	113.7
VI	79.7	85.0	107.7	112.7

specifik kvävenivå. Datan kan illustreras med ett spridningsdiagram.

```
ggplot(dat, aes(N, Y)) +
  geom_point()
```



Det finns en tydlig kväveeffekt.

En anova-modell kan i R skattas med funktionen `lm` (för *linjär modell*). Från modellobjektet kan man sedan plocka fram en anova-tabell (som bland annat anger utfallet av F-testet) och genomföra parvisa jämförelser genom `emmeans`.

```
mod <- lm(Y ~ N, data = dat)
```

Modellen anges som en formel `Y ~ N`, vilket kan utläsas *Y beroende på faktorn N*. Detta följs av ett argument för det objekt som innehåller datan i kolumner (här `dat`).

För anova-tabellen används funktionen `Anova` från paketet `car`.

```
library(car)
Anova(mod)
```

```
## Anova Table (Type II tests)
```

```
##
## Response: Y
##           Sum Sq Df F value   Pr(>F)
## N         6673.5  3  7.5563 0.001434 **
## Residuals 5887.8 20
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Anova-tabellen beräknas ett F-test. Testet har nollhypotesen att samtliga grupper har samma populationsmedelvärde - det låga p-värdet tyder på att nollhypotesen bör förkastas, vilket alltså pekar på att det finns någon eller några skillnader i medelvärde.

För att göra parvisa jämförelse används paketet `emmeans` och funktionen med samma namn. Funktionen tar modellobjektet som första argument och en formel för jämförelsetyp som andra argument (här `pairwise ~ N`, en parvis jämförelse mellan nivåer i N).

```
library(emmeans)
emmeans(mod, pairwise ~ N)
```

```
## $emmeans
##  N      emmean SE df lower.CL upper.CL
##  0.0cwt   79.4  7 20     64.8       94
##  0.2cwt   98.9  7 20     84.3      114
##  0.4cwt  114.2  7 20     99.6      129
##  0.6cwt  123.4  7 20    108.8      138
##
## Confidence level used: 0.95
##
## $contrasts
##  contrast        estimate   SE df t.ratio p.value
##  0.0cwt - 0.2cwt   -19.50 9.91 20  -1.968  0.2328
##  0.0cwt - 0.4cwt   -34.83 9.91 20  -3.516  0.0107
##  0.0cwt - 0.6cwt   -44.00 9.91 20  -4.442  0.0013
##  0.2cwt - 0.4cwt   -15.33 9.91 20  -1.548  0.4293
##  0.2cwt - 0.6cwt   -24.50 9.91 20  -2.473  0.0953
##  0.4cwt - 0.6cwt    -9.17 9.91 20  -0.925  0.7917
##
## P value adjustment: tukey method for comparing a family of 4 estimates
```

I den nedre tabellen med jämförelser ges alla parvisa jämförelser. Nollhypotesen är att de två grupper som jämförs har samma medelvärde - ett lågt p-värde tyder alltså på att de två grupperna är signifikant skilda. Notera också att p-värden justeras med tukey-metoden, även känt som Tukeys HSD.

Parvisa jämförelser presenteras ofta med signifikansbokstäver (en *compact letter display, cld*). Dessa kan plockas fram med `multcomp`-paketet.

```
em <- emmeans(mod, pairwise ~ N)

library(multcomp)
cld(em, Letters = letters)
```

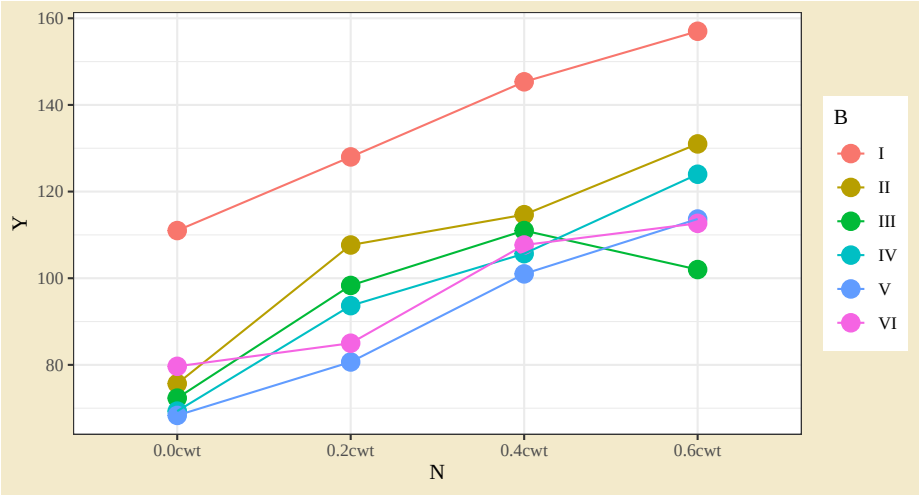
```
##  N      emmean SE df lower.CL upper.CL .group
##  0.0cwt   79.4  7 20     64.8       94  a
##  0.2cwt   98.9  7 20     84.3      114  ab
##  0.4cwt  114.2  7 20     99.6      129   b
##  0.6cwt  123.4  7 20    108.8      138   b
##
## Confidence level used: 0.95
## P value adjustment: tukey method for comparing a family of 4 estimates
## significance level used: alpha = 0.05
## NOTE: Compact letter displays can be misleading
##       because they show NON-findings rather than findings.
##       Consider using 'pairs()', 'pwpp()', or 'pwpm()' instead.
```

Tolkning av grupperingen till höger är att grupper som delar en bokstav inte är signifikant skilda. I det här fallet är den lägsta nivån skild från de två högsta. I övrigt finns inga signifikanta skillnader. Jämför gärna med p-värdena från tabellen med parvisa jämförelser. Man bör se att parvisa jämförelser med ett p-värde under fem procent motsvaras av att de behandlingarna inte delar någon bokstav i bokstavstabellen.

9.2 Variansanalys. En faktor med block

Modellen med en faktor var en förenkling av den faktiska försökssituationen. Som första utbyggnad av modellen noteras att försöket är ett blockförsök. En eventuell blockeffekt kan illustreras med ett punktdiagram kombinerat med ett linjediagram.

```
ggplot(dat, aes(N, Y, color = B, group = B)) +
  geom_point(size = 4) +
  geom_line()
```



Färg och linje sammanbinder observationer från samma block. Det finns en klar blockeffekt, vilket är särskilt tydligt för block I, som uppvisar klart högre värden än andra block.

Blockeffekten kan enkelt föras in i modellen genom att lägga till variabeln B i `lm`-funktionen. Anova-tabellen och parvisa jämförelser kan göras på samma sätt som tidigare, men nu tas blockeffekten i beaktande.

```
mod_bl <- lm(Y ~ N + B, data = dat)

Anova(mod_bl)
```

```
## Anova Table (Type II tests)
##
## Response: Y
##           Sum Sq Df F value    Pr(>F)
## N         6673.5  3  55.980 2.227e-08 ***
## B         5291.8  5  26.634 5.861e-07 ***
## Residuals  596.1 15
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

P-värdet från F-testet på variabeln N är nu klart mindre än tidigare. Detta beror på att en stor del av variationen kan förklaras med blockeffekten, vilket är tydligt i att blockeffekten också har ett litet p-värde i F-testet.

```
cld(emmeans(mod_bl, ~ N), Letters = letters)
```

```
##  N      emmean   SE df lower.CL upper.CL .group
##  0.0cwt   79.4 2.57 15     73.9     84.9  a
##  0.2cwt   98.9 2.57 15     93.4    104.4   b
##  0.4cwt  114.2 2.57 15    108.7    119.7    c
```

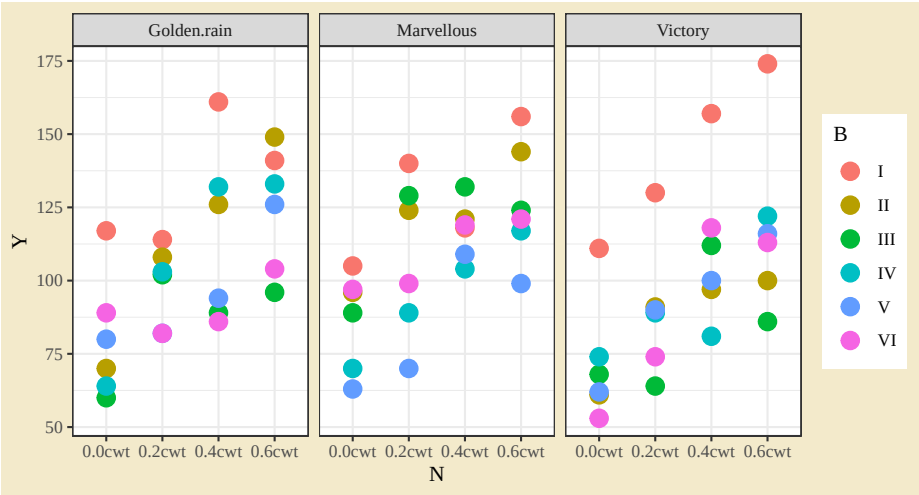
```
##  0.6cwt  123.4 2.57 15    117.9    128.9    c
##
## Results are averaged over the levels of: B
## Confidence level used: 0.95
## P value adjustment: tukey method for comparing a family of 4 estimates
## significance level used: alpha = 0.05
## NOTE: Compact letter displays can be misleading
##       because they show NON-findings rather than findings.
##       Consider using 'pairs()', 'pwpp()', or 'pwpm()' instead.
```

Även den parvisa jämförelsen påverkas av att ta med blocket. Signifikansbokstäver anger att den lägsta och näst lägsta nivån är skild från varandra och från de två högsta. En jämförelse med den tidigare tabellen över parvisa jämförelser visar att modellen med block ger samma medelvärdesskattningar men lägre medelfel (SE).

9.3 Variansanalys. Två faktorer med block

Den avslutande modellen tar med bägge faktorerna (sort och kväve) och blockfaktorn. Datan kan illustreras med ett punktdiagram där `facet_wrap` delar grafen efter sort.

```
dat <- MASS::oats
ggplot(dat, aes(N, Y, color = B)) +
  geom_point(size = 4) +
  facet_wrap(~ V)
```



Grafen visar samma kvävesamband som tidigare. Det finns inga tydliga skillnader sorter, möjligen har sorten Victory givit något lägre skörd än övriga. Det finns också en tydlig blockeffekt, till exempel har block I höga värden och block

V låga värden.

Modellen skattas genom att lägga till variabeln för sort (V för variety) i `lm`-formeln.

```
mod_two_fact <- lm(Y ~ N * V + B, data = dat)
```

Formeln är nu `Y ~ N * V + B`. Stjärnan mellan N och V anger modellen med en interaktion mellan sort och kväve. Eftersom varje kombination av sort och kväve förekommer en gång i varje block, är det inte möjligt att skatta någon interaktionseffekt med blockfaktorn - blocket är då istället en *additiv effekt*.

Anovatabellen kan plockas fram på samma sätt som tidigare.

```
Anova(mod_two_fact)
```

```
## Anova Table (Type II tests)
##
## Response: Y
##            Sum Sq Df F value    Pr(>F)
## N         20020.5  3 26.2510 1.135e-10 ***
## V          1786.4  2  3.5134   0.03665 *
## B         15875.3  5 12.4894 4.093e-08 ***
## N:V         321.8  6  0.2109   0.97187
## Residuals 13982.1 55
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Raden `N:V` gäller interaktionseffekten mellan kväve och sort. I det här fallet är det ingen signifikant interaktion - vilket tyder på att sorterna svarar på kvävebehandling på liknande sätt. Samtliga huvudeffekter (raderna för N, V och B) är signifikanta. Kvadratsummorna och p-värdena tyder på att kväve förklarar mer av variationen än sort, vilket också är i linje med grafen ovan.

Vid flerfaktoriella försök kan man presentera parvisa jämförelser på flera olika sätt. Man kan ange huvudeffekter för en faktor utan att ange den andra faktorn, man kan ange medelvärden för samtliga kombinationer av två faktorer, och man kan ange medelvärden uppdelat efter nivåer i en annan faktor.

```
emmeans(mod_two_fact, ~ N)
```

```
##  N      emmean   SE df lower.CL upper.CL
##  0.0cwt   79.4 3.76 55     71.9     86.9
##  0.2cwt   98.9 3.76 55     91.4    106.4
##  0.4cwt  114.2 3.76 55    106.7    121.8
##  0.6cwt  123.4 3.76 55    115.9    130.9
##
## Results are averaged over the levels of: V, B
## Confidence level used: 0.95
```

```
emmeans(mod_two_fact, ~ N + V)
```

```
##  N      V           emmean   SE df lower.CL upper.CL
##  0.0cwt Golden.rain   80.0 6.51 55     67.0     93.0
##  0.2cwt Golden.rain   98.5 6.51 55     85.5    111.5
##  0.4cwt Golden.rain  114.7 6.51 55    101.6    127.7
##  0.6cwt Golden.rain  124.8 6.51 55    111.8    137.9
##  0.0cwt Marvellous    86.7 6.51 55     73.6     99.7
##  0.2cwt Marvellous   108.5 6.51 55     95.5    121.5
##  0.4cwt Marvellous   117.2 6.51 55    104.1    130.2
##  0.6cwt Marvellous   126.8 6.51 55    113.8    139.9
##  0.0cwt Victory       71.5 6.51 55     58.5     84.5
##  0.2cwt Victory       89.7 6.51 55     76.6    102.7
##  0.4cwt Victory      110.8 6.51 55     97.8    123.9
##  0.6cwt Victory      118.5 6.51 55    105.5    131.5
##
## Results are averaged over the levels of: B
## Confidence level used: 0.95
```

```
emmeans(mod_two_fact, ~ N | V)
```

```
## V = Golden.rain:
##  N      emmean   SE df lower.CL upper.CL
##  0.0cwt   80.0 6.51 55     67.0     93.0
##  0.2cwt   98.5 6.51 55     85.5    111.5
##  0.4cwt  114.7 6.51 55    101.6    127.7
##  0.6cwt  124.8 6.51 55    111.8    137.9
##
## V = Marvellous:
##  N      emmean   SE df lower.CL upper.CL
##  0.0cwt   86.7 6.51 55     73.6     99.7
##  0.2cwt  108.5 6.51 55     95.5    121.5
##  0.4cwt  117.2 6.51 55    104.1    130.2
##  0.6cwt  126.8 6.51 55    113.8    139.9
##
## V = Victory:
##  N      emmean   SE df lower.CL upper.CL
##  0.0cwt   71.5 6.51 55     58.5     84.5
##  0.2cwt   89.7 6.51 55     76.6    102.7
##  0.4cwt  110.8 6.51 55     97.8    123.9
##  0.6cwt  118.5 6.51 55    105.5    131.5
##
## Results are averaged over the levels of: B
## Confidence level used: 0.95
```

Även här kan man göra jämförelser mellan nivåer genom att sätta `pairwise ~ N + V` eller beräkna signifikansbokstäver med `cld`.

Behandling	Hållfasthet
Ny	16.85
Ny	16.40
Ny	17.21
Ny	16.36
Ny	16.52
Ny	17.04
Ny	16.96
Ny	17.15
Ny	16.59
Ny	16.57
Standard	17.50
Standard	17.63
Standard	18.25
Standard	18.00
Standard	17.86
Standard	17.75
Standard	18.22
Standard	17.90
Standard	17.96
Standard	18.15

9.4 Övningar

Övning 9.1 (Jämförelse mellan t-test och anova)**.** Detta är ett datamaterial som jämför hållfastheten hos murbruk för två oberoende stickprov.

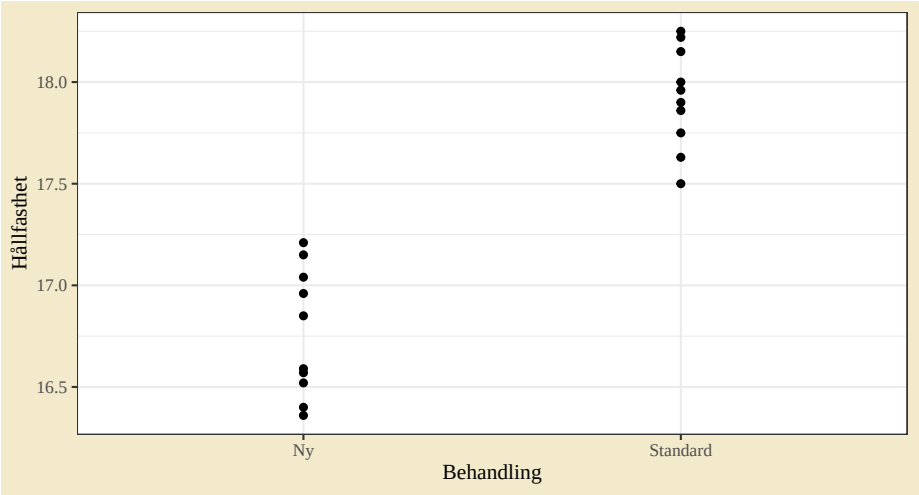
a. Gör ett t-test för att se om det är någon skillnad mellan de två behandlingarna. Anta att varianserna är lika för de två behandlingarna.

b. Använd envägs-anova för att se om det är skillnad. Är modellantaganden om normalfördelade *residualer* och lika varians inom grupper uppfyllda?

c. Jämför resultaten i (a) och (b).

Lösningsförslag 9.1 (Jämförelse mellan t-test och anova)**.** a.

```
dat <- tibble(Behandling = rep(c("Ny", "Standard"), each = 10),
              Hållfasthet = c(16.85,16.40,17.21,16.36,16.52,
                              17.04,16.96,17.15,16.59,16.57,
                              17.50,17.63,18.25,18.00,17.86,
                              17.75,18.22,17.90,17.96,18.15))

ggplot(dat, aes(Behandling, Hållfasthet)) +
  geom_point()
```



En graf visar på en mycket tydlig behandlingsskillnad. Det finns inga tydliga extremvärden och grupperna verkar ha samma varians inom gruppen.

```
t.test(Hållfasthet ~ Behandling, dat, var.equal = T)
```

```
##
##  Two Sample t-test
##
## data:  Hållfasthet by Behandling
## t = -9.1272, df = 18, p-value = 3.572e-08
## alternative hypothesis: true difference in means between group Ny and group Standard
## 95 percent confidence interval:
##  -1.4233204 -0.8906796
## sample estimates:
##       mean in group Ny mean in group Standard
##                 16.765                 17.922
```

Ett mycket lågt p-värde tyder på att det finns en skillnad mellan grupperna.

b. Modellen kan skattas med `lm`. Modellekvationen ges av `Hållbarhet ~ Behandling`, vilket kan utläsas som *hållbarhet beroende på behandling*.

```
mod <- lm(Hållfasthet ~ Behandling, dat)

library(car)
Anova(mod)
```

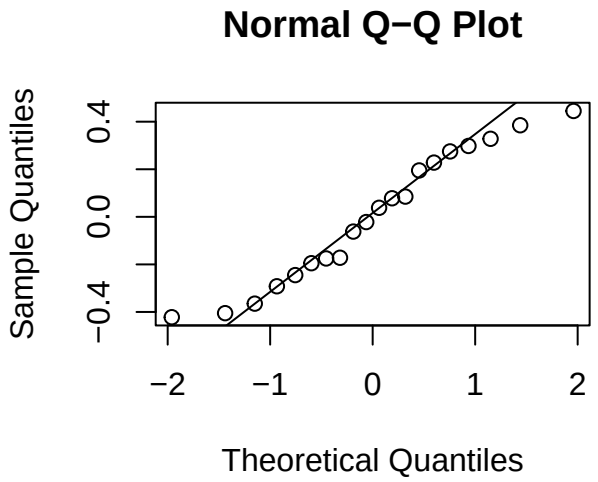
```
## Anova Table (Type II tests)
##
## Response: Hållfasthet
##             Sum Sq Df F value    Pr(>F)
## Behandling 6.6932  1  83.306 3.572e-08 ***
```

```
## Residuals  1.4462 18
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Ett mycket lågt p-värde tyder på att det finns en skillnad mellan grupperna.

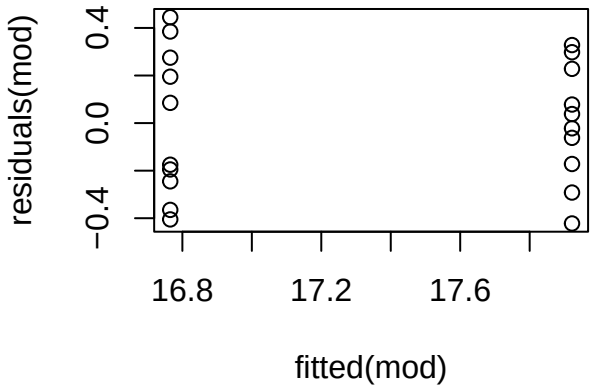
Modellantaganden (normalfördelning och lika varianser inom grupper) kan undersökas med diagnosplottar. QQ-grafen visar eventuella avvikelser från en normalfördelning (om data är normalfördelad följer punkterna diagonalen) och spridningsdiagram visar eventuella skillnader i spridning.

```
qqnorm(residuals(mod))
qqline(residuals(mod))
```




```
plot(residuals(mod) ~ fitted(mod))
```

Sort	Angrepp
A	72.3
A	59.5
A	42.4
B	44.7
B	38.2
B	46.8
C	48.7
C	43.1
C	54.8
D	52.4
D	61.7
D	67.3



Diagnostik-grafer pekar inte på några extrema avvikelser från normalantagandet (punkterna följer den diagonala linjen ganska väl) eller antagandet om lika varianser (de två kolumnerna med punkter har ungefär samma spridning).

c. F-testet i (b) ger samma p-värde som t-test i (a).

Övning 9.2 (Äppelinfektion)**.** En studie har givit ett mått på infektion hos äppelträd. Fyra sorter jämförs med tre replikat per sort.

a. Skatta anova-modellen och ta fram anova-tabellen.

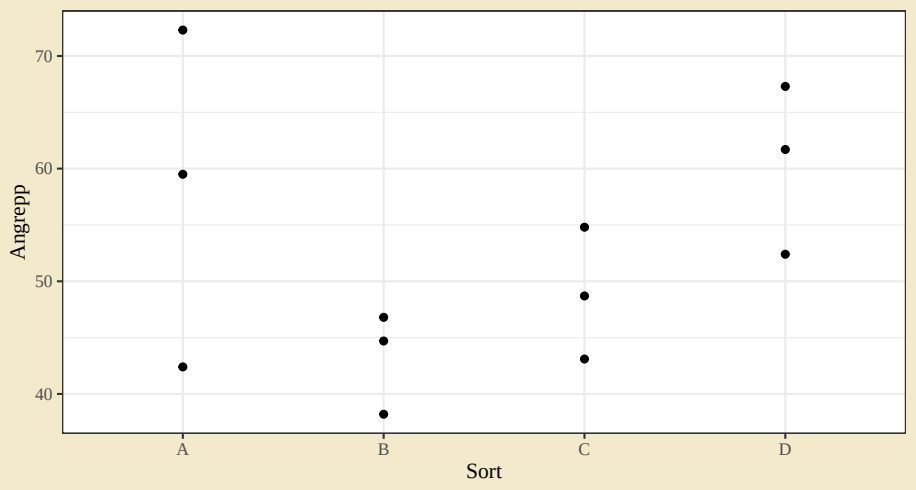
b. Undersök om residualerna är normalfördelade.

c. Jämför sorter med Tukeys HSD.

d. Jämför sorter med Fishers LSD.

Lösningsförslag 9.2 (Äppelinfektion)**.** Data kan läsas in från excelfilen med uppgiftsdata.

```
library(readxl)
dat <- read_excel("Data/Uppgiftsdata.xlsx", sheet = "Äppelangrepp")

ggplot(dat, aes(Sort, Angrepp)) +
  geom_point()
```



Grafen visar svaga tecken på skillnader, men inga *tydliga* mönster.

a.

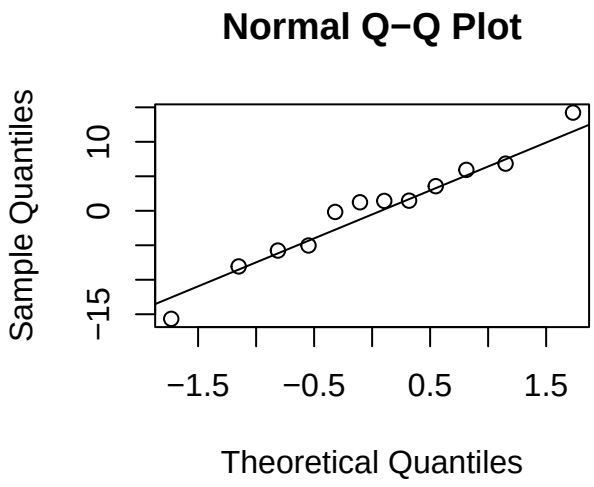
```
mod <- lm(Angrepp ~ Sort, dat)

library(car)
Anova(mod)
```

```
## Anova Table (Type II tests)
##
## Response: Angrepp
##           Sum Sq Df F value Pr(>F)
## Sort      580.28  3  2.3025 0.1538
## Residuals 672.07  8
```

F-testet testar nollhypotesen att alla grupper har samma populationsmedelvärde. Det höga p-värdet ger att det inte finns signifikanta skillnader mellan sorter.

b.

```
qqnorm(residuals(mod))
qqline(residuals(mod))
```



Ungefärligt normalfördelat.

c.

```
library(emmeans)
multcomp::cld(emmeans(mod, ~ Sort)) # Tukey
```

```
##  Sort emmean   SE df lower.CL upper.CL .group
##  B      43.2 5.29  8     31.0     55.4  1
##  C      48.9 5.29  8     36.7     61.1  1
##  A      58.1 5.29  8     45.9     70.3  1
##  D      60.5 5.29  8     48.3     72.7  1
##
## Confidence level used: 0.95
## P value adjustment: tukey method for comparing a family of 4 estimates
## significance level used: alpha = 0.05
## NOTE: Compact letter displays can be misleading
##       because they show NON-findings rather than findings.
##       Consider using 'pairs()', 'pwpp()', or 'pwpm()' instead.
```

Tukey-testet pekar på att det inte finns några skillnader mellan sorterna.

d.

```
multcomp::cld(emmeans(mod, ~ Sort, adjust = "none")) # Fisher
```

```
##  Sort emmean   SE df lower.CL upper.CL .group
```

Hybrid	Nordväst	Nordost	Centralt	Sydost	Sydväst
FR-11	62	64	64	65	66
BCM	63	63	66	67	64
DBC	61	64	65	62	65
RC-3	55	56	60	58	59

```
##  B      43.2 5.29  8     31.0     55.4  1
##  C      48.9 5.29  8     36.7     61.1  1
##  A      58.1 5.29  8     45.9     70.3  1
##  D      60.5 5.29  8     48.3     72.7  1
##
## Confidence level used: 0.95
## P value adjustment: tukey method for comparing a family of 4 estimates
## significance level used: alpha = 0.05
## NOTE: Compact letter displays can be misleading
##       because they show NON-findings rather than findings.
##       Consider using 'pairs()', 'pwpp()', or 'pwpm()' instead.
```

Fishers LSD pekar på att det inte finns några skillnader mellan sorterna.

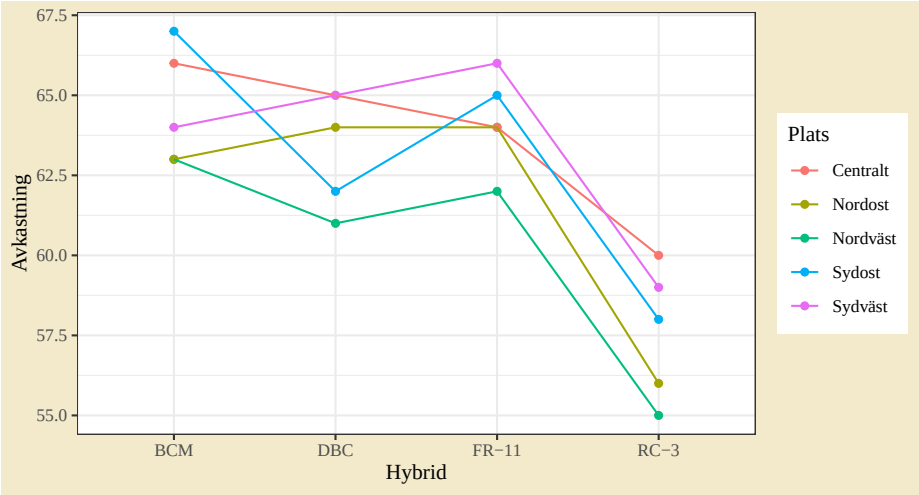
Övning 9.3 (Majshybrider)**.** Fyra majssorter planteras på fem platser (som agerar som fem block). Datan ges av följande tabell.

a. Skatta en anova-modell med block och ta fram anova-tabellen. Finns det signifikanta skillander mellan hybrider? Mellan block?

b. Använd Tukey-metoden för parvisa jämförelser mellan hybrider.

Lösningsförslag 9.3 (Majshybrider)**.** Data kan läsas in från excelfilen med uppgiftsdata.

```
dat <- read_excel("Data/Uppgiftsdata.xlsx", sheet = "Majshybrider")

ggplot(dat, aes(Hybrid, Avkastning, group = Plats, col = Plats)) +
  geom_point() +
  geom_line()
```



Tecken på både platseffekt (nordväst alltid lägst) och hybrideffekt (RC3 lägre än övriga).

```
mod <- lm(Avkastning ~ Hybrid + Plats, dat)
Anova(mod)
```

```
## Anova Table (Type II tests)
##
## Response: Avkastning
##           Sum Sq Df F value    Pr(>F)
## Hybrid    160.55  3 34.3422 3.607e-06 ***
## Plats      33.70  4  5.4064   0.01004 *
## Residuals  18.70 12
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
multcomp::cld(emmeans(mod, ~ Hybrid), Letters = letters)
```

```
##  Hybrid emmean    SE df lower.CL upper.CL .group
##  RC-3     57.6 0.558 12     56.4     58.8  a
##  DBC      63.4 0.558 12     62.2     64.6   b
##  FR-11    64.2 0.558 12     63.0     65.4   b
##  BCM      64.6 0.558 12     63.4     65.8   b
##
## Results are averaged over the levels of: Plats
## Confidence level used: 0.95
## P value adjustment: tukey method for comparing a family of 4 estimates
## significance level used: alpha = 0.05
## NOTE: Compact letter displays can be misleading
##       because they show NON-findings rather than findings.
##       Consider using 'pairs()', 'pwpp()', or 'pwpm()' instead.
```

Maskin	1	2	3	4	5	6
A	34	38	32	41	41	36
B	30	31	33	40	39	35
C	27	30	29	31	36	32

Klart signifikanta skillnader mellan hybrider. Post-hoc-tester ger att RC-3 har lägre avkastning av övriga.

Övning 9.4 (Maskiner med och utan block)**.** I en fabrik testas tre olika maskiner i produktionen. Utfallsvariabeln är hur mycket tid maskinen behöver för tryckpressa en stol. Man vill undersöka om det är en skillnad mellan maskiner. Resultatet ges nedan.

a. Antag att den som gjorde experimentet randomiserade ordningen på de 18 försöken och gjorde alla under en dag. Analysera försöket med envägs anova-modell för att se om det är någon skillnad mellan maskinerna.

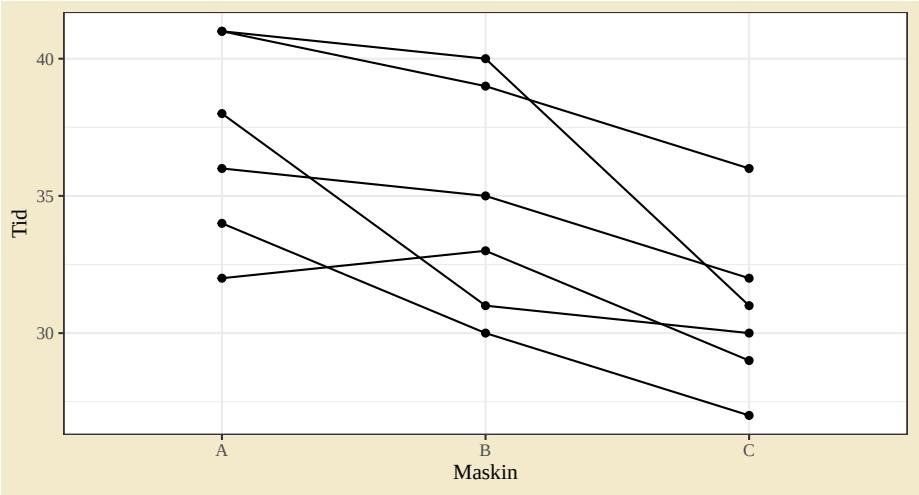
b. Om där är en skillnad, vilka maskiner skiljer sig åt?

c. Vilka antaganden gjordes i analysen? Undersök om dessa är uppfyllda genom att använda lämpliga diagnosgrafer.

d. Någon säger att försöket inte är gjort under en dag utan att replikat i själva verket anger vilken dag som försöket gjordes. Man vill därför använda replikat som block i försöket. Gör detta och undersök om detta förändrar resultatet. Tror du att replikat anger olika dagar?

Lösningsförslag 9.4 (Maskiner med och utan block)**.** Datan kan skrivas in som en `tibble` och illustreras med ett enkelt spridningsdiagram.

```
dat <- tibble(Maskin = rep(c("A", "B", "C"), each = 6),
              Replikat = rep(1:6, 3),
              Tid = c(34,38,32,41,41,36,
                      30,31,33,40,39,35,
                      27,30,29,31,36,32))

ggplot(dat, aes(Maskin, Tid, group = Replikat)) +
  geom_point() +
  geom_line()
```



Det finns tecken på skillnader i tid (Maskin C ligger lägre).

a. En enkel anova-modell skattas för att testa eventuella maskinskillnader.

```
mod <- lm(Tid ~ Maskin, dat)
Anova(mod)
```

```
## Anova Table (Type II tests)
##
## Response: Tid
##           Sum Sq Df F value  Pr(>F)
## Maskin    116.33  2  4.3589 0.03218 *
## Residuals 200.17 15
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

F-testet för faktorn Maskin ger ett p-värde på 0.03. Eftersom det är under fem procent förkastas nollhypotesen att maskinerna ger samma medelvärde.

b.

```
emmeans(mod, pairwise ~ Maskin)
```

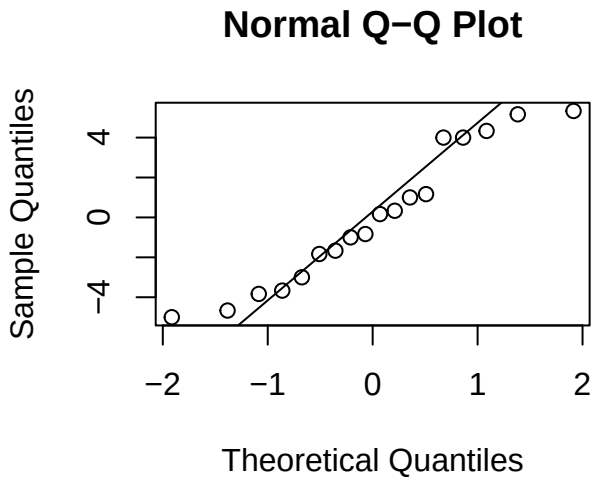
```
## $emmeans
##  Maskin emmean   SE df lower.CL upper.CL
##  A        37.0 1.49 15     33.8     40.2
##  B        34.7 1.49 15     31.5     37.8
##  C        30.8 1.49 15     27.7     34.0
##
## Confidence level used: 0.95
##
## $contrasts
```

```
##  contrast estimate   SE df t.ratio p.value
##  A - B        2.33 2.11 15   1.106  0.5249
##  A - C        6.17 2.11 15   2.924  0.0267
##  B - C        3.83 2.11 15   1.818  0.1976
##
## P value adjustment: tukey method for comparing a family of 3 estimates
```

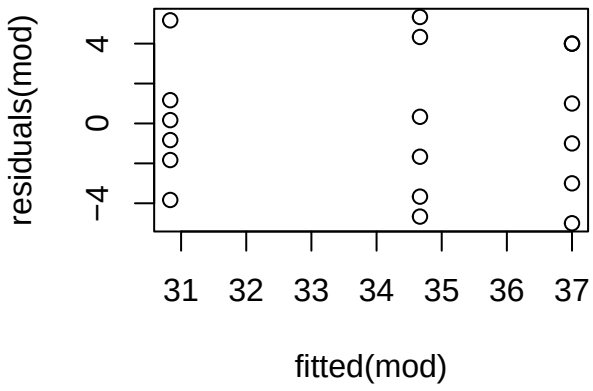
Post-hoc-tester visar på en skillnad mellan maskin A och C.

c. Modellen bygger på antagandet att residualerna är normalfördelade och att grupperna har samma varians.

```
qqnorm(residuals(mod))
qqline(residuals(mod))
```




```
plot(residuals(mod) ~ fitted(mod))
```



En graf över residualerna visar inte på några tydliga brister i antaganden.

d. Replikat används som en block-faktor i försöket. För att R ska tolka variabeln Replikat korrekt ändras dess typ till `character`.

```
dat <- dat %>% mutate(Replikat = as.character(Replikat))
mod <- lm(Tid ~ Maskin + Replikat, dat)

Anova(mod)
```

```
## Anova Table (Type II tests)
##
## Response: Tid
##           Sum Sq Df F value    Pr(>F)
## Maskin    116.33  2 15.7207 0.0008181 ***
## Replikat  163.17  5  8.8198 0.0019704 **
## Residuals  37.00 10
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
emmeans(mod, pairwise ~ Maskin)
```

```
## $emmeans
##  Maskin emmean    SE df lower.CL upper.CL
##  A        37.0 0.785 10     35.3     38.7
##  B        34.7 0.785 10     32.9     36.4
##  C        30.8 0.785 10     29.1     32.6
##
## Results are averaged over the levels of: Replikat
```

```
## Confidence level used: 0.95
##
## $contrasts
##  contrast estimate   SE df t.ratio p.value
##  A - B        2.33 1.11 10   2.101  0.1393
##  A - C        6.17 1.11 10   5.553  0.0006
##  B - C        3.83 1.11 10   3.452  0.0156
##
## Results are averaged over the levels of: Replikat
## P value adjustment: tukey method for comparing a family of 3 estimates
```

Om replikat tas med i modellen förtydligas behandlingseffekten. F-testet ger en mycket starkare signifikans än tidigare och post-hoc-testet ger en signifikant skillnad mellan B och C (utöver den tidigare signifikanta skillnaden mellan A och C).

Kapitel 10

Regression och korrelation

Regression och korrelation är metoder för att mäta ett samband mellan två kontinuerliga variabler. Regression skattar en variabel som en funktion av en annan, t.ex. kan man skatta en modell av en plantas höjd som en funktion av näringsinnehåll i marken. Korrelation är ett mått på samvariation mellan två variabler.

10.1 Regression

I en regression modelleras en variabel som en funktion av en annan variabel. Vid enkel linjär regression finns *en* sådan *förklarande variabel* och förhållandet mellan variablerna antas vara linjärt. Modellen kan uttryckas

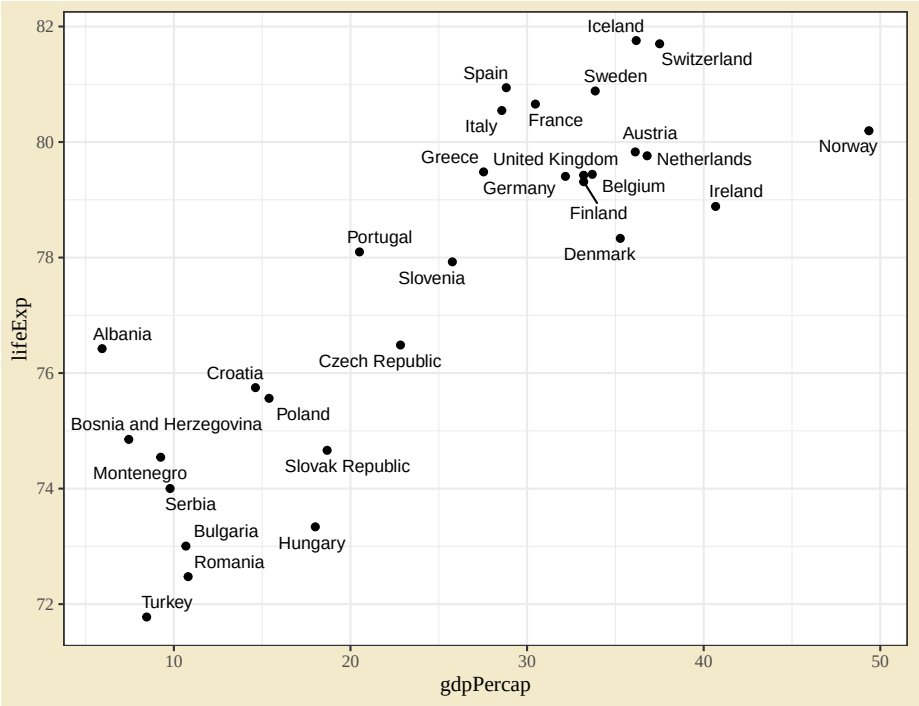
$$y_i = \beta_0 + \beta_1 x_i + \varepsilon_i,$$

där y_i är observation i av den förklarade variabeln, β_0 och β_1 är parametrar, x_i är observation i av den förklarande variabeln, och ε_i är en slumpmässig felterm.

Ta som exempel data på förväntad medellivslängd och bnp per capita. Datan hämtas från `gapminder`-paketet. Paketet `ggrepel` kan användas för att sätta punktetiketter som inte överlappar. För enklare tolkning av modellen transformeras bnp per capita till att vara i tusen dollar, snarare än dollar.

```
library(gapminder)
dat <- gapminder %>%
  filter(year == 2007, continent == "Europe") %>%
  mutate(gdpPercap = gdpPercap / 1000)

library(ggrepel)
ggplot(dat, aes(gdpPercap, lifeExp)) +
```

```
  geom_point() +
  geom_text_repel(aes(label = country), size = 3)
```



Datan visar ett positivt samband mellan variablerna - högre bnp per capita är kopplat till högre medelllivslängd. En regressionmodell kan i R skattas med `lm`-funktionen. Syntaxen är väldigt lik den för anovamodellen, men istället för en faktor som förklarande variabel används nu en kontinuerlig variabel.

```
mod <- lm(lifeExp ~ gdpPercap, data = dat)

summary(mod)
```

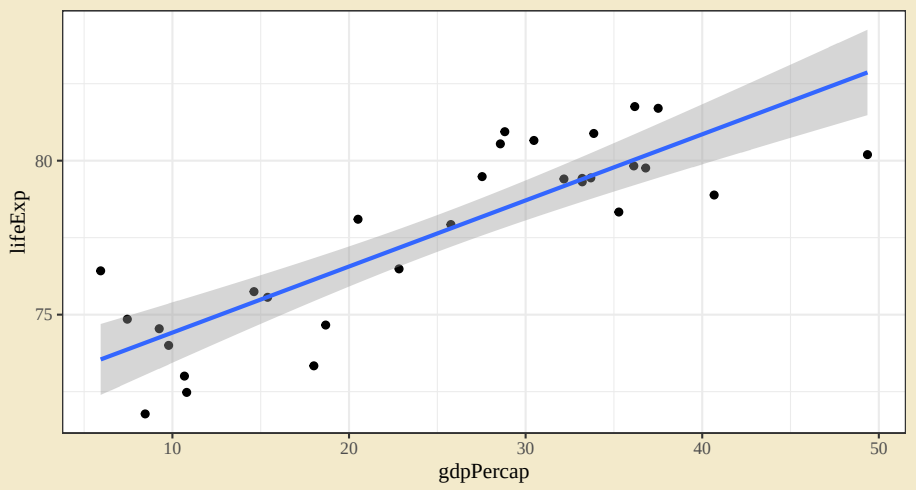
```
##
## Call:
## lm(formula = lifeExp ~ gdpPercap, data = dat)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -2.79839 -1.30472  0.00807  1.33443  2.87766
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 72.27106    0.69416 104.113  < 2e-16 ***
```

```
## gdpPercap    0.21463    0.02514   8.537  2.8e-09 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.598 on 28 degrees of freedom
## Multiple R-squared:  0.7225, Adjusted R-squared:  0.7125
## F-statistic: 72.88 on 1 and 28 DF,  p-value: 2.795e-09
```

Funktionen `summary` ger en sammanfattning av modellen. Skattningen av parametern β_0 ges som raden `(Intercept)` och dess tolkning är som förväntat värde i medellivslängd om bnp per capita är noll. Det är ofta lutningsparametern som är mer intressant. Skattningen av β_1 ges på den rad som har samma namn som den förklarande variabeln, här `gdpPercap`. Den skattade parametern är 0.2146. Lutningsparametern har den generella tolkning som ökningen i y-variabeln när x-variabeln ökar med 1. I det här fallet ger 0.2146 att ett lands medellivslängd ökar med ungefär 78 dagar när bnp per capita ökar med 1000 dollar.

Man kan rita ut regressionlinjen i en graf med `geom_smooth` och argumentet `method` satt till `lm"`.

```
ggplot(dat, aes(gdpPercap, lifeExp)) +
  geom_point() +
  geom_smooth(method = lm)
```



Den blå linjen illustrerar regressionlinjen $72.27 + 0.2146 \cdot x$. Det grå bandet kring linjen är ett konfidensintervall för skattningen av y-variabeln vid ett visst x-värde.

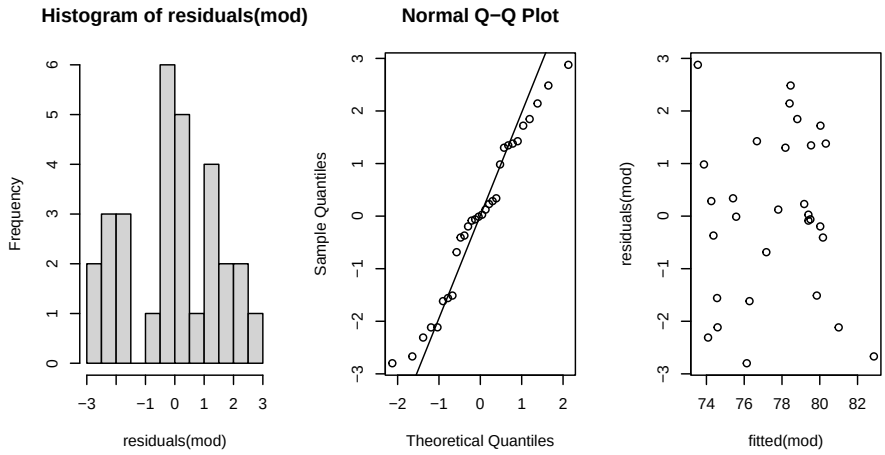
Utskriften från `summary` ger också tester av parametrarna (den högra kolumnen `Pr(>|t|)` ger p-värdet för ett test där nollhypotesen är att populationsparametern är noll). I det här fallet är både intercept och lutning skilda från noll. Motsvarande F-test för lutningen kan tas fram med en anova-tabell.

```
library(car)
Anova(mod)
```

```
## Anova Table (Type II tests)
##
## Response: lifeExp
##             Sum Sq Df F value    Pr(>F)
## gdpPercap 186.031  1  72.883 2.795e-09 ***
## Residuals  71.469 28
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Testerna av en regressionmodell bygger på ett normalfördelningsantagande oh ett antagande om homoskedasticitet (lika varians i y oavsett position på x-axeln). Antagandena kan undersökas genom att titta på skattningens *residualer* - skillnaden mellan det faktiska y-värdet och modellens värde. Residualerna kan undersökas med ett histogram eller en QQ-plot. En annan vanlig diagnosplot är ett spridningsdiagram med skattade värden på x-axeln och residualerna på y-axeln.

```
hist(residuals(mod), breaks = 10)
qqnorm(residuals(mod)); qqline(residuals(mod))
plot(residuals(mod) ~ fitted(mod))
```



Om data följer en normalfördelning bör histogrammet visa en ungefärlig normalkurva, QQ-plotten bör visa punkter på den diagonala linjen och spridningsdiagrammet bör visa en slumpmässig spridning av punkter. Graferna pekar i det här fallet inte på några tydliga avvikelser från normalfördelningsantagandet, möjligen pekar QQ-plotten på mindre spridning i svansarna än en teoretisk normalfördelning.

10.2 Korrelation

Korrelation ger ett mått mellan -1 och 1 på hur väl två variabler samvarierar. En korrelation över noll tyder på ett positivt samband mellan variablerna - en observation med ett högt värde i den ena variabeln har också ett högt värde på den andra - medan en korrelation under noll tyder på ett negativt samband. I R kan korrelation beräknas med `cor` och två variabler som första och andra argument. Funktionen `cor.test` ger ett test där nollhypotesen är att korrelationen är noll.

```
cor(dat$lifeExp, dat$gdpPercap)
```

```
## [1] 0.8499711
```

```
cor.test(dat$lifeExp, dat$gdpPercap)
```

```
##
##  Pearson's product-moment correlation
##
## data:  dat$lifeExp and dat$gdpPercap
## t = 8.5372, df = 28, p-value = 2.795e-09
## alternative hypothesis: true correlation is not equal to 0
## 95 percent confidence interval:
##  0.7058444 0.9265221
## sample estimates:
##       cor
## 0.8499711
```

Medellivslängd och bnp per capita har en stark positiv korrelation på 0.85 och den korrelation är signifikant skild från noll ($p = 2.795 \cdot 10^{-9}$). Notera att p-värdet är detsamma som testet av lutningsparametern i regressionen.

10.3 Övningar

Övning 10.1 (Blodtryck)**.** Följande fascinerande blodtrycksdata hämtas från kvinnor i Sala.

a. Skatta en enkel linjär regressionsmodell och tolka lutningskoefficienten i termer av de ursprungliga variablerna.

b. Beräkna ett konfidensintervall för lutningskoefficienten.

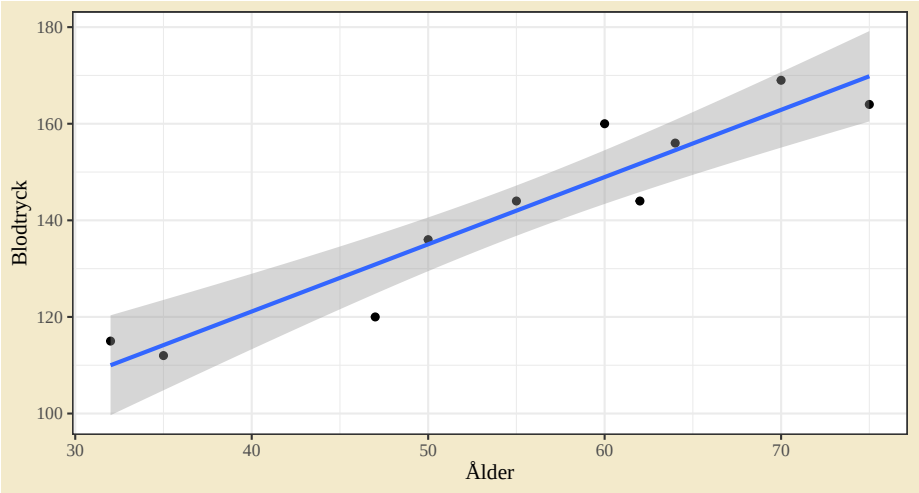
c. Undersök modellens antaganden (normalfördelade residualer, lika varians för skilda nivåer av x-variabeln).

d. Testa om lutningskoefficienten är skild från 1.

Lösningsförslag 10.1 (Blodtryck)**.** Data kan läsas in från excelfilen med uppgiftsdata. Två kontinuerliga variabler kan enklast illustreras med ett spridningsdiagram.

Ålder	Blodtryck
32	115
35	112
47	120
50	136
55	144
60	160
62	144
64	156
70	169
75	164

```
dat <- read_excel("Data/Uppgiftsdata.xlsx", sheet = "Blodtryck")

ggplot(dat, aes(Ålder, Blodtryck)) +
  geom_point() +
  geom_smooth(method = lm)
```



Det finns ett tydligt positivt samband mellan variablerna.

```
mod <- lm(Blodtryck ~ Ålder, dat)
summary(mod)
```

```
##
## Call:
## lm(formula = Blodtryck ~ Ålder, data = dat)
##
## Residuals:
```

```
##     Min      1Q  Median      3Q     Max
## -10.868  -4.915   1.217   4.254  11.042
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  65.4650     9.5384   6.863 0.000129 ***
## Ålder         1.3915     0.1685   8.259 3.47e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 7.145 on 8 degrees of freedom
## Multiple R-squared:  0.895,  Adjusted R-squared:  0.8819
## F-statistic: 68.21 on 1 and 8 DF,  p-value: 3.47e-05
```

Regressionslinjen ges av $y = 65.465 + 1.3915$.

a. Blodtrycket ökar med 1.39 för varje ökat år.

b. Ett konfidensintervall kan tas fram med `confint`.

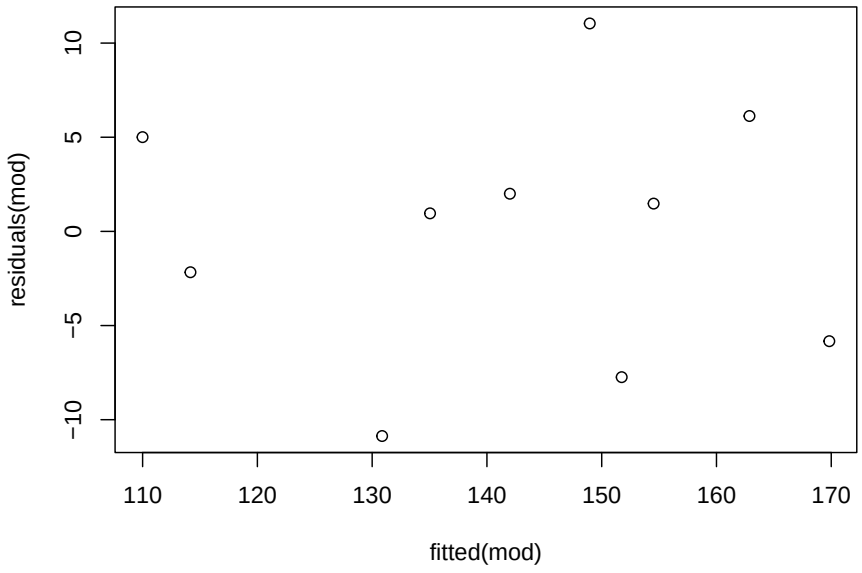
```
confint(mod)
```

```
##                 2.5 %    97.5 %
## (Intercept) 43.469261 87.460661
## Ålder        1.003005  1.780088
```

Ett 95-procentigt konfidensintervall för lutning ges av $(1.003, 1.780)$.

c. Modellen bygger på normalfördelade residualer och lika varians för skilda nivåer av ålder.

```
plot(residuals(mod) ~ fitted(mod))
```



Det finns inga extremvärden bland residualer och inga tydliga tecken på skillnader i varians.

d. Eftersom 1 ligger precis utanför ett 95-procentigt konfidensintervall, bör ett test på femprocentsnivån leda till att man förkastar nollhypotesen att lutningen är 1. Detta kan testas formellt genom `emtrends` från paketet `emmeans`.

```
test(emtrends(mod, ~ 1, "Ålder"), null = 1)
```

```
##  1       Ålder.trend    SE df null t.ratio p.value
##  overall        1.39 0.168  8    1   2.324  0.0486
```

P-värdet ligger precis under fem procent.

Övning 10.2 (Beach 2050)**.** Bland kursdata finns en fil med uppmätta temperaturer vid väderstationen i Falsterbo. Den kan läsas in från csv-filen. Se tidigare uppgift om beskrivande statistik för detaljer.

```
dat <- read_csv2("Data/smhi-opendata_1_52230_20210912_114534.csv", skip = 9) %>%
  mutate(Lufttemperatur = as.numeric(Lufttemperatur))
```

En sammanställning av medeltemperaturer de senaste fyrtio åren ges av följande tabell.

a. Skapa en passande graf med år på x-axeln och medeltemperatur på y-axeln. I ggplot kan en skattad regressionlinje illustreras genom `geom_smooth(method`

År	Medeltemperatur
1981	7.92
1982	8.53
1983	9.03
1984	8.38
1985	6.61
1986	7.20
1987	6.60
1988	8.76
1989	9.76
1990	9.91
1991	8.61
1992	9.55
1993	8.12
1994	9.26
1995	9.00
1996	7.32
1997	8.99
1998	8.58
1999	9.29
2000	9.72
2001	8.98
2002	9.51
2003	8.89
2004	8.92
2005	9.01
2006	9.63
2007	9.80
2008	9.81
2009	9.17
2010	7.79
2011	9.37
2012	9.20
2013	10.08
2014	11.53
2015	10.73
2016	10.58
2017	10.24
2018	11.42
2019	11.36
2020	11.44

= lm).

b. Skatta en regressionsmodell med år som förklarande variabel och medeltemperatur som förklarad variabel. Genomför ett t-test eller F-test för att se om lutningskoefficienten är skild från noll.

c. Funktionen `predict` kan användas för att skapa en skattning av populationsmedelvärdet för valfritt värde på den förklarande variabeln. Titta på hjälpsidan för `predict.lm` och använd funktionen för att förutsäga medeltemperatur för varje år fram till 2050. Illustrera skattningarna i en passande graf.

d. Vilken kritik finns mot den här modellen?

Lösningsförslag 10.2 (Beach 2050)**.** Data läses in med koden i uppgiften.

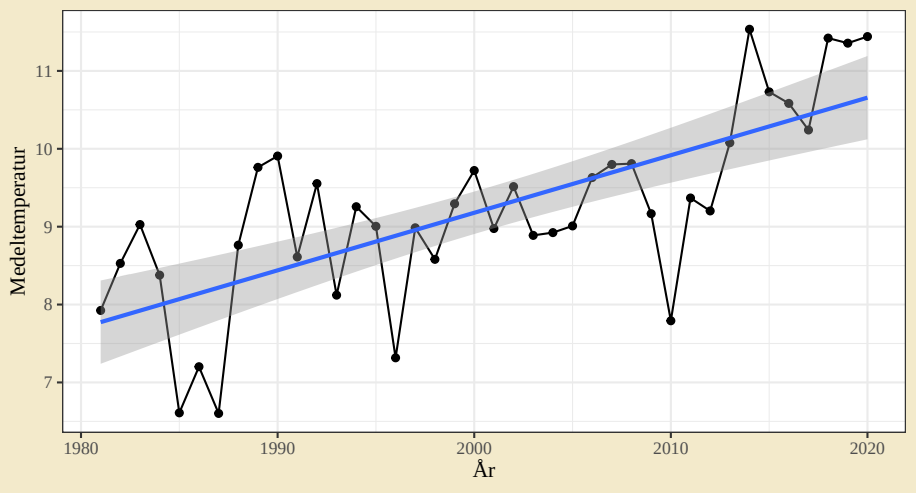
```
dat <- read_csv2("Data/smhi-opendata_1_52230_20210912_114534.csv", skip = 9) %>%
  mutate(Lufttemperatur = as.numeric(Lufttemperatur))
```

Årsmedelvärden kan beräknas genom att skapa en variabel för År med `year` från paketet `lubridate`. Därefter grupperas efter år och summeras med medelvärdet. Slutligen filteras datan för att ta fram de senaste 40 åren. Datan sparas under namnet `dat_medel`.

```
dat_medel <- dat %>%
  mutate(År = lubridate::year(Datum)) %>%
  group_by(År) %>%
  summarise(Medeltemperatur = mean(Lufttemperatur)) %>%
  filter(År > 1980, År < 2021)
```

a. Tidsdata illustreras ofta med en linjediagram. Regressionslinjen kan skapas med `geom_smooth(method = lm)`.

```
g1 <- ggplot(dat_medel, aes(År, Medeltemperatur)) +
  geom_line() +
  geom_point() +
  geom_smooth(method = lm)
g1
```



Det finns tecken på en ökning över tid.

b. Regressionmodellen skattas med funktionen `lm`. Medeltemperatur är den förklarade variabeln och år den förklarande - regressionsformeln skrivs därmed `Medeltemperatur ~ År`. Modellen har automatiskt ett intercept, men man hade även kunnat skriva `Medeltemperatur ~ 1 + År` för tydlighet. Resultatet skrivs ut med `summary`.

```
mod <- lm(Medeltemperatur ~ År, dat_medel)
summary(mod)
```

```
##
## Call:
## lm(formula = Medeltemperatur ~ År, data = dat_medel)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -2.1256 -0.5387  0.1005  0.5267  1.4678
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) -138.63357   23.31584  -5.946 6.75e-07 ***
## År             0.07391    0.01165   6.341 1.94e-07 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.8509 on 38 degrees of freedom
## Multiple R-squared:  0.5141, Adjusted R-squared:  0.5014
## F-statistic: 40.21 on 1 and 38 DF,  p-value: 1.939e-07
```

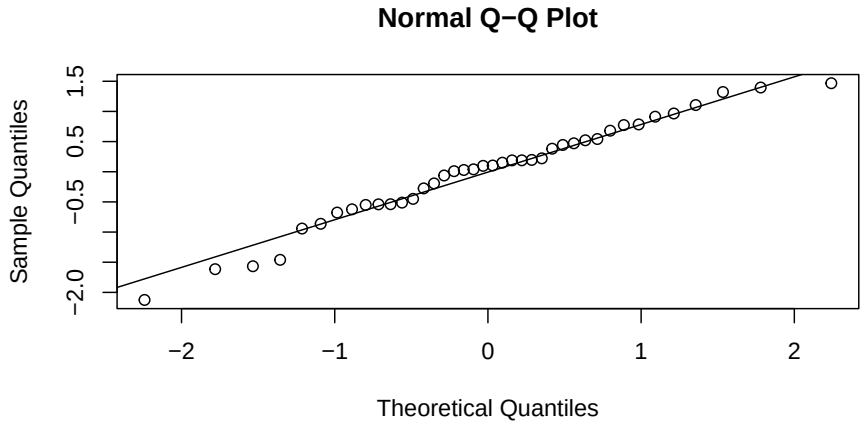
Linjen har ett intercept på -138.63 och en lutning på 0.074. Standardtolkningar

är att det var −138 grader år noll (hmmmm) och att medeltemperaturen ökar med 0.07 grader per år.

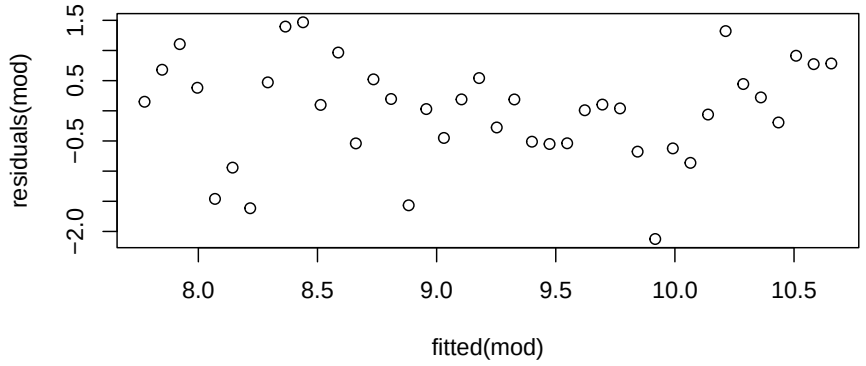
Testet för lutningen (ett t-test) ges på raden för `År`. Nollhypotesen är att lutningskoefficienten är 0 och det låga p-värdet tyder på att den nollhypotesen bör förkastas. Det finns en statistiskt säkerställd ökning i medeltemperatur över tid.

Diagnosgrafer ger inga uppenbara avvikelser från normalfördelning eller struktur i spridningen.

```
qqnorm(residuals(mod))
qqline(residuals(mod))
```

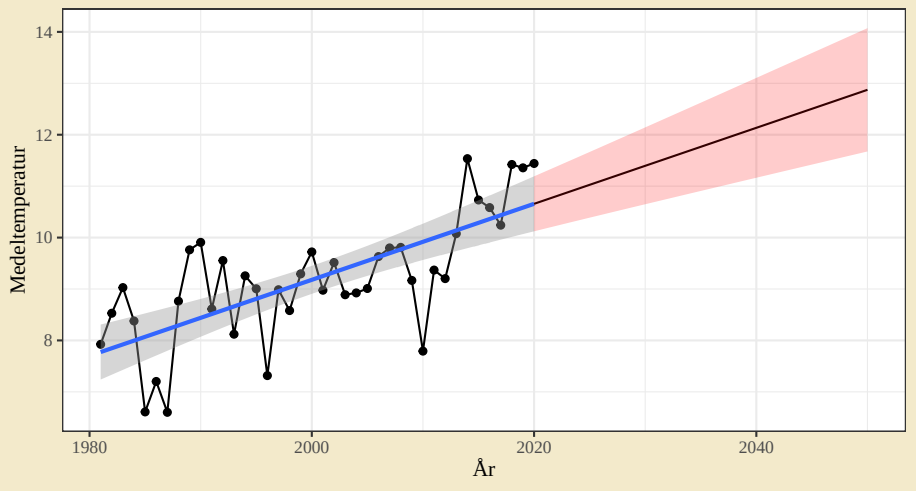



```
plot(fitted(mod), residuals(mod))
```



c. Det är möjligt att prognosticera värden med `predict`. Funktionen tar en skattad modell (här `mod`) och en ny datatabellen, som innehåller en kolumn med samma namn som den förklarande variabeln i modellen (här `År`). Funktionen kan även producera konfidensintervall för skattningen, här genom `interval = "conf"` för ett konfidensintervall.

```
dat_predict <- predict(mod, newdata = tibble(År = 2020:2050), interval = "conf") %>%
  as_tibble() %>% # Ändra till en tibble för enklare hantering
  mutate(År = 2020:2050) # Lägg till år för senare graf

g1 + # Samma grundgraf som tidigare
  geom_line(aes(År, fit), data = dat_predict) + # Lägger till en linje för skattningar för komma
  geom_ribbon(aes(År, ymin = lwr, ymax = upr),  # Lägger till ett band för konfidensintervallet
              inherit.aes = F, data = dat_predict,
              alpha = 0.2, fill = "red") # alpha sätter genomskinlighet för en yta
```



d. I en tidigare uppgift påpekas att data inte samlats in samma tider på dygnet under hela mätperioden - det är därmed möjligt att den observerade ökningen beror på förändringar i mätmetod. Valet av de fyrtio senaste åren var ett godtyckligt val som påverkar lutningen kraftigt (till exempel om man av en ren slump börjar med några kalla år och avslutar med några varma). Regressionsmodellen är en linjär modell, vilket ger den uppenbarligen orimliga tolkning av interceptet som en temperatur år 0.

Slutligen finns förstås det grundläggande problemet med prognoser: det kan ske oväntade saker som leder till strukturbrott och gör att sambandet mellan förklarande och förklarad variabel förändras.

Övning 10.3 (Metodjämförelse)**.** Någon vill jämföra två metoder för att mäta volym på någon planta. Hen mäter därför nio olika plantor med bägge metoderna, vilket ger följande värden.

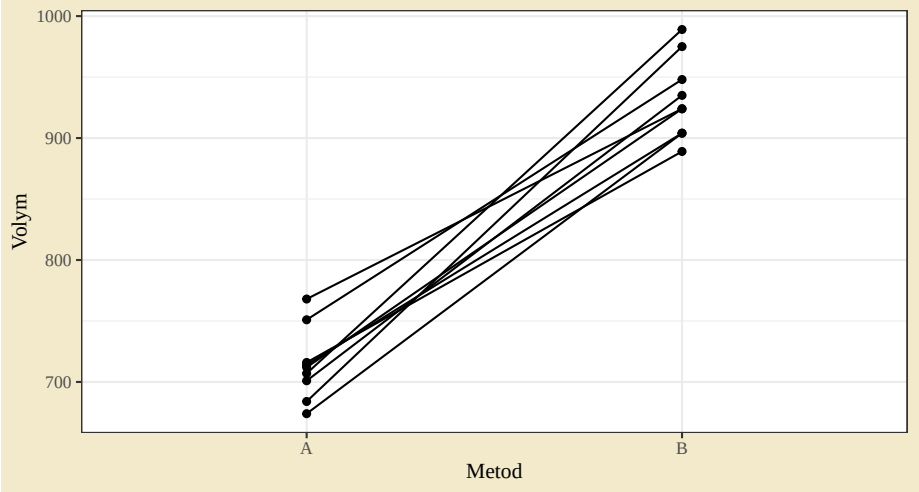
Metod	1	2	3	4	5	6	7	8	9
A	701	707	768	714	712	684	716	751	674
B	935	989	924	904	924	975	889	948	904

a. Illustrera datan med en lämplig figur. Testa med ett lämpligt test om metoderna ger samma medelvärdesvolym.

b. Beräkna korrelationen mellan metoderna. Metod A är en dyrare men bättre metod. Ger mätning med metod B en säker uppskattning av utfallet med metod A?

Lösningsförslag 10.3 (Metodjämförelse)**.** Data kan hämtas från excelfilen med uppgiftsdata. Ett diagram visar en tydlig skillnad mellan grupperna.

```
dat <- read_excel("Data/Uppgiftsdata.xlsx", sheet = "Metodjämförelse")

ggplot(dat, aes(Metod, Volym, group = Yta)) +
  geom_point() +
  geom_line()
```



a. Ibland behöver man inget test.

Men en jämförelse mellan två grupper kan förstås tas som ett parat t-test. Man kan först använda `pivot_wider` för att skriva om data till kolumnform.

```
dat_wide <- dat %>%
  pivot_wider(names_from = Metod, values_from = Volym)

t.test(dat_wide$A, dat_wide$B, paired = T)
```

```
##
```

```
##  Paired t-test
##
## data:  dat_wide$A and dat_wide$B
## t = -14.228, df = 8, p-value = 5.8e-07
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -253.7192 -182.9475
## sample estimates:
## mean of the differences
##               -218.3333
```

Parad data kan också ses som det enklaste exemplet på ett block, och analyseras som en anova med block. För att R ska tolka variabeln Yta som en faktor kan man ändra dess typ med `as.character`.

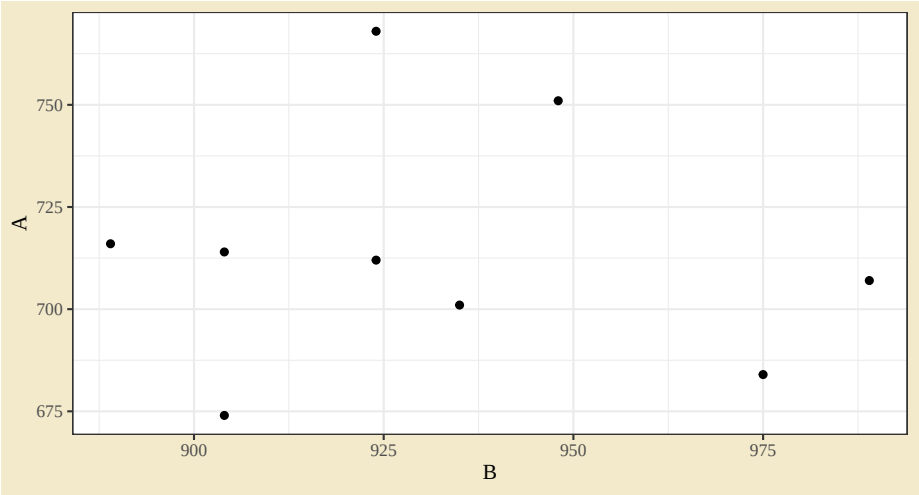
```
dat <- dat %>% mutate(Yta = as.character(Yta))

mod <- lm(Volym ~ Metod + Yta, dat)

Anova(mod)
```

```
## Anova Table (Type II tests)
##
## Response: Volym
##           Sum Sq Df  F value  Pr(>F)
## Metod     214512  1 202.4419 5.8e-07 ***
## Yta         7440  8   0.8777  0.5709
## Residuals   8477  8
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

P-värdet från ett F-test på metod är detsamma som det från det parade t-testet.

b.

```
ggplot(dat_wide, aes(B, A)) +
  geom_point()
```




```
cor.test(dat_wide$A, dat_wide$B)
```

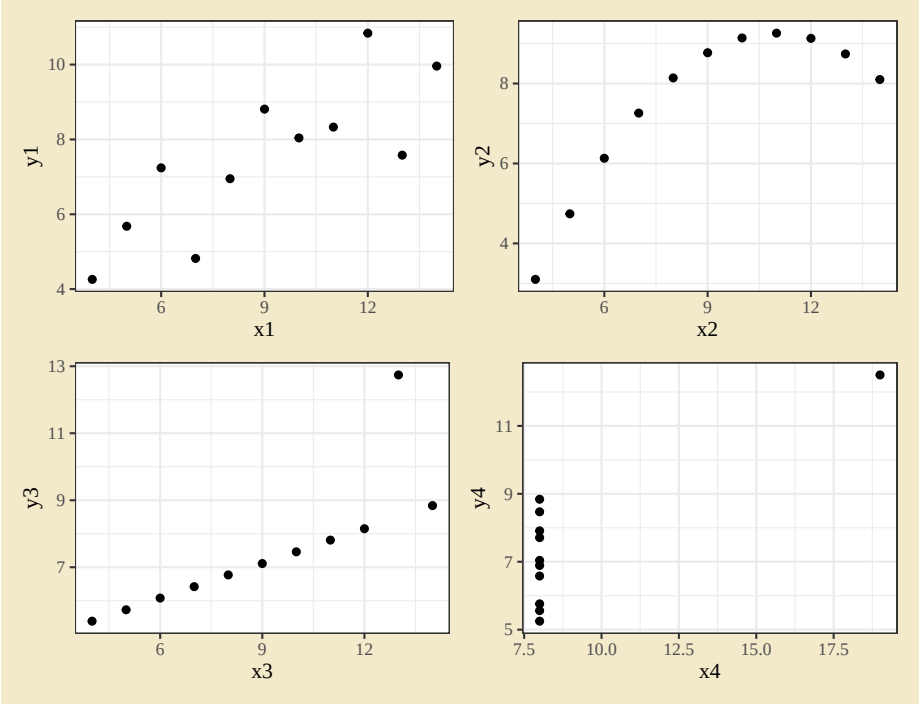
```
##
##  Pearson's product-moment correlation
##
## data:  dat_wide$A and dat_wide$B
## t = -0.17396, df = 7, p-value = 0.8668
## alternative hypothesis: true correlation is not equal to 0
## 95 percent confidence interval:
##  -0.6992626  0.6257789
## sample estimates:
##         cor
## -0.06560983
```

Det finns inget signifikant samband mellan metoderna. Mätning med metod B säger alltså inget om utfallet i metod A. Om metod A är den mer precisa metoden, tyder detta på att metod B inte bör användas.

Övning 10.4 (Anscombes data)**.** Den raka regressionslinjen eller det enkla korrelationsmåttet säger lite om hur data egentligen ser ut. En vanlig illustration av detta är *Anscombes kvartett*, fyra exempel konstruerade av den brittiske statistikern Francis Anscombe 1973. Datan finns tillgänglig i R som datasetet `anscombe`. Plotta de fyra graferna (`x1` paras med `y1` och så vidare) i spridningsdiagram och beräkna korrelation för varje par. Kommentera utfallet.

Lösningsförslag 10.4 (Anscombes data)**.** Data finns tillgänglig i R som `anscombe`, vilket är en tabell med åtta kolumner som består av fyra par (där `x1` är kopplad till `y1` och så vidare). Paren kan plottas med enkla spridningsdiagram.

```
g1 <- ggplot(anscombe, aes(x1, y1)) + geom_point()
g2 <- ggplot(anscombe, aes(x2, y2)) + geom_point()
g3 <- ggplot(anscombe, aes(x3, y3)) + geom_point()
g4 <- ggplot(anscombe, aes(x4, y4)) + geom_point()

library(patchwork)
(g1 + g2) / (g3 + g4)
```



Graferna ser olika ut.

```
cor(anscombe$x1, anscombe$y1)
```

```
## [1] 0.8164205
```

```
cor(anscombe$x2, anscombe$y2)
```

```
## [1] 0.8162365
```

```
cor(anscombe$x3, anscombe$y3)
```

```
## [1] 0.8162867
```

```
cor(anscombe$x4, anscombe$y4)
```

```
## [1] 0.8165214
```

Men har samma korrelation (till fjärde decimal).

En modern utveckling av Anscombes kvartett ges av *The Datasaurus Dozen*.

Övning 10.5 (Datasaurus Dozen)**.** Datasaurus-datan är en konstruerad datamängd som illustrerar hur skilda mönster i data kan ge samma punktskattningar (medelvärden, standardavvikelser och korrelationer). Datan finns tillgänglig som en del av TidyTuesday-projektet och kan hämtas med följande rad.

```
dat <- readr::read_csv('https://raw.githubusercontent.com/rfordatascience/tidytuesday/m
```

a. Datan innehåller en gruppering (`dataset`) och x- och y-koordinater. För varje grupp i kolumnen `dataset`, beräkna medelvärde och standardavvikelse för x och y, och beräkna korrelationen mellan x och y. Kommentera utfallet.

b. Illustrera materialet med en lämplig graf, t.ex. ett spridningsdiagram mellan x och y för varje grupp i kolumnen `dataset`.

Lösningsförslag 10.5 (Datasaurus Dozen)**.** a. Punkskattningar kan beräknas genom att gruppera enligt `dataset` och summera med lämpliga funktioner (`mean`, `sd` och `cor`).

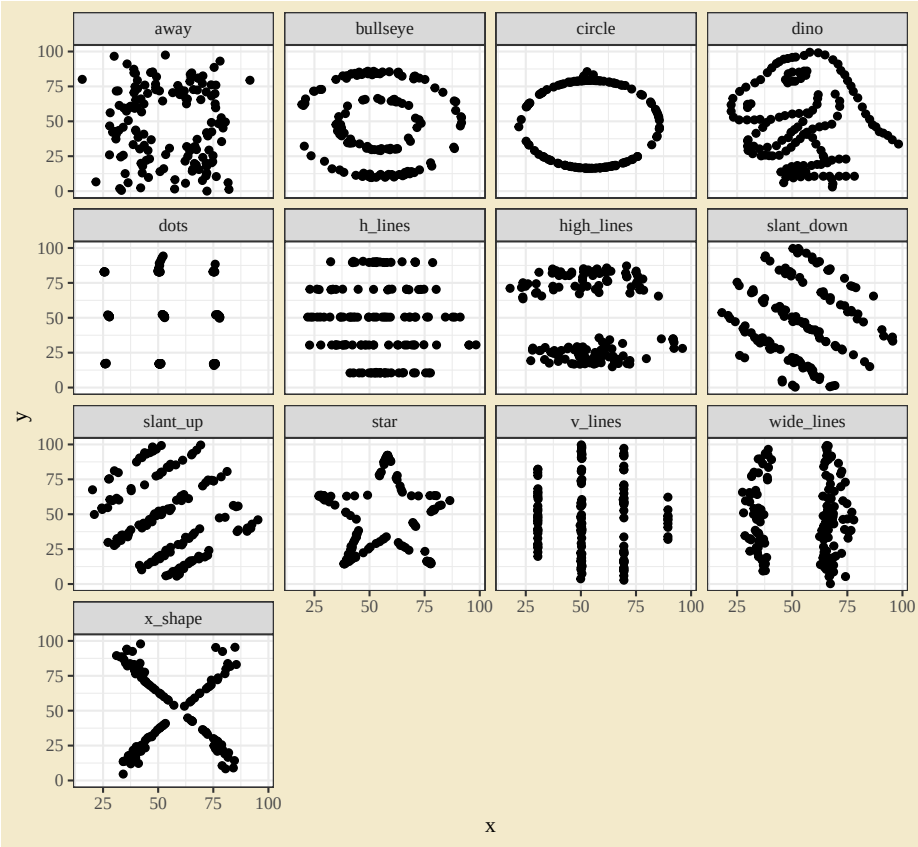
```
dat %>%
  group_by(dataset) %>%
  summarise(mean(x), mean(y), sd(x), sd(y), cor(x,y))
```

```
## # A tibble: 13 x 6
##    dataset    `mean(x)` `mean(y)` `sd(x)` `sd(y)` `cor(x, y)`
##    <chr>          <dbl>     <dbl>   <dbl>   <dbl>       <dbl>
##  1 away            54.3      47.8    16.8    26.9     -0.0641
##  2 bullseye        54.3      47.8    16.8    26.9     -0.0686
##  3 circle          54.3      47.8    16.8    26.9     -0.0683
##  4 dino            54.3      47.8    16.8    26.9     -0.0645
##  5 dots            54.3      47.8    16.8    26.9     -0.0603
##  6 h_lines         54.3      47.8    16.8    26.9     -0.0617
##  7 high_lines      54.3      47.8    16.8    26.9     -0.0685
##  8 slant_down      54.3      47.8    16.8    26.9     -0.0690
##  9 slant_up        54.3      47.8    16.8    26.9     -0.0686
## 10 star            54.3      47.8    16.8    26.9     -0.0630
## 11 v_lines         54.3      47.8    16.8    26.9     -0.0694
## 12 wide_lines      54.3      47.8    16.8    26.9     -0.0666
## 13 x_shape         54.3      47.8    16.8    26.9     -0.0656
```

Grupperna har samma medelvärde och standardavvikelser i x och y. Korrelationerna mellan x och y är mycket lika.

b. Datan kan illustreras med ett spridningsdiagram. Funktionen `facet_wrap` kan användas för separata fönster för skilda grupper.

```
ggplot(dat, aes(x, y)) +
  geom_point() +
  facet_wrap(~ dataset)
```



Figuren visar att det finns tydliga mönster i flera av grupperna och tydliga skillnader mellan dem.