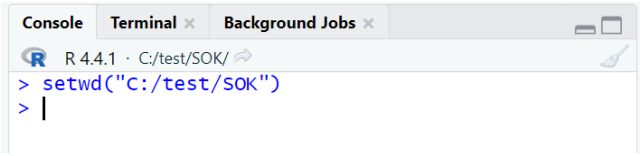
Datorlaboration 2, Statistisk översiktskurs (SÖK)

I datorlabb 2 arbetar vi med deskriptiv statistik, att sammanfatta data i tabeller och grafer, samt beräkningar av sammanfattande mått som medel, median och standardavvikelse. Föreläsning 2, 3 (första delen) och kursboken kap 4-6 diskuterar också dessa ämnen.

På näst sista sidan i labben kan du se hur du får fram olika tecken på ditt tangentbord (exv. tecknen **~** och **|**).

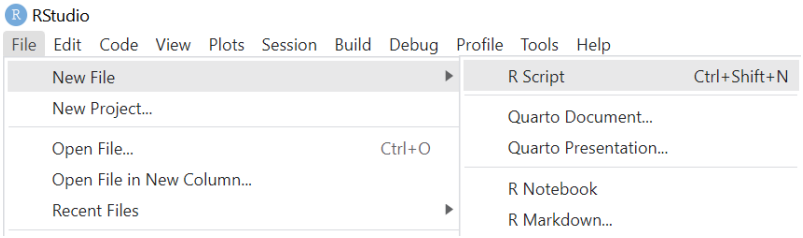
Innan själva uppgifterna börjar, gör följande (arbetskatalog, fil att spara i, ladda paket)

1. I Console i R, kör ditt **setwd**-kommando från labb 1 som sätter din arbetskatalog i R tlll din **SOK-mapp** på datorns hårddisk. Kommandot ser olika ut beroende på var på datorn din SOK-mapp finns.



Du kan alternativt göra detta steg genom menyerna i R, som vi gjorde i kapitel 6 i laboration 1, dvs: Välj menyn **Session** i R, sedan **Set Working Directory**, sedan **Choose Directory…** och klicka dig fram till mappen **SOK**.

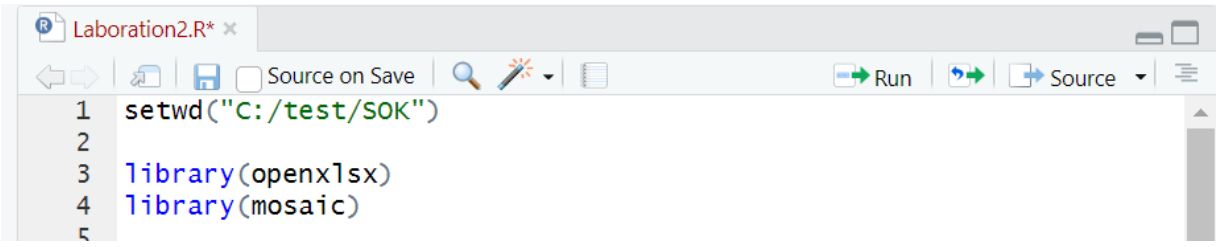
2. Skapa en textfil där du kommer spara dina labb2-kommandon (ett ”R-script”) genom att i menyn i R välja **File** och sedan **New File** och sedan **R script**.



3. Labb 2 förutsätter att följande paket finns installerade: **openxlsx** och **mosaic**

Paket installeras via kommandot install.packages('packagename'), där packagename är paketnamnet. Om du inte redan gjort installationen i labb 1, kör kommandona install.packages('openxlsx') och install.packages('mosaic') i Console i R.

Skriv sedan in följande library-kommandon i din textfil:



Kör sen de två library-kommandona, ställ dig på första raden med ett library-kommado och tryck på Run, två gånger.

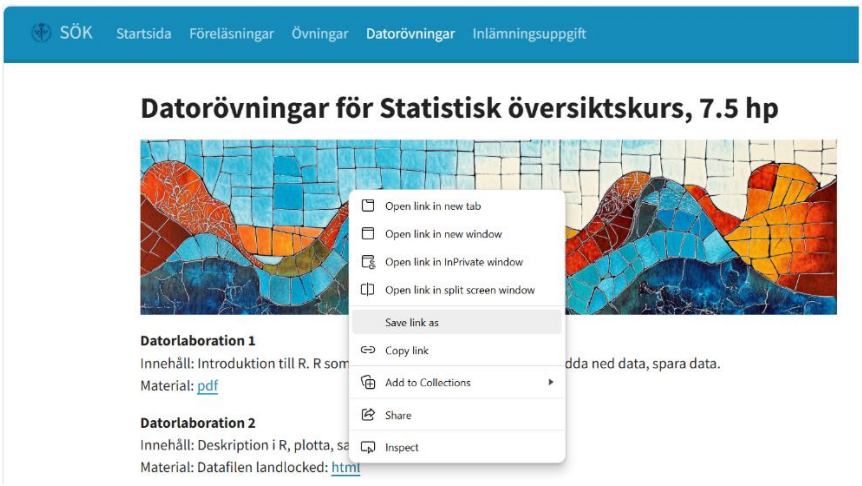
4. På samma sätt som i labb 1 har vi som första kommando i textfilen skrivit in kommandot som bestämmer arbetskatalogen i R. Du kan kopiera kommandot du körde i steg 1 ovan, in i din textfil (kopiera från Console eller från History-fliken i övre högra delen av RStudio.)

5. Spara din textfil genom att trycka på sparasymbolen  (se bilden steg 3). Välj lämpligt namn. **Spara din fil ofta.**

1. Läs in datasetet landlocked

Vi kommer först att läsa in och arbeta med ett dataset med tre variabler och knappt 200 observationer. Vi har data på länder och världsdelar, dessa data (och definitionerna på vad som är med i databasen) kommer dels från Utrikespolitiska institutet (https://www.ui.se/landguiden). Vi har också data från Encyclopedia Britannica på om ett land är ”landlocked”, dvs saknar kust, vid intresse, läs mer här: https://www.britannica.com/topic/landlocked-country

Filen heter **landlocked.txt** och kan laddas ner från kursens Githubsida, under Dataövningar. **Högerklicka** på filnamnet och välj sedan ”Save link as” (exakt vad det står kan bero lite på vilken dator du har).



Om det inte fungerar finns filen också att hämta på Athena, i katalogen Datafiler.

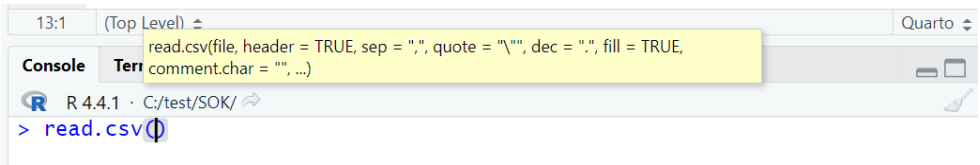
Spara filen **landlocked.txt** i din **SOK-mapp** på datorn.

Om man öppnar filen direkt på datorn ser de första raderna ut som följer:

```
Varldsdel,Land,Landlocked
Asien,Afghanistan,1
Europa,Albanien,0
Afrika,Algeriet,0
Europa,Andorra,1
Afrika,Angola,0
Nordamerika,Antigua & Barbuda,0
```

Första raden utgörs av tre variabelnamn. Såväl variabelnamnen på första raden som data på raderna som följer är åtskilda med kommatecken. Filen kan därför läsas in som en **CSV**-fil, en **Comma Separated Values**-fil.

Vi kan använda kommandot **read.csv**(file="landlocked.txt") för att läsa in filen i R. Om du bara skriver detta kommando, utan fler argument efter filnamnet, kommer kommandot att anta att första raden i data är variabelnamn och att filen är en CSV-fil (båda är rätt). Den gula hjälprutan i nästa bild illustrerar:



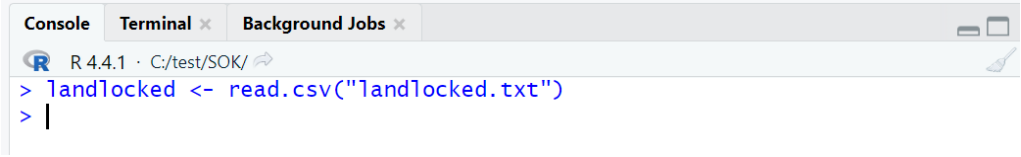
Du måste skriva in filnamn, inklusive ändelsen (.txt), och inklusive apostrofer eller citattecken runt namnet. Om argumentet header inte skrivs in antar R att header=TRUE, dvs. att första raden består av variabelnamn. Om sep (separator) inte skrivs in antar R sep=”,” – dvs. att variabler separeras av kommatecken.

Du kan testa något av dessa kommandon, alla ska ge samma resultat:

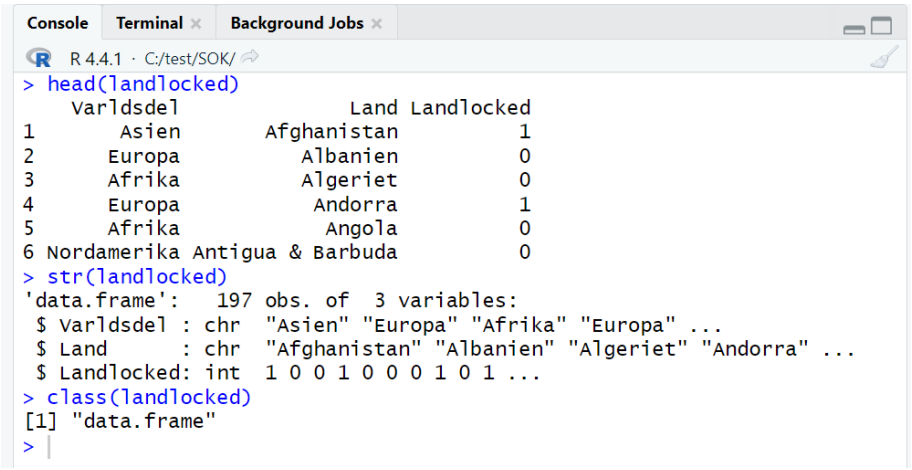
read.csv(file="landlocked.txt")

read.csv(file="landlocked.txt", header=TRUE)
read.csv(file="landlocked.txt", header=T)
read.csv(file="landlocked.txt", sep= ",")
read.csv(file="landlocked.txt", header=TRUE, sep= ",")

När vi kör ovanstående läser vi in data direkt i Console, men vi vill lagra data i en **data frame**, skriv därför:



Sen kan du köra följande tre kommandon, som visar de sex första raderna (**head**), datastrukturen (**str**) samt vilken klass data har (**class**).



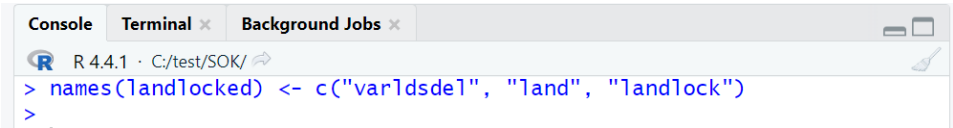
Vi ser hur data ser ut och att vi har **tidy** (välstrukturerade) data (en variabel – en kolumn, en observation – en rad), sen ser vi att två variabler är av typen **chr** (character – textsträng) och att variabeln Landlocked är av typen **int** (integer – heltal).

Varldsdel och Landlocked är båda **kategoriska variabler**. Landlocked=1 om landet saknar kust, annars =0. Längst till vänster när vi skriver ut data lägger R till ett radnummer, eller ett radindex, Afghanistan har nummer 1.

---

*Den kategoriska variabeln Landlocked har bara två nivåer och kan också kallas **binär, dikotom, indikator** eller **dummy**-variabel. Dessa begrepp dyker ofta upp när vi jobbar med data.*

Svenska bokstäver har undvikits i variabelnamnen. Variabeln Landlocked heter nästan samma som vår dataframe (landlocked), vi kan döpa om variabeln så att skillnaden blir större. Följande kommando, **names()**, döper (om) alla tre variablerna, och vi väljer att ha variabelnamnen med små bokstäver. Vi skapar alltså en vektor med tre namn (kommandot **c**(…)) och tilldelar (tecknet **<-**) dessa namn till våra variabler.



Testa sen att köra något av kommandona head(landlocked) eller names(landlocked) för att se att namnen ändrats.

Uppgift 1.1

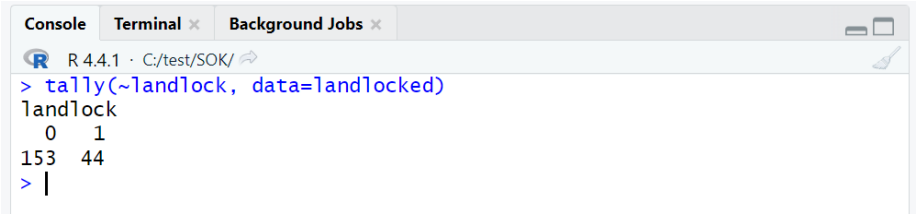
Gör de olika stegen ovan, om du inte redan gjort det.

2. Tabeller med en kategorisk variabel

I instruktionen nedan skrivs koden oftast direkt i Console. Men vi rekommenderar att du skriver din egen kod i din textfil, så att koden sparas. I koden kan du kan använda **#-tecknet** (hashtag, ”brädgård”) för att kommentera din kod. När R-koden körs kommer R automatiskt hoppa över raderna som börjar med #. Filen kan exempelvis se ut så här:

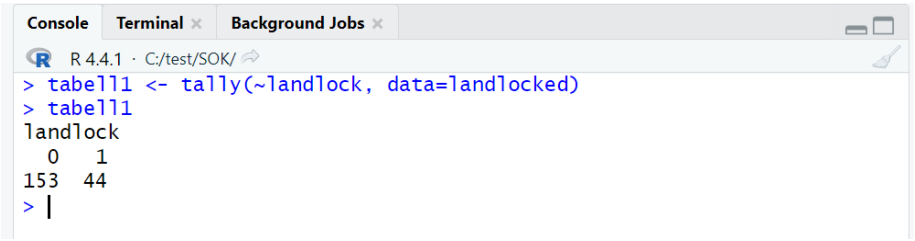
Du kan också kommentera kod på samma rad som själva koden, gör några mellanrum efter ditt kommando, sen hashtagsymbolen, sen kommentaren.

Vi tittar först på hur vi kan göra en enkel tabell av, och fördelning över, antalet länder som inte har respektive har kust (**landlock**-variabeln). Vi använder kommandot **tally** från paketet **mosaic** (se till att ha paketet installerat och laddat).

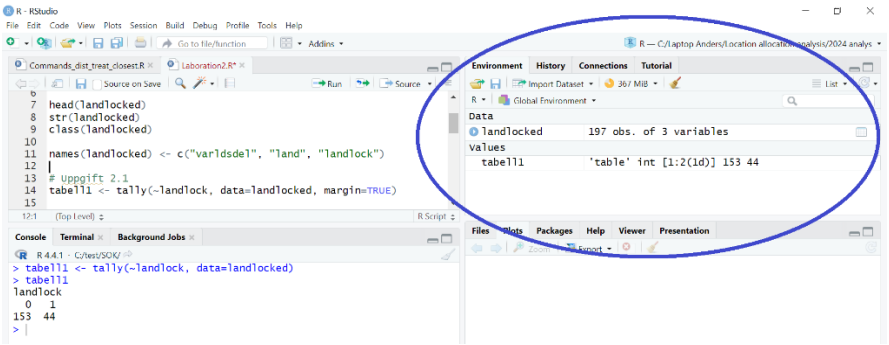


Var noga med att få med **tildetecknet** (~) före variabelnamnet (se appendix för hur du får fram tecknet). **data**-argumentet anger var **variabeln landlock** hämtas ifrån.

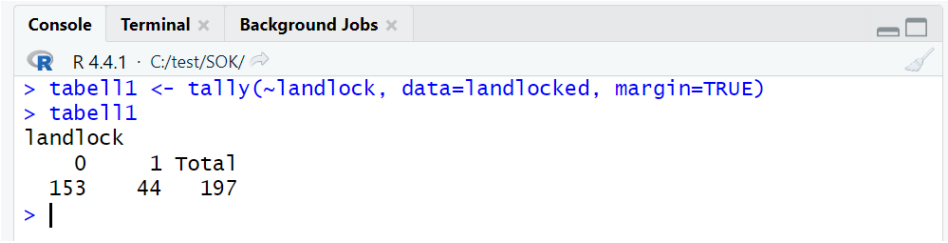
Istället för att enbart skriva ut tabellen, skapa ett **objekt** som består av själva tabellen, kalla den exv. tabell1:



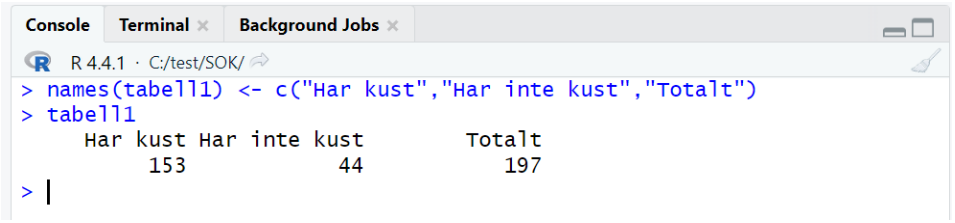
Här har vi skapat objektet tabell1, och också skrivit ut tabellen i Console. Bekräfta i fliken Environment uppe till höger i RStudio att tabell1 finns med i listan över **objekt**:



Tabellen vi gjort är en enkel **frekvenstabell**. Om vi vill ha med totala antalet länder (totala antalet observationer) kan vi lägga till argumentet **margin**:



Ge sen nya namn till tabellkolumnerna, exv. ”Har kust”, ”Har inte kust”, ”Totalt”. Skriv ut tabellen igen.



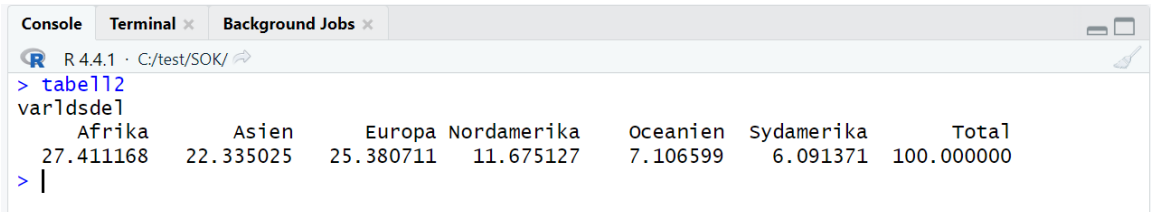
Uppgift 2.1

Kör all kod ovan, om du inte redan gjort det.

Med det extra argumentet **format** =”percent” eller **format**=”proportion”, i tally(), kan man ta fram (procent-)andelar istället för antal.

Uppgift 2.2

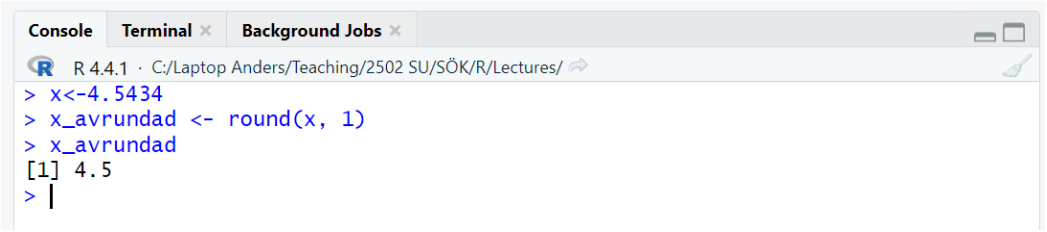
Gör en tabell över **fördelningen av andelen länder per världsdel** (i andelar eller i procentandelar). Kalla tabellen exv. tabell2. Slutresultatet (om du väljer procentandelar) ska se ut så här:



Du ska nu också ha ett nytt objekt (tabell2) under fliken **Environment** uppe till höger i RStudio.

Det kanske inte är så snyggt och lite onödigt med många decimaler, åtminstone om tabellen ovan är vår sluttabell som vi ska presentera. Med kommandot **round** kan du runda av talen, exempelvis till en eller två decimaler.

Säg att variabeln x=4.5434, då avrundar nedanstående kod x till en decimal (4.5), sen lägger koden in det avrundade värdet i en ny variabel x_avrundad, och skriver till sist ut x_avrundad:



Istället för att avrunda enskilda variabler kan du ha andra objekt som argument. Kommandot **round(**min_tabell, 2**)** avrundar exempelvis talen i objektet min_tabell till två decimaler.

Avrunda bara tal som du inte ska använda vidare i beräkningar. Mao, om du ska göra beräkningar med x ovan, använd x och inte x_avrundad.

Uppgift 2.3

Skapa och skriv ut en ny tabell (tabell2_avrundad), där du har samma information som i uppgift 2.2 men avrundat till två decimaler, resultatet ska se ut så här (om du valt procentandelar):

```
varldsdel
     Afrika       Asien      Europa Nordamerika    Oceanien  Sydamerika       Total
      27.41       22.34       25.38       11.68        7.11        6.09      100.00
```

Uppgift 2.4

Hur stor (procent-)andel av alla länder ligger i Europa?

---

*Ta bort objekt från fliken **Environment** med **rm()** - men spara alltid din kod!*

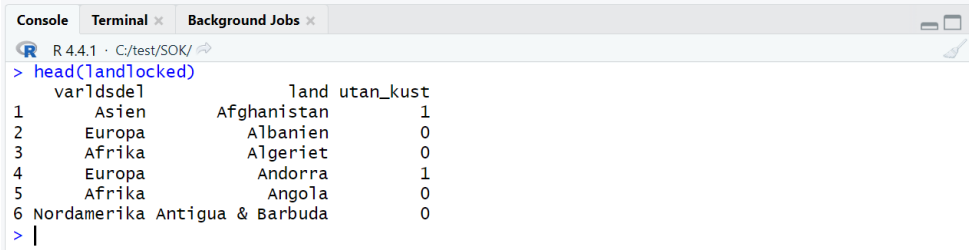
*Du kan vilja ta bort objekt från RStudio. Ovan skapade du kanske variabeln x. Du kan ta bort x med kommandot **rm(x)** (som betyder remove x). Andra objekt tas bort på samma sätt. Det du framförallt ska spara är din textfil med kod. **Spara ofta.***

Du kan också rensa Console med ”sopkvasten” (eller tryck ctrl + l) om du tycker det ser rörigt ut. Inga data försvinner och kommandohistoriken finns kvar (tryck uppåtpil när du står i Console eller gå till fliken History uppe till höger i RStudio.

Innan vi går vidare byter vi namn på variabeln **landlock** (igen), vi kallar den istället för ”**utankust**”. Vii slipper då blanda namn på engelska och svenska.

Uppgift 2.5

Byt namn på variabeln landlock, du kan köra samma kommando som vi gjorde för namnbyte ovan, med det tredje namnet ändrat. Efter namnbytet ska de första raderna av din data frame se ut så här:



3. Tabeller med två kategoriska variabler, olika typer av fördelningar

Vi ska nu göra två andra tabelltyper, med samma data. Vi såg tabellerna i föreläsning 2.

Kategoriska variabler är variabler vars utfall är kategorier. I uppgift 2 ovan gjorde vi frekvenstabeller med en variabel. En fördelning av en enda variabel, till exempel region ovan, kallas för **marginalfördelningen**. När vi betraktar två kategoriska variabler, till exempel y och z, så finns det olika fördelningar vi kan tänka oss:

- **Marginella fördelningar**, en fördelning för vardera variabel y och z.
- **Simultan fördelning** för variablerna y och z (endast en fördelning).
- **Betingade fördelningar**, en fördelning för vardera variabel y och z, betingat på den andra.

Notera att dessa tre typer av fördelningar finns oavsett om variablerna är numeriska eller kategoriska. När bägge variablerna är kategoriska kan vi studera fördelningarna i en så kallad **korstabell**. Att känna till de olika fördelningstyperna är mycket viktigt.

För att illustrera koncepten i ett praktiskt problem ska vi undersöka om olika världsdelar har olika andel länder utan kust.

Vi börjar med att skapa en korstabell. En korstabell är en tabell som innehåller frekvensen för varje utfall av variablerna. Notera att om vi har två kategoriska variabler, säg y och z, så är utfallen alla parvisa kombinationer av utfallen av de respektive variablerna.

```
Console  Terminal  Background Jobs
R 4.4.1 · C:/test/SOK/
> tally(~utan_kust + varldsdel, data=landlocked)
         varldsdel
utan_kust Afrika Asien Europa Nordamerika Oceanien Sydamerika
        0     38    34     34          23       14          10
        1     16    10     16           0        0           2
>
```

Med två värden på utan_kust och sex världsdelar totalt får vi två gånger sex = 12 olika värden (**utfall**) i tabellen.

När vi gör tally-kommandot med två kategoriska variabler lägger vi alltså till en variabel till kommandona i kapitel 2 (med plustecknet), och vi får våra data uppdelade på båda variablerna (en simultan fördelning). Vi ser exempelvis att Sydamerika har två länder utan kust.

Uppgift 3.1

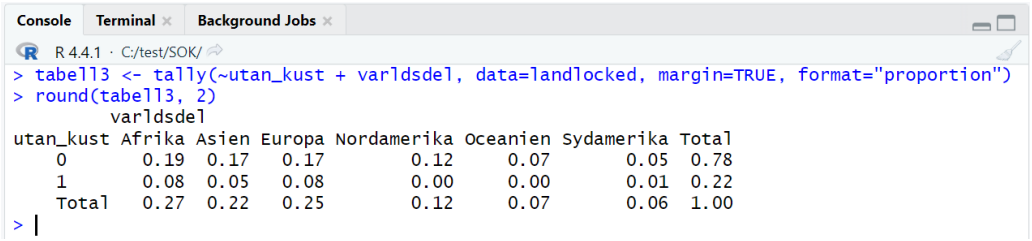
Testa att göra ovanstående tally-kommando men med ”&” istället för ”+” och testa sen också att ändra ordningen på de två variablerna. Blir det någon skillnad?

Det finns ofta flera sätt att skriva R-kommandon men vi kommer inte gå in på olika alternativ i den här kursen. Fråga lärarna om du vill läsa vidare om andra sätt att skriva R-kod.

Utveckla nu tabellen ovan så att vi får summan av varje rad och av varje kolumn (argumentet margin), och ändra formatet till ”proportion”:

```
Console  Terminal  Background Jobs
R 4.4.1 · C:/test/SOK/
> tally(~utan_kust + varldsdel, data=landlocked, margin=TRUE, format="proportion")
         varldsdel
utan_kust     Afrika      Asien     Europa Nordamerika   Oceanien Sydamerika      Total
    0     0.19289340 0.17258883 0.17258883  0.11675127 0.07106599 0.05076142 0.77664975
    1     0.08121827 0.05076142 0.08121827  0.00000000 0.00000000 0.01015228 0.22335025
    Total 0.27411168 0.22335025 0.25380711  0.11675127 0.07106599 0.06091371 1.00000000
>
```

Spara denna tabell som tabell 3 och skriv sen ut tabell 3 med två decimaler:



Uppgift 3.2

Gå igenom följande frågor

- Hur stor andel av alla länder har ingen kust?
- Hur stor andel av alla länder ligger i Europa?

Båda frågorna handlar om **marginella fördelningar** och läses av i högerkolumnen respektive i sista raden. För den första frågan tar vi inte hänsyn till variabeln världsdel och för den andra frågan tar vi inte hänsyn till variabeln utan_kust. Frågorna liknar analysen i kapitel 2 ovan, där vi inte alls hade någon uppdelning på en andra variabel.

Uppgift 3.3

Gå igenom följande frågor

- Hur stor andel av alla länder har inte kust **och** ligger i Europa?
- Hur stor andel av alla länder har kust **och** ligger i Sydamerika?

Båda dessa frågor handlar om den **simultana fördelningen** och läses av i en specifik ”cell” i tabellen. Summan av alla celler summerar till 1 (100%).

Det finns också **betingad fördelning**. Jämför följande två frågor.

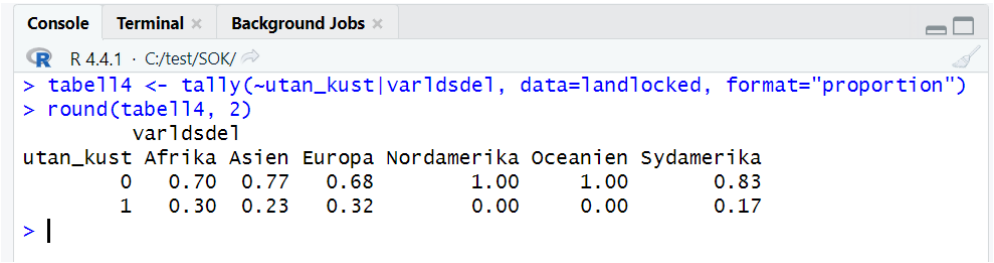
- Hur stor andel av alla länder ligger i Afrika **och** har inte kust?
- Hur stor andel av alla länder i Afrika har inte kust?

I den andra frågan säger vi att vi **betingar på** världsdel. Mer generellt vill vi veta fördelningen av en variabel y, givet värdet på en annan variabel z.

Vi tar fram en **korstabell med betingad fördelning** genom att betinga den ena variabeln på den andra variabeln, att betinga y på z liknar (lite informellt) att variabeln y är en funktion av z.

Istället för att skriva kommandot tally(~y+z) (”dela upp data på varje kombination av y och z”) skriver vi tally(~y|z) (”visa fördelningen av y för varje värde av z”). (vid behov, se appendix för hur du får fram tecknet | på din dator).

Om vi vill betinga variabeln utan_kust på världsdel kör vi alltså kommandot:



Här har vi också skapat ett tabellobjekt (tabell4), samt skrivit ut tabellen med två decimaler. Vi ser att, för varje världsdel, summerar andelarna till 1 (sista raden).

Uppgift 3.4

Gå igenom följande frågor

- Hur stor andel av alla länder i Afrika har inte kust?
- Vilken eller vilka världsdelar har högst respektive lägst andel länder utan kust?

Uppgift 3.5

Gör en betingad korstabell i R där du kan läsa av svaret på följande fråga (vilken variabel ska du betinga på för att få fram rätt tabell?):

- Hur stor andel av alla länder utan kust ligger i Europa?
- Vilken eller vilka världsdelar har högst andel av länderna utan kust?

Ibland finns flera olika sätt att skriva kommandon i R för att uppnå samma sak, exempelvis en korstabell med betingad fördelning, men vi går inte igenom fler än ett sätt i den här kursen.

---

4. Läs in datasetet titanic

Vi kommer nu att läsa in och arbeta med datasetet titanic, som vi också sett på föreläsningarna. Börja med att ladda ned Excelbladet Chapter_3.xlsx från följande hemsida (som ibland är lite långsam):
https://media.pearsoncmg.com/intl/ge/2021/cws/ge_deveaux_stats_5/statdm5d_datasets.html

Du kan också ladda ned filen från Athena (ligger i katalogen Datafiler).

Lägg Excelbladet i din **SOK-mapp** på datorn. Datasetet titanic är ett av bladen (**sheet**) i Excelfilen.

I labb 1 använde vi **kommandot** read.xlsx från **paketet** openxlsx. Vi kör samma kommando igen, nu med en annan fil och ett annat blad som argument (var noggrann med små och stora bokstäver, etc., enligt nedan):

```
Console   Terminal ×   Background Jobs ×
R 4.4.1 · C:/test/SOK/
> titanic = read.xlsx("Chapter_3.xlsx", sheet = "Titanic")
> head(titanic)
                        Name Survived     Boarded Class   MWC Age Adut_or_Chld    Sex  Paid
1           ABBING, Mr Anthony     Dead Southampton     3   Man  42        Adult   Male  7.55
2        ABBOTT, Mr Ernest Owen     Dead Southampton  Crew   Man  21        Adult   Male    NA
3      ABBOTT, Mr Eugene Joseph     Dead Southampton     3 Child  14        Child   Male 20.25
4    ABBOTT, Mr Rossmore Edward     Dead Southampton     3   Man  16        Adult   Male 20.25
5 ABBOTT, Mrs Rhoda Mary 'Rosa'    Alive Southampton     3 Woman  39        Adult Female 20.25
6    ABELSETH, Miss Karen Marie    Alive Southampton     3 Woman  16        Adult Female  7.65
  Ticket_No Boat_or_Body                  Job          Class_Dept Class_Full
1      5547         <NA>           Blacksmith 3rd Class Passenger          3
2      <NA>         <NA> Lounge Pantry Steward    Victualling Crew          V
3    CA2673         <NA>              Scholar 3rd Class Passenger          3
4    CA2673        [190]             Jeweller 3rd Class Passenger          3
5    CA2673            A                 <NA> 3rd Class Passenger          3
6    348125           16                 <NA> 3rd Class Passenger          3
> |
```

Här har vi både läst in datasetet och skrivit ut de sex första raderna.

Uppe i högra hörnet i RStudio, under fliken Environment, kan du klicka på den blå pilen framför titanic och titta på vilka variabler titanic innehåller (alternativt skriva **str**(titanic) i Console):

```
Data
titanic          2208 obs. of 14 variables
    $ Name         : chr  "ABBING, Mr Anthony" "ABBOTT, …
    $ Survived     : chr  "Dead" "Dead" "Dead" "Dead" ...
    $ Boarded      : chr  "Southampton" "Southampton" "S…
    $ Class        : chr  "3" "Crew" "3" "3" ...
    $ MWC          : chr  "Man" "Man" "Child" "Man" ...
    $ Age          : num  42 21 14 16 39 16 25 30 28 20 …
    $ Adut_or_Chld: chr  "Adult" "Adult" "Child" "Adult…
    $ Sex          : chr  "Male" "Male" "Male" "Male" ...
    $ Paid         : num  7.55 NA 20.25 20.25 20.25 ...
    $ Ticket_No    : chr  "5547" NA "CA2673" "CA2673" ...
    $ Boat_or_Body: chr  NA NA NA "[190]" ...
    $ Job          : chr  "Blacksmith" "Lounge Pantry St…
    $ Class_Dept   : chr  "3rd Class Passenger" "Victual…
    $ Class_Full   : chr  "3" "V" "3" "3" ...
```

Vi börjar med att ta bort några variabler som vi inte är intresserade av. För att ta bort från och med kolumn 10 (variabeln Ticket_No) kan du köra följande kommando, där minustecknet framför kolumnnumren innebär att dessa kolumner tas bort (kapitel 10 i labb 1 tog upp **indexering**). Vi kallar vår nya data fram titanic_small.

```
Console  Terminal  Background Jobs
R 4.4.1 · C:/test/SOK/
> titanic_small <- titanic[,-c(10,11,12,13,14)]
>
```

Vi tittar sen på titanic_small i Environment:

```
R  Global Environment
Data
titanic          2208 obs. of 14 variables
titanic_small    2208 obs. of 9 variables
    $ Name         : chr  "ABBING, Mr Anthony" "ABBOTT, …
    $ Survived     : chr  "Dead" "Dead" "Dead" "Dead" ...
    $ Boarded      : chr  "Southampton" "Southampton" "S…
    $ Class        : chr  "3" "Crew" "3" "3" ...
    $ MWC          : chr  "Man" "Man" "Child" "Man" ...
    $ Age          : num  42 21 14 16 39 16 25 30 28 20 …
    $ Adut_or_Chld: chr  "Adult" "Adult" "Child" "Adult…
    $ Sex          : chr  "Male" "Male" "Male" "Male" ...
    $ Paid         : num  7.55 NA 20.25 20.25 20.25 ...
```

Vi kommer inte heller använda variablerna ”Boarded” (kolumn 3) och ”MWC” (kolumn 5) och tar bort även dessa kolumner, och uppdaterar sedan titanic_small igen:

```
Console  Terminal  Background Jobs
R 4.4.1 · C:/test/SOK/
> titanic_small <- titanic_small[,-c(3,5)]
> head(titanic_small)
                        Name Survived Class Age Adut_or_Chld    Sex  Paid
1           ABBING, Mr Anthony     Dead     3  42        Adult   Male  7.55
2         ABBOTT, Mr Ernest Owen     Dead  Crew  21        Adult   Male    NA
3       ABBOTT, Mr Eugene Joseph     Dead     3  14        Child   Male 20.25
4     ABBOTT, Mr Rossmore Edward     Dead     3  16        Adult   Male 20.25
5 ABBOTT, Mrs Rhoda Mary 'Rosa'    Alive     3  39        Adult Female 20.25
6     ABELSETH, Miss Karen Marie    Alive     3  16        Adult Female  7.65
>
```

Vi har nu sju kolumner kvar. Genom kommandot **str**(titanic_small) kan vi se vilka variabler och variabeltyper vi har:

```
> str(titanic_small)
'data.frame':   2208 obs. of  7 variables:
 $ Name        : chr  "ABBING, Mr Anthony" "ABBOTT, Mr Ernest Owen" "ABBOTT, Mr Eugene Joseph"
"ABBOTT, Mr Rossmore Edward" ...
 $ Survived    : chr  "Dead" "Dead" "Dead" "Dead" ...
 $ Class       : chr  "3" "Crew" "3" "3" ...
 $ Age         : num  42 21 14 16 39 16 25 30 28 20 ...
 $ Adut_or_Chld: chr  "Adult" "Adult" "Child" "Adult" ...
 $ Sex         : chr  "Male" "Male" "Male" "Male" ...
 $ Paid        : num  7.55 NA 20.25 20.25 20.25 ...
>
```

Vi har 2208 observationer (antalet passagerare och besättning på Titanic), fem variabler av typen character (**chr**) och två numeriska variabler (**num**).

Följande två frågor är repetition av kapitel 2 ovan (och föreläsning 2).

Uppgift 4.1

Gör en frekvenstabell som visar antalet individer som överlevde respektive dog, samt totalantalet.

Uppgift 4.2

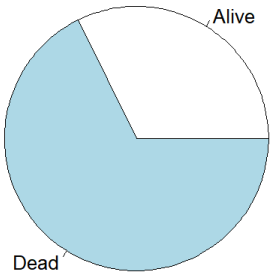
Gör en frekvenstabell som visar andelen resenärer i respektive klass.

5. Grafisk illustration av en kategorisk variabel

För att presentera kategoriska variabler grafiskt används oftast cirkeldiagram (pie chart) eller stapeldiagram (bar chart / bar plot). I R kan man använda sig av funktionerna **pie**() och **bargraph**() för respektive diagram.

För ett **cirkeldiagram** är det smidigt att använda sig av en sparad tabell som argument i funktionen:

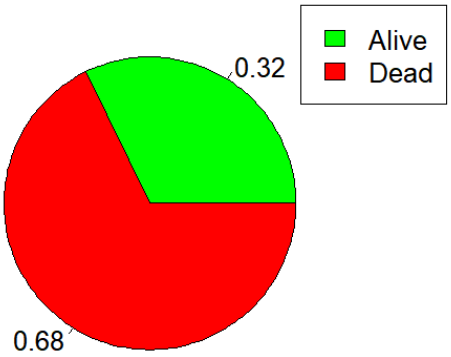
```
> rel_frekvenser <- tally(~ Survived, data = titanic_small, format = "proportion")
> pie(rel_frekvenser)
>
```



Ovan ser vi att vi först har sparat de relativa frekvenserna bland de som överlevde och de som dog, i tabellen **rel_frekvenser**, som sen används i funktionen **pie**(). Vi hade kunnat göra på andra sätt, exempelvis att istället skapa en vektor med de två frekvenserna (andel överlevande, andel döda).

Grafen ovan är inte så informativ. Det finns generellt många argument man kan använda sig av i R för att göra sina grafer så informativa som möjligt. Koden nedan är ett exempel.

```
Console  Terminal  Background Jobs
R 4.4.1 · C:/test/SOK/
> rel_frekvenser <- tally(~ Survived, data = titanic_small, format = "proportion")
> rel_frekvenser_avrundade <- round(rel_frekvenser,2)
>
> kategorinamn <- names(rel_frekvenser)
> kategorinamn
[1] "Överlevde"      "Överlevde inte"
> colors = c("green", "red")
>
> pie(rel_frekvenser, labels=rel_frekvenser_avrundade, col=colors)
> legend("topright", legend = kategorinamn, fill = colors)
>
```



round() används för att avrunda till önskat antal decimaler. **names**() som vi använt tidigare för kolumnnamn för data frames kan även användas för kolumnnamn för tabeller (både att ta fram namnen och att ändra namnen). Funktionen är inte nödvändig för att skapa ett cirkeldiagram, men den kan utnyttjas för att ge passande etiketter för de olika kategorierna. Ett alternativ är att skriva ut namnen manuellt, exempelvis kategorinamn = c("Alive", "Dead"). Lägg då märke till att ordningen spelar roll.

I funktionen **pie**() läggs de olika argumenten in för att skapa ett cirkeldiagram. Det första argumentet kommer att avgöra hur stora delarna blir. Det andra argumentet (**labels**) markerar etiketterna bredvid varje "tårtbit" (i detta fall hur stora andelar det finns i varje kategori) och det tredje argumentet (**col**) bestämmer färgen i själva figuren.

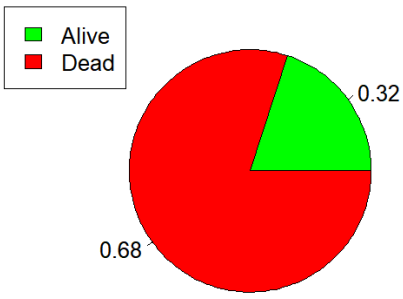
Efter pie() används även funktionen **legend**() som kan användas då man vill rita flera olika variabler eller flera olika kategorier. Det första argumentet i funktionen säger var man vill placera etiketterna för de olika kategorierna (i detta fall högst upp till höger). Det andra argumentet (legend) säger vilka namn man vill ha på etiketterna (kategorinamn används här). Det tredje argumentet (fill) specificerar vilka färger man vill ha (samma färger som vi använde oss av i pie()), här är det viktigt att färgerna kommer i rätt ordning.

Uppgift 5.2

Skapa ett cirkeldiagram av variabeln Class i titanic_small. Använd dig gärna av uträkningarna från uppgift 4.2 för att skapa grafen. Välj själv om du vill illustrera det som andelar eller i procent.

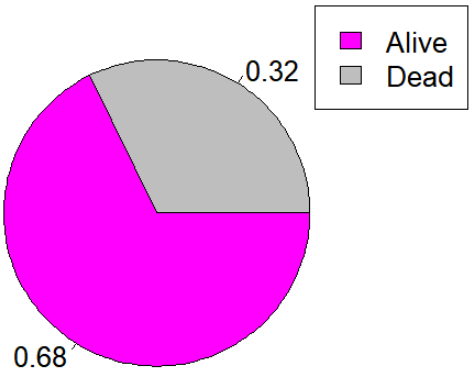
Uppgift 5.3

Vad är det för fel på detta diagram?



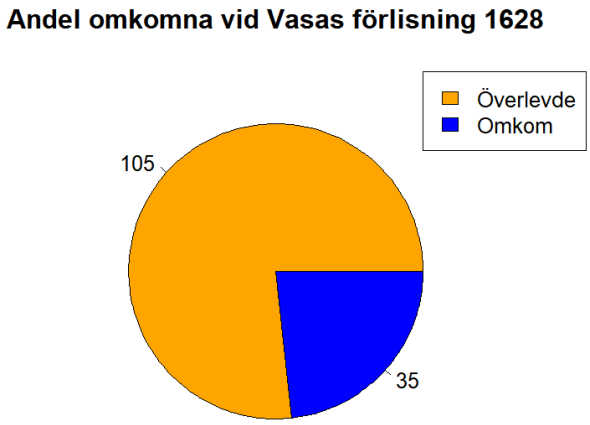
Uppgift 5.4

Givet vad du vet om överlevandegraden, vad är det för fel på detta diagram?

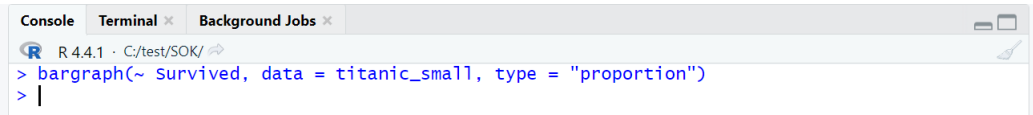


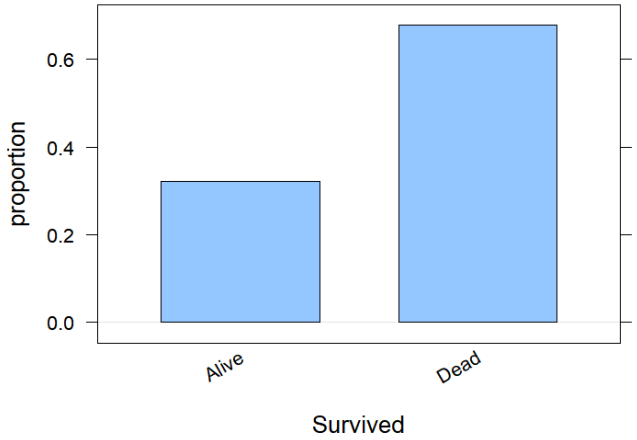
Uppgift 5.5

Enligt en (mycket osäker) uppgift hade regalskeppet Vasa en besättning på 150 personer varav 35 (23.3%) omkom vid förlisningen 1628. Hur många fel kan du hitta i följande figur?



Nedan finns ett exempel på hur man kan använda sig av **stapeldiagram**. Detta görs enklast med funktionen **bargraph**(), som liknar tally()-funktionen i och med att den börjar med ett tildetecken (~) följt av variabeln som vi vill illustrera och sedan datasetet variabeln kommer ifrån. Lägg dock märke till att istället för att skriva format = “proportion” skriver man här **type** = “proportion” om man vill visa staplarna som relativa frekvenser.





Här hade du istället kunnat skriva kommandot bargraph(~titanic$Survived, type = "proportion"), dvs du hämtar ut variabeln Survived med $-tecken från din data frame titanic, istället för att ange argumentet **data**.

Diagrammet ovan säger kanske inte så mycket. Från vilket sammanhang kommer “Alive” och “Dead”? Det finns ingen rubrik. Man kanske vill justera y-axeln, antingen så att den blir lite kortare eller lite längre.

Man kan börja med att skapa en rubrik. Detta kan exempelvis göras nedan genom att skapa en ny variabel ”my_title” som är en textsträng. Man kan också definiera passande namn för x-axeln och y-axeln, dessa är också textsträngar.

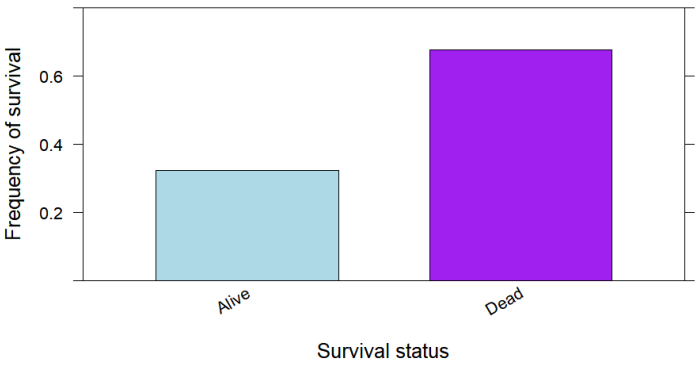
För att justera längden på y-axeln kan man skapa en vektor med passande värden. Här används en vektor med namnet length_y_axis som börjar på 0 och slutar på 0.80. Man kan även justera staplarna till valfria färger.

Sedan kan man enkelt sätta argumenten i bargraph() funktionen till de fördefinierade värdena:

```
my_title = "Figure 1: Frequency of Survival on the Titanic"
x_axis_title = "Survival status"
y_axis_title = "Frequency of survival"
length_y_axis = c(0, 0.80)
colors = c("lightblue", "purple")

bargraph(~ Survived, data = titanic_small, type = "proportion", main = my_title,
         xlab = x_axis_title, ylab = y_axis_title, ylim = length_y_axis, col = colors)
```

Figure 1: Frequency of Survival on the Titanic

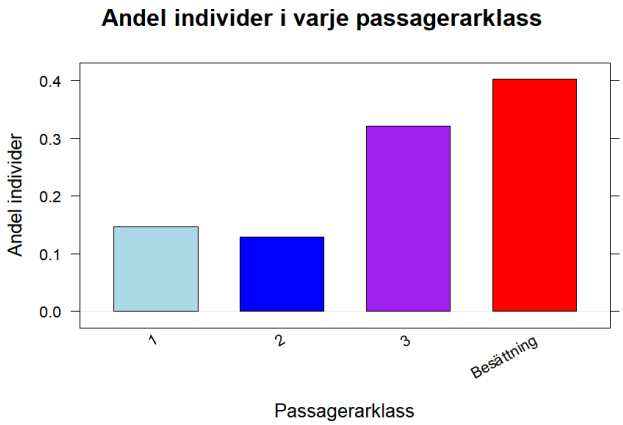


---

*När du gör en figur i R (och också andra kommandon i R) kan du antingen göra som ovan, dvs. att du skapar separata objekt (exv. **my_title**) som du sen använder inne i funktionen (main = **my_title** används inne i funktionen bargraph). Alternativt kan du skriva main="Figure 1: Frequency of Survival on the Titanic" direkt inne i bargraph-kommandot, båda varianterna fungerar. En nackdel med att skriva allt direkt inne i kommandot är att koden kan bli mindre överskådlig, och du kanske också vill återanvända vissa axelnamn, etc.*

Uppgift 5.6

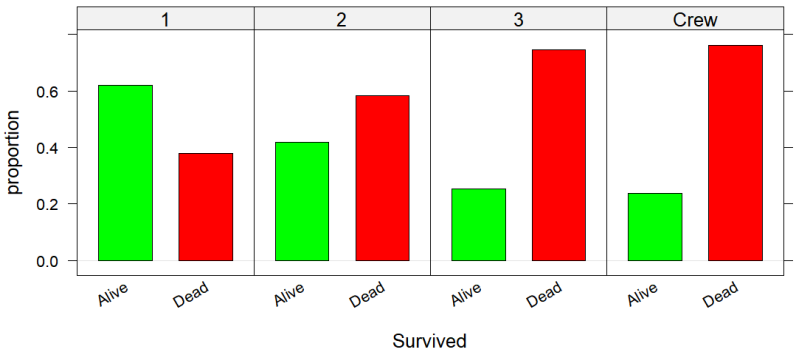
Återskapa följande figur från föreläsning 2 (att färgerna är samma är inte så noga).



6. Grafisk illustration av två kategoriska variabler – grupperat stapeldiagram

Vi kan också använda bargraph för att ta fram betingade fördelningar, exempelvis hur överlevandegraden varierade mellan passagerarklasserna på Titanic:

```
Console   Terminal   Background Jobs
R 4.4.1 · C:/test/SOK/
> bargraph(~Survived|Class, data=titanic_small, type="proportion", col=c("green", "red"))
> 
```

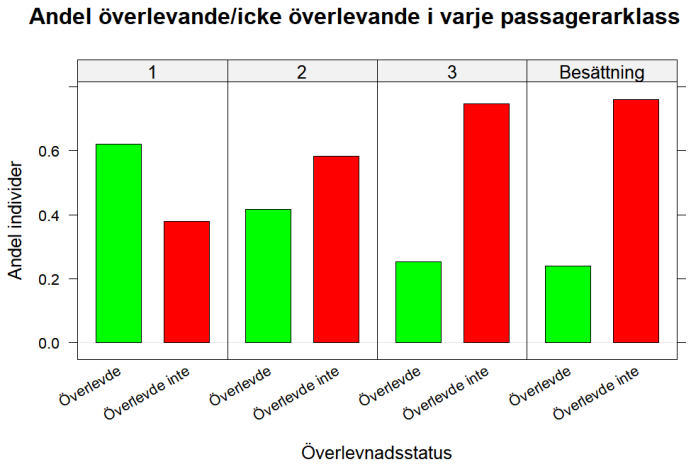


När vi betingar skriver vi återigen tecknet ”|” mellan variablerna (jämför kapitel 3 ovan), och ”~Survived|Class” läses som ”Survived givet Class” (eller Survived betingat på Class).

I det grupperade stapeldiagrammet ser man ganska tydligt att överlevandegraden sjunker då biljettklassen går från första till andra till tredje klass. Vi kan ändra rubrik och annan information:

I koden nedan har vi radbrytningar mellan olika argument, för att koden ska vara lättare att läsa. När du skriver kod i din textfil, ha för vana att göra radbrytningar mellan olika argument, åtminstone om koden blir lång.

```
bargraph(~Survived|Class,
         data=titanic_small,
         type="proportion",
         main=paste("Andel överlevande/icke överlevande i varje passagerarklass"),
         xlab=c("Överlevnadsstatus"),
         ylab=c("Andel individer"), col=c("green", "red")
)
```



Uppgift 6.1

Gör ett diagram som ovanstående men med data från landlocked, där du betingar variabeln ”utan_kust” på variabeln ”världsdel”. Du borde få 12 staplar totalt.

Uppgift 6.2* (vid intresse)

I appendix finns ett avsnitt om **staplade stapeldiagram**, som är ett alternativ till grupperade stapeldiagram.

7. Dataset som består av saknade värden

Låt oss titta närmare på variabeln Age (ålder) i datasetet titanic_small. Denna variabel har en del saknade värden och vissa standard-funktioner i R är känsliga för sådana. Man kan enkelt se om en variabel har saknade värden genom att använda **summary**() funktionen, som räknar ute lite sammanfattningsmått (mer om det senare) för variablerna i datamaterialet. R skriver **NA** (Not Available) eller **NaN** (Not A Number) för saknade värden, och vi kan se att variabeln Age har 3 saknade värden:

```
Console   Terminal   Background Jobs
R 4.4.1 · C:/test/SOK/
> summary(titanic_small)
     Name              Survived            Class                Age        Adut_or_Chld          Sex                Paid
 Length:2208        Length:2208        Length:2208        Min.   : 0.08   Length:2208        Length:2208        Min.   :  0.000
 Class :character   Class :character   Class :character   1st Qu.:22.00   Class :character   Class :character   1st Qu.:  7.896
 Mode  :character   Mode  :character   Mode  :character   Median :29.00   Mode  :character   Mode  :character   Median : 14.400
                                                          Mean   :30.15                                         Mean   : 33.010
                                                          3rd Qu.:37.00                                         3rd Qu.: 31.000
                                                          Max.   :74.00                                         Max.   :512.329
                                                          NA's   :3                                             NA's   :890
```

Det finns olika sätt att hantera saknade värden. Många funktioner kan hantera uträkningar för variabler med saknade värden om man specificerar det i funktionen. I detta fall är vi bara intresserade av en variabel och inte hela datasetet. Vi kommer därför ta bort de observationer som har saknade värden. Men i vanliga fall måste man tänka efter innan man tar bort sådana observationer, speciellt om man tittar på flera variabler samtidigt. Det finns mer att säga om att bara ta bort saknade värden när vi kommer till den statistiska analysen.

Funktionen **na.omit**() tar bort alla observationer som har NA eller NaN, dvs saknade värden.

Skapa en variabel Age som är lika med variabeln titanic$Age, men utan de saknade värdena. Titta sen hur många värden titanic$Age respektive Age har, med kommandot **length**():

```
Console  Terminal  Background Jobs
R 4.4.1 · C:/test/SOK/
> Age <- na.omit(titanic_small$Age)
> length(titanic_small$Age)
[1] 2208
> length(Age)
[1] 2205
>
```

Uppgift 7.1

Definiera en ny variabel Paid från kolumnen Paid i datasetet titanic_small, på liknande sätt som ovan. Rensa bort eventuella saknade värden med funktionen na.omit(). Hur många saknade värden har titanic_small$Paid? Hur många värden ingår i Paid? Bekräfta också att Paid dyker upp under Environment-fliken.

8. Grafisk presentation av numeriska variabler, lägesmått, spridningsmått

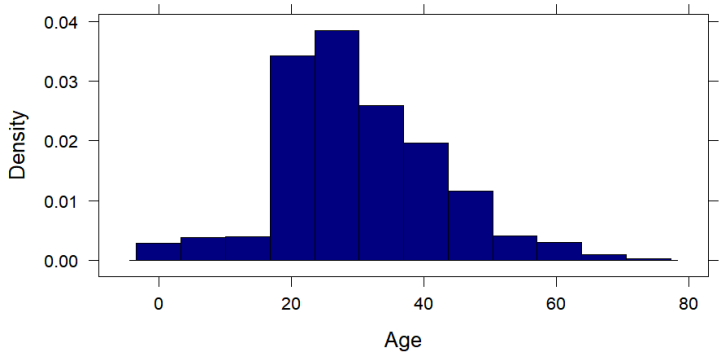
Numeriska variabler är sådana som naturligt består av numeriska värden. Som vi såg i föreläsning 2 kan numeriska variabler ha olika skaltyper. Numeriska variabler innehåller rikare information än kategoriska variabler, och man kan få ut mer information både numeriskt och grafiskt. En numerisk variabel kan alltid kodas om till en kategorisk variabel (en variabel på en lägre skalnivå) men det omvända gäller inte.

Att illustrera numeriska variabler grafiskt är betydelsefullt för att upptäcka om en fördelning har avvikande värden (så kallade **outliers**), eller om fördelningen ser **symmetrisk** eller **skev** ut, men också för att upptäcka om det är en **unimodal** eller **multimodal** fördelning. Vid de två senare fallen är histogram användbara.

De vanligaste graferna för numeriska variabler är **histogram** och låddiagram (**boxplots**).

Histogram kan ritas med hjälp av funktionen **histogram**() som finns i mosaic-paketet, eller med funktionen hist() som inte kräver något paket. Båda har sina fördelar, där histogram är mer smidig när man vill rita flera histogram samtidigt. Här är ett exempel med funktionen histogram():

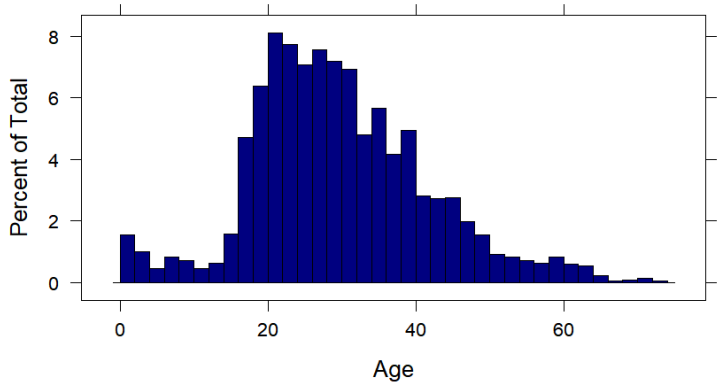
```
Console  Terminal  Background Jobs
R 4.4.1 · C:/test/SOK/
> histogram(~Age, data = titanic_small, col = "navyblue")
```



Fördelningen är inte symmetrisk här utan snarare **skev åt höger**, dvs. det finns en lång högersvans. Men det finns även en hel del observationer bestående av barn under fem år. Eftersom fördelningen är skev är det svårare att gissa sig till medelvärdet. De flesta individerna verkar dock ha varit mellan 15-45 år.

Argumentet **breaks** kan användas inuti funktionen histogram() för att ange hur många staplar man vill ha. R kommer då i så hög mån som möjligt rita ett histogram med det antal staplar som valts, här är ett exempel:

```
Console  Terminal  Background Jobs
R 4.4.1 · C:/test/SOK/
> histogram(~ Age, data = titanic_small, col = "navyblue", breaks=40)
>
```



Lite informellt kan man säga att i histogram vill du ha så många staplar att du inte döljer viktig variation som finns i variabeln du plottar, men du vill inte ha så många staplar att små (kanske slumpmässiga) variationer mellan värden ger stora utslag i hur histogrammet ser ut.

Uppgift 8.1

Rita ett histogram över variabeln Paid (hur mycket Titanicresan kostade). Testa att variera antalet staplar / stapelbredden.

Enheten för Paid är i pund.

Vid intresse (*)

Se appendix för täthetsplott (density plot), en typ av ”utjämnad” version av ett histogram.

Uppgift 8.2

Både **summary**() och **favstats**() ger dig läges- och spridningsmått över variabler, här ingår bland annat medelvärde, median och standardavvikelse (standard deviation). Du kan också använda **mean**(), **median**() och **sd**(), med flera kommandon.

Kör samtliga tio kommandon nedan (fem för Age och fem för Paid) och jämför dels vad respektive kommando gör, och jämför också (i uppgift nedan) hur Age resp. Paid kan analyseras. Vi ser att favstats (från mosaic-paketet) ger oss lite mer information än summary. Med favstats ser vi också att vi inte har några saknade värden (missing), vilket är förväntat eftersom vi skapade variablerna efter att ha tagit bort just saknade värden.

```
Console  Terminal  Background Jobs
R 4.4.1 · C:/test/SOK/
> summary(Age)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
   0.08   22.00   29.00   30.15   37.00   74.00
> favstats(Age)
  min Q1 median Q3 max     mean       sd    n missing
 0.08 22     29 37  74 30.14689 11.97386 2205       0
> mean(Age)
[1] 30.14689
> median(Age)
[1] 29
> sd(Age)
[1] 11.97386
>
```

```
Console  Terminal  Background Jobs
R 4.4.1 · C:/test/SOK/
> summary(Paid)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
  0.000   7.896  14.400  33.010  31.000 512.329
> favstats(Paid)
 min       Q1 median Q3      max     mean       sd    n missing
   0 7.895833   14.4 31 512.3292 33.01039 51.91079 1318       0
> mean(Paid)
[1] 33.01039
> median(Paid)
[1] 14.4
> sd(Paid)
[1] 51.91079
>
```

Uppgift 8.3

Innan du fortsätter med Paid och Age, tänk dig att du bor i en fiktiv och mycket liten kommun med 100 invånare, varav en individ grundade ett stort IT-företag.

Person 1-99 tjänar 1000 kronor per dag, person 100 (IT-grundaren) tjänar 901000 kronor per dag.

Hur stor är medelinkomsten, hur stor är medianinkomsten, och vilket av de två måtten sammanfattar bäst, enligt dig, inkomstläget i kommunen?

Använd R som miniräknare: Medelvärdet kan beräknas som (1/100)*(99*1000 + 901000)

Uppgift 8.4

Jämför medel och median för Age. Vilket av måtten sammanfattar bäst variabeln ålder?

(Gå också gärna tillbaka till histogrammet över Age)

Uppgift 8.5

Jämför medel och median för Paid. Vilket av måtten sammanfattar bäst variabeln? (kan det bero på?)

(Gå också gärna tillbaka till histogrammet över Paid)

Uppgift 8.6 – varians, standardavvikelse (vi kommer också ha en liknande övning på en räkneövning)

Vi ska nu, med R, räkna ut standardavvikelsen för talsekvensen bestående av de fyra talen {1, 2, 3, 4}

Vi behöver då medelvärdet (föreläsning 3):

- **Medelvärdet** av n värden $x_1, x_2, \ldots, x_n$ skrivs som $\bar{x}$ och ges av:

$$\bar{x} = \frac{1}{n}\sum_{i=1}^{n} x_i$$

Standardavvikelsen är roten ur variansen och variansen definieras som följer (föreläsning 3):

- **Variansen** av n värden $x_1, x_2, \ldots, x_n$ skrivs som s^2 och ges av:

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

Vi har n=4 och x_1=1, x_2 =2, x_3=3, x_4 =4.

Räkna först ut variansen, på två sätt, med R.

Steg för steg (gå igenom dessa steg):

1. Skapa en vektor **x_vektor1** med de fyra värdena ovan, med funktionen **c**(1, 2, 3, 4).
2. Räkna ut medelvärdet (och skapa variablen) **x_medel** = mean(x_vektor1)
3. Skapa en ny vektor **x_vektor2** som är lika med x_vektor1 minus x_medel
   (medelvärdet dras från varje tal i x_vektor1)
4. Skapa en ny vektor **x_vektor2_kvadrerad** = x_vektor2^2
   (varje tal i x_vektor2 kvadreras)
5. Summera alla tal i x_vektor2_kvadrerad med **sum**(x_vektor2_kvadrerad)
   (vi har nu ett enda tal)

6. Dividera summan med 4-1 = 3 (se formeln) för att få fram variansen

Med ett enda kommando:

7. Bekräfta att du får samma svar i steg 6 som om du använder kommandot **var**(x_vektor1)

Bekräfta sen att standardavvikelsen, dvs roten ur ditt resultat från steg 6 (använd kommandot square root, **sqrt**()), blir samma som om du använder kommandot sd (standard deviation) direkt, dvs. **sd**(x_vektor1).

Uppgift 8.7

Gå igenom alla mått som **favstats** ovan skriver ut och jämför med föreläsning 3.

---

*Om du vill, rensa i din lista under Environment-fliken med kommandot remove (rm(), exempelvis **rm**(x_vektor1)*

---

9 Boxplot och samband mellan numeriska och kategoriska variabler

I föreläsning 3 såg vi ett exempel med tre histogram, sida vid sida, med fördelningen av kostnaden för Titanicresan, betingat på passagerarklass.

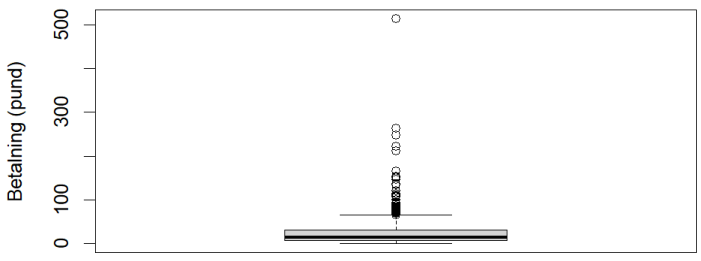
Ett annat sätt att illustrera samband mellan en numerisk variabel och kategoriska variabler är att använda en **boxplot** där man betingar på en eller flera kategoriska variabler. Vi går tillbaka till vår data frame titanic_small.

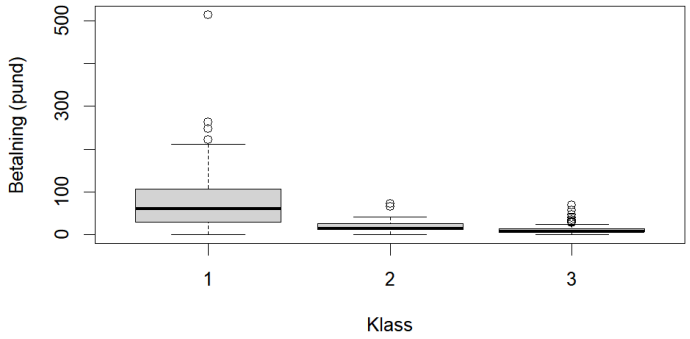
Följande kod skapar först en ny data frame titanic_smaller genom att ta bort alla variabler utom Paid och Class ur titanic_small (genom ett kommando som behåller bara kolumn 3 och 7), sedan tar koden bort alla rader som innehåller något saknat värde, sedan gör vi en första boxplot för Paid och en andra boxplot av Paid betingad på Class.

```
Console  Terminal  Background Jobs
R 4.4.1 · C:/test/SOK/
> titanic_smaller <- titanic_small[,c(3,7)]
> titanic_smaller <- na.omit(titanic_smaller)
> boxplot(Paid, data=titanic_smaller, ylab="Betalning (pund)")
> boxplot(Paid~Class, data=titanic_smaller, xlab="Klass", ylab="Betalning (pund)")
>
```

Notera att i boxplot-kommandot skrivs “betingat på” annorlunda än i tally och i bargraph, här har vi tilde mellan variablerna. Vi får följande två plottar, dessa kan sedan rubriksättas etc., på samma sätt som med tidigare plottar.

Den första plotten visar fördelningen av alla betalningar. Nästa plot visar betalningsfördelningen inom varje klass.





Uppgift 9.2

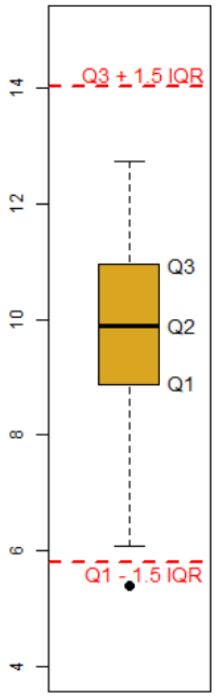
Komplettera ovanstående två plottar med lämpliga rubriker.

-

Här kommer till sist **mer detaljerad information om vad en boxplot visar**.

Medan ett histogram ger en mer komplett bild av fördelningen visar låddiagrammet (boxplot) (även kallat lådagram) läges- och spridningsmått (se föreläsning 3):

1. Medianen, Första kvartilen (Q1), Tredje kvartilen (Q3), Kvartilavståndet (IQR), största och minsta observation.
2. Figuren består av en låda (box), morrhår (whiskers) och (eventuellt) enskilda observationer/punkter.
3. Undre kanten av lådan mäter Q1.
4. Övre kanten av lådan mäter Q3.
5. Linjen som går genom lådan mäter medianen (Q2).
6. Lådans höjd mäter kvartilavståndet (IQR).
7. Stödlinjerna (röda i bilden) är inte en del av diagrammet. De är placerade vid Q1−1.5·IQR respektive Q3+1.5·IQR
8. Morrhåren sträcker sig till det minsta respektive största värde som ligger innanför stödlinjerna.
9. Eventuella observationer som ligger utanför stödlinjerna räknas som (potentiella) **outliers** och ritas ut som punkter.
10. Diagrammet kan också vara liggande.

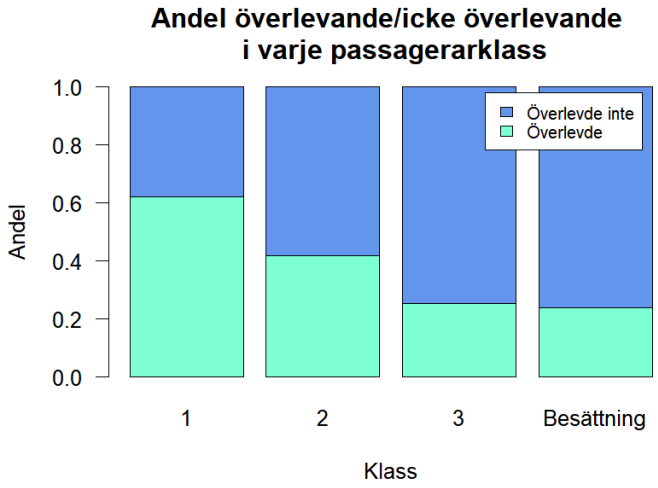


Appendix (* - vid intresse)

Två kategoriska variabler - Staplade stapeldiagram

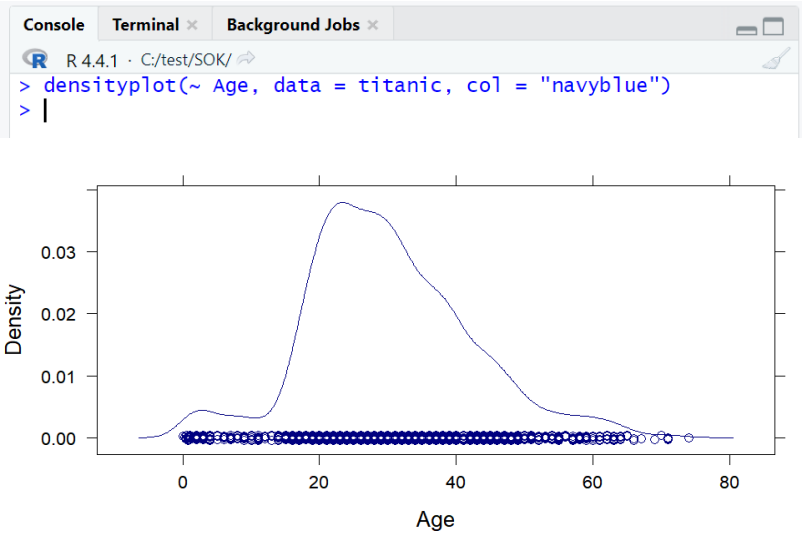
Vi kan använda kommandot **barplot** för att göra staplade stapeldiagram. Vi visar samma information om överlevandegrad på Titanic som i diagrammen i kapitel 6 men varje klass har nu en enda stapel. Nedan använder vi först tally och gör en tabell, tabell6, där Survived betingas på Class. tabell6 är sen ett argument i barplot, tillsammans med andra argument som bestämmer utseendet på figuren (las används för orienteringen på siffrorna, testa att ändra, och googla eller använd hjälpfunktionen i R). Vi har också en ruta som förklarar färgernas innebörd (legend), där cex bestämmer storleken (testa att ändra, och googla eller använd hjälpfunktionen i R).

```
tabell6 <- tally(~ Survived | Class, data = titanic_small, format = "prop")
barplot(tabell6, col = c("aquamarine", "cornflowerblue"),
        xlab = "Klass", ylab = "Andel",
        ylim = c(0,1),
        main=paste("Andel överlevande/icke överlevande", "\n", "i varje passagerarklass"),
        las=1)
legend(x=3.33, y=0.98, legend=c("Överlevde inte", "Överlevde"),
       fill = c("cornflowerblue", "aquamarine"), cex=0.75)
```



Täthetsplot

Ett histograms utseende beror på antalet staplar man väljer. En täthetsplot är ett sätt att plotta fördelningen av data som inte beror på antalet staplar. En täthetsplot kan ritas med hjälp av funktionen densityplot() som finns i mosaic-paketet.



Kortkommandon etc.

Description of Shortcuts and Special Characters in R

Description	Windows/Linux	Mac
Run code	Ctrl + Enter	⌘ + Enter
Run code and stay on current line	Alt + Enter	Option + Enter
Select All	Ctrl + A	⌘ + A
Copy	Ctrl + C	⌘ + C
Paste	Ctrl + V	⌘ + V
Cut selected text	Ctrl + X	⌘ + X
Delete line	Ctrl + D	⌘ + D
Undo	Ctrl + Z	⌘ + Z
Redo	Ctrl + Shift + Z	⌘ + Shift + Z
Select multiple lines/area	Shift + arrow key	Shift + arrow key
Duplicate line/area	Ctrl + Shift + D	⌘ + Shift + D
Scroll up/down	Ctrl + up/down arrow key	⌘ + up/down arrow key
Go to the top	Ctrl + Home	⌘ + Home
Go to the bottom	Ctrl + End	⌘ + End
Go to line	Shift + Alt + G	⌘ + Shift + Option + G
Save	Ctrl + S	⌘ + S
Save all documents	Ctrl + Alt + S	⌘ + Option + S
Open document	Ctrl + O	⌘ + O
$ symbol	Ctrl + Alt + 4	Option + 4
Vertical bar/pipe (logical OR): |	Ctrl + Alt + < or Alt gr + <	Option + 7
Assignment (<-)	Alt + -	Option + -
Zoom out/in	Ctrl + -/+	⌘ + -/+
Interrupt running code	Esc	Esc
Clear Console	Ctrl + L	⌘ + Option + L
Left square bracket [	Ctrl + Alt + 8 or Alt gr + 8	Option + [or Option + 8
Right square bracket]	Ctrl + Alt + 9 or Alt gr + 9	Option +] or Option + 9
Left curly brace {	Ctrl + Alt + 7 or Alt gr + 7	Option + Shift + 8
Right curly brace }	Ctrl + Alt + 0 or Alt gr + 0	Option + Shift + 9
Slash /	Shift + 7	Shift + 7
Backslash \	Alt gr + + or Ctrl + Alt + +	Option + Shift + 7
Tilde symbol ~	Alt gr + ~(key near Å)	Option + ~(key near Å)
Search expression	Ctrl + F	⌘ + F
Arrange indentation	Ctrl + I	⌘ + I
Show list of common shortcuts	Shift + Alt + K	Option + Shift + K

Denna version av dokumentet: 260331

Materialet i Statistisk översiktskurs har tagits fram av Ulf Högnäs och Anders Fredriksson, med inspiration och ibland direkt användande av material från andra kurser och personer, bland annat kurserna Statistik och dataanalys 1-3, med material av Michael Carlson, Ellinor Fackle Fornius, Jessica Franzén, Oskar Gustafsson, Oscar Oelrich, Mona Sfaxi, Karl Sigfrid, Mattias Villani, Valentin Zulj, med flera.