R-programmering VT2022

Föreläsning 1

Josef Wilzén

2023-01-23

Linköpings Universitet

Föreläsning 1:

- Introduktion till kursen
- R, RStudio
- Introduktion till R-programering
  - Miniräknare
  - Variabler
  - Vektorer
  - Hjälp
  - Funktioner
  - Logik

Vilka är vi?

Föreläsare och examinator:

- Josef Wilzén

Labbassistenter:

- Jack Hagerstål
- Oskar Storberg
- William Wiik

Kursens mål

Information om kursen finns i kursplanen

Lärandemål

- skapa enkla program i programspråket R med hjälp av grundläggande programmeringstekniker som inläsning och utskrift av data, tilldelning och manipulation av datastrukturer, skriva egna funktioner, upprepningar och villkorsstyrda satser.

Kursens mål

Information om kursen finns i kursplanen

Lärandemål

- skapa enkla program i programspråket R med hjälp av grundläggande programmeringstekniker som inläsning och utskrift av data, tilldelning och manipulation av datastrukturer, skriva egna funktioner, upprepningar och villkorsstyrda satser.

Vi sammanfattar detta till

- Bli bekväm med att använda R
- Hantera data med R
- Skriva program i R

Tidigare år

kursutvärderingar 2021:

1. Kursens ämnesinnehåll har gett mig möjlighet att uppnå kursens lärandemål. 3.54
2. Kursens olika undervisnings- och arbetsformer har varit relevanta i relation till kursens lärandemål. 3.86
3. Kursens examinerande moment har varit relevanta i relation till kursens lärandemål. 3.54
4. Vilket helhetsbetyg ger du kursen? 3.29

Annan föreläsare och examinator förra året.

- Mindre förändringar kommer att ske i föreläsningar/laborationer
- Minska ner omfånget på inlämningsuppgifter något
- Minska ner antalet uppgifter på tentan: 4 istället för 5
- Seminarier (från kursvecka 2)

Kursupplägg

Kursen består av två delar:

- Del 1: Grundläggande programering
- Del 2: Tillämpningar relaterade till statistik, grafik och datahantering

Kursupplägg

Kursen består av två delar:

- Del 1: Grundläggande programering
- Del 2: Tillämpningar relaterade till statistik, grafik och datahantering

Varje vecka

- En föreläsning
- Datorlaboration
  - Övningsuppgifter $\rightarrow$ viktigt!
  - Obligatoriska inlämningsuppgifter
- 4 timmar lärarledd laboration
- Inlämning:
  - Labbar: varje söndag kl. 18:00 via LISAM
- Seminarie (från kursvecka 2): koddemo, lösningar på övningsuppgifter, frågor,mm

Del 1: Grundläggande programering

- Grunderna i R
  - Lära sig hantera RStudio
- Fyra föreläsningar
- Fyra inlämningar (datorlaborationer)
- Labbarna görs **en och en**

Del 2: Tillämpningar

- Statistisk analys med R
- Fyra föreläsningar
- Fyra inlämningar
- Labbarna förs genom **parprogrammering** (grupper om två)
- **Projekt** i två delar

Del 2: Tillämpningar

- Statistisk analys med R
- Fyra föreläsningar
- Fyra inlämningar
- Labbarna förs genom **parprogrammering** (grupper om två)
- **Projekt** i två delar

Jobba med materialet och skriv egen kod!

Praktisk information

Kurslogistik

Kurshemsidan innehåller föreläsningar, labbar m.m. Finns: här

LISAM används för inlämning av labbar och kompletteringar

Teams används för kommunikation

Programvara

I denna kurs använder vi **R** och **RStudio**

Kurslitteratur I:

Kursboken

The Book Of R av Tilman M. Davies, 2016

Den finns som e-bok via biblioteket.

Artiklar

Dessa finns tillgängliga via kurshemsidan

- Dates and Times Made Easy with `lubridate`
- Handling and processing string in R
- Best practices for scientific computing

Kurslitteratur II:

Videoföreläsningar

- Google Developers videomaterial
- Roger Pengs föreläsningar

Länkar finns på kurshemsidan

Reference cards:

Olika referenskort med funktionsnamn och hjälp finns på kurshemsidan.

Examination

- Datorlaborationer, 8st
- Datortentamen i datorsal
  - Hjälpmedel: Inbyggda hjälpen i R/Rstudio, R reference card (digitalt) + några fler reference-cards. Information om vilka kommer komma på kurshemsidan. Dessa erhålls digitalt på tentamenstillfället, ni ska inte ta med er några papper.

Datorlaborationer

- Börja direkt
- Har obligatoriska inlämningsuppgifter
- Ungefär 15 h arbete per vecka.
- Laborationsmall finns på hemsidan
- Laborationer lämnas in via LISAM
- Autorättning används på en del av uppgifterna, se till att följa instruktionerna, $\rightarrow$ paketet markmyassignment
- 100% rätt för att bli godkänd

Datorlaborationer

- Arbetstakt:
  - Kursveckorna går måndag till söndag
  - Kursen går på halvfart ~20h/vecka. Ungefär 15h/vecka till labbar.
  - Deadline: söndag kl 18:00
- Kompletteringar:
  - Labb 1-4: komplettering strax efter halva kursen. Se kurshemsidan för info.
  - Komplettering i samband med tentan och omtentor.

Programmering

- Instruera en dator att utföra uppgifter

Programmering

- Instruera en dator att utföra uppgifter
- Mjukvaruutveckling - vetenskaplig programmering

Programmering

- Instruera en dator att utföra uppgifter
- Mjukvaruutveckling - vetenskaplig programmering
- Maskinkod - Lågnivåspråk - Högnivåspråk

Programmering

- Instruera en dator att utföra uppgifter
- Mjukvaruutveckling - vetenskaplig programmering
- Maskinkod - Lågnivåspråk - Högnivåspråk
- Kompileranmde språk - Interpreterande språk

Varför lära sig programera?

- Lösa problem

Varför lära sig programera?

- Lösa problem
- Hantera data av olika sorter: tabeller, databaser, listor, bilder, text

Varför lära sig programera?

- Lösa problem
- Hantera data av olika sorter: tabeller, databaser, listor, bilder, text
- Replikerbarhet

Varför lära sig programera?

- Lösa problem
- Hantera data av olika sorter: tabeller, databaser, listor, bilder, text
- Replikerbarhet
- Komplexa beräkningar

Varför lära sig programera?

- Lösa problem
- Hantera data av olika sorter: tabeller, databaser, listor, bilder, text
- Replikerbarhet
- Komplexa beräkningar
- Automatisera

Programmering och programmeringsspråk

- Programmering kräver ett programmeringsspråk

Programmering och programmeringsspråk

- Programmering kräver ett programmeringsspråk
- Det finns många olika programmeringsspråk

Programmering och programmeringsspråk

- Programmering kräver ett programmeringsspråk
- Det finns många olika programmeringsspråk
- Olika språk är bra på olika saker

Programmering och programmeringsspråk

- Programmering kräver ett programmeringsspråk
- Det finns många olika programmeringsspråk
- Olika språk är bra på olika saker
- Exempel

Programmering och programmeringsspråk

- Programmering kräver ett programmeringsspråk
- Det finns många olika programmeringsspråk
- Olika språk är bra på olika saker
- Exempel
  - Python, Javascript, C/C++, Java, Go, ...

Programmering och programmeringsspråk

- Programmering kräver ett programmeringsspråk
- Det finns många olika programmeringsspråk
- Olika språk är bra på olika saker
- Exempel
  - Python, Javascript, C/C++, Java, Go, ...
- Språk som är bra för dataanalys, statistik, maskininlärning, data science mm

Programmering och programmeringsspråk

- Programmering kräver ett programmeringsspråk
- Det finns många olika programmeringsspråk
- Olika språk är bra på olika saker
- Exempel
  - Python, Javascript, C/C++, Java, Go, ...
- Språk som är bra för dataanalys, statistik, maskininlärning, data science mm
  - R, Python, Julia, SQL, Matlab

Studieteknik

- Ni har ansvar för era egna studier och er egen inlärning

Studieteknik

- Ni har ansvar för era egna studier och er egen inlärning
- Inlärning kräver engagemang och arbete

Studieteknik

- Ni har ansvar för era egna studier och er egen inlärning
- Inlärning kräver engagemang och arbete
- Rekommenderas att ni planerar era studier: skriv ner vecka för vecka vad ni ska göra och när

Studieteknik

- Ni har ansvar för era egna studier och er egen inlärning
- Inlärning kräver engagemang och arbete
- Rekommenderas att ni planerar era studier: skriv ner vecka för vecka vad ni ska göra och när
- Programmering:

Studieteknik

- Ni har ansvar för era egna studier och er egen inlärning
- Inlärning kräver engagemang och arbete
- Rekommenderas att ni planerar era studier: skriv ner vecka för vecka vad ni ska göra och när
- Programmering:
  - Teoretisk färighet

Studieteknik

- Ni har ansvar för era egna studier och er egen inlärning
- Inlärning kräver engagemang och arbete
- Rekommenderas att ni planerar era studier: skriv ner vecka för vecka vad ni ska göra och när
- Programmering:
  - Teoretisk färighet
  - Till stor del en praktisk färdiget

Studieteknik

- Ni har ansvar för era egna studier och er egen inlärning
- Inlärning kräver engagemang och arbete
- Rekommenderas att ni planerar era studier: skriv ner vecka för vecka vad ni ska göra och när
- Programmering:
  - Teoretisk färighet
  - Till stor del en praktisk färdiget
- Viktigt att skriva kod kontinuerligt under hela kursen

Studieteknik

- Ni har ansvar för era egna studier och er egen inlärning
- Inlärning kräver engagemang och arbete
- Rekommenderas att ni planerar era studier: skriv ner vecka för vecka vad ni ska göra och när
- Programmering:
  - Teoretisk färighet
  - Till stor del en praktisk färdiget
- Viktigt att skriva kod kontinuerligt under hela kursen
- Kom förbereda till datorlaborationerna! $\rightarrow$ fråga om hjälp!!!

Studieteknik

Förslag på upplägg:

- Måndag: Föreläsning + läsa kurslitteratur $\sim$ 4 h

Studieteknik

Förslag på upplägg:

- Måndag: Föreläsning + läsa kurslitteratur $\sim$ 4 h
- Försök att arbeta $\sim$ 4 h med datorlaborationen innan ni kommer till labbpasset, per pass ($=$ 8 h)

Studieteknik

Förslag på upplägg:

- Måndag: Föreläsning + läsa kurslitteratur $\sim$ 4 h
- Försök att arbeta $\sim$ 4 h med datorlaborationen innan ni kommer till labbpasset, per pass ($=$ 8 h)
- Gå på de bokade labbpassen $\sim$ 4 h

Studieteknik

Förslag på upplägg:

- Måndag: Föreläsning + läsa kurslitteratur $\sim$ 4 h
- Försök att arbeta $\sim$ 4 h med datorlaborationen innan ni kommer till labbpasset, per pass ($=$ 8 h)
- Gå på de bokade labbpassen $\sim$ 4 h
- Egenstudier och seminarie $\sim$ 4 h

Studieteknik

Förslag på upplägg:

- Måndag: Föreläsning + läsa kurslitteratur $\sim$ 4 h
- Försök att arbeta $\sim$ 4 h med datorlaborationen innan ni kommer till labbpasset, per pass ($=$ 8 h)
- Gå på de bokade labbpassen $\sim$ 4 h
- Egenstudier och seminarie $\sim$ 4 h
- totalt: 20 h

Vad är R?

- R är ett populärt programmeringsspråk för statistiker $\rightarrow$ finns många funktioner för datahatering, statistik och visualisering

Vad är R?

- R är ett populärt programmeringsspråk för statistiker $\rightarrow$ finns många funktioner för datahatering, statistik och visualisering
- Öppen källkod

Vad är R?

- R är ett populärt programmeringsspråk för statistiker $\rightarrow$ finns många funktioner för datahatering, statistik och visualisering
- Öppen källkod
- Många utvecklare

Vad är R?

- R är ett populärt programmeringsspråk för statistiker $\rightarrow$ finns många funktioner för datahatering, statistik och visualisering
- Öppen källkod
- Många utvecklare
- Interpreterande högnivåspråk

Ett exempel på ett program i R

Skapa ett program som skriver ut talen från 10 till 1 och sen skriver “kör!”.

Ett exempel på ett program i R

Skapa ett program som skriver ut talen från 10 till 1 och sen skriver “kör!”.

I R ser det ut på följande sätt

```
start <- 10
for (i in 1:10) {
  print(start)
  start <- start - 1
}
print("Kör!")
```

Resultatet

Kör vi koden i R får vi följande resultat

```
## [1] 10
## [1] 9
## [1] 8
## [1] 7
## [1] 6
## [1] 5
## [1] 4
## [1] 3
## [1] 2
## [1] 1

## [1] "Kör!"
```

R och RStudio

- R är både ett program och ett programeringsspråk
- RStudio är en IDE för R
- Båda är gratis och går att ladda ner och installera på er egna dator.
- Se kurshemsidan för information.

Demo: RStudio

- Hur använder man RStudio?

Datorsalar: SU

- Datorlaborationerna är i SU-salarna i B-huset
- Har Linux (Ubuntu) som operativsystem
- Utanför de bokade passen: Om de är ledigt så kan ni använda SU-salar eller PC1-PC5 (E-huset) för självstudier.

Datorsalar: SU

Hur startar jag Rstudio och börjar labbba i en SU-sal?????

- Logga in med Liu-ID och lösenord
- Tryck: `Ctrl+Alt+T` för att öppna en terminal. Här kan du skriva olika kommandon, dessa aktiveras när du trycker enter.
- Skriv: `module add courses/732G33` och tryck enter
- Detta läser in kursmodulen, som innehåller de programvaror som behövs i kursen.
- Skriv `rstudio` i terminalen och tryck enter

Att hitta hjälp

- Inbyggd hjälp i R
- Sök i Google
- Sök på ENGELSKA
- Kolla på felmeddelandet

```
## Error in eval(expr, envir, enclos) : object 'x' not found
```

Variabler och vektorer

- Variabler kan spara värden
  - Sätts med <- (eller ->)
- Vektorer är en samling av likadana element
  - Skapas med `c()`
  - Välj element med `[ ]`

Exempel:

```
a <- 1
a
```

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

```
testVektor <- c(2,3,5,7,11,13)
testVektor[c(1,3)]
```

```
## [1] 2 5
```

Räkna med vektorer

- Beräkningar sker elementvis

```
testVektor <- c(2,3,5,7,11,13)
testVektor+1
```

```
## [1]  3  4  6  8 12 14
```

- Beräkningar mellan vektorer sker cykliskt

```
testVektor <- c(2,3,5,7,11,13)
testVektor+c(1,2)
```

```
## [1]  3  5  6  9 12 15
```

Olika typer av värden

- Värden kan vara en av flera olika typer
  - t.ex. heltal, flyttal, textsträngar etc.
- Dessa typer kallas atomära klasser
- Kan kolla vilken typ det är med `typeof( )`
- Kan konvertera med `as.`

```
as.character(4:8)
```

```
## [1] "4" "5" "6" "7" "8"
```

Olika typer av värden

- Värden kan vara en av flera olika typer
  - t.ex. heltal, flyttal, textsträngar etc.
- Dessa typer kallas atomära klasser
- Kan kolla vilken typ det är med `typeof( )`
- Kan konvertera med `as.`

```
as.character(4:8)
```

```
## [1] "4" "5" "6" "7" "8"
```

Beskrivning	Synonymer	`typeof()`	Exempel i R
Heltal ($\mathbb{Z}$)	`int`	`integer`	`-1, 0, 1`
Reella tal ($\mathbb{R}$)	`real`, `float`	`double`	`1.03, -2.872`
Komplexa tal ($\mathbb{C}$)	`cplx`	`complex`	`1 + 2i`
Logiska värden	`boolean`, `bool`	`logical`	`TRUE FALSE`
Textsträngar	`string`, `char`	`character`	`En textsträng`

Demo: Variabler

Funktioner i R

- En funktion utför något
- Tar noll eller flera argument
- Funktioner samlas i R-paket
- Många små funktioner, en funktion gör en sak.

Funktioner i R II

En funktion i R är uppbyggd av

- ett funktionsnamn, t.ex. `area`
- en funktionsdefinition: `function( )`
- 0 eller flera argument, t.ex. `hojd` och `bredd`
- "måsvingar" `{ }`
- kod, t.ex. `area <- hojd * bredd`
- returnera värde, t.ex. `return(area)`

Exempel på funktion i R

```r
area <- function(hojd, bredd){
  area <- hojd * bredd
  return(area)
}
```

```r
area(hojd = 2, bredd = 3)
```

```
## [1] 6
```

```r
area(hojd = 5, bredd = 11)
```

```
## [1] 55
```

Demo: Funktioner

Lokal miljö

"Det som sker i en funktion stannar i funktionen"

```
f <- function(x, y){
  z <- 5
  svar <- z*x + y
  return(svar)
}
```

`z` och `svar` kan inte användas utanför funktionen.

Lokal miljö II

```
ls()
```

```
## [1] "a"          "area"       "f"          "i"          "start"
## [6] "testVektor"
```

```
f(1,2)
```

```
## [1] 7
```

```
ls()
```

```
## [1] "a"          "area"       "f"          "i"          "start"
## [6] "testVektor"
```

Att tänka på

- Funktionen måste läsas in innan den fungerar.
- `return()` avslutar funktionen
- Skriv funktionen i flera delar
  - Skriv kod som gör det du vill
  - Lyft in koden i funktionen
  - Pröva funktionen

Demo: Funktioner

markmyassignment

- Ett R-paket
- Autorättar uppgifter
- Används i kursen för att rätta inlämnade funktioner
- Ni använder markmyassignment innan ni lämnar in

Demo: markmyassignment

Logik

- Logik är vanligt i programmering
  - Används i `if`-satser $\rightarrow$ kursvecka 3!
- I R finns de logiska värdena `TRUE`, `FALSE`, och `NA`
  - `NA` = not available = saknade värden
- Skapas på två olika sätt
  - Som vanliga vektorer
  - Genom relationsoperatorer
- Kan användas för att välja element i vektorer

Logik i R

Kan skapa en vektor med värdena `TRUE` och `FALSE`

```
testVektor <- c(2,3,5,7,11,13)
boolVektor <- c(TRUE, FALSE, FALSE, TRUE, FALSE, TRUE)
```

```
testVektor[boolVektor]
```

Logik i R

Kan skapa en vektor med värdena `TRUE` och `FALSE`

```
testVektor <- c(2,3,5,7,11,13)
boolVektor <- c(TRUE, FALSE, FALSE, TRUE, FALSE, TRUE)
```

```
testVektor[boolVektor]
```

```
## [1]  2  7 13
```

Logik i R

Kan skapa en vektor med värdena `TRUE` och `FALSE`

```
testVektor <- c(2,3,5,7,11,13)
boolVektor <- c(TRUE, FALSE, FALSE, TRUE, FALSE, TRUE)
```

```
testVektor[boolVektor]
```

```
## [1]  2  7 13
```

Kan också skapa vektor genom en relation

```
testVektor > 5
```

```
## [1] FALSE FALSE FALSE  TRUE  TRUE  TRUE
```

Relationsoperatorer

- Relationer används för att jämförelser
- Skapar logiska vektorer

Beskrivning	Operatorer i R
Lika med	`==`
Inte lika med	`!=`
Större än	`>`
Mindre än	`<`
Större än eller lika med	`>=`
Mindre än eller lika med	`<=`
Finns i	`%in%`

Logiska operatorer

- Boolesk algebra
- Operatorer:

Operator	Symbol	Operator i R
och	$\wedge$	&
eller	$\vee$	|
inte	$\neg$	!

Logiska operatorer

- Boolesk algebra
- Operatorer:

Operator	Symbol	Operator i R
och	$\wedge$	`&`
eller	$\vee$	`|`
inte	$\neg$	`!`

Symbol	A	B	$\neg A$	$A \wedge B$	$A \vee B$
i R	`A`	`B`	`!A`	`A & B`	`A | B`
	`TRUE`	`TRUE`	`FALSE`	`TRUE`	`TRUE`
	`TRUE`	`FALSE`	`FALSE`	`FALSE`	`TRUE`
	`FALSE`	`TRUE`	`TRUE`	`FALSE`	`TRUE`
	`FALSE`	`FALSE`	`TRUE`	`FALSE`	`FALSE`

Demo: Logik

Logik exempel

```
testVektor <- c(2,3,5,7,11,13,17,19,23,29,31)
boolVektor <- testVektor < 6 | !(testVektor < 20)
```

Vad blir följande uttryck?

```
testVektor[boolVektor]
```

Logik exempel

```
testVektor <- c(2,3,5,7,11,13,17,19,23,29,31)
boolVektor <- testVektor < 6 | !(testVektor < 20)
```

Vad blir följande uttryck?

```
testVektor[boolVektor]
```

```
## [1]  2  3  5 23 29 31
```