R-programmering VT2024

Föreläsning 5

Johan Alenlöv

Linköpings Universitet

Föreläsning 5

Innehåll föreläsning 5

- Information del 2
- Grafik
- Slumptal och simulering
- knitr och markdown
- Externa data och pxweb

Information del 2

R-programmering - del 2

- Del 1: Grunderna i programmering:
  - Variabler och tilldelning
  - Datastrukturer
  - Kontrollstrukturer
  - Funktioner
  - Debugging
  - Dokumentation
- Del 2: Tillämpningar

R-programmering - del 2

- Del 1: Grunderna i programmering:
- Del 2: Tillämpningar
  - Grafik
  - Slumptal
  - Statistik och analys
  - Externa data
  - Datum
  - Texthantering och regular expression
  - Linjär algebra
  - knitr, markdown, literate programming

R-programmering - del 2

- Labbarna görs nu i par
- Parprogrammering:
  - **Turas om att skriva koden**
  - Den som inte kodar är engagerad i koden och problemet
  - Byt vem som kodar var 10:e minut
  - Viktigt att kommentra koden: ROxygen och inline
  - **Båda** är delaktiga i programmeringen

Miniprojektet

Info finns på kurshemsidan.

- Hitta data på webben (eller lägg upp eget)
  - Kommunala (tvärsnitt) data
  - Tidsseriedata
- Presentera med grafik, knitr och markdown
- Rapporten ska vara reproducerbar
- Lämna in som **PDF** och **Rmd**
- Miniprojektet har egen inlämning

Basgrafik

Grafik i R

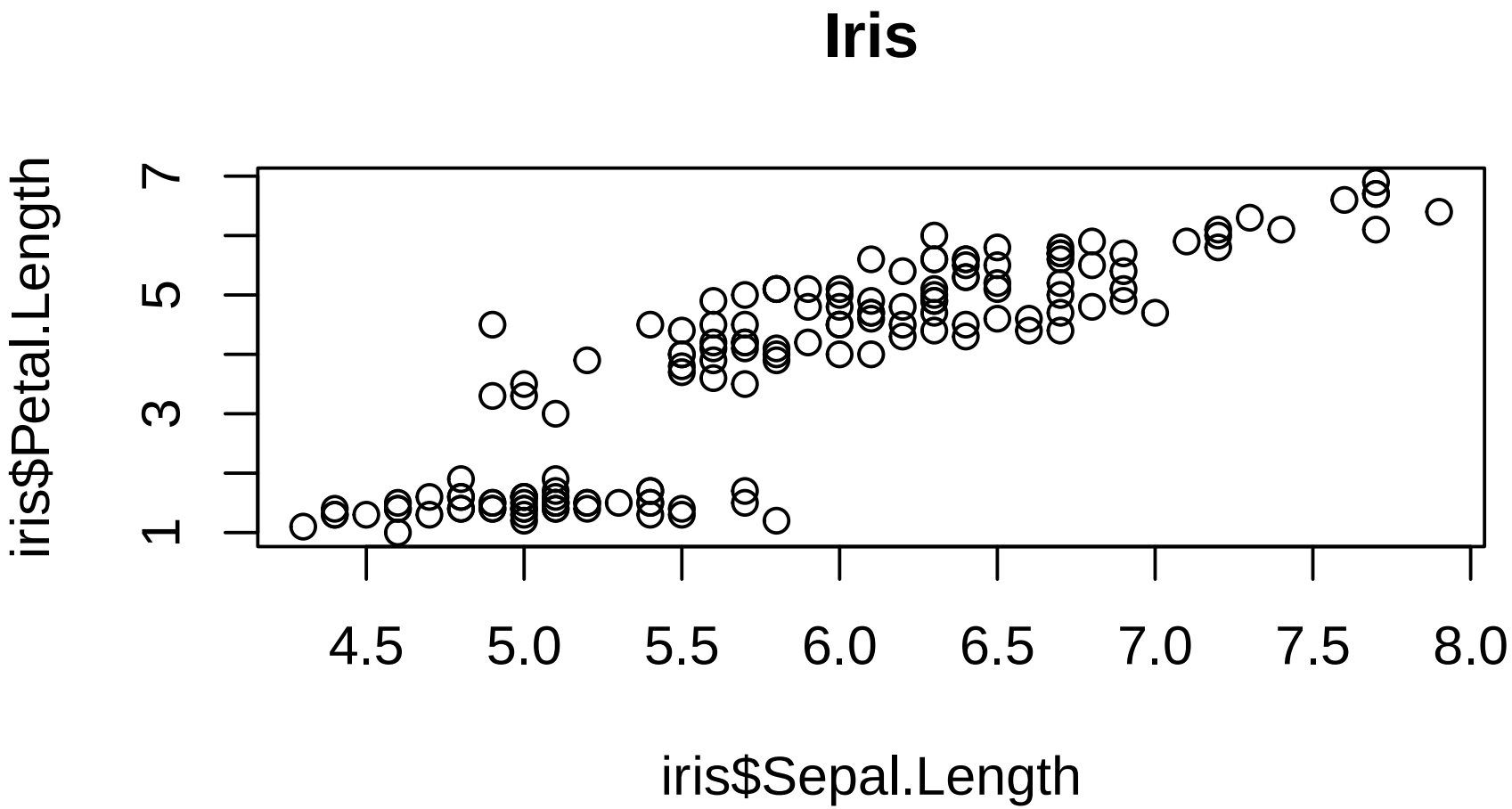
- Grafik är en styrka med R
- Massa olika paket: `ggplot2`, `lattice` m.m.
- Hög nivå (funktioner, `plot`)
- Låg nivå (bygga upp en plot steg för steg)
- Använd grafik för att:
  - Sammanfatta tabeller visuellt
  - Jämföra olika dataset
  - Rita ut funktioner

Enkel grafik: plot()

- `plot()` - kan plotta många olika objekt
  - `plot(x,y)` - ger en scatterplot
  - `plot(x)` - om `x` en `data.frame` så skapas en matrix-plot.
  - Vanliga argument:
    - `type =` Hur plotten ska se ut
    - `main =` Titel på plotten
    - `xlab =` Text på x-axeln
    - `ylab =` Text på y-axeln
    - `xlim =` Gränserna på x-axeln
    - `ylim =` Gränserna på y-axeln
    - `col =` Färgerna som ska användas

plot() - exempel

```
data(iris)
plot(x = iris$Sepal.Length,
     y = iris$Petal.Length, main = "Iris")
```

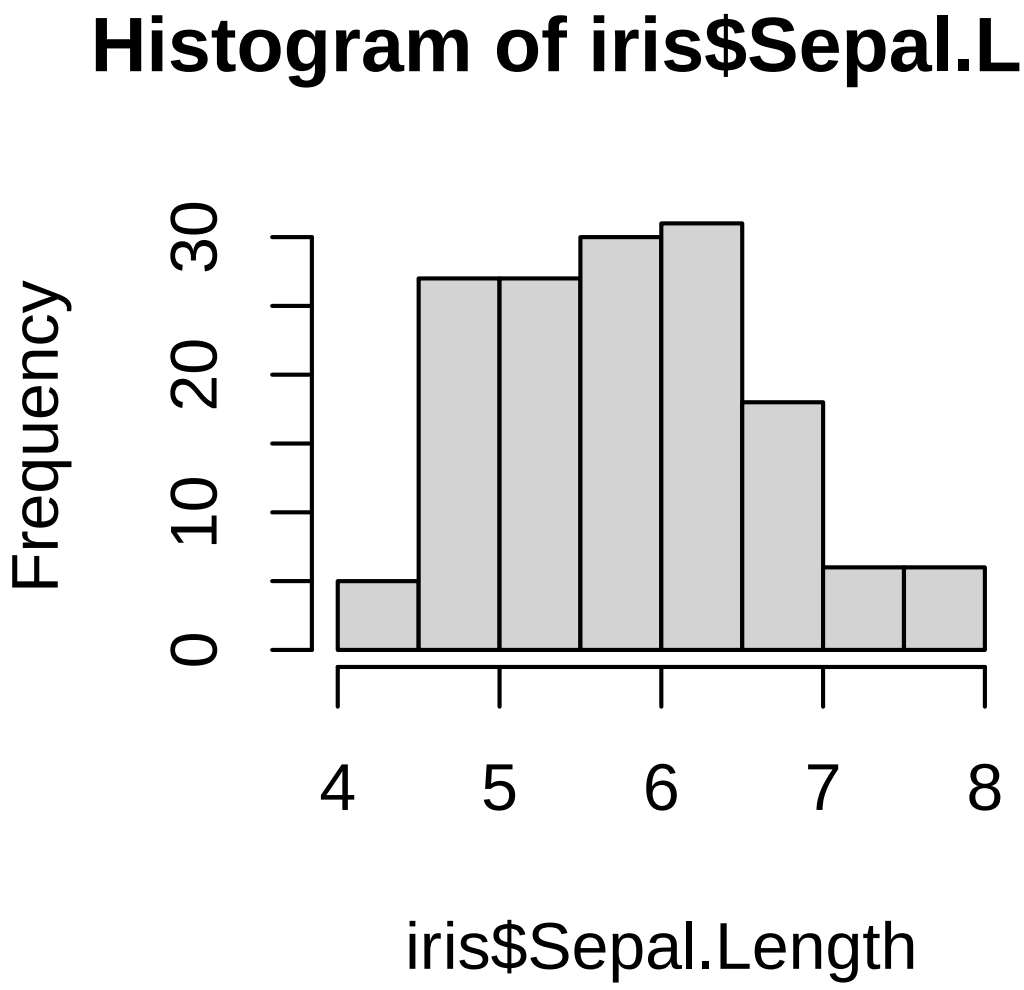


Diagramtyper

- `hist(x = , breaks = )` ger ett histogram,
  - `x` är en numerisk vektor.
  - `breaks` antalet intervall som data delas in i.
- `boxplot(x = )` ger boxplots
- `barplot(height = )` ger stapeldiagram
- `pie(x = )` geren piechart
- Använd hjälpen för att se exempel och fler argument

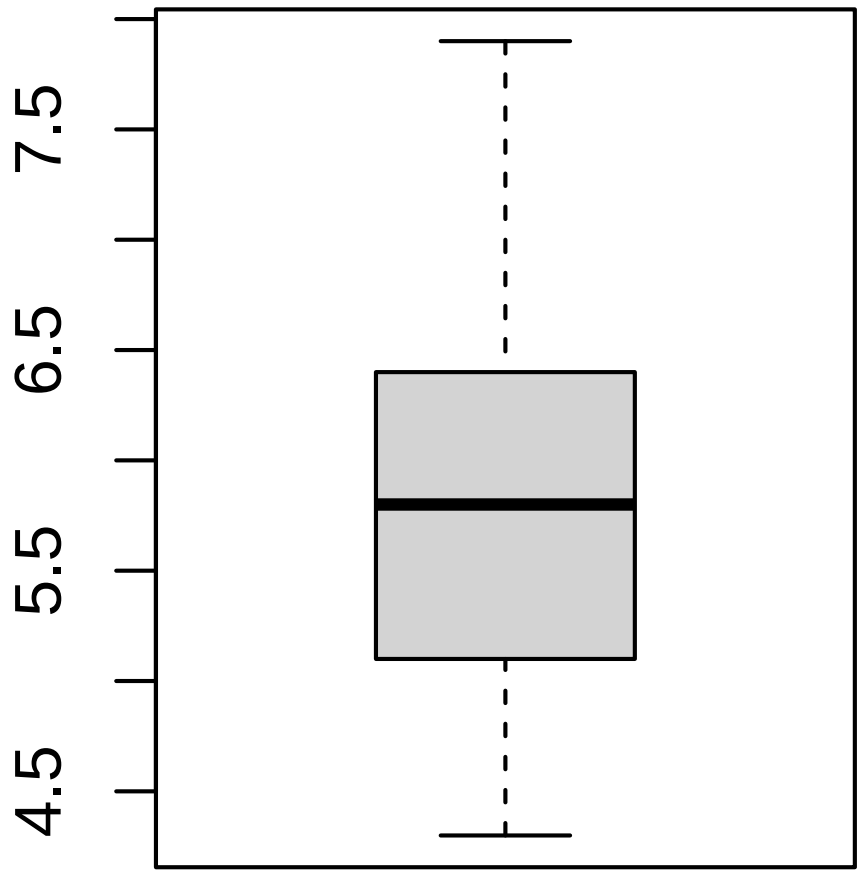
hist() - exempel

```
hist(iris$Sepal.Length)
```



bokxplot() - exempel

```
boxplot(iris$Sepal.Length)
```

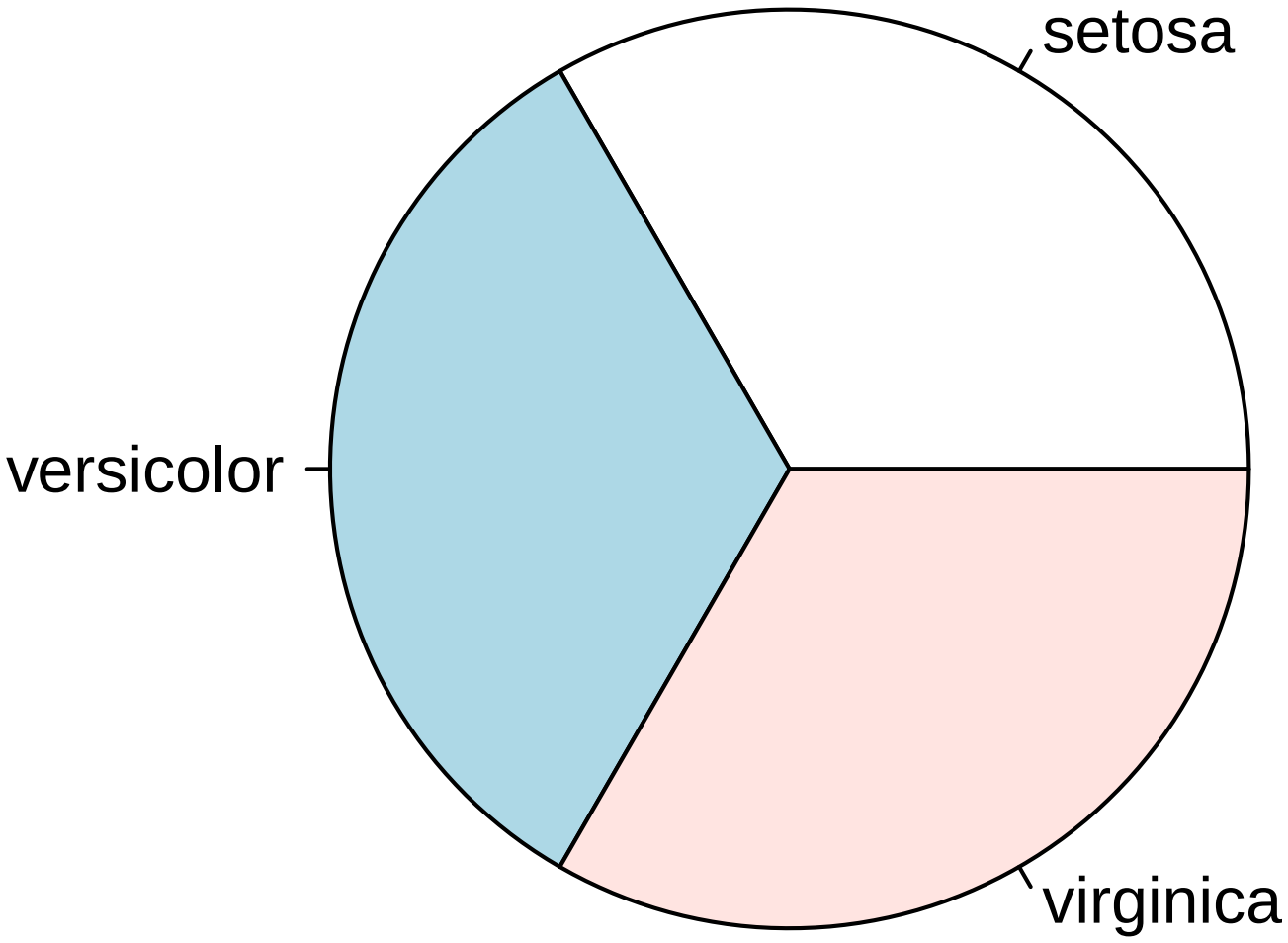


barplot() - exempel

```r
tab <- table(iris$Species)
barplot(tab)
```

pie() - exempel

```
tab <- table(iris$Species)
pie(tab)
```



Lågnivågrafik

- Kan användas för att bygga upp en graf från grunden
- Kör lager på lager:
  - `points(x, y)` lägger till punkter
  - `lines(x, y)` lägger till linjer
  - `abline(a, b, h, v)` lägger till räta linjer
  - `legend(x, y, legend)` lägger till en förklaringsruta
  - `par()` fler grafiska alternativ
- Går att bygga upp många olika sorters plottar och grafik
- Mycket smidigare att använda `ggplot2`

Demo

Demo: Grafik

Slumptal och simulering

Slumptal och simulering - I

- R har en stor uppsättning funktioner för fördelningar

prefix	Beskrivning	Exempel
`r`	simulera från fördelningen	`runif()`
`d`	täthetsfunktionen (pdf)	`dunif()`
`p`	kulmulativ fördelninsgfunktion (cdf)	`punif()`
`q`	inversa kulmulativa fördelningsfunktionen	`qunif()`

- Se `?Distributions` för fler fördelningar

Slumptal och simulering - II

- Observera att: **Det finns ingen riktig slump i datorer**
- Det finns slumptalsgeneratorer
  - Algoritmer där output ser slumpmässigt ut
- Kan styra slumpen genom att bestämma startvärdet, "slumpfröet"
  - i R använder vi `set.seed( )`
- För att dra ett slumpmässigt urval använder vi `sample( )`

Slumptal och simulering - Exempel

```
runif(n = 3, min = -1, max = 1)
```

```
## [1]  0.1166546 -0.1855843  0.7623537
```

```
set.seed(20220221)
runif(n = 3, min = -1, max = 1)
```

```
## [1] -0.2819856 -0.7837770 -0.4418922
```

```
set.seed(20220221)
runif(n = 3, min = -1, max = 1)
```

```
## [1] -0.2819856 -0.7837770 -0.4418922
```

Slumptal och simulering - Exempel

```
text <- c("Johan", "Josef", "Rasmus")
set.seed(20220221)
sample(x = text, size = 5, replace = TRUE)
```

```
## [1] "Josef"  "Johan"  "Johan"  "Josef"  "Rasmus"
```

```
sample(x = text, size = 5, replace = TRUE)
```

```
## [1] "Rasmus" "Josef"  "Josef"  "Josef"  "Josef"
```

Demo

Demo: Slumptal

R-Markdown, knitr och Notebooks

R-Markdown, knitr och Notebooks

- Kombinera text, kod och grafik i en fil
- Förenkla för era laborationer i denna och andra kurser
- Inbyggd del av R-studio
- Två delar:
  - R-Markdown (för text)
  - knitR (för R-kod)
- Kan producera, PDF, Word och/eller HTML filer
  - Alla slides och kurshemsidan är skapad med detta.
  - Laborationerna använder knitR med lyx istället för R-Markdown

R-Markdown

- Markupspråk
  - Markup används på Teams, Forum, Discord, Slack m.m.
- Väldigt enkelt att skriva
- Kan hantera matematik och matematiska formler (via LaTeX)
- Integrerat med R-Studio
  - Kan behöva installera LaTeX för att skapa PDFer
    - MikTex för Windows
    - MacTex för OS X
    - TexLive för Linux

knitr

- Kör R-kod och ersätter texten med resultatet
- Sätter ihop text och kod
- Ger dynamiska rapporter
- Skapa tabeller med `kable( )`
- Kan hantera R och Python (och annat)

R-Studio Notebooks

- Rmd-filer (samma som R-Markdown)
- Kan köras interaktivt direkt i R-Studio
- Inspirerat av Jupyter Notebooks
- Skillnad mellan R-Markdown och Notebooks:
  - Samma kod
  - i R-Markdown körs all kod när du genererar dokumentet
  - i Notebook körs en rad i taget
  - i Notebooks kan du köra kod direkt och se resultatet

Demo

Demo: Markdown

Externa data och pxweb

Externa data

- Mer och mer data finns på webben
- Vill kunna hantera data programtiskt
  - Vill kunna säga åt koden vilken data som ska användas
  - Samma data varje gång även om källmaterialet uppdateras
  - Inte vara beroende av en specifik nerladdad fil
- Kan vara lite klurigt i början
- Central del av reproducerbarheten i rapporter

Ladda ner och läsa in från webben

- Vill vi bara ladda ner: `downloader`
- Vill vi ladda in direkt i R: `repmis`

`repmis` hanterar:

- `.Rdata`
- `.csv`
- `.txt`

Vanliga källor för data

- Dropbox
- Google Docs
- Github

pxweb api

- Ett api är en dörr till ett system
- `pxweb` är ett paket för att gå in genom dörren
- Många myndigheter använder `pxweb` api
  - Till exempel SCB
- Kan användas för:
  - Navigera i datalager
  - Ladda ner förutbestämd data med kod

pxweb

```
install.packages("pxweb")
library(pxweb)

min_data <- pxweb_interactive()
```

Demo

Demo: PXWEB