

Lecture 2

Introduction II

Pedigree analysis in R with the *ped suite*

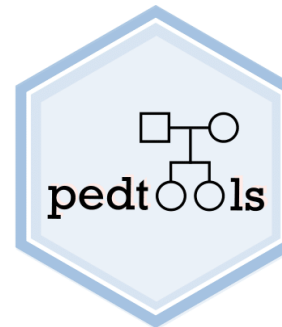
Magnus Dehli Vigeland

Statistical methods in genetic relatedness and pedigree analysis

NORBIS course, 6th – 10th of January 2020, Oslo

Plan

- What is **R**?
- Your first R session?
 - basic functionality
 - plotting
 - installing packages
- The *ped suite* packagers **pedtools** and **pedprobr**
 - creating and plotting pedigrees
 - marker genotypes
 - likelihoods





What is R?

- A framework for statistical and numerical computing
 - calculator
 - flexible plotting
 - large core of functions for data handling and numerical analysis
 - programming language
 - external packages
 - anyone can make one
 - thousands!
- About R:
 - it's free!
 - it's widely used
 - it *can* do anything (but it may not be easy)
 - challenging for beginners (*unless they have good teachers!*)



Time to get your hands dirty: Your first R session

Using R as a basic calculator

```
> 2 + 3
[1] 5
> 1+2 * 3
[1] 7
> (1 + 2) * 3
[1] 9
> 4^2
[1] 16
> exp(1)
[1] 2.718282
> log(100)
[1] 4.60517
> log(100, base = 10)
[1] 2
> log10(100)
[1] 2
```

Variables

For assignments, you have two choices: `=` or `<-`

```
> a = 5      or   a <- 5
> b = 3      or   b <- 3
> a
[1] 5
> a - 2*b
[1] -1
```

Changing a variable:

```
> a = a+1
> a
[1] 6
```

Creating new variables from old:

```
> myVariable_name = a^(b - 1)
> myVariable_name
[1] 36
```

Vectors

```
> c(3, 2, 6, -1)
[1] 3 2 6 -1
> 4:20
[1] 4 5 6 7 8 9 10 11 12
[10] 13 14 15 16 17 18 19 20
> 5:7 - 4
[1] 1 2 3
> c(10,20,30,40) + c(1,3,8,0)
[1] 11 23 38 40
> seq(from = 2, to = 15, by = 3)
[1] 2 5 8 11 14
```

There is a help page
for every function!

```
> ?seq
```

Try some variations:

```
> seq(10, by = 2, length = 5)
> seq(10, 0, by = -2)
> 5: (-5)
```

Matrix-like containers

Data frames: Each vector becomes a column

```
> x = data.frame(Name = c("Ali", "Bob", "Joe"), Weight = c(75, 81, 70))
> x
  Name Weight
1  Ali     75
2  Bob     81
3  Joe     70
```

Matrices:

```
> x = matrix(1:12, nrow = 3, ncol = 4)
> x
     [,1] [,2] [,3] [,4]
[1,]    1    4    7   10
[2,]    2    5    8   11
[3,]    3    6    9   12
```

Faster, but less flexible. Good for all-numeric data

Lists

```
> a = list(good = 1:3, bad = 0)
```

```
> a
```

```
$good
```

```
[1] 1 2 3
```

```
$bad
```

```
[1] 0
```

```
> a$good
```

```
[1] 1 2 3
```

The dollar operator:
Extracts a named list element

Easy to change lists:

```
> a$bad = NULL (delete item)
```

```
> a$ok = -1 (add new item)
```

```
> a$good = c(a$good, 10) (modify item)
```

```
> a
```

```
$good
```

```
[1] 1 2 3 10
```

```
$ok
```

```
[1] -1
```

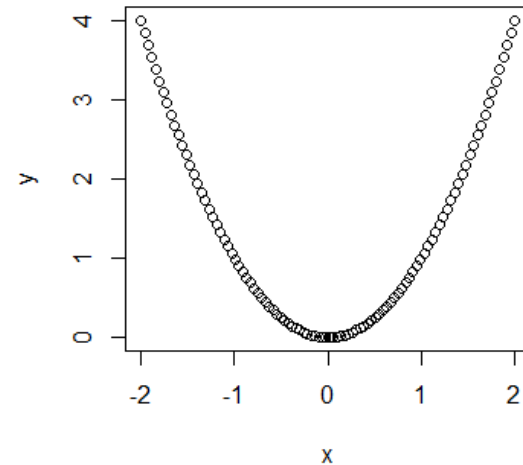

Basic plotting

Let's plot the graph of $y = x^2$!

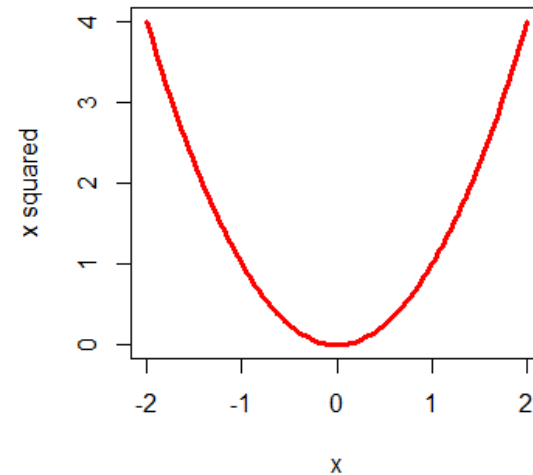
```
> x = seq(-2, 2, length = 100)
> y = x^2
> plot(x, y)
```

Many options to play with...

```
> plot(x, y, type="l", lwd = 3, col = "red",
      ylab = "x squared", main = "My plot!")
```



My plot



Basic R stuff that we skip in this brief introduction

- User-defined functions
- Non-numerical objects (character strings, boolean, ++)
- Random numbers / simulation
- Statistics (summary stats, hypothesis testing, regression analysis ++)
- ... and LOTS of other things...

Installing packages

To access the functions of an external package, you must:

- install the package
 - downloads it to your computer
 - this is done only once
 - **install.packages()** (needs internet connection - or local zip file)
- load it into R
 - every new session
 - **library()** or **require()**

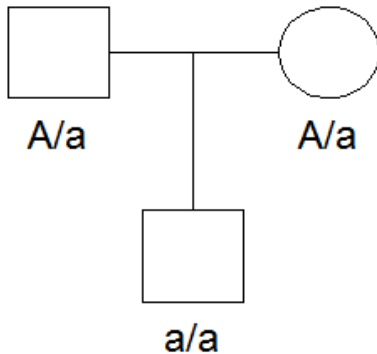
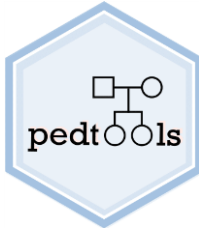
To check if a package is installed, simply try to load it:

```
> library(pedtools)
```

If you get an error, do:

```
> install.packages("pedtools")
```

pedtools: Tools for working with pedigrees in R



What it contains

pedigrees

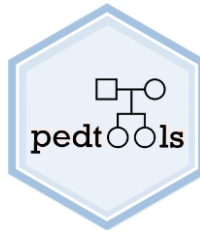
markers

What it does

- Create
- Manipulate
- Understand
- Plot

kinship2

pedtools



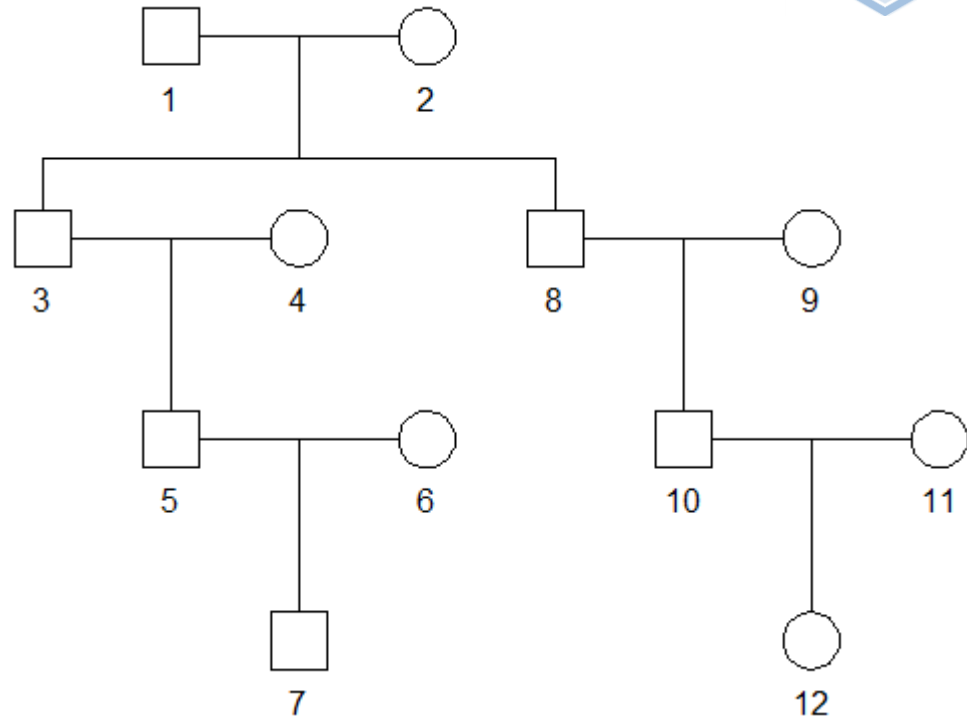
```
> library(pedtools)
> x = cousinPed(2)
> plot(x)
```

Changing genders:

```
> x2 = swapSex(x, 7)
```

Or several at once:

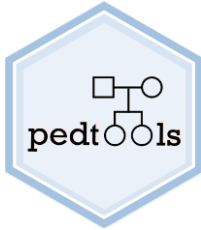
```
> x3 = swapSex(x, c(3,5,7))
> plot(x3)
```



When changing an R object, always store the result back in a variable, with:

- a different name, or
- the same name (if you don't need the old)





Create: basic

- nuclearPed
- linearPed
- cousinPed
- halfSibPed
- halfCousinPed

Create: complex

- doubleCousins
- quadHalfFirstCousins
- fullSibMating
- halfSibStack
- randomPed

Manipulate

- addChilden
- addParents
- removeIndividuals
- branch
- mergePed
- breakLoops
- reorderPed
- relabel

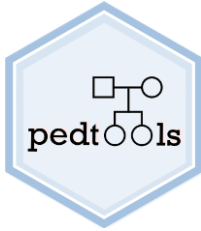
Member subsets

- founders
- nonfounders
- leaves
- males
- females
- typedMembers
- untypedMembers

Relatives

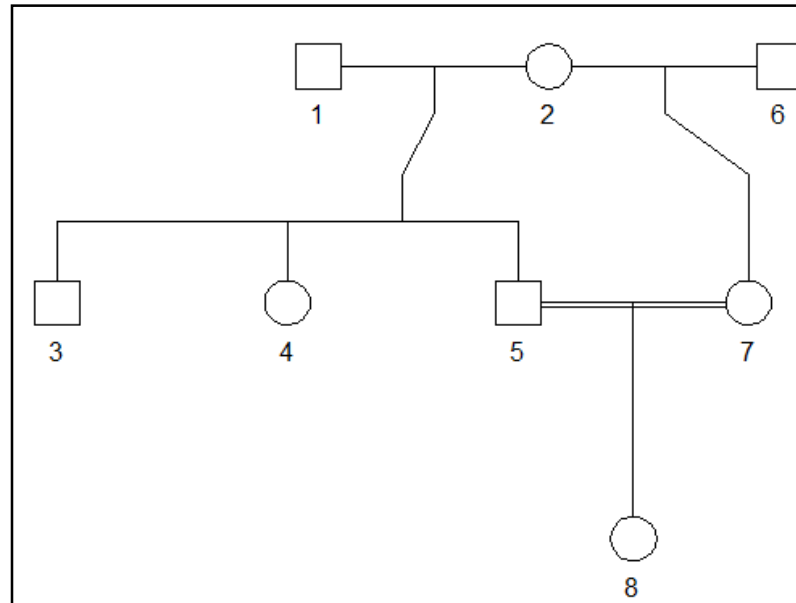
- father
- mother
- children
- siblings
- grandparents
- cousins
- spouses

- ancestors
- descendants
- unrelated



Now try this!

```
> x = nuclearPed(3, sex = c(1, 2, 1))
> plot(x)
> x = addDaughter(x, 2)
> plot(x)
> x = addChildren(x, father = 5, mother = 7, nch = 1, sex = 2)
> plot(x)
```



Tip: Plot after each step to keep track of what's going on!

Alternative creation process: Using pedigree files

- Pedigree file:
 - Text file describing the family structure
 - Extension **.ped** (by convention)
- Format: Compulsory columns (usually in this order):
 - (Family ID)
 - Individual ID
 - ID of father
 - ID of mother
 - Gender (1 = male, 2 = female, 0 = unknown)

For (old) experts: This is «pre-made ped format»

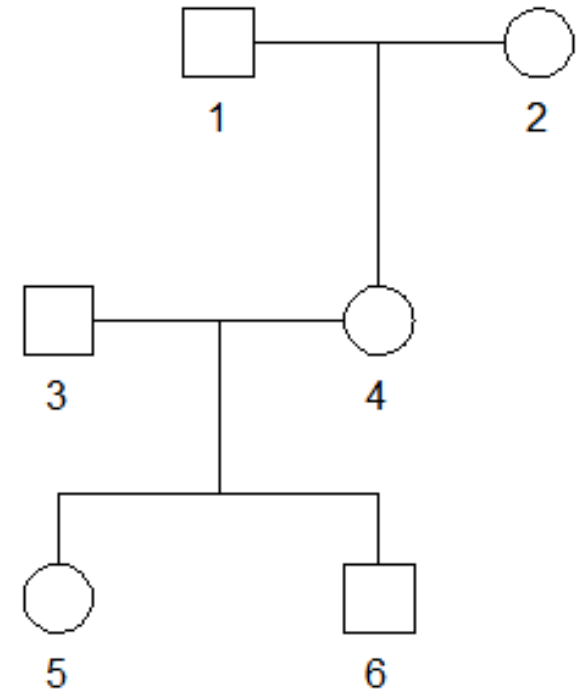
Example: pedigree file

Contents of *example.ped*:

FAMID	ID	FID	MID	SEX
1	1	0	0	1
1	2	0	0	2
1	3	0	0	1
1	4	1	2	2
1	5	3	4	2
1	6	3	4	1

0's if founder

1 = male
2 = female



In pedtools:

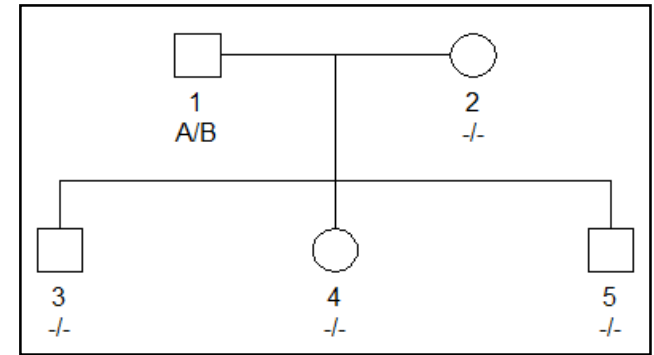
```
> x = readPed("example.ped")  
> plot(x)
```

Adding marker data

```
> x = nuclearPed(3, sex = c(1, 2, 1))
> m = marker(x, "1" = c("A", "B"))
> plot(x, m)
```

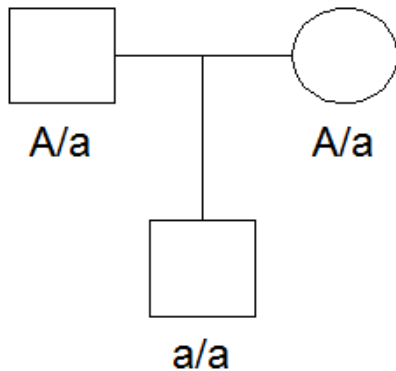
What is m??

```
> m
      [,1] [,2]
[1,]    1    2
[2,]    0    0
[3,]    0    0
[4,]    0    0
[5,]    0    0
attr(,"missing")
[1] 0
attr(,"alleles")
[1] "A" "B"
attr(,"nalleles")
[1] 2
attr(,"afreq")
[1] 0.5 0.5
attr(,"chrom")
[1] NA
```



Answer:
A matrix with *attributes*

pedprobr: Pedigree probabilities in R



What it does

Compute the probability

$$\mathcal{P}(\text{genotypes} \mid \text{pedigree}; \text{params})$$

Features

- arbitrary inbreeding
- autosomal & X-linked
- linked markers
- mutation models
- Elston-Stewart algorithm

Adding marker data

```
> x = nuclearPed(3, sex = c(1, 2, 1))  
> m = marker(x, "1" = c("A", "B"))  
> plot(x, m)
```

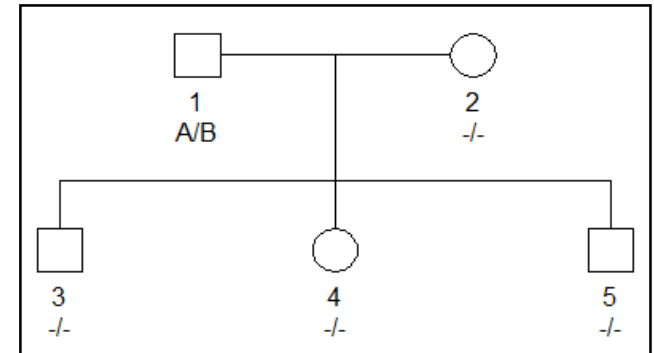
Compute the pedigree likelihood:

```
> library(pedprobr)  
> likelihood(x, m)  
[1] 0.5
```

Control:

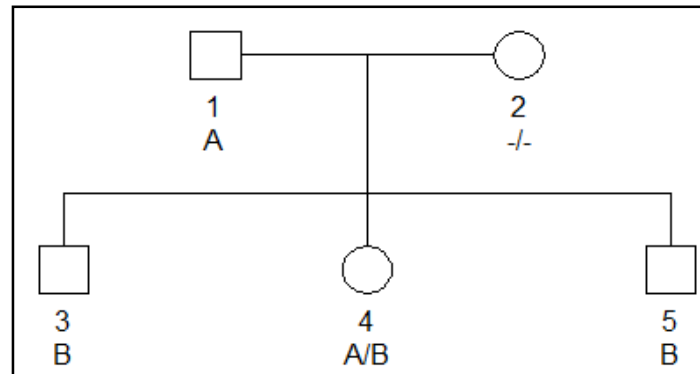
HWE implies $P(A/B) = 2pq$

```
> 2 * 0.5 * 0.5  
[1] 0.5
```



Adding X-linked marker data

```
> m2 = marker(x, alleles = c("A", "B"), chrom = "X")
> genotype(m2, 1) = "A"
> genotype(m2, 4) = c("A", "B")
> genotype(m2, c(3, 5)) = "B"
> plot(x, m2)
```

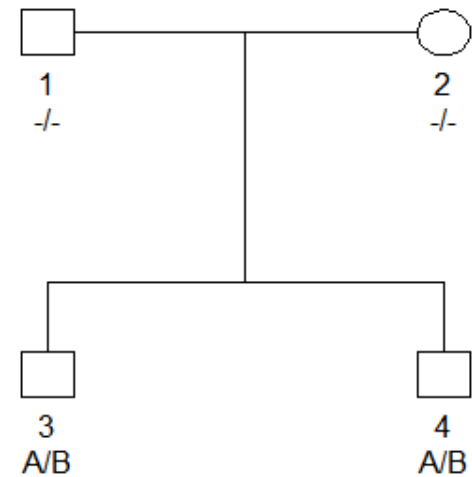


```
> likelihood(x, m2)
[1] 0.15625
```

oneMarkerDistribution

Computes the genotype distribution for one or several pedigree members, conditional on the observed genotypes

```
> x = nuclearPed(2)
> m = marker(x, afreq = c(A = 0.5, B = 0.5))
> genotype(m, id = 3:4) = c("A", "B")
> plot(x,m)
```

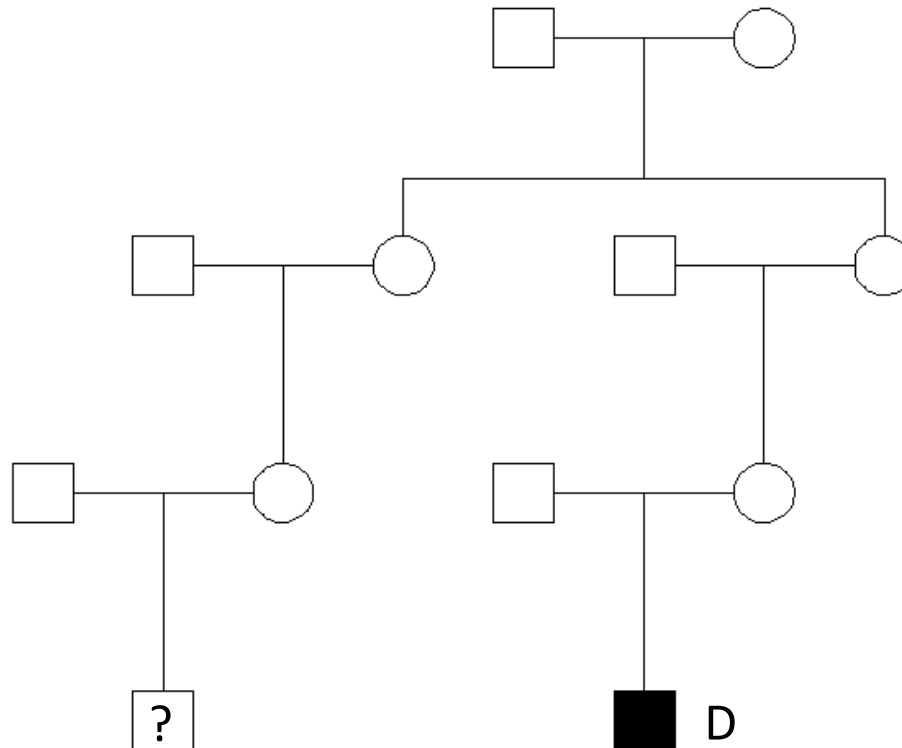
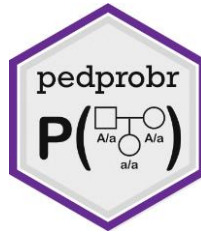


```
> oneMarkerDistribution(x, partial = m, id = 1:2)
```

Joint genotype probability distribution for individuals 1 and 2:

	A/A	B/B	A/B
A/A	0.0	0.2	0.1
B/B	0.2	0.0	0.1
A/B	0.1	0.1	0.2

Practical example with **oneMarkerDistribution**



Will my child have the disease?

Summary

- Basic R
 - arithmetic
 - variables
 - vectors
 - lists
 - plots
- Ped suite packages **pedtools** + **pedprobr**
 - create pedigrees
 - modify pedigrees
 - markers
 - likelihoods
 - genotype distributions