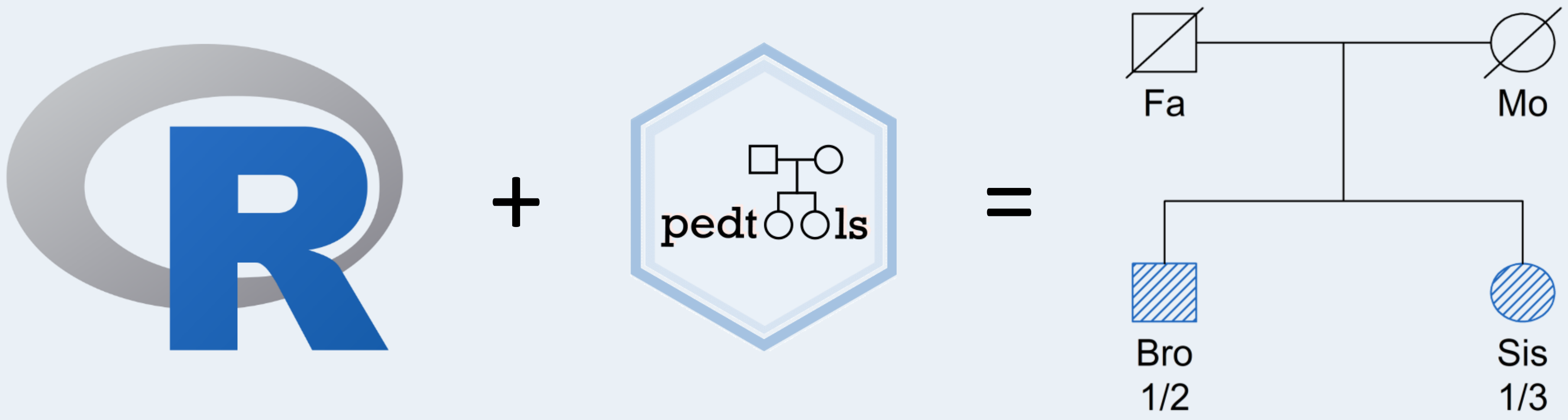


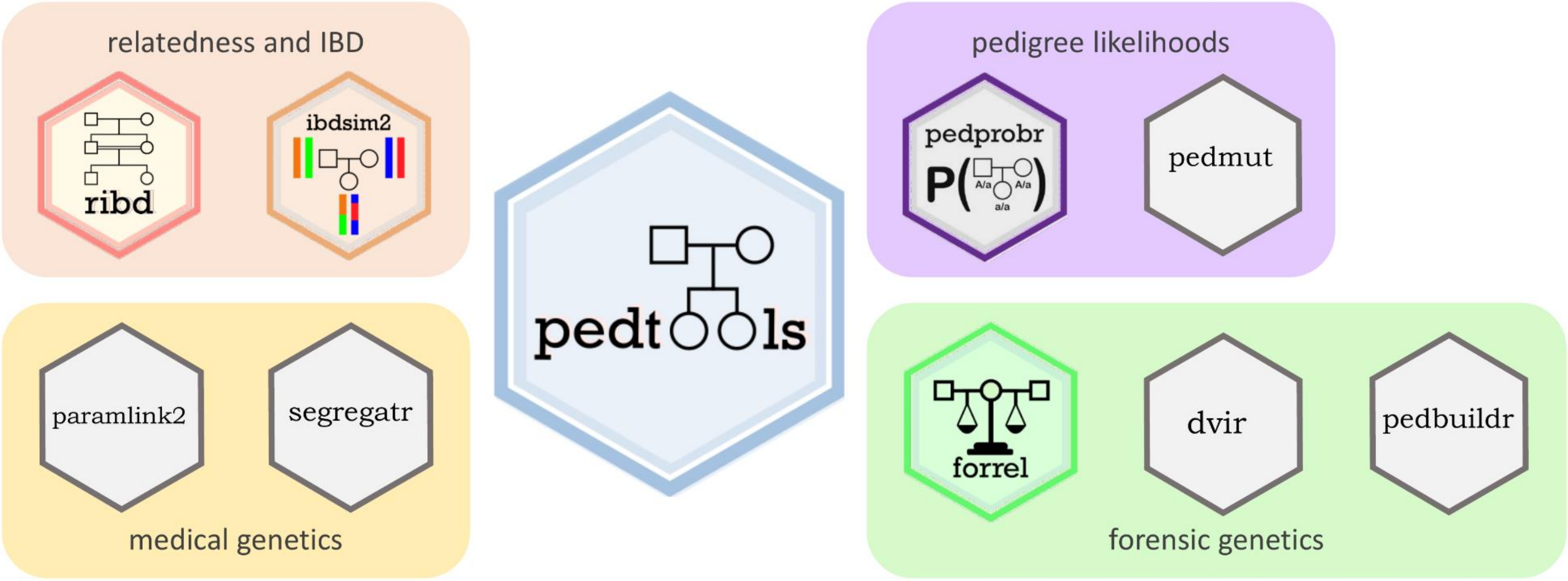
Lecture 2:
Working with pedigrees in R

Pedigree analysis in R

ISFG Summer School - Virtual Edition 2021

Magnus Dehli Vigeland & Thore Egeland

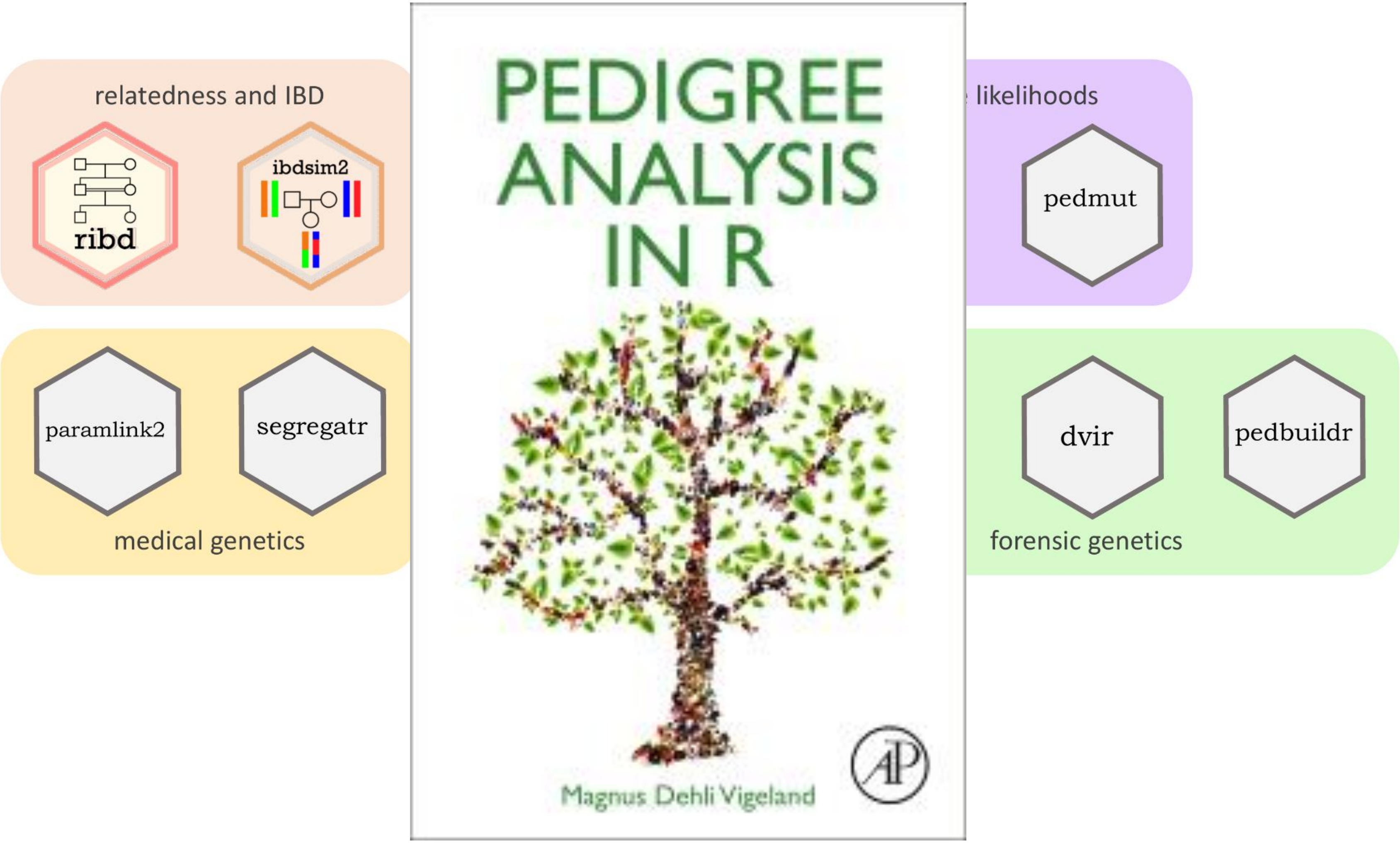
The ped suite: A collection of packages for pedigree analysis in R



Home page:
https://magnusdv.github.io/pedsuite

Source code available on GitHub:
https://github.com/magnusdv

The ped suite: A collection of packages for pedigree analysis in R



relatedness and IBD

ribd

ibdsim2

likelihoods

pedmut

paramlink2

segregatr

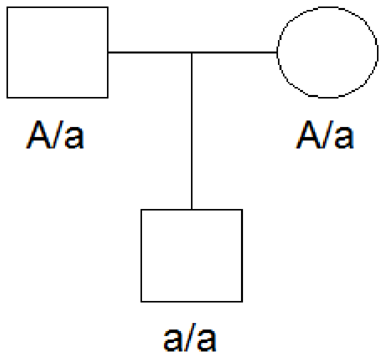
medical genetics

dvir

pedbuildr

forensic genetics

pedtools: Tools for working with pedigrees in R



What it contains

pedigrees

markers

What it does

- Create
- Manipulate
- Investigate
- Plot

kinship2

How do I load my pedigree into R?

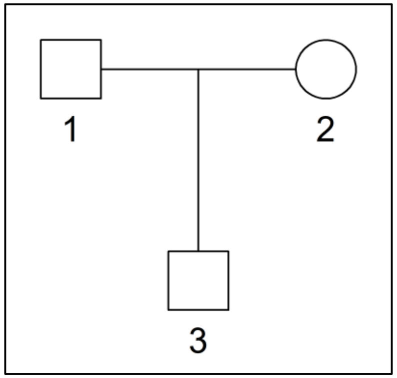
Either:

- ✓ Build with **pedtools**
- ✓ Read from file:
  - `readPed()`
  - `readFam()`

Building pedigrees

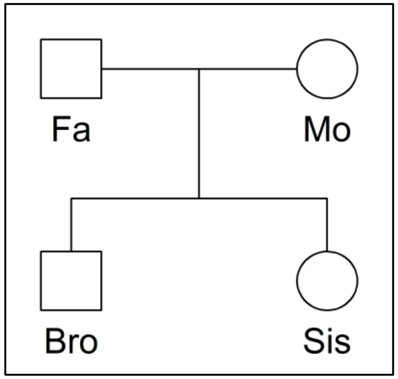
```
>   library(pedtools)

>   x = nuclearPed()
>   plot(x)
```



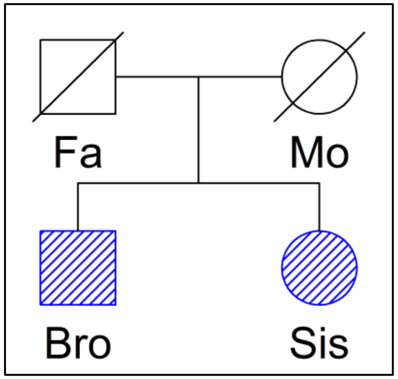
Names and sex:

```
>   y = nuclearPed(father = "Fa",
                   mother = "Mo",
                   child = c("Bro", "Sis"),
                   sex = 1:2)
>   plot(y)
```



Many ways to tweak the plot!

```
>   plot(y,
         deceased = c("Fa", "Mo"),
         hatched = c("Bro", "Sis"),
         col = list(blue = c("Bro", "Sis")),
         cex = 1.5)
```



Some useful functions

Create: basic

- singleton
- nuclearPed
- linearPed
- halfSibPed
- cousinPed
- halfCousinPed

Member subsets

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

Create: complex

- ancestralPed
- doubleCousins
- quadHalfFirstCousins
- fullSibMating
- randomPed

Relatives

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

- ancestors
- descendants
- unrelated

Manipulate

- addSon
- addDaugher
- addParents
- addChildren

- swapSex
- relabel
- removeIndividuals

- branch
- subset

- mergePed
- breakLoops

Another example

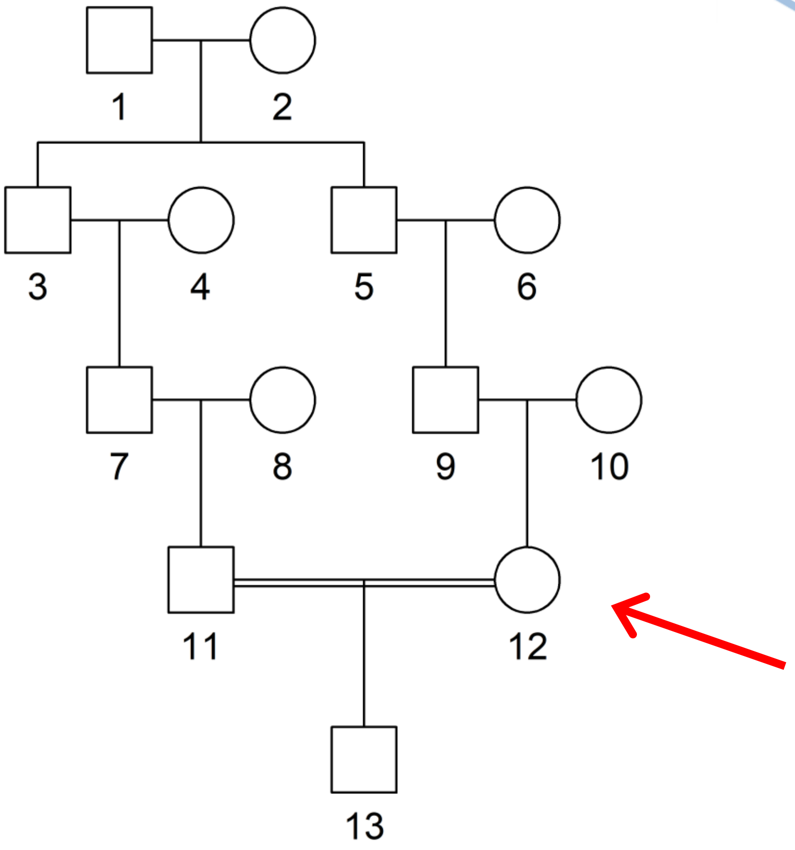
```
>   x = cousinPed(2)
>   plot(x)
```

Change gender:

```
>   x = swapSex(x, 12)
>   plot(x)
```

Add inbred child

```
>   x = addChildren(x, nch = 1,
      father = 11, mother = 12)
>   plot(x)
```



Remember

- Store the result after each change!
- It is OK to use the same name (if you don't need the previous object)

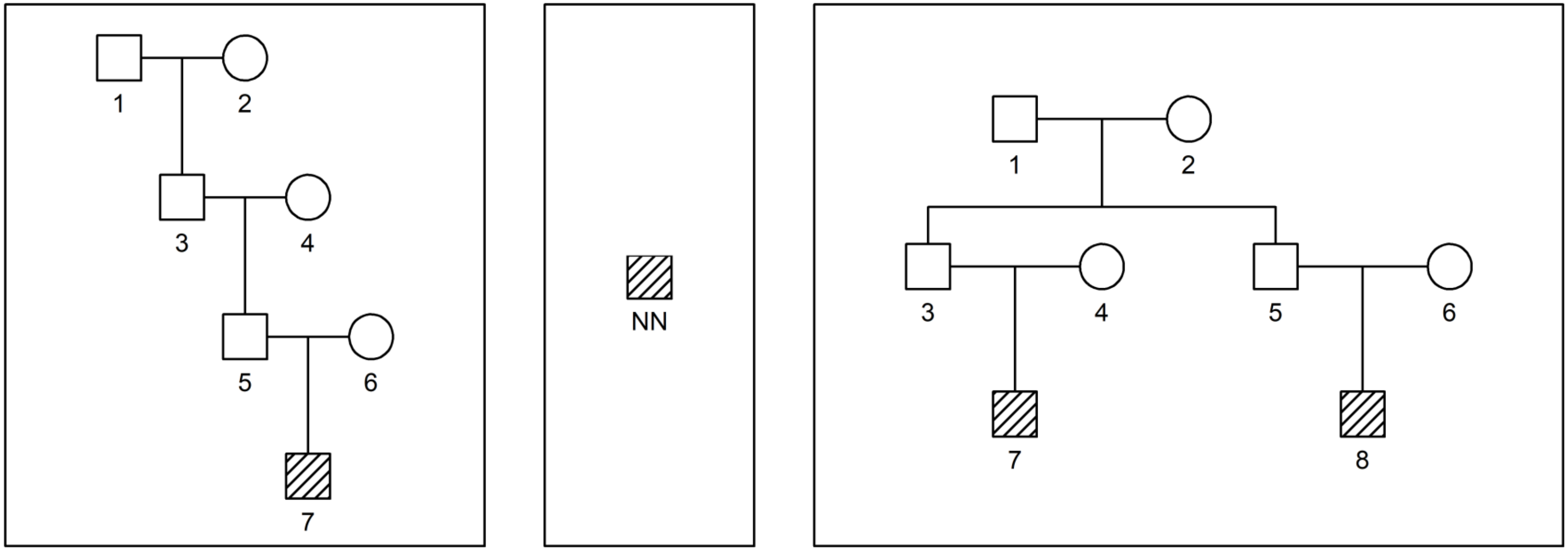
Shortcut command for this pedigree

```
> x = cousinPed(2, child = TRUE)
```

List of pedigrees

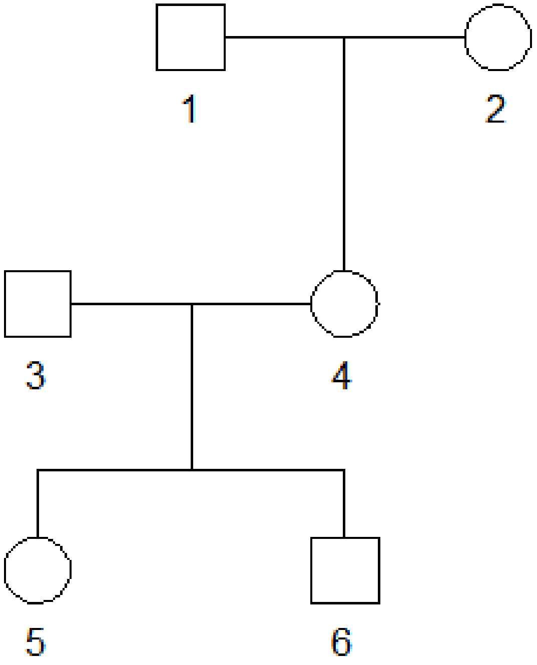
```
>   peds = list(linearPed(3),
                singleton("NN"),
                cousinPed(1))
```

```
>   plotPedList(peds,
                widths = c(2, 1, 3),
                hatched = leaves)
```



Alternative pedigree creation: ped file

A text file describing a pedigree structure.



0 if founder

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

Contents of *example.ped*

In pedtools:

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

Columns

famid = family ID (optional)

id = individual ID

fid = ID of father

mid = ID of mother

sex = 1 (male), 2 (female) or 0 (unknown)

Oh my! Do I have to write these ped-files by hand?

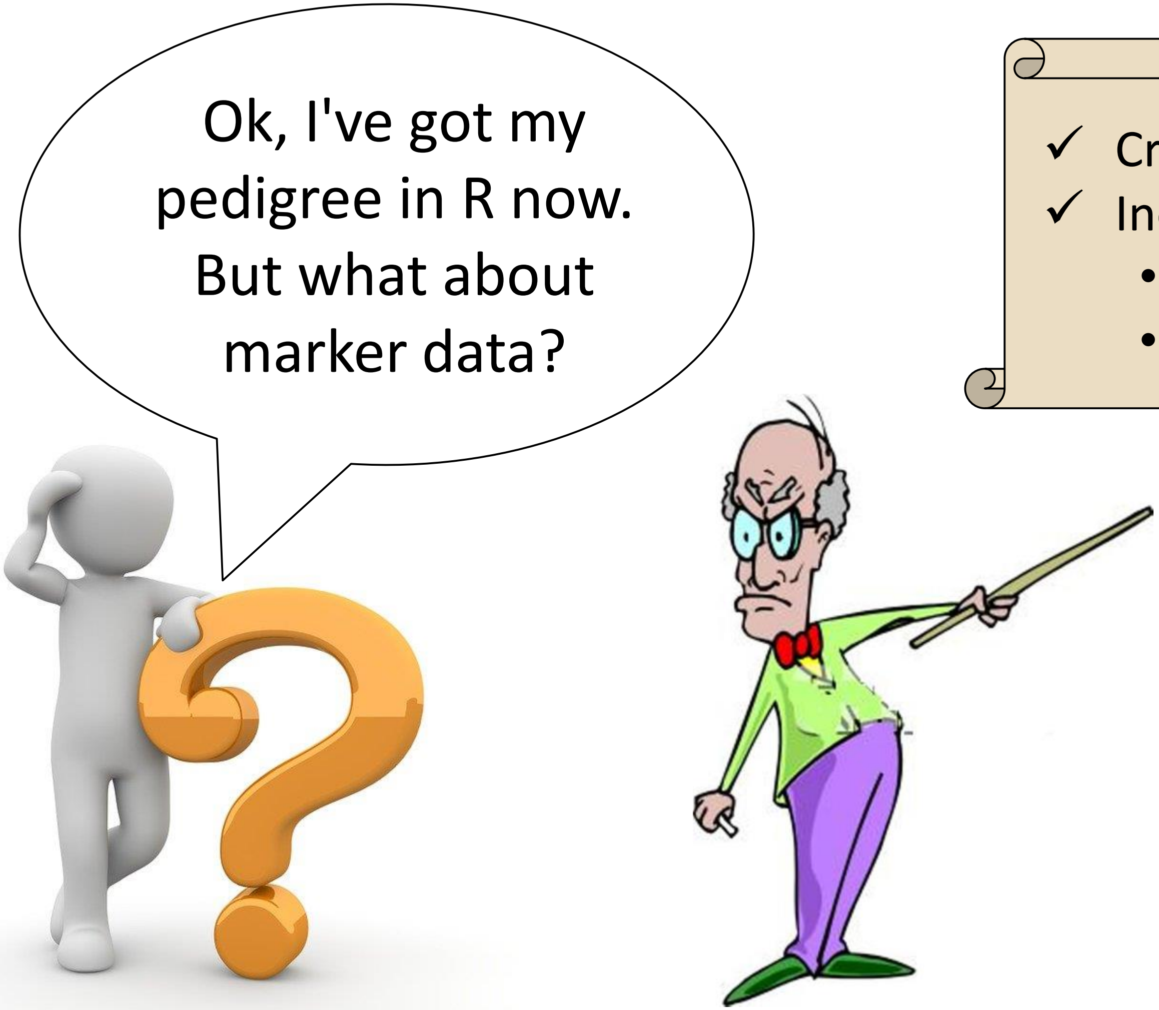
No, that is tedious and error-prone!

Free online tool: **QuickPed**[1]

[1] *https://magnusdv.shinyapps.io/quickped*

Quick demo of **QuickPed**

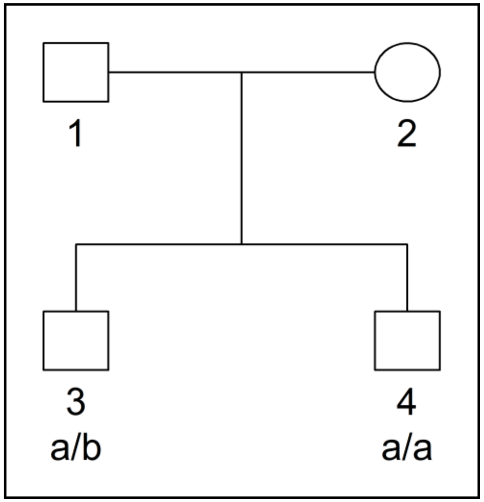
https://magnusdv.shinyapps.io/quickped


Ok, I've got my pedigree in R now. But what about marker data?

- ✓ Create with **pedtools**
- ✓ Include in
  - ped file
  - fam file

Marker data 1

```
>   x = nuclearPed(2)
>   m = marker(x, geno = c(NA, NA, "a/b", "a/a"))
>   plot(x, marker = m)
```

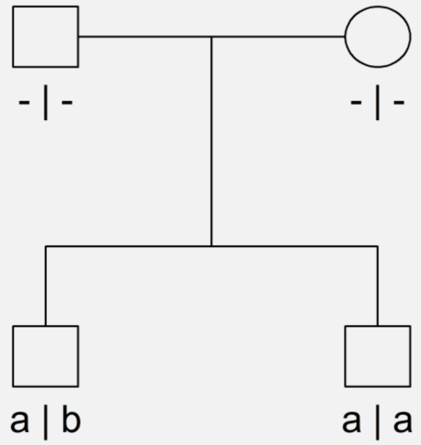


Print information about m:

```
>   m
 id  <NA>
  1   -/-
  2   -/-
  3   a/b
  4   a/a
* * * * *
Position: NA
Mutation: none
Frequencies:
   a   b
 0.5 0.5
```

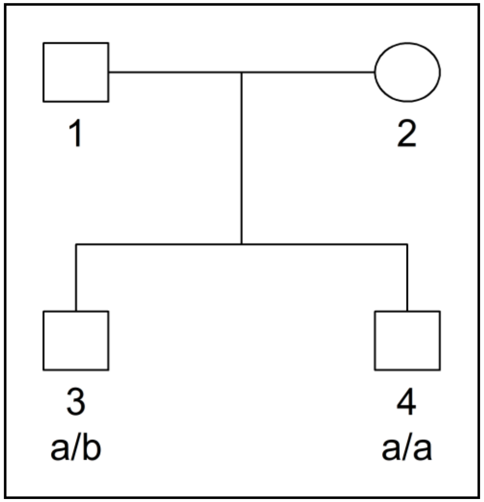
Plot options for markers:

```
>  plot(x, marker = m,
          labs = NULL,
          sep = " | ",
          showEmpty = TRUE)
```



Marker data 2

```
>   x = nuclearPed(2)
>   m = marker(x, geno = c(NA, NA, "a/b", "a/a"))
>   plot(x, marker = m)
```



It is often useful to **attach** m to x

```
>   x = setMarkers(x, m)
>   x
id fid mid sex <1>
  1   *   *   1 -/-
  2   *   *   2 -/-
  3   1   2   1 a/b
  4   1   2   1 a/a
```

new column!

Now we may write:

```
>   plot(x, marker = 1)
```

Changing allele frequencies

```
>   fr = c(a = 0.1, b = 0.9)
>   afreq(x, marker = 1) = fr
```

Inspect the result

```
>   afreq(x, marker = 1)
  a   b
0.1 0.9
```

Functions for manipulating marker data

Get/set attributes

- alleles
- afreq
- mutmod
- genotype
- chrom
- posMb
- name

Attach/remove

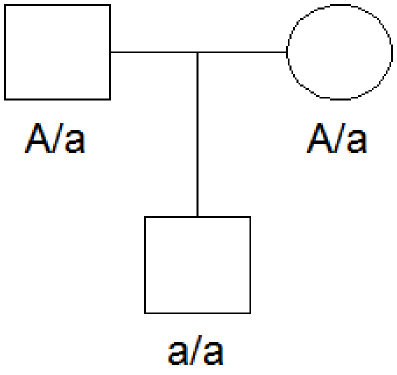
- setMarkers
- addMarkers
- selectMarkers
- removeMarkers

- transferMarkers

Frequency database

- getFreqDatabase
- setFreqDatabase
- readFreqDatabase
- writeFreqDatase

pedprobr: Pedigree probabilities in R



What it does

Compute the probability

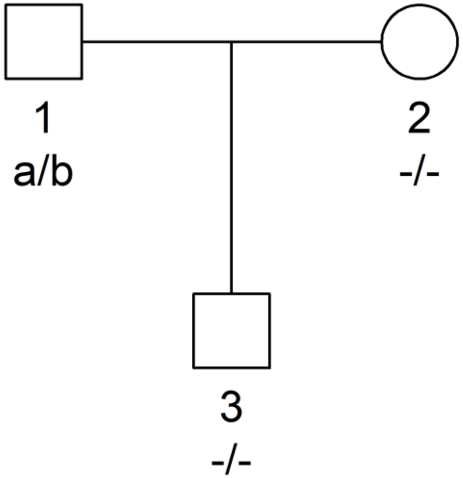
$$\mathcal{P}(genotypes \mid pedigree;\ params)$$

Features

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

A simple likelihood

```
>   library(pedprobr) # or library(pedsuite)
```



Create a pedigree with a SNP marker

```
>   x = nuclearPed(1)
>   m = marker(x, geno = c("a/b", NA, NA))
>   plot(x, m)
```

Compute the pedigree likelihood

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

Control

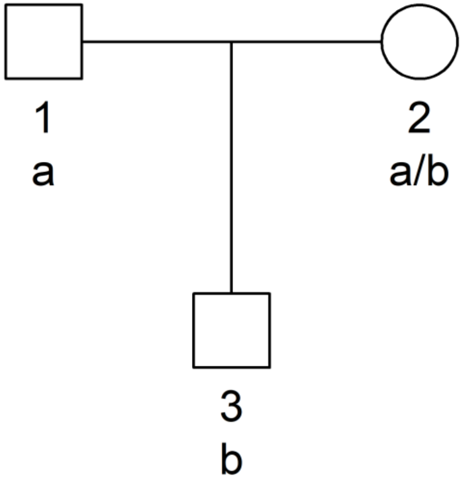
- By default, $P(a) = P(b) = 0.5$
- Thus HWE implies

$$P(a/b) = 2pq = 2 \cdot 0.5 \cdot 0.5$$
$$= 0.5$$

An example on X

Add X-marker to the pedigree

```
>   mx = marker(x, geno = c("a", "a/b", "b"),
                chrom = "X")
>   x = setMarkers(x, mx)
>   plot(x, marker = 1)
```



Compute the pedigree likelihood

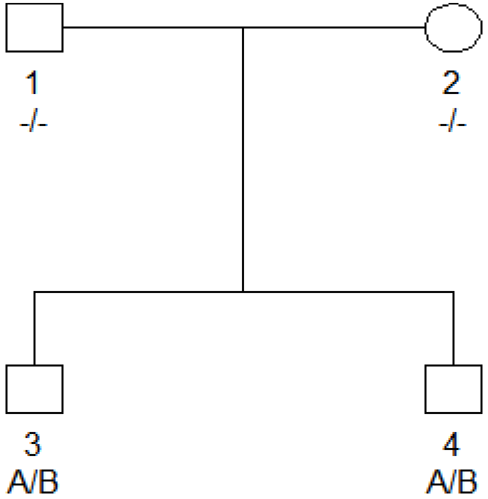
```
>   likelihood(x)
[1] 0.125
```

By default, `likelihood(x)` returns the likelihood for each attached markers.

oneMarkerDistribution

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

```
>   x = nuclearPed(2)
>   m = marker(x, geno = c(NA, NA, "A/B", "A/B"))
>   plot(x,m)
```



```
>   oneMarkerDistribution(x, partial = m, ids = 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
```

Now: Exercises!