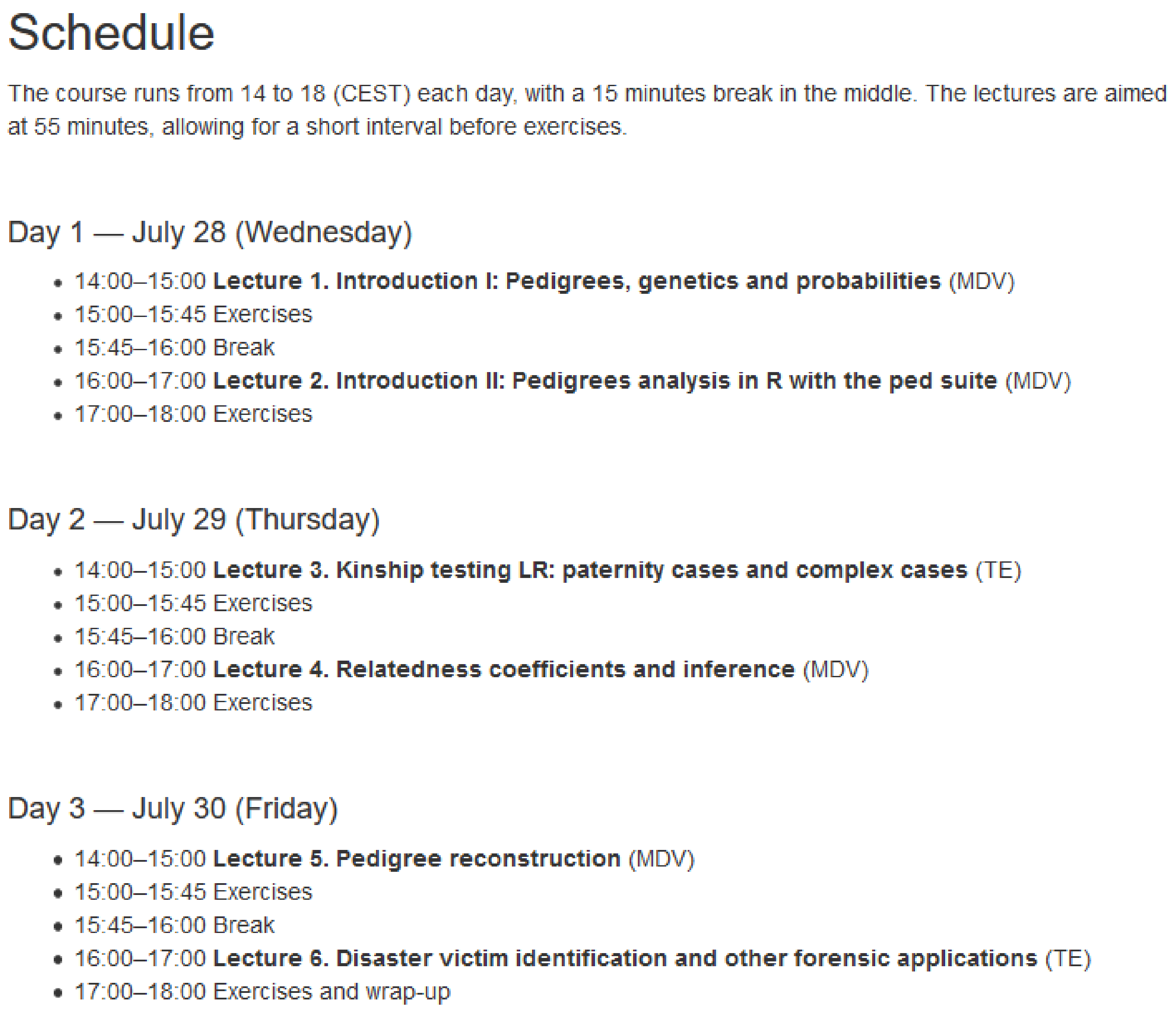
ISFG Summer School - Virtual Edition 2021

Pedigree analysis in R

Teachers:

Thore Egeland

Magnus Dehli Vigeland



Schedule

The course runs from 14 to 18 (CEST) each day, with a 15 minutes break in the middle. The lectures are aimed at 55 minutes, allowing for a short interval before exercises.

Day 1 — July 28 (Wednesday)

- 14:00–15:00 **Lecture 1. Introduction I: Pedigrees, genetics and probabilities** (MDV)
- 15:00–15:45 Exercises
- 15:45–16:00 Break
- 16:00–17:00 **Lecture 2. Introduction II: Pedigrees analysis in R with the ped suite** (MDV)
- 17:00–18:00 Exercises

Day 2 — July 29 (Thursday)

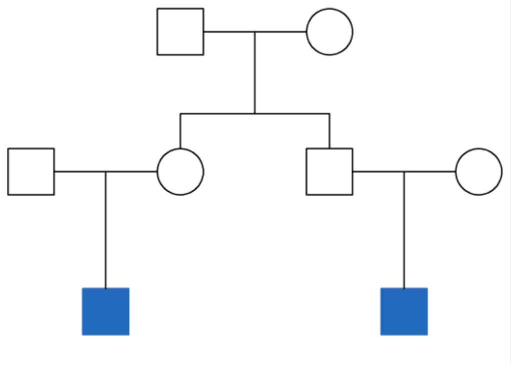
- 14:00–15:00 **Lecture 3. Kinship testing LR: paternity cases and complex cases** (TE)
- 15:00–15:45 Exercises
- 15:45–16:00 Break
- 16:00–17:00 **Lecture 4. Relatedness coefficients and inference** (MDV)
- 17:00–18:00 Exercises

Day 3 — July 30 (Friday)

- 14:00–15:00 **Lecture 5. Pedigree reconstruction** (MDV)
- 15:00–15:45 Exercises
- 15:45–16:00 Break
- 16:00–17:00 **Lecture 6. Disaster victim identification and other forensic applications** (TE)
- 17:00–18:00 Exercises and wrap-up

Home page

https://magnusdv.github.io/pedsuite/articles/web_only/course-isfg2021.html



Lecture 1: Introductions

Pedigree analysis in R

ISFG Summer School - Virtual Edition 2021

Magnus Dehli Vigeland

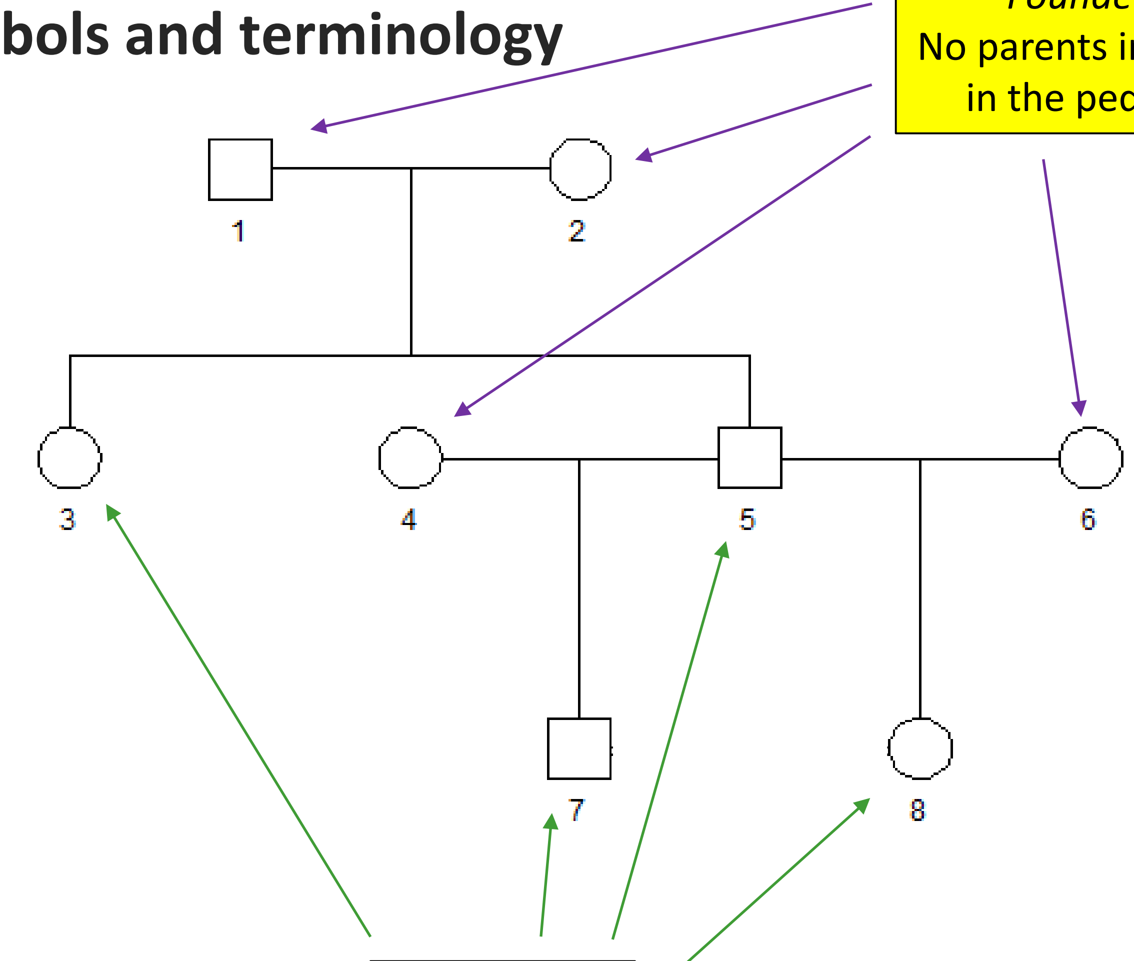
Outline

- Part I: *Pedigrees*
  - Pedigree symbols and terminology
  - Some common relationships
- Part II: *Genetics*
  - Terminology (Locus, allele, genotype, marker, ...)
  - Mendelian inheritance
  - Genetic probabilities
- Part III: R
  - What, why, how?
  - A session of basic R

Part I: Pedigrees

Pedigrees: Symbols and terminology

Founders:
No parents included
in the pedigree

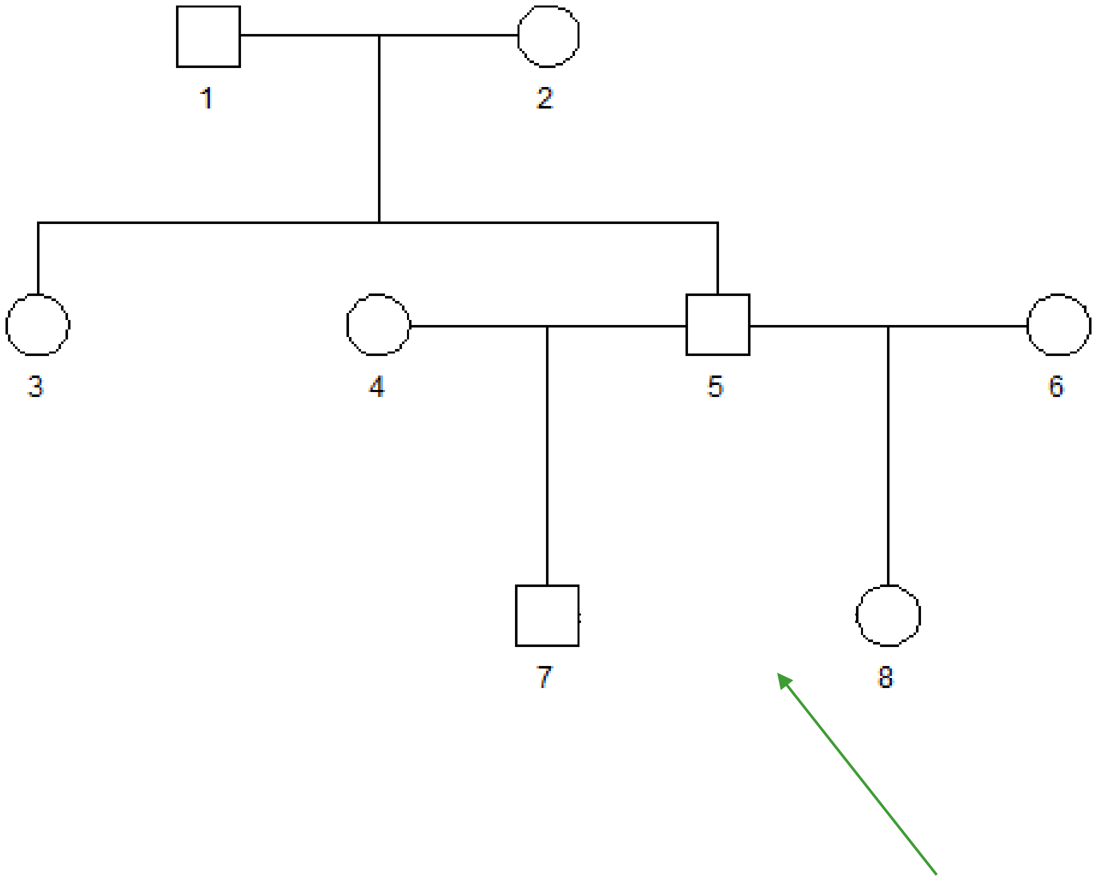


Nonfounders

□ = male

○ = female

Pedigrees: Symbols and terminology

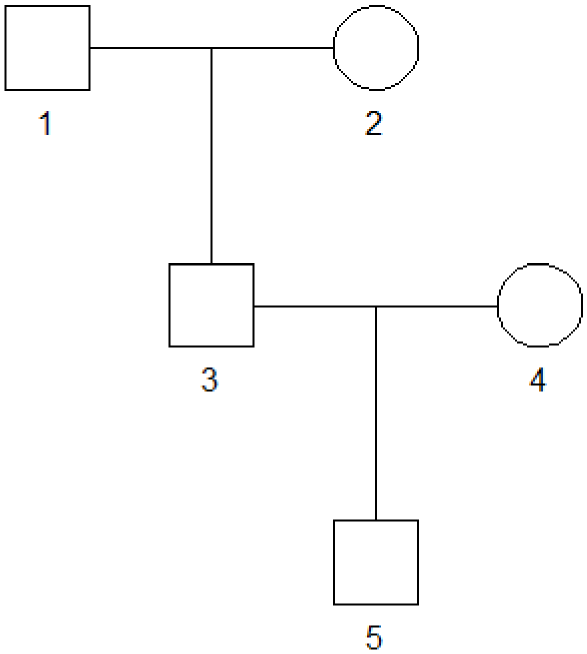


Consanguineous marriage

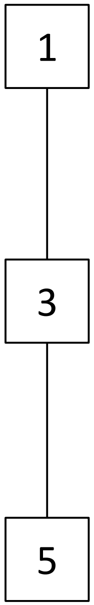
□ = male

○ = female

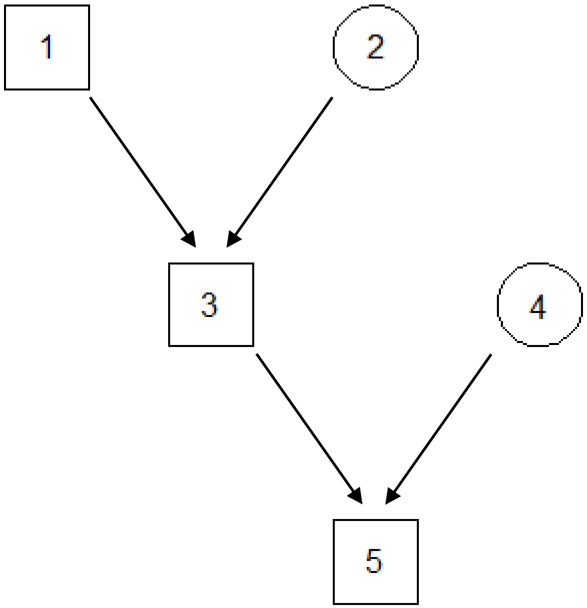
Alternative ways of drawing pedigrees



Standard



Simplified

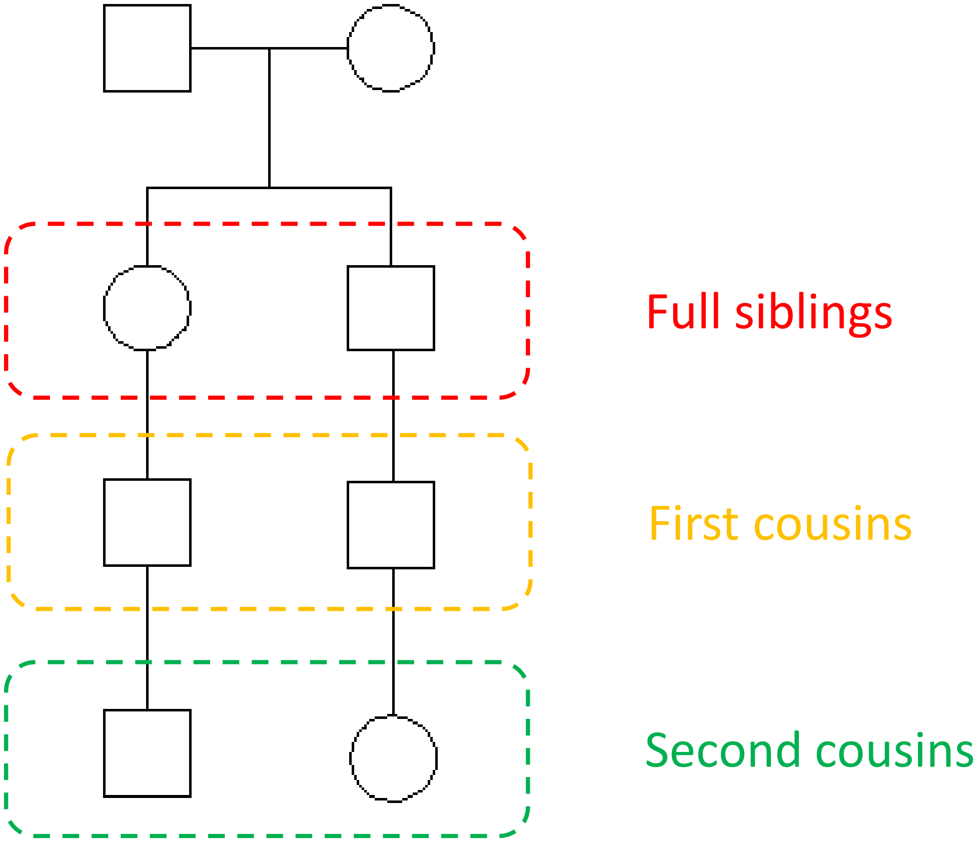


Directed acyclic graph

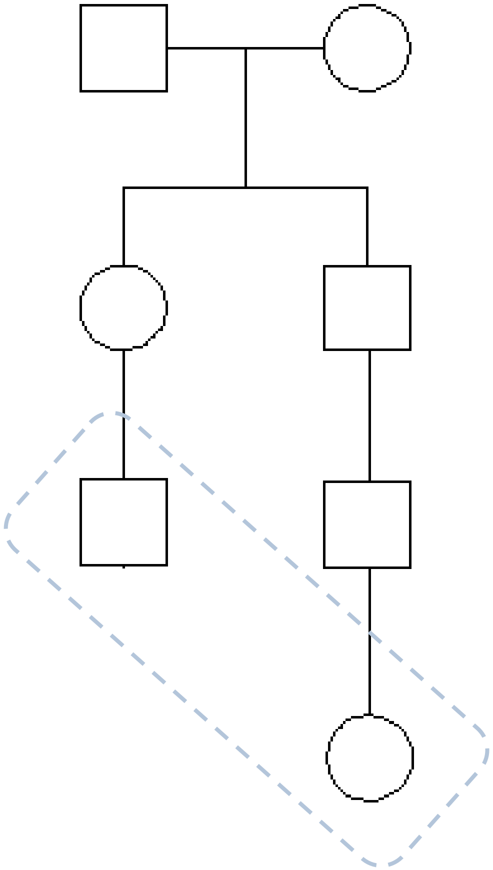
Some common relationships

(and some less common...)

Cousin relationships

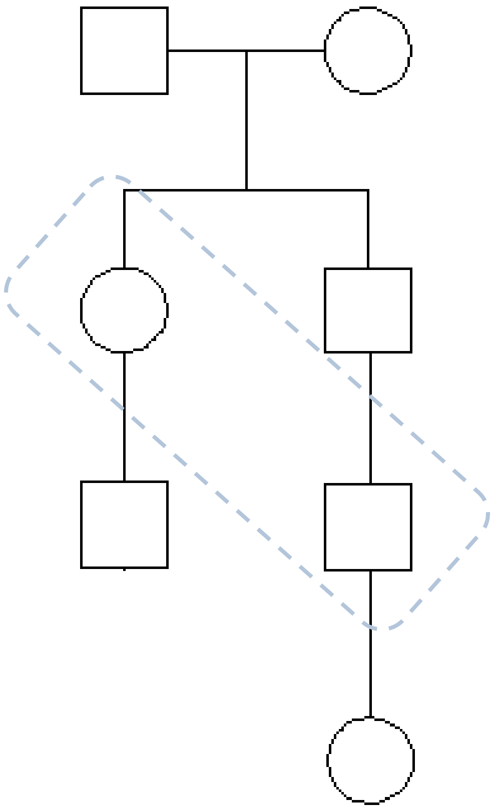


Cousin relationships



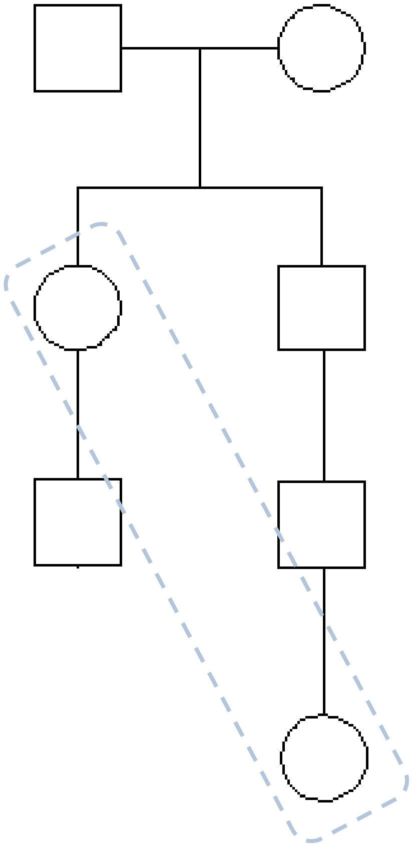
First cousins
once removed

Cousin relationships



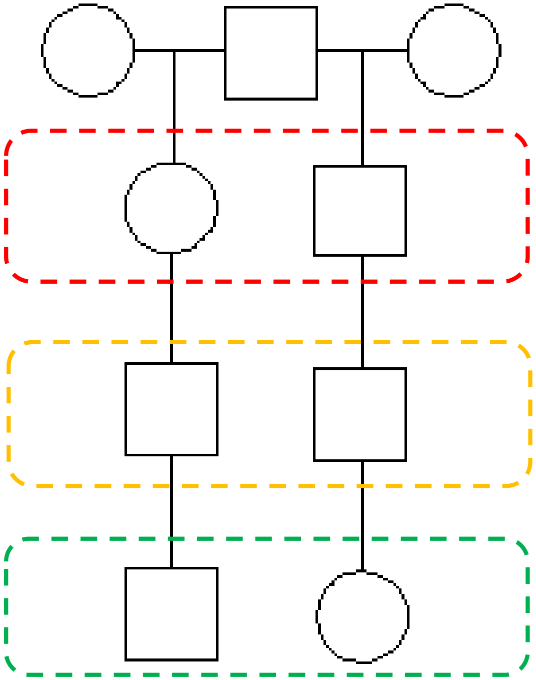
Aunt - nephew

Cousin relationships



Grandaunt

Half cousin relationships

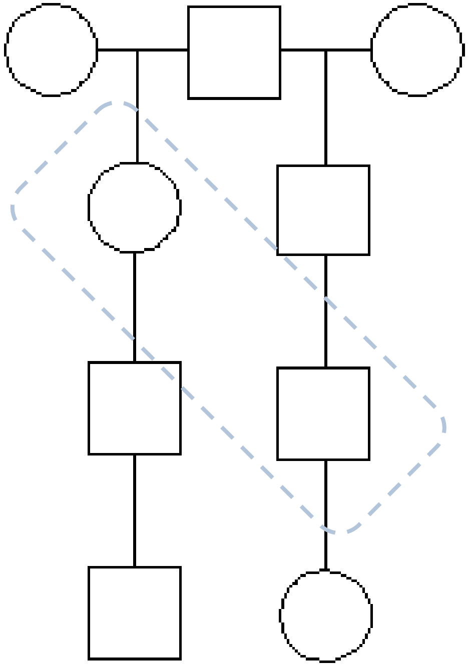


Half siblings (paternal)

Half first cousins

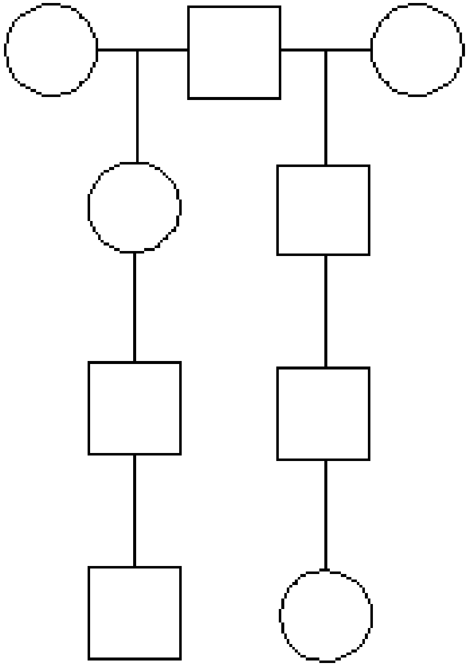
Half second cousins

Half cousin relationships

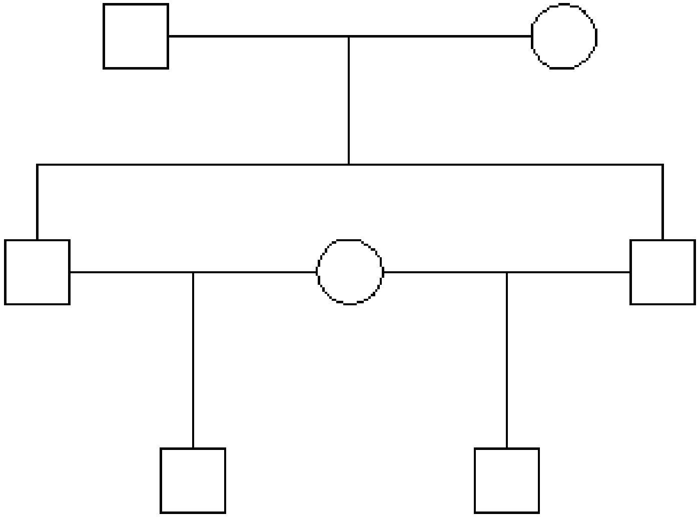


Half aunt /
half nephew

Half cousin relationships

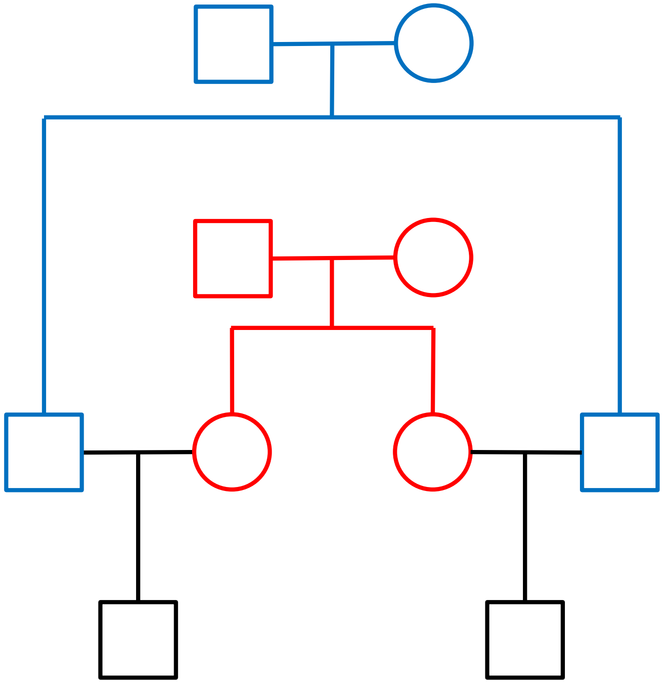


More complicated relationships



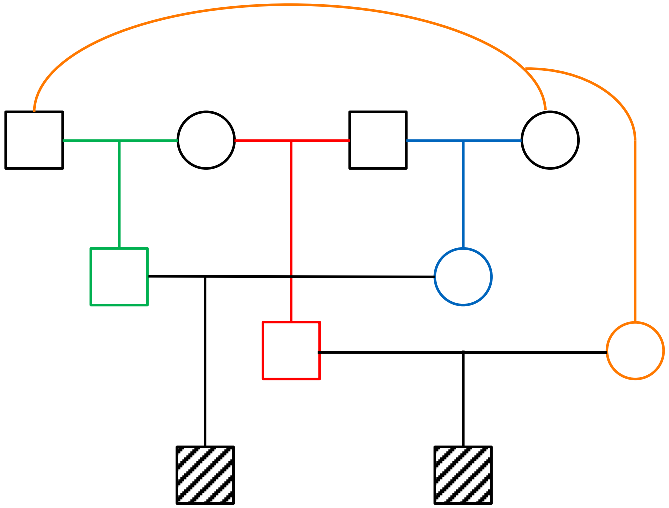
3/4 siblings

What about this?



Double first cousins

The connoisseur's favourite



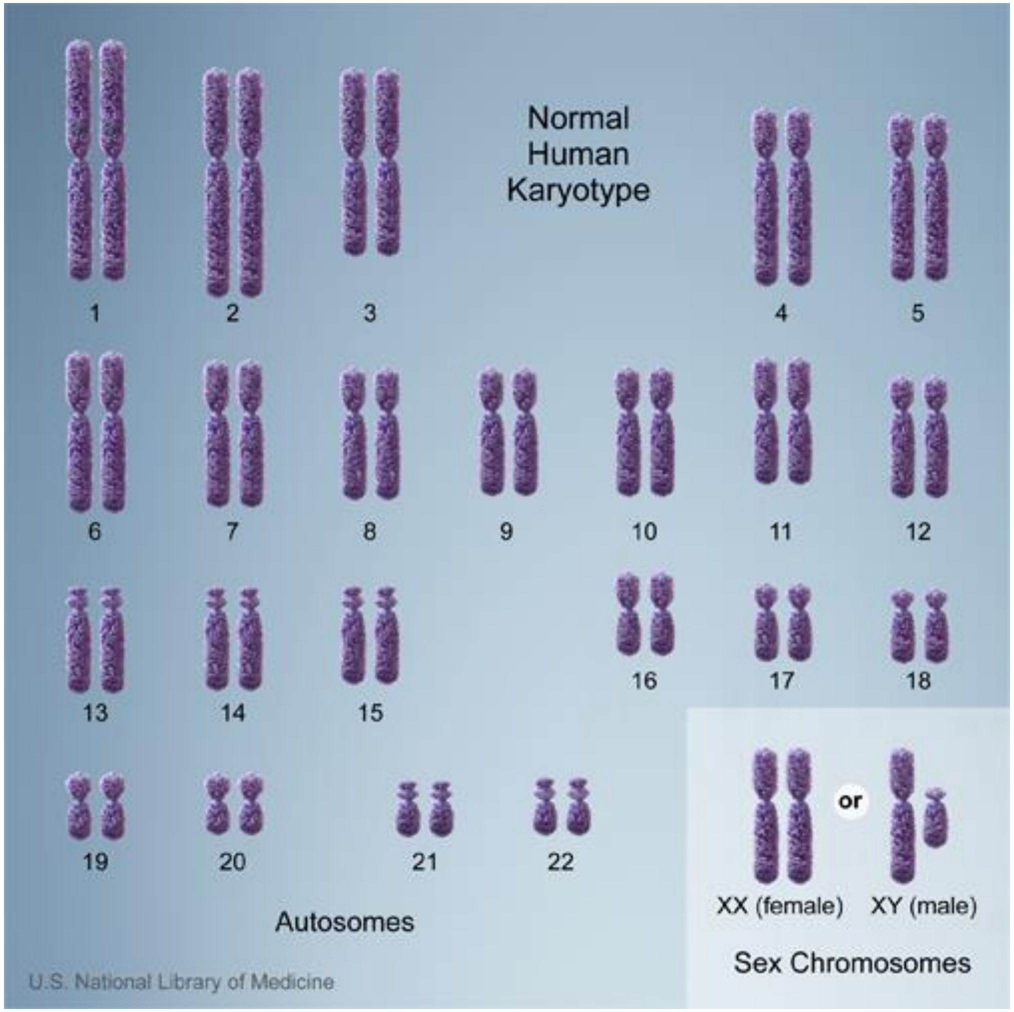
Quadruple half first cousins!

Part II: Genetics

Human Genetics

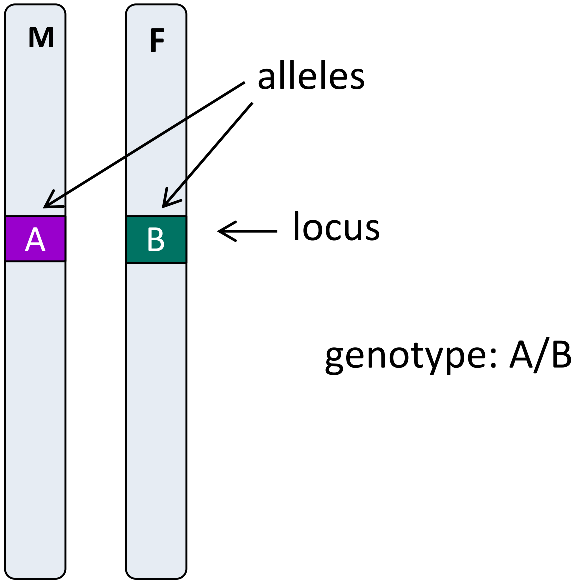
Some important terms

- Locus
- Allele
- Genotype
- Genetic markers
  - SNPs
  - microsatellites



U.S. National Library of Medicine

Locus, allele, genotype



Homologous chromosomes

- **LOCUS** = a specific place in the genome
- **ALLELE** = any of the alternative forms of a locus
- **GENOTYPE** = the set (usually: pair) of alleles carried at a given locus

Genetic markers

- Small parts of the genome which ...
  - have known position
  - vary in the population
  - are easy to genotype

- SNPs (single nucleotide polymorphisms)
  - two alleles
  - usual requirement: MAF > 1% = minor allele frequency
  - very common in the genome (millions!)
  - used in medical genetics +++

...CCGTTATATGGGC...

...CCGTTAGATGGGC...

...CCGTTATATGGGC...

...CCGTTATATGGGC...

...CCGTTAGATGGGC...

- STRs (short tandem repeats) = microsatellites
  - consecutive repeats of 2-5 bases
  - multiallelic: 5 - 50 alleles
  - allele names: # repeats
  - used in forensics

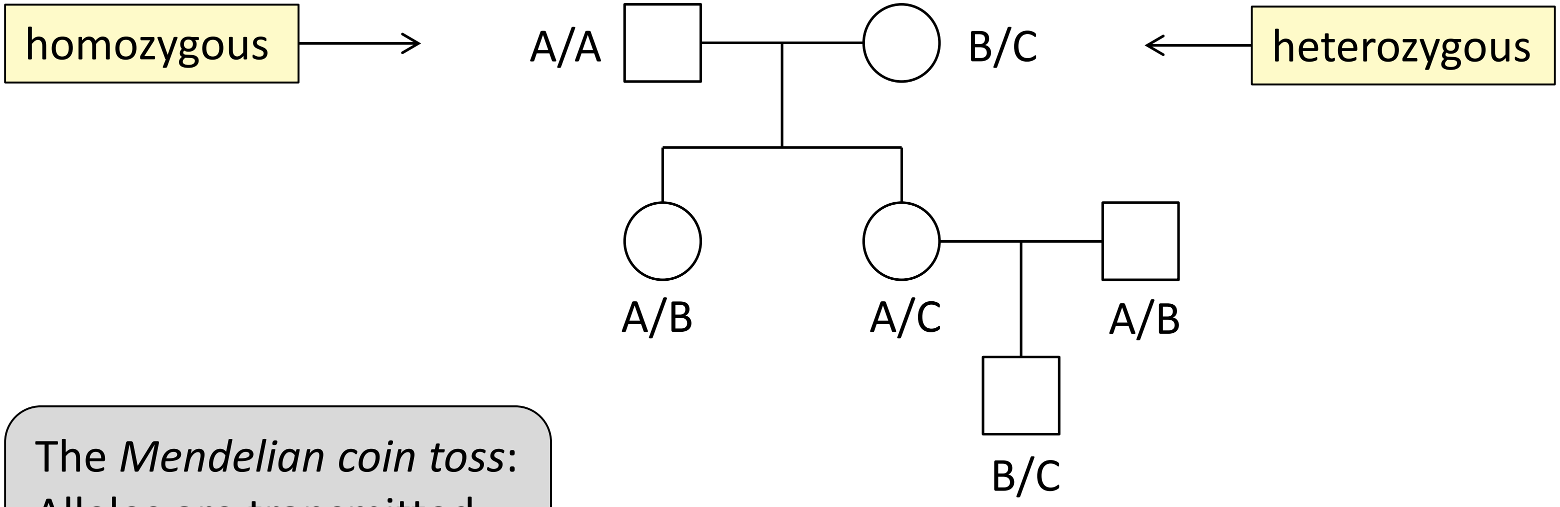
...ACG TTAG TTAG TTAG TTAG AAC..

...ACG TTAG TTAG AAC..

...ACG TTAG TTAG TTAG TTAG TTAG AAC..

Mendelian inheritance: Autosomal (chromosomes 1-22)

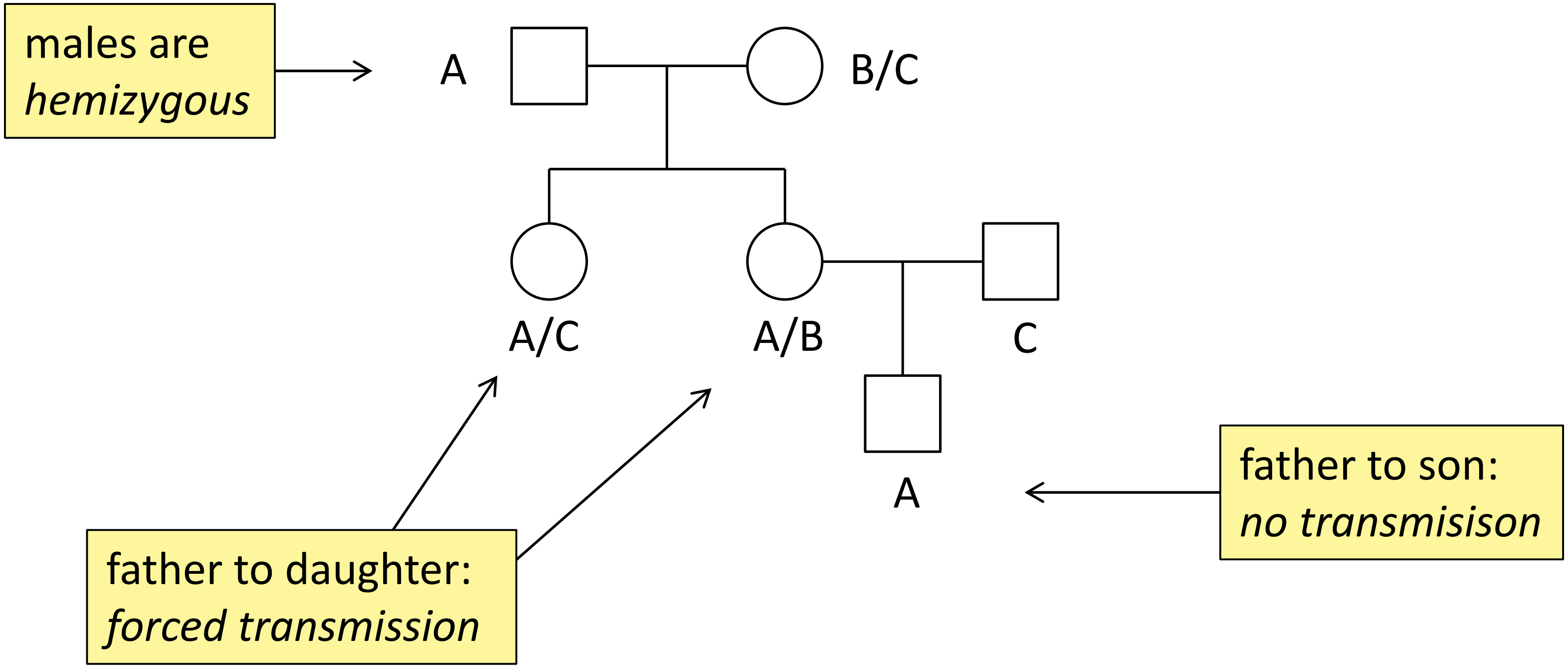
Example: autosomal marker with 3 alleles: A, B, C



The *Mendelian coin toss*:
Alleles are transmitted
with **50% chance**.

Mendelian inheritance: X-linked

Example: X-linked marker with 3 alleles: A, B, C



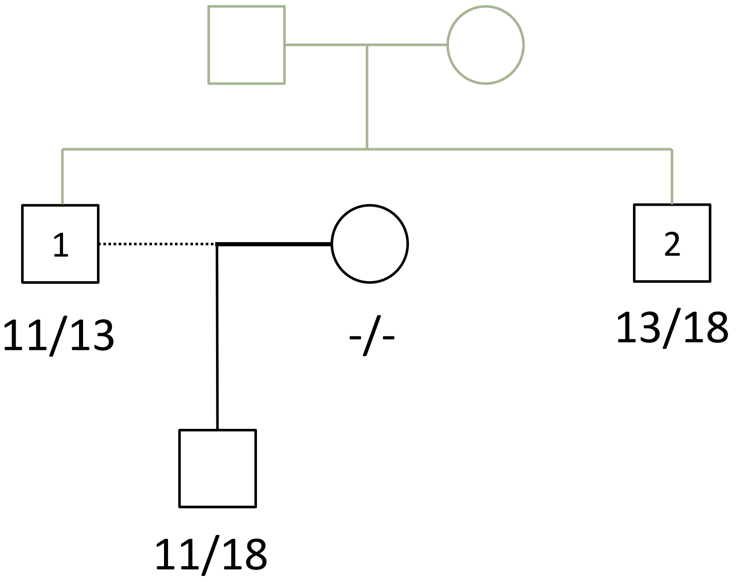
males are *hemizygous*

father to daughter: *forced transmission*

father to son: *no transmisison*

Questions related to pedigrees with genotypes

Questions related to pedigrees with genotypes

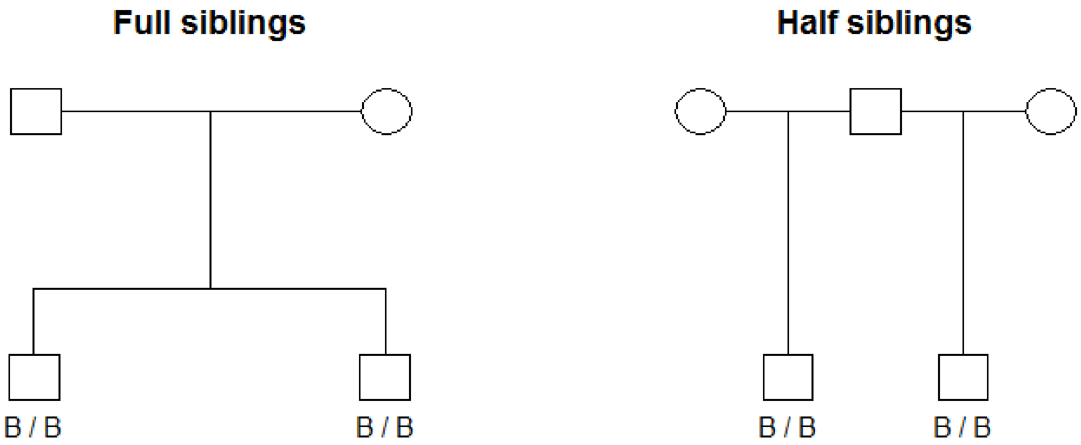


Suppose:

- 11 is common
- 18 is rare

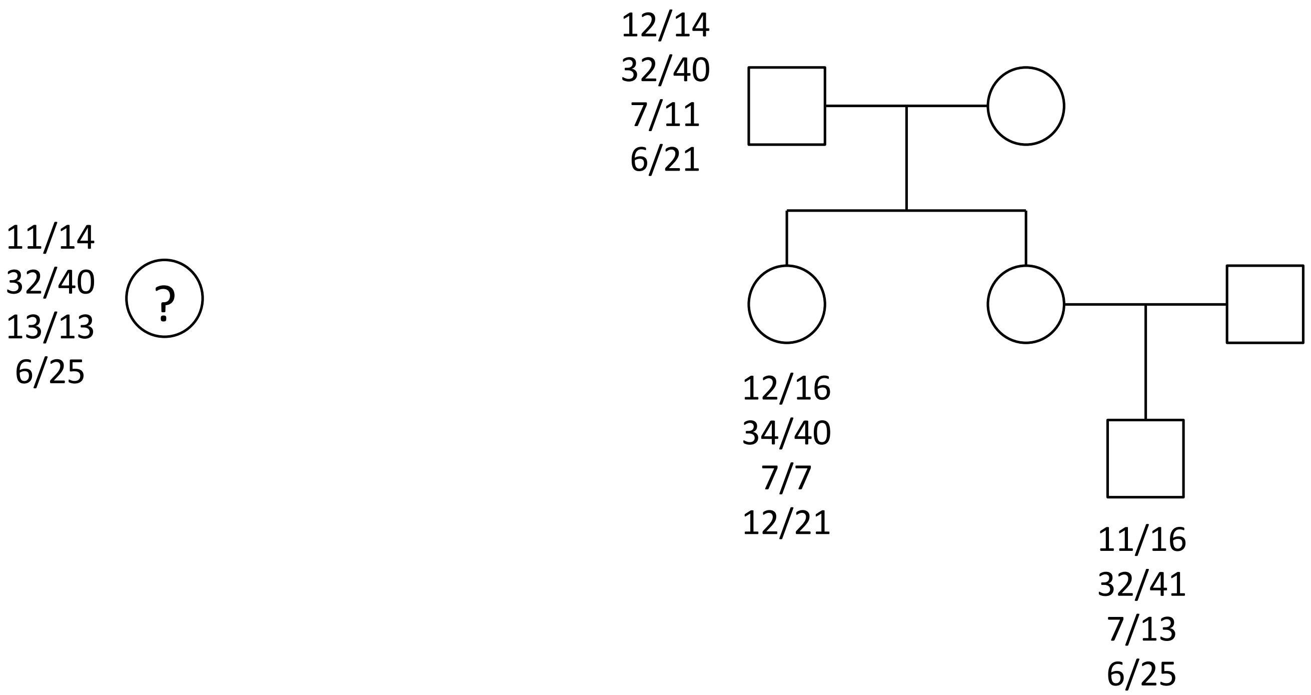
Who is the true father?

Questions related to pedigrees with genotypes



Brothers or half brothers?

Questions related to pedigrees with genotypes



Is this woman related to the family?

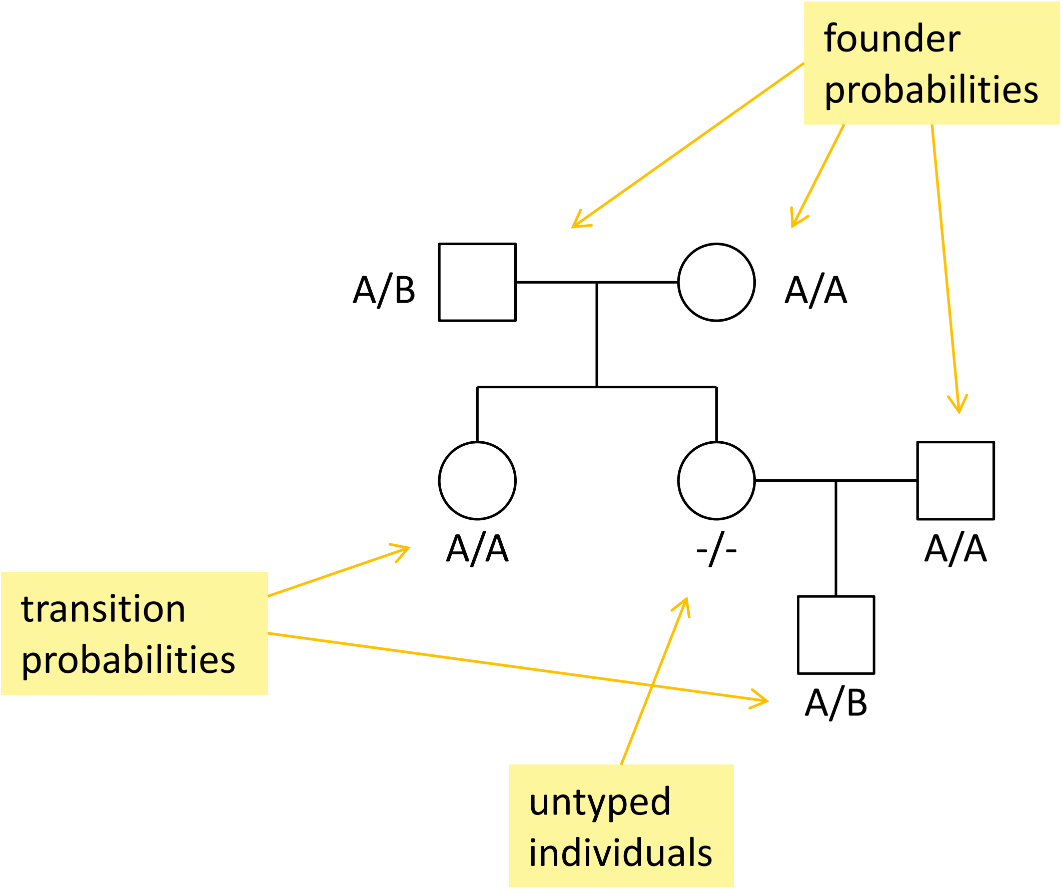
- Many applications involve probabilities of the following form

$$P(\overset{\boxed{\text{data}}}{\text{genotypes}} \mid \text{pedigree}, \overbrace{\text{inheritance model, allele freqs, ...}}^{\boxed{\Theta}})$$

- Often referred to as a *pedigree likelihood*:

$$L(\text{pedigree} \mid \text{data}) = P(\text{data} \mid \text{pedigree}, \Theta)$$

Ingredients for likelihood computations



Software for pedigree likelihoods

- Familias
  - GUI for forensic applications
  - Elston-Stewart
  - handles mutations, theta correction, ++
- MERLIN
  - command line program
  - Lander-Green
  - gold standard for cases with dense SNP markers (but not too large pedigrees)
  - used by FamLink & ped suite to handle linked markers
  - not mutations, not theta correction
- R/ped suite
  - Elston-Stewart
  - mutations, theta correction, ++

Part III: R

What is R?

- A framework for statistical computing
  - calculator
  - data handling and numerical analysis
  - flexible plotting
  - programming language
  - external packages
    - anyone can make one
    - thousands!

Pros

- free!
- very widely used
- anything is possible (but not always easy)
- scripting --> reproducibility

Cons

- steep learning curve
- packages come and go

Why should forensic geneticists use R?

Designed, built and proven for real world case work

Time to get your hands dirty: Trying out R

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

I use this

Two (mostly synonymous) ways to assign values: **=** or **<-**

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

Changing a variable:

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

Common beginners' mistake:
forgetting to assign after change

Creating new variables from old:

```
>   newVar = a^b
>   newVar
[1]  36
```

Most programmers stick to either
camelCase or **snake_case**
when naming their variables

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
```

The `c()` operator!

The `':'` operator
(shortcut for consecutive numbers)

There is a help page
for every function!
`> ?seq`

Character vectors:

```
>   c("Alice", "Bob")
```

Logical vectors:

```
>   c(TRUE, FALSE, T, F)
[1]  TRUE FALSE TRUE FALSE
```

Built-in logcial constants:
TRUE   short form: **T**
FALSE   short form: **F**

Matrix-like containers

Data frames: Collects vectors of the same length

```
>   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
```

Use **$** to refer to columns: **`x$Name`**

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
```

Note: No **$** for matrices!

First column: **`x[, 1]`**
First row: **`x[1, ]`**

Faster, but less flexible. Good for all-numeric (or all-character) data

Lists

```
>   a = list(good = 1:3, bad = 0)
>   a
$good
[1] 1  2  3

$bad
[1] 0
>   a$good
[1] 1  2  3
```

Alternative to **$**:
`a[["good"]]`

Easy to change lists:

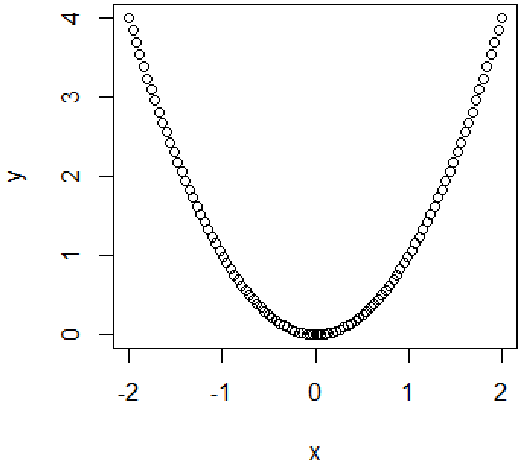
```
>   a$bad = NULL             (delete item)
>   a$ok = -1                (add new item)
>   a$good = c(a$good, 10)  (modify item)
>   a
$good
[1] 1  2  3  10

$ok
[1] -1
```

Basic plotting

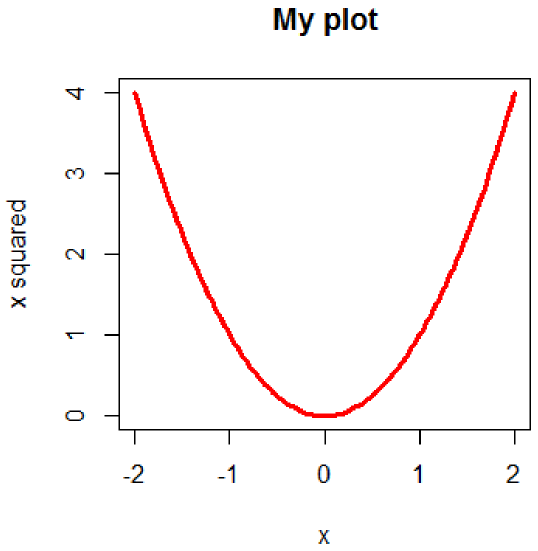
Let's plot the graph of *y* = x^2 !

```
>   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!")
```



R stuff skipped in this brief introduction

- User-defined functions
- Loops, `apply()`, `lapply()`, etc.
- Basic statistics, linear models + +
- Random numbers
- The "tidyverse" for data science
- ... 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()`
- load it into R
  - every new session
  - `library()`

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

```
>   library(pedsuite)
```

If you get an error, do:

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

Your turn: Exercises!