ISFG Summer School - Virtual Edition 2023

Pedigree analysis

Teachers:

Thore Egeland

Magnus Dehli Vigeland

Schedule

The course runs from 18 to 22 (CEST) each day, with a 15 minute break in the middle. The following schedule is tentative:

Aug 30 (Wednesday) – Pedigree analysis: Basic

- 18:00–19:15 Lecture 1. **Pedigrees and measures of relatedness** (MDV)
- 19:15–20:00 Exercise set 1
- 20:00–20:15 *Break*
- 20:15–21:00 Lecture 2. **Kinship testing** (TE)
- 21:00–21:45 Exercise set 2
- 21:45–22:00 Summary and discussion

Aug 31 (Thursday) – Pedigree analysis: Advanced

- 18:00–19:00 Lecture 3. **Relatedness inference and pedigree reconstruction** (MDV)
- 19:00–19:45 Exercise set 3
- 19:45–20:00 *Break*
- 20:00–21:00 Lecture 4. **Disaster victim identification** (TE)
- 21:00–21:45 Exercise set 4
- 21:45–22:00 Summary and discussion

Home page
https://magnusdv.github.io/pedsuite/articles/web_only/course-isfg2023.html

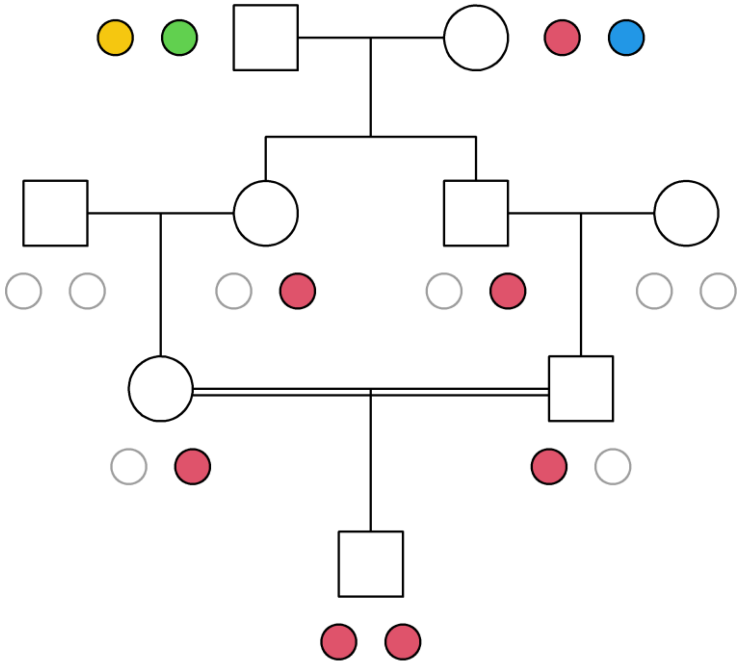
Inference of relatedness
&
Pedigree reconstruction

ISFG Summer School 2023. Workshop 4.2

Pedigree analysis: Advanced

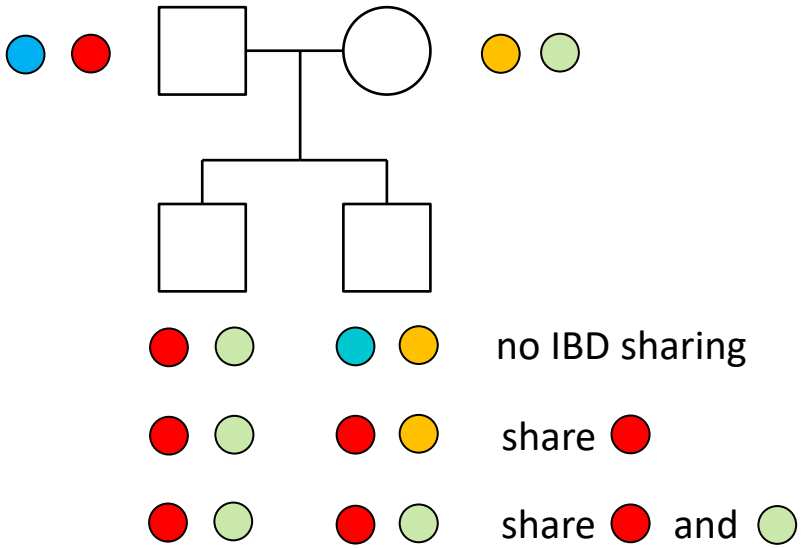
Magnus Dehli Vigeland

Identity by descent (IBD)



IBD coefficients

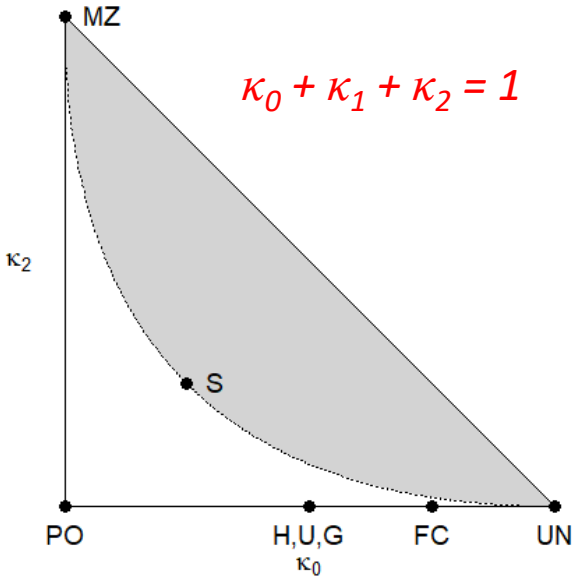
- How many alleles are IBD in each locus?



Definition

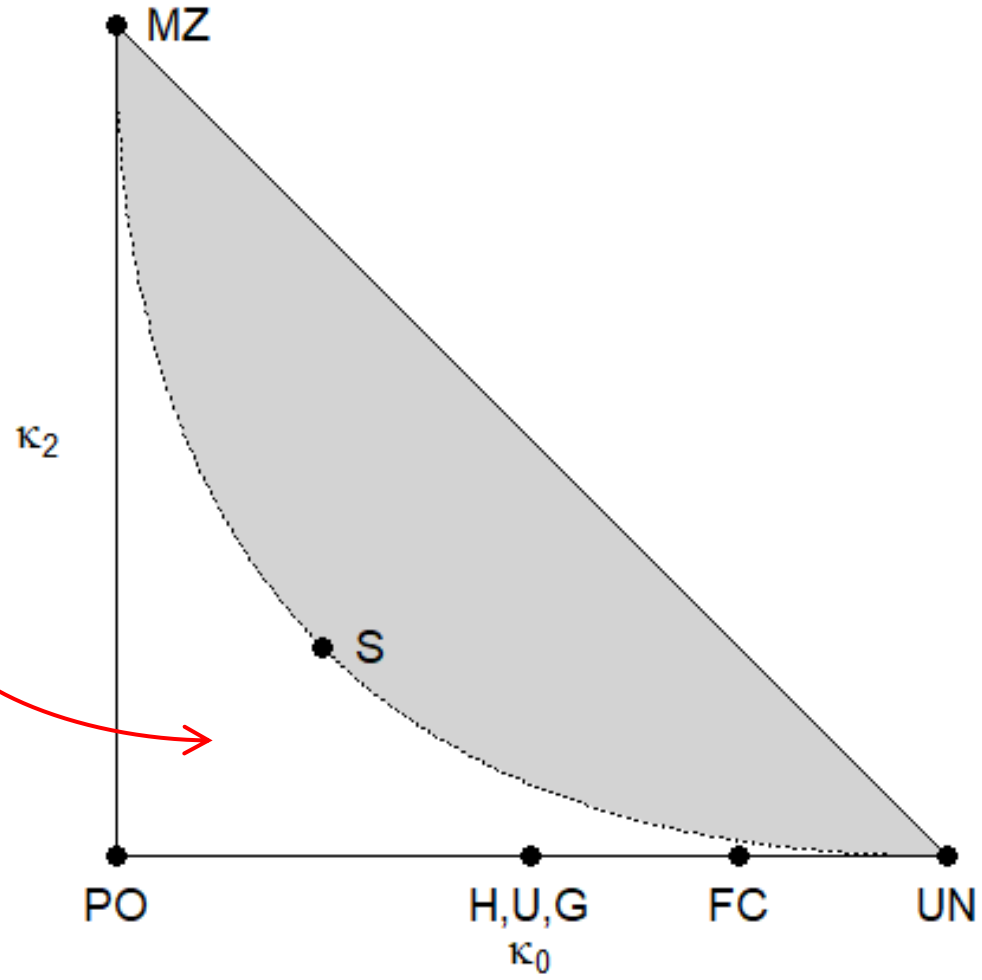
- κ_0 = *Pr*(0 alleles IBD)
- κ_1 = *Pr*(1 alleles IBD)
- κ_2 = *Pr*(2 alleles IBD)

(at random autosomal locus)



Part I: Inference of pairwise relatedness

where??



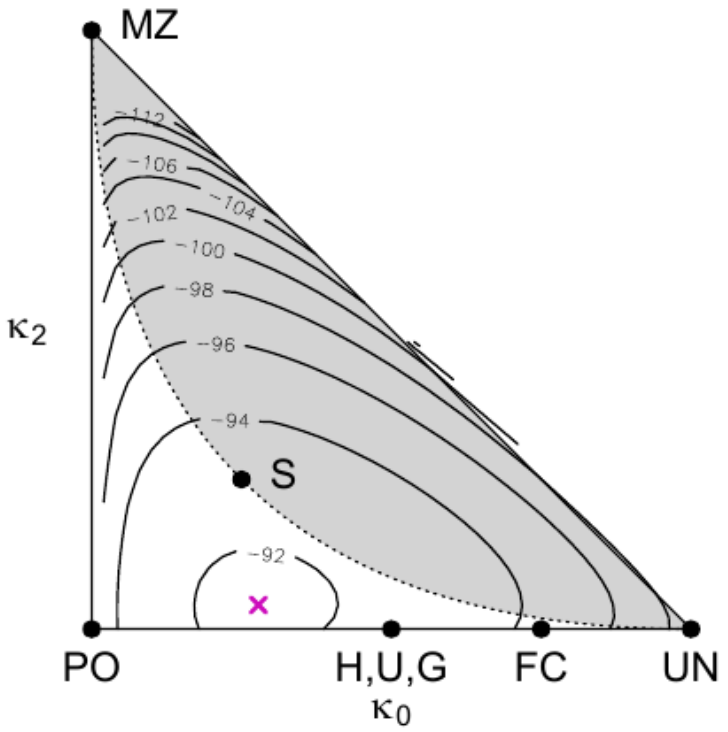
Maximum likelihood estimation of $\kappa = (\kappa_0, \kappa_1, \kappa_2)$

- Thompson (1975)
  - Given: marker genotypes for two individuals
  - The likelihood function

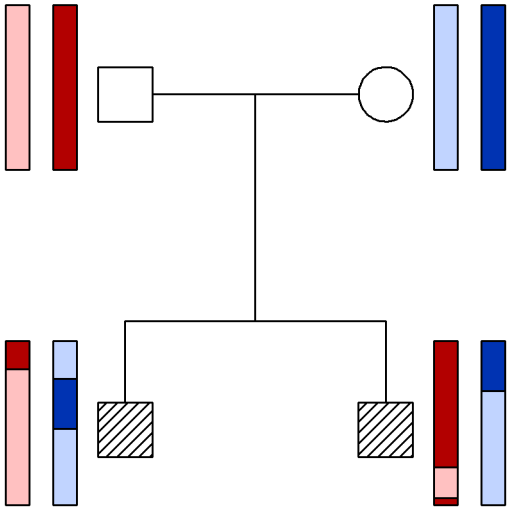
$$L(\kappa) = P(\text{genotypes} \mid \kappa)$$

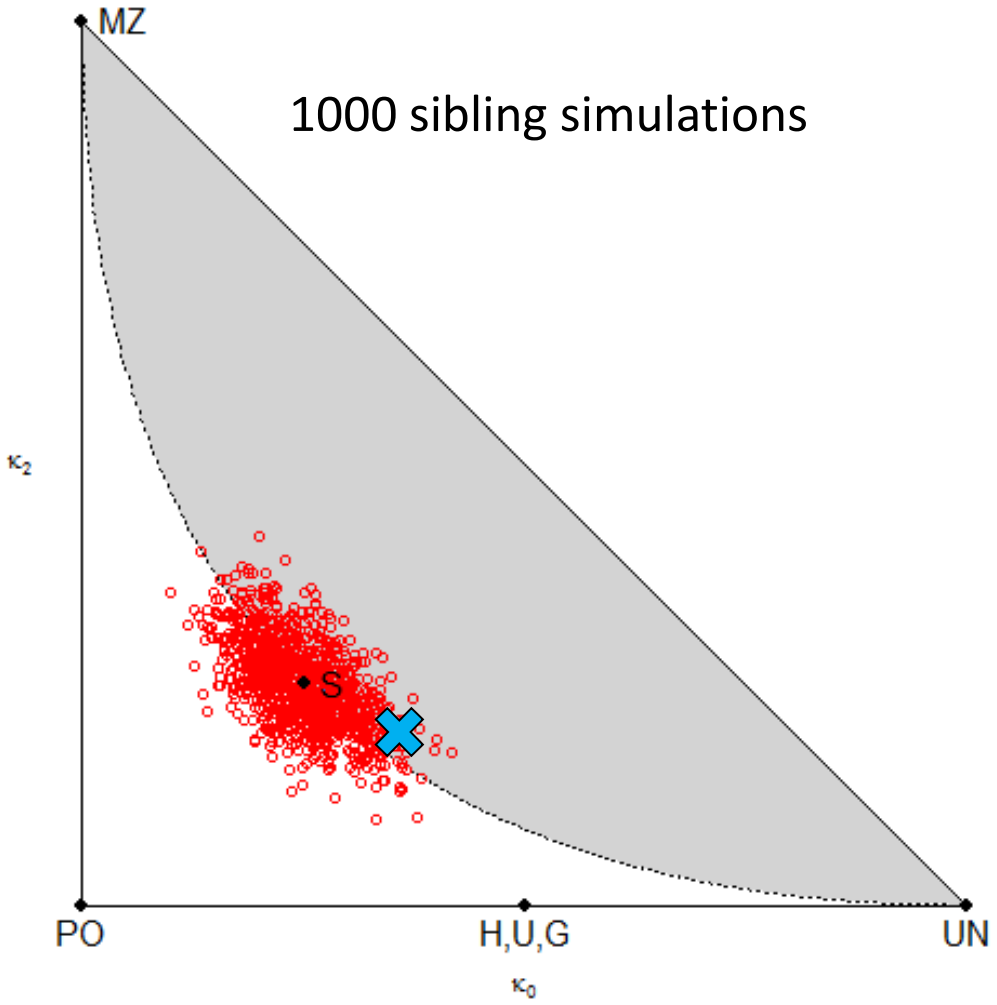
- Find the point *k* which maximizes *L*!
  - Called the maximum likelihood estimate (MLE)

- Assumptions:
  - known allele freqs
  - HWE
  - no inbreeding



What are we estimating?

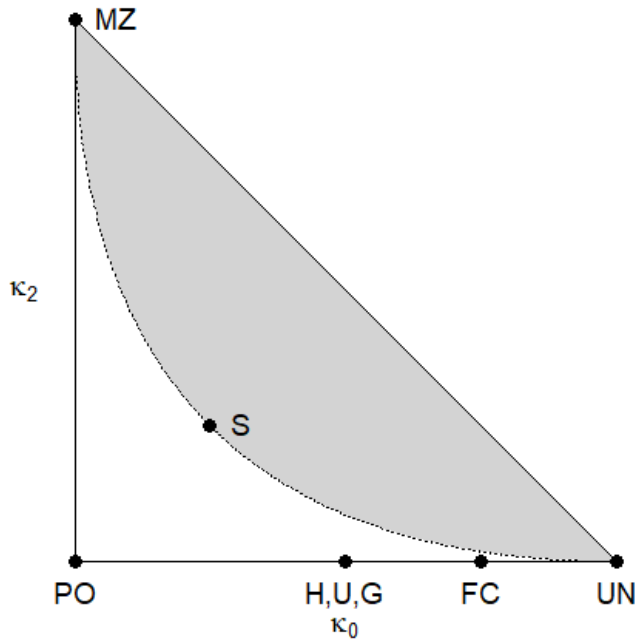




Answer: The *realised* coefficients!

Implementations

- **R**
  - **pedsuite (forrel)**
  - SNPrelate, GWASTools (optimized for association studies)
  - CrypticIBDcheck (as above, slow with many markers)
  - +++
- Other
  - PLINK
  - KING
  - Beagle
  - +++



Pairwise inference in R

- Key functions

```
> ibdEstimate()       # estimate kappa
> showInTriangle()    # visualize!
> ibdBootstrap()      # bootstrap confidence
> checkPairwise()     # detect pedigree errors
```

- Simulation

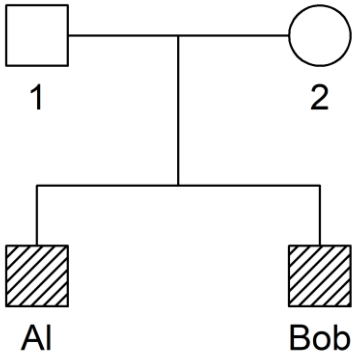
```
> markerSim()         # iid markers
> profileSim()        # complete profiles
```

(Both of these support conditioning on known genotypes)

Pairwise inference in R: Example

Simulate 100 SNPs for a pair of siblings

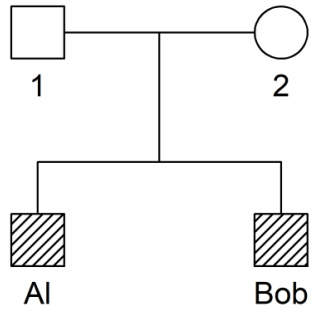
```
>   library(pedsuite) # includes forrel

>   ids = c("Al", "Bob")
>   x = nuclearPed(children = ids)

>   x = markerSim(x, N = 100, ids = ids,
              alleles = 1:2, seed = 1234)
>   x
  id fid mid sex <1> <2> <3> <4> <5>
   1   *   *   1 -/- -/- -/- -/- -/-
   2   *   *   2 -/- -/- -/- -/- -/-
  Al   1   2   1 1/1 1/2 1/1 1/2 2/2
 Bob   1   2   1 1/1 1/2 1/1 1/2 2/2
Only 5 (out of 100) markers are shown.

>   dat = list(subset(x, "Al"),
                subset(x, "Bob"))
```

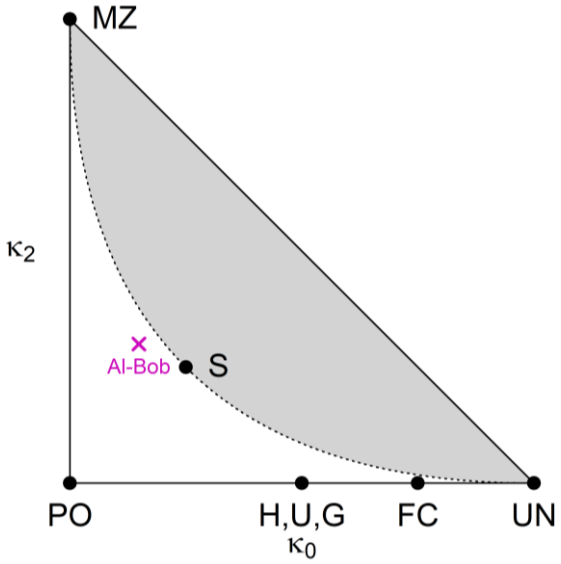


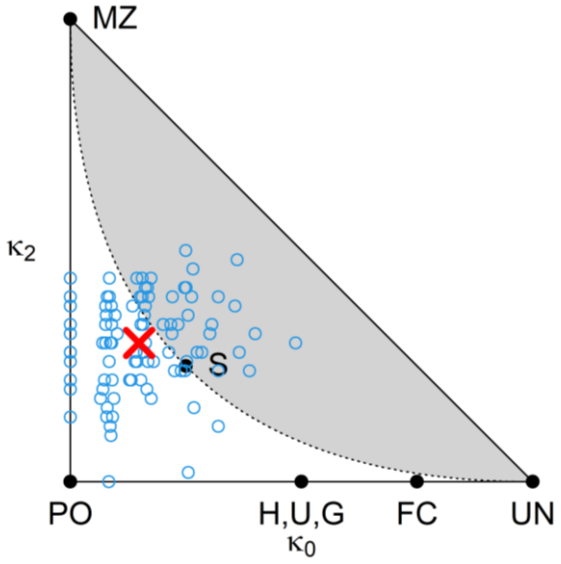
Pairwise inference in R: Example

Estimate kappa from the data

```
>   k = ibdEstimate(dat)
>   k
  id1 id2   N      k0       k1       k2
1  Al Bob 100 0.1486 0.55139 0.30002
>   showInTriangle(k, labels = T)
>   bs = ibdBootstrap(dat, ids, N = 100,
                      param = "kappa")
```

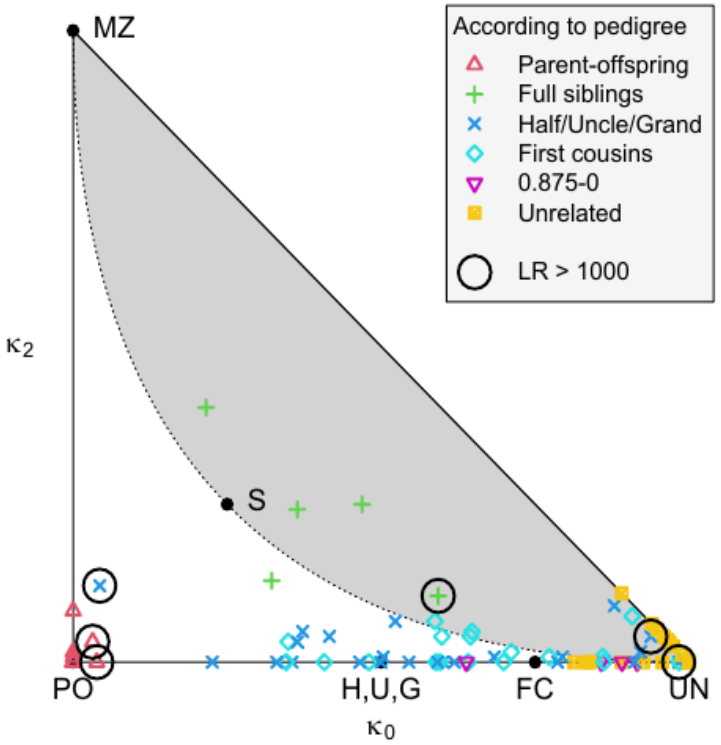




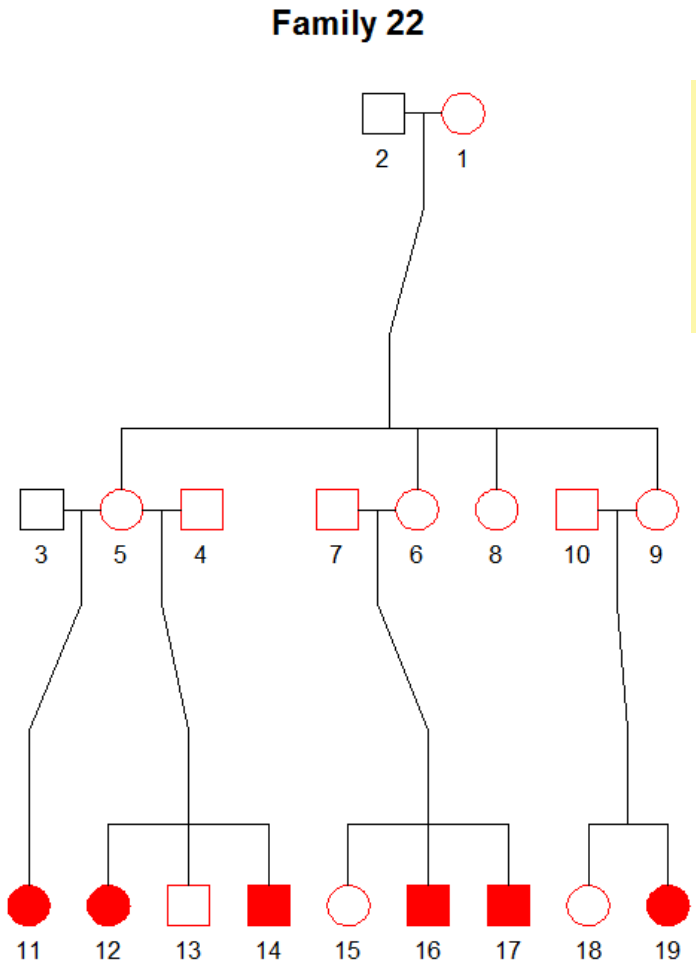


Application: Detecting pedigree errors

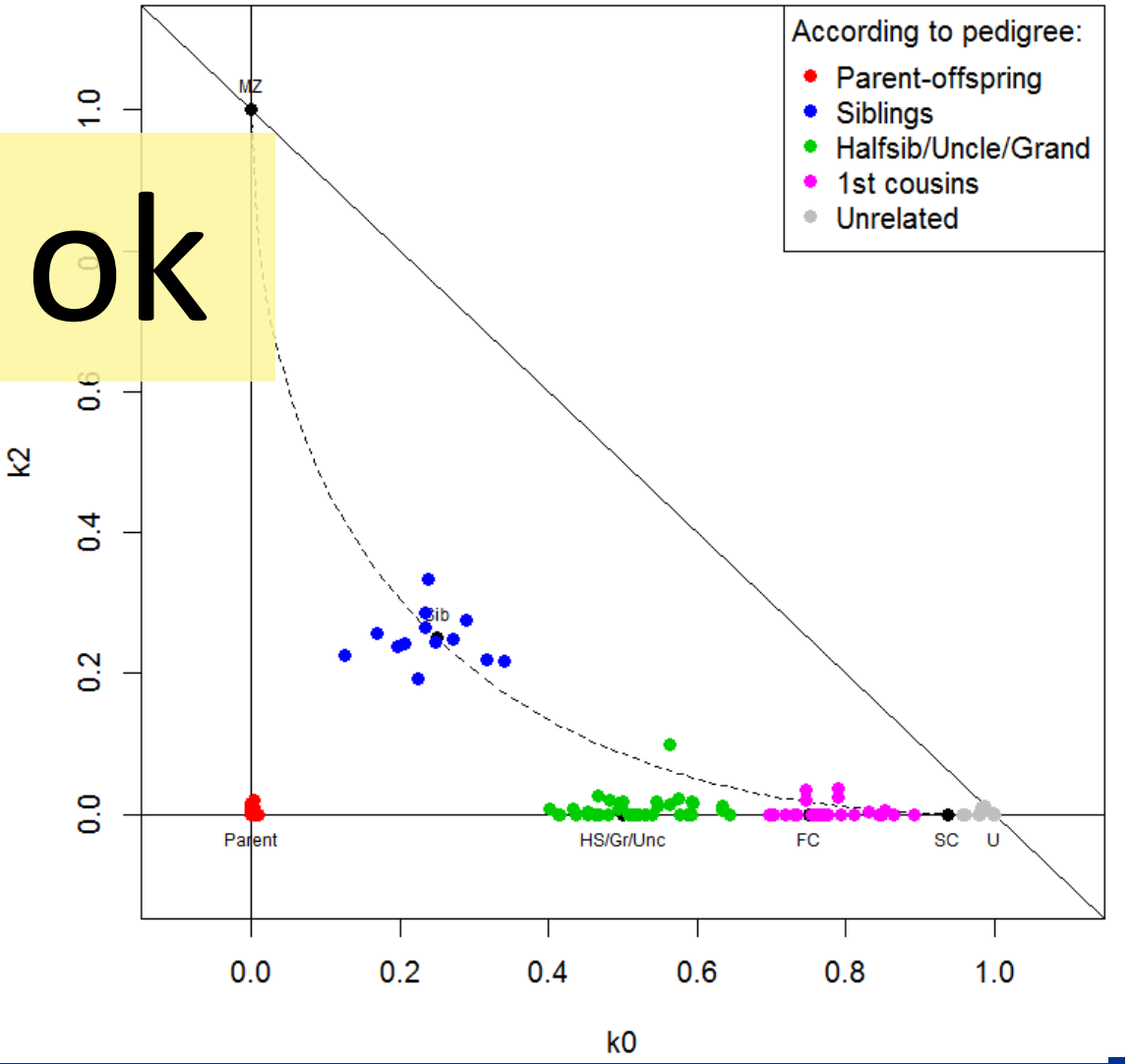
- Suppose `x` is a pedigree object with attached markers
- The function `checkPairwise(x)` computes:
  - pedigree-based kappa for all pairs: `kappaIBD(x)`
  - marker-based kappa estimates for all pairs: `ibdEstimate(x)`
  - LR comparing the two
  - Color-coded plot according to relationship claimed by pedigree



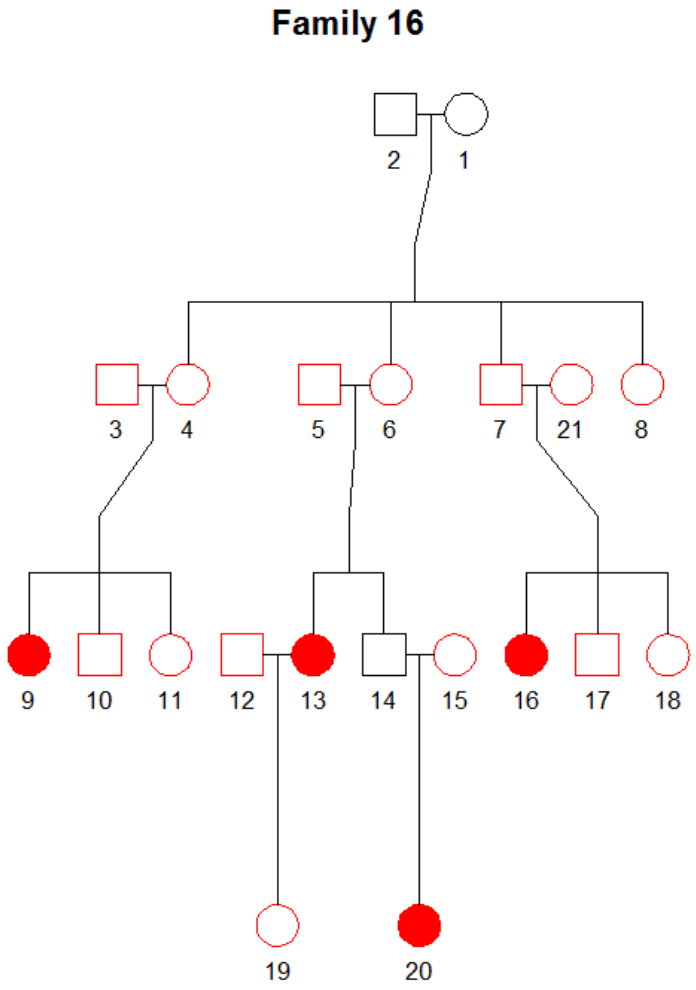
checkPairwise(): Example 1

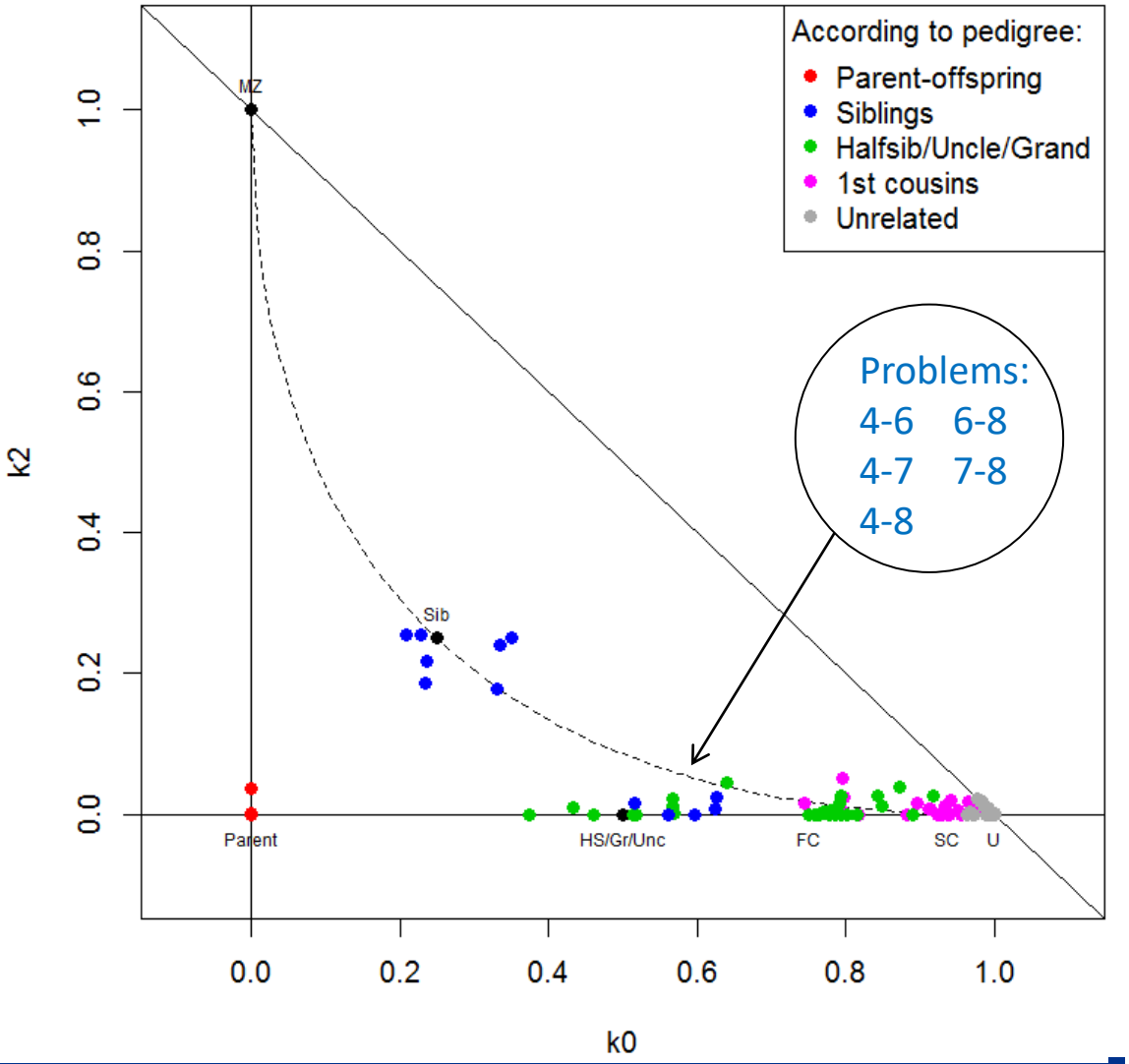


ok

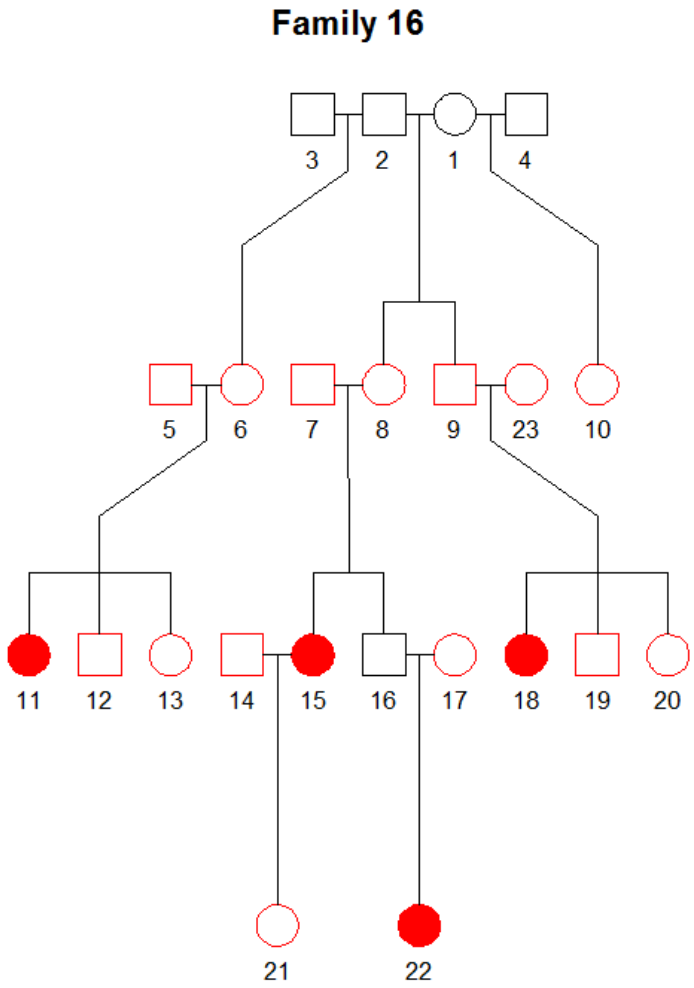


checkPairwise(): Example 2

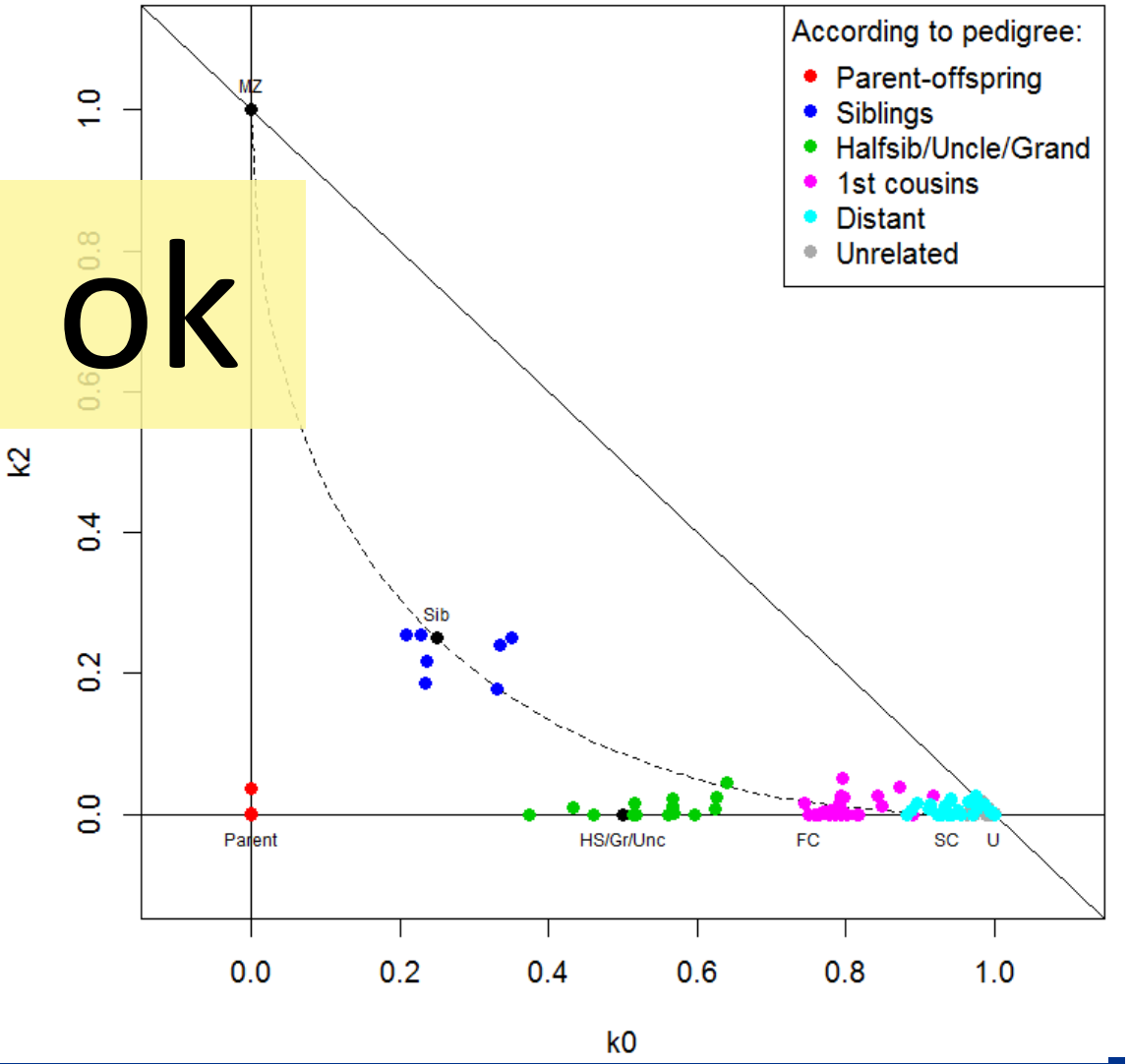




checkPairwise(): Example 2 - corrected



ok

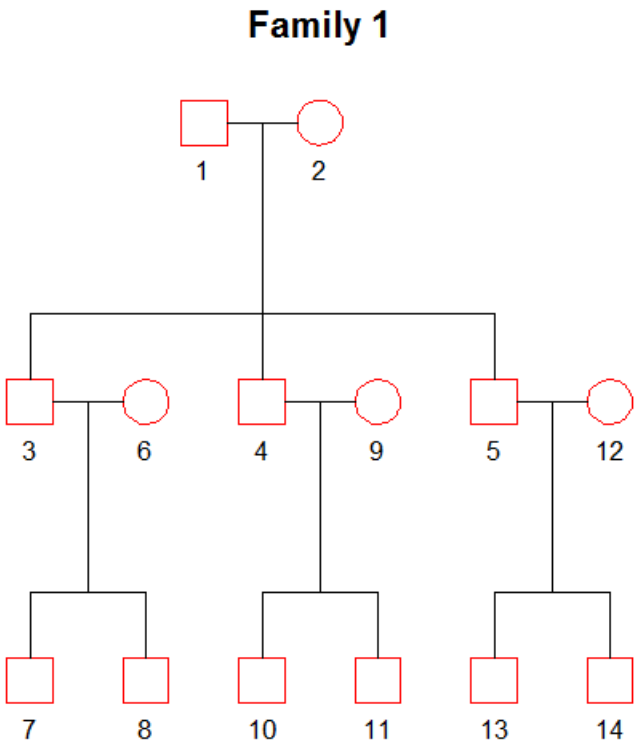


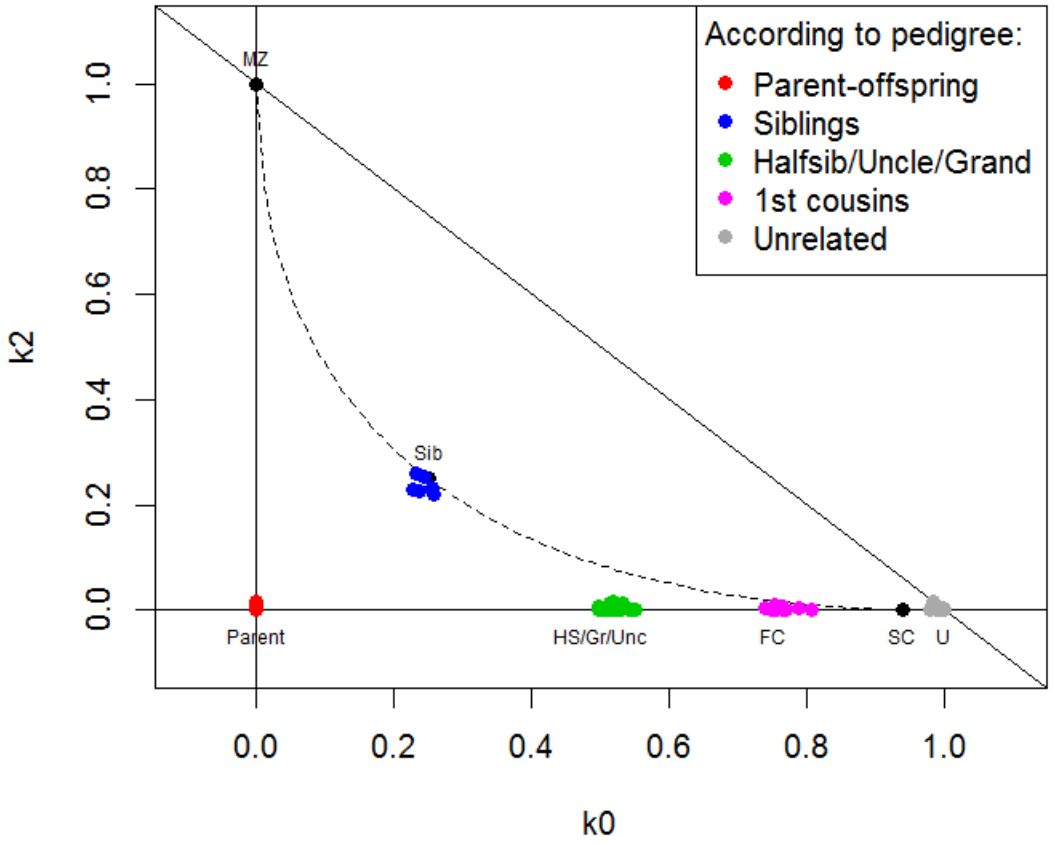
Relatedness inference vs. allele frequencies

- A little simulation experiment!

Simulation example

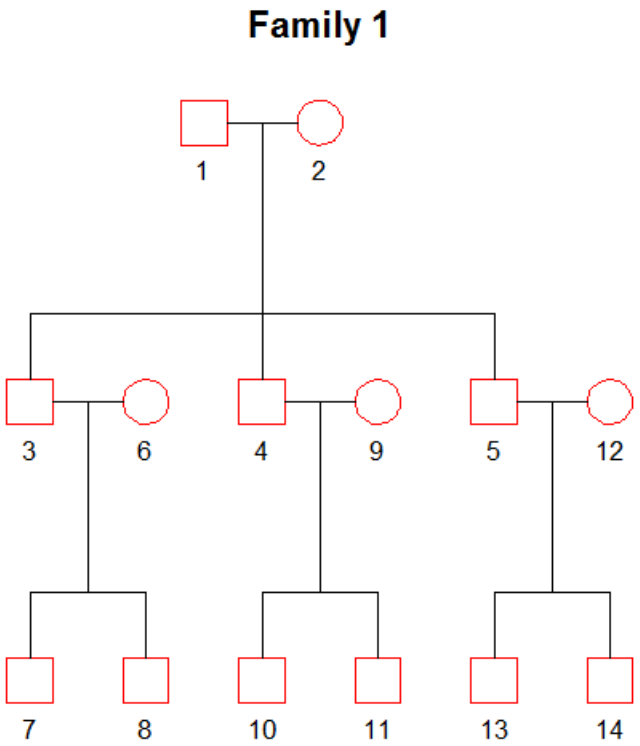
SNPs: 10 000
True frequency distr: Unif(0,1)
Frequencies used: Correct

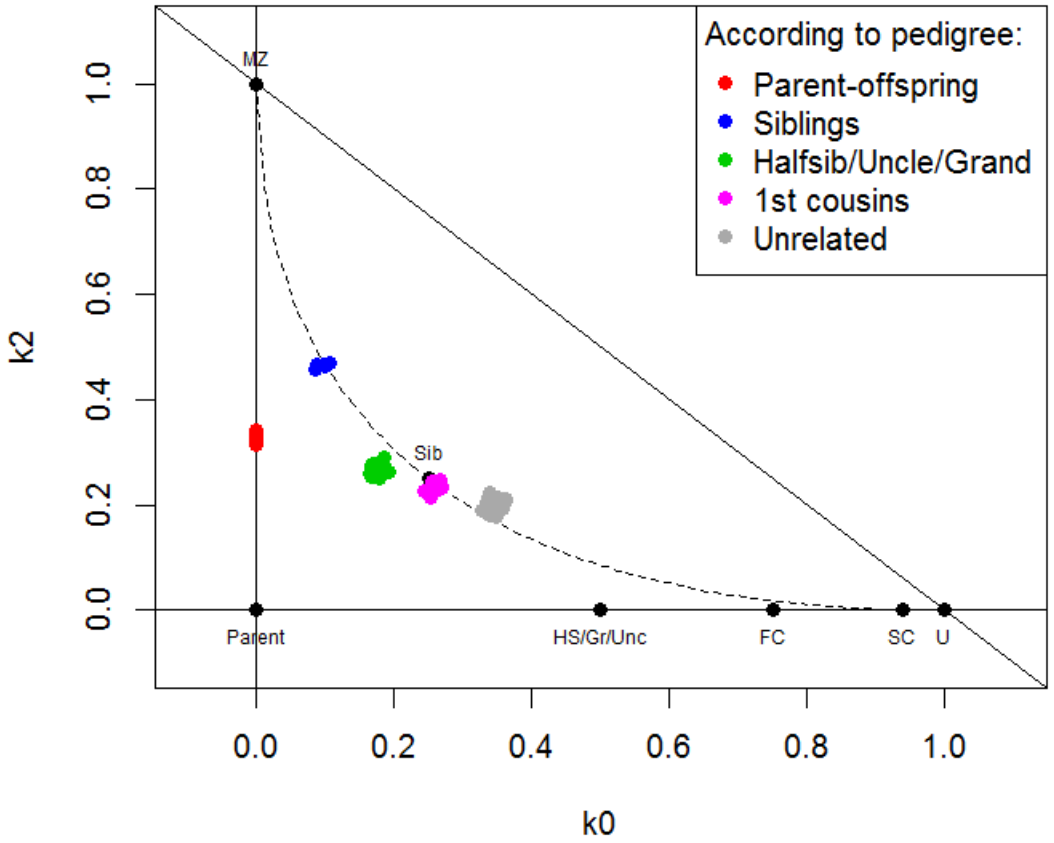




Simulation example

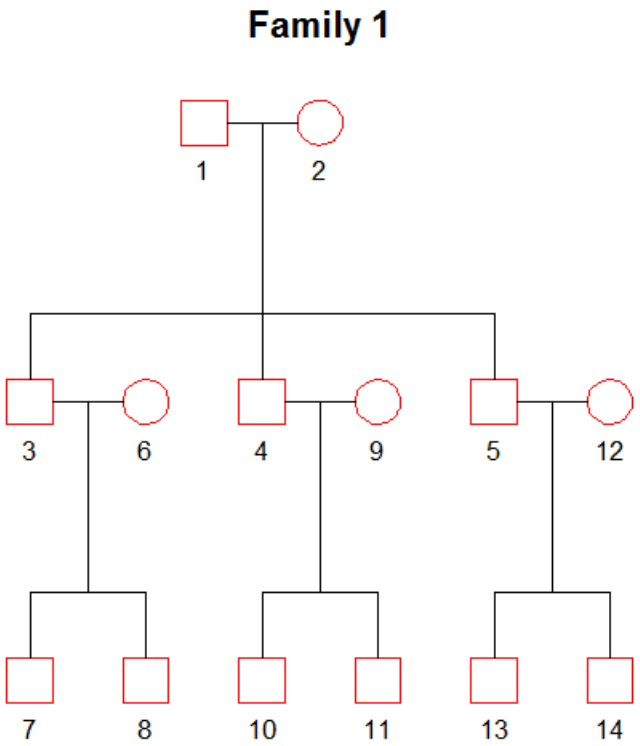
SNPs: 10 000
True frequency distr: Unif(0,1)
Frequencies used: All = 0.5

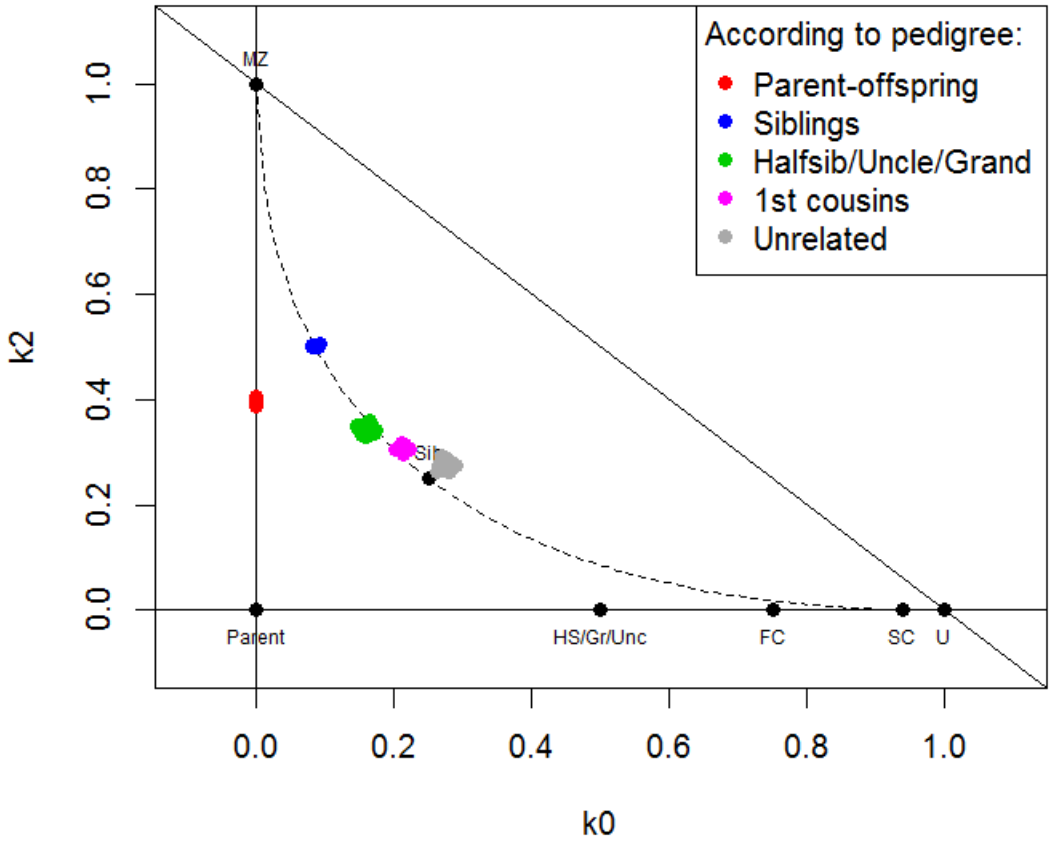




Simulation example

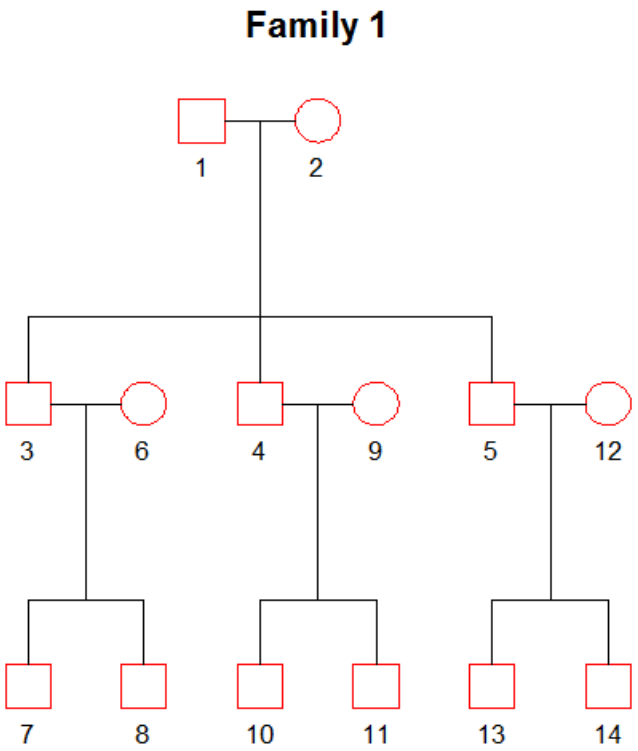
SNPs: 10 000
True frequency distr: Unif(0,1)
Frequencies used: Unif(0,1)

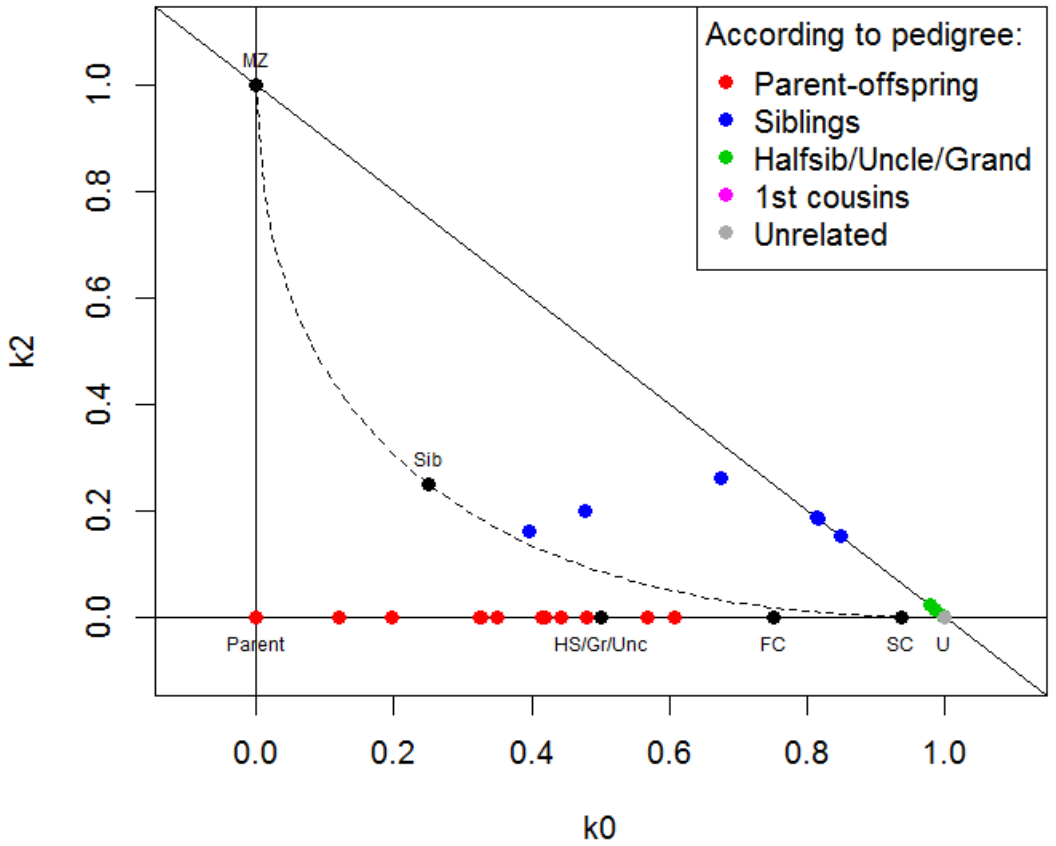




Simulation example

SNPs: 10 000
True frequency distr: Unif(0,1)
Frequencies used: Family estimate

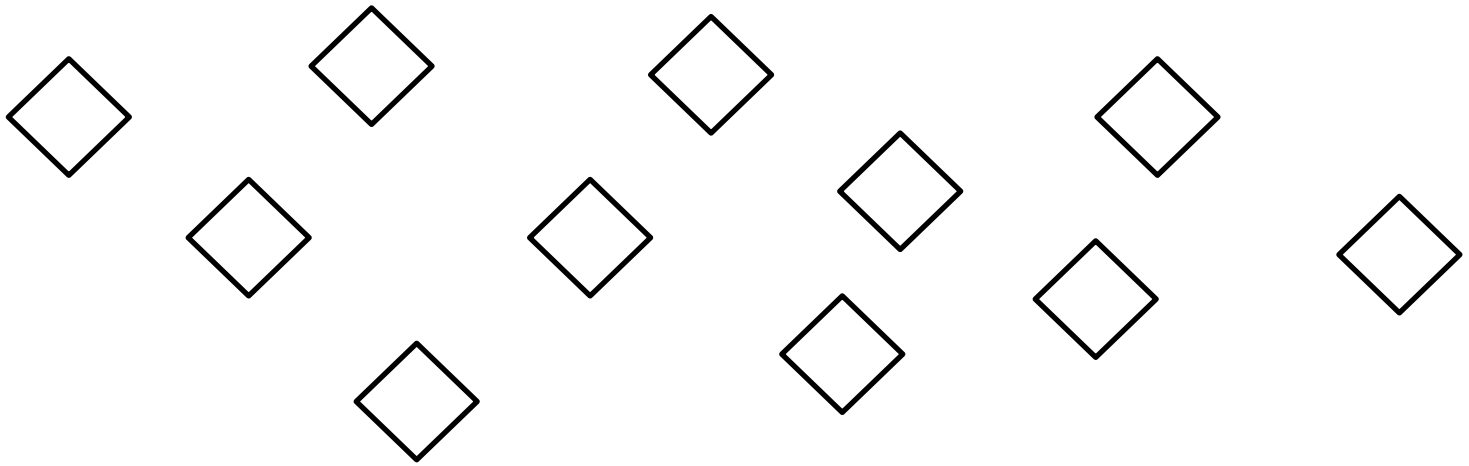




- Conclusion: Pairwise inference is quite sensitive to wrong allele frequencies

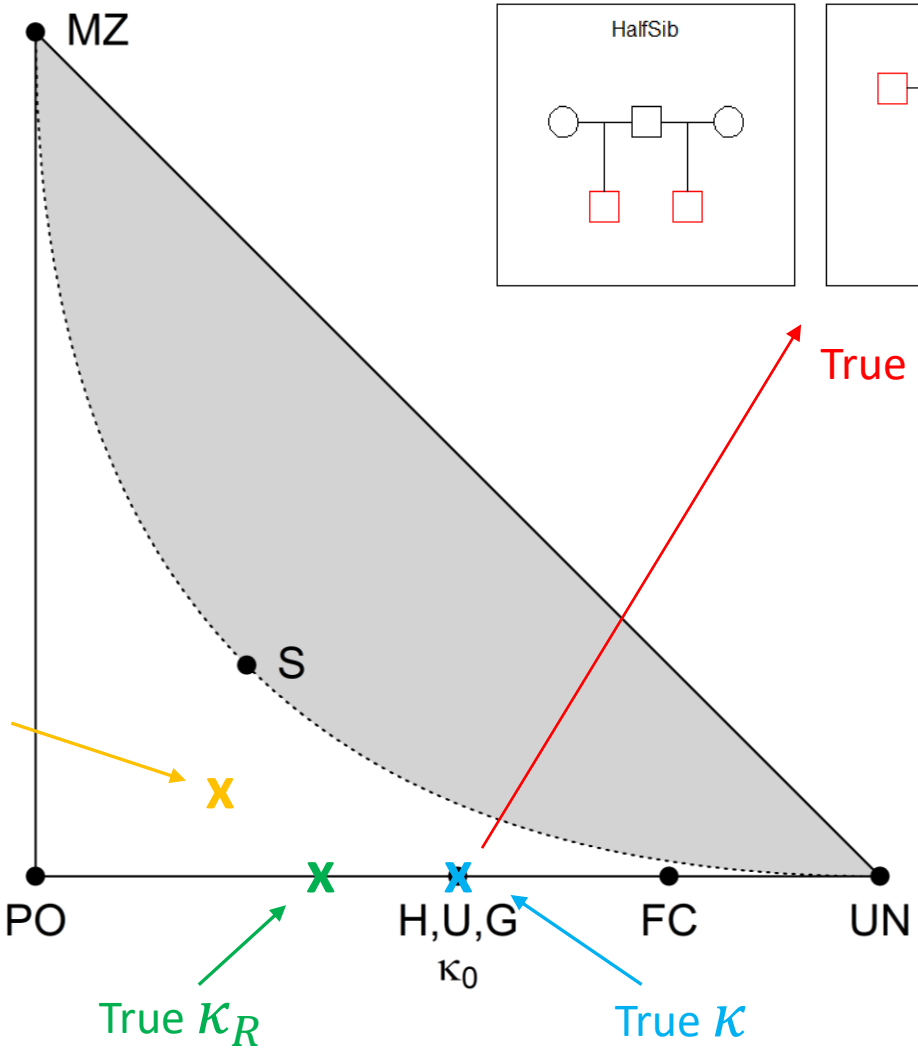
Part II: Pedigree reconstruction

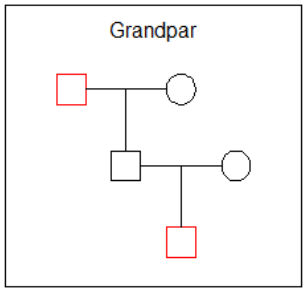
Pedigree reconstruction

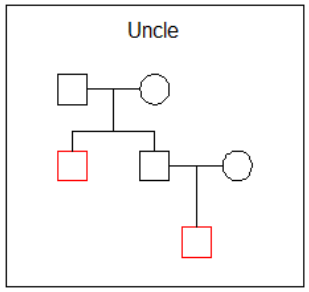


Goal: Reconstruct the complete pedigree from genotype data

Impossible - even in theory!





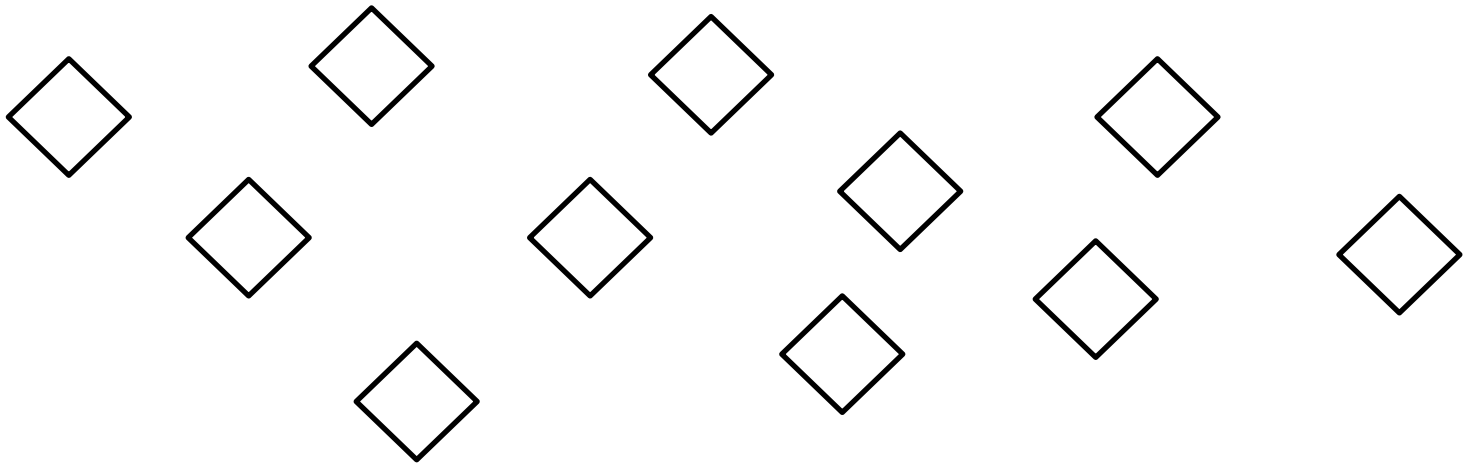


True pedigree

What can we trust?

- PO usually ok
- Sibs if good data

Pedigree reconstruction



Naive approach

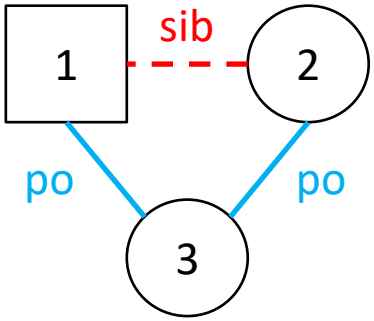
Step 1: Genders

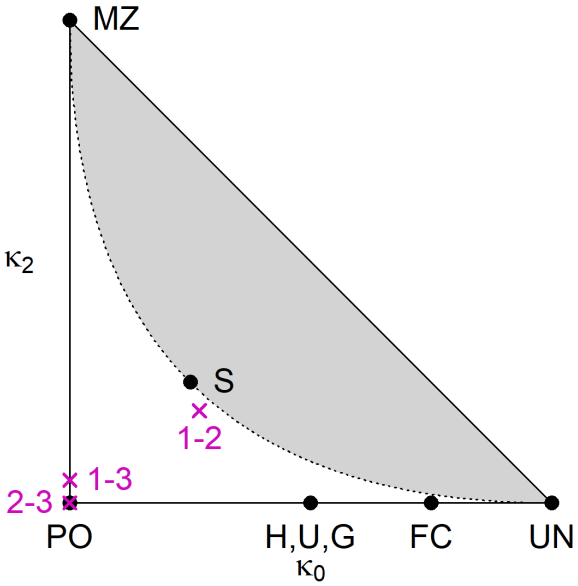
Step 2: Estimate pairwise relationships

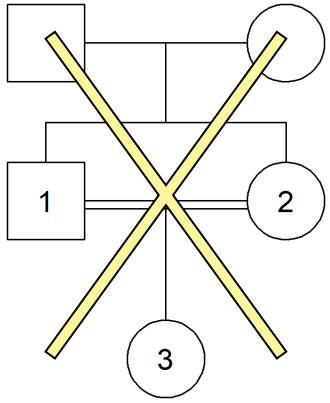
- Connect parent-child
- Exploit siblings

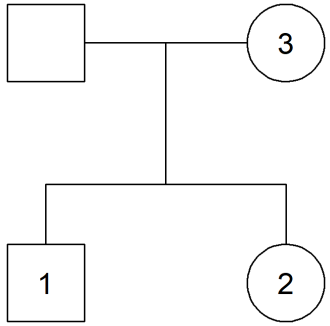
Step 3: Solve the puzzle!

Example









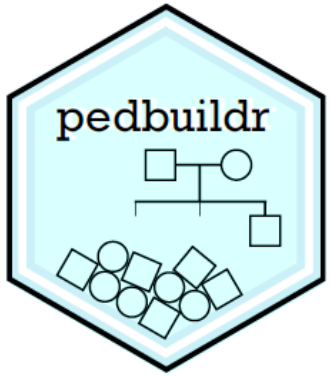
Better approach: Maximum likelihood

Idea:

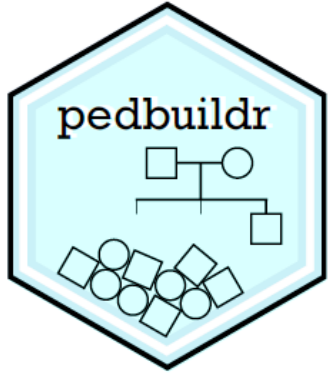
- Generate a list of "all possible" pedigrees connecting the individuals
- Compute the likelihood of each pedigree
- Sort and output the best pedigrees

Key functions:

```
> buildPeds()     # generate pedigrees

> reconstruct()   # main function!
> plot()          # plot top hits
```



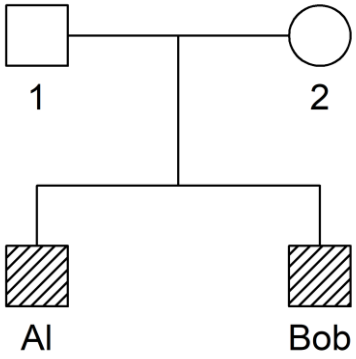
pedbuildr: Example



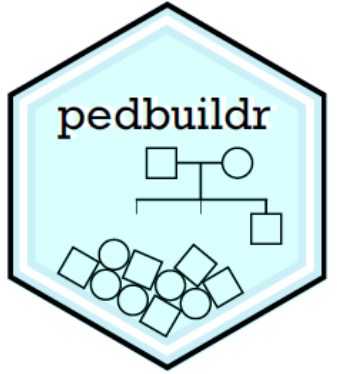
Same dataset as before:

Simulate 100 SNPs for a pair of siblings

```
>   library(pedsuite)

>   ids = c("Al", "Bob")
>   x = nuclearPed(children = ids)

>   x = markerSim(x, N = 100, ids = ids,
              alleles = 1:2, seed = 1234)
>   x
  id fid mid sex <1> <2> <3> <4> <5>
   1   *   *   1 -/- -/- -/- -/- -/-
   2   *   *   2 -/- -/- -/- -/- -/-
  Al   1   2   1 1/1 1/2 1/1 1/2 2/2
 Bob   1   2   1 1/1 1/2 1/1 1/2 2/2
Only 5 (out of 100) markers are shown.

>   dat = list(subset(x, "Al"),
                subset(x, "Bob"))
```

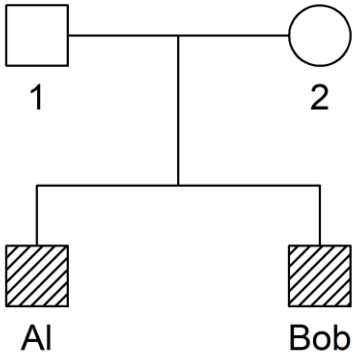


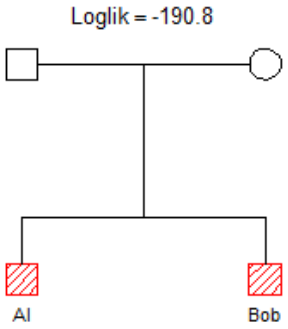
pedbuildr: Example

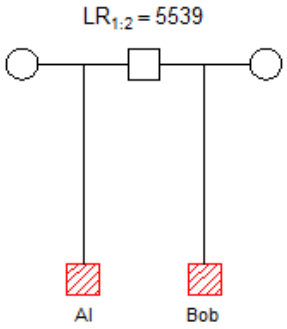


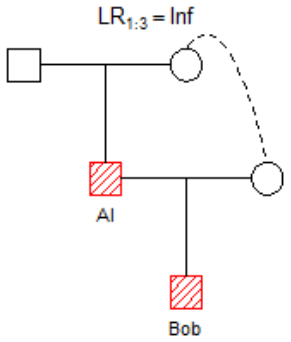
Reconstruct the most likely

```
>   library(pedbuildr)

>   r = reconstruct(dat)
Pedigree parameters:
  ID labels: Al, Bob
  Sex: 1, 1
  Extra: parents
  Age info: -
  Known PO: -
  ...

Building pedigree list:
  ...
Computing the likelihood of 6 pedigrees.

>   plot(r, top = 3)
```





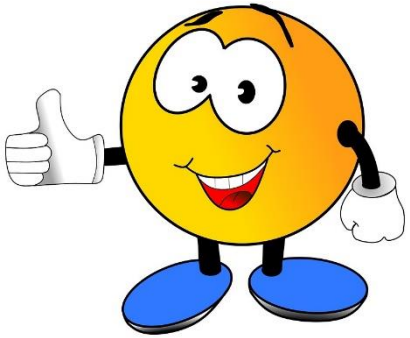


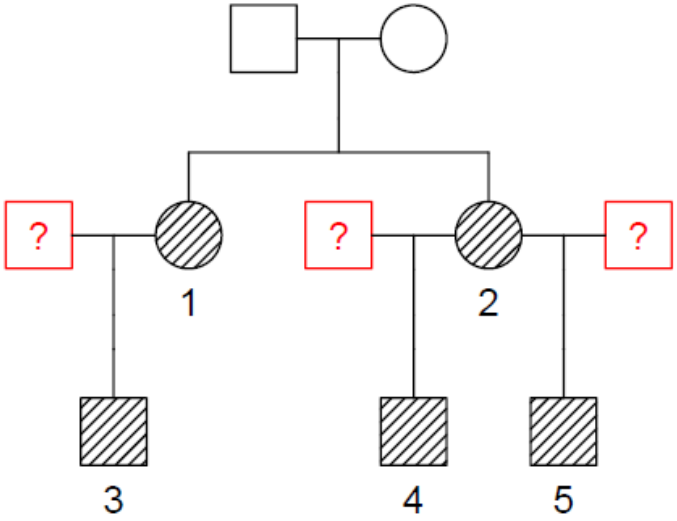


Optional parameters for restricting the search space

- **`extra`**: The max number of connecting individuals
  - default: **`extra = "parents"`** (suitable for small datasets)
- **`maxInbreeding:`** Default: 1/16 (e.g., first cousins)
- **`age`**: A vector of (relative) ages OR age inequalities, e.g. "Al > Bob"
- **`inferPO`**: If TRUE, an initial stage of pairwise IBD estimation is done
- **`knownPO`**: Known parent–offspring pairs
- **`allKnown`**: Is **`knownPO`** the *complete* list of POs?
- **`notPO`**: Pairs known not to be parent–offspring
- **`noChildren`**: Individuals known to have no children
- **`linearInb`**: Max incestuous generation gap (default: 0)
- **`connected`**: Set to FALSE to allow disconnected pedigrees
- **`sexSymmetry`**: Remove 'symmetric' versions. Default: TRUE

Your turn: Exercises!





Q: Do any of the children have the same father?