Project: exclusionPower

Thore, Elias and Magnus

Created May 30, 2019. Updated Jul 07, 2019

June 28 2019: Meeting Magnus-Thore

Below some are some suggestions for the `shiny-gui`.

1. Name (cf. Kanban board; not sure if/how one comments directly on the board.) We suggest epGUI (short for exclusionPowerGUI).
2. It would be nice to have the plots of the pedigrees available at all times. The Save/load work space square could move down right and plots could be to the left.
3. The could be a `Settings`, say to the left of `Results`. Some parameters (like `inbreeding parameters`, search founderInbreeding below for an example, `mutation model` and X/autosomal could be set. It's not obvious what goes here and what goes in the specific windows. Perhaps there should be choice for exact calculation (default) and/or simulation.)
   4.Genetic data. Allele designations, frequencies. The current version is fine, but could be more general (eg., the ped-suite allows one allele to be missing) and Magnus will, as he has planned before this project, implement a `readTable` function. This file could then be edited.
4. Genetic data. Reference profiles. Not yet implemented. These are marker data specified by the `known` parameter of `exclusionPower` and exemplified below. We suggest that this rather be added as columns in the claimPed-file (any such data in the claimTrue ped file can be ignored) as exemplified in example-sister-more/sister-known.ped. There are various ways to enter alleles that will be handled by the mentioned `readTable` function. For the X-chromosome, there will be at most one allele for males.
5. A long term goal would be to read the export from Windows Familias.
6. Plotting of unrelated does not work. This plotting is a bit awkward and not supported by `kinship2`. A possibility:

```
pedtools::plotPedList(list(singleton(3,1), singleton(4,2)), frames = F)
```

Status

The shiny app can be installed and run:

```
devtools::install_github('magnusdv/forrel', ref = 'shiny-gui')
forrel::epGUI()
```

This document and some examples are available at https://github.com/thoree/exclusion

Further work

Code improvements

The code of `exclusionPower()` is fairly old and has room for improvements. These include:

- Better validation of input data
- The syntax for indicating known genotypes is poor and deserves re-thinking
- Plots are created very ad hoc; better to use `pedtools::plotPedList()`

Magnus will work on all of these during this project.

Suggestions

- Is is possible to detect if calculation will take too long and if so do simulation? I assume the above code example can be improved.
- Future versions, extensions could deal with linked markers (but probably not linkage disequilibrium).. My feeling is that the demand would, should first come for X-chromosomal markers and that simulation using minx http://csg.sph.umich.edu/abecasis/merlin/reference.html would be the practical choice. The page http://famlink.se/fx_download.html gives a number of databases for X-chromosomal markers.

Goal, papers

The goal for the project is to make a nice interface to the R function `exclusionPower` to be used primarly by forensic case workers to calculate exclusion power in kinship cases involving autosomal or X-chromosomal markers. It should assume little if any knowledge of R. Also the three of us should try to write a short paper to describe the implementation, including possible modifications of the previous code, and also hopefully some new examples. The files mentioned below are attached

- The paper Egeland, Pinto, Vigeland (2014), including supplementary. This paper referers to the `R library paramlink`, later replaced by the `pedtools-suite`as described in Magnus' presentation:
  - 1.2_magnus.pdf, see also https://github.com/magnusdv/pedtools and https://github.com/magnusdv/forrel.

There is also a poster

- Exclusion_poster2012.pdf

Examples

The current version of Magnus' function is in the `forrel` library, installed from GitHub as follows.

```
devtools::install_github("magnusdv/forrel")
```

Example 1: from help file

The below code implements the examples, also the `not-run` ones, given on the help page of `exclusionPower()`. I have used the option `plot = FALSE` some places to limit the output.

```
library(forrel, quietly = T)

##########################################
### A standard case paternity case:
### Compute the power of exclusion when the claimed father is in fact
### unrelated to the child.
##########################################

# Claim: Individual 1 is the father of indiv 3
claim = nuclearPed(nch = 1, sex = 2)

# Truth: 1 and 3 are unrelated
true = list(singleton(id = 1), singleton(id = 3, sex = 2))

# Indivs 1 and 3 are available for genotyping
ids = c(1, 3)

# An equifrequent SNP
als = 2
afreq = c(0.5, 0.5)
```

```
# The exclusion power assuming no known genotypes
PE1 = exclusionPower(claim, true, ids = ids, alleles = als, afreq = afreq)

# Exclusion power if the child is known to have genotype 1/1:
PE2 = exclusionPower(claim, true, ids = ids, alleles = als, afreq = afreq,
                     known_genotypes = list(c(3, 1, 1)), plot = FALSE)
```

```
## Removed already typed individuals from internal computations: 3
```

```
# Exclusion power if the SNP is on the X chromosome
PE3 = exclusionPower(claim, true, ids = ids, alleles = als, afreq = afreq,
                     Xchrom = TRUE, plot = FALSE)

stopifnot(PE1==0.125, PE2==0.25, PE3==0.25)
```

```
##############################################
### Example from Egeland et al. (2012):
### Two females claim to be mother and daughter. Below we compute the power of various
### markers to reject this claim if they in reality are sisters.
##############################################

mother_daughter = nuclearPed(1, sex = 2)
sisters = relabel(nuclearPed(2, sex = c(2, 2)), c(101, 102, 2, 3))
ids = c(2, 3)
# Equifrequent SNP:
PE1 = exclusionPower(ped_claim = mother_daughter, ped_true = sisters, ids = ids,
                     alleles = 2)

# SNP with MAF = 0.1:
PE2 = exclusionPower(ped_claim = mother_daughter, ped_true = sisters, ids = ids,
                     alleles = 2, afreq = c(0.9, 0.1),  plot = F)

# Equifrequent tetra-allelic marker:
PE3 = exclusionPower(ped_claim = mother_daughter, ped_true = sisters, ids = ids,
                     alleles = 4, plot = FALSE)

# Tetra-allelic marker with one major allele:
PE4 = exclusionPower(ped_claim = mother_daughter, ped_true = sisters, ids = ids,
                     alleles = 4, afreq = c(0.7, 0.1, 0.1, 0.1), plot = FALSE)

stopifnot(round(c(PE1,PE2,PE3,PE4), 5) == c(0.03125, 0.00405, 0.08203, 0.03090))

####### How does the power change if the true pedigree is inbred?
sisters_LOOP = addParents(sisters, 101, father = 201, mother = 202)
```

```
## Father: Creating new individual with ID = 201
```

```
## Mother: Creating new individual with ID = 202
```

```
sisters_LOOP = addParents(sisters_LOOP, 102, father = 201, mother = 203)
```

```
## Mother: Creating new individual with ID = 203
```

```
# Equifrequent SNP:
PE5 = exclusionPower(ped_claim = mother_daughter, ped_true = sisters_LOOP,
                     ids = ids, alleles = 2)
```

```
# SNP with MAF = 0.1:
PE6 = exclusionPower(ped_claim = mother_daughter, ped_true = sisters_LOOP,
                     ids = ids, alleles = 2, afreq = c(0.9, 0.1), plot = FALSE)

stopifnot(round(c(PE5,PE6), 5) == c(0.03125, 0.00765))
```

```
PE7 = exclusionPower(ped_claim = mother_daughter, ped_true = sisters_LOOP,
                     ids = ids, alleles = 4, plot = FALSE)
# Tetra-allelic marker with one major allele:
PE8 = exclusionPower(ped_claim = mother_daughter, ped_true = sisters_LOOP,
                     ids = ids, alleles = 4, afreq = c(0.7, 0.1, 0.1, 0.1),
                     plot = FALSE)
stopifnot(round(c(PE7,PE8), 5) == c(0.07617, 0.03457))
```

Example 2: Paternal half sisters

This is an important special case:

- Claim: The girls 1 and 2 are paternal half sisters.
- True: The girls 1 and 2 are unrelated.

It's reasonable to use X-chromosal markers and below a case with two unlinked such markers in linkage equilibrium are considered.

```
halfSisters = halfSibPed(sex1 = 2, sex2 = 2)
unrelated = list(singleton(4, sex = 2), singleton(5, sex = 2))
ids = c(4, 5)

p1 = (1:4)/sum(1:4)
als1 = 1:length(p1)

PE9.1 = exclusionPower(ped_claim = halfSisters, ped_true = unrelated, Xchrom = TRUE,
                     ids = ids, alleles = als1, afreq = p1)

p2 = (1:10)/sum(1:10)
als2 = 1:length(p2)

PE9.2 = exclusionPower(ped_claim = halfSisters, ped_true = unrelated, Xchrom = TRUE,
                     ids = ids, alleles = als2, afreq = p2, plot = FALSE)

PE9 = 1 - (1 - PE9.1) * (1 - PE9.2)
```

Next, the answer is checked against the exact formula.

```
f = function(p){
  sum1 = sum(p^2*(1-p)^2)
  sum2 = 0
  n = length(p)
  for(i in 1:(n-1))
     for(j in (i+1):n)
       sum2 = sum2 + 2*p[i]*p[j]*(1-(p[i]+p[j]))^2
  sum1 + sum2
}

# Compute PE from formula
```

```
PE9.eq = 1 - (1 - f(p1)) * (1 - f(p2))

# Check that answers are the same
stopifnot(PE9 == PE9.eq)
```

Here's an indication of an alternative implementation using the parameter `markerindex`:

```
halfSisters = halfSibPed(sex1 = 2, sex2 = 2)
unrelated = list(singleton(4,2), singleton(5,2))

p1 = (1:4)/sum(1:4)
als1 = 1:length(p1)

m = marker(halfSisters,  afreq = p1, alleles = als1, chrom = 23)

halfSisters = addMarkers(halfSisters, m)
m = marker(unrelated[[1]], afreq = p1, alleles = als1)
unrelated[[1]] = addMarkers(unrelated[[1]], m)
m = marker(unrelated[[2]], afreq = p1, alleles = als1)
unrelated[[2]] = addMarkers(unrelated[[2]], m)

PE9.1.alt = exclusionPower(ped_claim = halfSisters, ped_true = unrelated, Xchrom = TRUE,
                     ids = ids, markerindex = 1, plot = FALSE)

stopifnot(PE9.1 == PE9.1.alt)
```

Example 3: Computational challenges

`exclusionPower` may take too long time or hang. The simplest example to demonstrate this is obtained by considering a marker with many alleles, below a marker with 70 alleles, roughly the number of alleles of the most polymorphic of the standard markers,`SE33`. Here's a modified version of the first example of this note. This takes some time as seen below:

```
claim = nuclearPed(nch = 1, sex = 2)
true = list(singleton(id = 1), singleton(id = 3, sex = 2))
ids = c(1, 3)
nAls = 70
system.time(PE1 <- exclusionPower(claim, true, ids = ids, alleles = 1:nAls,
                          afreq = rep(1/nAls, nAls), plot = F))
PE1
```

user system elapsed
2910.42 0.85 4403.26
[1] 0.9440729

For reference, the exact results is calculated from the formula:

```
PE.f = function(p){
  n = length(p)
  sum1 = sum(p^2*(1-p)^2)
  sum2 = 0
  for(i in 1:(n-1))
    for(j in (i+1):n)
      sum2 = sum2 + 2*p[i]*p[j]*(1-p[i]-p[j])^2
  sum1+sum2
}
```

```
nAls = 70
(PE.exact = PE.f(rep(1/nAls,nAls)))
```

```
## [1] 0.9440729
```

We can approximate $PE = P(LR = 0 \mid \text{true})$ by simulation

```
claim = nuclearPed(nch = 1, sex = 2)
true = list(singleton(id = 1), singleton(id = 3, sex = 2))
ids = c(1, 3)
nAls = 70
nsim = 1000
LR1 = rep(NA, nsim)
set.seed (123)
for (s in 1:nsim){
  simTrue = markerSim(true, N = 1, ids = ids, alleles = 1:nAls, afreq = rep(1/nAls, nAls),
                verbose = F)
  simClaim = transferMarkers(simTrue, claim)
  LR1[s] = LR(list(simClaim, simTrue), ref = 2)$LR[1]
}
PE.sim = length(LR1[LR1==0])/nsim
c("PE.exact" = PE.exact, "PE.sim" = PE.sim, "SE" = sqrt(PE.sim*(1-PE.sim)/nsim))
```

PE.exact PE.sim SE
0.944072886 0.948000000 0.007021111

Example 4: Linked markers

The supplementary of Egeland, Pinto, Vigeland (2014) presents an example for linked markers. Here it is, sligthly edited for the `ped-suite`:

In this final section we provide the computations for the example involving linked markers addressed in the discussion section of the paper. The question is the same as for the immigration example above: What is the probability of excluding an older sister as a mother? For simplicity we stick to the no-loop case here.

The marker D12S391 (not used in the paper as it was introduced to forensic applications very recently) is located on the short arm of chromosome 12 only 6.3 megabases from the established vWA, corresponding to a recombination fraction of $\theta = 0.089$. It is therefore of interest to compare the exclusion power for the combination of these two markers with $\theta = 0.089$ to the result obtained by ignoring linkage, i.e., setting $\theta = 0.5$. The calculations are done conditionally on the genotypes of the older sister, which we assume to be as shown in the figure below.

The unlinked case is easily dealt with: We find the PE for each marker separately using `exclusionPower()`, and compute their combined power as in Equation (2) in the main paper:

```
sister = nuclearPed(2, sex = 2)
mother = nuclearPed(1, "mother" = 3, "child" = "4", sex = 2)
a1 = c(16, 17, 99)
frq1 = c(0.218, 0.295, 0.4879)
frq1 = frq1/sum(frq1)
known1 = list(c(3, 16, 17))
PE1 = exclusionPower(mother, sister, ids = c(3,4), alleles = a1,
                     afreq = frq1, known = known1)
```

```
## Removed already typed individuals from internal computations: 3
```

```
a2 = c(17, 18, 99)
frq2 = c(0.129, 0.177, 0.693)
```

```
frq2 = frq2/sum(frq2)
known2 = list(c(3, 17, 18))
PE2 = exclusionPower(mother, sister, ids = c(3,4), alleles = a2,
                     afreq = frq2, known = known2, plot =F)
```

```
## Removed already typed individuals from internal computations: 3
```

According to Equation (2), the combined power of exclusion then becomes:

```
1-(1-PE1)*(1-PE2)
```

```
## [1] 0.1725608
```

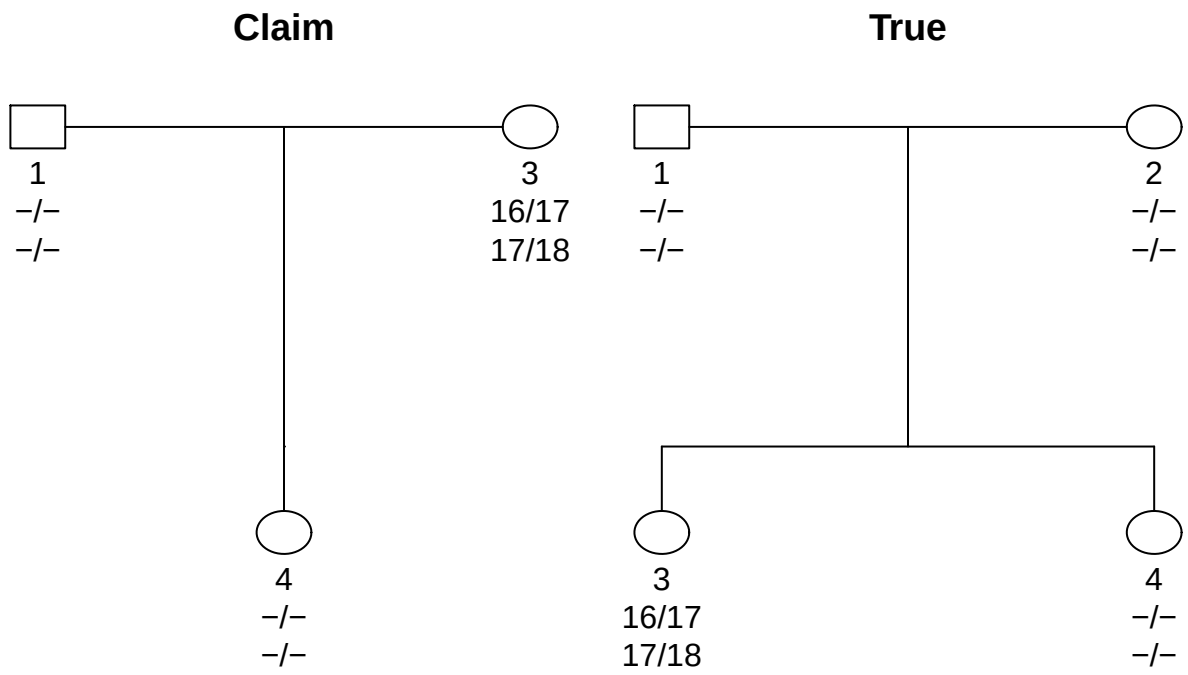
For linked markers there is no "black box" function provided to calculate exclusion powers, so we have to do a little more work ourselves. The formula for PE given in Equation (1) in the paper immediately applies, so the main task is to compute the joint genotype distribution of the markers for each pedigree. This is exactly what the `forrel` function `twoMarkerDistribution()` does. Interested readers are encouraged to study the user manual of `forrel` and online help pages for more info on the commands to follow.

We begin by creating `marker` objects associated to each of the pedigrees, holding the information on alleles, frequencies and known genotypes:

```
D12S391_m = marker(mother, "3" = c(16, 17), alleles = a1, afreq = frq1)
D12S391_s = marker(sister,  "3" = c(16, 17), alleles = a1, afreq = frq1)
vWA_m = marker(mother, "3" = c(17, 18), alleles = a2, afreq = frq2)
vWA_s = marker(sister, "3" = c(17, 18), alleles = a2, afreq = frq2)
```

To check that everything is correct we plot the pedigrees with the marker genotypes:

```
par(mfrow=c(1,2))
plot(mother, marker = list(D12S391_m, vWA_m), title = "Claim")
plot(sister, marker = list(D12S391_s, vWA_s), title = "True")
```



We are now ready for the main computations:

```
library(pedprobr, quietly = TRUE)
A1 = twoMarkerDistribution(mother, 4, D12S391_m, vWA_m, theta = 0.089, verbose = FALSE)
A2 = twoMarkerDistribution(sister, 4, D12S391_s, vWA_s, theta = 0.089, verbose = FALSE)
```

As in Equation (1) in the paper, the PE is then found as

```
I1 = (A1==0)
sum(I1 * A2)
```

```
## [1] 0.1596405
```

We can of course compute the unlinked PE in this way as well, which should give the same result as obtained in the beginning of this section:

```
A1 = twoMarkerDistribution(mother, 4, D12S391_m, vWA_m, theta = 0.5, verbose = FALSE)
A2 = twoMarkerDistribution(sister, 4, D12S391_s, vWA_s, theta = 0.5, verbose = FALSE)
```

```
I1 = (A1==0)
sum(I1 * A2)
```

```
## [1] 0.1725608
```

We conclude that the exclusion power for the two linked markers (0.160) is only slightly smaller than for the unlinked case (0.173).

Example 5: Inbred founders

Below, the father "1" of the true pedigree is inbreed with inbreeding coefficient f. Calculations are checked against formula:

```
claim = nuclearPed(1, father = "4", children = "5")
true = halfSibPed(1)
f = 1/16
founderInbreeding(true, 1) = f
ids = c(4,5)
als = 1:2
q = 0.5
p = c(q, 1-q)
PE = exclusionPower(claim, true, ids = ids, alleles = als,
                    afreq = p, known = NULL, plot = T)
PE.formel = (1-f)*q^2*(1-q)^2
c(PE, PE.formel)
```

```
## [1] 0.05859375 0.05859375
```

Example 6: Mutation

The mother and child are genotyped, but there is a mutation. Exclusion power for the father is required. The answers are checked against formulae.

```
x1 = pedtools::nuclearPed(1, father = "AF", mother = "MO", children = "CH")
als = 1:3
p = c(1,2,3)/6
M1 = matrix(c(0.99, 0.01, 0.00,
              0.01, 0.99, 0.00,
              0.00, 0.00, 1.00),
            ncol = 3, dimnames = list(1:3, 1:3))
mut = pedmut::mutationMatrix(model = "custom", matrix = M1)
m = pedtools::marker(x1, alleles = als, afreq = p, mutmod = mut,
                     "MO" = 1, "CH" = 2)
claim = pedtools::addMarkers(x1, m)
x2 = pedtools::singleton("AF")
x3 = pedtools::nuclearPed(1, father ="TF", mother = "MO", children = "CH")
true = list(x2, x3)
PE1 = forrel::exclusionPower(claim, true, ids = "AF", markerindex = 1,
                             plot = FALSE, verbose = F)
stopifnot(PE1 == p[3]^2)

# If all entries of the mutation matrix are positive,
# exclusion should be impossible:

mut =  pedmut::mutationModel(model = "equal",
            alleles = als, afreq = p, rate = 0.01)
m = pedtools::marker(x1, alleles = als, afreq = p, mutmod = mut,
                     "MO" = 1, "CH" = 2)
claim = pedtools::setMarkers(claim, m)
PE2 = forrel::exclusionPower(claim, true, ids = "AF", markerindex = 1,
                             plot = FALSE, verbose = FALSE)
stopifnot(PE2 == 0)
```

Consider next a strange mutation matrix, making the mother impossible The likelihood of the data is 0 for

`true` and an error message is reported as warranted:

```
M2 = matrix(c(1.00, 0.00, 0.00,
              0.00, 0.99, 0.01,
              0.00, 0.01, 0.99),
            ncol = 3, dimnames = list(1:3, 1:3))
mut = pedmut::mutationMatrix(model = "custom", matrix = M2)
m = pedtools::marker(x1, alleles = als, afreq = p, mutmod = mut,
                     "MO" = 1, "CH" = 2)

claim = pedtools::setMarkers(claim, m)
true = list(x2, x3)
forrel::exclusionPower(claim, true, ids = "AF", markerindex = 2,
                               plot = FALSE)
```