A pedigree-transmission likelihood for multiplex families

This document presents a tutorial describing Tianyu's work in summer 2022 to implement a pedigree-transmission likelihood. This summer project will form the core of her Master's thesis which is tentatively outlined below. Jinko asks Tianyu to remove this outline from the Rmd document and instead post it to a LaTeX thesis document on Overleaf (perhaps place it at the beginning of the draft thesis on Overleaf, to help guide writing).

- Chapter 1: Introduction and background
  - Pedigrees (e.g. what are founders, what are non-founders, what is a multiplex pedigree, medical genetic studies typically ascertain pedigrees to contain multiple affected relatives to increase the chances that the disease has a genetic cause, a rare variant is a variant with frequency less that 1 percent in the population, often only living affected relatives are genotyped for rare variants to save costs).
  - Bayesian networks (e.g. what are they, what are their components, introduce the software we will use to set them up and the features of this software that are relevant to our application.)
  - Casting a pedigree as a Bayesian network
- Chapter 2: A pedigree-transmission likelihood
- Chapter 3: Examples (probably the examples in this tutorial)
- Chapter 4: Conclusions
- Appendix: Link to GitHub repository with this code.

In the code chunk below, we load the packages used for Tianyu's summer project.

```
library(gRain)
library(Rgraphviz)
library(kinship2)
```

The computing in this project relies on Bayesian networks and in particular the `gRain` R package. Explain what `gRain` stands for and how it relates to Bayesian networks. We use `gRain` because it allows us to cast a pedigree as a Bayesian network and more easily calculate transmission-parameter likelihoods. We are interested in transmission-parameter likelihoods to evaluate the evidence that a rare variant (RV) is causally linked to the disease in the pedigree.

1. Getting to know the `gRain` package

We work through a simple example showing how to use `gRain` functions to calculate the probability that the affected individuals in a pedigree received a rare variant (RV) from a carrier founder, given that the RV is transmitted with probability 3/4.

The example pedigree is shown below, drawn with the `pedigree()` function of the `kinship2` package. The first step for plotting is to label each family member with an ID number and indicate their father and mother.

```
id = c(1, 2, 3, 4, 5, 6, 7, 8, 9)
# Label each family member with id number
father = c(0, 0, 1, 0, 1, 0, 3, 5, 5)
# Use id number to indicate the father of each family member
mother = c(0, 0, 2, 0, 2, 0, 4, 6, 6)
# Use id number to indicate the mother of each family member
```

Next indicate the biological sex, affection status and DNA availability of each individual. In our applications, DNA is typically available only from living disease-affected members of the pedigree.

```
sex = c(1, 2, 1, 2, 1, 2, 2, 2, 2)
# 1 represents male, 2 represents female.
affected = c(0, 0, 0, 0, 0, 0, 1, 1, 1)
# affected specifies which family members are affected by the disease.
# 0 and 1 represent individuals who don't and do have the disease, respectively.
avail = c(0, 0, 0, 0, 0, 0, 1, 1, 1)
# avail specifies which family members have DNA available.
# 0 and 1 represent individuals who don't and do have DNA available, respectively.
```

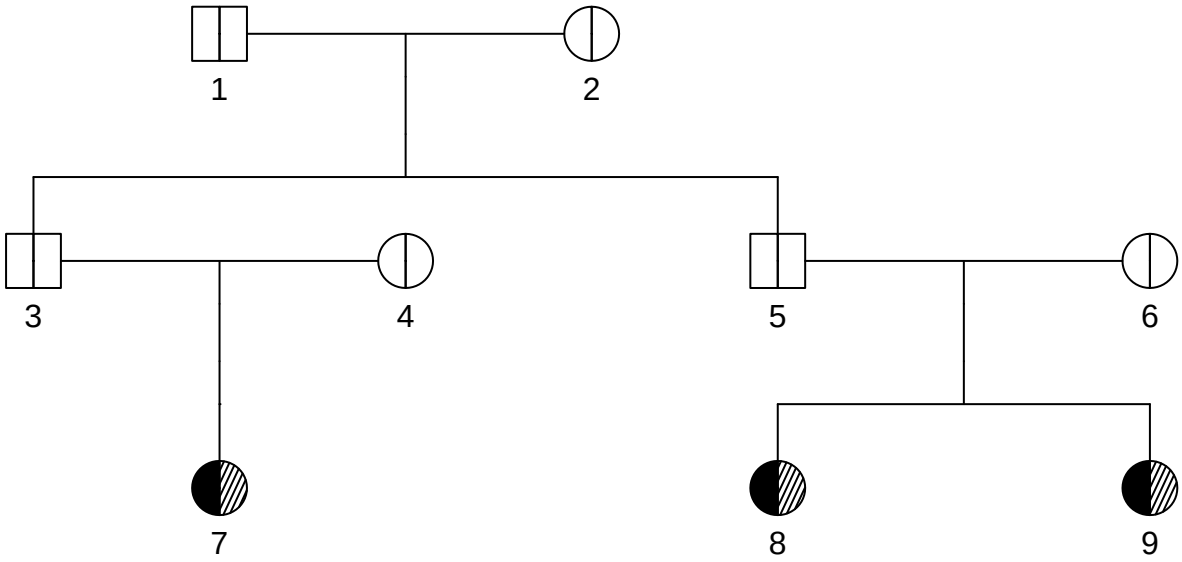
Assemble the information into a data frame for the pedigree and then convert the data frame into a pedigree object.

```
toyPed = as.data.frame(cbind(id,father,mother,sex,avail, affected))

toyPed_ob = pedigree(toyPed$id, toyPed$father, toyPed$mother, toyPed$sex, affected=cbind(toyPed$affecte
# Use pedigree() function from the kinship2 package to construct the pedigree object.
```

Finally, plot the resulting pedigree with affection status marked on the left and DNA availability marked on the right of nodes for individuals.

```
plot(toyPed_ob)
```



From the output above, we can see that ID's 7, 8 and 9 are the only affected members and the only ones with DNA available.

We would like to cast this pedigree as a Bayesian network. A Bayesian network models the joint distribution of a set of variables in terms of their conditional dependencies in a directed graph. A network consists of nodes (variables) and components known as conditional probability tables (CPTs). CPTs specify the local conditional dependencies between variables in the network. When the dependencies among variables are local in nature, it is a lot easier to understand them in terms of CPTs than in terms of the joint distributions of all the variables.

In our context, the variables are the RV status of individuals in a pedigree and the pedigree is the directed graph. The conditional probability tables specify the local dependency of a child's RV genotype given the RV genotypes of their two parents. For pedigrees with no inbreeding, the local dependency structure is straightforward, with children depending only on their parents.

To set up the Bayesian network, we first specify the levels for the RV status of pedigree members as 0, 1 or 2 copies of the RV.

```
copy = c("0", "1", "2")
# number of variants levels: 0, 1, 2
```

Next we consider all $3^3 = 27$ possible configurations of the RV status for the child and its two parents. For each configuration, we specify the conditional probability of the child's RV status given the RV status of the two parents assuming that the transmission probability of the RV from a parent to the child is $\tau = 3/4$. We will use these conditional probabilities to set up our CPTs.

```
# conditional probability of child in the first entry of the
# triplet marked below, given parents in the second and third
# entries of triplet.
child_1 = c(1, 0, 0, 1/4, 3/4, 0, 0, 3/4, 0)
# 000 100 200 010 110 210 020 120 220
child_2 = c(1/4, 3/4, 0, 1/16, 3/8, 9/16, 0, 1/4, 3/4)
# 001 101 201 011 111 211 021 121 221
child_3 = c(0, 1, 0, 0, 1/4, 3/4, 0, 0, 1)
# 002 102 202 012 112 212 022 122 222
child = c(child_1, child_2, child_3)
```

We may now specify the CPTs with the values given in the `child` vector. We start with the pedigree non-founders. These are the individuals who have information on both parents. According to the pedigree diagram above, the non-founders are IDs 3, 5, 7, 8 and 9. The non-founders with IDs 3 and 5 are siblings and have parents with IDs 1 and 2. The non-founder with ID 7 has parents with IDs 3 and 4. Finally, the non-founders with 8 and 9 are siblings and have parents with IDs 5 and 6. In the CPT `formula` argument, the child node comes first after the `~`, followed by the parent nodes, because the child's RV status is conditionally dependent on the parents' RV status.

```
# CPTs for non-founders
ID_3.12 = cptable(~3 + 1 + 2, values = child, levels = copy)
ID_5.12 = cptable(~5 + 1 + 2, values = child, levels = copy)
ID_7.34 = cptable(~7 + 3 + 4, values = child, levels = copy)
ID_8.56 = cptable(~8 + 5 + 6, values = child, levels = copy)
ID_9.56 = cptable(~9 + 5 + 6, values = child, levels = copy)
# Note the order of the variables after the + operator matters.
# The child nodes come first after ~ followed by the parents.
```

We then set up the conditional probability tables for the founders of the pedigree. Since founders have no parents, we specify the marginal probability of their RV genotypes in their CPT tables. Our application does not require these marginal probabilities but the `gRain` package does. We therefore set equal probabilities for the three RV genotypes in founders. According to the pedigree diagram above, the founders are IDs 1, 2, 4 and 6.

```
# Conditional probability tables for founders.
ID_1 = cptable(~1, values = rep(1/3, 3), levels = copy)
ID_2 = cptable(~2, values = rep(1/3, 3), levels = copy)
ID_4 = cptable(~4, values = rep(1/3, 3), levels = copy)
ID_6 = cptable(~6, values = rep(1/3, 3), levels = copy)
# We don't know the values for founders since they don't
# have parents in our pedigree.
```

Next we combine all the CPTs into a list for our Bayesian network:

```
plist_pedigree <-
    compileCPT(list(ID_1, ID_2, ID_3.12, ID_4, ID_5.12,
                    ID_6, ID_7.34, ID_8.56, ID_9.56))
```

From this combined CPT list we can extract the individual CPTs of individuals in the pedigree. For example, we can extract the CPT for the individual with ID 3 as follows.

```
plist_pedigree$`3`
```

```
## , , 2 = 0
##
##    1
## 3   0    1 2
##   0 1 0.25 0
##   1 0 0.75 1
##   2 0 0.00 0
##
## , , 2 = 1
##
##    1
## 3      0      1    2
##   0 0.25 0.0625 0.00
##   1 0.75 0.3750 0.25
##   2 0.00 0.5625 0.75
##
## , , 2 = 2
##
##    1
## 3   0    1 2
##   0 0 0.00 0
##   1 1 0.25 0
##   2 0 0.75 1
```

The output gives the conditional probabilities that ID 3 has 0, 1, or 2 copies of the RV, given the RV genotype status of its parents ID 1 and ID 2.

We are ready to create the network from the list of CPTs.

```
gin1 = grain(plist_pedigree)
summary(gin1)
```

```
## Independence network: Compiled: TRUE Propagated: FALSE
##  Nodes : chr [1:9] "1" "2" "3" "4" "5" "6" "7" "8" "9"
##  Number of cliques:                5
##  Maximal clique size:              3
##  Maximal state space in cliques:   27
```

The `summary()` of the network tells us it has five "cliques", corresponding to the CPT parent-child trios for the five children in the pedigree. The maximal state-space in these cliques is $3^3 = 27$, corresponding to possible RV-status outcomes for the child-parent trios.

We now have a Bayesian network for efficiently calculating joint RV probabilities. Being rare, the RV is assumed to enter the pedigree through a single founder, let's say ID 1. The next code chunk calculates the conditional joint probability that all three of the affected individuals carry the RV given that the founder with ID 1 introduced it into the pedigree. The RV transmission probability is taken to be $\tau = 3/4$. We focus on the affected individuals because in medical-genetic studies they are often the only pedigree members who are genotyped for RV status.

```
#Condition on founder id1 introducing the RV into the pedigree.
bn1 = setEvidence(gin1, nodes = c("1", "2", "4", "6"),
```

```
                  states = c("1", "0", "0", "0") )
```

In Bayesian networks, "evidence" is what is being conditioned on to get the probability. We use the `setEvidence()` function to condition on ID 1 being the founder who introduced the single copy of RV. The other founders IDs 2, 4, and 6 introduce no copies of the RV into the pedigree.

Let's start by getting the marginal distributions of RV status for each of the three affected individuals.

```
affs=c("7", "8", "9")
querygrain(bn1, nodes = affs, type = "marginal")
```

```
## $`7`
## 7
##      0      1      2
## 0.4375 0.5625 0.0000
##
## $`8`
## 8
##      0      1      2
## 0.4375 0.5625 0.0000
##
## $`9`
## 9
##      0      1      2
## 0.4375 0.5625 0.0000
```

Next, we move on to get the joint probability distribution for the three affected individuals.

```
config=c("1","1", "1") #RV configuration for affected members
names(config)=affs #link config to the affected IDs
querygrain(bn1, nodes=affs, type="joint")
```

```
## , , 9 = 0
##
##    8
## 7           0          1 2
##   0 0.1298828 0.06152344 0
##   1 0.1669922 0.07910156 0
##   2 0.0000000 0.00000000 0
##
## , , 9 = 1
##
##    8
## 7            0         1 2
##   0 0.06152344 0.1845703 0
##   1 0.07910156 0.2373047 0
##   2 0.00000000 0.0000000 0
##
## , , 9 = 2
##
##    8
## 7   0 1 2
##   0 0 0 0
##   1 0 0 0
##   2 0 0 0
```

The output lists the conditional joint distribution of the RV status of the affected individuals given that

the founder with ID 1 introduced the RV into the pedigree. This probability is calculated assuming that the transmission probability of the RV is $\tau = 3/4$. We see that the probability that all three of the affected individuals carry a single copy of the RV given that the RV was introduced into the pedigree by the founder with ID 1 is 0.2373. We may also get this result by repeated conditioning, also known as the "chain rule" for Bayesian networks.

```
prob=1
for(n in affs){
# calculate the marginal probability for the node, n
p=unlist(querygrain(bn1, nodes=n))
names(p)=copy
prob=prob*p[config[n]]
# condition on node n's state
bn1 = setEvidence(bn1, nodes=n, states=config[n])
}
prob
```

```
##         1
## 0.2373047
```

The output confirms that the probability that all three of the affected individuals carry a single copy of the RV is 0.2373.

2. Function to create a Bayesian network

In this section, we present a function to create a Bayesian network for our application. The function takes two arguments:

1. A `kinship2` pedigree dataframe, `pedfile`, and
2. A transmission probability of the RV, `tau`.

This function `BNcreate()` is defined in the code chunk below.

```
BNcreate = function(pedfile, tau){
  ## conditional prob of child in the first entry of the triplet
  ## given parents in the second and third entries of triplet with transmission
  ## probability tau
  geno_prob = c(
    # 000 100 200 010 110 210 020 120 220
    1, 0, 0, 1-tau, tau, 0, 0, 1, 0,
    # 001 101 201 011 111 211 021 121 221
    1-tau, tau, 0, (1-tau)^2, 2*tau*(1-tau), tau^2, 0, 1-tau, tau,
    # 002 102 202 012 112 212 022 122 222
    0, 1, 0, 0, 1-tau, tau, 0, 0, 1
  )

   # Construct the CPTs for founders
   # Founders: columns of father or mother are labelled as 0.

   founders = which(pedfile[, "father"] == 0) # the row numbers of founders
   founders_vec = rep(0, length(founders))
   # store the id number of founders (for user configuration)
   founders_cpt = list()
   # store the CPTs of founders in a list, which is denoted as [[i]]
   for (i in 1:length(founders)) {
     id = pedfile[founders[i], "id"]
```

```
    founders_vec[i] = id
    node = cptable(c(id), values = rep(1/3, 3), levels = copy)
    # CPTs of founders
    founders_cpt[i] = list(node)
  }

  # Construct the CPTs for non-founders

  pedfile_c = pedfile[-c(founders), ]
  # Pedigree data after removing founders
  family_ids = list()
  # Store the c(child, father, mother) for debugging
  nonfounders_cpt = list()
  # Store the CPTs for non-founders(for debugging)
  for (i in 1:nrow(pedfile_c)) {
    family_ids[i] = list(c(pedfile_c[i, 'id'], pedfile_c[i, "father"],
                           pedfile_c[i, "mother"]))
    c = pedfile_c[i, 'id'] # id numbers of child
    f = pedfile_c[i, "father"] # id numbers of father
    m = pedfile_c[i, "mother"] # id numbers of mother
    node_nf = cptable(c(c, f, m), values = geno_prob, levels = copy)
    # CPTs for non-founders
    nonfounders_cpt[i] = list(node_nf)
  }

  plist = compileCPT(founders_cpt, nonfounders_cpt)
  gin = grain(plist)
  # Create the Bayesian network from the CPTs of both founders and non-founders

  return(gin)
}
```

We apply the function to the pedigree dataframe from the previous example.

```
gin = BNcreate(toyPed, 3/4)
# Create the Bayesian network from the toy pedigree dataframe using a transmission probability of 3/4.
```

Let's check that we get the same summary as in the previous part of this document, when we created the network for the toy pedigree interactively.

```
summary(gin) #gives same summary as above
```

```
## Independence network: Compiled: TRUE Propagated: FALSE
##  Nodes : chr [1:9] "1" "2" "4" "6" "3" "5" "7" "8" "9"
##  Number of cliques:                5
##  Maximal clique size:              3
##  Maximal state space in cliques:   27
```

3. Construct a function for the transmission likelihood

This next section describes how to calculate the likelihood of a transmission probability in a multiplex pedigree, given data on the RV status of the affected members with available DNA. A multiplex pedigree is a pedigree that has been sampled or "ascertained" to contain several members (e.g. 3 or more) who are affected with the disease of interest. Many family studies in genetics rely on this sampling design.

Our goal is to calculate the likelihood of a value of the RV-transmission probability. For example, above we used a value of $\tau = 3/4$ for the transmission probability in the toy pedigree. As before, we assume that, being rare, the RV enters the pedigree through a single founder only. Therefore, in a pedigree with no inbreeding, the possible values for RV status are 0 (no copies) and 1 (a single copy). We take the prior probability that any founder is the one that introduced the RV to be uniform over all the pedigree founders. The likelihood is obtained from the conditional probability of the RV configuration for affected pedigree members with available DNA, given that exactly one of the founders introduced a single copy of the RV into the pedigree.

The event that exactly one founder introduced the RV is a union of the mutually exclusive events, F_i, that each of the founders, i, introduced the RV. These events are mutually exclusive because the RV is rare so the chance of more than one founder introducing it is effectively zero. The probability of the RV configuration C in the affected pedigree members with available DNA is thus

$$\begin{aligned} P(C|\cup_i F_i) &= \frac{P(C, \cup_i F_i)}{P(\cup_i F_i)} \\ &= \frac{\sum_i P(C, F_i)}{\sum_i P(F_i)} \\ &= \sum_i P(C|F_i) \frac{P(F_i)}{\sum_j P(F_j)}. \end{aligned}$$

Founders are assumed to have the same prior probability of introducing the RV; i.e., $P(F_1) = P(F_2) = P(F_3) = \cdots$. Thus $P(F_i)/\sum_j P(F_j)$ on the right-hand side of the above equation is equal to one over the number of founders. The conditional probability, $P(C|F_i)$, of the RV configuration given that founder i introduced the RV is obtained by "path counting" the number of independent transmissions connecting founder i to the affected members. For example, in the toy pedigree above, an RV from the founder with ID 1 can reach the affected pedigree member with ID 7 through two independent transmissions from ID 1 to ID 3 and then from ID 3 to ID 7, and can reach IDs 8 and 9 through one independent transmission to ID 5 and then two independent transmissions from ID 5 to IDs 8 and 9. Therefore, $P(C = (1,1,1)|F_1) = \tau^{2+1+2}$ and $P(C = (1,1,0)|F_1) = \tau^{2+1+1}(1-\tau)$, for example. In the calculation for $P(C = (1,1,0)|F_1)$, we count two transmissions of the RV from ID 1 to ID 7, one from ID 1 to ID 5, one from ID 5 to ID 8, and one transmission of the other variant in ID 5 (i.e. the non-RV) to ID 9.

The first step in the likelihood function is to call `BNcreate()` to set up a Bayesian network for a given value of the transmission probability. The pedigree-transmission likelihood function takes 3 arguments: `pedfile`, `tau` and `config` described below. The first argument,`pedfile`, is a `kinship2` pedigree object specifying the pedigree structure. The second argument, `tau`, is the transmission probability of RV. The third argument, `config`, is a numeric vector giving the RV status of affected individuals with DNA available. Each element of the vector corresponds to an affected pedigree member with DNA. The elements are ordered by their ID numbering. For example, if the affected individuals with available DNA have ID numbers 1, 3, 6, 7, 14 and `config =c(1, 0, 1, 1, 1)`, IDs 1, 6, 7, 14 carry one copy of the RV and ID 3 carries no copies.

The first step in the likelihood function is to call `BNcreate()` to set up a Bayesian network for a given value of the transmission probability. We assume the RV enters the pedigree on exactly one founder and loop through all the founders to condition on this event. We then calculate $P(\text{config}|\text{founder}i\text{introducedtheRV})$

```
likehd = function(pedfile, tau, config){

  gin = BNcreate(pedfile, tau)
  # Bayesian networks of the specified pedigree with transmission prob tau
  likelihood = 0

  founders = which(pedfile[, "father"] == 0)
  # find the row numbers of founders
  founders = as.character(pedfile$id[founders])

  affs = which(pedfile[, "affected"] == 1&pedfile[,"avail"]==1)
```

```
  # find the row numbers of affected individuals with DNA available
  affs=as.character(pedfile$id[affs])

  # Assume the RV enters pedigree on exactly one founder and
  # loop through all the founders
  for (i in 1:length(founders)) {
    state = rep(0, length(founders))
    # vector state specifies which founder introduces the RV
    state[i] = "1" #1 means carrying the RV.

    bn = setEvidence(gin, nodes = founders, states = state)
    # Set founder i to introduce the rare variant

    # calculate P(config | founder i introduced the rv)
    prob=1
    for(n in affs){
      if(prob > 0) # prevents conditioning on zero prob events
      {
        # calculate probability for this node, n
        p=unlist(querygrain(bn,nodes=n)) #get marginal prob
        names(p)=c("0","1","2")
        prob=prob*p[config[n]]
      # condition on node n's state
        bn = setEvidence(bn, nodes=n, states=config[n])
      }
    }
    likelihood = prob*1/length(founders)+likelihood
  }

  return(likelihood)
}
```

4. Testing the likelihood function

4.1 Toy pedigree

We start by testing the likelihood function on the toy pedigree above. We evaluate the likelihood of $\tau = 3/4$ when all three of the affected pedigree members with available DNA carry a single copy of the RV.

```
config=c("1","1","1")
names(config)=c("7","8","9")
likehd(toyPed, 3/4, config)
```

```
##         1
## 0.1186523
```

From the output above, we see that when the RV configuration is $C = (1, 1, 1)$, the likelihood of $\tau = 3/4$ is

about .1187. This likelihood value complies with our hand calculation from the "path counting" approach:

$$\begin{aligned} P(C=(1,1,1)|\cup_i F_i) &= \frac{1}{4}\left\{[\tau^2\times\tau(\tau^2)] + [\tau^2\times\tau(\tau^2)] + [\tau\times 0] + [\tau^2\times 0]\right\} \\ &= \frac{1}{4}\left\{[\tau^5] + [\tau^5] + [0] + [0]\right\} \\ &= \frac{1}{4}\left\{2\times\left(\frac{3}{4}\right)^5\right\} \\ &= .1187 \end{aligned}$$

Next, we evaluate the likelihood of $\tau = 3/4$ when the RV configuration for IDs 7, 8, and 9 is $C = (1, 1, 0)$

```
config=c("1","1","0")
names(config)=c("7","8","9")
likehd(toyPed, 3/4, config)
```

```
##          1
## 0.03955078
```

This values complies with our hand calculation of $\frac{1}{2}\tau^4(1-\tau) = \frac{1}{2}\left(\frac{3}{4}\right)^4\left(1-\frac{3}{4}\right) = 0.0396$ from the path-counting approach.

4.2 Modified kinship2 pedigree

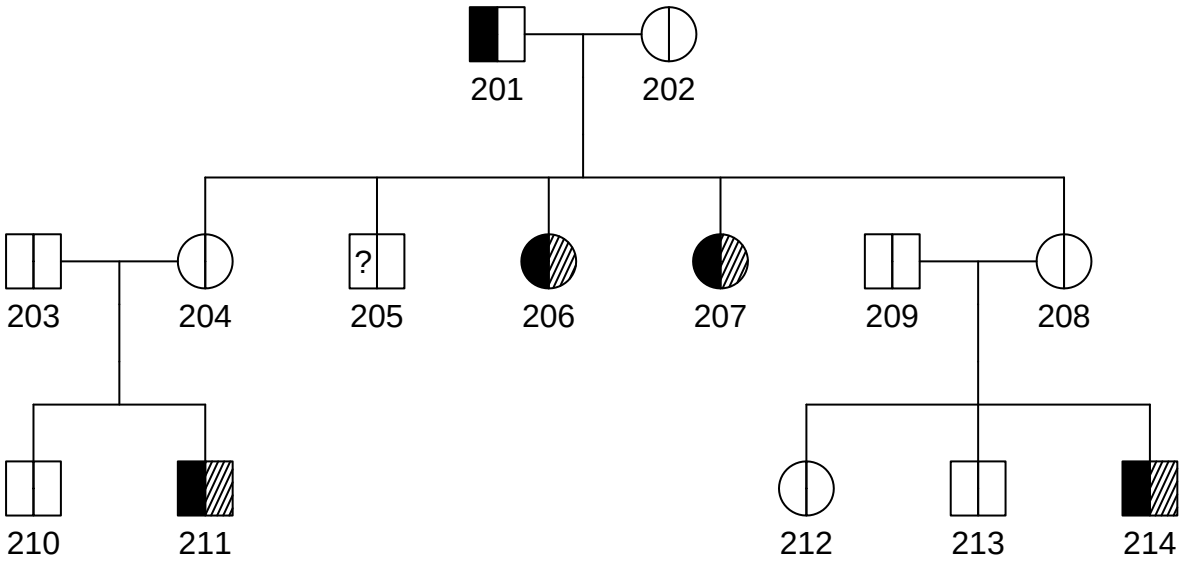
We will use a modified version of the second of the two sample pedigrees in the Kinship2 package.

```
# load pedigree data from Kinship2
data("sample.ped") #two pedigrees
testPed=sample.ped[sample.ped$ped==2,] #2nd pedigree
testPed[testPed$id==203,"affected"]=0
testPed[testPed$id==211,"affected"]=1
testPed[testPed$id==202,"affected"]=0
testPed[testPed$id==201,"avail"]=0
testPed[testPed$id==203,"avail"]=0
testPed[testPed$id==204,"avail"]=0
testPed[testPed$id==212,"avail"]=0
testPed # Unknown affection status has NA
```

```
##    ped  id father mother sex affected avail
## 42   2 201      0      0   1        1     0
## 43   2 202      0      0   2        0     0
## 44   2 203      0      0   1        0     0
## 45   2 204    201    202   2        0     0
## 46   2 205    201    202   1       NA     0
## 47   2 206    201    202   2        1     1
## 48   2 207    201    202   2        1     1
## 49   2 208    201    202   2        0     0
## 50   2 209      0      0   1        0     0
## 51   2 210    203    204   1        0     0
## 52   2 211    203    204   1        1     1
## 53   2 212    209    208   2        0     0
## 54   2 213    209    208   1        0     0
## 55   2 214    209    208   1        1     1
```

From the above data-frame, we see that ID 205 has unknown affection status. This person is marked as "?" in the pedigree plot below.

```
testPed_ob=pedigree(testPed$id, testPed$father,testPed$mother, testPed$sex, affected=cbind(testPed$affe
plot(testPed_ob)
```



The affected family members with DNA available have IDs 206, 207, 211 and 214. Let's evaluate the likelihood of $\tau = 1/2$ for various configurations of the RV status in these individuals. First, suppose that all four of them share a copy of the RV; i.e. $C = (1, 1, 1, 1)$.

```
config1 = c("1", "1", "1", "1")
names(config1)=c("206","207","211", "214")
likehd(testPed, 1/2, config1)
```

```
##         1
## 0.0078125
```

Check that the answer you get accords with path-counting and show your work in tutorial style.

config2 = c("1", "0", "1", "1", "1") names(config2)=c("201","203","206","207","214") likehd(pedfile, 1/2, config2)

Are all these configurations necessary? Maybe pick just one more that is interesting to discuss. config3 = c("0", "1", "1", "0", "1") likehd(pedfile, 1/2, config3)

config4 = c("1", "1", "0", "0", "0") likehd(pedfile, 1/2, config4)

5. Likelihood curves

Get the likelihood curve by plotting the likelihood values vs. tau values for a given configuration of RV status in the affected pedigree members with available DNA.

Show how the curve changes for 2-3 of different configurations in the same pedigree. Discuss what you are doing in tutorial style for the reader.

Add a new chunk by clicking the *Insert Chunk* button on the toolbar or by pressing *Cmd+Option+I*.

References

Include a short references section in this document. You should cite the gRain package, cite the kinship2 package, for example.