Power and sample size computations/simulations in Haplin

Miriam Gjerdevik & Håkon K. Gjessing

2015-10-01

Introduction

Haplin has the ability to compute statistical power and calculate sample size for many of the available analyses. For single-SNP analyses this is easily done with the functions snpPower and snpSampleSize. For haplotype analyses it is done through simulations. First, hapRun is used to perform Haplin runs on simulated haplotype data. The simulation results are then fed to hapPower, which computes the power.

For users who need the actual simulated data, hapSim will perform the simulations without doing the Haplin analyses.

Single-SNP power and sample size

Power computations

snpPower calculates power for a single SNP for a given number of case families, control families, relative risks, minor allele frequencies and type I errors. To compute the power for 300 case children and 200 control children, assuming a relative risk of 1.5, a minor allele frequency of 0.2 and a type I error of 5%, simply type

```
snpPower(cases = list(c=300), controls = list(c=200),
   RR = 1.5, MAF = 0.2, alpha = 0.05)
```

snpPower allows power computations for several family designs. The possible family designs for cases are mfc (full triad), mc (mother-child dyad), fc (father-child dyad) or c (a single case child). Additional to the case designs, the possible family designs for controls also include mf (mother-father dyad), m (a single control mother) and f (a single control father). snpPower enables power calculations for mixtures of the possible case and control designs. The argument cases could, for example, consist of a combination of 1000 full case triads and 500 single case children. This is indicated by cases = list(mfc=1000,c=500). The argument controls = list(mfc=0) implements a family design without any controls.

It is also feasible to compute power for several combinations of the input variables simultaneously (which is the reason for introducing the arguments cases and controls as lists instead of using the much easier data structure of atomic vectors). The command

```
snpPower(cases = list(mfc=c(500,1000),c=c(200,0)),
   controls = list(mfc=1000), RR = c(1.2,1.3), MAF = 0.1)
```

computes the power for the combination of 500 case triads and 200 case children, 1000 control triads and a relative risk of 1.2, and also for the combination of 1000 case triads, 1000 control triads and a relative risk of 1.3. A minor allele frequency of 0.1 and a type I error of 5% are used in both situations. This information is summarized in the output

```
  cases.mfc cases.c controls.mfc  RR MAF alpha     power
1       500     200         1000 1.2 0.1  0.05 0.4804947
2      1000       0         1000 1.3 0.1  0.05 0.9071292
```

Note that it is important to specify the zero in `c=c(200,0)`. If one were to forget this, i.e., typing only `c=200`, `snpPower` will assume 200 case children in both simulations. This should, however, be clear from the Haplin output.

Sample size calculations

`snpSampleSize` is the reverse function of `snpPower`. It computes the number of cases and controls required for a single SNP to attain the desired power for specified family designs and given values of relative risks, minor allele frequencies and type I error. To calculate the number of case-control children required to attain 80% power, assuming an equal number of cases and controls, a relative risk of 1.3, a minor allele frequency of 0.4 and a type I error of 0.05, simply type

```
snpSampleSize(fam.cases = "c", fam.controls = "c",
   fraction = 0.5, RR = 1.3, MAF = 0.4, alpha = 0.05,
   power = 0.80)
```

The possible family designs for cases and controls are the same as for `snpPower`. The argument `fraction` specifies the proportion of case families relative to controls. To calculate the sample size for cases only, i.e., assuming no control families, use `controls = "no_controls"` or `fraction = 1`.

`snpSampleSize` allows for sample size calculations of several combinations of the input variables at once. The command

```
snpSampleSize(fam.cases = "mfc",
   fam.controls = c("mfc","no_controls"),
   fraction = 1/3, RR = 1.5, MAF = 0.1,
   alpha = 0.05, power = c(0.9, 0.8))
```

computes the number of case and control triads required to attain 90% power assuming twice as many controls as cases, and also the number of case triads only required to attain 80% power. A relative risk of 1.5, a minor allele frequency of 0.1 and a type I error of 5% are used in both simulations.

Haplotype power simulations

To compute power in more complex situations, the functions `hapRun` and `hapPower` can be used in combination. `hapRun` simulates haplotype data and performs Haplin runs on the simulated data. `hapPower` calculates the statistical power by applying the simulation results of `hapRun`. The functions enable power calculations of fetal effects, maternal effects and/or parent-of-origin effects. Various family designs, i.e., triads, case-control, the hybrid design, and all intermediate designs, are possible. They also allow power calculation of gene-environment interaction effects and effects on X-chromosome markers.

The command

```
res.hapRun <- hapRun(nall = c(2), cases = c(mfc=500),
   haplo.freq = c(0.25,0.75), RR = c(1.3,1),
   RRstar = c(1,1), hapfunc = "haplin",
   response = "mult", n.sim = 1000)
```

followed by

```
hapPower(res.hapRun)
```

computes the power for 500 case triads on a single SNP, based on 1000 simulations. The least frequent allele is associated with twice the risk and only child effects are assessed. The output looks something like

```
The power was calculated using 1000 of 1000 files
  haplos pv.overall RR.p.value RRdd.p.value
1      1       0.73      0.736        0.736
2      2       0.73        ref          ref
```

The power is computed for three different tests. Here, `pv.overall` gives the likelihood p-value for the overall test assessing the difference between the null model (no effects) and the full model. `RR.p.value` and `RRdd.p.value` give the Wald p-value for testing the effect of a single and a double dose of the least frequent allele, respectively. As a multiplicative dose-response model is used, these two values are equal and will converge to the theoretical power of 74%, attained by applying

```
snpPower(cases = list(mfc=500), controls = list(mfc=0),
  RR = 1.3, MAF = 0.25)
```

Specifying the target effects and arguments to be used by `haplin`

The aimed target effects must be specified in `hapRun`, which performs Haplin runs on the simulated data using the function `haplin` (specified by `hapfunc = "haplin"`). In the example above, `response = "mult"` is explicitly given as an argument to `hapRun` in order to specify a multiplicative dose-response model. If one were to forget this, `haplin` will run with its default value `"free"`, i.e., estimating both single- and double dose effects. Specifying the target effects is further explained by another example. If, for instance, the least frequent allele is associated with twice the risk only when transmitted from the mother,

```
hapRun(nall = c(2), cases = c(mfc=500),
   haplo.freq = c(0.25,0.75), RRcm = c(2,1),
   RRcf = c(1,1), RRstar = c(1,1), poo = T,
   hapfunc = "haplin", response = "mult", n.sim = 1000)
```

will account for this POO effect, as seen in the output given by `hapPower`

```
The power was calculated using 1000 of 1000 files
  haplos pv.overall RRcm.p.value RRcf.p.value
1      1          1            1        0.054
2      2          1          ref          ref

  RRcm_RRcf.p.value RRdd.p.value
1              0.99        0.918
2               ref          ref
```

Here, `RRcm` and `RRcf` must be specified in order to simulate parent-of-origin effects. To test for these simulated parent-of-origin effects, however, one also has to specify `poo = TRUE`. It is thus possible to simulate POO effects without actually testing them. The same is true for maternal effects; `RR.mat` and `RRstar.mat` enables simulations of maternal effects but this effect is not tested unless `maternal = TRUE` . For information on all arguments to be passed on to `haplin`, please consult the help file.

Specifying the design and sample size

`hapRun` enables simulations of various designs through the arguments `cases` and `controls`. The possible family designs for cases, i.e., the possible names of the elements, are `"mfc"` (full triad), `"mc"` (mother-child dyad), `"fc"` (father-child dyad) or `"c"` (a single case child). In addition to the possible options for the case design, the control design also allows for `"mf"` (mother-father dyad), `"m"` (a single control mother) or `"f"` (a single control father).

The arguments specify the number of case and control families, respectively. For instance, `cases=c(mfc=200)` and `controls=c(c=400)` simulate 200 case triads together with 400 control children without any control parents, whereas `cases = c(c=500)` and `controls = c(c=500)` simulate both 500 case and control children. Note that the argument `design` (to be used by `haplin`, `haplinStrat` or `haplinSlide`) does not have to be specified, as `hapRun` understands which design to use.

`hapRun` also allows for simulation and calculations of mixtures of different family designs simultaneously. If, for example, one would like a design of 200 case triads in addition to 100 case mother-child dyads, this can be specified by `cases = c(mfc= 200,mc=100)`. To specify a design without control families, argument `controls` could be left missing. Defining `controls = c(mfc=0)` or `controls = c(c=0)` would, however, also work. The simulated files will then keep the column of the case/control status to the left of the genetic data, but this will have no impact on the power computations.

Simulation of missing data is done through the arguments `gen.missing.cases` and `gen.missing.controls`. By default, both equal `NULL`, which means that no missing data are generated. If the arguments are single numbers, missing data are generated at random with this proportion for all cases and/or controls. If the arguments are vectors of length equal to the number of loci, missing data are generated with the corresponding proportion for each locus. In the two-loci situation, if the second locus contains missing data at the same rate for children, mothers and fathers (both cases and controls), apply a command similar to

```
hapRun(nall = c(2,2), cases = c(mfc=500),
   controls = c(mfc=500), gen.missing.cases = c(0,0.1),
   gen.missing.controls = c(0,0.1), use.missing = "TRUE",
   haplo.freq = c(0.1,0.2,0.3,0.4), RR = c(2,1,1,1),
   RRstar = rep(1,4), hapfunc = "haplin",
   response = "mult", n.sim = 1000)
```

Note that triads with missing data are only included in the Haplin analysis if the argument `use.missing` is set to `TRUE`.

The arguments can also be matrices with the number of rows equal to the number of loci and three columns. Each row corresponds to a locus, and the columns correspond to mothers, fathers and children, respectively. To simulate haplotype data from the hybrid design (case-parent triads and control-parent triads) with control fathers missing at random, use, for instance,

```
hapRun(nall = c(2), cases = c(mfc=500),
   controls = c(mfc=500),
   gen.missing.controls = matrix(c(0,0,0.1),nrow=1),
   use.missing = "TRUE", haplo.freq = c(0.25,0.75),
   RR = c(2,1), RRstar = c(1,1), hapfunc = "haplin",
   response = "mult", n.sim = 1000)
```

Specifying the haplotype relative risks

The number of haplotypes used in the simulations is determined by the argument `nall`, since `prod(nall)` different haplotypes can be made from the specified number of markers, `length(nall)`. The arguments `haplo.freq`, `RR`, `RRcm`, `RRcf`, `RRstar`, `RR.mat` and `RRstar.mat` specifiy the haplotype frequencies and relative risks associated with each haplotype.

The haplotypes are determined by creating all possible haplotypes from the given markers, in a sequence where the first marker varies most quickly. For instance, if `nall = c(3,2)`, the first marker has 3 alleles, the second has 2, and 6 haplotypes are possible. Taken in order, the haplotypes are 1-1, 2-1, 3-1, 1-2, 2-2, and 3-2. When specifying, say, `RR = c(1,2,1,1,1,1)`, the haplotype 2-1 has a double risk compared to the rest. The simplest example would be with `nall = c(2)` and `RR = c(1,2)`, which would simulate a single SNP where the second allele has a double risk.

Explanation of parametrization models

In this section, we will explain the meaning of the arguments RR, RRstar, RR.mat, RRstar.mat, RRcm and RRcf in terms of multiplicative parametrization models. This section is more advanced and only inteded for those with extensive knowledge and special interest in the field.

There are different possible models for how the child- and parental fenotypes might influence the probability of disease. The simplest version is when we assume that the probability of disease depends on the child's genotype only, i.e., information about the mating type is irrelevant when the child's genoype is known. For the K-allele situation, assuming a multiplicative model, the probability of disease conditional on the child's genotype A_jA_l can then be parametrized as

$$P(D|A_jA_l) = \eta R_j R_l R^*_{jl}.$$

We define $R^*_{jl} = R^*_j$ when $j = l$ and $R^*_{jl} = 1$ when $j \neq l$, $j, l = 1, 2, ..., K$. Here, η is a baseline parameter which cannot be estimated due to the sampling design. To facilitate explanations, consider the diallelic situation with alleles A_1 and A_2. The parametrization simplifies to $P(D|A_1A_1) = \eta$, $P(D|A_1A_2) = R\eta$ and $P(D|A_2A_2) = R^2R^*$, where R (argument RR) denotes the relative risk (RR) associated with a single dose of A_2. If there is a multiplicative dose-response of A_2, we would have $P(D|A_2A_2) = R^2$. We thus see that R^* (RRstar) estimates how much double-dose children deviates from the risk expected in a multiplicative dose-response relationship.

Allowing for the possible effect of maternal alleles, we assume that the maternal alleles have a multiplicative effect in addition to the fetal alleles. This results in the model

$$P(D|(A_iA_j, A_kA_l)) = \eta R_j R_l R^*_{jl} M_i M_j M^*_{ij}.$$

The parameters $M_1, M_2, ..., M_K$ (RR.mat) and $M^*_1, M^*_2, ..., M^*_K$ (RRstar.mat) have an interpretation similar to their fetal counterparts. Parent-of-origin effects can be further derived from this model. We choose to assign different effects to the alleles in the child depending on whether they are transmitted from the mother or the father, which gives us

$$P(D|(A_iA_j, A_kA_l)) = \eta R^{(M)}_j R^{(F)}_l R^*_{jl} M_i M_j M^*_{ij}.$$

The only difference from the maternal effects model is that the single-dose effects R_j and R_l are separated into $R^{(M)}_j$ (RRcm) and $R^{(F)}_l$ (RRcf) depending on whether the allele is inherited from the mother or the father. The fraction $R^{(M)}_j / R^{(F)}_l$ is a measure of how much higher or lower the risk associated with allele A_j is, depending on which parent it is transmitted from.

Gene-environment interaction simulations

In order to assess gene-environment interactions, hapfunc must be set to "haplinStrat". The argument nall gives the number of strata. The stratum specific arguments cases, controls , haplo.freq, RR, RRstar, RR.mat, RRstar.mat, `RRcm` and `RRcf` are specified as lists. Each element of a list specifies the argument for one stratum. A simple example is `cases = list(c(mfc=200),c(mfc=500))`, which specifies 200 case triads in the first stratum and 500 case triads in the second. The argument `cases = list(c(mfc=200,c=300),c(c=500))` implements a combination of 200 case triads and 300 case children in stratum one. The second stratum contains 500 control children. In the same manner, `RR = list(c(1,1), c(2,1))` simulates genotype data for a single SNP where there are no associations in the first stratum, whereas the first allele has a double risk in the second stratum.

The list format is only needed for arguments that should vary across strata and can be simplified if else. That is, `cases = c(mfc=500)` in combination with `RR = list(c(1,1), c(2,1))` defines two strata, in which the relative risks vary but the case design and the number of case triads are the same. As an example,

```
hapRun(nall = c(2), n.strata = 2, cases = c(c=250),
   controls = list(c(c=250),c(c=500)),
   haplo.freq = c(0.25,0.75), RR = list(c(1,1),c(2,1)),
   RRstar = c(1,1), hapfunc = "haplinStrat",
   response = "mult", n.sim = 1000)
```

simulates and computes gene-environment effects. Here, the number of control children varies across strata, and the least frequent allele is only associated with disease in the second stratum.