A Genetic Analysis Package with R

Jing Hua Zhao

Department of Public Health and Primary Care, University of Cambridge, Cambridge, UK
https://jinghuazhao.github.io/

Contents
1 Introduction 1
2 Implementation 2
3 Independent programs 11
4 Demos 12
5 Examples 12
  5.1 Pedigree drawing 12
  5.2 Kinship calculation 12
  5.3 Study design 14
  5.4 Graphics examples 19
6 Polygenic modeling 25
7 Known bugs 25
8 Summary 25
9 Bibliographic note 26

1 Introduction

This package was initiated to integrate some C/ Fortran/SAS programs I have written or used over the years. As such, it would rather be a long-term project, but an immediate benefit would be something complementary to other packages currently available from CRAN, e.g. genetics, hwde, etc. I hope eventually this will be part of a bigger effort to fulfill most of the requirements foreseen by many, e.g. Guo and Lange (2000), within
the portable environment of R for data management, analysis, graphics and object-oriented programming. My view has been outlined more formally in Zhao and Tan (2006a) and Zhao and Tan (2006b) in relation to other package systems. Also reported are Zhao (2005) and Zhao (2006) on package \texttt{kinship}.

The number of functions are quite limited and experimental, but I already feel the enormous advantage by shifting to R and would like sooner rather than later to share my work with others. I will not claim this work as exclusively done by me, but would like to invite others to join me and enlarge the collections and improve them.

With my recent work on genomewide association studies (GWAS) especially protein GWAS, I have added many functions such as \texttt{METAL\_forestplot} which handles data from software METAL and \texttt{sentinels} which extracts sentinels from GWAS summary statistics in a way that is very appealing to us compared to some other established software. At the meantime, the size of the package surpasses the limit as imposed by CRAN, thus the old good feature of \texttt{S} as with \texttt{R} that value both code and data alike has to suffer slightly in that \texttt{gap.datasets} and \texttt{gap.examples} are spun off as two separate packages; they do deserve a glimpse however for some general ideas.

\section{Implementation}

The following list shows the data and functions currently available.
* ANALYSIS *

AE3
AE model using nuclear family trios

bt
Bradley-Terry model for contingency table

cccsize
Power and sample size for case-cohort design

cs
Credible set

fbsize
Sample size for family-based linkage and association design

gc.em
Gene counting for haplotype analysis

gcontrol
genomic control

gcontrol2
genomic control based on p values

gcp
Permutation tests using GENECOUNTING

gc.lambda
Estimation of the genomic control inflation statistic (lambda)

genecounting
Gene counting for haplotype analysis

gif
Kinship coefficient and genetic index of familiality

grmMCMC
Mixed modeling with genetic relationship matrices

hap
Haplotype reconstruction

hap.em
Gene counting for haplotype analysis

hap.score
Score statistics for association of traits with haplotypes

htr
Haplotype trend regression

hwe
Hardy-Weinberg equilibrium test for a multiallelic marker

hwe.cc
A likelihood ratio test of population Hardy-Weinberg equilibrium

hwe.hardy
Hardy-Weinberg equilibrium test using MCMC

invnormal
Inverse normal transformation

kin.morgan
kinship matrix for simple pedigree

LD22
LD statistics for two diallelic markers

LDkl
LD statistics for two multiallelic markers

lambda1000
A standardized estimate of the genomic inflation scaling to a study of 1,000 cases and 1,000 controls

log10p
log10(p) for a standard normal deviate

logp
log(p) for a normal deviate

masize
Sample size calculation for mediation analysis

mia
Multiple imputation analysis for hap

mtdt
Transmission/disequilibrium test of a multiallelic marker

mtdt2
Transmission/disequilibrium test of a multiallelic marker by Bradley-Terry model

mvmeta
Multivariate meta-analysis based on generalized least squares

pbsize
Power for population-based association design

pbsize2
Power for case-control association design

pfc
Probability of familial clustering of disease

pfc.sim
Probability of familial clustering of disease

pgc
Preparing weight for GENECOUNTING

print.hap.score
Print a hap.score object

s2k
Statistics for 2 by K table

sentinels
Sentinel identification from GWAS summary statistics

tscc
Power calculation for two-stage case-control design
<table>
<thead>
<tr>
<th>Package</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>asplot</td>
<td>Regional association plot</td>
</tr>
<tr>
<td>ESplot</td>
<td>Effect-size plot</td>
</tr>
<tr>
<td>circos.cnvplot</td>
<td>circos plot of CNVs</td>
</tr>
<tr>
<td>circos.cis.vs.trans.plot</td>
<td>circos plot of cis/trans classification</td>
</tr>
<tr>
<td>circos.mhtplot</td>
<td>circos Manhattan plot with gene annotation</td>
</tr>
<tr>
<td>cnvplot</td>
<td>genomewide plot of CNVs</td>
</tr>
<tr>
<td>METAL_forestplot</td>
<td>forest plot as R/meta’s forest for METAL outputs</td>
</tr>
<tr>
<td>makeRLEplot</td>
<td>make relative log expression plot</td>
</tr>
<tr>
<td>mhtplot</td>
<td>Manhattan plot</td>
</tr>
<tr>
<td>mhtplot2</td>
<td>Manhattan plot with annotations</td>
</tr>
<tr>
<td>mhtplot2d</td>
<td>2D Manhattan plot</td>
</tr>
<tr>
<td>mhtplot.trunc</td>
<td>truncated Manhattan plot</td>
</tr>
<tr>
<td>miamiplot</td>
<td>Miamiplot (Experimental)</td>
</tr>
<tr>
<td>pedtodot</td>
<td>Converting pedigree(s) to dot file(s)</td>
</tr>
<tr>
<td>plot.hap.score</td>
<td>Plot haplotype frequencies versus haplotype score statistics</td>
</tr>
<tr>
<td>qqfun</td>
<td>Quantile-comparison plots</td>
</tr>
<tr>
<td>qquunif</td>
<td>Q-Q plot for uniformly distributed random variable</td>
</tr>
</tbody>
</table>
**UTILITIES**

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SNP</td>
<td>Functions for single nucleotide polymorphisms (SNPs)</td>
</tr>
<tr>
<td>BFDP</td>
<td>Bayesian false-discovery probability</td>
</tr>
<tr>
<td>FPRP</td>
<td>False-positive report probability</td>
</tr>
<tr>
<td>ab</td>
<td>Test/Power calculation for mediating effect</td>
</tr>
<tr>
<td>b2r</td>
<td>Obtain correlation coefficients and their variance-covariances</td>
</tr>
<tr>
<td>chow.test</td>
<td>Chow’s test for heterogeneity in two regressions</td>
</tr>
<tr>
<td>chr.pos_a1_a2</td>
<td>Form SNPID from chromosome, position and alleles</td>
</tr>
<tr>
<td>cis.vs.trans.classification</td>
<td>a cis/trans classifier</td>
</tr>
<tr>
<td>comp.score</td>
<td>score statistics for testing genetic linkage of quantitative traits</td>
</tr>
<tr>
<td>h2</td>
<td>Heritability estimation according to twin correlations for case-control studies</td>
</tr>
<tr>
<td>klem</td>
<td>Haplotype frequency estimation based on a genotype table of two multiallelic markers</td>
</tr>
<tr>
<td>makeped</td>
<td>A function to prepare pedigrees in post-MAKEPED format</td>
</tr>
<tr>
<td>metap</td>
<td>Meta-analysis of p values</td>
</tr>
<tr>
<td>metareg</td>
<td>Fixed and random effects model for meta-analysis</td>
</tr>
<tr>
<td>muvar</td>
<td>Means and variances under 1- and 2- locus (diallelic) QTL model</td>
</tr>
<tr>
<td>read.ms.output</td>
<td>A utility function to read ms output</td>
</tr>
<tr>
<td>snptest_sample</td>
<td>A utility to generate SNPTEST sample file</td>
</tr>
<tr>
<td>twinan90</td>
<td>Classic twin models</td>
</tr>
<tr>
<td>whscore</td>
<td>Whittemore-Halpern scores for allele-sharing</td>
</tr>
<tr>
<td>GRM functions</td>
<td>ReadGRM, ReadGRMBin, ReadGRMPLINK, ReadGRMPCA, WriteGRM</td>
</tr>
<tr>
<td>heritability functions</td>
<td>handle genomic relationship matrix involving other software</td>
</tr>
</tbody>
</table>

Assuming proper installation, you will be able to obtain the list by typing `library(help=gap)` or view the list within a web browser via `help.start()`. Assuming that you have already loaded the package via `library(gap)`, you can use `lsf.str("package:gap")` and `data(package="gap")` to generate a list of functions and a list of datasets, respectively. If this looks odd to you, you might try `search()` within R to examine what is available in your environment before issuing the `lsf.str` command.

```r
[1] ".GlobalEnv" "package:gap" "package:stats"
[10] "package:base"
```

```r
AE3 : function (model, random, data, seed = 1234, n.sim = 50000, verbose = TRUE)
```
BFDP : function (a, b, pi1, W, logscale = FALSE)
Cox.T : function (parms, case, control, k)
Cox.est : function (case, ctl, k0, initial)
DevH0dominant : function (parms, case, control, k)
DevH0dominant.est : function (case, ctl, k0, initial)
DevH0recessive : function (parms, case, control, k)
DevH0recessive.est : function (case, ctl, k0, initial)
DevHaGdominant : function (parms, case, control, k)
DevHaGdominant.est : function (case, ctl, k0, initial)
DevHaGrecessive : function (parms, case, control, k)
DevHaGrecessive.est : function (case, ctl, k0, initial)
ESplot : function (ESdat, SE = TRUE, logscale = TRUE, alpha = 0.05, xlim = c(-2, 8), v = 1, ...)
FPRP : function (a, b, pi0, ORlist, logscale = FALSE)
HapDesign : function (HaploEM)
HapFreqSE : function (HaploEM)
KCC : function (model, GRR, p1, K)
LD22 : function (h, n)
LDk1 : function (n1 = 2, n2 = 2, h, n, optrho = 2, verbose = FALSE)
MCMCgrm : function (model, prior, data, GRM, eps = 0, n.thin = 10, n.burnin = 3000,
                       n.iter = 130000, ...)
METAL_forestplot : function (tbl, all, rsid, pdf = "INF1.fp.pdf", package = "meta",
                              parn : function (p, RRlist)
ReadGRM : function (prefix = 51)
ReadGRMBin : function (prefix, AllN = FALSE, size = 4)
ReadGRMPCA : function (prefix)
ReadGRMPLINK : function (prefix, diag = 1)
VR : function (v1, vv1, v2, vv2, c12)
WriteGRM : function (prefix = 51, id, N, GRM)
WriteGRMBin : function (prefix, grm, N, id, size = 4)
WriteGRMSAS : function (grmlist, outfile = "gwas")
a2g : function (a1, a2)
ab : function (type = "power", n = 25000, a = 0.15, sa = 0.01, b = log(1.19),
         sb = 0.01, alpha = 0.05, fold = 1)
allele.recode : function (a1, a2, miss.val = NA)
asplot : function (locus, map, genes, flanking = 1000, best.pval = NULL, sf = c(4, 4), logpmax = 10, pch = 21)
b2r : function (b, s, rho, n)
b: function (x)
ccsize : function (n, q, pD, p1, alpha, theta, power = NULL, verbose = FALSE)
chow.test : function (y1, x1, y2, x2, x = NULL)
chr_pos_a1_a2 : function (chr, pos, a1, a2, prefix = "chr", seps = c(":", ",", ","),
circos.cis.vs.trans.plot : function (hits, panel, id, radius = 1e+06)
circos.cnvplot : function (data)
circos.mhtplot : function (data, glist)
cis.vs.trans.classification : function (hits, panel, id, radius = 1e+06)
cnvplot : function (data)
comp.score : function (ibddata = "ibd_dist.out", phenotype = "pheno.dat", mean = 0, 
  var = 1, h2 = 0.3)
cov.invlogit : function (logit.p1, logit.p2, cov.logit)
cs : function (tbl, b = "Effect", se = "StdErr", log_p = NULL, cutoff = 0.95)
fbsize : function (gamma, p, alpha = c(1e-04, 1e-08, 1e-08), beta = 0.2, debug = 0, 
  error = 0)
g2a : function (g)
g2a.c : function (g)
gc.control : function (xdata = FALSE, convll = 1, handle.miss = 0, eps = 1e-06, tol 
  maxit = 50, pl = 0.001, assignment = "assign.dat", verbose = T)
gc.em : function (data, locus.label = NA, converge.eps = 1e-06, maxiter = 500, 
  handle.miss = 0, miss.val = 0, control = gc.control())
gc.lambda : function (p)
gcode : function (a1, a2)
gcontrol : function (data, zeta = 1000, kappa = 4, tau2 = 1, epsilon = 0.01, ngib = 
  burn = 50, idum = 2348)
gcontrol2 : function (p, col = palette()[4], lcol = palette()[2], ...) 
cgp : function (y, cc, g, handle.miss = 1, miss.val = 0, n.sim = 0, locus.label = NULL, 
  quietly = FALSE)
genecounting : function (data, weight = NULL, loci = NULL, control = gc.control())
genorecode : function (geno, miss.val = 0)
getPTE : function (b1, b2, rho, sdx1 = 1, sdx2 = 1)
getbistar : function (b1, b2, rho, sdx1 = 1, sdx2 = 1)
gf : function (data, gifset)
grec2g : function (h, n, t)
h2 : function (mzDat = NULL, dzDat = NULL, rmz = NULL, rdz = NULL, nmz = NULL, 
  ndz = NULL, selv = NULL)
h2.jags : function (y, x, G, eps = 1e-04, sigma.p = 0, sigma.r = 1, parms = c("b", 
  "p", "r", "h2"), ...)
h2G : function (V, VCOV, verbose = TRUE)
h2GE : function (V, VCOV, verbose = TRUE)
h2L : function (K = 0.05, P = 0.5, h2, se, verbose = TRUE)
hap : function (id, data, nloci, loci = rep(2, nloci), names = paste("loci", 
  1:nloci, sep = ""), control = hap.control())
hap.control : function (mb = 0, pr = 0, po = 0.001, to = 0.001, th = 1, maxit = 100, 
  n = 0, ss = 0, rs = 0, rp = 0, ro = 0, rv = 0, sd = 0, mm = 0, mi = 0, 
  mc = 50, ds = 0.1, de = 0, q = 0, hapfile = "hap.out", assignfile = "assign.out")
hap.em : function (id, data, locus.label = NA, converge.eps = 1e-06, maxiter = 500, 
  miss.val = 0)
hap.score : function (y, geno, trait.type = "gaussian", offset = NA, x.adj = NA, skip 
  locus.label = NA, miss.val = 0, n.sim = 0, method = "gc", id = NA,
z : function (p1, p2, n1, n2, r)

Package LibPath
[1,] "gap" "/rds/user/jhz22/hpc-work/work/RtmpjyLDK/Rinst443bf15f9fb33"
[2,] "gap" "/rds/user/jhz22/hpc-work/work/RtmpjyLDK/Rinst443bf15f9fb33"
[3,] "gap" "/rds/user/jhz22/hpc-work/work/RtmpjyLDK/Rinst443bf15f9fb33"
[4,] "gap" "/rds/user/jhz22/hpc-work/work/RtmpjyLDK/Rinst443bf15f9fb33"
[5,] "gap" "/rds/user/jhz22/hpc-work/work/RtmpjyLDK/Rinst443bf15f9fb33"
[6,] "gap" "/rds/user/jhz22/hpc-work/work/RtmpjyLDK/Rinst443bf15f9fb33"
[7,] "gap" "/rds/user/jhz22/hpc-work/work/RtmpjyLDK/Rinst443bf15f9fb33"
[8,] "gap" "/rds/user/jhz22/hpc-work/work/RtmpjyLDK/Rinst443bf15f9fb33"
[9,] "gap" "/rds/user/jhz22/hpc-work/work/RtmpjyLDK/Rinst443bf15f9fb33"
[10,] "gap" "/rds/user/jhz22/hpc-work/work/RtmpjyLDK/Rinst443bf15f9fb33"
[11,] "gap" "/rds/user/jhz22/hpc-work/work/RtmpjyLDK/Rinst443bf15f9fb33"
[12,] "gap" "/rds/user/jhz22/hpc-work/work/RtmpjyLDK/Rinst443bf15f9fb33"
[13,] "gap" "/rds/user/jhz22/hpc-work/work/RtmpjyLDK/Rinst443bf15f9fb33"
[14,] "gap" "/rds/user/jhz22/hpc-work/work/RtmpjyLDK/Rinst443bf15f9fb33"
[15,] "gap" "/rds/user/jhz22/hpc-work/work/RtmpjyLDK/Rinst443bf15f9fb33"
[16,] "gap" "/rds/user/jhz22/hpc-work/work/RtmpjyLDK/Rinst443bf15f9fb33"
[17,] "gap" "/rds/user/jhz22/hpc-work/work/RtmpjyLDK/Rinst443bf15f9fb33"
[18,] "gap" "/rds/user/jhz22/hpc-work/work/RtmpjyLDK/Rinst443bf15f9fb33"
[19,] "gap" "/rds/user/jhz22/hpc-work/work/RtmpjyLDK/Rinst443bf15f9fb33"
[20,] "gap" "/rds/user/jhz22/hpc-work/work/RtmpjyLDK/Rinst443bf15f9fb33"

<table>
<thead>
<tr>
<th>Item</th>
<th>Title</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.</td>
<td>&quot;OPGall (OPG)&quot; &quot;Internal functions for gap&quot;</td>
</tr>
<tr>
<td>2.</td>
<td>&quot;OPGrsid (OPG)&quot; &quot;Internal functions for gap&quot;</td>
</tr>
<tr>
<td>3.</td>
<td>&quot;OPGtbl (OPG)&quot; &quot;Internal functions for gap&quot;</td>
</tr>
<tr>
<td>4.</td>
<td>&quot;PD&quot; &quot;Internal functions for gap&quot;</td>
</tr>
<tr>
<td>5.</td>
<td>&quot;aldh2&quot; &quot;Internal functions for gap&quot;</td>
</tr>
<tr>
<td>6.</td>
<td>&quot;apoapoc&quot; &quot;Internal functions for gap&quot;</td>
</tr>
<tr>
<td>7.</td>
<td>&quot;cf&quot; &quot;Internal functions for gap&quot;</td>
</tr>
<tr>
<td>8.</td>
<td>&quot;cnv&quot; &quot;Internal functions for gap&quot;</td>
</tr>
<tr>
<td>9.</td>
<td>&quot;crohn&quot; &quot;Internal functions for gap&quot;</td>
</tr>
<tr>
<td>10.</td>
<td>&quot;fa&quot; &quot;Internal functions for gap&quot;</td>
</tr>
<tr>
<td>11.</td>
<td>&quot;fsnps&quot; &quot;Internal functions for gap&quot;</td>
</tr>
<tr>
<td>12.</td>
<td>&quot;hla&quot; &quot;Internal functions for gap&quot;</td>
</tr>
<tr>
<td>13.</td>
<td>&quot;inf1&quot; &quot;Internal functions for gap&quot;</td>
</tr>
<tr>
<td>14.</td>
<td>&quot;jma.cojo&quot; &quot;Internal functions for gap&quot;</td>
</tr>
<tr>
<td>15.</td>
<td>&quot;l51&quot; &quot;Internal functions for gap&quot;</td>
</tr>
<tr>
<td>16.</td>
<td>&quot;lukas&quot; &quot;Internal functions for gap&quot;</td>
</tr>
<tr>
<td>17.</td>
<td>&quot;mao&quot; &quot;Internal functions for gap&quot;</td>
</tr>
<tr>
<td>18.</td>
<td>&quot;meyer&quot; &quot;Internal functions for gap&quot;</td>
</tr>
<tr>
<td>19.</td>
<td>&quot;mfblong&quot; &quot;Internal functions for gap&quot;</td>
</tr>
<tr>
<td>20.</td>
<td>&quot;nep499&quot; &quot;Internal functions for gap&quot;</td>
</tr>
</tbody>
</table>
A PDF version of this file can be viewed with command `vignette("gap",package="gap")`.

You can cut and paste examples at end of each function’s documentation.

Both `genecounting` and `hap` are able to handle SNPs and multiallelic markers, with the former be flexible enough to include features such as X-linked data and the later being able to handle large number of SNPs. But they are unable to recode allele labels automatically, so functions `gc.em` and `hap.em` are in `haplo.em` format and used by a modified function `hap.score` in association testing.

It is notable that multilocus data are handled differently from that in `hwde` and elegant definitions of basic genetic data can be found in the `genetics` package.

Incidentally, I found my C mixed-radixed sorting routine as in Zhao and Sham (2003) is much faster than R’s internal function.

With exceptions such as function `pfc` which is very computer-intensive, most functions in the package can easily be adapted for analysis of large datasets involving either SNPs or multiallelic markers. Some are utility functions, e.g. `muvar` and `whscore`, which will be part of the other analysis routines in the future.

The benefit with R compared to standalone programs is that for users, all functions have unified format. For developers, it is able to incorporate their C/C++ programs more easily and avoid repetitive work such as preparing own routines for matrix algebra and linear models. Further advantage can be taken from packages in Bioconductor, which are designed and written to deal with large number of genes.

### 3 Independent programs

To facilitate comparisons and individual preferences, The source codes for 2LD, EHPLUS, GENECOUNTING, HAP, now hosted at GitHub, have enjoyed great popularity ahead of the genomewide association studies (GWAS) therefore are likely to be more familiar than their R counterparts in `gap`. However, you need to follow their instructions to compile for a particular computer system.

I have included ms code (which is required by `read.ms.output`) and .xls files to accompany `read.ms.output` and `FPRP` and `BFDP` functions as with
a classic twin example for ACE model in OpenMx. The package is now available from CRAN. For these models it is actually simpler to use facilities as in package mets, which I now suggest.

A final category is twinan90, which is now dropped from the package function list due to difficulty to keep up with the requirements by the R environment but nevertheless you will still be able to compile and use otherwise.

4 Demos
You can also try several simple examples via demo:

```r
library(gap)
demo(gap)
```

5 Examples
I would like to highlight pedtodot pbsize, fbsize and ccsize functions used for pedigree drawing and power/sample calculations in a genome-wide association study as reported in Zhao (2007).

5.1 Pedigree drawing
I have included the original file for the R News as well as put examples in separate vignettes. They can be accessed via vignette("rnews",package="gap.examples") and vignette("pedtodot", package="gap.examples"), respectively.

5.2 Kinship calculation
Next, I will provide an example for kinship calculation using kin.morgan. It is recommended that individuals in a pedigree are ordered so that parents always precede their children. In this regard, package pedigree can be used, and package kinship2 can be used to produce pedigree diagram as with kinship matrix.

Pedigree diagram

```r
> # pedigree diagram
> data(lukas,package="gap.datasets")
> library(kinship2)
> ped <- with(lukas,pedigree(id,father,mother,sex))
```
The pedigree diagram is as follows,

Kinship calculation

We then turn to the kinship calculation.

> # unordered individuals
> library(gap)
We see that in the second case, the result agrees with \texttt{kinship2}.

5.3 Study design

Family-based design

The example involving family-based design is as follows,

```r
> library(gap)
> models <- matrix(c(
+ 4.0, 0.01,
+ 4.0, 0.10,
+ 4.0, 0.50,
+ 4.0, 0.80,
+ 2.0, 0.01,
+ 2.0, 0.10,
+ 2.0, 0.50,
+ 2.0, 0.80,
+ 1.5, 0.01,
+ 1.5, 0.10,
+ 1.5, 0.50,
+ 1.5, 0.80), ncol=2, byrow=TRUE)
```
```r
> outfile <- "fbsize.txt"
> cat("gamma","p","Y","N_asp","P_A","H1","N_tdt","H2","N_asp/tdt","L_o","L_s\n",
+ file=outfile,sep="\t")
> for(i in 1:12) {
+   g <- models[i,1]
+   p <- models[i,2]
+   z <- fbsize(g,p)
+   cat(z$gamma,z$p,z$y,z$n1,z$pA,z$h1,z$n2,z$h2,z$n3,z$lambdao,z$lambda,
+       file=outfile,append=TRUE,sep="\t")
+   cat("\n",file=outfile,append=TRUE)
+ }
> table1 <- read.table(outfile,header=TRUE,sep="\t")
> nc <- c(4,7,9)
> table1[,nc] <- ceiling(table1[,nc])
> dc <- c(3,5,6,8,10,11)
> table1[,dc] <- round(table1[,dc],2)
> unlink(outfile)
> # APOE-4, Scott WK, Pericak-Vance, MA & Haines JL
> # Genetic analysis of complex diseases 1327
> g <- 4.5
> p <- 0.15
> cat("\nAlzheimer's:\n\n")

Alzheimer's:

> fbsize(g,p)

$gamma
[1] 4.5

$p
[1] 0.15

$y
[1] 0.6256916

$n1
[1] 162.6246

$pA
[1] 0.8181818

$h1
[1] 0.4598361
```
Population-based design

The example involving population-based design is as follows,

```
> library(gap)
> kp <- c(0.01, 0.05, 0.10, 0.2)
> models <- matrix(c(
+   4.0, 0.01,
+   4.0, 0.10,
+   4.0, 0.50,
+   4.0, 0.80,
+   2.0, 0.01,
+   2.0, 0.10,
```

16
Case-cohort design

For case-cohort design, we obtain results for ARIC and EPIC studies.

```
> library(gap)
> # ARIC study
> outfile <- "aric.txt"
> n <- 15792
> pD <- 0.03
> p1 <- 0.25
> alpha <- 0.05
```

```

gamma p p1 p5 p10 p20
1 4.0 0.01 46681 8959 4244 1887
2 4.0 0.10 8180 1570 744 331
3 4.0 0.50 10891 2091 991 441
4 4.0 0.80 31473 6041 2862 1272
5 2.0 0.01 403970 77530 36725 16323
6 2.0 0.10 52709 10116 4792 2130
7 2.0 0.50 35285 6772 3208 1426
8 2.0 0.80 79391 15237 7218 3208
9 1.5 0.01 1599920 307056 145448 64644
10 1.5 0.10 192105 36869 17465 7762
11 1.5 0.50 98013 18811 8911 3961
12 1.5 0.80 192105 36869 17465 7762
```
> theta <- c(1.35,1.40,1.45)
> beta1 <- 0.8
> s_nb <- c(1463,722,468)
> cat("n","pD","p1","hr","q","power","ssize\n",file=outfile,sep="\t")
> for(i in 1:3)
+ {
+   q <- s_nb[i]/n
+   power <- ccsize(n,q,pD,p1,alpha,log(theta[i]))
+   ssize <- ccsize(n,q,pD,p1,alpha,log(theta[i]),beta1)
+   cat(n,"\t",pD,"\t",p1,"\t",theta[i],"\t",q,"\t",signif(power,3),"\t",ssize,"\n",
+       file=outfile,append=TRUE)
+ }
> read.table(outfile,header=TRUE,sep="\t")

<table>
<thead>
<tr>
<th>n</th>
<th>pD</th>
<th>p1</th>
<th>hr</th>
<th>q</th>
<th>power</th>
<th>ssize</th>
</tr>
</thead>
<tbody>
<tr>
<td>15792</td>
<td>0.03</td>
<td>0.25</td>
<td>1.35</td>
<td>0.09264184</td>
<td>0.8</td>
<td>1463</td>
</tr>
<tr>
<td>15792</td>
<td>0.03</td>
<td>0.25</td>
<td>1.40</td>
<td>0.04571935</td>
<td>0.8</td>
<td>722</td>
</tr>
<tr>
<td>15792</td>
<td>0.03</td>
<td>0.25</td>
<td>1.45</td>
<td>0.02963526</td>
<td>0.8</td>
<td>468</td>
</tr>
</tbody>
</table>

> unlink(outfile)
> # EPIC study
> outfile <- "epic.txt"
> n <- 25000
> alpha <- 0.00000005
> power <- 0.8
> s_pD <- c(0.3,0.2,0.1,0.05)
> s_p1 <- seq(0.1,0.5,by=0.1)
> s_hr <- seq(1.1,1.4,by=0.1)
> cat("n","pD","p1","hr","alpha","ssize\n",file=outfile,sep="\t")
> # direct calculation
> for(pD in s_pD)
+ {
+   for(p1 in s_p1)
+     {
+       for(hr in s_hr)
+         {
+           ssize <- ccsize(n,q,pD,p1,alpha,log(hr),power)
+           if (ssize>0) cat(n,"\t",pD,"\t",p1,"\t",hr,"\t",alpha,"\t",ssize,"\n",
+             file=outfile,append=TRUE)
+         }
+     }
+ }
> read.table(outfile,header=TRUE,sep="\t")

18
n  pD  p1  hr  alpha  ssize
1 25000 0.3 0.1 1.3 5e-08 14391
2 25000 0.3 0.1 1.4 5e-08 5732
3 25000 0.3 0.2 1.2 5e-08 21529
4 25000 0.3 0.2 1.3 5e-08 5099
5 25000 0.3 0.2 1.4 5e-08 2613
6 25000 0.3 0.3 1.2 5e-08 11095
7 25000 0.3 0.3 1.3 5e-08 3490
8 25000 0.3 0.3 1.4 5e-08 1882
9 25000 0.3 0.4 1.2 5e-08 8596
10 25000 0.3 0.4 1.3 5e-08 2934
11 25000 0.3 0.4 1.4 5e-08 1611
12 25000 0.3 0.5 1.2 5e-08 7995
13 25000 0.3 0.5 1.3 5e-08 2786
14 25000 0.3 0.5 1.4 5e-08 1538
15 25000 0.2 0.1 1.4 5e-08 9277
16 25000 0.2 0.2 1.3 5e-08 7725
17 25000 0.2 0.2 1.4 5e-08 3164
18 25000 0.2 0.3 1.3 5e-08 4548
19 25000 0.2 0.3 1.4 5e-08 2152
20 25000 0.2 0.4 1.2 5e-08 20131
21 25000 0.2 0.4 1.3 5e-08 3648
22 25000 0.2 0.4 1.4 5e-08 1805
23 25000 0.2 0.5 1.2 5e-08 17120
24 25000 0.2 0.5 1.3 5e-08 3422
25 25000 0.2 0.5 1.4 5e-08 1713
26 25000 0.1 0.2 1.4 5e-08 8615
27 25000 0.1 0.3 1.4 5e-08 3776
28 25000 0.1 0.4 1.3 5e-08 13479
29 25000 0.1 0.4 1.4 5e-08 2824
30 25000 0.1 0.5 1.3 5e-08 10837
31 25000 0.1 0.5 1.4 5e-08 2606

> unlink(outfile)

5.4 Graphics examples
I now include some figures from the documentation that may be of interest.

Genome-wide association
The following code is used to obtain a Q-Q plot via \textit{qqunif} function,

\begin{verbatim}
> library(gap)
> pdf("figures/qqunif.pdf",height=10,width=10)
\end{verbatim}
> u_obs <- runif(1000)
> r <- qqunif(u_obs, pch=21, bg="blue", bty="n")
> u_exp <- r$y
> hits <- u_exp >= 2.30103
> points(r$x[hits], u_exp[hits], pch=21, bg="green")
> legend("topleft", sprintf("GC.lambda=\%.4f", gc.lambda(u_obs)))
> dev.off()

null device
1

Based on a chicken genome scan data, the code below generates a Manhattan plot, demonstrating the use of gaps to separate chromosomes.

> library(gap)
> ord <- with(w4, order(chr, pos))
> w4 <- w4[ord,]
> pdf("figures/w4.pdf", height=9, width=10)
> oldpar <- par()
> par(cex=0.6)
> colors <- c(rep(c("blue", "red"), 15), "red")
> mhtplot(w4, control=mht.control(colors=colors, gap=1000), pch=19, srt=0)
> axis(2, cex.axis=2)
The code below obtains a Manhattan plot with gene annotation,

```r
> suggestiveline <- -log10(3.60036E-05)
> genomewideline <- -log10(1.8E-06)
> abline(h=suggestiveline, col="blue")
> abline(h=genomewideline, col="green")
> abline(h=0)
> dev.off()
```

The code below obtains a Manhattan plot with gene annotation,

```r
> library(gap)
> png("figures/mhtplot.png", height=10, width=16, units="cm", res=300)
> data <- with(mhtdata,cbind(chr,pos,p))
> glist <- c("IRS1", "SPRY2", "FTO", "GRIK3", "SNED1", "HTR1A", "MARCH3", "WISP3", "PPP1R3B", "RP1L1", "FDFT1", "SLC39A14", "GFRA1", "NC4R")
> hdata <- subset(mhtdata,gene%in%glist)[c("chr", "pos", "p", "gene")]
> color <- rep(c("lightgray", "gray"),11)
> glen <- length(glist)
> hcolor <- rep("red", glen)
> par(las=2, xpd=TRUE, cex.axis=1.8, cex=0.4)
> ops <- mht.control(colors=color,yline=1.5,xline=3)
> hops <- hmht.control(data=hdata,colors=hcolor)
```
> mhtplot(data, ops, hops, pch=19)
> axis(2, pos=2, at=1:16, cex.axis=0.5)
> title("Manhattan plot with genes highlighted", cex.main=1)
> dev.off()

All these look familiar, so revised form of the function called **mhtplot2** was created for additional features such as centering the chromosome ticks, allowing for more sophisticated coloring schemes, using prespecified fonts, etc. Please refer to the function’s documentation example.

We could also go further with a circos Manhattan plot,

> library(gap)
> library(gap.datasets)
> png("figures/circos-mhtplot.pdf")
> circos.mhtplot()
> dev.off()
The code below obtains a regional association plot with the `asplot` function,

```r
> library(gap)
> library(gap.datasets)
> pdf("figures/asplot.pdf",height=14,width=14)
> asplot(CDKNlocus, CDKNmap, CDKgenes, best.pval=5.4e-8, sf=c(3,6))
> title("CDKN2A/CDKN2B Region")
> dev.off()
```
The function predates the currently popular **locuszoom** software but leaves the option open for generating such plots on the fly and locally.

**Effect size plot**

The code below obtains an effect size plot via the `ESplot` function.

```r
> library(gap)
> pdf("figures/ESplot.pdf",height=10,width=10)
> options(stringsAsFactors=FALSE)
> testdata <- data.frame(models=c("Basic model","Adjusted","Moderately adjusted", + "Heavily adjusted","Other"),
+ OR = c(4.5,3.5,2.5,1.5,1),
+ SElogOR = c(0.2,0.1,0.5,0.5,0.2))
> ESplot(testdata,v=1)
```
Note that all these can serve as templates to customize features of your own.

6 Polygenic modeling

In line with the recent surge of interest in the polygenic models, a separate vignette is available through `vignette("h2", package="gap.examples")` demonstrating aspect of the models on heritability.

7 Known bugs

Unaware of any bug. However, better memory management is expected.

8 Summary

I believe by now the package should have given you a flavour of initiatives I have made so far in relation to how the project was envisaged. More
importantly, it is clear that availability of the package will serve as a platform on which future work can be accumulated and collaboration can be built.

9 Bibliographic note

The main references are Chow (1960); Guo and Thompson (1992); Williams et al. (1992); Gholamic and Thomas (1994); Hartung et al. (2008); Risch and Merikangas (1996); Spielman and Ewens (1996); Risch and Merikangas (1997); Miller (1997); Sham (1997); Elston (1975); Sham (1998); Devlin and Roeder (1999); Zhao et al. (1999); Guo and Lange (2000); Hirotsu et al. (2001); Zhao et al. (2002); Zaykin et al. (2002); Zhao (2004); Wacholder et al. (2004); Wang (2005); Skol et al. (2006); Wakefield (2007).

References


