

CGEN(Case-control.GENetics) Package

April 16, 2015

```
> library(CGEN)
```

Example of snp.logistic

Load the ovarian cancer data and print the first 5 rows.

```
> data(Xdata, package="CGEN")
> Xdata[1:5, ]
```

	id	case.control	BRCA.status	oral.years	n.children	age.group	ethnic.group
1	sub1	0	0	0	1	1	3
2	sub2	1	1	0	2	4	1
3	sub3	0	0	0	2	4	1
4	sub4	1	0	0	3	3	1
5	sub5	1	0	0	3	1	2

	BRCA.history	gynSurgery.history	family.history
1	0	0	0
2	0	0	2
3	0	0	0
4	0	0	0
5	0	0	0

For this analysis, the main effects will be "age.group", "n.children", and "oral.years". We will let "age.group" be a categorical variable in the model and we will create dummy variables for it. The dummy variables will be called "age.group_1", "age.group_2", ... "age.group_5".

```
> for (a in unique(Xdata[, "age.group"])) {
+   dummyVar <- paste("age.group_", a, sep="")
+   Xdata[, dummyVar] <- 0
+   temp <- Xdata[, "age.group"] == a
+   if (any(temp)) Xdata[temp, dummyVar] <- 1
+ }
```

To determine the baseline category, and if any categories need to be combined, get the frequency counts for the age.group variable by case-control status.

```
> table(Xdata[, "case.control"], Xdata[, "age.group"], exclude=NULL)
```

	1	2	3	4	5	<NA>
0	68	137	155	218	169	0
1	31	163	205	240	193	0
<NA>	0	0	0	0	0	0

We will let "age.group_4" will be the reference category, "case.control" be the response variable and "BRCA.status" be the SNP variable. Let the variables "oral.years" and "n.children" also interact with the SNP variable. Also let the stratification variable for the constrained maximum likelihood method (CML) be "ethnic.group".

```
> mainVars <- c("oral.years", "n.children", "age.group_1",
+              "age.group_2", "age.group_3", "age.group_5")
> fit <- snp.logistic(Xdata, "case.control", "BRCA.status",
+                   main.vars=mainVars,
+                   int.vars=c("oral.years", "n.children"),
+                   strata.var="ethnic.group")
```

Compute a summary table for the models.

```
> getSummary(fit)
```

\$UML

	Estimate	Std.Error	Z.value	Pvalue
Intercept	-0.05218922	0.13346145	-0.3910434	6.957651e-01
oral.years	-0.04931037	0.02570532	-1.9182947	5.507366e-02
n.children	-0.03611838	0.02911900	-1.2403715	2.148380e-01
age.group_1	-0.74504465	0.25261026	-2.9493840	3.184081e-03
age.group_2	-0.03278004	0.16379592	-0.2001273	8.413810e-01
age.group_3	0.03711828	0.15484252	0.2397163	8.105502e-01
age.group_5	0.07350744	0.14974476	0.4908849	6.235079e-01
BRCA.status	3.63850273	0.65340234	5.5685487	2.568699e-08
BRCA.status:oral.years	0.05282430	0.10376623	0.5090703	6.107030e-01
BRCA.status:n.children	-0.19683136	0.21145427	-0.9308460	3.519332e-01

\$CML

	Estimate	Std.Error	Z.value	Pvalue
Intercept	-0.075567125	0.13031659	-0.57987342	5.620000e-01
oral.years	-0.055859074	0.02585978	-2.16007523	3.076685e-02
n.children	-0.040560606	0.02957534	-1.37143331	1.702399e-01
age.group_1	-0.861804761	0.24071782	-3.58014520	3.434033e-04
age.group_2	0.100146409	0.15130732	0.66187420	5.080518e-01
age.group_3	0.202837978	0.14268782	1.42155077	1.551567e-01
age.group_5	-0.005926216	0.14186752	-0.04177289	9.666797e-01
BRCA.status	3.337052065	0.32525743	10.25972601	1.070099e-24
BRCA.status:oral.years	0.082378446	0.03077064	2.67717694	7.424542e-03
BRCA.status:n.children	-0.083520091	0.04956159	-1.68517780	9.195427e-02

\$EB

	Estimate	Std.Error	Z.value	Pvalue
Intercept	-0.07487117	0.13037548	-0.5742734	5.657828e-01

oral.years	-0.05545995	0.02666606	-2.0797956	3.754429e-02
n.children	-0.04045957	0.02965025	-1.3645609	1.723911e-01
age.group_1	-0.84125102	0.24363153	-3.4529644	5.544621e-04
age.group_2	0.04736410	0.16101655	0.2941567	7.686382e-01
age.group_3	0.11436147	0.15558658	0.7350342	4.623186e-01
age.group_5	0.01151712	0.14485344	0.0795088	9.366279e-01
BRCA.status	3.38995511	0.41948315	8.0812665	6.409754e-16
BRCA.status:oral.years	0.08016092	0.03787611	2.1163976	3.431100e-02
BRCA.status:n.children	-0.10879888	0.12741046	-0.8539242	3.931470e-01

Compute Wald tests for the main effect of the SNP and interactions.

```
> getWaldTest(fit, c("BRCA.status", "BRCA.status:oral.years", "BRCA.status:n.children"))
```

```
$UML
```

```
$UML$test
```

```
[1] 110.048
```

```
$UML$df
```

```
[1] 3
```

```
$UML$pvalue
```

```
[1] 1.071535e-23
```

```
$CML
```

```
$CML$test
```

```
[1] 122.7287
```

```
$CML$df
```

```
[1] 3
```

```
$CML$pvalue
```

```
[1] 1.993932e-26
```

```
$EB
```

```
$EB$test
```

```
[1] 120.381
```

```
$EB$df
```

```
[1] 3
```

```
$EB$pvalue
```

```
[1] 6.388153e-26
```

Example of snp.scan.logistic

The `snp.scan.logistic` function is used for computing many single SNP analyses. Unlike the `snp.logistic` function where the data must exist in memory

when calling the function, the `snp.scan.logistic` function requires that the data be in separate external files. One file is for the genotype data and the other is for the phenotype data. Both files must contain subject ids in order to link the two together. For this example, the sample genotype data file is delimited by a vertical bar (|) and the first row contains the subject ids. Starting from the second row is the SNP name followed by the genotypes for each subject. Missing genotypes are denoted by "NA". Get the path to the genotype data and define the `snp.list` input argument.

```
> g <- system.file("sampleData", "SNPdata.rda", package="CGEN")
> snp.list <- list(file=g, file.type=1, delimiter="|", in.miss="NA")
```

Define the `pheno.list` argument. Use the ovarian cancer data.

```
> f <- system.file("sampleData", "Xdata.txt", package="CGEN")
> pheno.list <- list(file=f, id.var="id", file.type=3, delimiter="\t",
+                   response.var="case.control", strata.var="ethnic.group",
+                   main.vars=c("oral.years", "n.children"),
+                   int.vars="oral.years")
```

Define the output file and options list.

```
> out.file <- "snp.scan.logistic.output.file_h5jb47j.txt"
> op <- list(out.file=out.file)
```

Call the scan function. Since the "cc.var" option was not specified in `pheno.list`, the minor allele for each SNP will be determined from using all subjects instead of using only the controls, and a note will be printed stating that "cc.var" was not specified in `pheno.list`.

```
> ret <- snp.scan.logistic(snp.list, pheno.list, op=op)
```

```
[1] "For the analysis, 1579 observations will be used."
response0
  0    1
747 832
```

Read in the output file and print the first 5 rows.

```
> x <- read.table(out.file, sep="\t", header=TRUE)
> x[1:5, ]
```

	SNP	Alleles	MAF	CML.omnibus.test	CML.omnibus.pvalue
1	rs11102647	A T	0.4873096	1.0296980	0.5975908
2	rs6695241	G C	0.4971501	0.6640028	0.7174863
3	rs12567796	G C	0.4694561	0.8668416	0.6482876
4	rs2810583	T A	0.4920836	1.9779150	0.3719643
5	rs4654986	T A	0.4967078	1.3799270	0.5015943

	CML.omnibus.df	UML.omnibus.test	UML.omnibus.pvalue	UML.omnibus.df
1	2	0.06016725	0.9703644	2
2	2	1.18504400	0.5529310	2
3	2	0.78915980	0.6739631	2
4	2	1.92497300	0.3819420	2

5	2	0.15666350	0.9246576	2
	EB.omnibus.test	EB.omnibus.pvalue	EB.omnibus.df	
1	0.4990702	0.7791629	2	
2	0.6771836	0.7127734	2	
3	0.8273197	0.6612258	2	
4	1.9494080	0.3773041	2	
5	0.4272128	0.8076662	2	

Create a QQ-plot of the unconstrained p-values.

```
> ret <- QQ.plot(x[, "UML.omnibus.pvalue"])
```

Get the path to the locus map file which contains the chromosome and location of each SNP.

```
> map <- system.file("sampleData", "LocusMapData.txt", package="CGEN")
```

Display the first 5 lines of this file and define the list for the locus map data.

```
> read.table(map, sep="\t", header=1, nrow=5)
```

	SNP	CHROMOSOME	LOCATION
1	rs11102647	1	113783261
2	rs6695241	1	172626514
3	rs12567796	1	18262009
4	rs2810583	1	41549436
5	rs4654986	1	21883504

```
> lmap.list <- list(file=map, file.type=3, header=1, delimiter="\t",
+                  snp.var="SNP", chr.var="CHROMOSOME", loc.var="LOCATION")
```

Define the vector of p-value variables to be plotted and the options list for the chromosome.plot function.

```
> plot.vars <- c("UML.omnibus.pvalue", "CML.omnibus.pvalue", "EB.omnibus.pvalue")
> op <- list(pch=c(21, 22, 23), alt.colors=1)
> #plot.new()
```

Create a chromosome plot of the p-values from all methods.

```
> ret <- chromosome.plot(out.file, plot.vars, lmap.list, op=op)
```

Get the top 10 hits of the scan based on the unconstrained maximum likelihood method.

```
> temp <- sort(x[, "UML.omnibus.pvalue"], index.return=TRUE)$ix
> topHits <- x[temp, ][1:10,]
> topHits
```

	SNP	Alleles	MAF	CML.omnibus.test	CML.omnibus.pvalue
105	rs679218	A T	0.4894870	14.428570	0.000735996
162	rs1899650	A T	0.4846416	11.968640	0.002517930
229	rs5928051	A T	0.4946168	11.629060	0.002983887
83	rs6476055	T A	0.4867004	11.039350	0.004007158
66	rs7800565	G C	0.4987334	5.656796	0.059107480

205	rs2828958	C G	0.4962001	6.181408	0.045469940
131	rs234584	G C	0.4841672	10.408220	0.005493928
111	rs7299271	G C	0.4840426	9.106715	0.010531780
197	rs4254559	G C	0.4851172	11.428090	0.003299305
55	rs6914547	A T	0.4881557	9.547868	0.008447084
	CML.omnibus.df	UML.omnibus.test	UML.omnibus.pvalue	UML.omnibus.df	
105	2	10.199950	0.006096903	2	
162	2	8.490510	0.014332080	2	
229	2	8.268689	0.016013160	2	
83	2	8.089834	0.017511160	2	
66	2	7.506675	0.023439390	2	
205	2	7.151767	0.027990680	2	
131	2	6.939406	0.031126270	2	
111	2	6.670272	0.035609740	2	
197	2	6.484036	0.039084950	2	
55	2	5.998464	0.049825310	2	
	EB.omnibus.test	EB.omnibus.pvalue	EB.omnibus.df		
105	14.612250	0.0006714149	2		
162	11.755490	0.0028010960	2		
229	11.943800	0.0025493950	2		
83	10.919670	0.0042542660	2		
66	6.093153	0.0475213300	2		
205	5.785607	0.0554206100	2		
131	9.596111	0.0082457650	2		
111	8.976304	0.0112414000	2		
197	9.045778	0.0108576100	2		
55	9.642355	0.0080572940	2		

Example of snp.matched

First let us use "age.group1", "gynSurgery.history" and "BRCA.history" to match the subjects finely into small sets. We will perform the matching only within each ethnic group. We check the case control distribution within ethnic groups.

```
> table(Xdata$case.control, Xdata$ethnic.group)
```

```

      1  2  3
0 509 183 55
1 593 193 46
```

Thus, allowing matched sets of size 3 should be enough to match all the subjects in each ethnic group. For illustration, let us use maximum matched set size of 4 for ethnic groups 1 and 2 and that of 3 for ethnic group 3. Let us use daisy to compute the distance matrix, which automatically chooses Gower's distance if there are one or more categorical variables.

```

> library("cluster")
> size <- ifelse(Xdata$ethnic.group == 3, 3, 4)
> d <- daisy(Xdata[,c("age.group_1", "gynSurgery.history", "BRCA.history")])
> mx <- getMatchedSets(d, CC=TRUE, NN=TRUE, ccs.var = Xdata$case.control,
+   strata.var = Xdata$ethnic.group, size = size, fixed = TRUE)
```

The return object `mx` contains vectors corresponding to CC and NN matching as well as corresponding summary matrices `tblCC` and `tblNN`. Summaries can be inspected to see how many matched sets of each size were created (along rows) for each of the ethnic groups (along columns). The strata vectors are then appended to the data.frame, before calling the analysis function `snp.matched`.

```
> mx$CC[1:10]

[1] 355 29 73 72 272 272 30 42 30 42

> mx$tblCC

      strat
      1 2 3
[1,] 86 12 32
[2,]  0  0  0
[3,]  0  0 23
[4,] 254 91  0

> Xdata <- cbind(Xdata, CCStrat = mx$CC, NNStrat = mx$NN)
> Xdata[1:5,]

      id case.control BRCA.status oral.years n.children age.group ethnic.group
1 sub1             0           0         0         1         1             3
2 sub2             1           1         0         2         4             1
3 sub3             0           0         0         2         4             1
4 sub4             1           0         0         3         3             1
5 sub5             1           0         0         3         1             2
      BRCA.history gynSurgery.history family.history age.group_1 age.group_4
1             0             0             0             1             0
2             0             0             2             0             1
3             0             0             0             0             1
4             0             0             0             0             0
5             0             0             0             1             0
      age.group_3 age.group_2 age.group_5 CCStrat NNStrat
1             0             0             0      355      86
2             0             0             0       29      48
3             0             0             0       73      48
4             1             0             0       72      48
5             0             0             0      272      77
```

We will look at the interaction of `BRCA.status` with `oral.years` and `n.children` using formulas.

```
> intVars <- ~ oral.years + n.children
> snpVars <- ~ BRCA.status
> fit <- snp.matched(Xdata, "case.control", snp.vars=snpVars,
+                   main.vars=intVars, int.vars=intVars,
+                   cc.var="CCStrat", nn.var="NNStrat")
```

Compute a summary table for the fitted CLR and CCL models.

```
> getSummary(fit, method = c("CLR", "CCL"))
```

```
$CLR
```

	Estimate	Std.Error	Z.value	Pvalue
BRCA.status	4.16822318	0.78192505	5.3307196	9.782438e-08
oral.years	-0.05347220	0.02682722	-1.9932071	4.623878e-02
n.children	-0.04837260	0.03255916	-1.4856831	1.373630e-01
BRCA.status:oral.years	0.01770353	0.10022474	0.1766383	8.597925e-01
BRCA.status:n.children	-0.25123219	0.24434755	-1.0281756	3.038672e-01

```
$CCL
```

	Estimate	Std.Error	Z.value	Pvalue
BRCA.status	3.50438133	0.39501310	8.8715573	7.213002e-19
oral.years	-0.05745608	0.02668810	-2.1528726	3.132870e-02
n.children	-0.04862828	0.03290015	-1.4780565	1.393927e-01
BRCA.status:oral.years	0.12175915	0.04049286	3.0069293	2.639012e-03
BRCA.status:n.children	-0.03125679	0.06495237	-0.4812263	6.303557e-01

Compute Wald tests for the omnibus effect of BRCA.status for the HCL method.

```
> getWaldTest(fit$HCL, c( "BRCA.status" , "BRCA.status:oral.years" , "BRCA.status:n.childr
```

```
$test
[1] 119.3774
```

```
$df
[1] 3
```

```
$pvalue
[1] 1.050813e-25
```

Session Information

```
> sessionInfo()
```

```
R version 3.2.0 RC (2015-04-08 r68161)
Platform: x86_64-apple-darwin10.8.0 (64-bit)
Running under: OS X 10.6.8 (Snow Leopard)
```

```
locale:
[1] en_US.UTF-8/en_US.UTF-8/en_US.UTF-8/C/en_US.UTF-8/en_US.UTF-8
```

```
attached base packages:
[1] stats      graphics  grDevices  utils      datasets  methods   base
```

```
other attached packages:
[1] cluster_2.0.1  CGEN_3.2.0      mvtnorm_1.0-2   survival_2.38-1
```

```
loaded via a namespace (and not attached):
[1] tools_3.2.0    splines_3.2.0
```