Prepare Marker Sets

Izel Fourie Sørensen, Pernille Merete Sarup, Palle Duun Rohde

May 8, 2017

The following shows how genetic markers (SNPs) can be grouped into sets based on genomic features. Genetic marker sets and be used in set tests or as "features" in GFBLUP prediction models. This example shows how SNPs sets can be defined by sequence ontology, flybase (fb) gene ids, chromosomes and gene ontology terms. As an example we are using the resistance to starvation phenotype available in the *Drosophila melanogaster* Genetic Reference Panel (DGRP) dataset.

The SNPs included in the centered and scaled genotype matrix (**W**), as prepared earlier, are used for preparing the marker sets. Annotations are prepared earlier in the qgg user guide. Load the centered and scaled genotype matrix, `W` (`dgrp2_W2.Rdata`), and the SNP annotation matrix, `snpA` (`snpA_W2.Rdata`).

```
load(file="./genotypes/dgrp2_W2.Rdata")
load(file="./annotation/snpA_W2.Rdata")
```

SNP sets based on sequence ontology

Remove SNPs without annotation. `summary(snpA)` shows that there are NA's for some SNPs.

```
snpA <- na.omit(snpA)
```

Create a list of sequence ontology sets (`seqSets`) by splitting column 3 ("SeqOnt") of the `snpA` data frame, such that each element of the list (i.e. sequence ontology term) contains the relevant SNPs.

```
seqSets <- split( rownames(snpA),f=as.factor( snpA[,3]) )
str(seqSets, vec.len=3)
```

```
## List of 14
##  $ DOWNSTREAM           : chr [1:258237] "2L_10008898" "2L_10008930" "2L_10008944" ...
##  $ EXON                 : chr [1:11020] "2L_10506345" "2L_10506362" "2L_10506377" ...
##  $ INTRON               : chr [1:884290] "2L_10000016" "2L_10000033" "2L_10000089" ...
##  $ NON_SYNONYMOUS_CODING: chr [1:52142] "2L_10000016" "2L_10000089" "2L_10000135" ...
##  $ NON_SYNONYMOUS_START : chr [1:14] "2L_11591491" "2L_7857121" "2R_12186085" ...
##  $ START_GAINED         : chr [1:4090] "2L_10060881" "2L_10060894" "2L_10060983" ...
##  $ START_LOST           : chr [1:35] "2L_12111149" "2L_15895897" "2L_1791822" ...
##  $ STOP_GAINED          : chr [1:196] "2L_10236254" "2L_10363747" "2L_10527805" ...
##  $ STOP_LOST            : chr [1:71] "2L_10013272" "2L_10279121" "2L_11519730" ...
##  $ SYNONYMOUS_CODING    : chr [1:178949] "2L_10000033" "2L_10000487" "2L_10000538" ...
##  $ SYNONYMOUS_STOP      : chr [1:203] "2L_10055872" "2L_1166360" "2L_1199578" ...
##  $ UPSTREAM             : chr [1:258001] "2L_10002466" "2L_10002474" "2L_10002491" ...
##  $ UTR_3_PRIME          : chr [1:49454] "2L_10008023" "2L_10008437" "2L_10008502" ...
##  $ UTR_5_PRIME          : chr [1:24115] "2L_10002418" "2L_10002429" "2L_10003939" ...
```

```
save(seqSets, file="./annotation/seqSets2.Rdata")
```

SNP sets based on flybase gene ids

Create the logicals `inGene`, `inGene500bp` and `inGene1kbp` for SNPs with a distance of 0, 500 and 1000 base pairs (bp) to genes.

```
inGene <- as.numeric(snpA[,4])==0
inGene500bp <- as.numeric(snpA[,4])<=500
inGene1kbp <- as.numeric(snpA[,4])<=1000
```

Create fb gene sets (`fbSets`) by splitting column 1 (FBgn) of the snpA data frame, such that each element of the list (i.e. fb gene) consists of SNPs associated with the flybase gene and are 0 bp from the transcription start site.

```
fbSets <- split( rownames(snpA[inGene,]),f=as.factor(snpA[inGene,1]) )
str(fbSets[1:5], vec.len=3)
```

```
## List of 5
##  $ FBgn0000008: chr [1:853] "2R_18024616" "2R_18024682" "2R_18024761" ...
##  $ FBgn0000014: chr [1:297] "3R_12633508" "3R_12633542" "3R_12633571" ...
##  $ FBgn0000015: chr [1:450] "3R_12753398" "3R_12753600" "3R_12753792" ...
##  $ FBgn0000017: chr [1:286] "3L_16615697" "3L_16615763" "3L_16616747" ...
##  $ FBgn0000018: chr [1:21] "2L_10973623" "2L_10973738" "2L_10973758" ...
```

```
save(fbSets, file="./annotation/fbSets2.Rdata")
```

Use the `split` function as before to group SNPs according fb gene id's. In this case including SNPs 500 (fbSets500bp) and 1000 (fbSets1kbp) bp from the transcription start site.

```
fbSets500bp <- split( rownames(snpA[inGene500bp,]),f=as.factor(snpA[inGene500bp,1]) )
fbSets1kbp <- split( rownames(snpA[inGene1kbp,]),f=as.factor(snpA[inGene1kbp,1]) )

save(fbSets500bp, file="./annotation/fbSets2_500bp.Rdata")
save(fbSets1kbp, file="./annotation/fbSets2_1kbp.Rdata")
```

SNP sets based on chromosomes

Create chromosome sets. With `sapply` split each of the column names (e.g. 2L_10000016) of `W` at the underscore "_". Resulting in e.g. 2L 10000016. The resulting first element of the row name (`[1]` here e.g."2L") is saved in a vector `chr`. `unique(chr)` shows the different chromosomes.

```
chr <- sapply(colnames(W), function(x){ strsplit(x,split="_")[[1]][1]  } )
unique(chr)
```

```
## [1] "2L" "2R" "3L" "3R" "4"  "X"
```

```
chrSets <- unstack( colnames(W), colnames(W) ~ as.factor( chr ))
str(chrSets)
```

```
## List of 6
##  $ 2L: chr [1:406577] "2L_5317" "2L_5372" "2L_5390" "2L_5403" ...
##  $ 2R: chr [1:327967] "2R_10037" "2R_10468" "2R_10959" "2R_12079" ...
##  $ 3L: chr [1:390711] "3L_39998" "3L_40145" "3L_40202" "3L_40635" ...
##  $ 3R: chr [1:368096] "3R_1339" "3R_1651" "3R_2158" "3R_2318" ...
##  $ 4 : chr [1:2686] "4_61790" "4_62622" "4_62905" "4_62908" ...
##  $ X : chr [1:229718] "X_19380" "X_19797" "X_20390" "X_20491" ...
```

```
save(chrSets, file="./annotation/chrSets2.Rdata")
```

SNP sets based on gene ontology (GO) terms

The bioconducter package contains the genome wide annotation for *Drosophila melanogaster*. Annotation is primarily based on mapping using Entrez Gene identifiers. Install the Bioconducter annotation package. For more information about the bioconducter package go to: https://bioconductor.org/packages/release/data/annotation/html/org.Dm.eg.db.html.

```
source("https://bioconductor.org/biocLite.R")
biocLite("org.Dm.eg.db")
```

```
library(org.Dm.eg.db, verbose = FALSE, quietly = TRUE)
```

Load flybase gene ids 0 bp from transcription start site.

```
load(file="./annotation/fbSets2.Rdata")
```

Link GO terms to Entrez Gene ids with the bioconducter package. Information about the gene ontology project can be found here: http://www.geneontology.org/. Since our dataset consists of fb gene ids, the fb gene ids have to be linked to entrez gene ids in order to be linked to GO terms.

Link fb gene ids to Entrez Gene ids.

```
fb2eg <- org.Dm.egFLYBASE2EG
mapped_genes <- mappedkeys(fb2eg)
fb2eg <- as.list(fb2eg[mapped_genes])
str(fb2eg[1:5])
```

```
## List of 5
##  $ FBgn0040373: chr "30970"
##  $ FBgn0040372: chr "30971"
##  $ FBgn0261446: chr "30972"
##  $ FBgn0000316: chr "30973"
##  $ FBgn0005427: chr "30975"
```

Link Entrez Gene ids to fb gene ids.

```
eg2fb <- org.Dm.egFLYBASE
mapped_genes <- mappedkeys(eg2fb)
eg2fb <- as.list(eg2fb[mapped_genes])
str(eg2fb[1:5])
```

```
## List of 5
##  $ 10178776: chr "FBgn0261702"
##  $ 10178777: chr "FBgn0262024"
##  $ 10178779: chr "FBgn0058354"
##  $ 10178780: chr "FBgn0053929"
##  $ 10178781: chr "FBgn0262141"
```

Link GO ids to Entrez Gene ids.

```
go2eg <- as.list(org.Dm.egGO2EG)
str(go2eg[1:5])
```

```
## List of 5
##  $ GO:0000001: Named chr [1:2] "36309" "53567"
##   ..- attr(*, "names")= chr [1:2] "IMP" "IMP"
##  $ GO:0000002: Named chr [1:8] "33567" "34307" "35236" "38046" ...
##   ..- attr(*, "names")= chr [1:8] "IBA" "IMP" "IMP" "IBA" ...
```

```
##  $ GO:0000003: Named chr [1:10] "31309" "31382" "32140" "32435" ...
##   ..- attr(*, "names")= chr [1:10] "NAS" "IMP" "IMP" "IMP" ...
##  $ GO:0000011: Named chr "34110"
##   ..- attr(*, "names")= chr "IBA"
##  $ GO:0000012: Named chr [1:4] "31451" "33530" "41872" "42322"
##   ..- attr(*, "names")= chr [1:4] "IEA" "IBA" "ISS" "ISS"
```

Link fb gene ids to GO ids.

```
go2fb <- lapply(go2eg,function(x){
    fb <- na.omit(unlist(eg2fb[x]))
    m <- match(fb,names(fbSets))
    fb <- fb[!is.na(m)]
    fb <- fb[!duplicated(fb)]
    return(fb)
    })
str(go2fb[1:5], vec.len = 3)
```

```
## List of 5
##  $ GO:0000001: Named chr "FBgn0261618"
##   ..- attr(*, "names")= chr "53567"
##  $ GO:0000002: Named chr [1:7] "FBgn0031540" "FBgn0032154" "FBgn0040268" ...
##   ..- attr(*, "names")= chr [1:7] "33567" "34307" "35236" ...
##  $ GO:0000003: Named chr [1:8] "FBgn0000479" "FBgn0029709" "FBgn0030341" ...
##   ..- attr(*, "names")= chr [1:8] "31309" "31382" "32140" ...
##  $ GO:0000011: Named chr "FBgn0261064"
##   ..- attr(*, "names")= chr "34110"
##  $ GO:0000012: Named chr [1:4] "FBgn0026751" "FBgn0260817" "FBgn0026737" ...
##   ..- attr(*, "names")= chr [1:4] "31451" "33530" "41872" ...
```

Only include GO terms that consist of 5 or more fb genes. Save the resulting GO SNP sets.

```
    go2fb <- go2fb[sapply(go2fb,length)>=5]
    save(go2fb,file="./annotation/go2fb.Rdata")
```

Create GO sets: SNPs linked to GO terms.

```
goSets <- lapply(go2fb,function(x){ unique(unlist(fbSets[x])) })
goSets <- goSets[sapply(goSets,length)>199]
str(goSets[1:5])
```

```
## List of 5
##  $ GO:0000003: chr [1:1999] "X_3070691" "X_3070725" "X_3070853" "X_3070919" ...
##  $ GO:0000022: chr [1:315] "2R_2636861" "2R_2637023" "2R_2637293" "2R_2637711" ...
##  $ GO:0000027: chr [1:568] "X_8608972" "X_8608981" "X_8609121" "X_8609333" ...
##  $ GO:0000028: chr [1:268] "X_1376457" "X_1376829" "X_1376835" "X_9448856" ...
##  $ GO:0000045: chr [1:1116] "X_7208847" "X_7209453" "X_7209470" "X_7209504" ...
```

```
save(goSets,file="./annotation/goSets2.Rdata")
```