Cleaning wheat SNP data

M. Humberto Reyes-Valdés

June 3, 2019

CABANA workshop

Genomic analysis of crop diversity using R

In this tutorial, we'll perform quality control in wheat SNP data downloaded from the CIMMYT platform. The data set comprises hexaploid wheat accessions collected in Mexico, and genotyped by 3,000 SNP loci. Besides the first identifier columns, all of them represent accessions, whereas the rows represent alleles. Each pair of rows belongs to one locus. In the respective cells, the numbers represent allele frequencies, with values either 0, 0.5 or 1. To adjust for heterocigosity and represent allele frequencies, the cells originally coded as 1, 1, were coded as 0.5, 0.5.

Initial preparation

```
#Set working directory
setwd("~/cursos/cabanaIrapuato/lectures")
#Load data
load("data/sample6000GID.bin")
#Inspect objects
ls()
```

```
## [1] "sample6000GID"
```

```
dat<-sample6000GID
rm(sample6000GID)
#Inspect data
dim(dat)
```

```
## [1] 6000 7994
```

```
dat[1:10,1:10]
```

```
##    data_id cluster_ID                 allele_id
## 5        5      80736        1002968|F|0--60:C>T
## 6        6      80736 1002968|F|0--60:C>T-60:C>T
## 7        7     180444        1084711|F|0--49:A>G
## 8        8     180444 1084711|F|0--49:A>G-49:A>G
## 11      11      79420        1063889|F|0--45:T>C
## 12      12      79420 1063889|F|0--45:T>C-45:T>C
## 37      37     110463        1033423|F|0--49:G>A
## 38      38     110463 1033423|F|0--49:G>A-49:G>A
## 69      69       9938        1221834|F|0--55:G>C
## 70      70       9938 1221834|F|0--55:G>C-55:G>C
##                                                                  seq
## 5  TGCAGAGACGGTTCTTCACTGACGAATTCTTTTCTTTCAGAATGATCAGCATTTGTTCCACTGGCGCTG
## 6  TGCAGAGACGGTTCTTCACTGACGAATTCTTTTCTTTCAGAATGATCAGCATTTGTTCCATTGGCGCTG
## 7  TGCAGCAGGGAAGCAGGAATCCCCCTCCAATGTCACTACCCGCCCGCCTACTTCTCCGCAGATCTTTGG
## 8  TGCAGCAGGGAAGCAGGAATCCCCCTCCAATGTCACTACCCGCCCGCCTGCTTCTCCGCAGATCTTTGG
```

```
## 11 TGCAGATAAGTAGTACAATGATATTGTTTCAAGAATCTTTTATAGTATCATTCTGTTCATGTACCTGCC
## 12 TGCAGATAAGTAGTACAATGATATTGTTTCAAGAATCTTTTATAGCATCATTCTGTTCATGTACCTGCC
## 37 TGCAGATTCATTGAAGAAAAGTGTGAGGCACACACAGCCTAAGAGCACGGTTTTCCCTTGCTCACAAGG
## 38 TGCAGATTCATTGAAGAAAAGTGTGAGGCACACACAGCCTAAGAGCACGATTTTCCCTTGCTCACAAGG
## 69 TGCAGAAGCTCATCACCGAATCCACCACATCTCCCGATGACGATGGCGGCAAGCCGTGCACCGTGCAGC
## 70 TGCAGAAGCTCATCACCGAATCCACCACATCTCCCGATGACGATGGCGGCAAGCCCTGCACCGTGCAGC
##    SNP_pos val_alel marker allele SEEDDIV2819 SEEDDIV2891
## 5       60        C      3      1           0         1.0
## 6       60        T      3      2           1         0.0
## 7       49        A      4      1           1         1.0
## 8       49        G      4      2           0         0.0
## 11      45        T      6      1           1         0.5
## 12      45        C      6      2           0         0.5
## 37      49        G     19      1          NA         1.0
## 38      49        A     19      2          NA         0.0
## 69      55        G     35      1           0          NA
## 70      55        C     35      2           1          NA
```

Trim to have only marker and allele as indicator columns

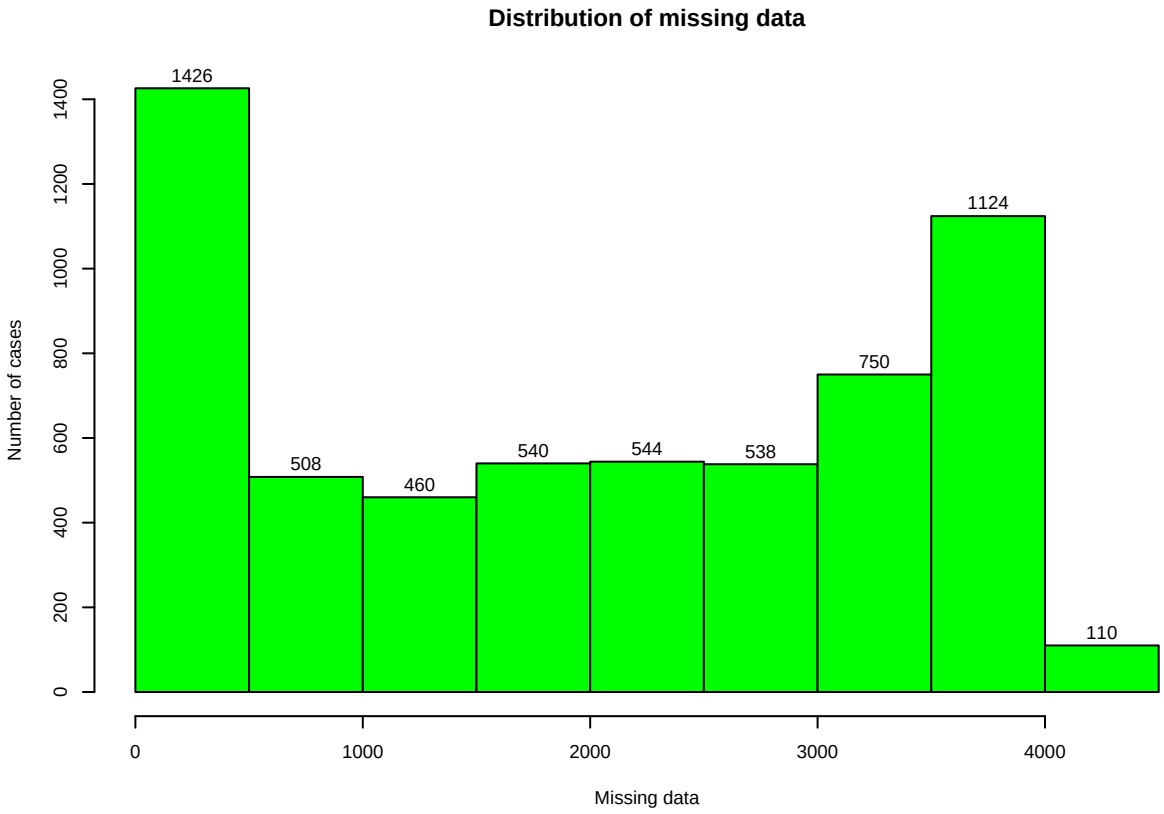
```
dat<-dat[-c(1:6)]
dim(dat)
```

```
## [1] 6000 7988
```

```
row.names(dat)<-c(1:dim(dat)[1])
```

One issue with DNA markers is missing data. Let's inspect the incidence of missing data.

```
#How many missing data points?
#Create a function to count NAs in a vector
counter<-function(x)sum(is.na(x))
#Check the distribution of missing data points among alleles.
nas<-apply(dat,1,counter)
par(cex=.6)
hist(nas, labels=T, col="green",
xlab="Missing data", ylab="Number of cases", main="Distribution of missing data")
```



Now, we decide to select the 20% of markers with less NAs.

```
#Range of the number of NAs
range(nas)
```

```
## [1]    0 4251
```

```
#Remember that each locus comprise two rows
#we must eliminate complete loci
#Check the first allele of each marker
#Odd rows
x<-seq(1,length(nas),2)
head(x)
```

```
## [1]  1  3  5  7  9 11
```

```
#NAs in odd rows
nas1<-nas[x]
head(nas1)
```

```
##   1   3   5   7   9  11
##  80 379  14  91 217 449
```

```
#Select the cleanest 20%
selec<-x[nas1<=quantile(nas1,.2)]
head(selec)
```

```
## [1]  1  5  7  9 13 15
```

```
#Include both alleles
selec<-c(selec,selec+1) #Both alleles for each locus
#Sort
selec<-sort(selec)
length(selec)
```

```
## [1] 1202
```

```
#Select the subset
dat<-dat[selec,]
dim(dat)
```

```
## [1] 1202 7988
```

Selection of informative loci

Not all loci are informative for several applications like GWAS. Thus, we'll select only polymorphic loci. Although we can use one of the diversity measures discussed so far, we'll use a trick: calculate row variances. Any row with variance = 0, will represent an uninformative allele, because its frequency is the same among all observations.

```
#Removing noninformative loci
#Check row one
var(as.numeric(dat[1,-c(1,2)]),na.rm=T)
```

```
## [1] 0.2456323
```

```
#Function to check a row
check.i<-function(x)var(as.numeric(x),na.rm=T)
#Apply to rows (alleles)
checker<-apply(dat[-c(1,2)],1,check.i)
#Number of noninformative alleles
sum(checker==0)
```

```
## [1] 100
```

```
#Remove noninformative alleles
dat<-dat[checker>0,]
dim(dat)
```

```
## [1] 1102 7988
```

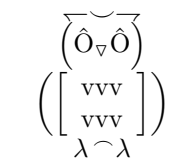
Write clean data frame

```
write.csv(dat,file="tables/CleanWheat.csv",row.names=F) #7986 lines with 1102 alleles. Low NA.
```

Write data for CoreHunter

CoreHunter is a suite of applications for core subsets. Missing data are represented by blank cells.

```
dat<-apply(dat,2, function(x) ifelse(is.na(x),"",x))
write.csv(dat,"tables/CleanWheatBlank.csv",row.names=F) #7986 lines with 1102 alleles. Low NA.
```



Humberto Reyes