Population structure

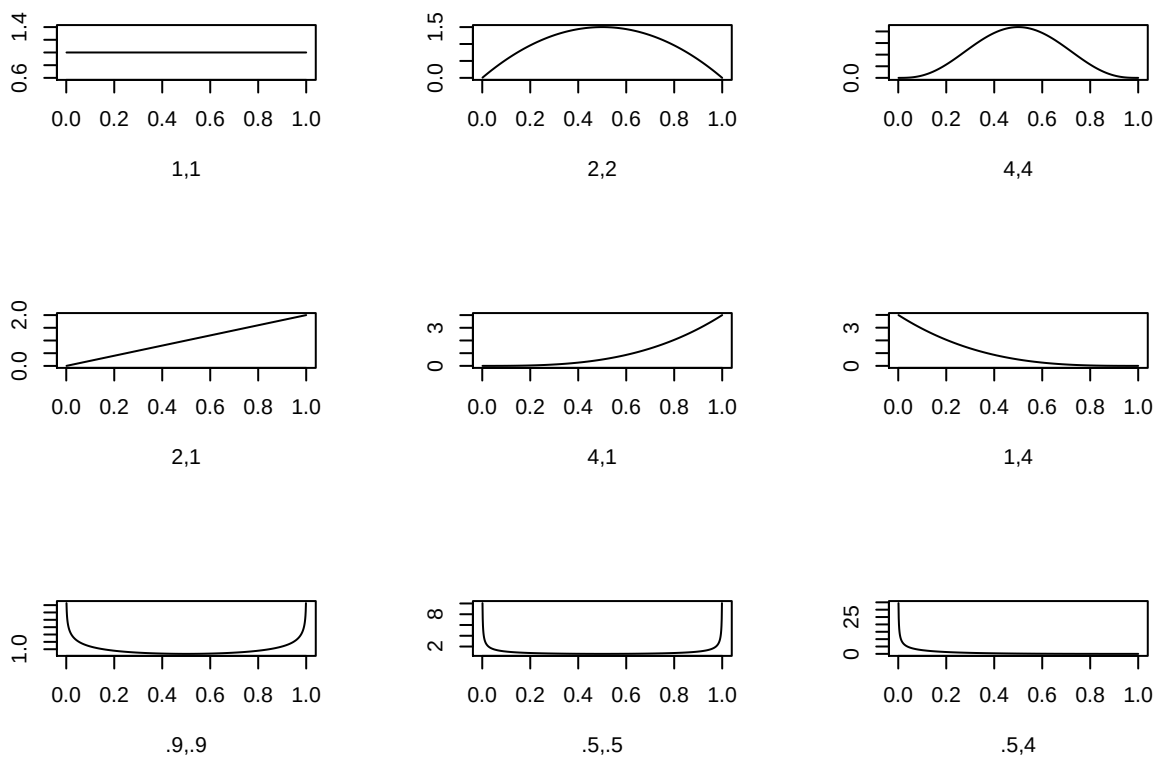
Jerome Goudet

2015-07-14

Contents

reproduces slide 146 and illustrates the different shapes a beta distribution can take:

```
x<-1:1000/1000
par(mfrow=c(3,3))
plot(x,dbeta(x,1,1),xlab="1,1",ylab="",type="l")
plot(x,dbeta(x,2,2),xlab="2,2",ylab="",type="l")
plot(x,dbeta(x,4,4),xlab="4,4",ylab="",type="l")
plot(x,dbeta(x,2,1),xlab="2,1",ylab="",type="l")
plot(x,dbeta(x,4,1),xlab="4,1",ylab="",type="l")
plot(x,dbeta(x,1,4),xlab="1,4",ylab="",type="l")
plot(x,dbeta(x,.9,.9),xlab=".9,.9",ylab="",type="l")
plot(x,dbeta(x,.5,.5),xlab=".5,.5",ylab="",type="l")
plot(x,dbeta(x,.5,4),xlab=".5,4",ylab="",type="l")
```



Now load the `hierfstat` library and the FBI dataset, and do a few initial calculations

```r
library(hierfstat)
#points to the folder with dataset
setwd("c:/Users/jgoudet/Dropbox/Teaching/SISG2015/")
fbiwh02<-read.table("./fbiwh02.txt",header=TRUE)
head(fbiwh02)
```

```
##   Pop D3S1358 vWA  FGA D8S1179 D21S11 D18S51 D5S818 D13S317 D7S820 CSF1PO
## 1   1     609 405 1623     304    508    514    505     303    103   1003
## 2   1     506 505 1618     508    506    816    905     203    103    105
## 3   1     606 506 1821     204    811   1013    305     202   1105   1004
## 4   1     808 607 1113     205    405   1013    305     806   1003    105
## 5   1     505 303  923     306    606    810    304     204    103    103
## 6   1     808 607 1121     505   1112   1010    304     303    107    304
##   TPOX THO1 D16S539
## 1  607  808     804
## 2  702  304     204
## 3  802  305     802
## 4  708  408     405
## 5  708  801     801
## 6  507  407     804
```

```r
###some exploration of the dataset
bs.fbi<-basic.stats(fbiwh02)
names(bs.fbi)
```

```
## [1] "n.ind.samp" "pop.freq"   "Ho"         "Hs"         "Fis"
## [6] "perloc"     "overall"
```

```r
bs.fbi$n.ind.samp # nb inds per pop
```

```
##           1   2   3
## D3S1358 210 203 209
## vWA     180 196 203
## FGA     180 196 203
## D8S1179 180 196 203
## D21S11  179 196 203
## D18S51  180 196 203
## D5S818  180 195 203
## D13S317 179 196 203
## D7S820  210 203 209
## CSF1PO  210 203 209
## TPOX    209 203 209
## THO1    210 203 209
## D16S539 209 202 208
```

```r
bs.fbi$Hs #gene diversities
```

```
##              1      2      3
## D3S1358 0.7651 0.7966 0.7203
## vWA     0.8111 0.8130 0.7707
## FGA     0.8656 0.8619 0.8797
```

```
## D8S1179 0.7802 0.7986 0.7943
## D21S11  0.8629 0.8556 0.8129
## D18S51  0.8751 0.8782 0.8767
## D5S818  0.7408 0.6835 0.7195
## D13S317 0.6903 0.7735 0.8294
## D7S820  0.7833 0.8076 0.7738
## CSF1PO  0.7829 0.7356 0.7086
## TPOX    0.7653 0.6227 0.6081
## THO1    0.7290 0.7847 0.7585
## D16S539 0.8003 0.7693 0.7731
```

```
bs.fbi$overall #different sum stats
```

```
##      Ho      Hs      Ht     Dst     Htp    Dstp     Fst    Fstp     Fis
##  0.7853  0.7810  0.7914  0.0104  0.7966  0.0156  0.0131  0.0196 -0.0056
##    Dest
##  0.0712
```

```
wcfbi<-wc(fbiwh02)
names(wcfbi)
```

```
## [1] "call"      "sigma"     "sigma.loc" "per.al"    "per.loc"   "FST"
## [7] "FIS"
```

```
wcfbi$FST
```

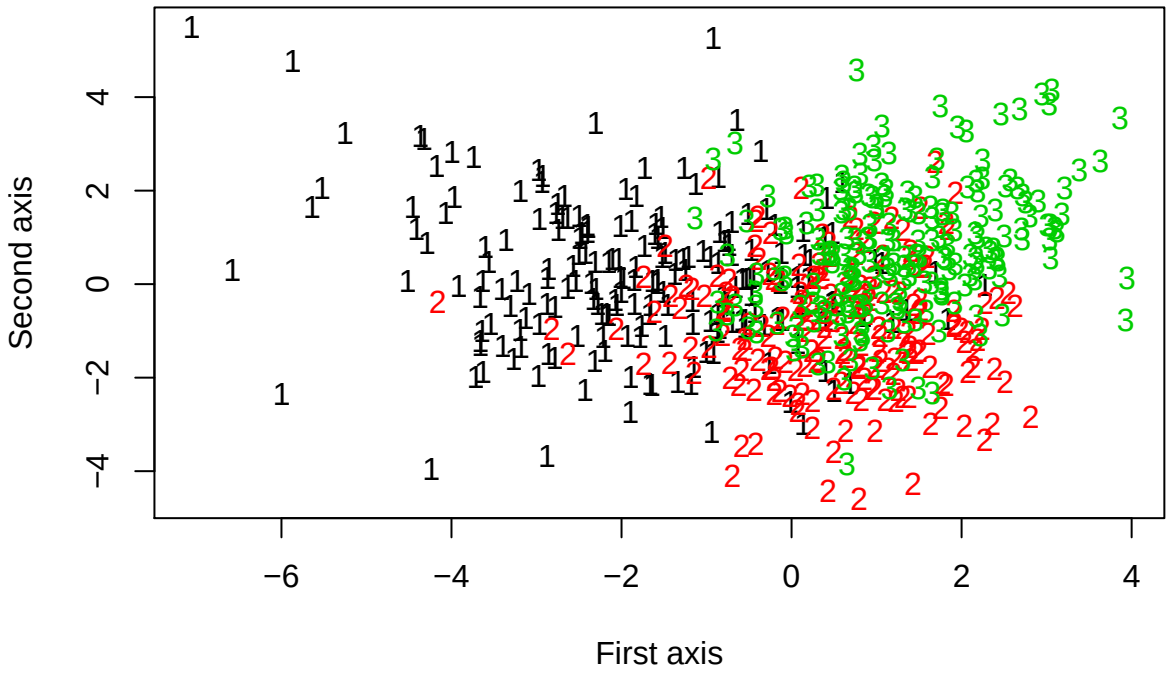
```
## [1] 0.01945741
```

```
wcfbi$FIS
```

```
## [1] -0.005266951
```

Individual pca of the fbi dataset

```
fbi.ipca<-indpca(fbiwh02)
plot(fbi.ipca,col=fbiwh02[,1])
```



Estimation of the β's, slides 151 and followings

```
(bfbi<-betas(fbiwh02))
```

```
## $betaiovl
## [1] 0.009955292 0.016869419 0.031853108
##
## $Hi
##           [,1]      [,2]      [,3]      [,4]      [,5]      [,6]      [,7]
## [1,] 0.7651665 0.8112968 0.8655834 0.7801764 0.8629955 0.8751006 0.7408852
## [2,] 0.7966551 0.8129339 0.8619709 0.7986847 0.8556162 0.8782035 0.6834882
## [3,] 0.7203309 0.7706988 0.8797056 0.7943928 0.8129174 0.8766892 0.7194916
##           [,8]      [,9]     [,10]     [,11]     [,12]     [,13]
## [1,] 0.6902806 0.7833958 0.7829185 0.7653093 0.7289578 0.8003396
## [2,] 0.7733441 0.8075169 0.7356808 0.6227817 0.7845527 0.7693782
## [3,] 0.8293620 0.7737886 0.7085929 0.6080227 0.7585855 0.7730653
##
## $Hb
##  [1] 0.7753621 0.8136889 0.8743065 0.8037197 0.8559500 0.8880990 0.7303376
##  [8] 0.7963372 0.7967187 0.7484664 0.6858119 0.7929260 0.7937741
```

```
names(bfbi)
```

```
## [1] "betaiovl" "Hi"       "Hb"
```

```
#now looking at pairs of populations
(bfbi12<-betas(fbiwh02[fbiwh02[,1]!=3,]))
```

```
## $betaiovl
## [1] 0.01703520 0.02389988
##
## $Hi
##           [,1]      [,2]      [,3]      [,4]      [,5]      [,6]      [,7]
## [1,] 0.7651665 0.8112968 0.8655834 0.7801764 0.8629955 0.8751006 0.7408852
## [2,] 0.7966551 0.8129339 0.8619709 0.7986847 0.8556162 0.8782035 0.6834882
##           [,8]      [,9]     [,10]     [,11]     [,12]     [,13]
## [1,] 0.6902806 0.7833958 0.7829185 0.7653093 0.7289578 0.8003396
## [2,] 0.7733441 0.8075169 0.7356808 0.6227817 0.7845527 0.7693782
##
## $Hb
##  [1] 0.7889749 0.8276219 0.8688988 0.8128472 0.8621950 0.8913124 0.7277564
##  [8] 0.7517244 0.7957835 0.7668719 0.7155703 0.8199097 0.8006182
```

```
(bfbi13<-betas(fbiwh02[fbiwh02[,1]!=2,]))
```

```
## $betaiovl
## [1] 0.01550016 0.03727533
##
## $Hi
##           [,1]      [,2]      [,3]      [,4]      [,5]      [,6]      [,7]
## [1,] 0.7651665 0.8112968 0.8655834 0.7801764 0.8629955 0.8751006 0.7408852
## [2,] 0.7203309 0.7706988 0.8797056 0.7943928 0.8129174 0.8766892 0.7194916
##           [,8]      [,9]     [,10]     [,11]     [,12]     [,13]
## [1,] 0.6902806 0.7833958 0.7829185 0.7653093 0.7289578 0.8003396
## [2,] 0.8293620 0.7737886 0.7085929 0.6080227 0.7585855 0.7730653
##
## $Hb
##  [1] 0.7551094 0.8083402 0.8759168 0.8020731 0.8626056 0.8950944 0.7578818
##  [8] 0.8142940 0.7932673 0.7551435 0.7222248 0.7706083 0.8012629
```

```
(bfbi23<-betas(fbiwh02[fbiwh02[,1]!=1,]))
```

```
## $betaiovl
## [1] 0.004087104 0.019265606
##
## $Hi
##           [,1]      [,2]      [,3]      [,4]      [,5]      [,6]      [,7]
## [1,] 0.7966551 0.8129339 0.8619709 0.7986847 0.8556162 0.8782035 0.6834882
## [2,] 0.7203309 0.7706988 0.8797056 0.7943928 0.8129174 0.8766892 0.7194916
##           [,8]      [,9]     [,10]     [,11]     [,12]     [,13]
## [1,] 0.7733441 0.8075169 0.7356808 0.6227817 0.7845527 0.7693782
## [2,] 0.8293620 0.7737886 0.7085929 0.6080227 0.7585855 0.7730653
##
## $Hb
##  [1] 0.7820020 0.8051046 0.8781040 0.7962388 0.8430494 0.8778903 0.7053745
##  [8] 0.8229931 0.8011054 0.7233837 0.6196408 0.7882598 0.7794412
```

Let's move to hapmap data. We will use only data from Chromosome 2.
You should have downloaded and uncompressed the data file `hm_chr2_1.txt.gz` before proceeding

```
################################################
dat<-data.frame(matrix(scan("./hm_chr2_1.txt",skip=1),byrow=TRUE,nrow=209))
rn<-scan("./hm_chr2_1.txt",nlines=1,what=character())
names(dat)<-rn
loc.pos<-read.table("pos_loc_chr2.txt",header=T)
lpos<-loc.pos[(1:(dim(loc.pos)[1]%/%4))*4,2]
head(dat[,1:10])
```

```
##   Pop rs10193286 rs4632379 rs11510409 rs7594188 rs7594567 rs4637157
## 1   1         22        11         22        11        11        22
## 2   1         22        11         12        11        11        12
## 3   1         22        11         22        11        11        22
## 4   1         22        11         22        11        11        22
## 5   1         22        11         22        11        11        22
## 6   1         22        11         22        11        11        22
##   rs4311069 rs4522651 rs4271774
## 1        11        22        22
## 2        11        22        22
## 3        12        22        22
## 4        22        22        22
## 5        11        22        22
## 6        11        22        22
```

If everything went well, the data has been loaded into R and we can start working. We now estimate β for each snp and population, and store it in object `bi`:

```
bhmc2<-betas(dat) #takes a while...
#check sweep help page, very useful.
bi<-t(with(bhmc2,1-sweep(Hi,2,Hb,FUN="/")))
dim(bi)
```

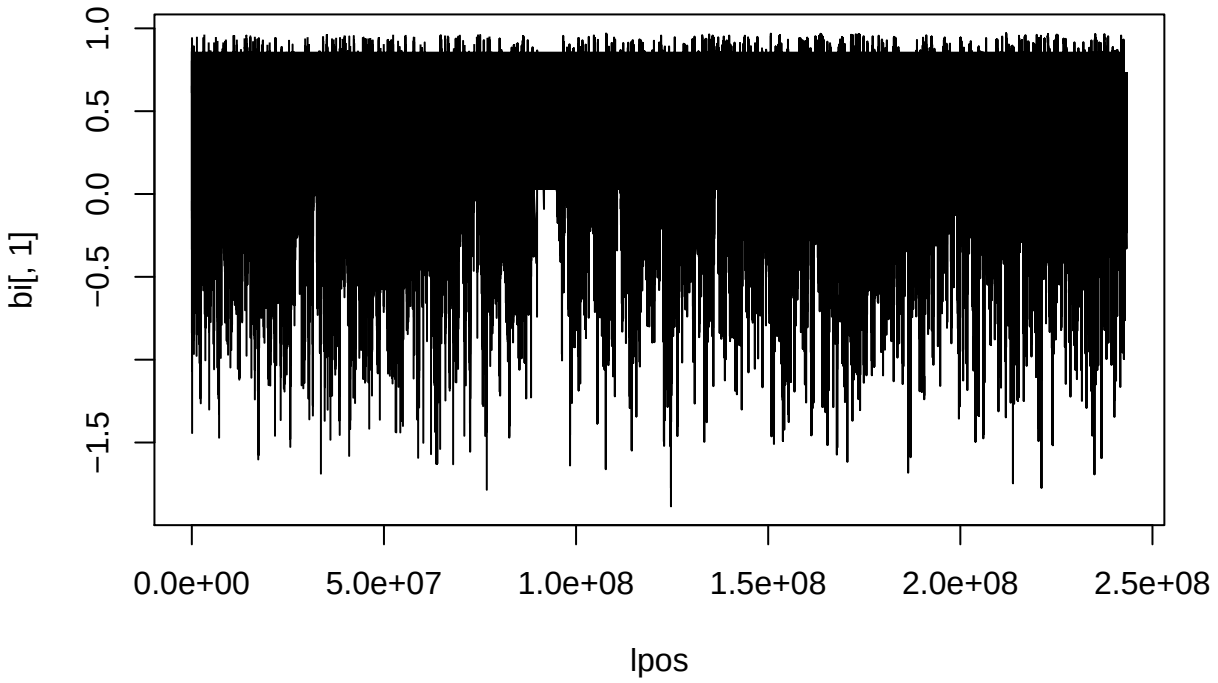
```
## [1] 54649     4
```

```
head(bi)
```

```
##           [,1]         [,2]         [,3]        [,4]
## [1,] 0.8590328  0.905225463  0.903071497 -1.12107802
## [2,] 0.6524029  0.536537196  0.526003950 -1.61720172
## [3,] 0.6105363 -0.006296871 -0.002691688  0.02592066
## [4,] 0.8050504  0.868932220  0.865953406 -1.53682240
## [5,] 0.7844109  0.855055887  0.851761702 -1.62269674
## [6,] 0.5207322  0.378600010  0.207651096 -0.53118908
```

Each column is a different population, and each row is a SNP. Let's plot the first column (Europeans)

```
plot(lpos,bi[,1],type="l")
```



Looks like a mess... The solution is to use a sliding window, and to obtain a moving average along this sliding windows:

```
m<-2500000 #half a windows size i.e 2.5Mb
#lapply apply function to each element of a list
ul<-lapply(lpos,function(x) max(which(lpos<=min(max(lpos),x+m))))
ll<-lapply(lpos,function(x) min(which(lpos>=max(lpos[1],x-m))))
all<-cbind(unlist(ll),unlist(ul))
head(all)
```

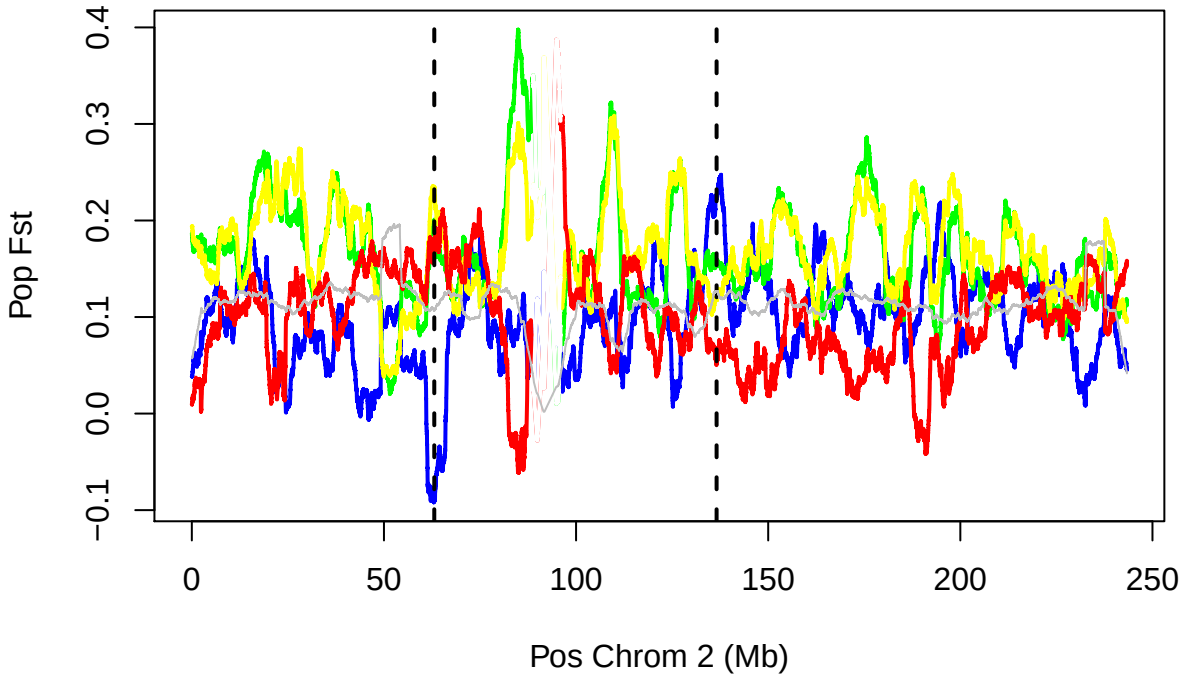
```
##      [,1] [,2]
## [1,]    1  573
## [2,]    1  573
## [3,]    1  574
## [4,]    1  574
## [5,]    1  574
## [6,]    1  574
```

```
eur<-unlist(apply(all,1,function(x) mean(bi[x[1]:x[2],1],na.rm=T))) #mov average ceu
chi<-unlist(apply(all,1,function(x) mean(bi[x[1]:x[2],2],na.rm=T))) #mov avg chb
jap<-unlist(apply(all,1,function(x) mean(bi[x[1]:x[2],3],na.rm=T))) #mov avg jpt
yor<-unlist(apply(all,1,function(x) mean(bi[x[1]:x[2],4],na.rm=T))) #mov avg yri
```

(Note that we took the averages of the β. As an exercice, estimate βs using the sum of numerators divided by the sums of denominators.)

Now we are ready to plot the population specific β along chromosome 2:

```
posmb<-lpos/1000000 #snp position, in megabase
nesnp<-which((all[,2]-all[,1])<400)
plot(posmb,eur,ylim=range(c(eur,chi,jap,yor)),type="l",col="blue",
     xlab="Pos Chrom 2 (Mb)",ylab="Pop Fst",lwd=2)
lines(posmb,chi,col="green",lwd=2)
lines(posmb,jap,col="yellow",lwd=2)
lines(posmb,yor,col="red",lwd=2)
lines(posmb[nesnp],eur[nesnp],col="white",lwd=2)
lines(posmb[nesnp],chi[nesnp],col="white",lwd=2)
lines(posmb[nesnp],jap[nesnp],col="white",lwd=2)
lines(posmb[nesnp],yor[nesnp],col="white",lwd=2)
abline(v=136.570,lwd=2,lty=2) #lactase gene pos omim 603202
abline(v=63.1,lwd=2,lty=2) #prostate cancer gene omim 611868
lines(posmb,(all[,2]-all[,1])/10000,col="grey") #nb snps .
```



Time allowing, look at *Galba truncatula* dataset. Data set described and analysed in Trouve et al.

```
data(gtrunchier)
table(gtrunchier[,2:1])
```

```
##      Locality
## Patch  1  2  3  4  5  6
##    1  13  0  0  0  0  0
##    2  12  0  0  0  0  0
##    3  15  0  0  0  0  0
```

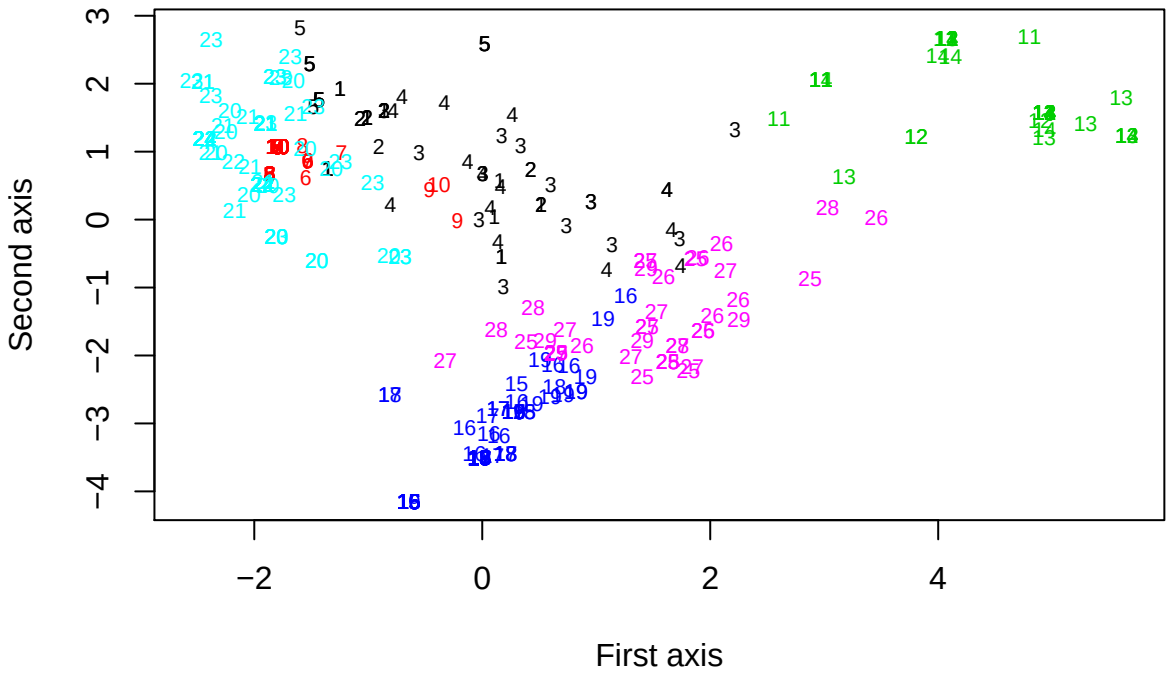
```
##    4  15  0  0  0  0  0
##    5  13  0  0  0  0  0
##    6   0 12  0  0  0  0
##    7   0 15  0  0  0  0
##    8   0 14  0  0  0  0
##    9   0 13  0  0  0  0
##    10  0 15  0  0  0  0
##    11  0  0 11  0  0  0
##    12  0  0 11  0  0  0
##    13  0  0 13  0  0  0
##    14  0  0 13  0  0  0
##    15  0  0  0 14  0  0
##    16  0  0  0 15  0  0
##    17  0  0  0 15  0  0
##    18  0  0  0 15  0  0
##    19  0  0  0 13  0  0
##    20  0  0  0  0 14  0
##    21  0  0  0  0 15  0
##    22  0  0  0  0 14  0
##    23  0  0  0  0 15  0
##    24  0  0  0  0 15  0
##    25  0  0  0  0  0 12
##    26  0  0  0  0  0  8
##    27  0  0  0  0  0 10
##    28  0  0  0  0  0  5
##    29  0  0  0  0  0  5
```

```
vc.gtr<-with(gtrunchier,varcomp.glob(data.frame(Locality,Patch),gtrunchier[,-c(1,2)]))
vc.gtr$F
```

```
##           Locality     Patch       Ind
## Total    0.5158366 0.6426780 0.9152763
## Locality 0.0000000 0.2619806 0.8250101
## Patch    0.0000000 0.0000000 0.7628926
```

Do we see this structure in an individual based PCA?

```
x<-indpca(gtrunchier[,-2],ind.labels=gtrunchier[,2]) #ind label is patch of origin
plot(x,col=gtrunchier[,1],cex=0.7) # color is population
```


Second axis
First axis
−4
−3
−2
−1
0
1
2
3
−2
0
2
4