

# IEA

## LEA Overview

- LEA is an R package
- population structure
- genome-wide tests
- identifying genetic polymorphisms

## lea tutorial

```
# make directory
getwd()
setwd("G:/R CLASS practice/lea2")
#first install LEA and call it
install.packages("LEA")
library(LEA)
```

## tutorial data

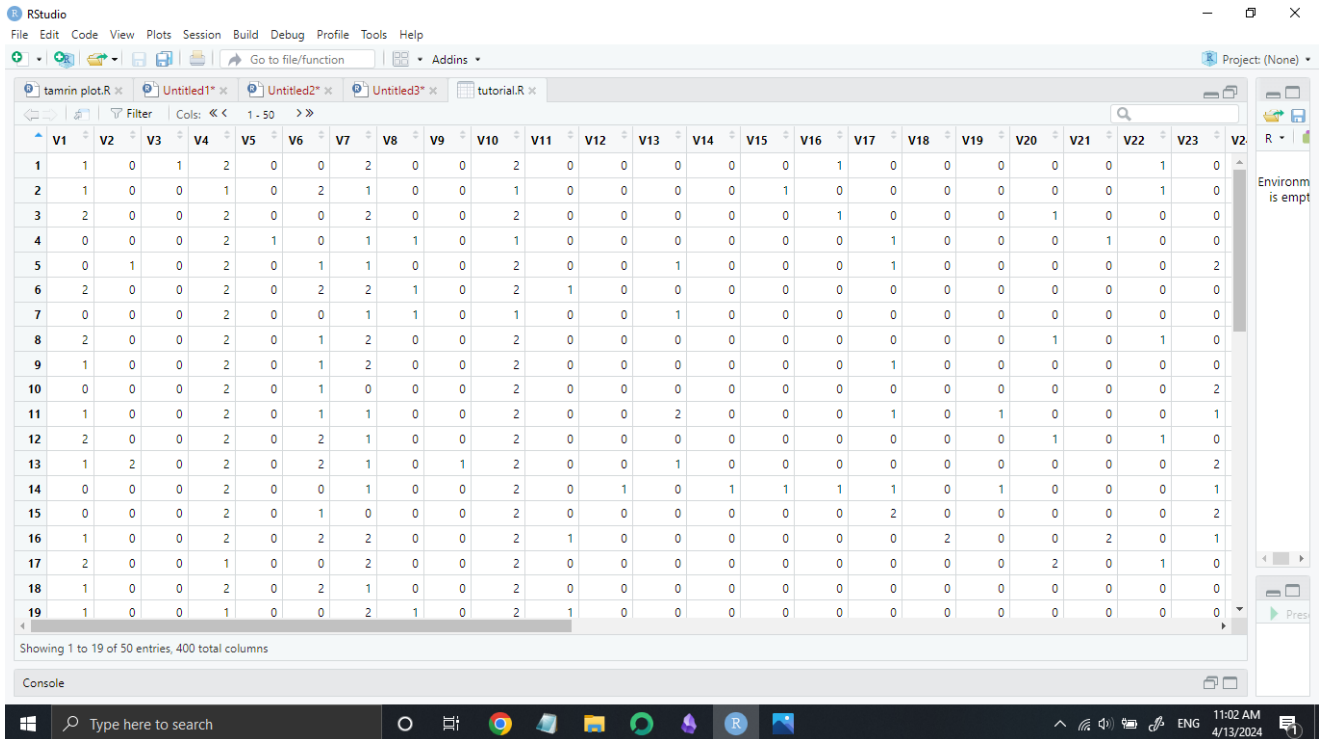
LEA have a small tutorial dataset consisting of 400 SNPs genotyped for 50 diploid individuals in side the the package.

```
data("tutorial")
#type of file you need
write.lfmm(tutorial.R, "genotypes.lfmm")
write.geno(tutorial.R, "genotypes.geno")

write.env(tutorial.C, "gradients.env")
# creation of an environment gradient file:
```

```
gradient.env. # The .env file contains a single
ecological variable
```

## raw data




The screenshot shows the RStudio interface with a data table loaded. The table has 50 columns labeled V1 through V50 and 19 rows of data. The data consists of binary values (0s and 1s). The status bar at the bottom indicates 'Showing 1 to 19 of 50 entries, 400 total columns'.

	V1	V2	V3	V4	V5	V6	V7	V8	V9	V10	V11	V12	V13	V14	V15	V16	V17	V18	V19	V20	V21	V22	V23	V24
1	1	1	0	1	2	0	0	2	0	0	2	0	0	0	0	0	1	0	0	0	0	0	1	0
2	1	0	0	0	1	0	2	1	0	0	1	0	0	0	0	1	0	0	0	0	0	0	1	0
3	2	0	0	0	2	0	0	2	0	0	2	0	0	0	0	0	1	0	0	0	1	0	0	0
4	0	0	0	0	2	1	0	1	1	0	1	0	0	0	0	0	1	0	0	0	1	0	0	0
5	0	1	0	2	0	1	1	0	0	2	0	0	1	0	0	0	1	0	0	0	0	0	0	2
6	2	0	0	2	0	2	2	1	0	2	1	0	0	0	0	0	0	0	0	0	0	0	0	0
7	0	0	0	2	0	0	0	1	1	0	1	0	0	1	0	0	0	0	0	0	0	0	0	0
8	2	0	0	2	0	1	2	0	0	2	0	0	0	0	0	0	0	0	0	1	0	1	0	0
9	1	0	0	2	0	1	2	0	0	2	0	0	0	0	0	0	1	0	0	0	0	0	0	0
10	0	0	0	2	0	1	0	0	0	2	0	0	0	0	0	0	0	0	0	0	0	0	0	2
11	1	0	0	2	0	1	1	0	0	2	0	0	2	0	0	0	1	0	1	0	0	0	0	1
12	2	0	0	2	0	2	1	0	0	2	0	0	0	0	0	0	0	0	0	1	0	1	0	0
13	1	2	0	2	0	2	1	0	1	2	0	0	1	0	0	0	0	0	0	0	0	0	0	2
14	0	0	0	2	0	0	1	0	0	2	0	0	1	0	1	1	1	0	1	0	0	0	0	1
15	0	0	0	2	0	1	0	0	0	2	0	0	0	0	0	0	2	0	0	0	0	0	0	2
16	1	0	0	2	0	2	2	0	0	2	1	0	0	0	0	0	0	2	0	0	2	0	1	0
17	2	0	0	1	0	0	2	0	0	2	0	0	0	0	0	0	0	0	0	2	0	1	0	0
18	1	0	0	2	0	2	1	0	0	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0
19	1	0	0	1	0	0	2	1	0	2	1	0	0	0	0	0	0	0	0	0	0	0	0	0

## genotypes in the lfmm format

Git CLASS practice\lea\genoM.lfmm - Notepad++

File Edit Search View Encoding Language Settings Tools Macro Run Plugins Window ?



Boxplot\_data.csv [x] CART-p-o-t-g geno [x] genotypes.geno [x] CART-p-o-t-g geno [x] genoM.geno [x] **genoM.lfmm [x]**

1	1	0	1	2	0	0	2	0	0	2	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0	0	0	0	0	2	0	2	0	1	2	0	0	1	0	0	0			
2	1	0	0	1	0	2	1	0	0	1	0	0	0	1	0	0	0	0	0	0	0	1	0	0	0	0	0	0	2	0	2	0	0	2	0	1	0	1	0	0				
3	2	0	0	2	0	0	2	0	0	2	0	0	0	1	0	0	0	1	0	0	0	1	0	0	0	1	0	0	0	2	0	2	0	0	2	0	1	0	0	0	0			
4	0	0	0	2	1	0	1	1	0	1	0	0	0	0	0	1	0	0	0	1	0	0	0	0	0	0	0	0	1	0	2	0	0	2	0	0	0	0	0	0	0			
5	0	1	0	2	0	1	1	0	0	2	0	0	1	0	0	0	1	0	0	0	0	0	2	0	0	0	0	0	2	0	2	0	9	2	0	0	0	0	0	0	0	0		
6	2	0	0	2	0	2	2	1	0	2	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	2	2	0	2	1	0	0	0	0	0	0			
7	0	0	0	2	0	0	1	1	0	1	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2	0	2	0	0	9	0	1	0	0	0	0	0	0		
8	2	0	0	2	0	1	2	0	0	2	0	0	0	0	0	0	0	0	1	0	1	0	0	0	0	0	0	2	0	2	0	1	2	0	1	0	1	0	0	0	0			
9	1	0	0	2	0	1	2	0	0	2	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0	0	0	2	2	2	1	0	2	0	0	0	0	0	0	0	1		
10	0	0	0	2	0	1	0	0	0	2	0	0	0	0	0	0	0	0	0	0	0	0	2	0	0	0	0	0	2	0	2	0	0	0	0	0	1	0	0	0	0	0	0	
11	1	0	0	2	0	1	1	0	0	2	0	0	2	0	0	1	0	1	0	0	0	1	0	0	0	0	0	0	2	1	2	0	0	1	0	0	0	0	0	0	0	0	0	0
12	2	0	0	2	0	2	1	0	0	2	0	0	0	0	0	0	0	0	1	0	1	0	0	0	0	0	0	0	2	1	0	0	0	2	0	0	0	0	0	0	0	0	0	0
13	1	2	0	2	0	2	1	0	1	2	0	0	1	0	0	0	0	9	0	0	0	0	2	0	0	0	0	0	2	0	2	1	0	1	0	0	0	0	0	0	0	0	0	0
14	0	0	0	2	0	9	1	0	0	2	0	1	0	1	1	1	1	0	1	0	0	0	1	0	0	0	0	2	0	2	0	0	2	0	2	0	0	0	0	0	0	0	0	0
15	0	0	0	2	0	1	0	0	0	2	0	0	0	0	0	2	0	0	0	0	0	2	0	1	0	0	0	2	0	2	0	0	2	0	0	1	0	1	0	0	0	0	0	0
16	1	0	0	2	0	2	2	0	0	2	1	0	0	0	0	0	2	0	0	2	0	1	0	0	0	1	0	2	0	2	2	0	2	0	0	0	0	0	0	0	0	0	0	0
17	2	0	0	1	0	0	2	0	0	2	0	0	0	0	0	0	0	0	2	0	1	0	0	0	0	0	1	2	0	2	0	1	2	0	1	1	1	0	0	0	0	0	0	
18	1	0	0	2	0	2	1	0	0	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2	0	2	1	0	2	0	0	0	0	0	0	0	0	0	0
19	1	0	0	1	0	0	2	1	0	2	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1	1	2	2	0	2	1	0	0	0	0	0	0

Normal text file

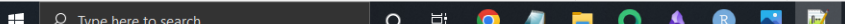
length: 40,050 lines: 51

Ln: 1 Col: 1 Pos: 1

Windows (CR LF)

UTF-8

INS



Type here to search

11:21 AM 4/13/2024

## genotypes in the geno format

```
G:\R CLASS practice\lea\genoM.geno - Notepad++
File Edit Search View Encoding Language Settings Tools Macro Run Plugins Window ?
Boxplot_data.csv CART-p-o-t-g.geno genotypes.geno CART-p-o-t-g.geno genoM.geno
1 11200202101210012111010101102111102191110011001120
2 0000100000002000000100000001000000000900000100000
3 1000000000000000000000000000000000000000000000000
4 21222222222222212122122222212222122122222222222
5 0001000000000000000000000000000000000000000000000
6 02001201111229120200111001012000010220111000011101
7 21211212201111022122120211010121011211122202021211
8 0001011000000000001000000000000000000000000000000
9 0000000000001000000000000000000000000000000020000000010
10 21212212222222222221212212221222222222222222202222
11 0000010000000001001000000000000000010100000000001001
12 000000000000010000000000000101010000000000000000900
13 0000101000201000000000000001000010000010000000000
14 000000000000010000000000020000010100000000000000000
15 010000000000010000000000000100000000000000000000000
16 101000000000010000000000000000000000000000000000000
17 000110001010012000010000000000000000000000000000000
18 000000000000000200000000000000000000000000000000000
19 000000000001091000001000000000000000000000000000000
20 001000010001000020000000000100000000000000000000000
Normal text file length: 20,800 lines: 401 Ln: 11 Col: 51 Pos: 571 Windows (CR LF) UTF-8 INS
11:13 AM 4/13/2024
```

The `geno` format has one row for each SNP. Each row contains 1 character for each individual: 0 means zero copy of the reference allele. 1 means one copy of the reference allele. 2 means two copies of the reference allele. 9 means missing data.

```
G:\R CLASS practice\lea\gradients.env - Notepad++
File Edit Search View Encoding Language Settings Tools Macro Run Plugins Window ?
Boxplot_data.csv CART-p-o-t-g.geno genotypes.geno CART-p-o-t-g.geno genoM.geno genoM.firm gradients.env
1 -3.95986621384311
2 -5.11799803084197
3 3.34656631220578
4 -16.9864977408982
5 16.4681065702462
6 -12.8833622098789
7 10.5444709244714
8 4.08582820627048
9 -3.70181816426244
10 16.3619643058549
11 26.506755060744
12 -7.86345637580724
13 2.01040080790134
14 13.9810170241959
15 13.7076202935426
16 4.33744779886798
17 -16.4646171785029
18 1.50687525209416
19 -18.9727835950147
20 7.42485845090329
Normal text file length: 923 lines: 51 Ln: 1 Col: 1 Pos: 1 Windows (CR LF) UTF-8 INS
11:44 AM 4/13/2024
```

## Analysis of population structure

The R package LEA implements two classical approaches for the estimation of population genetic structure: principal component analysis (pca) and admixture analysis using sparse nonnegative matrix factorization (snmf)

## Principal Component Analysis

```
pc = pca("genotypes.lfmm", scale = TRUE)
tw = tracy.widom(pc)
# Available options, K (the number of PCs),
# center and scale.
# Create files: genotypes.eigenvalues - eigenvalues,
# genotypes.eigenvectors - eigenvectors,
# genotypes.sdev - standard deviations,
# genotypes.projections - projections,
# Create a pcaProject object: pc.
```

## Result

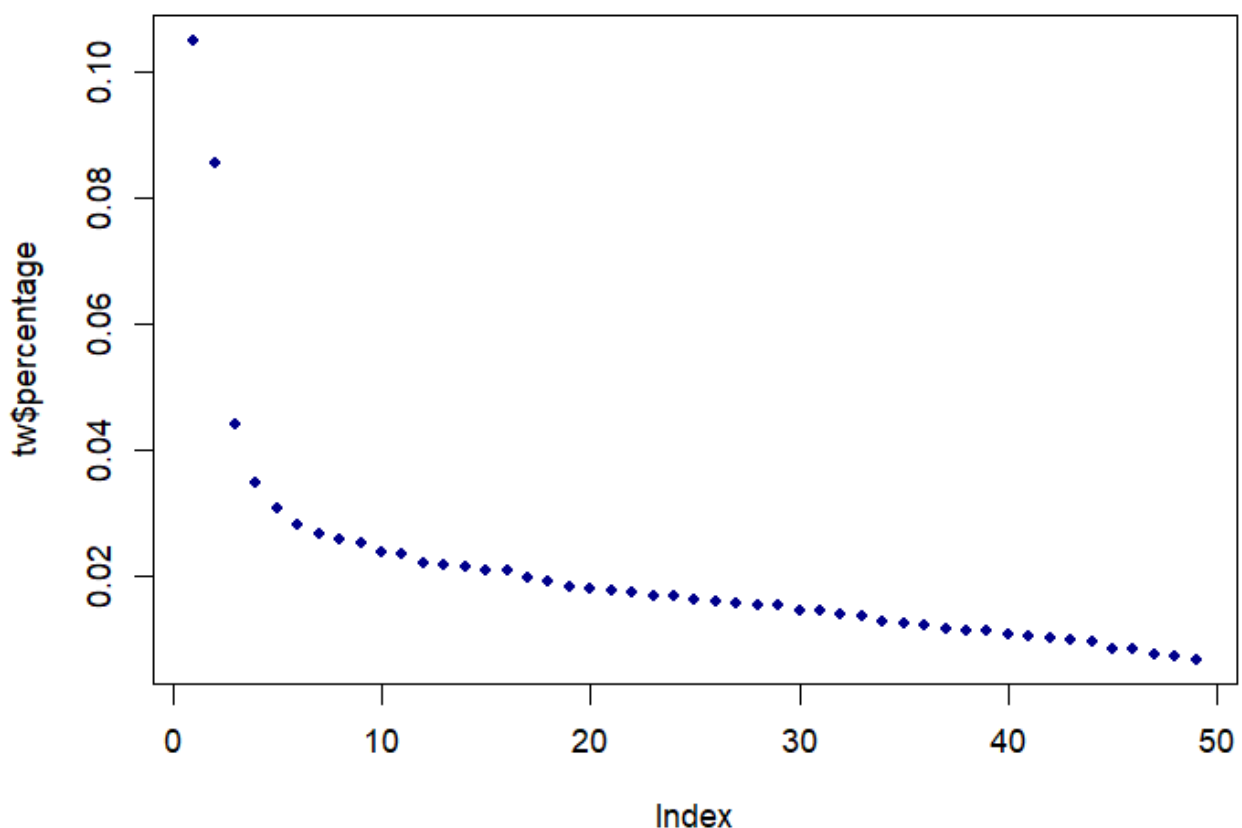
```
-n (number of individuals)          50
-L (number of loci)                 400
-K (number of principal components) 50
-x (genotype file)                  G:\R CLASS
practice\lea\genotypes.lfmm
-a (eigenvalue file)                 G:\R CLASS
practice\lea\genotypes.pca/genotypes.eigenvalues
-e (eigenvector file)                G:\R CLASS
practice\lea\genotypes.pca/genotypes.eigenvectors
-d (standard deviation file)         G:\R CLASS
practice\lea\genotypes.pca/genotypes.sdev
-p (projection file)                 G:\R CLASS
```

```
practice\lea\genotypes.pca/genotypes.projections  
-s data centered and scaled
```

## percentage of variance

```
#plot the percentage of variance explained by each  
component plot(tw$percentage, pch = 19, col =  
"darkblue", cex = .8)
```

## percentage of variance

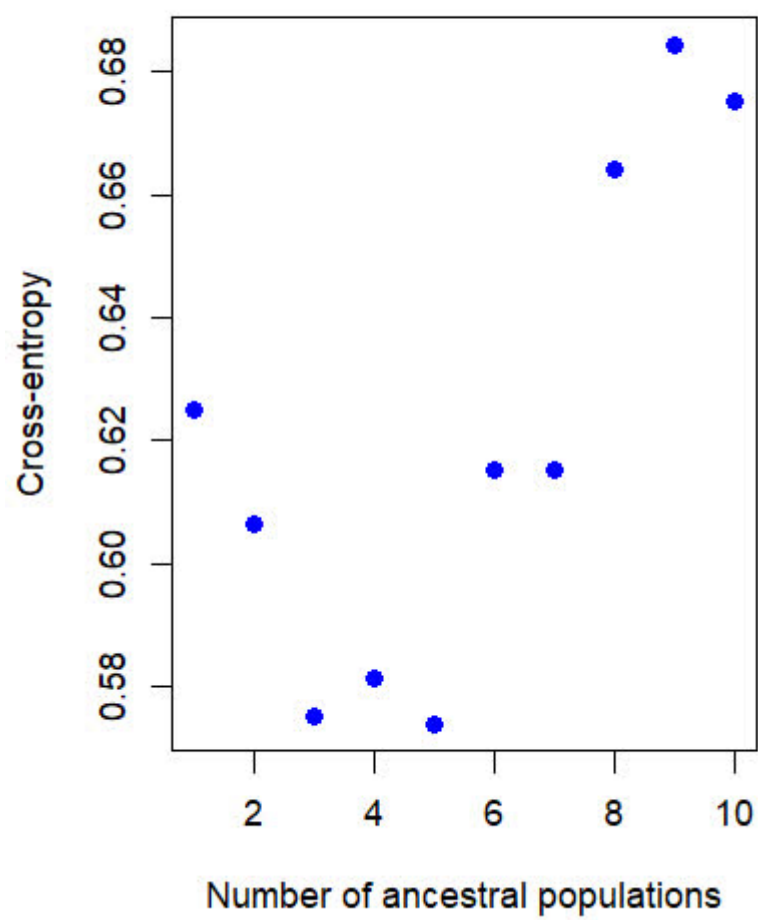


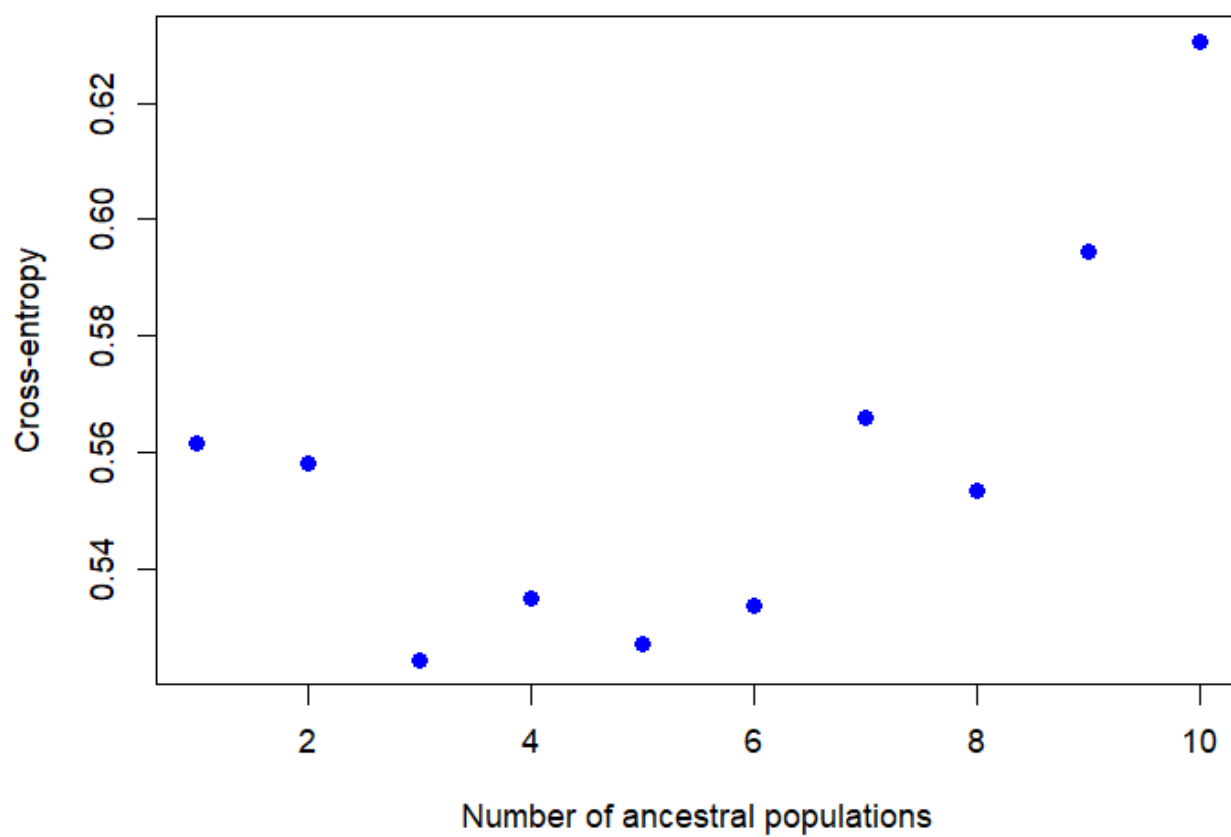
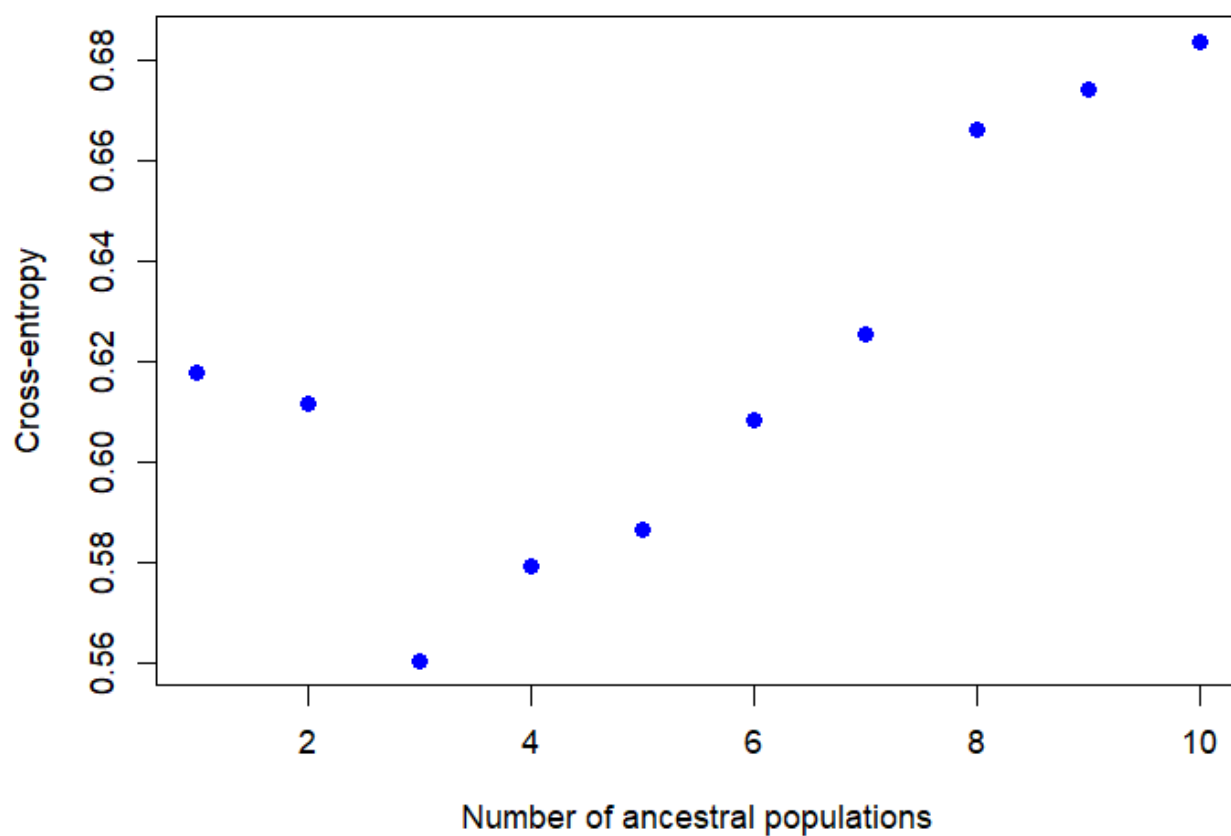
## STRUCTURE

```
# main options  
# K = number of ancestral populations
```

```
# entropy = TRUE computes the cross entropy criterion,  
# CPU = 4 is the number of CPU used (hidden input)  
project = NULL  
project = snmf("genotypes.geno", K = 1:10, entropy =  
TRUE, repetitions = 10, project = "new")  
plot(project, col = "blue", pch = 19, cex = 1.2)
```

## Result



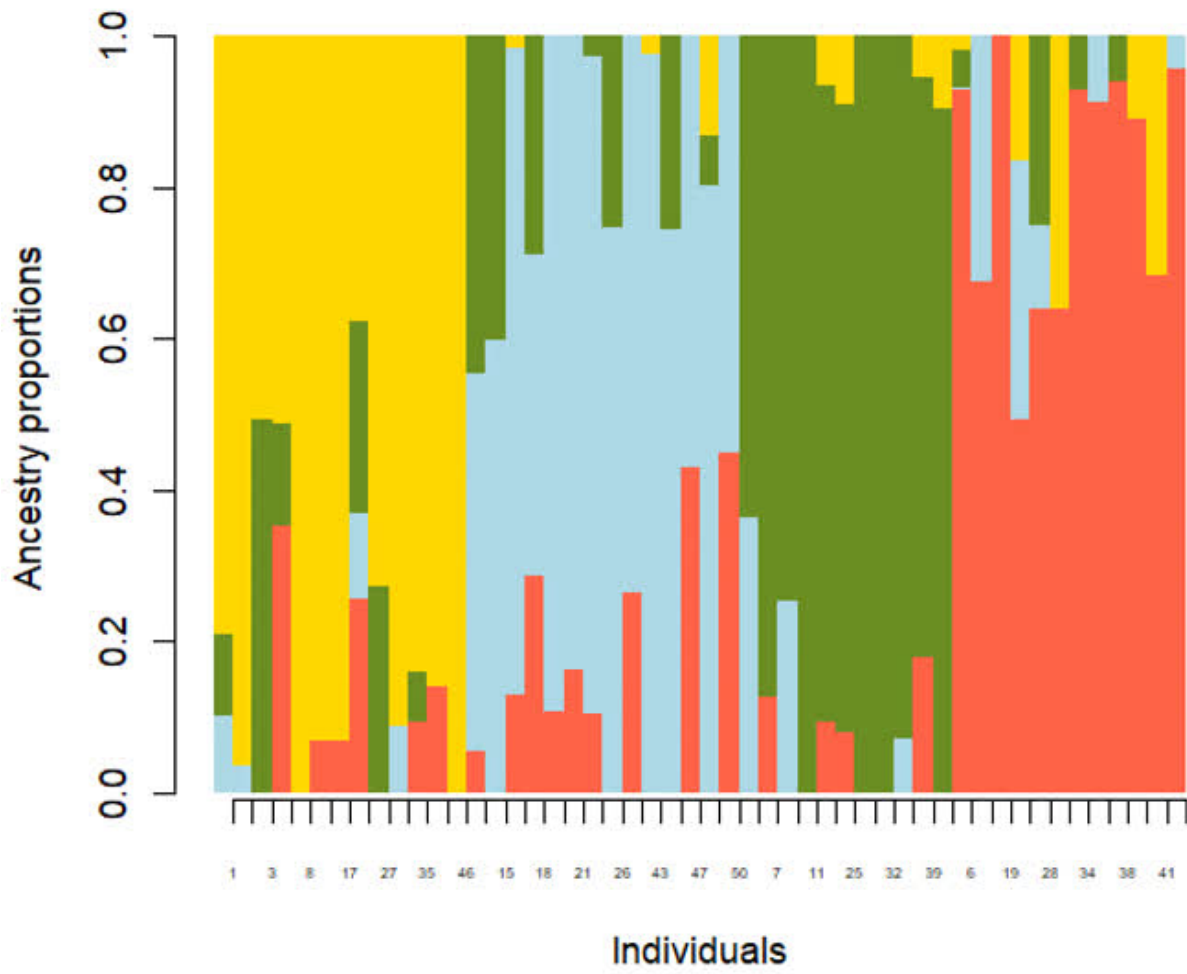




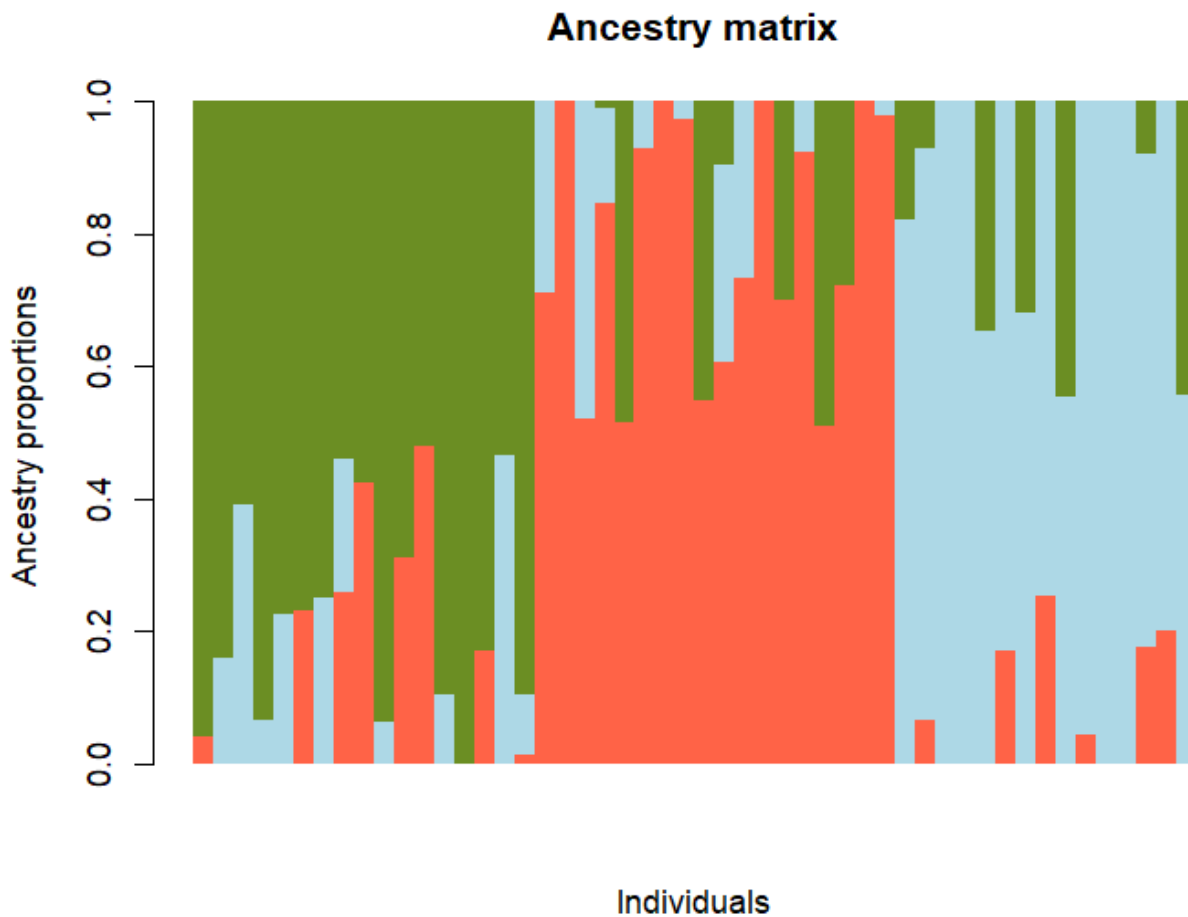
```
best = which.min(cross.entropy(project, K = 3))
my.colors <- c("tomato", "lightblue",
               "olivedrab", "gold")
barchart(project, K = 3, run = best,
          border = NA, space = 0,
          col = my.colors,
          xlab = "Individuals",
          ylab = "Ancestry proportions",
          main = "Ancestry matrix") -> bp
axis(1, at = 1:length(bp$order),
     labels = bp$order, las=1,
     cex.axis = .4)
```

## Result

Ancestry matrix



k=5



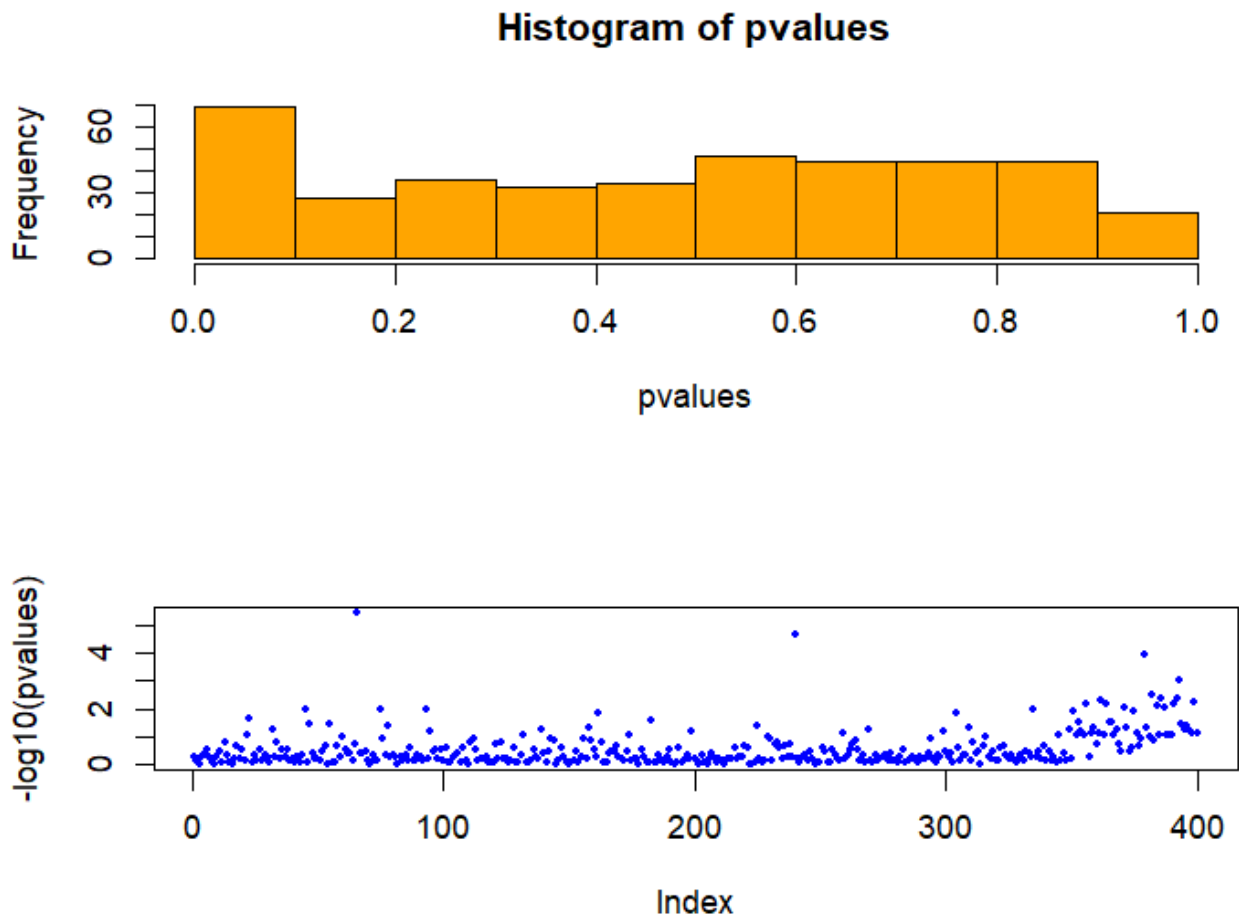
k=3

## Population differentiation tests

```
p = snmf.pvalues(project,
                  entropy = TRUE,
                  ploidy = 2,
                  K = 3)

pvalues = p$pvalues
par(mfrow = c(2,1))
hist(pvalues, col = "orange")
plot(-log10(pvalues), pch = 19, col = "blue", cex = .5)
```

## Result



P-values for population differentiation tests with `snmf()`

## Ecological association tests using `lfmm`

```
# creation of a genotype matrix with missing genotypes
dat = as.numeric(tutorial.R)
dat[sample(1:length(dat), 100)] <- 9
dat <- matrix(dat, nrow = 50, ncol = 400)
write.lfmm(dat, "genoM.lfmm")
## [1] "genoM.lfmm"
project.missing = snmf("genoM.lfmm", K = 4,
                      entropy = TRUE, repetitions = 10,
                      project = "new")

####
# select the run with the lowest cross-entropy value
best = which.min(cross.entropy(project.missing, K = 4))
```

```

# Impute the missing genotypes
impute(project.missing, "genoM.lfmm",
        method = 'mode', K = 4, run = best)
## Missing genotype imputation for K = 4
## Missing genotype imputation for run = 1
## Results are written in the file:
genoM.lfmm_imputed.lfmm
# Proportion of correct imputation results
dat.imp = read.lfmm("genoM.lfmm_imputed.lfmm")
mean( tutorial.R[dat == 9] == dat.imp[dat == 9] )
## [1] 0.83

project = NULL
project = lfmm("genotypes.lfmm",
               "gradients.env",
               K = 6,
               repetitions = 5,
               project = "new")

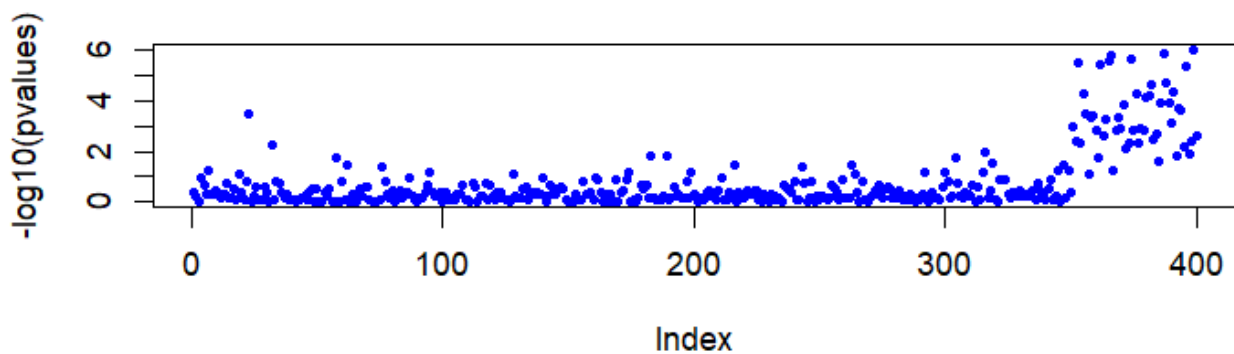
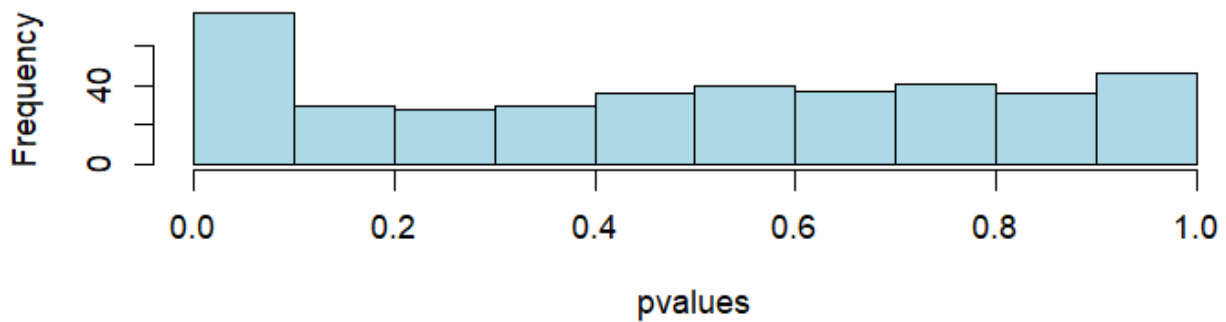
p = lfmm.pvalues(project, K = 6)
pvalues = p$pvalues

par(mfrow = c(2,1))
hist(pvalues, col = "lightblue")
plot(-log10(pvalues), pch = 19, col = "blue", cex = .7)

```

## Result

Histogram of pvalues



```
for (alpha in c(.05,.1,.15,.2)) {  
  # expected FDR  
  12  
  print(paste("Expected FDR:", alpha))  
  L = length(pvalues)  
  # return a list of candidates with expected FDR alpha.  
  # Benjamini-Hochberg's algorithm:  
  w = which(sort(pvalues) < alpha * (1:L) / L)  
  candidates = order(pvalues)[w]  
  # estimated FDR and True Positive Rate  
  Lc = length(candidates)  
  estimated.FDR = sum(candidates <= 350)/Lc  
  print(paste("Observed FDR:",  
              round(estimated.FDR, digits = 2)))  
  estimated.TPR = sum(candidates > 350)/50  
  print(paste("Estimated TPR:",
```

```
round(estimated.TPR, digits = 2))
```

```
[1] "Expected FDR: 0.05"  
[1] "Observed FDR: 0.02"  
[1] "Estimated TPR: 0.84"  
[1] "Expected FDR: 0.1"  
[1] "Observed FDR: 0.06"  
[1] "Estimated TPR: 0.88"  
[1] "Expected FDR: 0.15"  
[1] "Observed FDR: 0.13"  
[1] "Estimated TPR: 0.94"  
[1] "Expected FDR: 0.2"  
[1] "Observed FDR: 0.13"  
[1] "Estimated TPR: 0.96"
```

## Ecological association tests using lfmm2

```
# load simulated data  
data("offset_example")  
# 200 diploid individuals genotyped at 510 SNP  
Y <- offset_example$geno  
# 4 environmental variables  
X <- offset_example$env  
mod.lfmm2 <- lfmm2(input = Y, env = X, K = 2)  
  
# Simulate non-null effect sizes for 10 target loci  
# individuals  
n = 100  
# loci  
L = 1000  
# Environmental variable  
X = as.matrix(rnorm(n))
```

```

# effect sizes
B = rep(0, L)
target = sample(1:L, 10)
# GEA significance test
# showing the K = 2 estimated factors
plot(mod.lfmm2@U, col = "grey", pch = 20,
      xlab = "Factor 1",
      ylab = "Factor 2")

B[target] = runif(10, -10, 10)
# Create 3 hidden factors and their loadings
U = t(tcrossprod(as.matrix(c(-1,0.5,1.5)), X)) +
  matrix(rnorm(3*n), ncol = 3)
V <- matrix(rnorm(3*L), ncol = 3)

pv <- lfmm2.test(object = mod.lfmm2,
                 input = Y,
                 env = X,
                 full = TRUE)
plot(-log10(pv$pvalues), col = "grey", cex = .5, pch =
19)
abline(h = -log10(0.1/510), lty = 2, col = "orange")

```

## Result



