

# Simulate Admixed Populations with **bnpsd**

*Alejandro Ochoa and John D. Storey*

*2019-02-11*

## 1 Introduction

The **bnpsd** package simulates the genotypes of an admixed population. In the PSD model (Pritchard, Stephens, and Donnelly 2000), admixed individuals draw their alleles with individual-specific probabilities (admixture proportions) from  $K$  intermediate subpopulations. We impose the BN model (Balding and Nichols 1995) to the intermediate subpopulation allele frequency, which thus evolve independently with subpopulation-specific inbreeding coefficients ( $F_{ST}$  values) from a common ancestral population  $T$ . The kinship coefficients and generalized  $F_{ST}$  of the admixed individuals were derived in recent work (Ochoa and Storey 2016a). A simulated admixed population was used to benchmark kinship and  $F_{ST}$  estimators in the accompanying paper (Ochoa and Storey 2016b). Here we briefly summarize the notation and intuition behind the key parameters (see (Ochoa and Storey 2016a) for precise definitions).

### 1.1 The BN-PSD population structure

The population structure determines how individuals are related to each other. The key parameters are the inbreeding coefficients of the intermediate subpopulations ( $f_{S_u}^T$  below) and the admixture proportions of each individual for each subpopulation ( $q_{ju}$ ), which are treated as fixed variables.

Each intermediate subpopulation  $S_u$  ( $u \in \{1, \dots, K\}$ ) evolved independently from a shared ancestral population  $T$  with an inbreeding coefficient denoted by  $f_{S_u}^T$ . Each admixed individual  $j \in \{1, \dots, n\}$  draws each allele from  $S_u$  with probability given by the admixture proportion  $q_{ju}$  ( $\sum_{u=1}^K q_{ju} = 1 \forall j$ ). In this case the coancestry coefficients  $\theta_{jk}^T$  between individuals  $j, k$  (including  $j = k$  case) and the  $F_{ST}$  of the admixed individuals are given by:

$$\theta_{jk}^T = \sum_{u=1}^K q_{ju} q_{ku} f_{S_u}^T, \quad F_{ST} = \sum_{j=1}^n \sum_{u=1}^K w_j q_{ju}^2 f_{S_u}^T,$$

where  $0 < w_j < 1$ ,  $\sum_{j=1}^n w_j = 1$  are user-defined weights for individuals (default  $w_j = \frac{1}{n} \forall j$ ). Note  $\theta_{jk}^T$  equals the kinship coefficient for  $j \neq k$  and the inbreeding coefficient for  $j = k$ .

The bias coefficient  $s$  is defined by

$$s = \frac{\bar{\theta}^T}{F_{ST}}$$

where  $\bar{\theta}^T = \sum_{j=1}^n \sum_{k=1}^n w_j w_k \theta_{jk}^T$ . This  $0 < s \leq 1$  approximates the proportional bias of  $F_{ST}$  estimators that assume independent subpopulations, and one **bnpsd** function below fits its parameters to yield a desired  $s$ .

### 1.2 Random allele frequencies and genotypes

This section details the distributions of the allele frequencies and genotypes of the various populations or individuals of the BN-PSD model.

Every biallelic locus  $i$  in the ancestral population  $T$  has an ancestral reference allele frequency denoted by  $p_i^T$ . By default the **bnpsd** code draws

$$p_i^T \sim \text{Uniform}(a, b)$$

with  $a = 0.01, b = 0.5$ , but the code accepts  $p_i^T$  from arbitrary distributions (see below).

The distribution of the allele frequency at locus  $i$  in subpopulation  $S_u$ , denoted by  $p_i^{S_u}$ , is the BN distribution:

$$p_i^{S_u}|T \sim \text{Beta}(\nu_s p_i^T, \nu_s (1 - p_i^T)),$$

where  $\nu_s = \frac{1}{f_{S_u}^T} - 1$ . Allele frequencies for different loci and different subpopulations ( $S_u, S_v, u \neq v$ ) are drawn independently.

Each admixed individual  $j$  at each locus  $i$  draws alleles from a mixture of Bernoulli distributions from each intermediate subpopulation, which is mathematically equivalent to assigning what we call “individual-specific allele frequencies” (IAFs)  $\pi_{ij}$  constructed as:

$$\pi_{ij} = \sum_{u=1}^K p_i^{S_u} q_{ju}.$$

The unphased genotype  $x_{ij}$  (encoded to count the number of reference alleles) is drawn as:

$$x_{ij}|\pi_{ij} \sim \text{Binomial}(2, \pi_{ij}).$$

## 2 Simulation examples

### 2.1 Population structure: 1D geography

Let’s generate the same population structure used in the simulation of (Ochoa and Storey 2016b).

```
library(RColorBrewer) # for nice colors
# load this package (bnpsd) and a related popgen package (popkin)
# since both packages have an "fst" function, load "bnpsd" last so its function isn't masked
library(popkin) # for visualizing coancestry matrix with plotPopkin
library(bnpsd)

##
## Attaching package: 'bnpsd'

## The following object is masked from 'package:popkin':
##
##      fst

# dimensions of data/model
n <- 100 # number of individuals (NOTE this is 10x less than in publication!)
k <- 10 # number of intermediate subpops

# define population structure
F <- 1:k # subpopulation FST vector, up to a scalar
s <- 0.5 # desired bias coefficient
Fst <- 0.1 # desired FST for the admixed individuals
obj <- q1d(n, k, s=s, F=F, Fst=Fst) # admixture proportions from 1D geography
Q <- obj$Q
F <- obj$F

# get pop structure parameters of the admixed individuals
Theta <- coanc(Q,F) # the coancestry matrix
# verify that we got the desired Fst!
Fst2 <- fst(Q,F)
Fst2
```

```
## [1] 0.1
```

```
# this should also equal Fst
```

```
inbr <- diag(Theta)
```

```
popkin::fst(inbr)
```

```
## [1] 0.1
```

```
# verify that we got the desired s too!
```

```
s2 <- mean(Theta)/Fst
```

```
s2
```

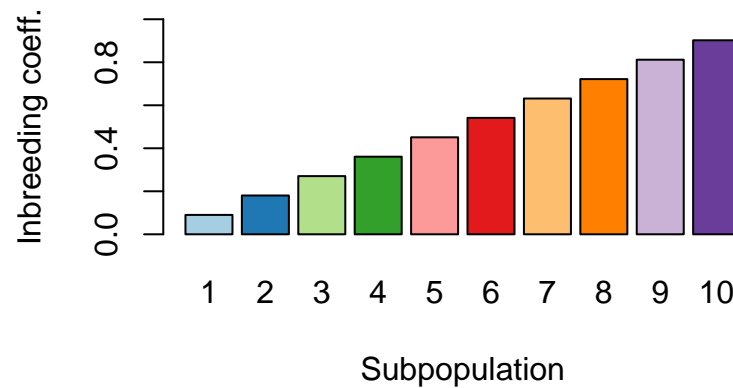
```
## [1] 0.5
```

```
# visualize the per-subpopulation inbreeding coefficients (FSTs)
```

```
par(mar=c(4,4,0,0)+0.2) # shrink default margins
```

```
colIS <- brewer.pal(k, "Paired") # indep. subpop. colors
```

```
barplot(F, col=colIS, names.arg=1:k, ylim=c(0,1),  
        xlab='Subpopulation', ylab='Inbreeding coeff.')
```

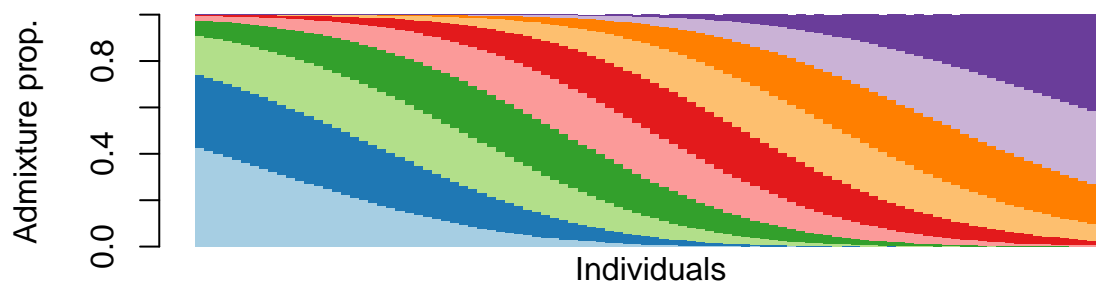


```
# visualize the admixture proportions
```

```
par(mar=c(1,4,0,0)+0.2) # shrink default margins
```

```
barplot(t(Q), col=colIS, border=NA, space=0, ylab='Admixture prop.')
```

```
mtext('Individuals', 1)
```



```
# Visualize the coancestry matrix using "popkin"!
```

```
# set outer margin for axis labels (left and right are non-zero)
```

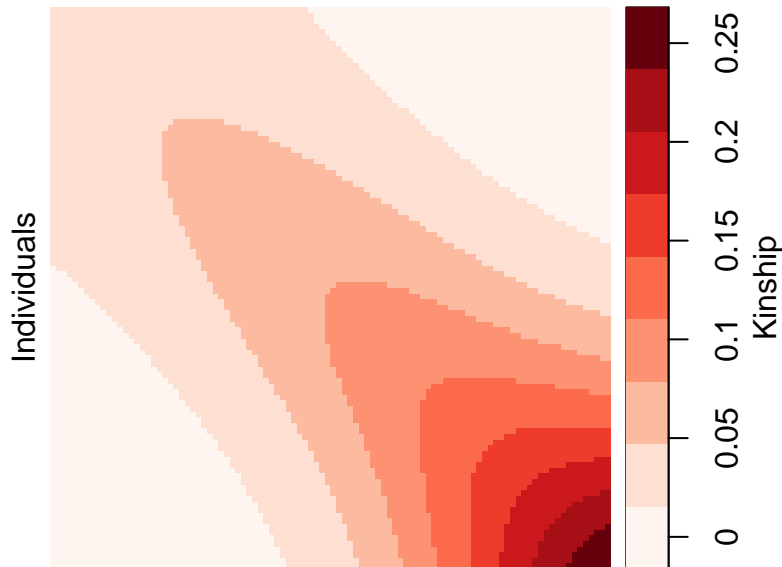
```
par(oma=c(0,1.5,0,3))
```

```
# zero inner margin (plus padding) because we have no individual or subpopulation labels
```

```
par(mar=c(0,0,0,0)+0.2)
```

```
# now plot!
```

```
plotPopkin(Theta)
```

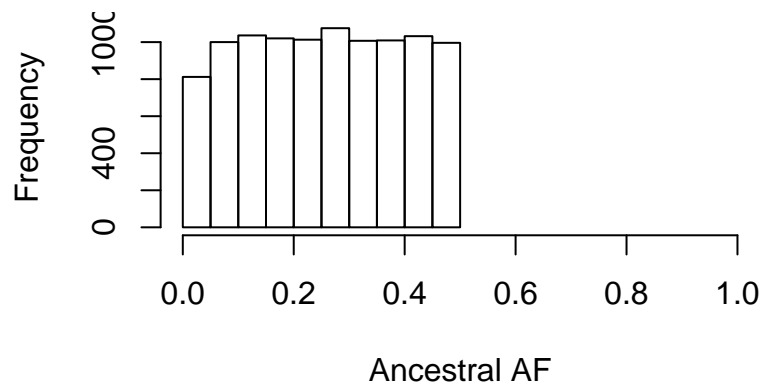


## 2.2 Draw random allele frequencies and genotypes

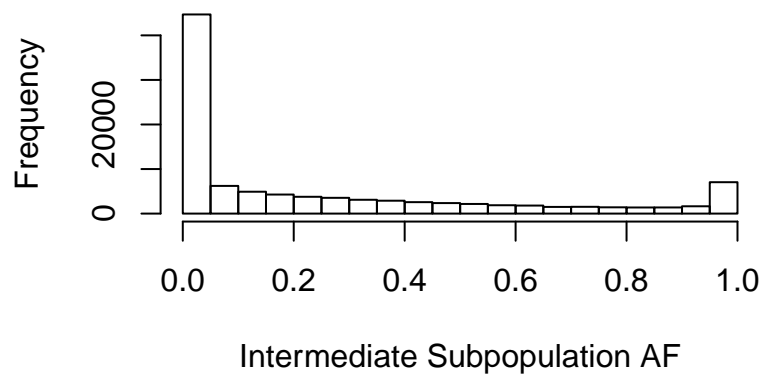
Now let's draw all the random allele frequencies and genotypes from the population structure. The easiest way is to use `rbnpsd` (the initial `r` is for drawing “random” samples in this and the following functions, in analogy to the `runif`, `rnorm`, `rbeta`, etc. functions from the `stats` R package).

```
m <- 10000 # number of loci in simulation (NOTE this is 30x less than in publication!)
# draw all random Allele Freqs (AFs) and genotypes
# reuse the previous F,Q
out <- rbnpsd(Q, F, m)
X <- out$X # genotypes
P <- out$P # IAFs (individual-specific AFs)
B <- out$B # intermediate AFs
pAnc <- out$Pa # ancestral AFs
```

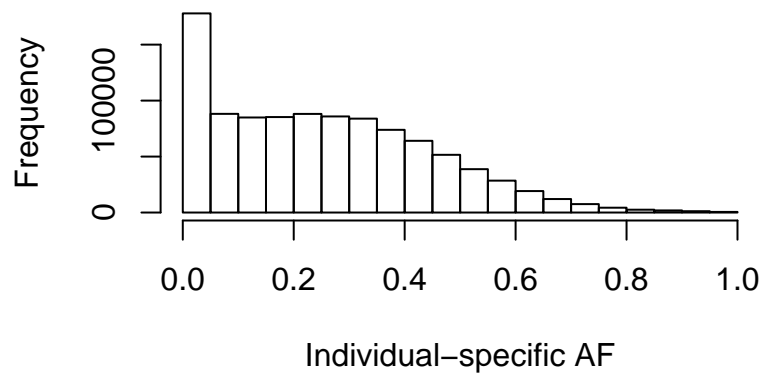
```
# inspect distribution of ancestral AFs (~ Uniform(0.01,0.5))
par(mar=c(4,4,0,0)+0.2) # shrink default margins for these figures
hist(pAnc, xlab='Ancestral AF', main='', xlim=c(0,1))
```



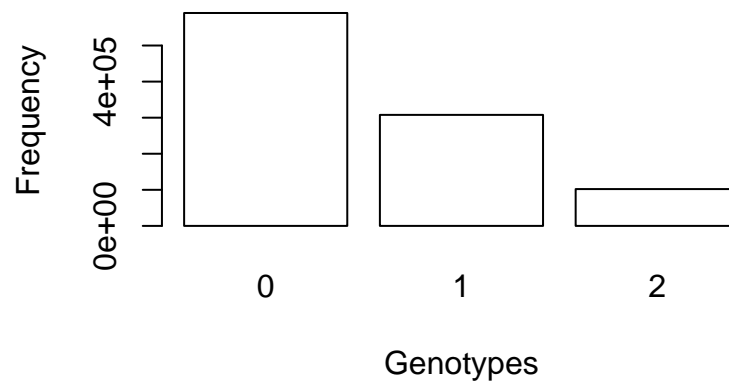
```
# distribution of intermediate population AFs
# (all subpopulations combined)
# (will be more dispersed toward 0 and 1 than ancestral AFs)
hist(B, xlab='Intermediate Subpopulation AF', main='', xlim=c(0,1))
```



```
# distribution of IAFs (admixed individuals)
# (admixture reduces differentiation, so these resemble ancestral AFs a bit more)
hist(P, xlab='Individual-specific AF', main='', xlim=c(0,1))
```



```
# genotype distribution of admixed individuals
barplot(table(X), xlab='Genotypes', ylab='Frequency', col='white')
```



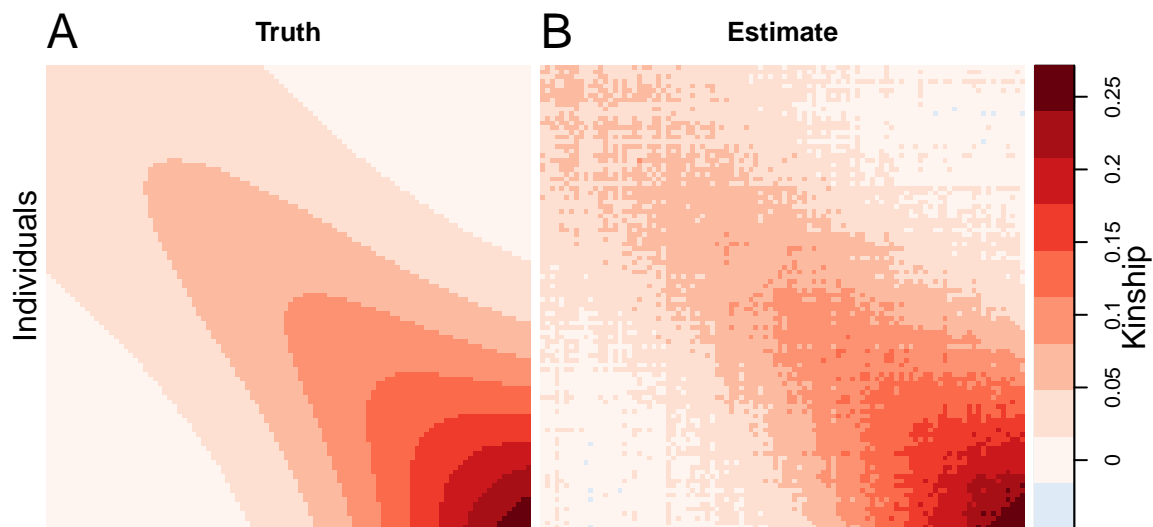
Lastly, let's verify that the correlation structure of the genotypes matches the theoretical coancestry matrix we constructed earlier. For this we use the `popkin` function of the package with the same name.

```
# for best estimates, group individuals into subpopulations using the geography
# this averages more individuals in estimating the minimum kinship
subpops <- ceiling( (1:n)/n*k )
table(subpops) # got k=10 subpops with 100 individuals each
```

```
## subpops
## 1 2 3 4 5 6 7 8 9 10
```

```
## 10 10 10 10 10 10 10 10 10 10
# now estimate kinship using popkin
PhiHat <- popkin(X, subpops)
# replace diagonal with inbreeding coeffs. to match coancestry matrix
ThetaHat <- inbrDiag(PhiHat)

# Visualize the coancestry matrix using "popkin"!
# set outer margin for axis labels (left and right are non-zero)
par(oma=c(0,1.5,0,3))
# increase inner top margin for panel titles
par(mar=c(0,0,2.5,0)+0.2)
# now plot!
x <- list(Theta, ThetaHat)
titles <- c('Truth', 'Estimate')
plotPopkin(x, titles)
```



## 2.3 Customizing the allele frequency and genotype pipeline

The random variables generated by `rbnpd` above can also be generated separately using the following functions (where  $p$  is the usual variable symbol for allele frequencies):

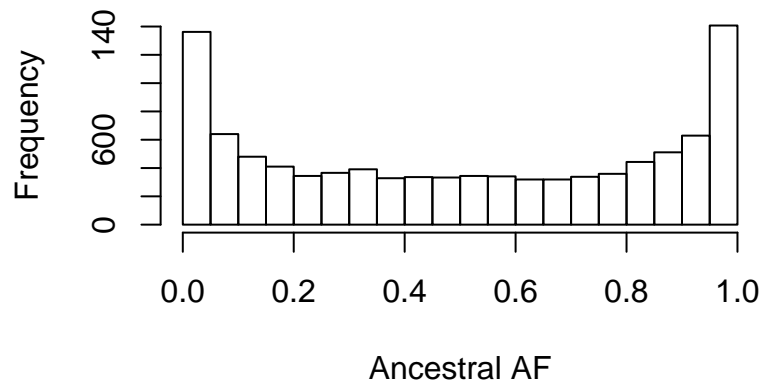
- `rpanc` (Random  $p$  ANcestral)
- `rpint` (Random  $p$  INtermediate)
- `rpiaf` (Random  $p$  Individual-specific Allele Frequency)
- `rgeno` (Random GENotypes)

Here is the step-by-step procedure for drawing AFs and genotypes in `rbnpd`:

```
# reuse the previous m,F,Q
pAnc <- rpanc(m) # draw ancestral AFs
B <- rpint(pAnc, F) # draw intermediate AFs
P <- rpiaf(B, Q) # draw IAFs (individual-specific AFs)
X <- rgeno(P) # draw genotypes
```

We provide functions for these separate steps to allow for more flexibility. For example, the ancestral allele frequencies could be drawn from a Symmetric Beta:

```
alpha <- 1/2 # this increases rare alleles
pAnc <- rbeta(m, alpha, alpha)
par(mar=c(4,4,0,0)+0.2) # shrink default margins for these figures
hist(pAnc, xlab='Ancestral AF', main='', xlim=c(0,1))
```



You could also draw genotypes from the ancestral population or the intermediate populations:

```
# draw genotypes for one individual from the ancestral population
# use "cbind" to turn the vector pAnc into the column matrix "rgeno" expects
Xanc <- rgeno(cbind(pAnc))
# returns a column matrix:
dim(Xanc)
```

```
## [1] 10000      1
```

```
# draw genotypes from intermediate populations
# draws one individual per intermediate population
Xint <- rgeno(B)
```

## 2.4 Low-memory genotype simulation algorithm

If you desire to simulate a very large number of individuals ( $n$ ) and loci ( $m$ ), you might run out of memory while running `rbnpsd`. Memory consumption is reduced by passing the `lowMem=TRUE` option to `rbnpsd`, which draws the genotypes  $X$  directly from the subpopulation AF matrix  $B$  and the admixture coefficients  $Q$ , without storing the whole IAF matrix  $P$  at any given time. However, this algorithm is much slower than the default one (`lowMem=FALSE`).

```
out <- rbnpsd(Q, F, m, lowMem=TRUE)
X <- out$X # genotypes
B <- out$B # intermediate AFs
pAnc <- out$Pa # ancestral AFs
# NOTE: out$P is not computed in this mode!
```

The same option exists for `rgeno`, which instead of accepting the IAF matrix  $P$  as input requires both  $B$  and  $Q$  as above:

```
X <- rgeno(B, Q, lowMem=TRUE)
```

### 3 Additional population structures

Here we show examples for functions that create admixture matrices for various simple population structures. The admixture scenarios implemented in **bnpsd** are generated by functions that start with **q** ( $Q$  is the admixture proportions matrix that these functions create):

- **q1d**: (Linear) 1D geography
- **q1dc**: Circular 1D geography
- **qis**: Independent Subpopulations

The first example was **q1d**, the rest follow.

#### 3.1 Circular 1D geography

This is a twist on the earlier 1D geography where subpopulations and individuals are placed on a circumference, so random walks wrap around and the appropriate density is the Von Misses distribution.

Let's generate an analogous population structure to the original "linear" example.

```
# reuse earlier (n,k) dimensions
n <- 100 # number of individuals
k <- 10 # number of intermediate subpops

# define population structure
F <- 1:k # subpopulation FST vector, up to a scalar
s <- 0.5 # desired bias coefficient
Fst <- 0.1 # desired FST for the admixed individuals
obj <- q1dc(n, k, s=s, F=F, Fst=Fst) # admixture proportions from *circular* 1D geography
Q <- obj$Q
F <- obj$F

# get pop structure parameters of the admixed individuals
Theta <- coanc(Q,F) # the coancestry matrix
# verify that we got the desired Fst!
fst(Q,F)

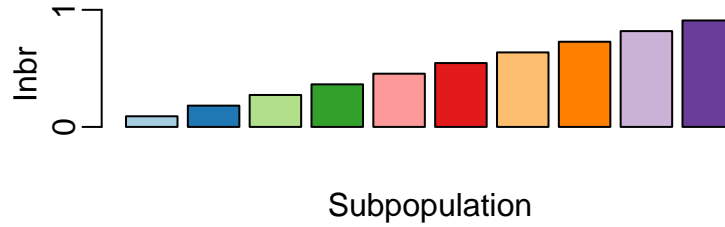
## [1] 0.1

# verify that we got the desired s too!
mean(Theta)/Fst

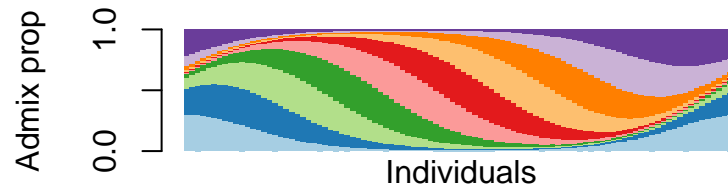
## [1] 0.5

# visualize the per-subpopulation inbreeding coefficients (FSTs)
par(mar=c(2.5,2.5,0.3,0)+0.2, lab=c(2,1,7), mgp=c(1.5,0.5,0)) # tweak margins/etc
colIS <- brewer.pal(k, "Paired") # indep. subpop. colors
barplot(F, col=colIS, names.arg=colnames(Q), xlab='Subpopulation', ylab='Inbr')
```

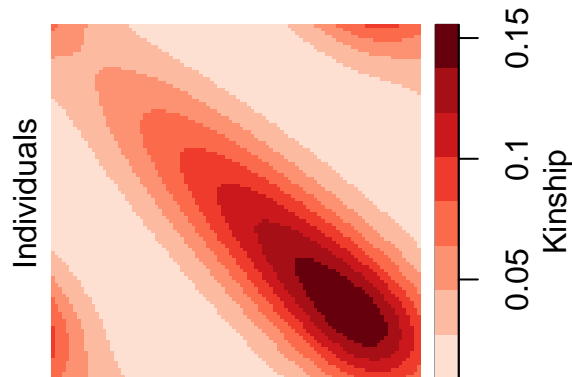




```
# visualize the admixture proportions
par(mar=c(1,4,0.4,0)+0.2, lab=c(2,2,7)) # tweak margins/etc
barplot(t(Q), col=colIS, border=NA, space=0, ylab='Admix prop')
mtext('Individuals', 1)
```



```
# Visualize the coancestry matrix using "popkin"!
par(oma=c(0,1.5,0,3), mar=c(0,0,0.4,0)+0.2) # tweak margins/etc
plotPopkin(Theta, nPretty=3)
```



### 3.2 Independent subpopulations

The independent subpopulations model, where individuals are actually unadmixed, is the most trivial form of the BN-PSD admixture model.

```
# define population structure
# we'll have k=3 subpopulations, each with these sizes:
k <- 3
n1 <- 100; n2 <- 50; n3 <- 20
# here's the labels (for simplicity, list all individuals of S1 first, then S2, then S3)
labs <- c( rep.int('S1', n1), rep.int('S2', n2), rep.int('S3', n3) )
# data dimensions inferred from labs:
length(labs) # number of individuals "n"

## [1] 170

length(unique(labs)) # number of subpopulations "k"

## [1] 3
```

```

# desired admixture matrix ("is" stands for "Independent Subpopulations")
Q <- qis(labs)
# got a boolean matrix with a single TRUE value per row
# (denoting the sole subpopulation from which each individual draws its ancestry)
head(Q, 2)

##          S1    S2    S3
## [1,] TRUE FALSE FALSE
## [2,] TRUE FALSE FALSE

# construct the intermediate subpopulation FST vector
Fst <- 0.2 # the desired final FST
F <- 1:k # subpopulation FST vector, unnormalized so far
F <- F/popkin::fst(F)*Fst # normalized to have the desired Fst
popkin::fst(F) # verify FST for the intermediate subpopulations

## [1] 0.2

# get coancestry of the admixed individuals
Theta <- coanc(Q,F) # the coancestry matrix
# before getting FST for individuals, weigh then inversely proportional to subpop sizes
w <- weightsSubpops(labs) # function from 'popkin' package
# verify Fst for individuals (same as for intermediate subpops for this pop structure)
fst(Q, F, w)

## [1] 0.2

# visualize the per-subpopulation inbreeding coefficients (FSTs)
par(mar=c(2.5,2.5,0,0)+0.2, lab=c(2,1,7), mgp=c(1.5,0.5,0)) # tweak margins/etc
colIS <- brewer.pal(k, "Paired") # indep. subpop. colors
barplot(F, col=colIS, names.arg=colnames(Q), xlab='Subpopulation', ylab='Inbr')

```



```

# visualize the admixture proportions
par(mar=c(1,4,0.4,0)+0.2, lab=c(2,2,7)) # tweak margins/etc
barplot(t(Q), col=colIS, border=NA, space=0, ylab='Admix prop')
mtext('Individuals', 1)

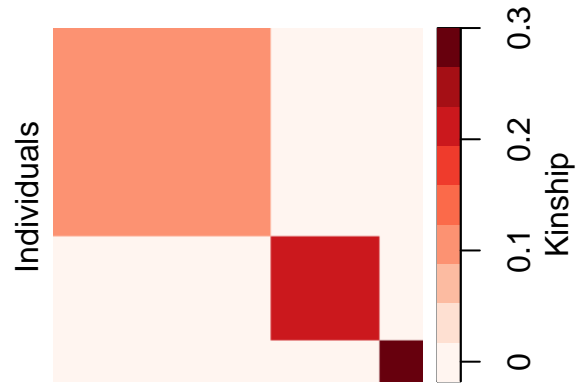
```



```

# Visualize the coancestry matrix using "popkin"!
par(oma=c(0,1.5,0,3), mar=c(0,0,0.4,0)+0.2) # tweak margins/etc
plotPopkin(Theta, nPretty=3)

```



## References

- Balding, D. J., and R. A. Nichols. 1995. "A Method for Quantifying Differentiation Between Populations at Multi-Allelic Loci and Its Implications for Investigating Identity and Paternity." *Genetica* 96 (1-2): 3–12.
- Ochoa, Alejandro, and John D. Storey. 2016a. " $F_{ST}$  And Kinship for Arbitrary Population Structures I: Generalized Definitions." *bioRxiv* doi:10.1101/083915. Cold Spring Harbor Labs Journals. <https://doi.org/10.1101/083915>.
- . 2016b. " $F_{ST}$  And Kinship for Arbitrary Population Structures II: Method of Moments Estimators." *bioRxiv* doi:10.1101/083923. Cold Spring Harbor Labs Journals. <https://doi.org/10.1101/083923>.
- Pritchard, J. K., M. Stephens, and P. Donnelly. 2000. "Inference of Population Structure Using Multilocus Genotype Data." *Genetics* 155 (2): 945–59.