

polyRAD tutorial

Lindsay V. Clark
University of Illinois, Urbana-Champaign

January 9, 2018

1 Introduction

polyRAD is an R package that assists with genotype calling from DNA sequence datasets such as genotyping-by-sequencing (GBS) or restriction site-associated DNA sequencing (RAD) in polyploids and diploids. Genotype likelihoods are estimated from allelic read depth, genotype prior probabilities are estimated from population parameters, and then genotype posterior probabilities are estimated from likelihoods and prior probabilities. Posterior probabilities can be used directly in downstream analysis, converted to weighted mean genotypes for analyses of additive genetic effects, or used for export of the most probable genotypes for analyses that require discrete genotypic data.

Analyses in polyRAD center around objects of an S3 class called “RADdata”. A single “RADdata” object contains the entire dataset of read depth and locus information, as well as parameters that are estimated during the course of analysis.

2 Summary of available functions

For any function named in this section, see its help page for more information. (For example by typing `?VCF2RADdata` into the R console.)

Several functions are available for import of read depth data and (optionally) alignment information into a RADdata object:

- `VCF2RADdata`
- `readTagDigger`
- `readStacks1`
- `readHMC`

More generally, the `RADdata` function is used for constructing RADdata objects; see the help page for that function for more information on what data are needed.

Several pipelines are available for genotype estimation, depending on how the population is structured (i.e. what the genotype prior probabilities should be.):

- `PipelineMapping2Parents`
- `IterateHWE`
- `IteratePopStruct`

Lastly, for exporting the estimated genotypes to other software:

- `ExportGAPIT`
- `Export_rrBLUP_Amat`
- `Export_rrBLUP_GWAS`

If you need continuous numerical genotypes exported in some other format, see `GetWeightedMeanGenotypes`. Also, `GetLikelyGen` returns the most likely genotypes for a single sample.

3 Estimating genotype probabilities in a mapping population

In this example, we'll import some data from an F1 mapping population of *Miscanthus sinensis* that were output by the UNEAK pipeline. These data are from a study by Liu *et al.* (2015; doi:10.1111/gcbb.12275; data available at <http://hdl.handle.net/2142/79522>), and can be found in the “doc” folder of the polyRAD installation. *Miscanthus* is an ancient tetraploid that has undergone diploidization. Given the ability of the UNEAK pipeline to filter paralogs, we expect most loci to behave in a diploid fashion, but some may behave in an allotetraploid fashion.

We'll start by loading polyRAD and importing the data into a “RADdata” object. The `possiblePloidies` argument indicates the expected inheritance modes: diploid (2) and allotetraploid (2 2).

```
> library(polyRAD)
> mydata <- readHMC("HapMapClareMap1subset.hmc.txt",
+                  possiblePloidies = list(2, c(2, 2)))
> mydata

## RADdata object ##
299 taxa and 200 loci
2711740 total reads
Assumed sample cross-contamination rate of 0.001
```

Possible ploidies:
Autodiploid (2)
Allotetraploid (2 2)

We can view the imported taxa names (subsetting here for space).

```
> GetTaxa(mydata)[c(1:10,293:299)]

[1] "IGR-2011-001"      "Kaskade-Justin"   "Map1-001"         "Map1-002"
[5] "Map1-003"         "Map1-005"         "Map1-008"         "Map1-011"
[9] "Map1-016"         "Map1-018"         "Map1-488"         "Map1-489"
[13] "Map1-490"         "Map1-491"         "Zebrinus-Justin"  "p196-150A-c"
[17] "p877-348-b"
```

All names starting with “Map” are progeny. “Kaskade-Justin” and “Zebrinus-Justin” are the parents. “IGR-2011-001”, “p196-150A-c”, and “p877-348-b” aren’t part of the population, but were doubled haploid lines that were used to screen for paralogous markers. We can tell polyRAD which taxa are the parents; since this is an F1 population it doesn’t matter which is “donor” and which is “recurrent”.

```
> mydata <- SetDonorParent(mydata, "Kaskade-Justin")
> mydata <- SetRecurrentParent(mydata, "Zebrinus-Justin")
```

Now we can run the pipeline. We’ll specify that we don’t want to include the parents or doubled haploid lines in the estimation of allele frequencies or evaluation of genotype frequencies (`excludeTaxa` argument). The `allowedDeviation` argument indicates how different the apparent allele frequency (based on read depth ratios) can be from an expected allele frequency (determined based on ploidy and mapping population type) and still be classified as that allele frequency. The default settings assume an F1 population, but the population type can be adjusted using the `n.gen.backcrossing`, `n.gen.intermating`, and `n.gen.selfing` arguments.

```
> mydata <- PipelineMapping2Parents(mydata,
+                                   freqExcludeTaxa = c("Kaskade-Justin",
+                                                       "Zebrinus-Justin",
+                                                       "IGR-2011-001",
+                                                       "p196-150A-c",
+                                                       "p877-348-b"),
+                                   freqAllowedDeviation = 0.06)
```

We can examine the allele frequencies. Allele frequencies that fall outside of the expected ranges will be recorded as NA.

```
> table(mydata$alleleFreq)

0.125 0.25 0.375 0.5 0.625 0.75 0.875
  11   89   47   94   47   89   11
```

Genotype likelihood is also stored in the object for each possible genotype at each locus, taxon, and ploidy. This is the probability of seeing the observed distribution of reads.

```
> mydata$alleleDepth[3,1:10]
```

```
TP29_0 TP29_1 TP37_0 TP37_1 TP42_0 TP42_1 TP52_0 TP52_1 TP58_0 TP58_1
      8      0      4      6      0     16      6      5     20      0
```

```
> mydata$genotypeLikelihood[[1]][,3,1:10]
```

```

      TP29_0      TP29_1      TP37_0      TP37_1      TP42_0      TP42_1
0 2.328306e-26 9.970039e-01 6.614687e-11 5.121828e-20 9.900467e-01 1.529327e-55
1 3.914069e-03 3.914069e-03 2.048729e-01 2.048729e-01 1.519787e-05 1.519787e-05
2 9.970039e-01 2.328306e-26 5.121828e-20 6.614687e-11 1.529327e-55 9.900467e-01
      TP52_0      TP52_1      TP58_0      TP58_1
0 1.126520e-19 1.091423e-13 3.171212e-63 9.950119e-01
1 2.254729e-01 2.254729e-01 9.632565e-07 9.632565e-07
2 1.091423e-13 1.126520e-19 9.950119e-01 3.171212e-63
```

```
> mydata$genotypeLikelihood[[2]][,3,1:10]
```

```

      TP29_0      TP29_1      TP37_0      TP37_1      TP42_0      TP42_1
0 2.328306e-26 9.970039e-01 6.614687e-11 5.121828e-20 9.900467e-01 1.529327e-55
1 1.544286e-05 9.997951e-02 1.465818e-01 1.622200e-02 9.942715e-03 2.347003e-10
2 3.914069e-03 3.914069e-03 2.048729e-01 2.048729e-01 1.519787e-05 1.519787e-05
3 9.997951e-02 1.544286e-05 1.622200e-02 1.465818e-01 2.347003e-10 9.942715e-03
4 9.970039e-01 2.328306e-26 5.121828e-20 6.614687e-11 1.529327e-55 9.900467e-01
      TP52_0      TP52_1      TP58_0      TP58_1
0 1.126520e-19 1.091423e-13 3.171212e-63 9.950119e-01
1 2.676630e-02 8.078123e-02 9.465741e-13 3.171212e-03
2 2.254729e-01 2.254729e-01 9.632565e-07 9.632565e-07
3 8.078123e-02 2.676630e-02 3.171212e-03 9.465741e-13
4 1.091423e-13 1.126520e-19 9.950119e-01 3.171212e-63
```

Above, for one individual (Map1-001), we see its read depth at the first ten alleles (first five loci), followed by the genotype likelihoods under diploid and tetraploid models. For example, at locus TP29 we see that homozygosity for allele 0 is the most likely, although heterozygosity is not impossible, and homozygosity for allele 1 could happen in the very, very unlikely event that all eight of the reads are due to contamination. At TP37, heterozygosity is by far the most likely state, and if there is tetraploidy there are probably one or two copies of allele 0 rather than three copies.

The prior genotype probabilities (expected genotype distributions) are also stored in the object for each possible ploidy. These distributions are estimated based on the most likely parent genotypes. Low confidence parent genotypes can be ignored by increasing the `minLikelihoodRatio` argument to `PipelineMapping2Parents`.

```
> mydata$priorProb[[1]][,1:10]
```

	TP29_0	TP29_1	TP37_0	TP37_1	TP42_0	TP42_1	TP52_0	TP52_1	TP58_0	TP58_1
0	0.25	0.25	0.0	0.5	0.0	0.5	0.5	0.0	0	0
1	0.50	0.50	0.5	0.5	0.5	0.5	0.5	0.5	1	1
2	0.25	0.25	0.5	0.0	0.5	0.0	0.0	0.5	0	0

```
> mydata$priorProb[[2]][,1:10]
```

	TP29_0	TP29_1	TP37_0	TP37_1	TP42_0	TP42_1	TP52_0	TP52_1	TP58_0	TP58_1
0	0	0	0.0	0.0	0	0	0.5	0.0	0	0
1	0	0	0.0	0.5	0	1	0.5	0.0	0	0
2	1	1	0.5	0.5	0	0	0.0	0.0	1	1
3	0	0	0.5	0.0	1	0	0.0	0.5	0	0
4	0	0	0.0	0.0	0	0	0.0	0.5	0	0

Here we see some pretty big differences under the diploid and allotetraploid models. For example, if TP29 is behaving in a diploid fashion we expect F2-like segregation since both parents were heterozygous. However, if TP29 is behaving in an allotetraploid fashion, we expect that both parents are homozygous at differing paralogous loci, resulting in no segregation.

Now we want to determine which ploidy is the best fit for each locus. This is done by comparing genotype prior probabilities to genotype likelihoods and estimating a χ^2 statistic. Lower values indicate a better fit.

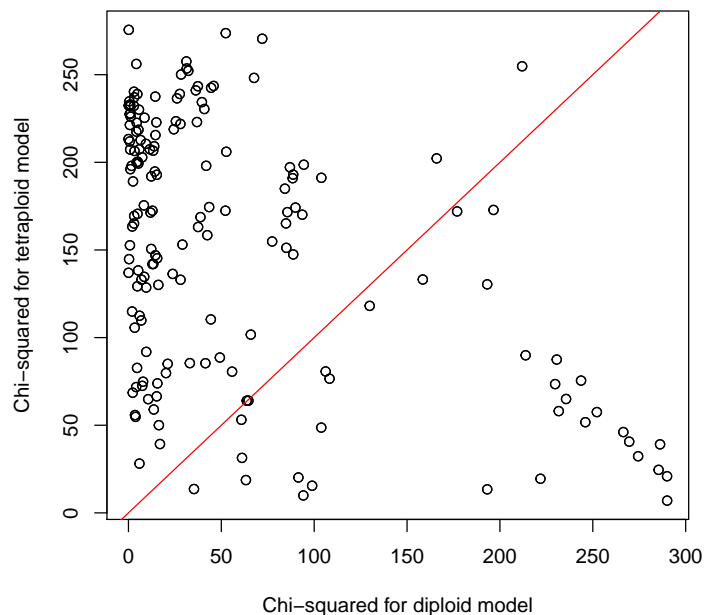
```
> mydata$ploidyChiSq[,1:10]
```

	TP29_0	TP29_1	TP37_0	TP37_1	TP42_0	TP42_1	TP52_0
[1,]	12.93418	12.93418	8.498903	8.498903	37.38438	37.38438	15.20986
[2,]	172.40358	172.40358	134.680604	134.680604	243.42034	243.42034	66.47934

	TP52_1	TP58_0	TP58_1
[1,]	15.20986	77.36922	77.36922
[2,]	66.47934	154.83699	154.83699

We can make a plot to get an overall sense of how well the markers fit the diploid versus tetraploid model.

```
> plot(mydata$ploidyChiSq[1,], mydata$ploidyChiSq[2,],
+       xlab = "Chi-squared for diploid model",
+       ylab = "Chi-squared for tetraploid model")
> abline(a = 0, b = 1, col = "red")
```



Alleles above the red line fit the diploid model better, and alleles below the red line fit the tetraploid model better.

Now we'll examine the posterior genotype probabilities. These are still estimated separately for each ploidy.

```
> mydata$posteriorProb[[1]][,3,1:10]
```

	TP29_0	TP29_1	TP37_0	TP37_1	TP42_0	TP42_1
0	2.317110e-26	9.922095e-01	0.000000e+00	2.500003e-19	0.000000e+00	1.006277e-50
1	7.790494e-03	7.790494e-03	1.000000e+00	1.000000e+00	1.000000e+00	1.000000e+00
2	9.922095e-01	2.317110e-26	2.500003e-19	0.000000e+00	1.006277e-50	0.000000e+00

	TP52_0	TP52_1	TP58_0	TP58_1
0	4.996257e-19	0.000000e+00	0	0
1	1.000000e+00	1.000000e+00	1	1
2	0.000000e+00	4.996257e-19	0	0

```
> mydata$posteriorProb[[2]][,3,1:10]
```

	TP29_0	TP29_1	TP37_0	TP37_1	TP42_0	TP42_1	TP52_0	TP52_1
0	0	0	0.00000000	0.00000000	0	0	4.208727e-18	0.000000e+00
1	0	0	0.00000000	0.07337121	0	1	1.000000e+00	0.000000e+00
2	1	1	0.92662879	0.92662879	0	0	0.000000e+00	0.000000e+00
3	0	0	0.07337121	0.00000000	1	0	0.000000e+00	1.000000e+00
4	0	0	0.00000000	0.00000000	0	0	0.000000e+00	4.208727e-18

	TP58_0	TP58_1
0	0	0
1	0	0
2	1	1
3	0	0
4	0	0

We can export the results for use in downstream analysis. The function below weights possible ploidies for each allele based on the results in `mydata$ploidyChiSq`, and for each taxon outputs a continuous, numerical genotype that is the mean of all possible genotypes weighted by genotype posterior probabilities. By default, one allele per locus is discarded in order to avoid mathematical singularities in downstream analysis. The continuous genotypes also range from zero to one by default, which can be changed with the `minval` and `maxval` arguments.

```
> mywm <- GetWeightedMeanGenotypes(mydata)
> mywm[3:6, 1:10]
```

	TP29_1	TP37_1	TP42_1	TP52_0	TP58_1	TP86_1	TP110_1
Map1-001	0.03851695	0.4989112	0.46671674	0.4534520362	0.5	0.5	0.5888637
Map1-002	0.03489354	0.0148696	0.03328326	0.4534520362	0.5	0.5	0.3695186
Map1-003	0.03489444	0.2574198	0.25000000	0.0008427731	0.5	0.5	0.3947991
Map1-005	0.96510646	0.4979527	0.46671674	0.0001482761	0.5	0.5	0.4022034

	TP111_0	TP116_1	TP154_1
Map1-001	NA	0.4927330	0.5000000
Map1-002	NA	0.4882984	0.9958151
Map1-003	NA	0.0427502	0.9879935
Map1-005	NA	0.0126574	0.9958151

Note that the parent weighted mean genotypes are not necessarily correct, since they were estimated using the genotype priors for the progeny. We can view parent genotypes with `GetLikelyGen`, which was used internally by `AdGenotypePriorProb_Mapping2Parents` to estimate the parent genotypes.

```
> kaskadeGen <- GetLikelyGen(mydata, "Kaskade-Justin")
> zebrinusGen <- GetLikelyGen(mydata, "Zebrinus-Justin")
> kaskadeGen[,1:10]
```

	TP29_0	TP29_1	TP37_0	TP37_1	TP42_0	TP42_1	TP52_0	TP52_1	TP58_0	TP58_1
2	1	1	2	0	2	0	0	2	2	0
4	2	2	4	0	4	0	0	4	4	0

```
> zebrinusGen[,1:10]
```

	TP29_0	TP29_1	TP37_0	TP37_1	TP42_0	TP42_1	TP52_0	TP52_1	TP58_0	TP58_1
2	1	1	1	1	1	1	1	1	0	2
4	2	2	1	3	2	2	1	3	0	4

To subset parent genotypes in order to match the progeny genotypes:

```
> kaskadeGenSubset <- sweep(kaskadeGen[, -OneAllelePerMarker(mydata)],
+                           1, c(2,4), "/")
> zebrinusGenSubset <- sweep(zebrinusGen[, -OneAllelePerMarker(mydata)],
+                             1, c(2,4), "/")
> kaskadeGenSubset[, 1:10]
```

	TP29_1	TP37_1	TP42_1	TP52_1	TP58_1	TP86_1	TP110_1	TP111_1	TP116_1	TP154_1
2	0.5	0	0	1	0	1	0.50	0.50	0	0.50
4	0.5	0	0	1	0	1	0.25	0.75	0	0.75

```
> zebrinusGenSubset[, 1:10]
```

	TP29_1	TP37_1	TP42_1	TP52_1	TP58_1	TP86_1	TP110_1	TP111_1	TP116_1	TP154_1
2	0.5	0.50	0.5	0.50	1	0	0.50	0.5	0.50	0.5
4	0.5	0.75	0.5	0.75	1	0	0.75	NA	0.75	0.5

4 Estimating genotype probabilities in a diversity panel

Pipelines in polyRAD for processing a diversity panel (i.e. a germplasm collection, a set of samples collected in the wild, or a panel for genome-wide association analysis or genomic prediction) use iterative algorithms. Essentially, allele frequencies are re-estimated with each iteration until convergence is reached.

Here we'll import a RAD-seq dataset from a large collection of wild and ornamental *Miscanthus* from Clark *et al.* (2014; doi:10.1093/aob/mcu084, data available at <http://hdl.handle.net/2142/49963>).

```
> mydata <- readHMC("HapMapMsiSubset1.hmc.txt",
+                  possiblePloidies = list(2, c(2,2)))
```

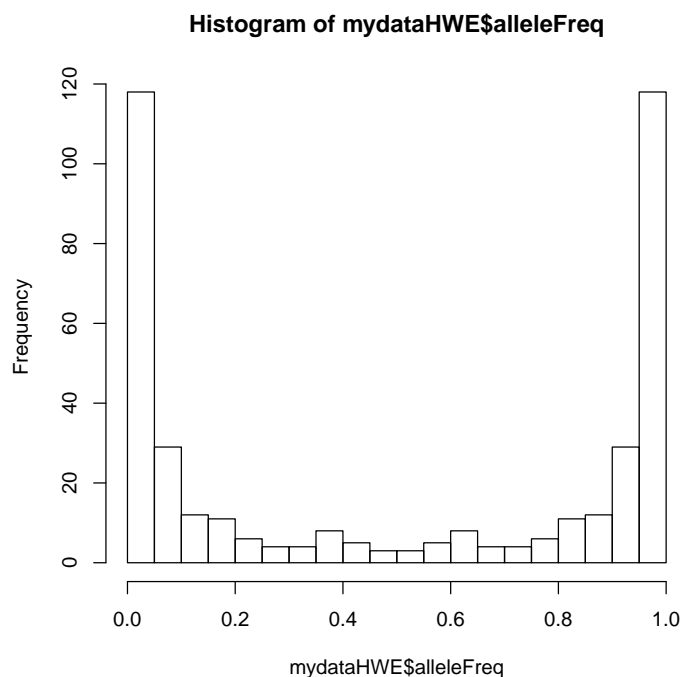
We can iteratively estimate genotype probabilities assuming Hardy-Weinberg equilibrium. The argument `tol` is set to a higher value than the default here in order to help the tutorial run more quickly.

```
> mydataHWE <- IterateHWE(mydata, tol = 1e-3)
```

```
Starting iteration 1
Mean difference in allele frequencies of 0.00883270513429876
Starting iteration 2
Mean difference in allele frequencies of 0.004211651772166
Starting iteration 3
Mean difference in allele frequencies of 0.00219754259949768
Starting iteration 4
Mean difference in allele frequencies of 0.00121707625086124
Starting iteration 5
Mean difference in allele frequencies of 0.000738011961142255
```


Let's take a look at allele frequencies:

```
> hist(mydataHWE$alleleFreq, breaks = 20)
```



There are bumps around 0.4 and 0.6, which may be artefactual due to paralogues in the dataset.

We can do a different genotype probability estimation that models population structure and variation in allele frequencies among populations. We don't need to specify populations, since principal components analysis is used to assess population structure assuming an isolation-by-distance model, with gradients of gene flow across many groups of individuals. This dataset includes a very broad sampling of *Miscanthus* across Asia, so it is very appropriate to model population structure in this case.

```
> mydataPopStruct <- IteratePopStruct(mydata)
```

```
Starting iteration 1
```

```
PCs used: 8
```

```
Mean difference in allele frequencies of 0.0160045773908589
```

```
Starting iteration 2
```

```
PCs used: 8
```

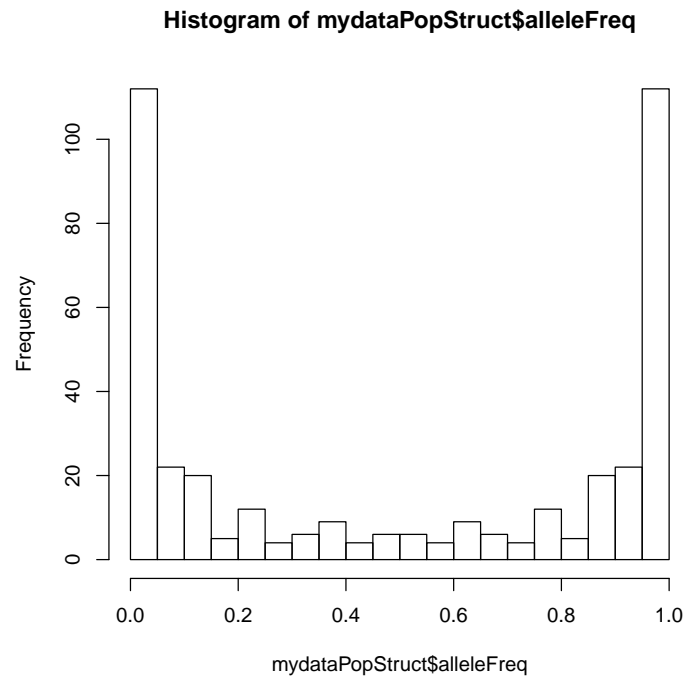
```
Mean difference in allele frequencies of 0.00685662019312055
```

```
Starting iteration 3
```

PCs used: 8
Mean difference in allele frequencies of 0.00412240974909007
Starting iteration 4
PCs used: 8
Mean difference in allele frequencies of 0.00287111965639146
Starting iteration 5
PCs used: 8
Mean difference in allele frequencies of 0.00216995917121211
Starting iteration 6
PCs used: 8
Mean difference in allele frequencies of 0.00171103111010015
Starting iteration 7
PCs used: 8
Mean difference in allele frequencies of 0.00140112655620124
Starting iteration 8
PCs used: 8
Mean difference in allele frequencies of 0.00120608600229948
Starting iteration 9
PCs used: 8
Mean difference in allele frequencies of 0.0010306684651437
Starting iteration 10
PCs used: 8
Mean difference in allele frequencies of 0.000900068029583399

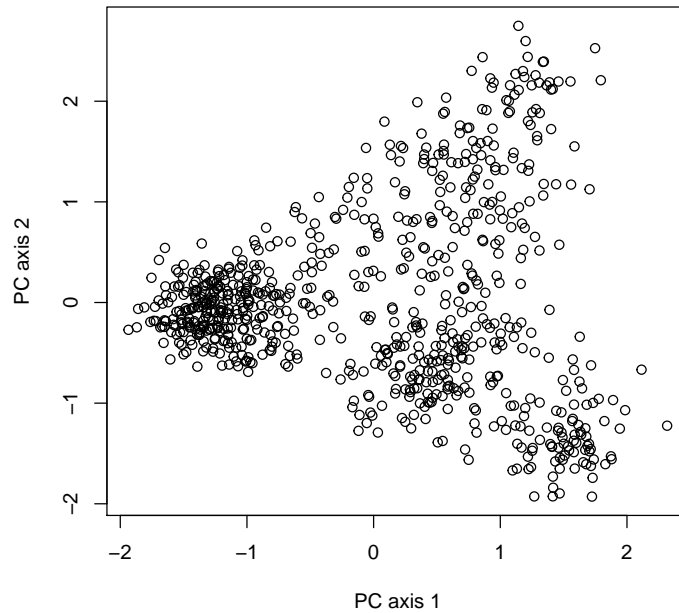
The allele frequency bumps have somewhat gone away:

```
> hist(mydataPopStruct$alleleFreq, breaks = 20)
```



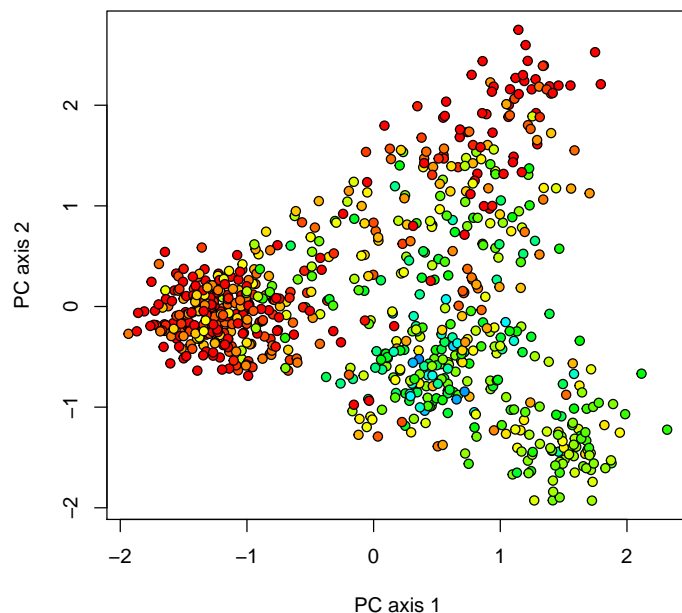
Here's some of the population structure that was used for modeling allele frequencies:

```
> plot(mydataPopStruct$PCA[,1], mydataPopStruct$PCA[,2],  
+       xlab = "PC axis 1", ylab = "PC axis 2")
```



And here's an example of allele frequency varying across the environment. Allele frequencies were estimated for each taxon, and are stored in the `$alleleFreqByTaxa` slot.

```
> myallele <- 11
> freqcol <- rainbow(101)[round(mydataPopStruct$alleleFreqByTaxa[,myallele] * 100) + 1]
> plot(mydataPopStruct$PCA[,1], mydataPopStruct$PCA[,2],
+       xlab = "PC axis 1", ylab = "PC axis 2", pch = 21,
+       bg = freqcol)
```



As before, we can export the weighted mean genotypes for downstream analysis.

```
> wmgenoPopStruct <- GetWeightedMeanGenotypes(mydataPopStruct)
> wmgenoPopStruct[1:10,1:5]
```

	TP105_0	TP111_0	TP167_0	TP210_0
2p-558-157-b	0.0001000000	1.079994e-06	9.486673e-06	3.490808e-02
Akeno	0.0001491128	1.172591e-04	4.945789e-08	8.237302e-01
CANE9233-US47-0011	0.0079570841	7.368834e-01	1.082111e-04	1.328000e-02
DC-2010-001-A	0.0081712379	2.112446e-01	2.712736e-13	4.864535e-01
DC-2010-001-E	0.0001000000	9.493695e-04	1.478490e-05	5.351421e-03
GZ-2010-007	0.0001000000	1.000000e+00	9.798027e-05	4.122422e-08
Gunma	0.0001000000	1.154582e-07	8.407692e-06	7.897973e-01
IGR-2011-001	0.0001000000	3.078391e-02	2.550028e-05	3.973039e-01
JM0001-001	0.0009202689	7.536538e-02	3.385559e-09	4.999998e-01
JM0001-002	0.0001000000	3.954945e-04	3.643582e-07	5.024183e-01
	TP234_0			
2p-558-157-b	0.0019469577			
Akeno	0.0129608969			
CANE9233-US47-0011	0.0085375805			
DC-2010-001-A	0.0135018644			
DC-2010-001-E	0.0165358844			

GZ-2010-007	0.0098879739
Gunma	0.0032421760
IGR-2011-001	0.0007070493
JM0001-001	0.0034708816
JM0001-002	0.0001000000