Scripts for populations analysis in ref_map.pl pipeline

#populations b=1 r=.9 m=5

cd /Volumes/GABE/Helen/output/refmap_m5

populations -b 1 -P . -M /Volumes/GABE/Helen/pangolin_populations.txt -r .9 -t 20 --verbose --structure --genepop --plink --write_single_snp -k --vcf --fasta

-b: Batch ID to examine when exporting from the catalog

-P: path to the Stacks output files.

-M: path specifying file describing which individuals belong to which populations (same file as Appendix S7)

-r: minimum percentage of individuals in a population required to process a locus for that population

-t: number of threads to run in parallel sections of code

--write_random_snp: restrict data analysis to one random SNP per locus

--write_single_snp: restrict data analysis to only the first SNP per locus

-k: enable kernel-smoothed FIS, π, FST, FST', and PhiST calculations

--verbose: turns on additional logging

--structure: output results in Structure format

--genepop: output results in GenePop format

--plink: output genotypes in PLINK format

--vcf: output results in Variant Call Format (VCF)

--fasta: output full sequence for each allele, from each sample locus in FASTA format

Scripts for PLINK v1.9 (Purcell et al. 2007) and missing data results

#run in BASH to handle larger files sizes

cd /Volumes/GABE/Helen/output/refmap_m5/Helen_plink/

#Removing SNPs that have 10% missing data

plink --file /Volumes/GABE/Helen/output/refmap_m5/batch_1.plink --allow-extra-chr --geno 0.1 --make-bed --out Helen_refmap_geno

#keeps any loci with less than 10% missing data

#Removing individuals with more than 15% amount of missing loci

#plink --bfile Helen_refmap_geno --allow-extra-chr --mind 0.15 --make-bed --out Helen_refmap_genomind

#Pruning SNPs that are strongly genetically linked

#plink --bfile Helen_refmap_genomind --allow-no-sex --allow-extra-chr --indep-pairwise 25 10 0.9

#Extracting SNP data

#plink --file /Volumes/GABE/Helen/output/refmap_m5/batch_1.plink --extract plink.prune.in --allow-extra-chr --make-bed --out Helenpruned

#this command has to come from the original files

#Formating output data in structure format

#plink --bfile Helenpruned --allow-extra-chr --recode structure --out Helen_refmap_plinkoutput

#Summary of missing data

#plink --bfile Helenpruned --allow-extra-chr --missing --allow-no-sex

##Commands

#--file: takes a single parameter, the root of the input file names, and will look MAP and PED files

#--geno: maximum per-SNP missing

#--mind: Maximum per-person missing: removes individuals with more than threshold amount of missing loci

#--recode-structure

#missing data results for 83 Sunda pangolin samples. PLINK output below is the IMISS file.

Family ID	Individual ID	Missing phenotype Y/N	N_MISS	N_GENO	F_MISS
2	MZBR_0270	Y	73	12150	0.006008
2	MZBR_0271	Y	122	12150	0.01004
2	MZBR_0272	Y	65	12150	0.00535
2	MZBR_0273	Y	33	12150	0.002716
2	MZBR_0274	Y	103	12150	0.008477
2	MZBR_0275	Y	119	12150	0.009794
2	MZBR_0276	Y	45	12150	0.003704
2	MZBR_1030	Y	81	12150	0.006667
2	MZBR_1031	Y	77	12150	0.006337
2	MZBR_1032	Y	124	12150	0.01021
2	MZBR_1033	Y	79	12150	0.006502
2	MZBR_1034	Y	27	12150	0.002222
2	MZBR_1035	Y	109	12150	0.008971
2	MZBR_1036	Y	81	12150	0.006667
2	MZBR_1037	Y	995	12150	0.08189
2	MZBR_1038	Y	44	12150	0.003621
2	MZBR_1040	Y	18	12150	0.001481
2	MZBR_1041	Y	28	12150	0.002305
2	MZBR_1042	Y	20	12150	0.001646
2	MZBR_1043	Y	18	12150	0.001481
2	MZBR_1044	Y	21	12150	0.001728
2	MZBR_1045	Y	44	12150	0.003621
2	MZBR_1047	Y	169	12150	0.01391
2	MZBR_1048	Y	373	12150	0.0307
2	MZBR_1049	Y	44	12150	0.003621
2	MZBR_1051	Y	173	12150	0.01424
2	MZBR_1052	Y	371	12150	0.03053
2	MZBR_1053	Y	20	12150	0.001646
2	MZBR_1054	Y	28	12150	0.002305
2	MZBR_1055	Y	14	12150	0.001152
2	MZBR_1056	Y	74	12150	0.006091
2	MZBR_1060	Y	80	12150	0.006584
2	MZBR_1062	Y	71	12150	0.005844
2	MZBR_1063	Y	73	12150	0.006008
2	MZBR_1065	Y	51	12150	0.004198
2	MZBR_1066	Y	101	12150	0.008313
2	MZBR_1067	Y	1267	12150	0.1043
2	MZBR_1068	Y	108	12150	0.008889
2	MZBR_1069	Y	68	12150	0.005597
2	MZBR_1070	Y	54	12150	0.004444
2	MZBR_1071	Y	232	12150	0.01909
2	MZBR_1072	Y	58	12150	0.004774
2	MZBR_1073	Y	55	12150	0.004527
2	MZBR_1074	Y	65	12150	0.00535
2	MZBR_1075	Y	469	12150	0.0386

2	MZBR_1076	Y	25	12150	0.002058
2	MZBR_1078	Y	36	12150	0.002963
2	MZBR_1079	Y	77	12150	0.006337
2	MZBR_1080	Y	50	12150	0.004115
2	MZBR_1081	Y	406	12150	0.03342
2	MZBR_1082	Y	103	12150	0.008477
2	MZBR_1083	Y	38	12150	0.003128
2	MZBR_1085	Y	32	12150	0.002634
2	MZBR_1087	Y	25	12150	0.002058
2	MZBR_1088	Y	128	12150	0.01053
2	MZBR_1157	Y	36	12150	0.002963
2	MZBR_1159	Y	52	12150	0.00428
2	MZBR_1160	Y	30	12150	0.002469
2	MZBR_1161	Y	96	12150	0.007901
2	MZBR_1162	Y	468	12150	0.03852
2	MZBR_1163	Y	32	12150	0.002634
2	MZBR_1164	Y	377	12150	0.03103
2	MZBR_1166	Y	503	12150	0.0414
2	MZBR_1167	Y	37	12150	0.003045
2	MZBR_1177	Y	474	12150	0.03901
2	MZBR_1179	Y	37	12150	0.003045
2	MZBR_1182	Y	861	12150	0.07086
2	MZBR_1183	Y	548	12150	0.0451
2	MZBR_1184	Y	608	12150	0.05004
2	MZBR_1185	Y	299	12150	0.02461
2	MZBR_1186	Y	607	12150	0.04996
2	MZBR_1187	Y	68	12150	0.005597
2	MZBR_1188	Y	28	12150	0.002305
2	MZBR_1189	Y	31	12150	0.002551
2	MZBR_1190	Y	49	12150	0.004033
3	rescue_1	Y	3	12150	0.0002469
3	rescue_2	Y	3	12150	0.0002469
3	rescue_3	Y	3	12150	0.0002469
3	rescue_5a	Y	3	12150	0.0002469
3	rescue_6b	Y	3	12150	0.0002469
3	rescue_7b	Y	3	12150	0.0002469
3	rescue_8b	Y	3	12150	0.0002469
4	Malaysian	Y	887	12150	0.073

Parameters and Results of BayeScan v2.1 (Foll & Gaggiotti 2008)

The three populations referred to in our 12150SNPs_Verif file output below included Singaporean versus non-Singaporean Sunda pangolins, and the Chinese pangolin:

Summary of parameters and input files.

There are 12150 loci.

There are 3 populations.

Burn in: 50000

Thining interval: 10

Sample size: 5000

Resulting total number of iterations: 100000

Nb of pilot runs: 20

Length of each pilot run: 5000

Allele counts:

Pop. 1 locus 1 : 1 149

Pop. 1 locus 2 : 1 149

Pop. 1 locus 3 : 9 139

Pop. 1 locus 4 : 38 112

Pop. 1 locus 5 : 59 87

Pop. 1 locus 6 : 11 139

Pop. 1 locus 7 : 3 147

Pop. 2 locus 1 : 0 14

Pop. 2 locus 2 : 0 14

Pop. 2 locus 3 : 0 14

Pop. 2 locus 4 : 0 14

Pop. 2 locus 5 : 6 8

Pop. 2 locus 6 : 0 14

Pop. 2 locus 7 : 0 14

Pop. 3 locus 1 : 0 2

Pop. 3 locus 2 : 0 2

Pop. 3 locus 3 : 0 2

Pop. 3 locus 4 : 0 2

Pop. 3 locus 5 : 1 1

Pop. 3 locus 6 : 0 2

Pop. 3 locus 7 : 0 2

#And so on for all 12150 loci

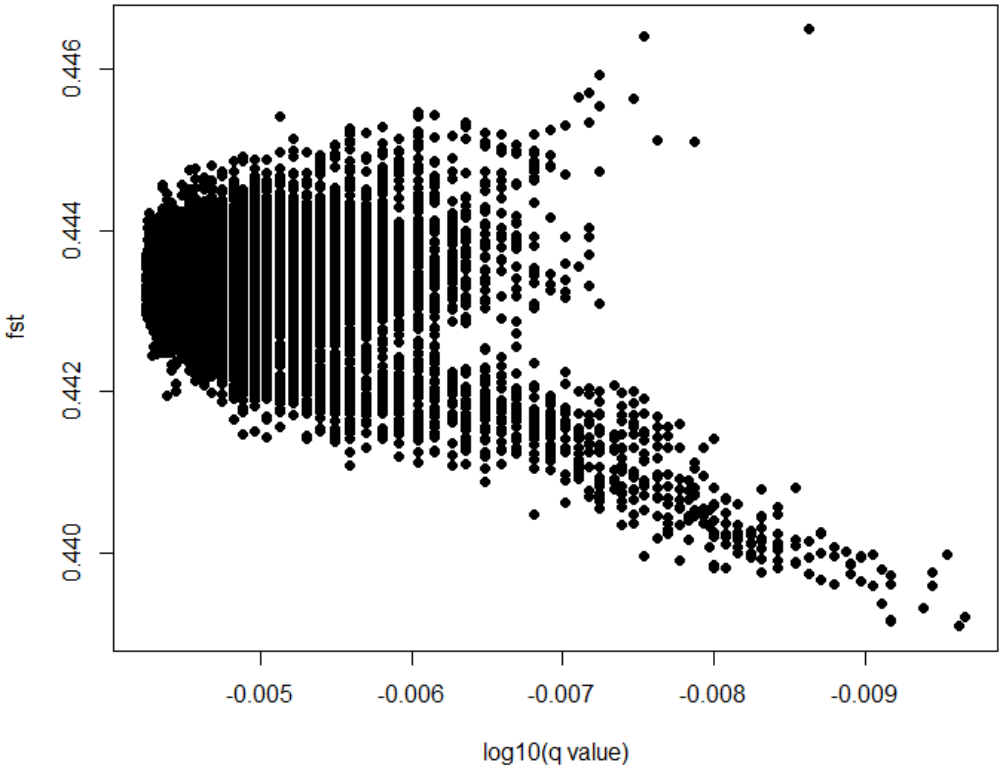
#Next we used plot_R to check for selection with the 12150SNPs_fst file. Instructions below.

```
setwd("C:\\Users\\Helen\\Desktop\\RADseq\\Helen_plink2\\thin8191\\GESTE")
#Then click on the R console.
#Next click on File in top menu.
#Then click Source R code and find your plot_R (This was downloaded when you downloaded
#Bayescan)Mine was saved here:
#C:\Users\Helen\Desktop\RADseq\Helen_plink2\thin8191\GESTE\BayeScan2.1\BayeScan2.1\R
#functions

plot_bayescan("Helen12150SNPs_fst.txt",FDR=0.05)
#Lists the outliers having a q-value lower than 5% and produces a figure.

#My results stated
#$outliers
#Integer(0)

#$nb_outliers
#[1] 0
```



#Finally, we also checked in plot_R that our model converged. Scripts below.

install.packages("coda")

library(coda)

getwd()

setwd("C:\\Users\\Helen\\Desktop\\RADseq\\Helen_plink2\\thin8191\\GESTE")

chain<-read.table("Helen12150SNPs.sel",header=TRUE)

chain<-chain[-c(1)]

chain<-mcmc(chain,thin=10)

plot(chain)

To check for convergence we used Geweke's convergence diagnostic based on the comparison of #the means of the first and last parts of a Markov chain. The diagnostic reports the z - scores for #each parameter. For example, with α = 0 .05, the critical values of z are - 1.96 and +1.96. We #reject H 0 (equality of means => convergence) if z < - 1.96 or z > +1.96

geweke.diag(chain,frac1=0.1,frac=0.5)

PCA Scripts and Fst Estimation in SNPrelate and Results

PCA

We used the SNPrelate package in R v3.2 (R Core Team, 2016) to compare principal components of 12150 SNPs across 83 Sunda pangolin samples. The default settings of SNPRelate removed 223 loci prior to Principal Components Analysis (PCA) of 11927 loci.

Our PCA results (below) suggest there are three distinct genetic clusters across our Sunda pangolins, Sumatra/Singapore, Java, and Borneo. The wild pangolin samples of known origin were used to identify the origin of each cluster. The first two eigenvectors of PCA held the largest percentage of variance among the population, principal component 1 = 4.47 % and principal component 2 = 4.38 %.

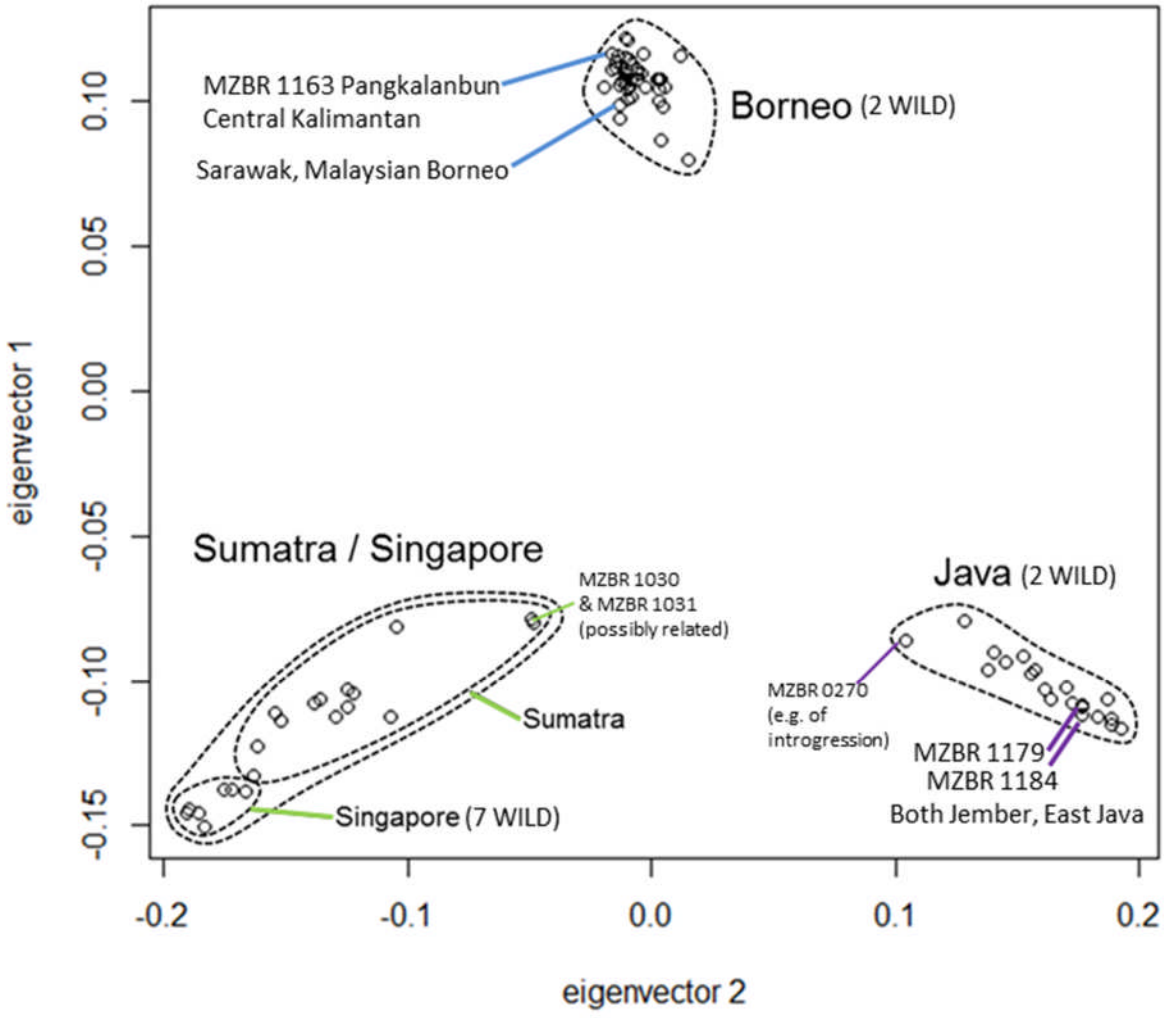


Figure 1 PCA of 11927 SNPs across 83 Sunda pangolin samples. The three genetic clusters are bottom left Sumatra/Singapore, bottom right Java, and centre top Borneo.

We downloaded the latest version of R, available at:

https://cran.r-project.org/bin/windows/base/

R 3.2.4 for Windows (62 megabytes, 32/64 bit)

PCA Scripts are given below

```
source("http://bioconductor.org/biocLite.R")
biocLite("gdsfmt")
biocLite("SNPRelate")

library(gdsfmt)
library(SNPRelate)

bed.fn <- "C:/Users/Helen/Desktop/RADseq/Helen_plink2/Helenpruned3.bed"

fam.fn <- "C:/Users/Helen/Desktop/RADseq/Helen_plink2/Helenpruned3.fam"

bim.fn <- "C:/Users/Helen/Desktop/RADseq/Helen_plink2/Helenpruned3.bim"

setwd("C:\\Users\\Helen\\Desktop\\RADseq\\Helen_plink2\\ ")

snpgdsBED2GDS(bed.fn, fam.fn, bim.fn, "test.gds")

snpgdsSummary("test.gds")

genofile <- snpgdsOpen("test.gds")

pop_code <- scan("pangolin_populations4a.txt", what=character())

#Our populations text file was a simple one column list of species i.e. Chinese or Sunda. The
#order is important. NB. Later I had to edit the order to match the sample_ID order in R. See
#later note.

head(pop_code)

pca <- snpgdsPCA(genofile, autosome.only=FALSE)

#NB the autosome,only=FALSE command is important.
```

#Next calculate the percent of variation which is accounted for by the principal component
#for the first 16 PCs.

```
pc.percent <- pca$varprop*100
head(round(pc.percent, 2))
```

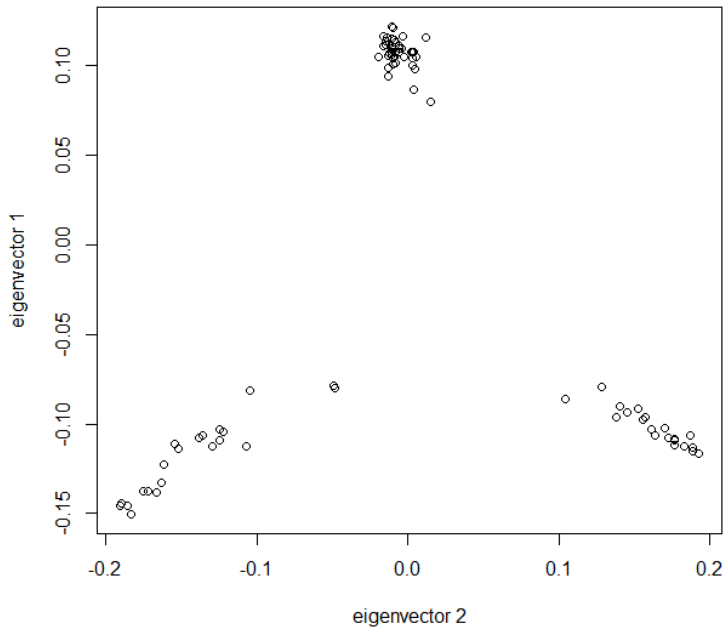
#[1] 4.47 4.38 2.65 2.52 2.37 2.01

#In our case, the first two eigenvectors hold the largest percentage of variance among the
#population, although the total variance accounted for is still less than one-tenth of the
#total.

#Next plot the results of the first and second eigenvectors.

```
tab <- data.frame(sample.id = pca$sample.id,
EV1 = pca$eigenvect[,1],
EV2 = pca$eigenvect[,2],
stringsAsFactors = FALSE)
head(tab)
#  sample.id        EV1       EV2
#1 MZBR_0270 -0.08602192  0.1044291
#2 MZBR_0271 -0.10755401 -0.1382413
#3 MZBR_0272 -0.10608550 -0.1355269
#4 MZBR_0273 -0.11392056 -0.1521897
#5 MZBR_0274 -0.13271594 -0.1630330
#6 MZBR_0275 -0.11087483 -0.1546654

plot(tab$EV2, tab$EV1, xlab="eigenvector 2", ylab="eigenvector 1")
```



From the PCA results we are able to look at the geographic assignment of each sample, using the wild samples of known origin to determine where each of the three clusters originated.

Samples highlighted in yellow below group in the Java cluster. Samples highlighted in turquoise or grey are the Sumatra/Singapore cluster. All others samples not highlighted are the Borneo cluster.

	sample.id	pop	EV1	EV2
1	MZBR_0270	Indonesia	-0.08602192	0.104429142
2	MZBR_0271	Indonesia	-0.10755401	-0.138241310
3	MZBR_0272	Indonesia	-0.10608550	-0.135526851
4	MZBR_0273	Indonesia	-0.11392056	-0.152189721
5	MZBR_0274	Indonesia	-0.13271594	-0.163032982
6	MZBR_0275	Indonesia	-0.11087483	-0.154665422
7	MZBR_0276	Indonesia	-0.12261905	-0.161247907
8	MZBR_1030	Indonesia	-0.07856239	-0.049550920
9	MZBR_1031	Indonesia	-0.07978341	-0.048551287
10	MZBR_1032	Indonesia	-0.10303572	-0.124318403
11	MZBR_1033	Indonesia	-0.10901740	-0.124781135
12	MZBR_1034	Indonesia	0.11285827	-0.008141493
13	MZBR_1035	Indonesia	0.10767100	0.001896185
14	MZBR_1036	Indonesia	0.10761672	0.003153126
15	MZBR_1037	Indonesia	0.08676132	0.003557393
16	MZBR_1038	Indonesia	0.11602484	0.012167382
17	MZBR_1040	Indonesia	0.09847476	0.004382110
18	MZBR_1041	Indonesia	0.10925106	-0.009971440
19	MZBR_1042	Indonesia	0.11245910	-0.012386331
20	MZBR_1043	Indonesia	0.10959786	-0.005863153
21	MZBR_1044	Indonesia	0.10970435	-0.010799891

22	MZBR_1045	Indonesia	0.10532990	0.005285829
23	MZBR_1047	Indonesia	0.10504013	-0.002804297
24	MZBR_1048	Indonesia	0.10188779	-0.008077858
25	MZBR_1049	Indonesia	0.10696341	-0.012450233
26	MZBR_1051	Indonesia	0.10768684	-0.005574045
27	MZBR_1052	Indonesia	0.09403196	-0.012927689
28	MZBR_1053	Indonesia	0.10436063	0.003057642
29	MZBR_1054	Indonesia	0.11130873	-0.016435277
30	MZBR_1055	Indonesia	0.11442609	-0.009305769
31	MZBR_1056	Indonesia	0.11574367	-0.013794139
32	MZBR_1060	Indonesia	-0.11662558	0.192593151
33	MZBR_1062	Indonesia	-0.10442033	-0.122078961
34	MZBR_1063	Indonesia	-0.11515194	0.188589247
35	MZBR_1065	Indonesia	-0.11221374	-0.129506860
36	MZBR_1066	Indonesia	-0.10826036	0.176549049
37	MZBR_1067	Indonesia	-0.08159705	-0.104809420
38	MZBR_1068	Indonesia	-0.09637061	0.138008730
39	MZBR_1069	Indonesia	-0.09011350	0.140124497
40	MZBR_1070	Indonesia	0.08014680	0.015060651
41	MZBR_1071	Indonesia	-0.10306580	0.160939062
42	MZBR_1072	Indonesia	-0.09316255	0.144989060
43	MZBR_1073	Indonesia	0.11650718	-0.003749093
44	MZBR_1074	Indonesia	0.10768580	0.004207347
45	MZBR_1075	Indonesia	-0.07935799	0.127970295
46	MZBR_1076	Indonesia	-0.09623861	0.157175403
47	MZBR_1078	Indonesia	-0.11249796	-0.106669934
48	MZBR_1079	Indonesia	0.11122040	-0.005794797
49	MZBR_1080	Indonesia	0.12157482	-0.009762326

50 MZBR_1081 Indonesia 0.10094382 -0.009916011

51 MZBR_1082 Indonesia 0.10516492 -0.019456217

52 MZBR_1083 Indonesia 0.11383098 -0.014656412

53 MZBR_1085 Indonesia 0.10493065 -0.009416701

54 MZBR_1087 Indonesia 0.10749774 -0.010914269

55 MZBR_1088 Indonesia 0.10782964 -0.008508207

56 MZBR_1157 Indonesia 0.11089224 -0.010450285

57 MZBR_1159 Indonesia 0.11168945 -0.014296440

58 MZBR_1160 Indonesia 0.12179242 -0.010504054

59 MZBR_1161 Indonesia 0.10561840 -0.012994002

60 MZBR_1162 Indonesia 0.10961390 -0.004400338

61 WILD MZBR_1163 Indonesia 0.11671589 -0.016099718 3rd from left of Borneo cluster, and 3rd from top.

62 MZBR_1164 Indonesia 0.10427482 -0.009935305

63 MZBR_1166 Indonesia 0.10641787 -0.010970644

64 MZBR_1167 Indonesia 0.11498112 -0.010691785

65 MZBR_1177 Indonesia 0.10060050 0.003217525

66 WILD MZBR_1179 Indonesia -0.11166323 0.176303312 8th from the right

67 MZBR_1182 Indonesia -0.09750466 0.155381829

68 MZBR_1183 Indonesia -0.09150629 0.152572660

69 WILD MZBR_1184 Indonesia -0.10875386 0.176843337 6th from the right

70 MZBR_1185 Indonesia -0.11297747 0.188117822

71 MZBR_1186 Indonesia -0.10209396 0.170234489

72 MZBR_1187 Indonesia -0.10652355 0.163292582

73 MZBR_1188 Indonesia -0.11214581 0.182968213

74 MZBR_1189 Indonesia -0.10799604 0.172334892

75 MZBR_1190 Indonesia -0.10650033 0.186523041

76 rescue_1 Singapore -0.13835237 -0.166784345

77 rescue_2 Singapore -0.13727178 -0.171887596

78 rescue_3 Singapore -0.14577717 -0.190253583

79 rescue_5a Singapore -0.14411055 -0.189399463

80 rescue_6b Singapore -0.14582817 -0.185701213

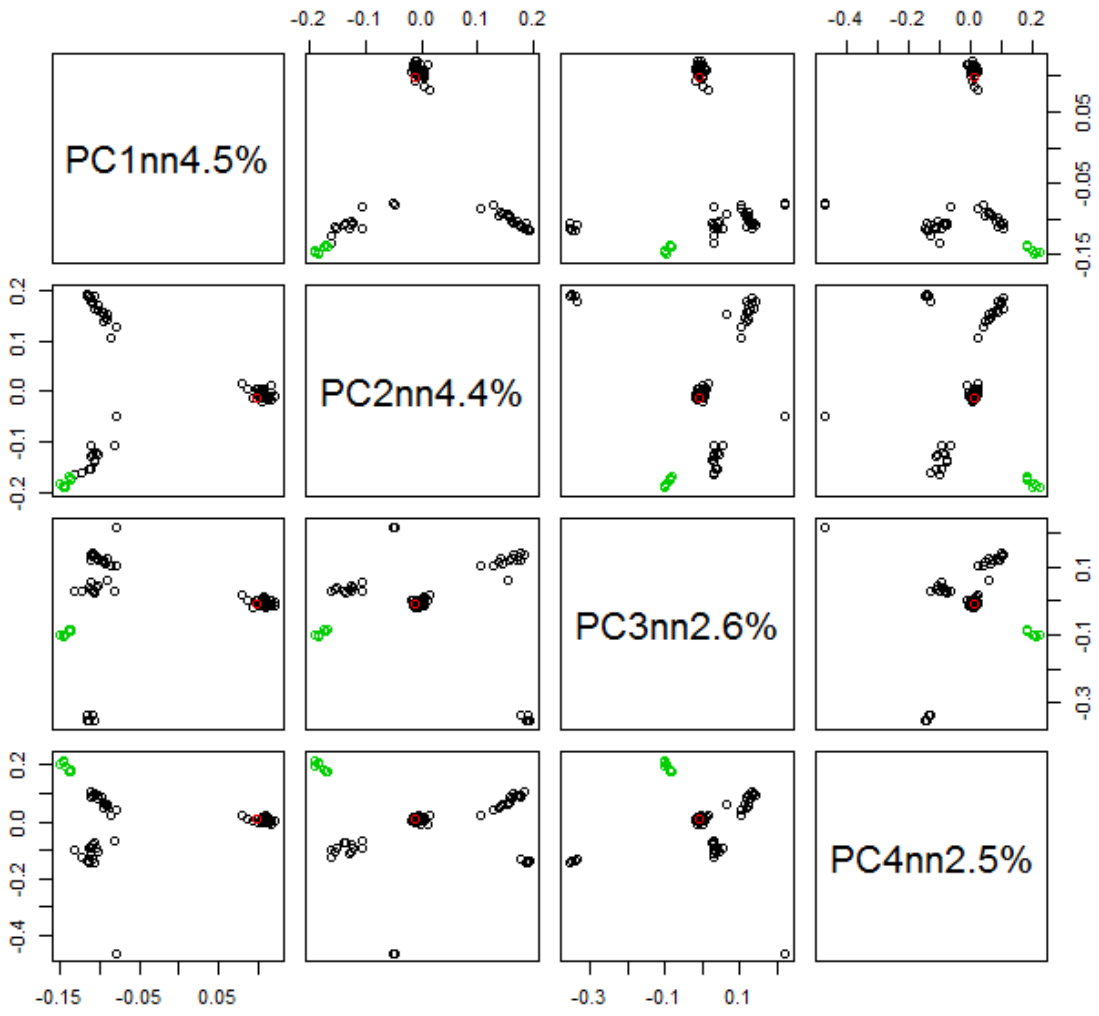
81 rescue_7b Singapore -0.13739473 -0.175612623

82 rescue_8b Singapore -0.15008233 -0.183354888

83 WILD Sarawak Malaysia 0.09862128 -0.012711965

we also checked clustering arrangements using the top 32 eigenvectors.

```
chr <- read.gdsn(index.gdsn(genofile, "snp.chromosome"))
CORR <- snpgdsPCACorr(pca, genofile, eig.which=1:4)
```



Scripts for Fst Estimation in SNPRelate

We downloaded R version 3.4.3 (2017-11-30) -- "Kite-Eating Tree".

Next in R we used the scripts below for installation of packages:

```
source("http://bioconductor.org/biocLite.R")
biocLite("gdsfmt")
biocLite("SNPRelate")
```

Then the scripts below to load the packages:

```
library(gdsfmt)
library(SNPRelate)
```

Then we created a genofile using the bed, fam and bim files from our PLINK output of our earlier STACKS pipeline (see manuscript).

```
bed.fn <- "C:/Users/Helen/Desktop/SNPRelateFst/Helenpruned3.bed"

fam.fn <- "C:/Users/Helen/Desktop/SNPRelateFst/Helenpruned3.fam"

bim.fn <- "C:/Users/Helen/Desktop/SNPRelateFst/Helenpruned3.bim"


snpgdsBED2GDS(bed.fn, fam.fn, bim.fn, "test.gds")


snpgdsSummary("test.gds")


genofile <- snpgdsOpen("test.gds")
```

For Fst estimation using the genofile, the scripts are given below:

```
sample.id <- read.gdsn(index.gdsn(genofile, "sample.id"))

pop_code <- scan("pop_group.txt", what=character())

#the pop_group.txt file is a simple list of the population codes of each
sample, i.e. Borneo, Java, or SumSin


#For pairwise Fst you can ask the programme to compare just two specific
population labels, e.g. Borneo and Java.

flag <- pop_code %in% c("Borneo", "Java")

samp.sel <- sample.id[flag]

pop.sel <- pop_code[flag]

snpgdsFst(genofile, sample.id=samp.sel, population=as.factor(pop.sel),

 method="W&C84", autosome.only=FALSE)
```

Fst Results

Method: Weir & Cockerham, 1984 in SNPRelate

Borneo versus Java

62 samples, 9,313 SNPs

Borneo (42), Java (20)

# of Populations: 2

Fst = 0.1350858

MeanFst = 0.05561589

Borneo versus Sumatra/Singapore

63 samples, 10,618 SNPs

Borneo (42), Sum_Sin (21)

# of Populations: 2

Fst = 0.1105821

MeanFst = 0.04456568

Java versus Sumatra/Singapore

41 samples, 7,816 SNPs

Java (20), Sum_Sin (21)

# of Populations: 2

Fst = 0.1635602

MeanFst = 0.0683543

Across all populations

83 samples, 11,927 SNPs

# of Populations: 3

Borneo (42), Java (20), Sum_Sin (21)

Fst = 0.130758

MeanFst = 0.05447444

We also looked at the Fst Results if an introgressed sample, MZBR 0270, was removed from the sample set and Fst increased when this sample was removed:

#you can simply change the name of a sample in the pop_group.txt file so that it gets excluded from analysis.

Borneo versus Java

61 samples, 9,163 SNPs

# of Populations: 2

Borneo (42), Java (19)

Fst = 0.1391943

MeanFst = 0.05750187

Borneo versus Sumatra/Singapore

63 samples, 10,618 SNPs

# of Populations: 2

Borneo (42), Sum_Sin (21)

Fst = 0.1105821

MeanFst = 0.04456568

Java versus Sumatra/Singapore

40 samples, 7,658 SNPs

# of Populations: 2

Java (19), Sum_Sin (21)

Fst = 0.168714

MeanFst = 0.07108101

Across all populations

82 samples, 11,801 SNPs

# of Populations: 3

Borneo (42), Java (19), Sum_Sin (21)

Fst = 0.1329475

MeanFst = 0.05562351

Structure Method

For STRUCTURE (Pritchard et al. 2000), we tested K = 1 to K = 7 to investigate whether there might be 1 to 7 genetic clusters across our 83 samples, and we used 5 iterations for each value of K. We applied a burn-in of 100,000 followed by 500,000 MCMC replications, with an admixture model of ancestry, and correlated allele frequencies. We used Structure Harvester Web version 0.6.94 (Earl & vonHoldt 2012) and the Evanno method (Evanno et al. 2005) to determine the most likely value of K.

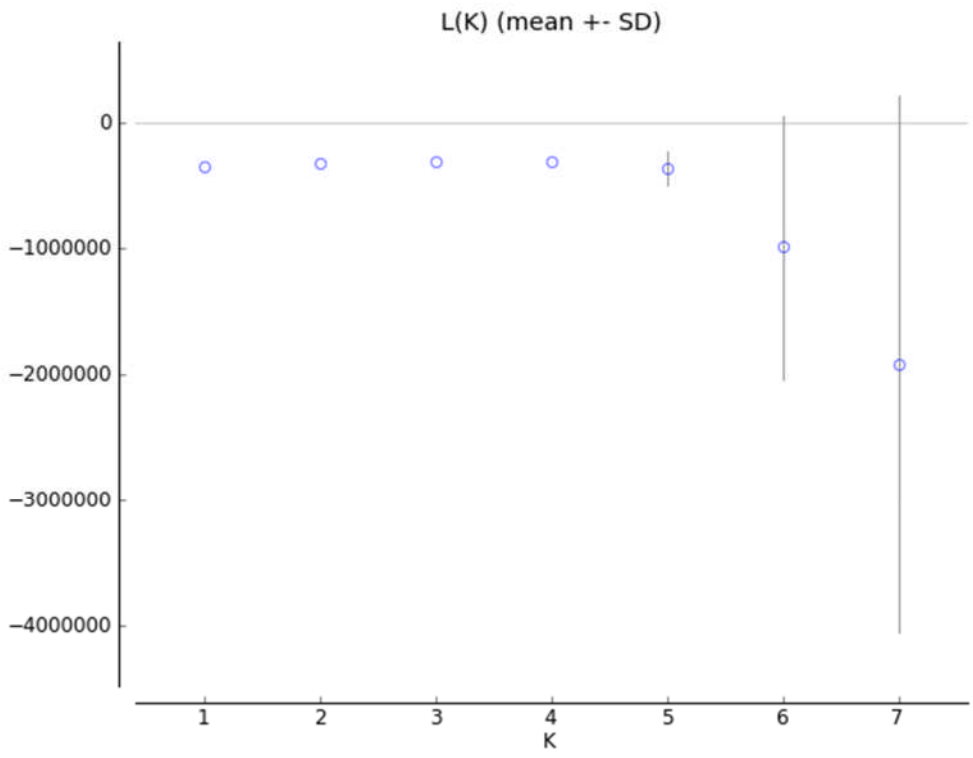
Structure Harvester Results

We used Structure Harvester Web v0.6.94 (Earl & vonHoldt 2012).

Across our Structure outputs for K = 1 to K = 7, DeltaK and the Evanno method suggest the most likely value of K = 3, which means three distinct genetic population clusters are most likely across our sample set according to these methodologies.

Single file archive including this page, all images, all clumpp files: download. [.tar.gz]

L(K)



L(K): pdf eps

Clumpp files

K = 1 Clumpp indfile K = 1 Clumpp popfile

K = 2 Clumpp indfile K = 2 Clumpp popfile

K = 3 Clumpp indfile K = 3 Clumpp popfile

K = 4 Clumpp indfile K = 4 Clumpp popfile

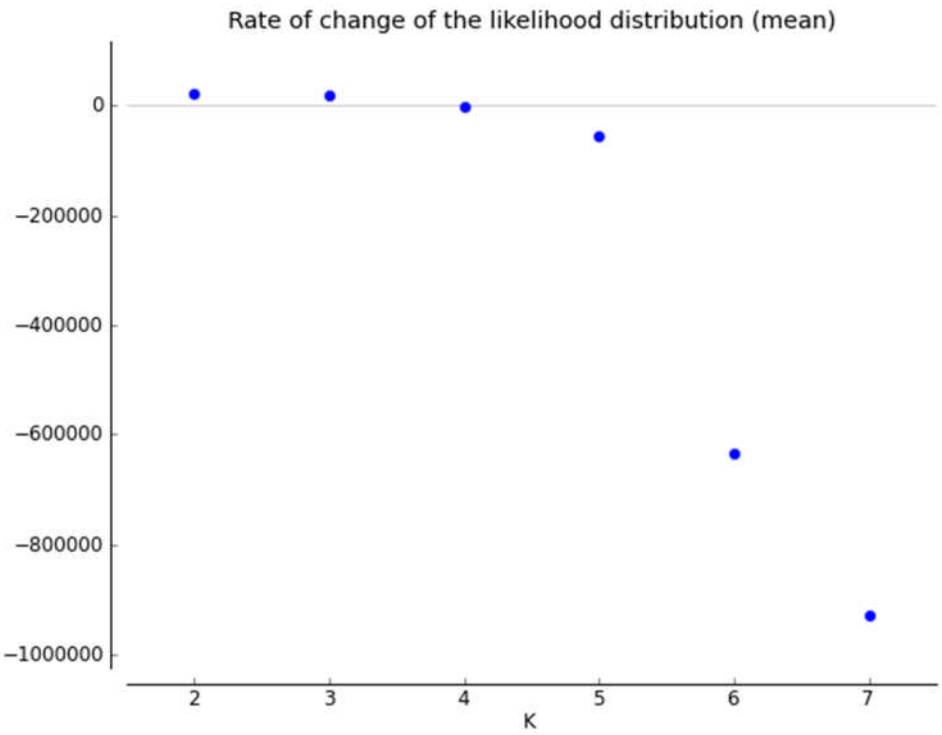
K = 5 Clumpp indfile K = 5 Clumpp popfile

K = 6 Clumpp indfile K = 6 Clumpp popfile

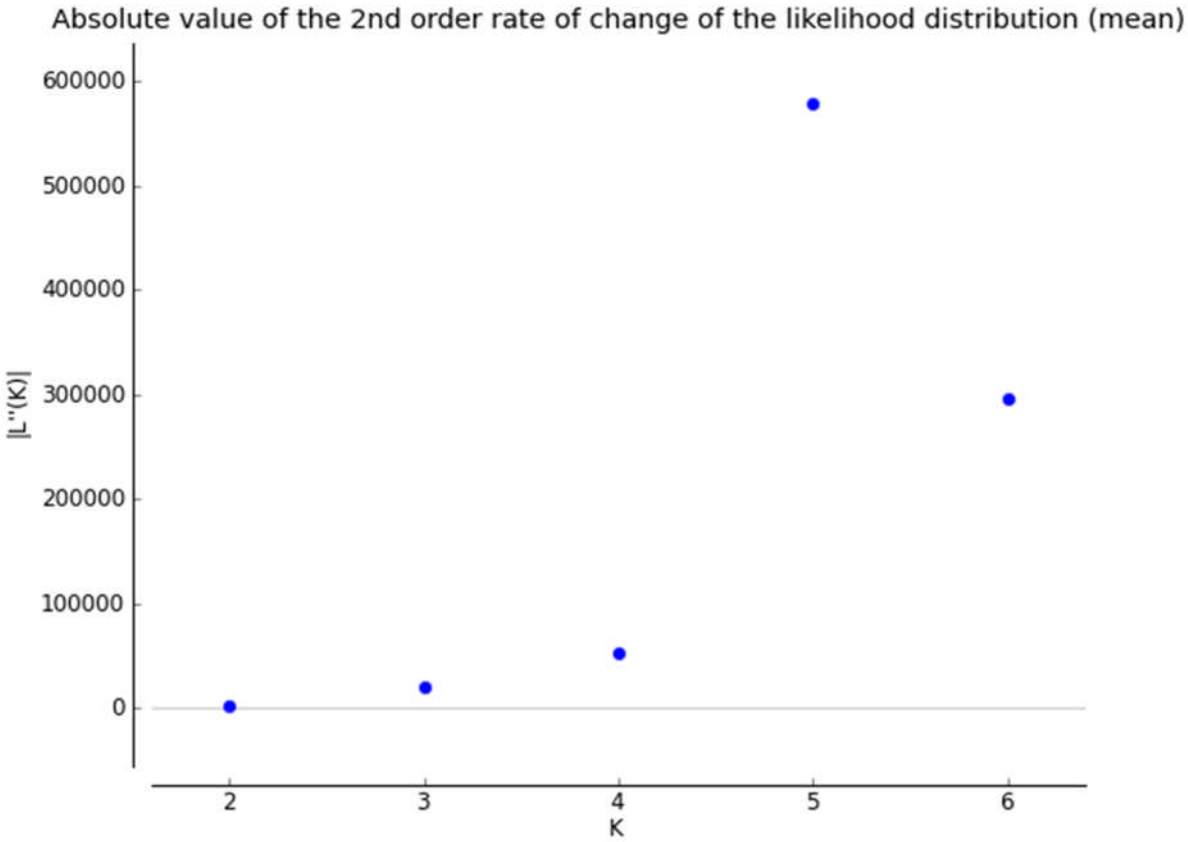
K = 7 Clumpp indfile K = 7 Clumpp popfile

Evanno method

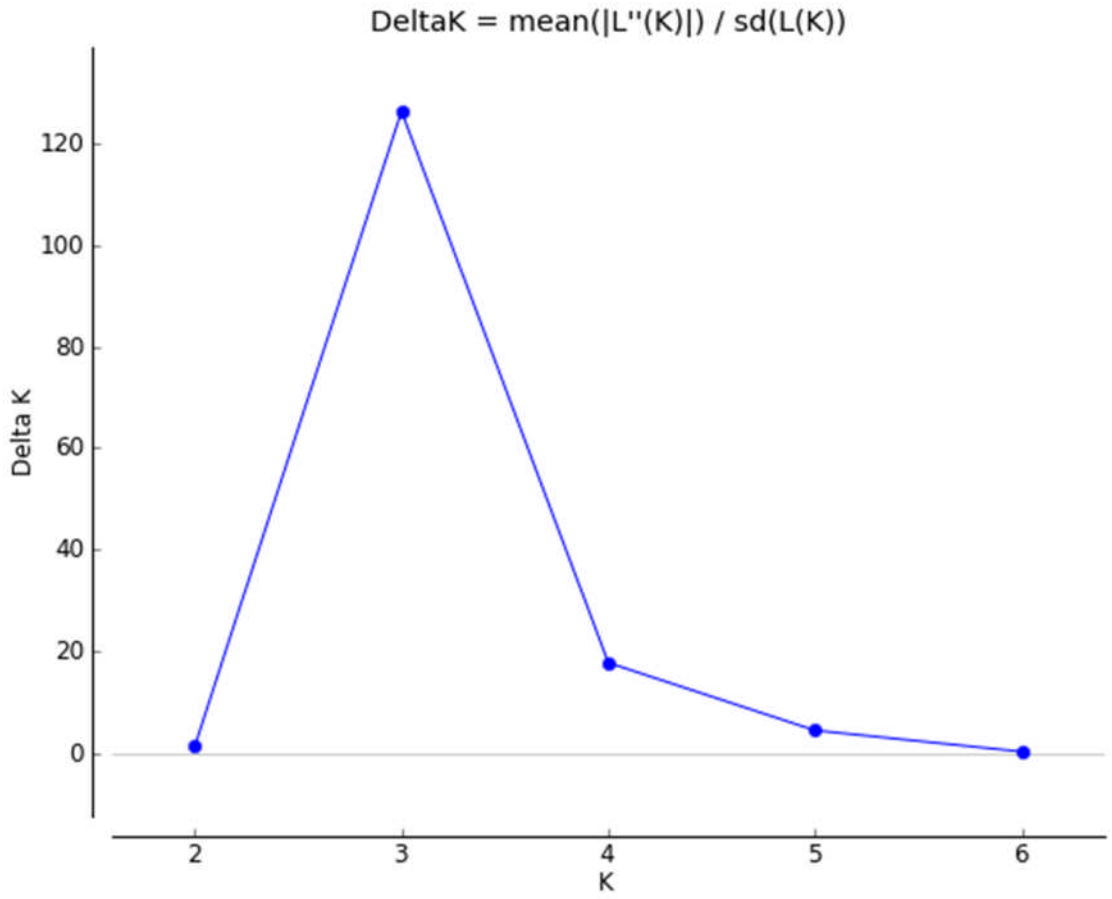
*Evanno et al., 2005. Molecular Ecology 14, 2611 - 2620. How are we calculating this? Look at the FAQ.



L'(K): pdf eps


Absolute value of the 2nd order rate of change of the likelihood distribution (mean)
600000
500000
400000
300000
200000
100000
0
|L''(K)|
2
3
4
5
6
K

|L''(K)|: pdf eps



Delta K: pdf eps

The Evanno table output is also available as a tab-delimited text file (for use with Excel) here.

K	Reps	Mean LnP(K)	Stdev LnP(K)	Ln'(K)	|Ln''(K)|	Delta K
1	5	-342199.840000	3.697026	—	—	—
2	5	-320813.040000	1946.774829	21386.800000	2527.900000	1.298507
3	5	-301954.140000	161.244265	18858.900000	20381.720000	126.402759
4	5	-303476.960000	3024.617094	-1522.820000	53528.680000	17.697672
5	5	-358528.460000	130276.638459	-55051.500000	578451.720000	4.440180
6	5	-992031.680000	1043974.080464	-633503.220000	295015.620000	0.282589
7	5	-1920550.520000	2135831.331001	-928518.840000	—	—

Raw STRUCTURE output

The raw STRUCTURE output is also available as a tab-delimited text file (for use with Excel) here.

File name	Run #	K	Est. Ln prob. of data	Mean value of Ln likelihood	Variance of Ln likelihood
standard2_run_5_f	5	1	-342199.0	-339295.8	5806.5
standard2_run_3_f	3	1	-342204.9	-339295.9	5818.0
standard2_run_1_f	1	1	-342196.5	-339295.6	5801.8
standard2_run_4_f	4	1	-342202.3	-339295.7	5813.2
standard2_run_2_f	2	1	-342196.5	-339295.9	5801.3
Standard2_run_9_f	9	2	-322866.8	-316932.2	11869.3
Standard2_run_10_f	10	2	-318965.2	-313090.4	11749.6
Standard2_run_7_f	7	2	-322834.9	-316932.4	11805.0
Standard2_run_8_f	8	2	-319007.1	-313091.2	11831.9
Standard2_run_6_f	6	2	-320391.2	-315619.5	9543.3
standard2_run_13_f	13	3	-301772.1	-294254.4	15035.5
standard2_run_14_f	14	3	-302017.6	-294260.5	15514.3
standard2_run_15_f	15	3	-301837.3	-294261.1	15152.3
standard2_run_11_f	11	3	-302184.7	-294245.1	15879.3
standard2_run_12_f	12	3	-301959.0	-294256.7	15404.4
standard2_run_20_f	20	4	-298069.1	-289675.2	16787.8
standard2_run_19_f	19	4	-304932.2	-296598.6	16667.3
standard2_run_16_f	16	4	-304673.4	-296603.7	16139.6
standard2_run_18_f	18	4	-304875.4	-296605.3	16540.2
standard2_run_17_f	17	4	-304834.7	-296614.1	16441.2
standard2_run_25_f	25	5	-300825.4	-292008.5	17633.7
standard2_run_24_f	24	5	-300797.8	-292017.7	17560.2
standard2_run_21_f	21	5	-591569.8	-292164.0	598811.6
standard2_run_22_f	22	5	-300589.4	-292015.5	17147.8
standard2_run_23_f	23	5	-298859.9	-289684.3	18351.2
standard2_run_28_f	28	6	-2736646.0	-290013.1	4893265.8
standard2_run_29_f	29	6	-404200.5	-288519.9	231361.3
standard2_run_27_f	27	6	-1197522.3	-289490.8	1816062.9
standard2_run_30_f	30	6	-301365.7	-289227.8	24275.9

standard2_run_26_f	26	6	-320423.9	-288821.6	63204.6
standard2_run_35_f	35	7	-5649444.9	-287407.6	10724074.8
standard2_run_34_f	34	7	-1093424.5	-286373.5	1614102.1
standard2_run_31_f	31	7	-295464.7	-282181.5	26566.4
standard2_run_32_f	32	7	-964052.0	-289976.6	1348150.8
standard2_run_33_f	33	7	-1600366.5	-286509.3	2627714.5

CITATION

Earl, Dent A. and vonHoldt, Bridgett M. (2012)
STRUCTURE HARVESTER: a website and program for visualizing
STRUCTURE output and implementing the Evanno method.
Conservation Genetics Resources vol. 4 (2) pp. 359-361 doi: 10.1007/s12686-011-9548-7

Core version: vA.2 July 2014
Plot version: vA.1 November 2012
Web version: v0.6.94 July 2014

CLUMPP v1.1.2 Scripts and Results

We used CLUMPP version 1.1.2 to determine the optimal alignment of clusters (Jakobsson & Rosenberg 2007). The most likely value of K is visualized below:

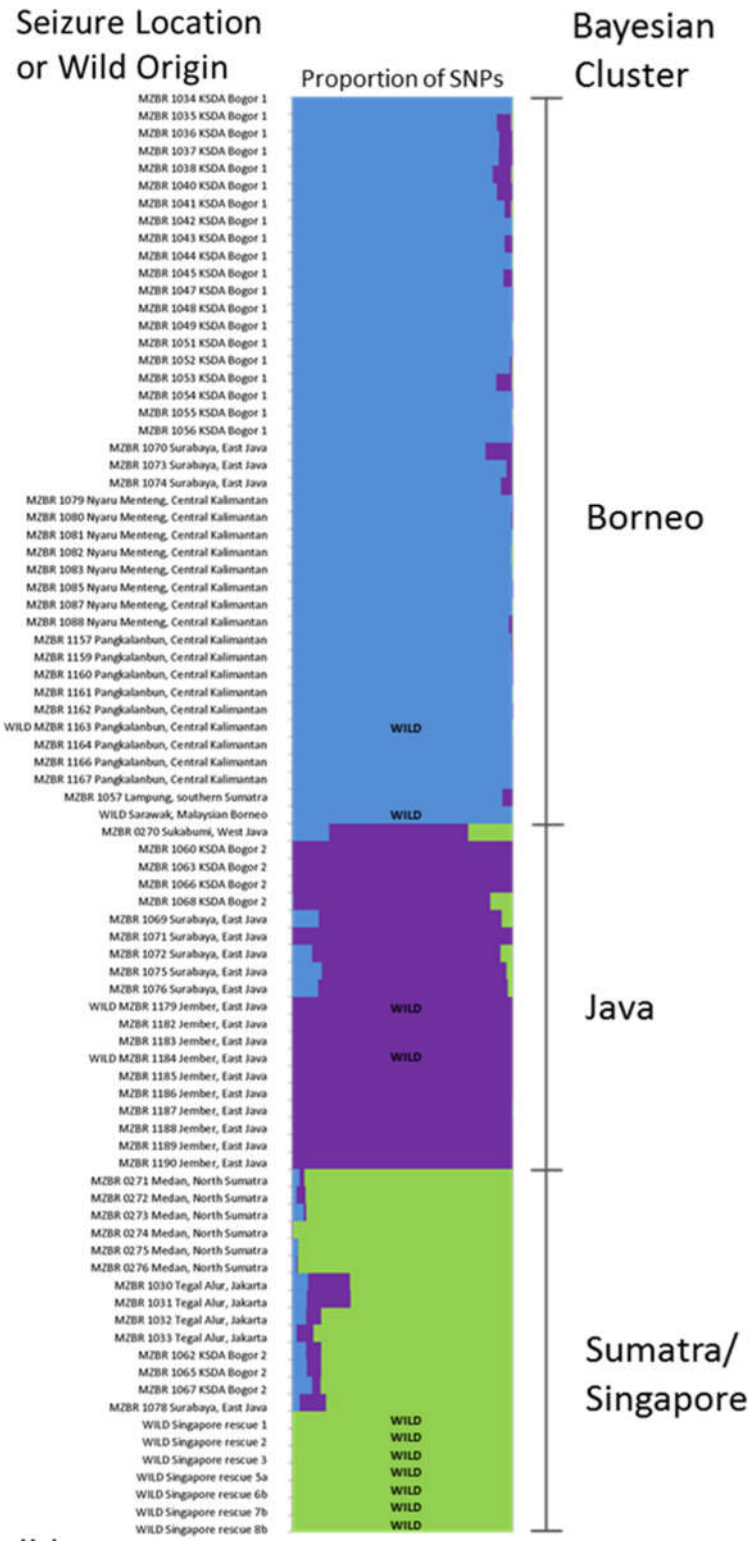


Figure 1. Bayesian cluster results of optimal K = 3 using 12,150 SNPs, indicating three distinct genetic clusters across 83 Sunda pangolins. Wild pangolins of known origin are labelled as WILD.

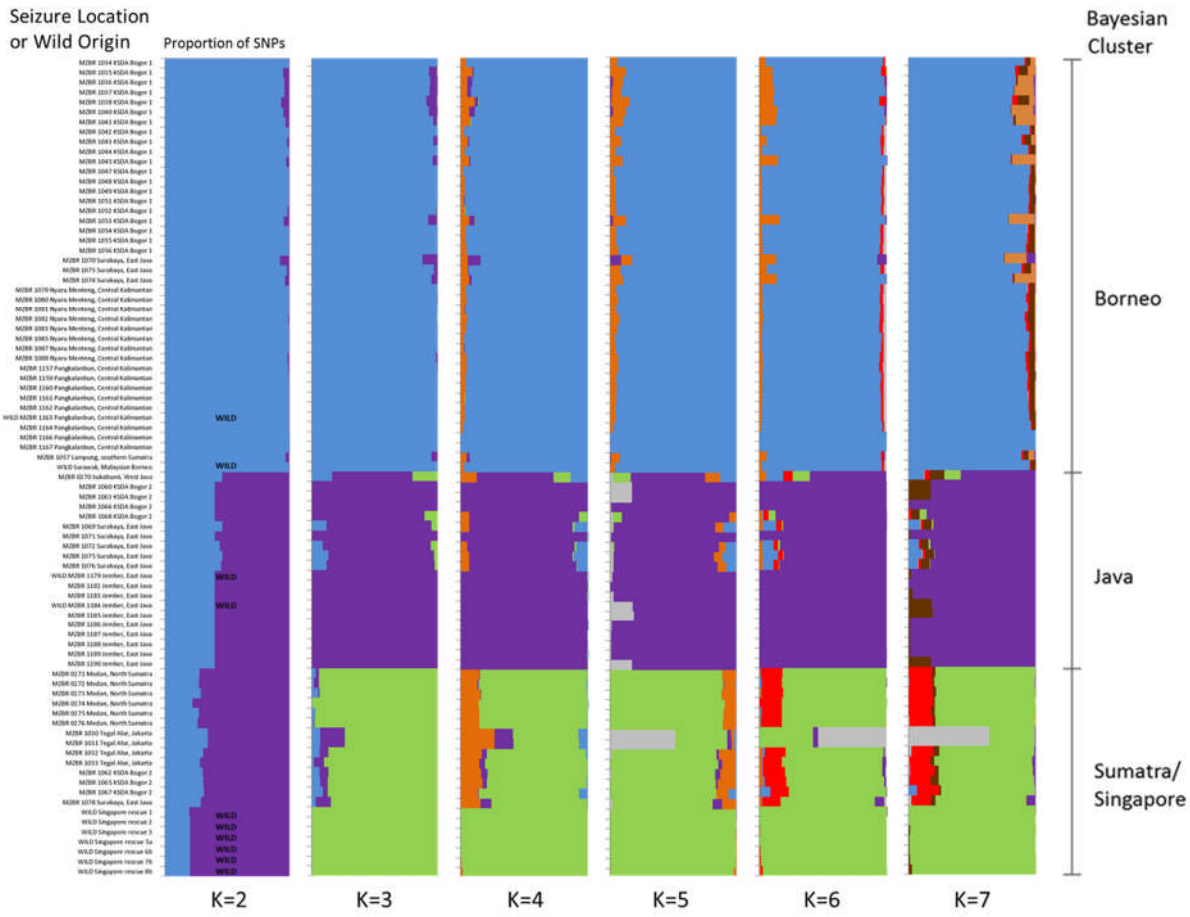


Figure 2. Bayesian cluster results of K = 2 to K = 7 using 12,150 SNPs across 83 Sunda pangolins. Wild samples of known origin are labelled as WILD.

<u>Below are the paramfile details</u>

This is the file that sets the parameters for the program CLUMPP

Everything after "#" will be ignored by the program. Parameters are:

K, C, R, M, W, GREEDY_OPTION, REPEATS, PERMUTATIONFILE, PRINT_PERMUTED_DATA,

PERMUTED_DATAFILE, PRINT_EVERY_PERM and EVERY_PERMFILE.

All parameter names shall be followed by at least one blank space and then

the parameter-value.

```
# ---------- Main parameters ------------------------------------------

DATATYPE 0                      # The type of data to be read in.
                                # 0 = individual data in the file
                                # specified by INDFILE, 1 = population
                                # data in the file specified by
                                # POPFILE.

INDFILE K3.indfile              # The name of the individual datafile.
                                # Required if DATATYPE = 0.

POPFILE K3.popfile              # The name of the population datafile.
                                # Required if DATATYPE = 1.

OUTFILE K3.outfile              # The average cluster membership
                                # coefficients across the permuted runs
                                # are printed here.

MISCFILE K3.miscfile            # The parameters used and a summary of
                                # the results are printed here.

K 3                             # Number of clusters.

C 83                            # Number of individuals or populations.

R 5                             # Number of runs.

M 1                             # Method to be used (1 = FullSearch,
                                # 2 = Greedy, 3 = LargeKGreedy).
```

```
W 1                                          # Weight by the number of individuals
                                             # in each population as specified in
                                             # the datafile (1 if yes, 0 if no).

S 2                                          # Pairwise matrix similarity statistic
                                             # to be used. 1 = G, 2 = G'.

# - Additional options for the Greedy and LargeKGreedy algorithm (M = 2 or 3) -

GREEDY_OPTION 2                              # 1 = All possible input orders,
                                             # 2 = random input orders,
                                             # 3 = pre-specified input orders.

REPEATS 1000                                 # If GREEDY_OPTION = 2, then REPEATS
                                             # determines the number of random input
                                             # orders to be tested. If GREEDY_OPTION
                                             # = 3, then REPEATS is the number of
                                             # input orders in PERMUTATIONFILE.

PERMUTATIONFILE arabid.permutationfile       # The permutations of the runs in
                                             # PERMUTATIONFILE will be used, if
                                             # GREEDY_OPTION = 3.

# ---------- Optional outputs ----------------------------------------

PRINT_PERMUTED_DATA 1                        # Print the permuted data (clusters) in
                                             # INDFILE or POPFILE to
                                             # PERMUTED_DATAFILE (0 = don't print,
                                             # 1 = print into one file, 2 = print
                                             # into separate files for each run).
```

```
PERMUTED_DATAFILE arabid.perm_datafile          # The permuted data (clusters) will be
                                                # printed to this file (if
                                                # PRINT_PERMUTED_DATA = 2, several
                                                # files with the extensions "_1" to
                                                # "_R" will be created).

PRINT_EVERY_PERM 0                              # Print every tested permutation of the
                                                # runs and the corresponding value of
                                                # SSC to a file specified by
                                                # EVERY_PERMFILE (0 = don't print,
                                                # 1 = print).
                                                # Note that printing may result in a
                                                # very large file.

EVERY_PERMFILE arabid.every_permfile# Every tested permutation of the runs
                                                # and the corresponding SSC will be
                                                # printed here.

PRINT_RANDOM_INPUTORDER 0                       # Print random input orders of runs to
                                                # RANDOM_INPUTORDER (0 = don't print,
                                                # 1 = print). This option is only
                                                # available if GREEDY_OPTION = 2.

RANDOM_INPUTORDERFILE arabid.random_inputorderfile # Every random input order
                                                # of the runs (generated by CLUMPP if
                                                # GREEDY_OPTION = 2) will be printed
                                                # here.

# ---------- Advanced options -----------------------------------------
```

```
OVERRIDE_WARNINGS 0                 # This option allows the user to
                                    # override non-crucial warnings from
                                    # the program (0 allow warnings, 1 do
                                    # not issue non-crucial warnings).

ORDER_BY_RUN 1                      # Permute the clusters of the output
                                    # files by the specified run. (0 to
                                    # not specify a run, 1 to R specifies
                                    # a run in the INDFILE or POPFILE).

# ---------- Additional comments ----------------------------------------

# The term ''permutation'' is used in two different contexts, permutations of
# membership coefficients, or clusters, and permutations of runs.

# For example, if the datafile has has data A B C D E (each letter indicates a
# column corresponding to a cluster), then permutation 3 2 5 1 4 of the
# clusters means C B E A D.

# Permutation 4 1 2 3 of runs 1-4 would mean start with run 4, then run 1, then
# run 2, and then run 3.

# ---------- Command line arguments -------------------------------------

# -i INDFILE
# -p POPFILE
# -o OUTFILE
# -j MISCFILE
# -k K
```

```
# -c C

# -r R

# -m M

# -w W

# -s S

# ------------------------------------------------------------------------------
```