Dog Population Splits and Mixtures from Genome-wide Allele Frequency Data

Shijia Bian
Department of Statistical Science
Duke University
Durham, NC 27705
shijia.bian@duke.edu

Abstract

In this paper, the clustering method is going to be applied to analyze the canine's population structure. Population structure is defined as the composition of the population. The structure of human population have been heavily studied in various fields. For example, the linguistics define the sub-population should belong to the same population based on the similarity in their languages. In genetics, the population structure is studied based on the similarity in the genome-wide information. The sub-population should be classified into one population if there is some levels of similarity in their genome-wide information. However, there are limited studies carried towards the other species. In this final project, the study is carried on the genome information from different dog breeds. Statistical method will be applied to analyze the population structure of these dogs based on their genome information.

1 Introduction

This project is motivated by Pickrell & Pritchard 2012 [1]. The goal of Pickrell & Pritchard 2012 is to use the genome-wide information to analyze the historical information of population by applying statistical inference. The analysis gives the splits and the mixture of the populations. The analysis also illustrates the migration among populations that is a novel part of the work and has been rarely accomplished in other research. *TreeMix* is the software published by this research group. *TreeMix* can give the visualization of the population splits, mixture and migration by taking the genome information as the inputs. My project will concentrate on the experiment given in the section of "Application to Dogs". The data used in this final group project is the same data set that is used in Pickrell & Pritchard 2012. The goal of my project is, first, carry on the preliminary covariance analysis given in this paper. The covariance analysis can give how each dog breed varies with each other. Second, the clustering methods will be applied to study the dog breed population structure, which dog breeds are more closed to each other. Third, compare the result with the results from Pickrell & Pritchard 2012.

2 Related Work

Pritchard & Stephens 2000 [2] conducts statistical inference on population structure by using the multilocus genotype data. The previous research focuses on the mixture analysis without admixture: assuming that each individual is independently originated from one population. This paper generalizes the framework to accommodate the admixture analysis on population structure: individual originates from multiple population. This new admixture model has a new defined parameter, Q, the proportion of the population that an individual is from. MCMC or EM algorithm can be

applied for this inference. Rosenberg & Pritchard (2002) has done a complete case study on the human population by using the method from this work [3]. However, the admixture has to specify the number of population, K. The process of the K selection has to be conducted by some ad hoc analysis. Although the method in this paper can give the individual's population components, it does not specify the historical information of the individual.

In Pickrell & Pritchard 2012, the author gives an analysis of analyzing the splitting history of the population structure. It proposes that some of the breeds should have a common ancestor. The population belonging to the same ancestor should have higher variance than the population does not belong to the common ancestor. The assumption of the model is that the breeds that do not have a common ancestor should have covariance be equal to 0 (Figure 4 A & B, X_1 and X_2 have the common ancestor). This work further analyzes that there is possible migration between breeds that do not have common ancestor. Thus some breeds might have non-zero covariance that should be assumed to be 0 in the previous case (Figure 4 C & D, X_2 is migrated to X_3).

3 Methods

3.1 Overview of the Data

The data table is from the Cornell mirror of the UCSC genome browser [4]. The set up of the browser is shown in Figure 2. The extracted data has 61,468 rows. Each row corresponds to the genome information at one specific chromosome. This table contains the allele information on the 39 plus X chromosomes of the 85 dog breeds. Each of the dog breed corresponds to one pair of allele. For each allele from the allele pair, the data has its count information. The partial screen shot of the table is listed below: the count of allele 1 at chromosome 1 ([3200955, 3200956]) for Jackal is 0, and the count of allele 2 at the same chromosome is 4. In other words, we have the same information as shown in Table 1 for each of the 85 dog or canine breeds.

Table 1: Dog Breed Allele Counts Data

chromosome	chromStart	chromEnd	Jackal Allele1	Jackal Allele2
chr1	3200955	3200956	0	4
chr1	3398479	3398480	0	4
chr1	3453894	3453895	0	16
...	...	...	...	...
chr38	26869797	26869798	0	18

3.2 Data Processing

The allele frequency of the allele at a locus can reflect the characteristics of a specific breed. Given the data above, we can derive the allele frequency at each diploid chromosome for every breed. The allele frequency at each chromosome is defined as the percentage of allele 1 count of the total sum of allele 1 and allele 2 counts. The derived information used for final analysis is shown in Table 2.

Table 2: Dog Breed Allele Counts Data

chromosome	chromStart	chromEnd	Jackal Allele Frequency
chr1	3200955	3200956	0%
chr1	3398479	3398480	0%
chr1	3453894	3453895	0%
...	...	...	.....
chr38	26869797	26869798	0%

In addition, all the missing data should be excluded, because it will bring large bias into the analysis [1]. Therefore, we need exclude all the rows that have the total counts of allele 1 and allele 2 be

equal to 0. Even though some of the chromosome might miss some counts, the total sum of the allele 1 count and allele 2 counts is smaller than some of the total counts for the same breeds. For example, there might be some missing counts for the first three chromosome in Table 1, because their total counts are 4, 4 and 16. These are less than 18 in the last total count at chromosome 38. Since we are interested in the proportion. We are assuming the missing count is also proportional to the non-missing count, we do not need exclude these cases. After excluding all the missing data, there are 38,312 remaining rows.

In the end, this project selects the allele information from 38 dog breeds. Each of the dog breed has 38,312 allele frequency for 38,312 corresponding chromosome. The 38 breeds are categorized into 6 populations: wild canines, ancient breeds, spitz, toy dog, herding and mastiff.

3.3 Covariance Analysis

As suggested in the paper, the data can be divided into k blocks, then a block design is applied to calculate the covariance between the six population. This can avoid the correlation of allele frequency among the nearby SNPs. Our data can be evenly divided into six blocks. As the covariance is calculated by using $\hat{W}_{ij} = \frac{\sum_{k=1}^{p} \hat{W}_{ijk}}{p}$. The distribution of $\hat{W}_{ij}$ is modeled as $\hat{W}_{ij} \sim N(G, \sigma_{ij}^2)$ and $\sigma_{ij}^2 = \frac{\sum_{k=1}^{p} (\hat{W}_{ijk} - \hat{W}_{ij})^2}{p(p-1)}$. The resulting covariance in our case is shown in table 3. It shows that the toy and spitz are the most correlated. Herding and wild canines are the list correlated. This illustrates that toy and spitz is the pair that has the most similarities. The population structure of the two dog breeds is the most related.

Table 3: Covariance of the Six Population

	Wild Canines	Ancient Breed	Spitz	Toy	Herding	Mastiff
Wild Canines	0.0890	0.0055	0.0138	0.0041	-0.0020	-0.0035
Ancient Breed	0.0055	0.0263	0.0132	0.0139	0.0105	0.0101
Spitz	0.0138	0.0132	0.0332	0.0159	0.0084	0.0077
Toy	0.0041	0.0139	0.0159	0.0639	0.0107	0.0103
Herding	-0.0019	0.0105	0.0084	0.0107	0.0204	0.0107
Mastiff	-0.0034	0.0101	0.0077	0.0102	0.0107	0.0218

G is a asylic graph that can imitate the structure shown in Figure 4 A. It is the expected covariance of the population. The calculation of G is computational expensive. This is not the focus in this final project.

3.4 Hierarchical Clustering Analysis

The Hierarchical Clustering Analysis (HCA) can be used to analyze the distance between different categories. By using this analysis with different types of linkage, ways of measuring distance, the cluster of the six population for the 36 dog breeds is plotted below. Figure 1 applies average linkage, and Figure 2 applies complete linkage.

In terms of the two plots, this bottom-up agglomerative clustering algorithm speaks very well to the true population of dog breeds. For example, Figure 1 first clusters Chihuahua and Kuvasz together. This implies that the two dog breeds are the most similar. Chihuahua and Kuvasz are both classified as ancient breeds. Boxer is the last merged breed and is classified of mastiff dog. In terms of the covariance matrix, mastiff generally has the least or relatively small covariance, implying least or small similarity, with other population compared to the other breeds. Figure 2 also behaves in the similar way. But Figure 2 is more balanced than Figure 1.

This HCA also speaks very well to the population structure plot in Figure 5 A. However, the two plots are carried by different approaches. The plots shown in the paper is accomplished through the maximum likelihood analysis for the covariance matrix: $L(\hat{W}|W) = \prod_{i=1}^{m} \prod_{j=1}^{m} N(\hat{W}_{ij}|G, \hat{\sigma}_{ij}^2)$. The breeds that have larger likelihood will be merged together. Out approach used the average distance as a criteria. The two breeds that have smaller average distance will be merge together.

Figure 1: Hierarchical Clustering Analysis for Six Population by Average Linkage

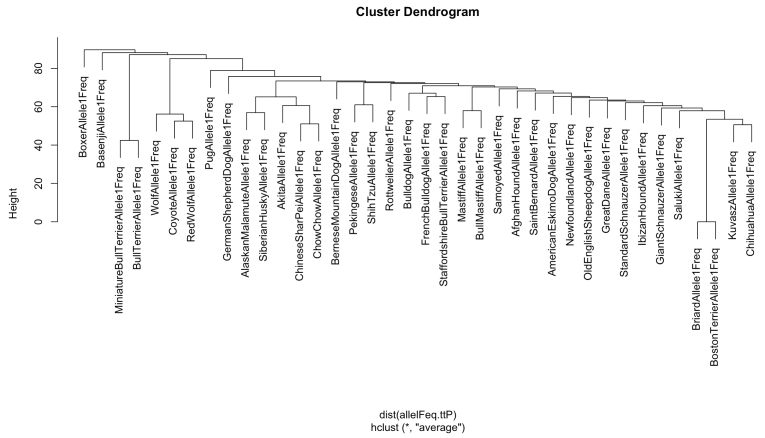
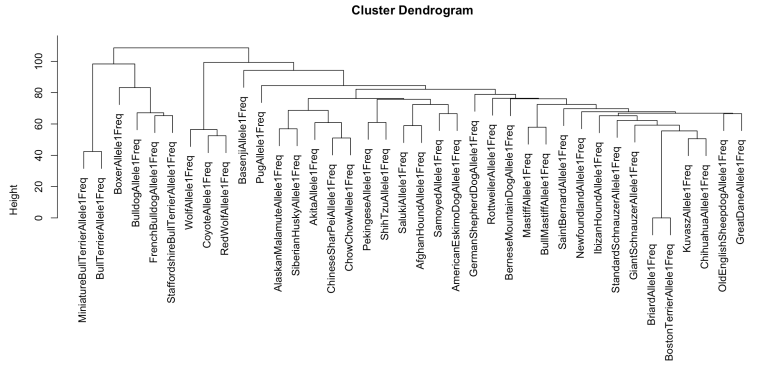


Figure 2: Hierarchical Clustering Analysis for Six Population by Complete Linkage



4 Results

The bottom-up hierarchical clustering analysis shows a similar result as illustrated by using the maximum likelihood analysis used in the paper, even though the approaches are different . In addition, the two plots also speak well to the covariance matrix. The breeds those are from different population might merge together in the early stage if the two population have high covariance in the covariance matrix, such as the toy and spitz in Figure 1 & 2.

5 Discussion

In terms of the two HCA, complete linkage can classify all the wild canines into the same clusters. It can also classify most breeds to the right clusters. From this perspective, the complete linkage is a better option than the average linkage, because it has better clustering power. However, the overfitting issue incurred by using the two linkage can be further deployed as well for the final decision.

In addition, the clustering analysis is lack of the ability to describe the migration. Therefore, the choice the the expected population covariance matrix G needs to be carefully chosen. Thus, a more sophisticated method for measuring the distance between each population can be constructed to define the true ancestor of the dog breeds. More important, the iteration of $f = 1 - \frac{\sum_{i=1}^{m}\sum_{j=1+1}^{m}(R_{ij}-\bar{R}_{ij})^2}{\sum_{i=1}^{m}\sum_{j=1+1}^{m}(\hat{W}_{ij}-\bar{\hat{W}}_{ij})^2}$ needs to be carried over for learning the migration across different

population. In terms of ”learning from the residual”, a gradient boosting method for studying this covariance might be applied further. It might be able to accomplish the migration part without using the method in the paper.

6 Conclusion

Overall, this final project replicate the dog population clustering analysis by using the genome-wide information data. The result can successfully support the population structure (covariance matrix among population) for the dog breeds shown in the paper.

Acknowledgments

This research was supported by Professor Sayan Mukherjee. Welcome any comments and critiques for this final project.

References

[1] Pickrell J.K. & Pritchard J.K. (2012) Inference of Population Splits and Mixtures from Genome-Wide Allele Frequency Data. *PLoS Genet*, 8(11): e1002967. doi:10.1371/journal.pgen.1002967.

[2] Falush D. & Stephens M. & Pritchard J.K. (2002) Inference of Population Structure Using Multilocus Genotype Data: Linked Loci and Correlated Allele Frequencies/ *GENETICS* August 1, 2003 vol. 164 no. 4 1567-1587

[3] Rosenberg N.A. & Pritchard J.K. & Weber J.L. & Cann H.M. & Kidd K.K. & Zhivotovsky L.A. & Feldman M.W. (2002). Genetic structure of human populations. *Science*, 298(5602), 2381-2385.

[4] Cornell Mirror of the UCSC Genome Browser

Appendix: Figure

Figure 3: Cornell Mirror of the UCSC Genome Browser

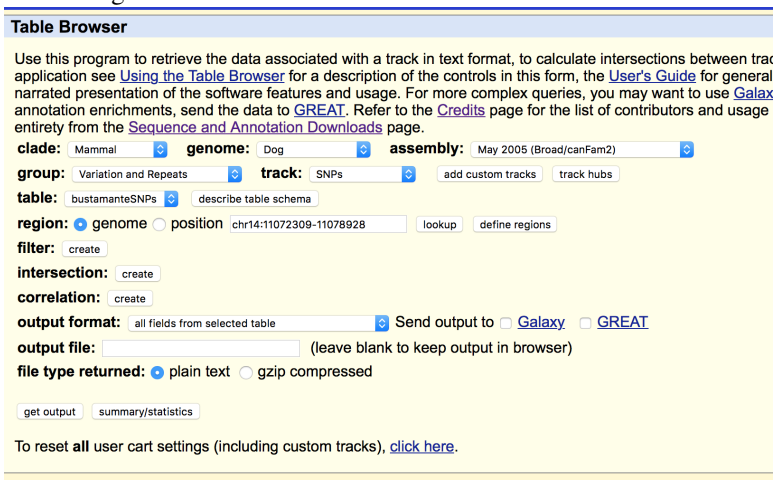
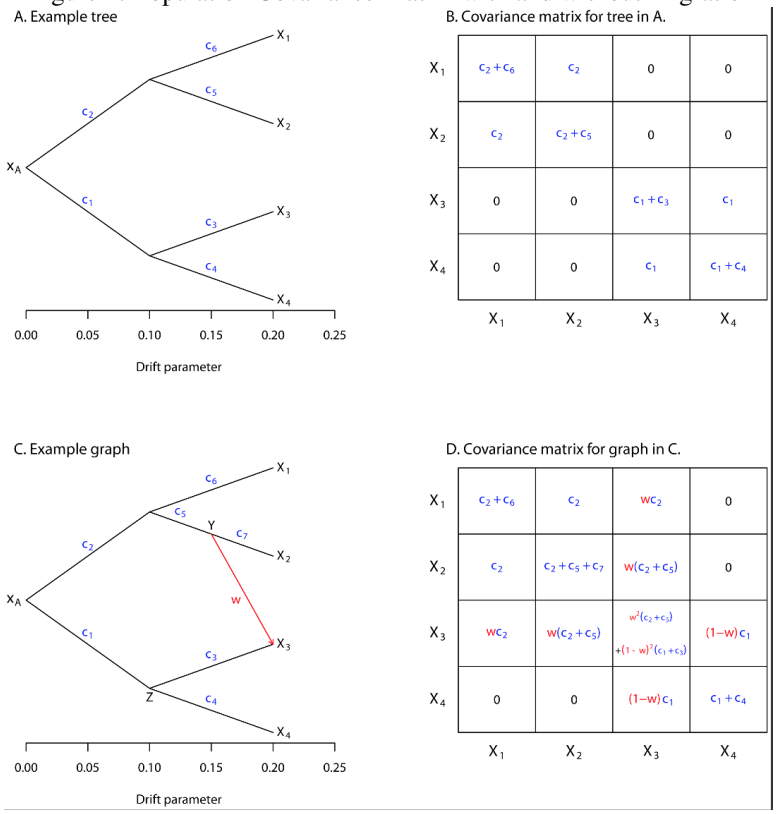


Figure 4: Population Covariance Matrix with and without Migration



Appendix: R Code

```
setwd("~/Dropbox/Duke Statistics Courses/Spring 2016 Duke Course/STA 613 COMSCI BIO/Final Proj

dat = read.table("dat.txt", header = TRUE)

names(dat)
colnames(dat)
dat.df = data.frame(dat)   # 61468   288

View(dat.df)
breed = c("wildCanids", "wildCanids",
          "wildCanids", "wildCanids",
          "wildCanids", "wildCanids",
          "acientBreeds", "acientBreeds",
          "acientBreeds", "acientBreeds",
          "acientBreeds", "acientBreeds",
          "acientBreeds", "acientBreeds",
          "acientBreeds", "acientBreeds",
          "acientBreeds", "acientBreeds",
          "spitz", "spitz",
          "spitz", "spitz",
          "spitz", "spitz",
          "spitz", "spitz",
          "spitz", "spitz",
          "spitz", "spitz",
          "spitz", "spitz",
          "toy", "toy",
          "toy", "toy",
          "herding", "herding",
          "herding", "herding",
          "herding", "herding",
          "herding", "herding",
          "herding", "herding",
          "mastiff", "mastiff",
          "mastiff", "mastiff",
          "mastiff", "mastiff",
          "mastiff", "mastiff",
          "mastiff", "mastiff",
          "mastiff", "mastiff",
          "mastiff", "mastiff",
          "mastiff", "mastiff",
          "mastiff", "mastiff",
          "mastiff", "mastiff",
          "mastiff", "mastiff",
          "mastiff", "mastiff",
          "mastiff", "mastiff",
          "mastiff", "mastiff",
          "mastiff", "mastiff")

# extract the breeds that will be used for the analysis
df.pop = cbind(dat.df[,1:6],
               dat.df$CoyoteAllele1, dat.df$CoyoteAllele2,
               dat.df$RedWolfAllele1, dat.df$RedWolfAllele2,
               dat.df$WolfAllele1, dat.df$WolfAllele2,
               dat.df$BasenjiAllele1, dat.df$BasenjiAllele2,
               dat.df$SalukiAllele1, dat.df$SalukiAllele2,
               dat.df$AfghanHoundAllele1, dat.df$AfghanHoundAllele2,
               dat.df$KuvaszAllele1, dat.df$KuvaszAllele2,
               dat.df$ChihuahuaAllele1, dat.df$ChihuahuaAllele2,
               dat.df$IbizanHoundAllele1, dat.df$IbizanHoundAllele2,
               ## spitz
               dat.df$ChineseSharPeiAllele1, dat.df$ChineseSharPeiAllele2,
               dat.df$ChowChowAllele1, dat.df$ChowChowAllele2,
               dat.df$AkitaAllele1, dat.df$AkitaAllele2,
```

```
                dat.df$AlaskanMalamuteAllele1, dat.df$AlaskanMalamuteAllele2,
                dat.df$SiberianHuskyAllele1, dat.df$SiberianHuskyAllele2,
                dat.df$SamoyedAllele1, dat.df$SamoyedAllele2,
                dat.df$AmericanEskimoDogAllele1, dat.df$AmericanEskimoDogAllele2,
                ## toy
                dat.df$PekingeseAllele1, dat.df$PekingeseAllele2,
                dat.df$ShihTzuAllele1, dat.df$ShihTzuAllele2,
                ## herding
                dat.df$GermanShepherdDogAllele1, dat.df$GermanShepherdDogAllele2,
                dat.df$StandardSchnauzerAllele1, dat.df$StandardSchnauzerAllele2,
                dat.df$BriardAllele1, dat.df$BriardAllele1,
                dat.df$GiantSchnauzerAllele1, dat.df$GiantSchnauzerAllele2,
                dat.df$OldEnglishSheepdogAllele1,dat.df$OldEnglishSheepdogAllele2,
                ## mastiff
                dat.df$PugAllele1, dat.df$PugAllele2,
                dat.df$BostonTerrierAllele1, dat.df$BostonTerrierAllele1,
                dat.df$FrenchBulldogAllele1, dat.df$FrenchBulldogAllele2,
                dat.df$StaffordshireBullTerrierAllele1, dat.df$StaffordshireBullTerrierAllele2,
                dat.df$MiniatureBullTerrierAllele1, dat.df$MiniatureBullTerrierAllele2,
                dat.df$BullTerrierAllele1, dat.df$BullTerrierAllele2,
                dat.df$BulldogAllele1, dat.df$BulldogAllele2,
                dat.df$BoxerAllele1, dat.df$BoxerAllele2,
                dat.df$MastiffAllele1, dat.df$MastiffAllele2,
                dat.df$BullMastiffAllele1, dat.df$BullMastiffAllele2,
                dat.df$GreatDaneAllele1, dat.df$GreatDaneAllele2,
                dat.df$RottweilerAllele1, dat.df$RottweilerAllele2,
                dat.df$SaintBernardAllele1, dat.df$SaintBernardAllele2,
                dat.df$BerneseMountainDogAllele1, dat.df$BerneseMountainDogAllele2,
                dat.df$NewfoundlandAllele1, dat.df$NewfoundlandAllele2)

# dim(df.pop) = 61468    82

# rename the dataframe
names(df.pop) = c(names(dat.df)[1:6],substring(names(df.pop)[7:82], 8))


# build the allele frequency table
freq.Table = matrix(0, nrow = 61468, ncol = 38)
for (i in 1:38) {
  #i =38
  freq.Table[,i] = df.pop[, 5 + 2*i] /(df.pop[, 5 + 2*i] + df.pop[, 6 + 2*i])
}

# save the freq table as a data frame
freq.Table= as.data.frame(freq.Table)
# 61468    38

freq.TableName = c()
for (i in 1:38) {
  dummyName = paste(names(df.pop)[5 + 2*i], "Freq", sep = '')
  freq.TableName = c(freq.TableName, dummyName)
}

# rename the frequency data
names(freq.Table) = freq.TableName
dim(freq.Table) # 61468    38
# bind the two table
df.pop.freq = cbind(df.pop, freq.Table)

# if value is missing
# build the allele frequency table
missing.Table = matrix(0, nrow = 61468, ncol = 38)
for (i in 1:38) {
  #i =38
  missing.Table[,i] = (df.pop[, 5 + 2*i] + df.pop[, 6 + 2*i])
```

```
}

missing.Table = as.data.frame(missing.Table)

# delete the rows that have 0
dummy.freq.Table = freq.Table

for (i in 1:38) {
  control = missing.Table[,i] != 0
  dummy.freq.Table = dummy.freq.Table[control, ]
  missing.Table = missing.Table[control, ]
}

#> dim(dummy.freq.Table)
#[1] 38312    38
#> dim(freq.Table)
#[1] 61468    38

# dim(missing.Table) 38312    38
# dim(dummy.freq.Table) 38312    38
################
# dummy.freq.Table is the table for final analyzing
save(file = "dummy.freq.Table.Rdata", dummy.freq.Table)

---
title: "STA 613 Report Markdown Draft"
author: "Shijia Bian"
date: "May 3, 2016"
output: pdf_document
---

```{r setup, include=FALSE}
knitr::opts_chunk$set(echo = TRUE)
```

Load the cleaned up date set `dummy.freq.Table.Rdata`, and write down the marks for the breed.

dummy.freq.Table.Rdata has the column be the allele frequency for each of the species, the bre

```{r}
load("~/Dropbox/Duke Statistics Courses/Spring 2016 Duke Course/STA 613 COMSCI BIO/Final Proje

breed = c("wildCanids", "wildCanids",
          "wildCanids", "wildCanids",
          "wildCanids", "wildCanids",
          "acientBreeds", "acientBreeds",
          "acientBreeds", "acientBreeds",
          "acientBreeds", "acientBreeds",
          "acientBreeds", "acientBreeds",
          "acientBreeds", "acientBreeds",
          "acientBreeds", "acientBreeds",
          "spitz", "spitz",
          "spitz", "spitz",
          "spitz", "spitz",
          "spitz", "spitz",
          "spitz", "spitz",
          "spitz", "spitz",
          "spitz", "spitz",
          "toy", "toy",
          "toy", "toy",
          "herding", "herding",
          "herding", "herding",
          "herding", "herding",
          "herding", "herding",
          "herding", "herding",
```
```

```
          "mastiff", "mastiff",
          "mastiff", "mastiff",
          "mastiff", "mastiff",
          "mastiff", "mastiff",
          "mastiff", "mastiff",
          "mastiff", "mastiff",
          "mastiff", "mastiff",
          "mastiff", "mastiff",
          "mastiff", "mastiff",
          "mastiff", "mastiff",
          "mastiff", "mastiff",
          "mastiff", "mastiff",
          "mastiff", "mastiff",
          "mastiff", "mastiff")


breadMark = c()
for (i in 1:38) {
  breadMark.du = breed[2*i]
  breadMark = c(breadMark, breadMark.du)
}
```


calculate the population mean for each of the population, and the overall mean
```{r}
wildCanidsMean = apply(dummy.freq.Table[,breadMark=="wildCanids"], 1, mean)
acientBreedsMean = apply(dummy.freq.Table[,breadMark=="acientBreeds"], 1, mean)
spitzMean = apply(dummy.freq.Table[,breadMark=="spitz"], 1, mean)
toyMean = apply(dummy.freq.Table[,breadMark=="toy"], 1, mean)
herdingMean = apply(dummy.freq.Table[,breadMark=="herding"], 1, mean)
mastiffMean = apply(dummy.freq.Table[,breadMark=="mastiff"], 1, mean)
overallMean = apply(dummy.freq.Table, 1, mean)
```

Calculate the covariance matrix
```{r}
popMean = cbind(wildCanidsMean, acientBreedsMean, spitzMean, toyMean, herdingMean, mastiffMean
dim(popMean) # 38312     6
popMean.matrix = as.matrix(popMean)
cov(popMean.matrix)
```

```{r}
library("mmod")
data("nancycats")
nancycats

#install.packages("poppr")
library("poppr")

#install.packages("ape")
library("ape") # To visualize the tree using the "nj" function

#install.packages("magrittr")
library("magrittr")
data(microbov)
set.seed(10)
ten_samples <- sample(nInd(microbov), 10)
mic10       <- microbov[ten_samples]
(micdist    <- provesti.dist(mic10))
```

```{r}


```

```
allelFeq = as.matrix(dummy.freq.Table)
allelFeq.t = t(allelFeq)
allelFeq.tt=na.omit(allelFeq.t)

allelFeq.ttP = allelFeq.tt
clusters <- hclust(dist(allelFeq.ttP), method = 'complete')
plot(clusters)
```

```
composite liklihood
```{r}
# https://cran.r-project.org/web/packages/CompRandFld/CompRandFld.pdf
# page 11
#install.packages("CompRandFld")
library(CompRandFld)

#install.packages("RandomFields")
library(RandomFields)

# install.packages("scatterplot3d")
library(scatterplot3d)
set.seed(31231)
# Set the coordinates of the points:
x <- runif(100, 0, 10)
y <- runif(100, 0, 10)
coords<-cbind(popMean[,1],popMean[,2])[1:1000,]
#################################################################
###
### Example 1. Plot of covariance and variogram functions
### estimated from a Gaussian random field with exponent
### correlation. One spatial replication is simulated.
###
###
#################################################################
# Set the model's parameters:
corrmodel <- "exponential"
mean <- 0
sill <- 1
nugget <- 0
scale <- 2
# Simulation of the Gaussian random field:
data <- RFsim(coordx=coords, corrmodel=corrmodel, param=list(mean=mean,
sill=sill, nugget=nugget, scale=scale))$data
# Maximum composite-likelihood fitting of the Gaussian random field:

start<-list(scale=scale,sill=sill,mean=mean(data))
fixed<-list(nugget=nugget)
# Maximum composite-likelihood fitting of the random field:
fit <- FitComposite(data, coordx=coords, corrmodel=corrmodel,likelihood="Marginal",
type="Pairwise",start=start,fixed=fixed,maxdist=6)
# Results:
print(fit)
# Empirical estimation of the variogram:
vario <- EVariogram(data, popMean[,1][1:1000], popMean[1:1000])
# Plot of covariance and variogram functions:
par(mfrow=c(1,2))
Covariogram(fit, show.cov=TRUE, show.range=TRUE,
show.vario=TRUE, vario=vario,pch=20)


#################################################################
##
### Example 2. Plot of covariance and extremal coefficient
### functions estimated from a max-stable random field with
```
```

```
### exponential correlation. n idd spatial replications are
### simulated.
###
############################################################
set.seed(1156)
# Simulation of the max-stable random field:
data <- RFsim(coordx=coords, corrmodel=corrmodel, model="ExtGauss", replicates=20,
param=list(mean=mean,sill=sill,nugget=nugget,scale=scale))$data
start=list(sill=sill,scale=scale)
# Maximum composite-likelihood fitting of the max-stable random field:
fit <- FitComposite(data, coordx=coords, corrmodel=corrmodel, model='ExtGauss',
replicates=20, varest=TRUE, vartype='Sampling',
margins="Frechet",start=start)
```

The Block Design

Equation 20, estimate G

```{r}
dim(dummy.freq.Table)
#### divide the table into 8 blocks
popMean = cbind(wildCanidsMean, acientBreedsMean, spitzMean, toyMean, herdingMean, mastiffMean
dim(popMean) # 38312     6
popMean.matrix = as.matrix(popMean)
cov(popMean.matrix)
```

Equation 24, estimate W hat:

```{r}
w_ij = cov(popMean.matrix)
col_mean = apply(popMean.matrix, 1, mean)
G_ij = t(popMean.matrix - col_mean)  %*% (popMean.matrix - col_mean)
```

Equation 27, estimate sigma ij hat:

Divide the popMean.matrix into 4 blocks
```{r}
dim(popMean.matrix)
seq(1, 38312, length.out = 5)

popMean.matrix1 = popMean.matrix[1:9578, ]
popMean.matrix2 = popMean.matrix[9579:19156, ]
popMean.matrix3 = popMean.matrix[19157:28734, ]
popMean.matrix4 = popMean.matrix[28735:38312, ]

block1 = cov(popMean.matrix1)
block2 = cov(popMean.matrix2)
block3 = cov(popMean.matrix3)
block4 = cov(popMean.matrix4)

w_hat_ij = (block1+block2+block3+block4)/4

sigma_ij_square = ((block1-w_hat_ij)^2+(block2-w_hat_ij)^2+(block3-w_hat_ij)^2+(block4-w_hat_i
sigma_ij_square
```

Eq 28
```{r}
overall_Exp = mean(w_hat_ij)
w_ij = matrix(overall_Exp, 6, 6)
temp = 1
```

```
for (i in 1:dim(w_ij)[1]) {
  for (j in 1:dim(w_ij)[2]) {
    stor = pnorm(w_hat_ij[i,j], w_ij[i,j], sqrt(sigma_ij_square[i,j]))
    if (stor != 0) {
      temp = temp * stor
      print(temp)
    }
  }
}
temp
```