# Sample Answer Lab01: Evaluation of Centroids and Directional Data

**Handed out:** Wednesday, January 22, 2020

**Return date:** Wednesday, February 05, 2020, at the beginning of the class.

### Task 1: Cluster identification [2 points]

Open the file **ClusteredPoints.dbf** to identify bivariate normal distributed clusters is this dataset with the function **Mclust( )**.

* 1. Estimate the optimal number of clusters in this dataset [0.5 points]

library(foreign)

library(mclust)

ClusterP <- read.dbf('./ClusteredPoints.dbf')

p.Mclust <- Mclust(ClusterP, G=c(3,4,5), modelNames="VVV")

summary(p.Mclust)

----------------------------------------------------

Gaussian finite mixture model fitted by EM algorithm

----------------------------------------------------

Mclust VVV (ellipsoidal, varying volume, shape, and

orientation) model with 4 components:

log.likelihood n df BIC ICL

-734.7653 200 23 -1591.392 -1608.453

Clustering table:

1 2 3 4

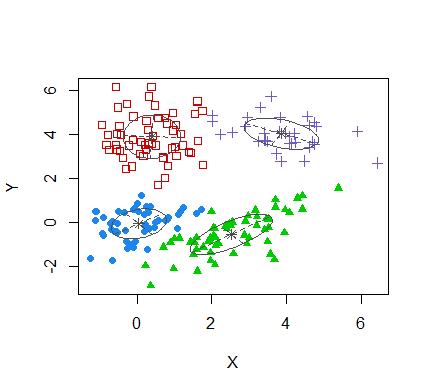
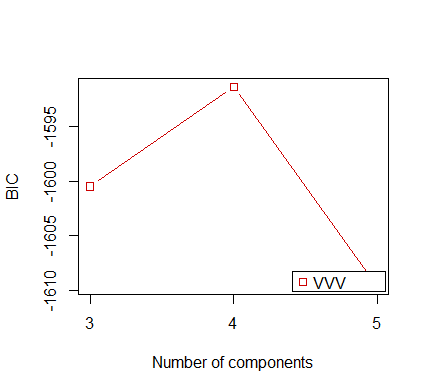
50 58 59 33

The estimated optimal number of clusters is 4.

* 1. Generate a plot of the estimated classification [0.5 points]

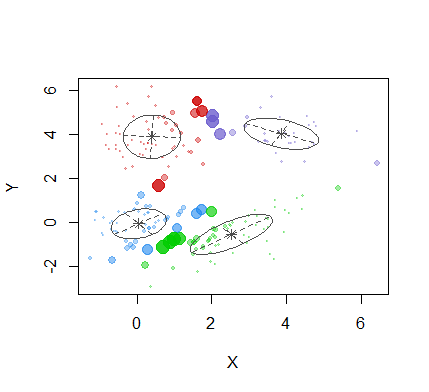
plot(p.Mclust, what = "classification")

plot(p.Mclust,what = "BIC")



* 1. Generate a plot of the estimated uncertainty of assigning a point to a particular cluster and interpret the plot. [0. 5 points]

plot(p.Mclust, what = "uncertainty")



* 1. Report in a table the estimate number of points in a cluster and the cluster centroids. [0.5 points]

df1 <- data.frame(p.Mclust$parameters$mean)

colnames(df1) <- table(p.Mclust$classification)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Cluster No. | 1 | 2 | 3 | 4 |
| No. of points | 50p | 58 | 59 | 33 |
| Cluster centroids | X 0.049  Y -0.046 | X 0.399  Y 3.913 | X 2.529  Y -0.528 | X 3.868  Y 4.047 |

### Task 2: Calculation of City Centroids/Medians [2 points]

Open the map in the file **MyStudyArea.zip**.

Task 2.1: Calculate the simple city centroids for each region. [0.25 points]

Shape <- rgdal::readOGR(dsn="./MyStudyArea/MyFourRegions.shp",integer64="warn.loss")

City <- rgdal::readOGR(dsn="./MyStudyArea/MyPopPlaces.shp",integer64="warn.loss")

CityDf <- as.data.frame(City)

MeanX <- tapply(CityDf[,"coords.x1"],CityDf[,"REGIONID"],mean,simplify=F)

MeanY <- tapply(CityDf[,"coords.x2"],CityDf[,"REGIONID"],mean,simplify=F)

(cbind(MeanX,MeanY))

MeanX MeanY

1 -0.001546545 0.001479636

2 -0.001352091 0.001024545

3 -0.0009622 0.0004012

4 -0.0005575714 0.001822714

Task 2.2: Calculate the population weighted city centroids for each region. [0.25 points]

regionSet <- unique(CityDf[,"REGIONID"])

xy\_ma <- matrix(NA,length(regionSet),2)

for(i in 1:length(regionSet)){

idxReg <- regionSet[i]

City.id <- which(CityDf[,"REGIONID"] == idxReg)

x.w.mean <- weighted.mean(CityDf[City.id,4],CityDf[City.id,"POPULATION"],na.rm=TRUE)

y.w.mean <- weighted.mean(CityDf[City.id,5],CityDf[City.id,"POPULATION"],na.rm=TRUE)

xy\_ma[i,] <-c(x.w.mean,y.w.mean)

}

xy\_ma

[x] [y]

[1,] -0.0014238401 0.0014111868

[2,] -0.0008487914 0.0015543882

[3,] -0.0014386750 0.0011918997

[4,] -0.0011261227 0.0004243523

Task 2.3: Calculate the population weighted Euclidian city medians for each region. [0.25 points]

tol <- 0.0001

maxIter <- 80

nofPoints <- 10

xy\_eu <- matrix(NA,length(regionSet),2)

for(i in 1:length(regionSet)){

iIter <- 0

City.id <- which(CityDf[,"REGIONID"] == regionSet[i])

w <- CityDf[City.id,"POPULATION"]

x <- CityDf[City.id,4]

y <- CityDf[City.id,5]

xOld <- xy\_ma[i,1]

yOld <- xy\_ma[i,2]

repeat{

d <- sqrt((x-xOld)^2+(y-yOld)^2) # distance equation in step 1

xNew <- sum(w\*x/d)/sum(w/d) # update equation in step 2

yNew <- sum(w\*y/d)/sum(w/d)

xyDiff <- sqrt((xNew-xOld)^2+(yNew-yOld)^2) # calculate tolerance score

xOld <- xNew

yOld <- yNew

iIter <- iIter + 1

print(xyDiff)

if (xyDiff < tol | iIter > maxIter) break

Sys.sleep(0.5)

} #end::repeat

xy\_eu[i,] <- c(xNew,yNew)

}

xy\_eu

[x] [y]

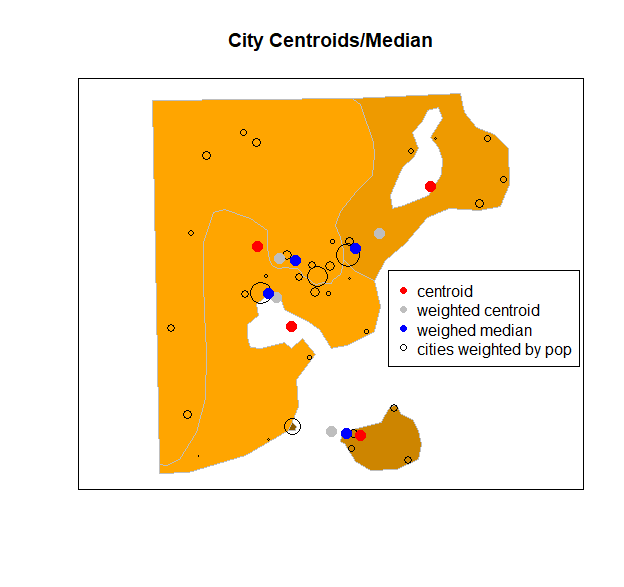
[1,] -0.001328753 0.0013968477

[2,] -0.001002505 0.0014595419

[3,] -0.001486036 0.0012146427

[4,] -0.001151801 0.0004233249

Task 2.4: Map all three calculate centroids/medians jointly into a map with a proper map legend. Discuss which centroids or the median is most appropriate to show the center of the population distribution in a region. [0.5 points]



Weighted median is the most appropriate to show the center of population distribution within a region. The left two regions have concave shapes, the centroids are outside of the region, the region on the right has a hole inside, the centroid is also outside of the area. Since population distributions within a region are highly positively skewed, the weighted centroids are greatly impacted by the skewness, for example, the weighted centroid is outside of the bottom region. The island region has a small bridge head on the mainland. Therefore, the weighted centroid is pulled outside the study area.

The weighted median points are all within their region, and they best summarized the centers of population distributions within each region.

Task 2.5: The -script **EuclidianMedian.r** (see Lecture01) can be substantially optimized. [0.75 points]

Use the user-defined function **euclidMedian( )** in the script in combination with the  optimization function **nlm( )** to find the population weighted Euclidian Median. Show that section of your script which uses this optimization approach.

euclidMedian <- function(xy){

## Function that evaluates the cost at any point XY given

## the global vector argments: x, y, and w

cost <- sum(w \* sqrt( (x-xy[1])^2 + (y-xy[2])^2 ))

return(cost)

} #end::euclidMedian

**Med.nlm<- nlm(euclidMedian, c=(5,5))**

cat("x-Median = ",round(nlmOpti$estimate[1],3) ,"\n",

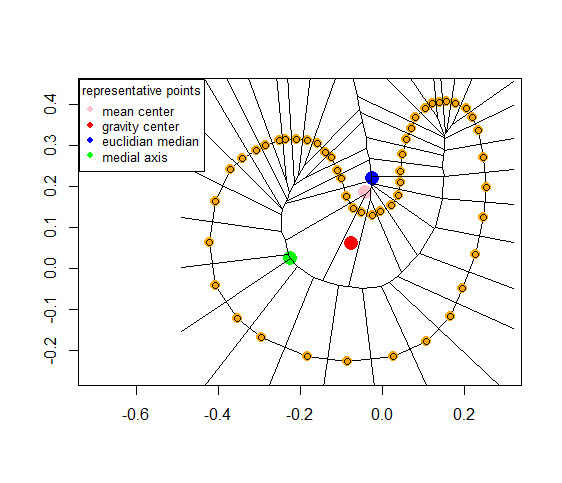
"y-Median = ",round(nlmOpti$estimate[2],3) ,"\n",

"Min. Cost= ",nlmOpti$minimum,"\n")

### Task 3: Polygon Centers [2 points]

Open the shape file **ShapeKindey.shp** which outlines the boundary of a polygon. You will identify several more or less representative central points for this polygon. In tasks 3.1 to 3.4 plot these points into the polygon and make also sure to highlight the boundary shape points. All distinct points should be in one plot with a legend. No need to your code.

Task 3.1: Add the ***arithmetic mean center*** to the map of your polygon. Does this point reasonably well represent the center of the polygon? [0.5 points]



Note: the orange points depict the shape points of the polygon. These are used to evaluate the different polygon centers. Each shape point has an equal weight.

Task 3.2: Add the ***center of gravity*** to the map of the polygon. Does this point reasonably well represent the center of the polygon? [0.5 points]

Outside of the polygon.

Task 3.3: Add the ***Euclidian median*** to the map of the polygon. Does this point reasonably well represent the center of the polygon? [0.5 points]

Inside the polygon but pulled toward the region of high density shape points. This point is calculated by default for shape-files. It can be extracted from a polygon with command **coordinates( )** .

Task 3.4: Add the ***medial axis*** point to the map of the polygon. Does this point reasonably well represent the center of the polygon? [0.5 points]

Inside the polygon and furthest away from the boundary edge.

### Task 4: Directional Data Analysis [3 points]

The file **CarCrashDirection.dbf** holds information about the time and direction a car was travelling, when an accident occurred. An insurance analyst wants to know if, dependent on the time of day, accidents happen at predominate travelling directions.

The analyst breaks the time of the crash down into a factor with four categories (a) morning rush hour, (b) day time, (c) evening rush hour, and (d) night time.

It is known that irrespectively of the time of day an equal number of cars is travelling in any direction, that is, the overall travel direction of all trips is uniformly distributed at any time.

Task 4.1: Why is it important for the analysis of the accident directions that the overall travel direction of all trips at any time is uniformly distributed? [1 point]

If the general traffic at any time would have a directional bias and assuming that the traffic accidents are proportional to the overall traffic, then we would expect to see coincidentally more accidents where more traffic takes place. On the other hand, if the general traffic follows a uniform directional pattern than any excessive direction of accidents points at a directional accident bias. Accident directions are a subset of the overall travel directions.

Task 4.2: Generate 4 rose diagrams for accident directions at each time of day and calculate the average direction and mean resultant length . Interpret the plots. [1 point]

|  |  |
| --- | --- |
|  |  |
| N:259, average: 95.27  Rho: 0.360, *p*-val: 0 | N:155, average: 18.90  Rho: 0.018, p-val: 0.949 |
|  |  |
| N:341, average: -87.59  Rho: 0.324, p-val: 0 | N:99, average: -61.86  Rho: 0.101, p-val: 0.367 |

The four plots indicate that the car accidents in the day and night follow an uniform distribution, however during the morning rush hours, we can observe more car accidents in the eastern direction and during the evening rush hours, more accidents can be observed in the western direction.

The -values for the day and night plots are close to 0, which also suggest an approximately uniform distribution. For the two plots on the right side, their -values are 0.360 and 0.324 respectively, which means their distributions are less likely to be uniform. Moreover, their direction means suggest that morning rush hours have more accidents on the eastern direction and evening rush hours have more accidents on the western direction. This may be related to the blinding effects of the sun being low on the horizon.

Task 4.3: Perform Rayleigh tests for each time period to test whether the accidents at a given time period occur uniformly distributed or at a predominate direction. Draw your conclusions with regards to your visual interpretation in task 4.2. [1 point]

See 𝑝-values in above plot. The morning and evening accident distributions have a significant directional bias, excessive crashes in the morning for vehicles driving into the rising easterly sun and in the evening for vehicles driving into the westerly setting sun.

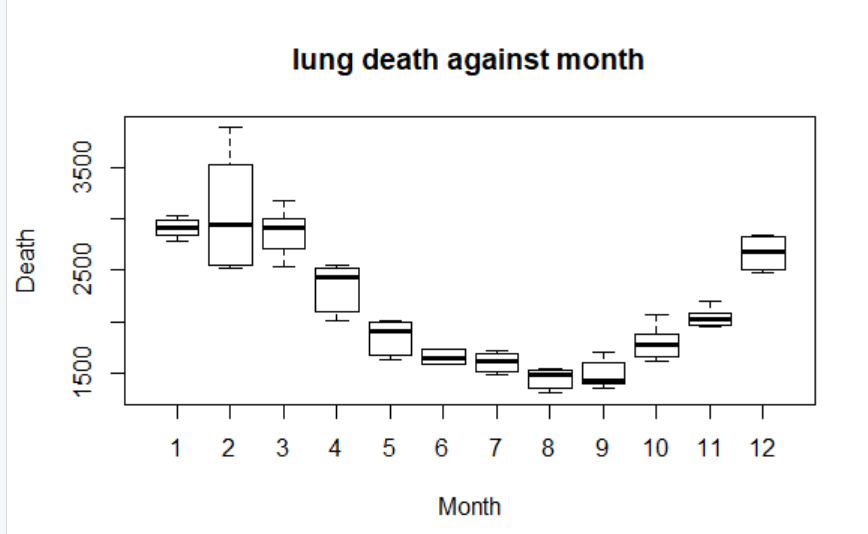
### Task 5: Regression with a Circular Response Variables [1 point]

The workspace **LungDf.RData** contains data of lung disease related death counts by month over seven years.

Task 5.1: Discuss the seasonal pattern of the lung disease related death counts using a boxplot stratified by the months. [0.25 points]

boxplot(lungDeath~monthNum,data=lungDf, main="lung death against month",

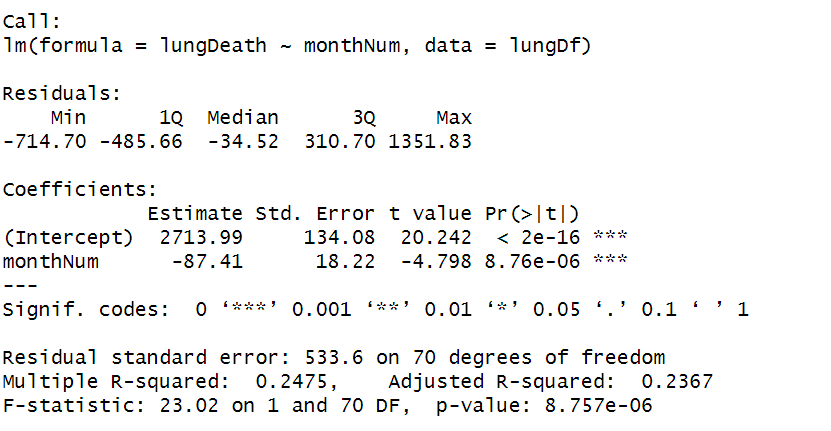
xlab="Month", ylab="Death")



Lung death has a higher frequency in both spring and winter, has a lower frequency in summer and autumn.

Task 5.2: Test with linear regression whether there is a *trend* in the lung disease related death counts over time [0.25 points]

summary(lm(lungDeath~monthNum,data=lungDf))



The number of lung disease has a significant temporal pattern (correlated with time trend).

Task 5.3: Estimate with linear regression the *amplitude* and of the seasonal behavior of the lung disease related death counts. [0.5 points]

yrs <- 6

t <- seq(0, (yrs\*12)-1, by=1)

lambda1 <- 1/12

## Generate time-dependent cos and sin proxy variable

yCos1 <- cos(t\*lambda1\*2\*pi)

ySin1 <- sin(t\*lambda1\*2\*pi)

## Estimate model

yrs.lm <- lm(lungDf$lungDeath~lungDf$monthNum + yCos1 + ySin1)

summary(yrs.lm)

Coefficients:

Estimate Std. Error t value Pr(>|t|)

(Intercept) 2275.59 87.79 25.920 < 2e-16 \*\*\*

lungDf$monthNum -19.96 12.85 -1.554 0.124809

yCos1 710.21 40.43 17.566 < 2e-16 \*\*\*

ySin1 240.42 61.39 3.917 0.000211 \*\*\*

---

Signif. codes: 0 ‘\*\*\*’ 0.001 ‘\*\*’ 0.01 ‘\*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 230 on 68 degrees of freedom

Multiple R-squared: 0.8642, Adjusted R-squared: 0.8582

F-statistic: 144.2 on 3 and 68 DF, p-value: < 2.2e-16

best.lm <- lm(lungDf$lungDeath~ yCos1 + ySin1 )

summary(best.lm)

Call:

lm(formula = lungDf$lungDeath ~ yCos1 + ySin1)

Residuals:

Min 1Q Median 3Q Max

-449.75 -117.12 -22.94 123.32 955.37

Coefficients:

Estimate Std. Error t value Pr(>|t|)

(Intercept) 2145.82 27.38 78.361 < 2e-16 \*\*\*

yCos1 730.17 38.73 18.855 < 2e-16 \*\*\*

ySin1 314.93 38.73 8.132 1.15e-11 \*\*\*

---

Signif. codes: 0 ‘\*\*\*’ 0.001 ‘\*\*’ 0.01 ‘\*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 232.4 on 69 degrees of freedom

Multiple R-squared: 0.8594, Adjusted R-squared: 0.8553

F-statistic: 210.8 on 2 and 69 DF, p-value: < 2.2e-16

plot(x = t,y = lungDf$lungDeath,lwd = 2 ,type = "l", col = 'blue')

lines(x = t,y = predict(best.lm),lwd = 2,col = 'red')

