####################################################################################################

# 階層型クラスター分析(ward法)アルゴリズムを自作実装する

#

# Ward(1963)に則った正統派アルゴリズムにする。計算過程の内部状態を全て保存し、後で参照可能とする。

# 非類似度SDpqはWard(1963)に従って平方ユークリッド距離を使う

# SDpq = {(Np x Nq)/(Np + Nq)} X ||Cp - Cq||^2

# Lance-Williamsの更新式もSDpqに関する式とする

# Murtagh and Legendre(2014)で記載されている Ward1 相当のアルゴリズム。

#

# 扱う多変量データを N行 × M列 とする。SSDSE-2020Cなら(N = 47, M = 213)。

#

# 本プログラムでは変数名に一部大文字を使用している。エストレーラ5月号本文の記載に合わせたためである。

# （R言語のコーディングスタイルガイドでは、変数及び関数名は小文字でつけるべきであるとされているが。）

#

# 「非類似度行列キューブ」 DisS[i,j,k] は3次元キューブ(N × N × N-1)とする。

# エストレーラ本文ではDisS[k,i,j]の順番で添え字を記載したが、csvに保存した際の認視性が

# 高いことからプログラム上ではDisS[i,j,k]の順に変更している。ご注意いただきたい。

# k(1:初期状態からN-1)を全て残す。非類似度行列のサイズは工程が進むにつれて1つ小さくなる。

#

# 「クラスターベクトル行列」 CV[k,i]は2次元行列とする。

#

# 「残存クラスター識別番号ベクトル行列」 CNV[k,i]も2次元行列とする。

# クラスター識別番号は、スタート時点では1からnであるが、凝集が進むにつれて1増やす。

# クラスターが1つになった時点で、2N-1になる。

#

# 「残存クラスター要素数ベクトル行列」 CNNV[k,n]も2次元行列とする。

#

# 凝集時の最小非類似度ベクトルminDS[k]は要素数n-1のベクトルである

#

# SSDSE-2020c をCSVファイルでダウンロードしたものから、自作ward法でクラスタベクトル行列を作成する

#

# hclust$order の作成は、CV[k,i]だけから作成できる。

# hclust$merge の作成は、クラスター凝集工程処理の中で並行して作成する。

#

# エストレーラ５月号の本文と読み比べやすいように、適宜コメントで対応付けしてある。

####################################################################################################

library(dplyr)

######################

# SSDSE-2020C.csv をtidydataに変換する前処理 と csvファイルへの保存

######################

# 統計センターのホームページからダウンロードした SSDSE-2020C.csv を読み込む

ssdse\_2020c <- read.csv("ssdse-2020C.csv")

# SSDSE-2020C.csvの不要行と不要列を削除する（結果をssdse\_2020c\_edited.csvに保存し、変数ssdseに再度読み込む）

ssdse\_2020c\_small <- select(ssdse\_2020c, -X2017\_2019, -City, -LA03, -LB01, -LB02, -LB03, -LB04, -LB05, -LB06, -LB07, -LB08, -LB09, -LB10, -LB11, -LB12)

ssdse\_2020c\_small\_pref <- filter(ssdse\_2020c\_small, Prefecture != "全国")

colnames(ssdse\_2020c\_small\_pref) <- ssdse\_2020c\_small\_pref[1,]

ssdse\_2020c\_edited\_pref <- filter(ssdse\_2020c\_small\_pref, 都道府県!="都道府県")

rownames(ssdse\_2020c\_edited\_pref) <- c(ssdse\_2020c\_edited\_pref$都道府県)

ssdse <- select(ssdse\_2020c\_edited\_pref, -都道府県)

# 全て文字型となってしまっているので、いったんCSVファイルに""抜きで書き出して改めて読み込む

write.csv(ssdse, file="ssdse\_2020c\_edited.csv", quote = FALSE)

ssdse <- read.csv("ssdse\_2020c\_edited.csv", row.name = 1)

######################

# SSDSE-2020C.csv をtidydataに変換したものを標準化する前処理 と csvファイルへの保存

######################

# ssdseを観測データ間の絶対値のばらつきをそろえるために、行で比率化する

ssdse.p <- (ssdse %>% select(names(ssdse)[2]:names(ssdse)[214]) %>% sweep(.,1,c(as.numeric(ssdse[,1])),"/")\*100)

# 変量間の大小のばらつきをそろえるために列で標準化（平均=0、標準偏差=1）し、csvファイルへ保存する

ssdse.std <- scale(ssdse.p)

write.csv(ssdse.std, file="ssdse\_std.csv", quote = FALSE)

######################

# 初回のward非類似度行列の計算 と csvファイルへの保存

######################

ssdse.dist <- dist(ssdse.std)

ssdse.halfdistD2 <- (1/2)\*{ssdse.dist^2} # 平方ユークリッド距離 × 係数項(Np\*Nq)/(Np + Nq): 初回は1/2

ssdse.halfdist.mat <- as.matrix(ssdse.halfdistD2)

write.csv(ssdse.halfdist.mat, file="ssdse\_halfdistD2.csv", quote = FALSE)

#対角要素以外の非類似度が>0であることのチェック

Tempssdse.halfdist.mat <- ssdse.halfdist.mat

for(i in 1:nrow(ssdse.halfdist.mat)){

Tempssdse.halfdist.mat[i,i] <- 1 # 非類似度が0である対角要素に1を代入する。

}

if(min(Tempssdse.halfdist.mat) == 0){

stop("観測データに完全一致データが複数存在します。") # 対角要素以外に0値が存在する場合は処理修了

}

####################################################################################################

####################################################################################################

# ここからward法の独自実装（非類似度は平方距離で正しく処理する。Ward(1963)に則った正統な処理とする）

#######################

# 初回の非類似度行列からデータサンプル数Nを取得する

N <- nrow(ssdse.halfdist.mat)

#「非類似度行列キューブ」(DisS)のサイズを定義 DisS[i,j,k] 凝集工程kの非類似度行列

# csvに書き込んだ時に見やすくなる様に、エストレーラ本文とは添え字の並び方を変えてあるので注意

DisS <- array(1, dim = c(N,N,N-1))

#「クラスターベクトル行列」(CV[k,i])のサイズを定義

CV <- array(1, dim = c(N,N))

CV[1,] <- 1:N

#「残存クラスター識別番号ベクトル行列」(CNV[k,i])のサイズを定義

CNV <- array(1, dim = c(N,N))

CNV[1,] <- 1:N

#「残存クラスター要素数ベクトル行列」(CNNV[k,i])のサイズを定義

CNNV <- array(1, dim = c(N,N))

CNNV[1,] <- rep(x = 1, times=N)

#「凝集最小非類似度ベクトル」(minDS[k])のサイズを定義(minDS[1] = 初回の最小非類似度)

minDS <- numeric(N-1)

#hclust$merge相当の行列(CVmerge[k,1:2])のサイズを定義する

CVmerge <- array(1, dim = c(N-1, 2) )

#次工程の非類似度行列作成のためのゼロ行(サイズ長N+1)とゼロ列（サイズ長N+2）を用意する

ZeroColNplus1 <- matrix(0,nrow=N+1,ncol=1)

ZeroRowNplus2 <- matrix(0,nrow=1,ncol=N+2)

#初期状態の非類似度行列を加工用にコピーする

TempDisS <- ssdse.halfdist.mat

###################################################################################

#凝集工程を進めながらクラスターが一つになるまで以下の処理を繰り返す（初期状態をk=1とする）

##################

for(k in 1:(N-1)){

##################################################

#前の工程で作成した凝集工程kの非類似度行列を「非類似度行列キューブ」に保存する

DisS[,,k] <- TempDisS

################################################

#ここから非類似度行列の最小値と行と列を抽出する

#まずはじめに、要素0となっている項（対角要素や非類似度行列が縮小する際に充填した0行0列など）を最大値＋１に置き換える。

#これは、min()関数を使って最小の非類似度を抽出できるようにするため。

#前提として、観測データに多変量値完全一致のものが無い限り非類似度の最小値は>0であることを利用している。

MaxSVal <- max(TempDisS)

TempDisS[TempDisS == 0] <- MaxSVal+1

#非類似度行列の最小値(MinSVal)とそのrow,colを抽出 #######最小非類似度ペアが複数存在した場合の対応も考慮

MinSVal <- min(TempDisS)

MinSRowCol <- which(TempDisS == MinSVal, arr.ind=TRUE)

p <- min(MinSRowCol[1,]) ################## 最小非類似度ペアのうちの一つ目のペアを選択している

q <- max(MinSRowCol[1,]) ################## 最小非類似度ペアのうちの一つ目のペアを選択している

####################################

#凝集時非類似度ベクトル(minDS)の更新

minDS[k] <- MinSVal

#############################

#「クラスターベクトル行列」(CV)の更新

TempCV <- CV[k,] #いったんコピー

#クラスター識別番号が CNV[k,p] と CNV[k,q] のものを N+k に書き換える

TempCV[TempCV == CNV[k,p]] <- N+k

TempCV[TempCV == CNV[k,q]] <- N+k

CV[k+1,] <- TempCV

#####################

# CVmerge[k,i]の更新（この変数は、凝集の履歴を保存するものである。S３オブジェクトの”hclust”クラスの編集処理で必要となる）

#

# merge 対象であるクラスター識別番号CNV[k,p]とCNV[k,q]の大小によって

# CVmerge[k,1]とCVmerge[k,2]に保存するものを決定して保存する。 hclust$merge の仕様では、

# 要素番号＝クラスター識別番号(CNV[k,p] < N+1)の時には CNV[k,p]\*(-1)

# 要素番号≠クラスター識別番号(CNV[k,p] > N )の時には CNV[k,p]-N

# を保存する。絶対値の小さいほうがCVmerge[k,1]、大きいほうがCVmerge[k,2]に入る。

# まず、CVmerge[k,]に保存する数字にCNV[k,p]とCNV[k,q]を変換する。

if(CNV[k,p] < N+1 ){

MergeCluster1 <- CNV[k,p]\*(-1)

}else{

MergeCluster1 <- CNV[k,p]-N

}

if(CNV[k,q] < N+1 ){

MergeCluster2 <- CNV[k,q]\*(-1)

}else{

MergeCluster2 <- CNV[k,q]-N

}

# 次に、絶対値でSmallCluster と LargeCluster を判定する。

if(abs(MergeCluster1) < abs(MergeCluster2)){

SmallCluster <- MergeCluster1

LargeCluster <- MergeCluster2

}else{

SmallCluster <- MergeCluster2

LargeCluster <- MergeCluster1

}

# SmallCluster > 0 かつ LargrCluster < 0 の場合は、入れ替える。クラスター識別番号が[k,2]に入るように。

if(SmallCluster > 0 && LargeCluster < 0){

TempCluster <- SmallCluster

SmallCluster <- LargeCluster

LargeCluster <- TempCluster

}

# hclust$merge 相当の行列 CVmerge[k,1:2] を作成する。

CVmerge[k,1] <- SmallCluster

CVmerge[k,2] <- LargeCluster

#######################################

#「残存クラスター識別番号ベクトル行列」(CNV)の更新 「ESTRELA」2022年5月号（その２）の記事の図7の処理

TempCNV <- CNV[k,] # いったんコピー

#残存クラスター識別番号ベクトルの削除カラムを詰めて、その右端に N+k を差し込む

#まずは、TempCNVの要素値(N+k-1)の直後にc(N+K,0)をアペンドする

TempCNVA <- append(TempCNV, c(N+k,0), after=N-k+1)

#次に、TempCNVのMinSRowMin番目とMinSRowMaxを削除して詰める

TempCNVDel <- TempCNVA[-c(p,q)]

CNV[k+1,] <- TempCNVDel

##############################################

#「残存クラスター要素数ベクトル行列」(CNNV)の更新 「ESTRELA」2022年5月号（その２）の記事の図8の処理

TempCNNV <- CNNV[k,] #いったんコピー

#クラスター要素数更新

TempCNNVA <- append(TempCNNV, c(CNNV[k,p]+CNNV[k,q],0), after=N-k+1)

TempCNNVDel <- TempCNNVA[-c(p,q)]

CNNV[k+1,] <- TempCNNVDel

######################################

#次工程(k+1)の非類似度行列(DisS[,,k+1])をTempDisSとして編集する 「ESTRELA」2022年5月号（その２）の記事の図9の処理

#再び前工程(k)の非類似度行列を取り出す（対角要素などを0に戻すため）

TempDisS <- DisS[,,k]

#非類似度行列に追加する列の作成

Skplus1 <- matrix(nrow=N,ncol=1)

for(i in 1:N ){

nrplusni <- CNNV[k,p]+CNNV[k,q]+CNNV[k,i]

Skplus1[i] <- {(CNNV[k,p]+CNNV[k,i])/nrplusni}\*DisS[p,i,k] + {(CNNV[k,q]+CNNV[k,i])/nrplusni}\*DisS[q,i,k] - {CNNV[k,i]/nrplusni}\*DisS[p,q,k]

}

#非類似度行列に追加する列の結合

if(k == 1){

TempDisSA <- cbind(TempDisS[,1:N], Skplus1)

}else if(k == 2){

TempDisSA <- cbind(TempDisS[,1:N-1], Skplus1,TempDisS[,N])

}else{

TempDisSA <- cbind(TempDisS[,1:(N-k+1)], Skplus1,TempDisS[,(N-k+2):N])

}

#非類似度行列の下に追加する行の作成

tSkplus1 <- t(Skplus1)

ZeroM <- matrix(0,nrow=1,ncol=1)

tSkplus1A <- cbind(tSkplus1, ZeroM)

#非類似度行列に追加する行の結合

if(k == 1){

TempDisSAA <- rbind(TempDisSA[1:N,], tSkplus1A)

}else if(k == 2){

TempDisSAA <- rbind(TempDisSA[1:N-1,], tSkplus1A,TempDisSA[N,])

}else{

TempDisSAA <- rbind(TempDisSA[1:(N-k+1),], tSkplus1A,TempDisSA[(N-k+2):N,])

}

#非類似度行列の右端に追加するゼロ列の結合

TempDisSAAA <- cbind(TempDisSAA, ZeroColNplus1)

#非類似度行列の下端に追加するゼロ行の結合

TempDisSAAAA <- rbind(TempDisSAAA, ZeroRowNplus2)

#凝集済み行と列の削除

TempDisSAAAADelRowCol <- TempDisSAAAA[ -c(p,q), -c(p,q) ]

#次工程の非類似度行列の完成

TempDisS <- TempDisSAAAADelRowCol

}

##################

#凝集工程繰り返し処理の最終行

###################################################################################

#######################################

# 計算が全て完了した変数をcsvファイルに保存

#

write.csv(CV, file="ssdse\_halfdistD2\_CV.csv", quote = FALSE)

write.csv(CNV, file="ssdse\_halfdistD2\_CNV.csv", quote = FALSE)

write.csv(CNNV, file="ssdse\_halfdistD2\_CNNV.csv", quote = FALSE)

write.csv(minDS, file="ssdse\_halfdistD2\_minDS.csv", quote = FALSE)

write.csv(DisS, file="ssdse\_halfdistD2\_DisS.csv", quote = FALSE)

write.csv(CVmerge, file="ssdse\_halfdistD2\_merge.csv", quote = FALSE)

#######################

#

# ここまでがward法の独自実装

#

####################################################################################################

####################################################################################################

####################################################################################################

####################################################################################################

#

# ここから以降は、比較検証実験のための処理プログラム

#

########################################################################################

################################################################################

#### 検証実験1用 同じ非類似行列を入力として hclust("ward.D")で分析する

methname <- "ward.D" # method を指定

ssdse.hclust <- hclust(ssdse.halfdistD2, method=methname)

plot(ssdse.hclust)

write.csv(ssdse.hclust$height, file="ssdse\_halfdistD2\_hclust\_wardD\_minDS.csv", quote = FALSE)

# pngファイルに描き出し 「ESTRELA」2022年4月号（その１）の記事の図3

png("ssdse.hclust.ward.D.dendrogram.png",width = 1000, height = 800)

plot(ssdse.hclust)

dev.off()

####

#######################################################################################################

#### 検証実験2用 「初回ward非類似度行列」の平方根を入力として hclust("ward.D2")で分析し結果を二乗する

methname <- "ward.D2" # method を指定

ssdse.SRhalfdistD2 <- sqrt(ssdse.halfdistD2) # ward(1963)の非類似度の平方根をとって"ward.D2"に渡す。

ssdse.hclustD2 <- hclust(ssdse.SRhalfdistD2, method=methname)

plot(ssdse.hclustD2)

write.csv(ssdse.hclustD2$height, file="ssdse\_SRhalfdistD2\_hclust\_wardD2\_minDS.csv", quote = FALSE)

ssdse.SQ\_SRhalfdistD2\_height <- (ssdse.hclustD2$height)^2 #heightを2乗して平方距離の非類似度に戻す

write.csv(ssdse.SQ\_SRhalfdistD2\_height, file="ssdse\_SQ\_SRhalfdistD2\_hclust\_wardD2\_minDS.csv", quote = FALSE)

ssdse.SQ\_SRhalfdistD2 <- ssdse.hclustD2 # ”ward.D2” で作成されたhclustオブジェクトをコピー

ssdse.SQ\_SRhalfdistD2$height <- ssdse.SQ\_SRhalfdistD2\_height # heightを二乗したものを作成

# pngファイルに描き出し 「ESTRELA」2022年4月号（その１）の記事の図4

png("ssdse.hclust.ward.D2.dendrogram.png",width = 1000, height = 800)

plot(ssdse.hclustD2)

dev.off()

####

# heightだけを二乗したものでpngファイルに描き出し 「ESTRELA」2022年4月号（その１）の記事の図5

png("ssdse.SQ\_SRhalfdistD2.ward.D2.dendrogram.png",width = 1000, height = 800)

plot(ssdse.SQ\_SRhalfdistD2)

dev.off()

####

########################################################################################

#

# hclust関数の分析結果と比較するために、

#

# ward法 独自実装の結果 から、

# S3オブジェクトの "hclust"クラス のリスト要素を編集する処理

# 実証実験1で作成した ssdse.hclust を参考に編集

#

########################################################################################

##########

# CV[k,i] から hclust$order 相当（都道府県の並び順）を作り出すプログラム

#

#######

CV\_copy <- read.csv("ssdse\_halfdistD2\_CV.csv", row.name=1, header=TRUE)

CV\_copy <- as.matrix(CV\_copy)

SSDSE2020CEDITED <- read.csv("ssdse\_2020c\_edited.csv", row.name=1, header=TRUE)

PREF\_name <- rownames(SSDSE2020CEDITED)

K <- nrow(CV\_copy)

N <- ncol(CV\_copy)

for(k in K:2){

CV\_diff <- CV\_copy[k,] - CV\_copy[k-1,]

max\_CV\_diff <- max(CV\_diff)

diff\_Vec <- which(CV\_diff != 0)

max\_Vec <- which(CV\_diff == max\_CV\_diff)

min\_vec <- which((CV\_diff != 0) & (CV\_diff != max\_CV\_diff) )

diff\_count <- length(diff\_Vec)

max\_count <- length(max\_Vec)

min\_count <- diff\_count - max\_count

start\_col <- diff\_Vec[1]

end\_col <- diff\_Vec[diff\_count]

MAT\_max <- matrix(1,nrow=K,ncol=max\_count)

MAT\_min <- matrix(1,nrow=K,ncol=min\_count)

i <- 1

j <- 1

#### 列の入れ替えを行う

for(h in start\_col:end\_col){

if(CV\_diff[h] == max\_CV\_diff){

MAT\_max[,i] <- CV\_copy[,h]

i <- i + 1

}else{

MAT\_min[,j] <- CV\_copy[,h]

j <- j + 1

}

}

CV\_copy[,start\_col:(start\_col+max\_count-1)] <- MAT\_max[,1:max\_count]

CV\_copy[,(start\_col+max\_count):end\_col] <- MAT\_min[,1:min\_count]

}

write.csv(CV\_copy, "CV\_copy.csv")

ORDER <- PREF\_name[CV\_copy[1,]]

write.csv(ORDER,"order.csv")

#########################################################

# S3オブジェクト hclust クラス(ssdse.ORGhclust)を作成する。

# これによって、plot()関数を用いてデンドログラムが描ける。

#

# class "hclust" をコピーする

ssdse.ORGhclust <- ssdse.hclust

ssdse.ORGhclust$merge <- CVmerge

ssdse.ORGhclust$height <- minDS

#CV[]やCV\_copy[]がnamed num であるため、名前無しに変換する

temporder <- c(1:47)

temporder <- CV\_copy[1,]

names(temporder) <- NULL

ssdse.ORGhclust$order <- temporder

# plot.hclust では、この$methodの内容がそのまま使われる

ssdse.ORGhclust$method <- "Kimura Original ward"

# languageオブジェクト の $call には quote関数で値を入力する

# plot.hclust では、タイトル表示に Originalward と ssdse.halfdist.mat が使われる

ssdse.ORGhclust$call <- quote(OriginalWard(d = ssdse.halfdist.mat, method="ward.D") )

################################################

# 独自実装プログラムによるward法分析結果(ssdse.ORGhclust)を、plot()関数によってデンドログラム描画する

################################################

plot(ssdse.ORGhclust)

# pngファイルに描き出し 「ESTRELA」2022年4月号（その1）の記事の図2

png("ssdse.ORGhclust.dendrogram.png",width = 1000, height = 800)

plot(ssdse.ORGhclust)

dev.off()