```
# Partition
require('Cairo')
require('cluster')
require('stringi')

MIN_CLUSTERS <- 6
CLUSTERS <- 6
CLUSTER_COLORS <- c('blue','darkturquoise','green3','deeppink','gold1','red')
IS_BW <- TRUE # Black-white mode

runMDS <- function(dss, nclusters) {
  # Subdirectory for organization of clusters
  clustdir <- paste(outdir, sprintf("%i clusters", nclusters), sep="\\")
  if (!file.exists(clustdir)) {
    dir.create(clustdir);
  }

  # Compute clusters for points
  cl <- pam(dss, nclusters, diss=FALSE)

  # MDS display
  results <- cmdscale(dss, k=3, eig=TRUE)

  x <- results$points[,1]
  y <- results$points[,2]
  z <- results$points[,3]

  names(x) <- iconv(names(x), to="UTF-8")
  names(y) <- iconv(names(y), to="UTF-8")
  names(z) <- iconv(names(z), to="UTF-8")

  CairoPNG(file=paste(clustdir, sprintf("%s - %i clusters MDS Plot 1x2.png", label,
nclusters), sep="\\"),width=1000,height=1000)
  plot(x, y, type="n", xlab="Dimension 1", ylab="Dimension 2")
  points(x, y+.01, col=CLUSTER_COLORS[cl$clustering])
  text(x, y, rownames(results$points), cex=1.0, offset=5.0,
col=CLUSTER_COLORS[cl$clustering])
  dev.off()

  CairoPNG(file=paste(clustdir, sprintf("%s - %i clusters MDS Plot 1x3.png", label,
nclusters), sep="\\"),width=1000,height=1000)
  plot(x, z, type="n", xlab="Dimension 1", ylab="Dimension 3")
  points(x, z+.01, col=CLUSTER_COLORS[cl$clustering])
  text(x, z, rownames(results$points), cex=1.0, offset=5.0,
col=CLUSTER_COLORS[cl$clustering])
  dev.off()

  fileConn<-file(paste(clustdir, sprintf("%s - %i clusters MDS Coordinates.txt", label,
nclusters), sep="\\"))
  sink(fileConn, append=FALSE, split=TRUE)

  cat('Dimension 1\n')
  print(round(x[sort.list(as.numeric(x))], 3))

  cat('\n\nDimension 2\n')
  print(round(y[sort.list(as.numeric(y))], 3))

  cat('\n\nDimension 3\n')
  print(round(z[sort.list(as.numeric(z))], 3))
  sink()
  close(fileConn)
}
```

```
#####################################################################
#
# Function to render clustered results
#
#####################################################################

displayResults <- function(dss, nclusters, clusterMatches) {
  # Subdirectory for organization of clusters
  clustdir <- paste(outdir, sprintf("%i clusters", nclusters), sep="\\")
  if (!file.exists(clustdir)) {
    dir.create(clustdir);
  }

  # Subdirectory for output related to individual mss
  mssdir <- paste(clustdir, 'mss', sep="\\")
  if (!file.exists(mssdir)) {
    dir.create(mssdir);
  }

  # Switch out names since long names are desired for silhouette
  mediumNames = row.names(dss)

  # Short names for silhouette
  row.names(dss) <- shortNames
  cl <- pam(dss, nclusters, diss=FALSE)
  sil <- silhouette(cl)

  # Dates
  row.names(dss) <- DATE_YEARS_NEW[dateIndicesNew]
  clDates <- pam(dss, nclusters, diss=FALSE)

  # Restore medium names
  row.names(dss) <- mediumNames

  cl <- pam(dss, nclusters, diss=FALSE)

  if (IS_BW) {
    CairoPNG(file=paste(clustdir, sprintf("%s - %i clusters by level BW.png", label,
nclusters), sep="\\"),width=600,height=600)
    plot(cl, which=1, labels=4, lines=0, cex=1.2, main='Readings by Local Genealogical
Level', plotchar=FALSE, col.p='black', pch=c('1','2','3')[nodeLevels])
    dev.off()
  } else {
    CairoPNG(file=paste(clustdir, sprintf("%s - %i clusters by level.png", label,
nclusters), sep="\\"),width=1000,height=1000)
    plot(cl, which=1, labels=2, cex.txt=1, col.p=LEVEL_COLORS[nodeLevels])
    dev.off()
  }

  # Old dates
  CairoPNG(file=paste(clustdir, sprintf("%s - %i clusters by date.png", label,
nclusters), sep="\\"),width=1000,height=1000)
  plot(cl, which=1, labels=2, cex.txt=1, col.p=DATE_COLORS[dateIndices])
  dev.off()

  # New dates
  if (IS_BW) {
    CairoPNG(file=paste(clustdir, sprintf("%s - %i clusters by date BW.png", label,
nclusters), sep="\\"),width=600,height=600)
    plot(clDates, which=1, labels=2, lines=0, cex=1.5, cex.txt=1, main='Readings by Dated
Witnesses', plotchar=FALSE, col.p='black', pch=c(' '))
    dev.off()
```

```
    CairoPNG(file=paste(clustdir, sprintf("%s - %i clusters by date BW (closeup).png",
label, nclusters), sep="\\"),width=2400,height=2400)
    plot(clDates, which=1, labels=2, lines=0, cex=3, cex.txt=2, main='Readings by Dated
Witnesses', plotchar=FALSE, col.p='black', pch=c(' '))
    dev.off()
  }

  CairoPNG(file=paste(clustdir, sprintf("%s - %i clusters by color code.png", label,
nclusters), sep="\\"),width=1000,height=1000)
  plot(cl, which=1, labels=2, cex.txt=1, col.p=CLUSTER_COLORS[cl$clustering])
  dev.off()

  LATINS <- 8
  # Graphical distribution of Latin witnesses among clusters (multiplots)
  if (IS_BW) {
    plotname <- "%s - %i clusters by Latin Witness BW.png"
  } else {
    plotname <- "%s - %i clusters by Latin Witness.png"
  }
  for (i in 1:LATINS) {
    if (i %% LATINS == 1) {
      CairoPNG(file=paste(clustdir, sprintf(plotname, label, nclusters),
sep="\\"),width=1600,height=900)
      par(mfrow = c(2, 4))
    }
    if (IS_BW) {
      plot(cl, which=1, labels=4, lines=0, shade=FALSE, main="",
xlab=colnames(inData)[i], ylab="", cex.lab=4, cex=c(2,3)[(inData+1)[,i]], plotchar=FALSE,
col.p='black', pch=c(5,18)[(inData+1)[,i]])
    } else {
      plot(cl, which=1, labels=4, lines=0, shade=TRUE, main="", xlab=colnames(inData)[i],
ylab="", cex.lab=4, cex=4, col.p=c('black','red')[(inData+1)[,i]])
    }
  }
  dev.off()

  # Graphical distribution of Greek witnesses among clusters (multiplots)
  if (IS_BW) {
    plotname <- "%s - %i clusters by Greek Witness BW.png"
  } else {
    plotname <- "%s - %i clusters by Greek Witness.png"
  }
  for (i in LATINS+1:nwitnesses-LATINS) {
    if (i %% nwitnesses-LATINS == 1) {
      CairoPNG(file=paste(clustdir, sprintf(plotname, label, nclusters),
sep="\\"),width=3200,height=1700)
      par(mfrow = c(4, 8))
    }
    if (IS_BW) {
      plot(cl, which=1, labels=4, lines=0, shade=FALSE, main="",
xlab=colnames(inData)[i], ylab="", cex.lab=4, cex=c(2,3)[(inData+1)[,i]], plotchar=FALSE,
col.p='black', pch=c(5,18)[(inData+1)[,i]])
    } else {
      plot(cl, which=1, labels=4, lines=0, shade=TRUE, main="", xlab=colnames(inData)[i],
ylab="", cex.lab=4, cex=4, plotchar=TRUE, col.p=c('black','red')[(inData+1)[,i]])
    }
  }
  dev.off()

  # Plot of Greek Mainstream and Old Latin layers and Alexandrian base
  CairoPNG(file=paste(clustdir, sprintf("%s - %i clusters w Holmes' layers BW.png",
label, nclusters), sep="\\"),width=600,height=600)
```

```
  plot(cl, which=1, labels=4, lines=0, main=sprintf("Readings supported by key layers"),
cex=2, plotchar=FALSE, col.p='black', pch=c(1,18,4,17)[nodeLayers])
  dev.off()

  CairoPNG(file=paste(clustdir, sprintf("%s - %i clusters w Holmes' layers BW
(letters).png", label, nclusters), sep="\\"),width=600,height=600)
  plot(cl, which=1, labels=4, lines=0, main=sprintf("Readings supported by key layers"),
cex=2, plotchar=FALSE, col.p='black', pch=c('A','B','?','L')[nodeLayers])
  dev.off()

  WITNESS_COLORS =
c('red','chartreuse3','darkgoldenrod3','turquoise4','coral2','green4','darkorchid4','blue
3','darkred','springgreen4','violetred3','steelblue4','slateblue3','sienna3','tan3','toma
to3','firebrick2','deepskyblue2','deeppink','royalblue3','lightsalmon3','mediumorchid3')

  clustmatch <- integer(0)
  cnames <- character(0)
  for (j in 1:nclusters){
    cnames <- c(cnames, c(sprintf("%i.C%i", nclusters, j)))
  }
  cnames <- c(cnames, c(sprintf("%i.SUM", nclusters)))
  for (i in 1:nwitnesses){
    # Graphical distribution of each witness per cluster
    if (IS_BW) {
      CairoPNG(file=paste(mssdir, sprintf("%s - %i clusters with %s BW.png", label,
nclusters, colnames(inData)[i]), sep="\\"),width=600,height=600)
      plot(cl, which=1, labels=0, lines=0, main=sprintf("Readings supported by %s",
colnames(inData)[i]), cex.txt=3, cex=c(1,2)[(inData+1)[,i]], plotchar=FALSE,
col.p='black', pch=c(5,18)[(inData+1)[,i]])
      dev.off()
    } else {
      CairoPNG(file=paste(mssdir, sprintf("%s - %i clusters with %s.png", label,
nclusters, colnames(inData)[i]), sep="\\"),width=1000,height=1000)
      plot(cl, which=1, labels=2, cex.txt=1,
col.p=c('black',WITNESS_COLORS[(i%%length(WITNESS_COLORS))+1])[(inData+1)[,i]])
      dev.off()
    }

    # How many times does current witness occur in current cluster?
    crow <- rep(0, nclusters+1)
    for (j in 1:nclusters){
      crow[j] <- length(which(cl$clustering[which(inData[,i]==1)]==j))
    }
    crow[nclusters+1] <- sum(crow[1:nclusters])
    clustmatch <- rbind(clustmatch, crow)
  }
  colnames(clustmatch) <- cnames
  clusterMatches <- cbind(clusterMatches, clustmatch)

  clusters <- cl$clustering

  fileConn<-file(paste(clustdir, sprintf("%s - k-means (binary) %i cluster results.txt",
label, nclusters), sep="\\"))
  sink(fileConn, append=FALSE, split=TRUE)

  for (i in 1:nclusters) {
    # display cluster
    cat('CLUSTER ', toString(i), '\n')
    cat(' Coord   Lv Ly Dt Nb Description\n')
    clSil <- sil[which(sil[,'cluster']==i),]
    for (j in 1:dim(clSil)[1]) {
      # display row
      shortName <- row.names(clSil)[j]
```

```
    obsIndex <- match(shortName, shortNames)
    longName <- longNames[obsIndex]
    nodeLayer <- nodeLayers[obsIndex]
    nodeLevel <- nodeLevels[obsIndex]
    dateIndex <- dateIndices[obsIndex]
    width <- round(clSil[,'sil_width'], 4)[j]
    wstr <- toString(width)
    if (width >= 0) { cat(' ') } # for minus sign
    cat(wstr)
    strl <- stri_length(wstr)
    if (width < 0) { strl <- strl - 1 } # move one left for minus sign
    if (strl == 6) {
      cat('  ') }
    else if (strl == 7) { # neg sign
      cat('  ') }
    else if (strl == 5) {
      cat('   ') }
    else if (strl == 4) {
      cat('    ') }
    else if (strl == 3) {
      cat('     ') }
    else {
      cat('    ') }
    cat(nodeLevel)
    cat('  ')
    cat(nodeLayer)
    cat('  ')
    cat(dateIndex)
    if (is.na(dateIndex)) {
      cat(' ') }
    else {
      cat('  ') }
    cat(clSil[,'neighbor'][j])
    cat('  ')
    cat(longName)
    cat('\n')
  }
  cat('\n')
}

cat('\n\nCLUSTER INFO\n')
print(cl$clusinfo)

cat('\n\nCLUSTER ISOLATION\n')
print(cl$isolation)

cat('\n\nMEDOID OBSERVATIONS\n')
medoidNames <- longNames[cl$id.med]
cat('    Index Description\n')
for (i in 1:length(medoidNames)) {
  cat('C')
  cat(toString(i))
  cat('  ')
  cat(cl$id.med[i])
  strl <- stri_length(toString(cl$id.med[i]))
  if (strl == 1) {
    cat('    ') }
  else if (strl == 2) {
    cat('   ') }
  else { # strl == 3
    cat('  ') }
  cat(medoidNames[i])
  cat('\n')
```

```
  }

  row.names(cl$medoids) <- iconv(shortNames[cl$id.med], to="UTF-8")
  colnames(cl$medoids) <- iconv(shortNames, to="UTF-8")

  for (i in 1:nclusters) {
    cat('\n\nCLUSTER ', toString(i), ' COORDINATES\n')
    print(round(cl$medoids[,which(clusters==i)], 4))
  }

  for (i in 1:nclusters) {
    clusterNames <- longNames[which(clusters==i)]
    cat('CLUSTER ', toString(i), '\n')
    for (j in 1:length(clusterNames)) {
      cat(clusterNames[j]) # Original 'verse' order
      cat('\n')
    }
    cat('\n')
  }
  sink()
  close(fileConn)

#  fileConn<-file(paste(clustdir, sprintf("%s - k-means (binary) %i cluster data.txt",
label, nclusters), sep="\\"))
#  sink(fileConn, append=FALSE, split=TRUE)
#  cat(cl$data)
#  sink()
#  close(fileConn)

  clusterMatches
}

#######################################################################
#
# Start script Execution
#
#######################################################################

options(max.print=5E5, echo = FALSE)

# Constants
LEVEL_COLORS <- c('darkgoldenrod2', 'green4', 'red')
DATE_COLORS <- c('red', 'darkorange', 'gold1', 'gold1', 'green4', 'green4','blue','blue')
DATED_WITNESSES <- c('P66','P75','Te','Or','Cy','e.2','X01','Hil')

# Dating indices in order (match returns index of first hit in this vector)
# NA28 for MSS
# Oxford Dictionary of the Christian Church 3d rev ed, 2005
# P66=200, Te=225, Or=254, P75=250, Cy=258, a=350, 01=350, 03=350, Hil=367
DATED_WITNESSES_NEW <- c('P66','Te','P75','Or','Cy','a.3','X01','X03','Hil','d.5')
DATE_YEARS_NEW <- c('200','225','iii','254','258','iv','iv','iv','367','400')



# Script logic
cmdArgs <- commandArgs()

library(yaml)
config <- yaml.load_file(cmdArgs[4])
input <- cmdArgs[5]

label <- config$label[input]
```

```
# Read data
infile <- config$infile[input]

# Create directory for current label if not exists
outdir <- paste(config$outdir, label, sep="")
if (!file.exists(outdir)) {
  dir.create(outdir);
}
outdir <- paste(outdir, 'part', sep="\\")
if (!file.exists(outdir)) {
  dir.create(outdir);
}

# Read data
inFrame <- read.delim(file=toString(infile), strip.white=TRUE)

######################################################################
#
# Extract short, medium, and long row names from data
#
######################################################################

shortNames <- inFrame[,1]
longNames <- iconv(inFrame[,2], to="UTF-8")
nodeLevels <- inFrame[,3]
nodeLayers <- inFrame[,4]

# exclude metadata columns for shortNames + longNames + nodeLevels
inData <- data.matrix(frame=inFrame[,5:dim(inFrame)[2]])
Encoding(rownames(inData)) <- "UTF-8"

# Compute dating indices
dateIndices <- integer()
for (i in 1:dim(inData)[1]) {
  dateIndices <- c(dateIndices, c(match(1,inData[i,DATED_WITNESSES])))
}

# Compute new dating indices
dateIndicesNew <- integer()
for (i in 1:dim(inData)[1]) {
  dateIndicesNew <- c(dateIndicesNew, c(match(1,inData[i,DATED_WITNESSES_NEW])))
}

# exclude four dating indices (excluded from results)
inData <- inData[,5:dim(inData)[2]]
nwitnesses <- dim(inData)[2]

######################################################################
#
# NUMBER OF CLUSTERS - WITHIN GROUPS SUM OF SQUARES (Everitt & Hothorn (pg. 251))
#
# A plot of the within groups sum of squares by number of clusters extracted
# can help determine the appropriate number of clusters. The analyst looks
# for a bend in the plot similar to a scree test in factor analysis.
#
######################################################################

# Determine number of clusters
wss <- (nrow(inData)-1)*sum(apply(inData,2,var))
for (i in 2:15) wss[i] <- sum(kmeans(inData, centers=i)$withinss)

CairoPNG(file=paste(outdir, sprintf("%s - Within groups sum of squares X Number of
Clusters.png", label), sep="\\"),width=1000,height=1000)
```

```
plot(1:15, wss, type="b", xlab="Number of Clusters", ylab="Within groups sum of squares")
dev.off()

#########################################################################
#
# BINARY - PARTITIONING AROUND MEDIOIDS
#
#########################################################################

dss <- as.matrix(dist(inData, method="binary"))

# Determine number of clusters
wss <- (nrow(dss)-1)*sum(apply(dss,2,var))
for (i in 2:15) wss[i] <- sum(kmeans(dss, centers=i)$withinss)

CairoPNG(file=paste(outdir, sprintf("%s - Binary within groups sum of squares X Number of
Clusters.png", label), sep="\\"),width=1000,height=1000)
plot(1:15, wss, type="b", xlab="Number of Clusters", ylab="Within groups sum of squares")
dev.off()

# Track witness occurrences per cluster
clusterMatches <- integer(0)

# Results for different numbers of clusters
for (i in MIN_CLUSTERS:CLUSTERS) {
  clusterMatches <- displayResults(dss, i, clusterMatches)
  runMDS(dss, i)
}

# Produce cluster match CSV
row.names(clusterMatches) <- colnames(inData)

write.table(file=paste(outdir, sprintf("cluster-counts.txt", label), sep="\\"),
clusterMatches, sep='\t')
```