# Code for demonstrating PCA analyses in R

TMA = read.table("tma505ver2.txt",header=T)

Y = as.matrix(TMA[,1:3])

hg = factor(TMA[,5])

nuc = factor(TMA[,6])

# Does PCA of TMA data

pca1 = prcomp(Y)

pca1 = prcomp(Y,scale=T) # Talk about the scale option

# results deal use scale =T option

summary(pca1)

> summary(pca1)

Importance of components:

PC1 PC2 PC3

Standard deviation 0.268 0.205 0.0575

Proportion of Variance 0.611 0.360 0.0283

Cumulative Proportion 0.611 0.972 1.0000

# Gives a screeplot

plot(pca1) # Could also use screeplot(pca1)

# Plot scores from first and second principal components

plot(pca1$x[,1],pca1$x[,2])

# See if anything about grade or nuclear staining correlations

# with the PC scores occurs or not

# First, with grade (high = 1 (H); low = 0 (L))

plot(pca1$x[,1],pca1$x[,2],type="n")

points(pca1$x[,1][TMA[,5] == 0],pca1$x[,2][TMA[,5] == 0],pch="L")

points(pca1$x[,1][TMA[,5] == 1],pca1$x[,2][TMA[,5] == 1],pch="H")

# Next, with nuclear staining (stains = 1 (S); no stain = 0 (N))

plot(pca1$x[,1],pca1$x[,2],type="n")

points(pca1$x[,1][TMA[,6] == 0],pca1$x[,2][TMA[,6] == 0],pch="S")

points(pca1$x[,1][TMA[,6] == 1],pca1$x[,2][TMA[,6] == 1],pch="N")

# Another idea: fit regression models with scores as

# predictors

glm1 = glm(TMA[,5] ~ pca1$x[,1]+pca1$x[,2],family=binomial)

glm2 = glm(TMA[,6] ~ pca1$x[,1]+pca1$x[,2],family=binomial)

summary(glm1)

summary(glm2)

# Bootstrap for constructing CI's for eigenvalues

obs = pca1$sdev^2

bootval = matrix(0,1000,3)

set.seed(1031)

for (i in 1:1000) {

newY = Y[sample(1:50,size=50,replace=T),]

tmp = prcomp(newY,scale=T)

bootval[i,] = tmp$sdev^2

}

c(quantile(bootval[,1],0.01),quantile(bootval[,1],1-0.01))

c(quantile(bootval[,2],0.01),quantile(bootval[,2],1-0.01))

c(quantile(bootval[,3],0.01),quantile(bootval[,3],1-0.01))

> c(quantile(bootval[,1],0.01),quantile(bootval[,1],1-0.01))

1% 99%

1.628752 1.983453

>

> c(quantile(bootval[,2],0.01),quantile(bootval[,2],1-0.01))

1% 99%

0.9088037 1.3106216

>

> c(quantile(bootval[,3],0.01),quantile(bootval[,3],1-0.01))

1% 99%

0.02467718 0.14386740

# Biplot

biplot(pca1)

# R code for mixture models

# Author: Debashis Ghosh

# Iris data

library(mclust)

# hierarchical classification

hcVVViris <- hc(modelName = "VVV", data = iris[,-5])

cl = hclass(hcVVViris) # Default: look at 1....9 groups

clPairs(data = iris[,-5], classification = cl[,"8"]) # Pretty pictures

# do both 1) and 2)

# automatically does all models from 1...9 groups (i.e., default for modelnames)

# to look at model parameter estimates, need to reduce models considered

iris.mm1 = Mclust(iris[,-5],modelnames=c("VEV","EII","EEE"))

SIR

# R code for implementing

# sliced inverse regression

# available from the dr library

TMA = read.table("tma505ver2.txt",header=T)

library(dr)

sir1 = dr(hg~mild+moderate+medium,data=TMA)

> summary(sir1)

Call:

dr(formula = hg ~ mild + moderate + medium, data = TMA)

Method:

sir with 2 slices, n = 50.

Slice Sizes:

16 34

Estimated Basis Vectors for Central Subspace:

mild moderate medium

0.1615 0.7010 0.6946

Dir1

Eigenvalues 0.1053

R^2(OLS|dr) 1.0000

Large-sample Marginal Dimension Tests:

Stat df p.value

0D vs >= 1D 5.267 3 0.1532

> names(sir1)

> names(sir1)

[1] "x" "y"

[3] "weights" "method"

[5] "cases" "qr"

[7] "group" "chi2approx"

[9] "evectors" "evalues"

[11] "numdir" "raw.evectors"

[13] "M" "slice.info"

[15] "call" "y.name"

[17] "terms"