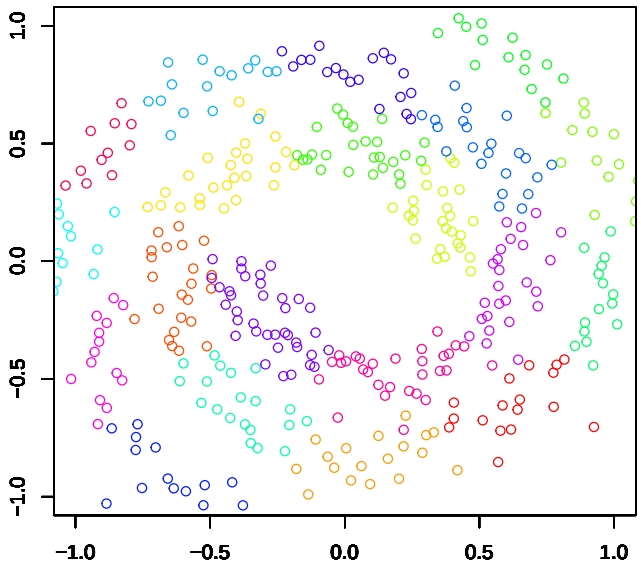
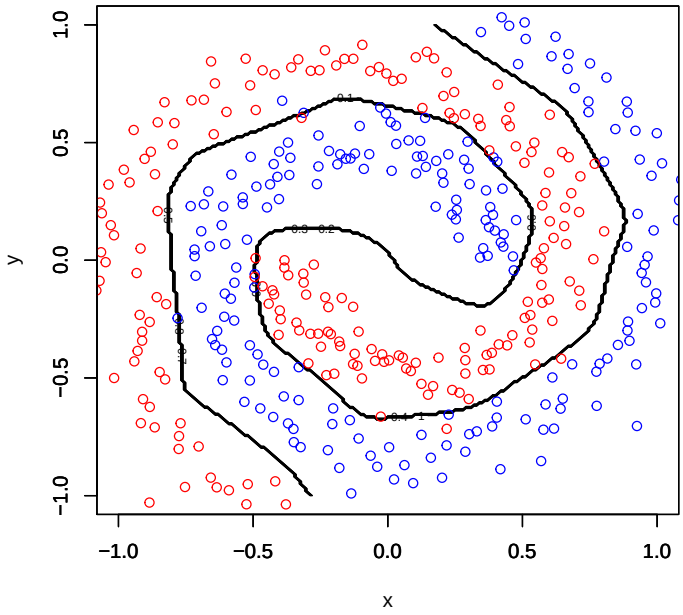
```
> rm(list=ls())
> library('MASS')
> library('plot3D')
> library('mlbench')
> distance <- function(xt, centers){
+   distMatrix <- matrix(NA, nrow=dim(xt)[1], ncol=dim(centers)[1])
+   for(i in 1:nrow(centers)) {
+     distMatrix[,i] <- sqrt(rowSums(t(t(xt)-centers[i,])^2))
+   }
+   distMatrix
+ }
> mykmeans <- function(x, k, maxIter) {
+   clusterOld <- c()
+   centerOld <- c()
+   centers <- x[sample(nrow(x), k),]
+
+   flag <- FALSE
+   i <- 0
+   while(i <= maxIter && flag==FALSE) {
+     i <- i + 1
+     if(i > 1) {
+       clusterOld <- clusters
+       centerOld <- centers
+     }
+     distsToCenters <- distance(x, centers)
+     clusters <- apply(distsToCenters, 1, which.min)
+     centers <- apply(x, 2, tapply, clusters, mean)
+     flag <- identical(clusters,clusterOld)
+   }
+
+   return(list(clusters=clusters, centers=centers))
+ }
> pdfnvar <- function(x,m,k,V){
+   ((1/(sqrt(((2*pi)^k)*det(V))))
+    *exp(-0.5*(t(x-m) %*% solve(V) %*% (x-m))))}
> class <- function(pxc1,pxc2,pc1,pc2){ifelse(((pxc1/pxc2) >= (pc2/pc1)), 1, 2)}

> ######################################################################
> k <- c(10,10)
> cores <- rainbow(k[1]+k[2])
> maxIter <- 100
> # sd2 <- (c(0.3, 0.4)^2)
> N <- 400
> minseq <- -1.0
> maxseq <- 1.0
> ######################################################################
```

```
> # S1 <- matrix(c(sd2[1],0,0,sd2[1]),byrow=T,ncol=2)
> # c1g1 <- mvrnorm(N,mu=c(1,1), Sigma=S1)
> # c1g2 <- mvrnorm(N,mu=c(2,2), Sigma=S1)
> # c1g3 <- mvrnorm(N,mu=c(1,3), Sigma=S1)
> # cg1 <- rbind(c1g1,c1g2,c1g3)
> #
> # S2 <- matrix(c(sd2[2],0,0,sd2[2]),byrow=T,ncol=2)
> # c2g1 <- mvrnorm(N,mu=c(4,3), Sigma=S2)
> # c2g2 <- mvrnorm(N,mu=c(3,5), Sigma=S2)
> # c2g3 <- mvrnorm(N,mu=c(5,5), Sigma=S2)
> # c2g4 <- mvrnorm(N,mu=c(5,3), Sigma=S2)
> # cg2 <- rbind(c2g1,c2g2,c2g3,c2g4)
>
> ######################################################################
> spirals <- mlbench.spirals(N, cycles=1.2, sd=0.08)
> cg1 <- spirals$x[which(spirals$classes==1),]
> cg2 <- spirals$x[which(spirals$classes==2),]
> ######################################################################
> c1 <- mykmeans(cg1, k[1], maxIter)$clusters
> c2 <- mykmeans(cg2, k[2], maxIter)$clusters
> for(i in 1:k[1])
+ {
+   plot(cg1[c1==i,1],cg1[c1==i,2],type='p',col=cores[i],xlab='',ylab='',xlim=c(minseq,maxse
+   par(new=T)
+ }
> for(i in 1:k[2])
+ {
+   plot(cg2[c2==i,1],cg2[c2==i,2],type='p',col=cores[k[1]+i],xlab='',ylab='',xlim=c(minseq,
+   par(new=T)
+ }
```

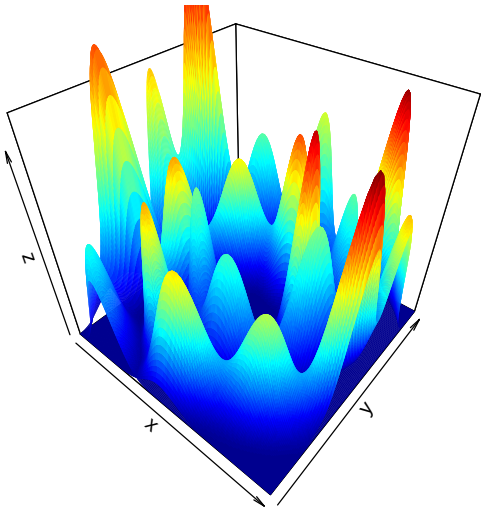



```
> ########################################################################
> seqi <- seq(minseq,maxseq,0.01)
> seqj <- seq(minseq,maxseq,0.01)
> pxc1 <- matrix(0,nrow=length(seqi),ncol=length(seqj))
> pxc2 <- matrix(0,nrow=length(seqi),ncol=length(seqj))
> ########################################################################
> pc1 <- dim(cg1)[1]/(dim(cg1)[1]+dim(cg2)[1])
> pc2 <- dim(cg2)[1]/(dim(cg1)[1]+dim(cg2)[1])
> ########################################################################
> ci <- 0
> for (i in seqi)
+ {
+   cj <- 0
+   ci<- ci + 1
+   for (j in seqj)
+   {
+     cj <- cj + 1
+
+     for(m in 1:k[1])
+     {
+       aux <- cg1[c1==m,]
+       pxc1[ci,cj]<- pxc1[ci,cj] + pdfnvar(c(i,j),colMeans(aux),dim(cg1)[2],cov(aux))
```

```
+     }
+     for(m in 1:k[2])
+     {
+       aux <- cg2[c2==m,]
+       pxc2[ci,cj]<- pxc2[ci,cj] + pdfnvar(c(i,j),colMeans(aux),dim(cg2)[2],cov(aux))
+     }
+   }
+ }
> classx <- 1*(pxc1 > (pc2/pc1)*pxc2)
> ######################################################################
> contour2D(classx,seqi,seqj,col='black')
> par(new=T)
> plot(cg1[,1],cg1[,2],type='p',col='blue',xlab='',ylab='',xlim=c(minseq,maxseq),ylim=c(mins
> par(new=T)
> plot(cg2[,1],cg2[,2],type='p',col='red',xlab='',ylab='',xlim=c(minseq,maxseq),ylim=c(minse
```




```
> ######################################################################
> persp3D(seqi,seqj,pxc1,colkey=F,sub='')
> persp3D(seqi,seqj,pxc2,add=T,colkey=F,sub='')
```


z
x
y