Supervised Dimension Reduction

S. Tonini, F. Chiaromonte (special thanks to J. Di Iorio, L. Insolia and L. Testa)

March 13th 2023

Introduction

Libraries

We are going to use:

```
library(tidyverse)  # data manipulation and visualization
library(plotly)     # plots in 3D
library(ggplot2)    # plots in 2D
library(ggpubr)     # to combine multiple ggplot objects (ggarrenge)
library(mvtnorm)    # to generate multivariate normal distribution
library(dr)         # SIR
library(factoextra) # PCA-related functions
```

Data

Let's first define a function to generate Gaussian data. This function takes four arguments:

- n: number of observations;
- center: the mean vector
- sigma: the covariance matrix
- label: the cluster label

```
generateGaussianData <- function(n, center, sigma, label) {
  data = rmvnorm(n, center, sigma)
  data = data.frame(data)
  names(data) = c("x", "y", "z")
  data = data %>% mutate(class=factor(label))
  data
}
```

Now let's simulate a dataset.

```
covmat <- diag(3)

# cluster 1
n = 200
center = c(2, 8, 6)
sigma = covmat
group1 = generateGaussianData(n, center, sigma, 1)

# cluster 2
n = 200
center = c(12, 8, 6)
sigma = covmat
group2 = generateGaussianData(n, center, sigma, 2)

# cluster 3
n = 200
center = c(22, 8, 6)
sigma = covmat
group3 = generateGaussianData(n, center, sigma, 3)

# all data
df = bind_rows(group1, group2, group3)

head(df)
```

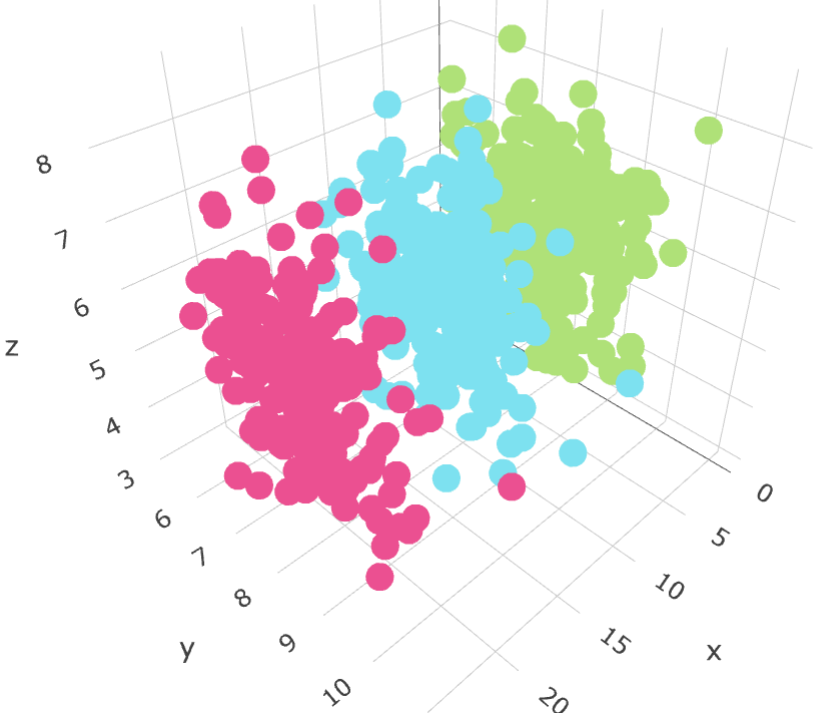
```
##          x        y        z class
## 1 3.216645 8.520231 5.553803     1
## 2 1.392790 7.765584 5.596614     1
## 3 1.509999 9.479470 5.991667     1
## 4 1.505011 8.030041 6.076543     1
## 5 1.602398 7.566892 5.396093     1
## 6 1.501660 8.801586 6.566381     1
```

```
summary(df)
```

```
##        x                y                z         class
##  Min.   :-0.2712   Min.   : 5.193   Min.   :3.078   1:200
##  1st Qu.: 2.8419   1st Qu.: 7.477   1st Qu.:5.197   2:200
##  Median :12.2014   Median : 8.067   Median :5.989   3:200
##  Mean   :12.1182   Mean   : 8.046   Mean   :5.897
##  3rd Qu.:21.3840   3rd Qu.: 8.634   3rd Qu.:6.607
##  Max.   :24.1700   Max.   :11.206   Max.   :8.685
```

And plot our simulated data.

```
fig <- plot_ly(df, x = ~x, y = ~y, z = ~z,
               color = ~class, colors = c('#b3e378', '#81e5f0', '#ed5391'))
fig <- fig %>% add_markers()
fig <- fig %>% layout(scene = list(xaxis = list(title = 'x'),
                                   yaxis = list(title = 'y'),
                                   zaxis = list(title = 'z')))
fig
```



PCA vs LDA

PCA

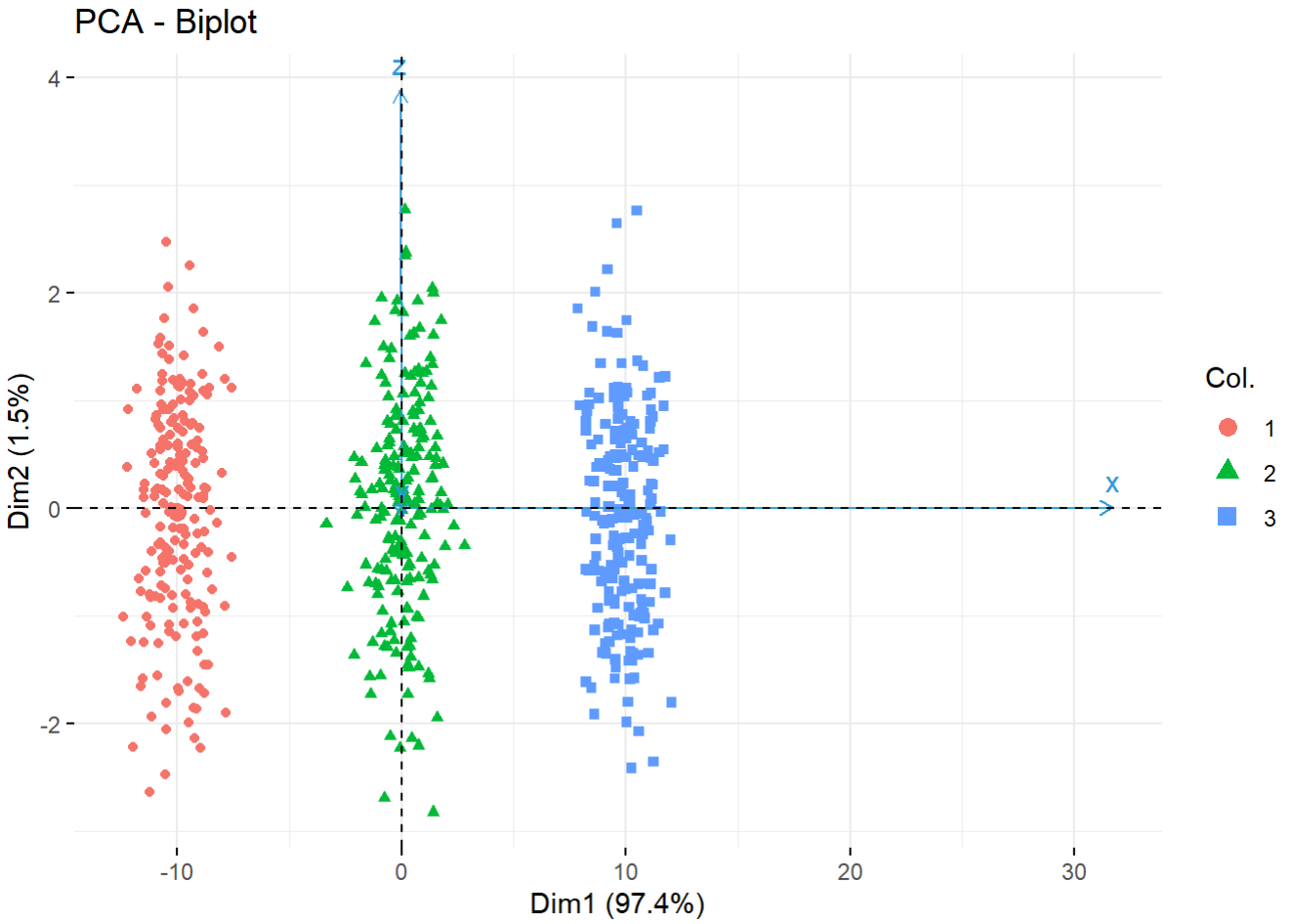
Now let us perform PCA.

```
pc <- prcomp(df[,c(1,2,3)])
get_eig(pc)
```

```
##       eigenvalue variance.percent cumulative.variance.percent
## Dim.1 66.8715472        97.359713                    97.35971
## Dim.2  1.0067725         1.465782                    98.82549
## Dim.3  0.8067096         1.174506                   100.00000
```

This is the corresponding biplot.

```
fviz_pca_biplot(pc, col.var= "#2E9FDF",
                col.ind= df$class, label="var")
```



Note that just considering the first principal component it is possibile to notice differences within the three groups.

LDA

Let's perform LDA:

```
lda.df <- lda(factor(class) ~ x + y + z, data = df)
lda.df
```

```
## Call:
## lda(factor(class) ~ x + y + z, data = df)
##
## Prior probabilities of groups:
##         1         2         3
## 0.3333333 0.3333333 0.3333333
##
## Group means:
##           x        y        z
## 1  2.138319 7.994808 5.876804
## 2 12.202091 8.089994 5.980764
## 3 22.014175 8.053455 5.834857
##
## Coefficients of linear discriminants:
##           LD1           LD2
## x  1.04387176 -9.032245e-05
## y  0.02937824 -5.651024e-01
## z -0.02966913 -8.616465e-01
##
## Proportion of trace:
##    LD1    LD2
## 0.9999 0.0001
```

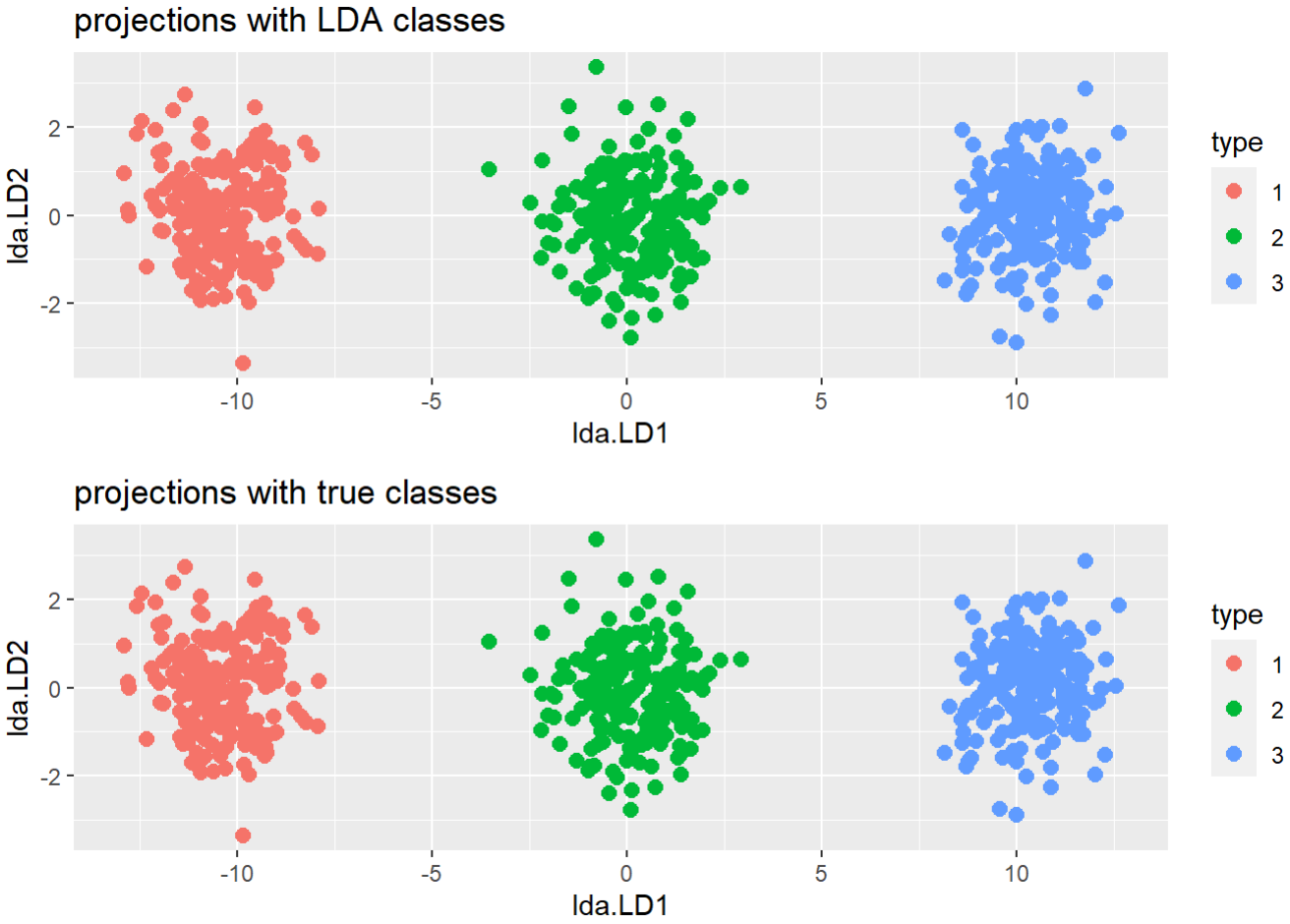
Let us plot the projections on LD1 and LD2

```
# prediction on df to get projections
predmodel.lda = predict(lda.df, data=df)

# projections with LDA classes
estclass <- as.factor(apply(predmodel.lda$posterior, 1, which.max))
newdata2 <- data.frame(type = estclass, lda = predmodel.lda$x)
p1 <- ggplot(newdata2) +
        geom_point(aes(lda.LD1, lda.LD2, colour = type), size = 2.5) +
        ggtitle("projections with LDA classes")

# projections with true classes
newdata <- data.frame(type = df$class, lda = predmodel.lda$x)
p2 <- ggplot(newdata) +
        geom_point(aes(lda.LD1, lda.LD2, colour = type), size = 2.5) +
        ggtitle("projections with true classes")


ggarrange(p1,p2,
          nrow=2)
```



SIR

Now we use the SIR (Sliced Inversion Regression) in the dr package

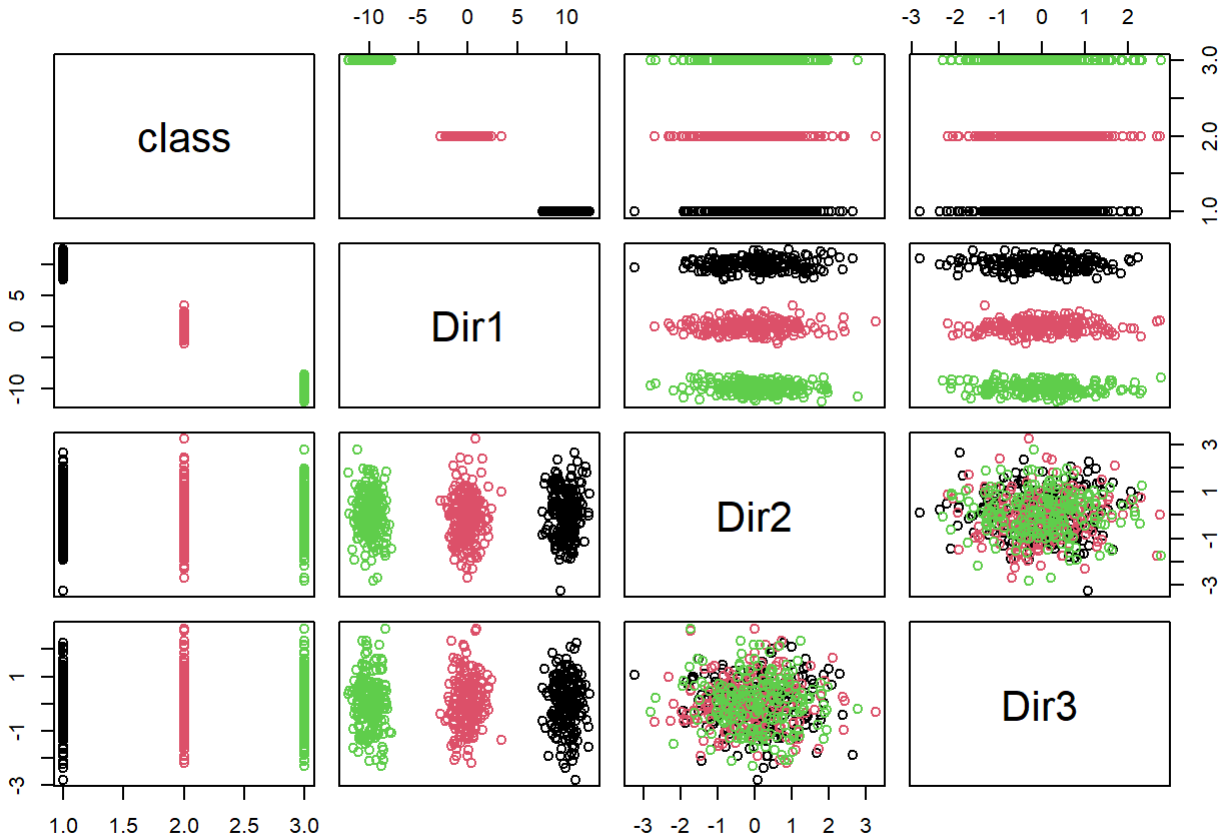
```
# default fitting method is "sir"
help(dr)

dr_res <- dr(class ~ x + y + z, data = df, method='sir')

dr_res
```

```
##
##  dr(formula = class ~ x + y + z, data = df, method = "sir")
## Estimated Basis Vectors for Central Subspace:
##          Dir1          Dir2         Dir3
## x  0.99920102 -8.765554e-05  0.003592764
## y  0.02812104 -5.484169e-01 -0.886083708
## z -0.02839949 -8.362051e-01  0.463511332
## Eigenvalues:
## [1]  9.863258e-01  4.666252e-03 -8.131516e-20
```

```
plot(dr_res, col=df$class)
```




```
names(dr_res)
```

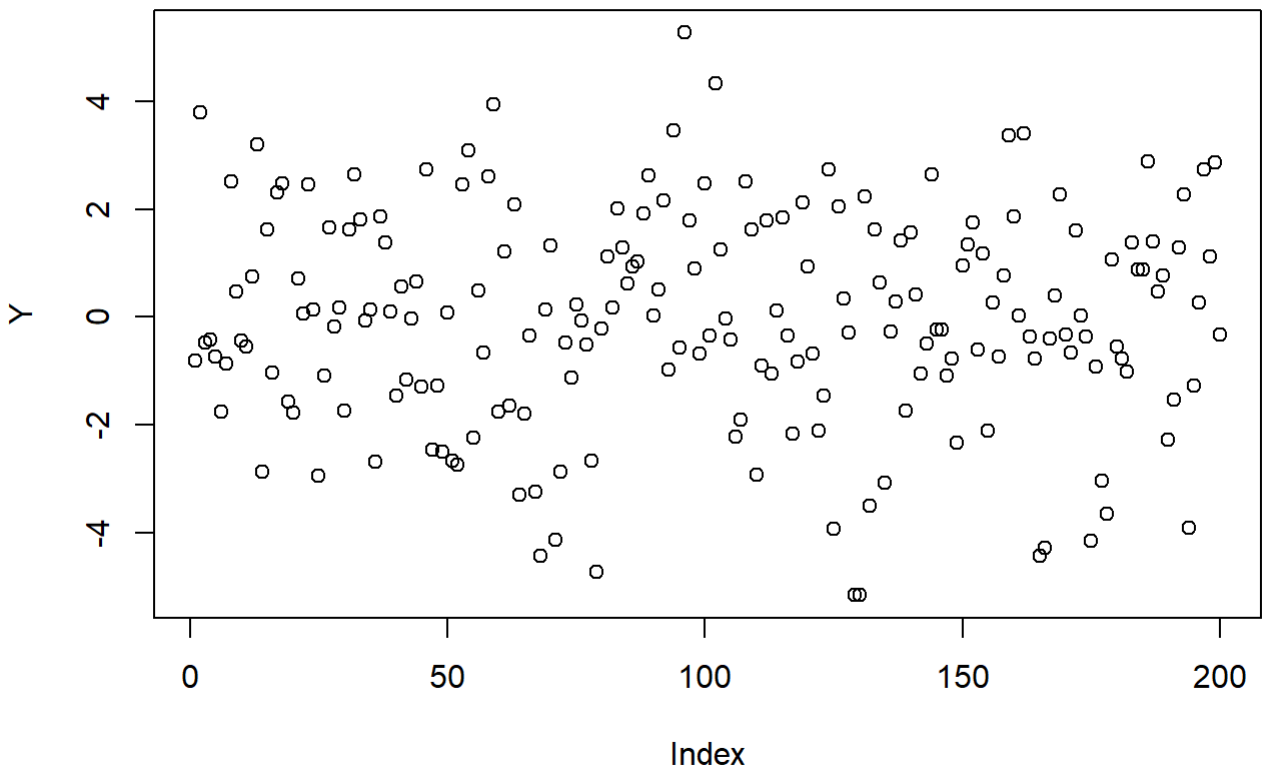
```
##  [1] "x"           "y"           "weights"     "method"      "cases"
##  [6] "qr"          "group"       "chi2approx"  "evectors"    "evalues"
## [11] "numdir"      "raw.evectors" "M"          "slice.info"  "call"
## [16] "y.name"      "terms"
```

SIR with continuous variable

We generated covariates in a classical continuous framework

```
x <- mvrnorm(n, c(0,0,0), diag(3))
eps <- rnorm(n, 0, 1)
Y <- x[,1] + x[,2] + x[,3] + eps
df2 = data.frame(cbind(Y, x))

plot(Y)
```

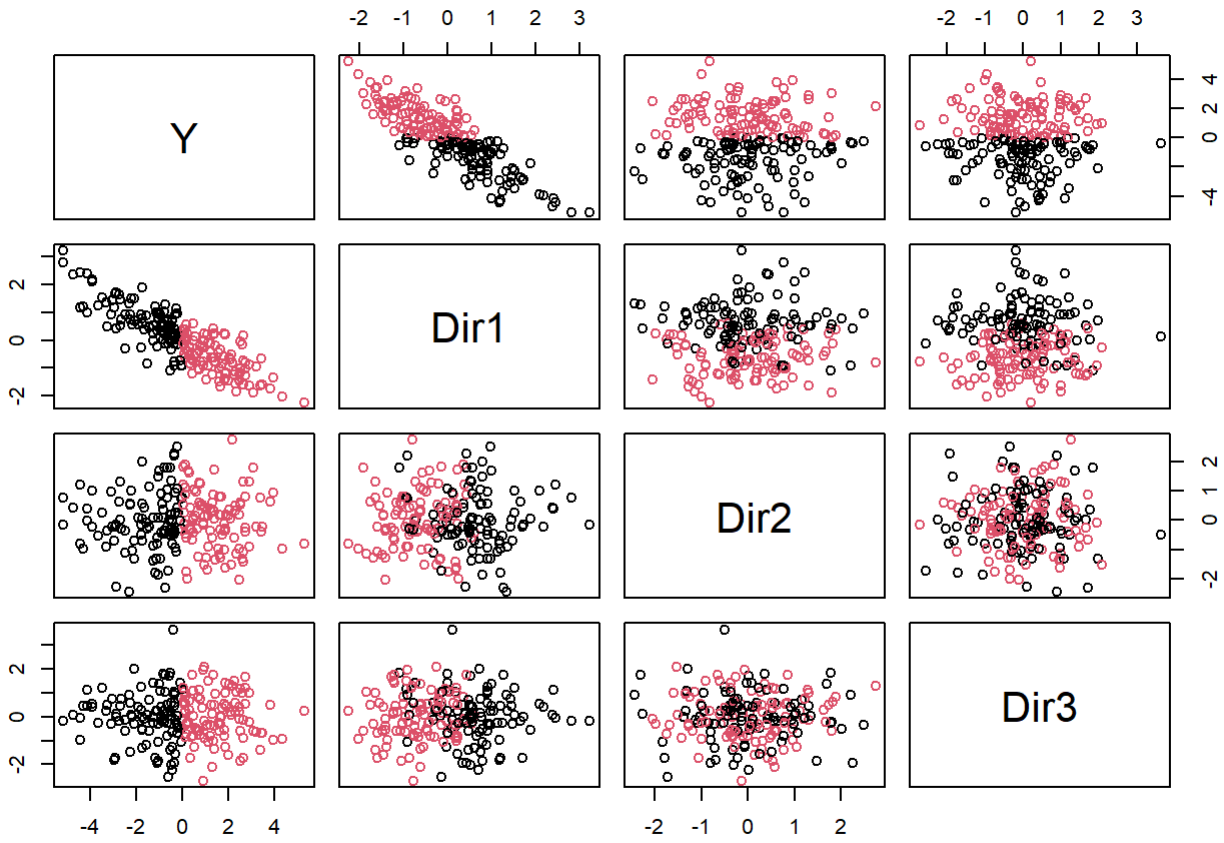



```
dr_res2 <- dr(Y ~ ., data = df2, method='sir', nslices=2)

summary(dr_res2)
```

```
##
## Call:
## dr(formula = Y ~ ., data = df2, method = "sir", nslices = 2)
##
## Method:
## sir with 2 slices, n = 200.
##
## Slice Sizes:
## 100 100
##
## Estimated Basis Vectors for Central Subspace:
##     V2     V3     V4
## 0.5590 0.5500 0.6205
##
##                Dir1
## Eigenvalues 0.4626
## R^2(OLS|dr) 0.9990
##
## Large-sample Marginal Dimension Tests:
##               Stat df p.value
## 0D vs >= 1D 92.51  3       0
```

```
plot(dr_res2, col=dr_res2$slice.info$slice.indicator)
```



Now we perform the same analysis with 5 slices

```
dr_res2 <- dr(Y ~ ., data = df2, method='sir', nslices=4)

summary(dr_res2)
```

```
##
## Call:
## dr(formula = Y ~ ., data = df2, method = "sir", nslices = 4)
##
## Method:
## sir with 4 slices, n = 200.
##
## Slice Sizes:
## 50 50 50 50
##
## Estimated Basis Vectors for Central Subspace:
##      Dir1    Dir2    Dir3
## V2 0.5388  0.8506  0.0688
## V3 0.5631 -0.2267 -0.8463
## V4 0.6266 -0.4745  0.5283
##
##                Dir1    Dir2      Dir3
## Eigenvalues 0.6096 0.02429 0.0001656
## R^2(OLS|dr) 0.9979 0.99829 1.0000000
##
## Large-sample Marginal Dimension Tests:
##                   Stat df p.value
## 0D vs >= 1D 126.81177  9  0.0000
## 1D vs >= 2D   4.89184  4  0.2986
## 2D vs >= 3D   0.03311  1  0.8556
```

```
plot(dr_res2, col=dr_res2$slice.info$slice.indicator)
```