

Capítulo 8

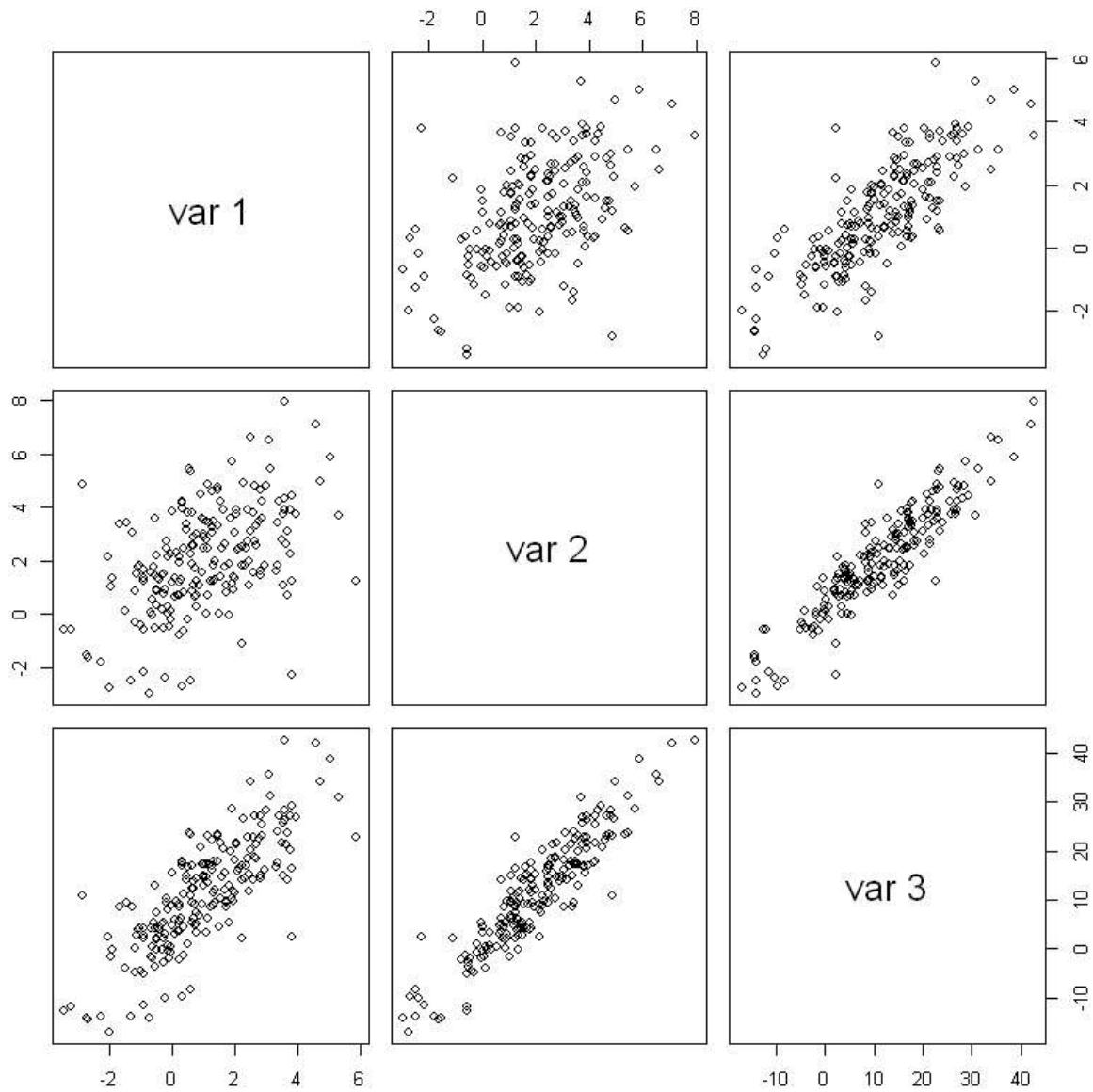
Modelos multivariados gaussianos

8.1 PCA: Componentes Principais

Exercício 2: Neste exercício, voce vai gerar alguns vetores gaussianos tri-dimensionais que, de fato vivem em duas dimensoes.

```
In [7]:  
require(MASS)  
nsims=200  
Sigma = matrix(c(3,2,2,4),2,2)  
pts = mvrnorm(nsims, c(1, 2), Sigma)  
pts = cbind(pts, 3*pts[,1]+4*pts[,2])  
pairs(pts)  
library(scatterplot3d)  
scatterplot3d(pts)  
library(rgl)  
plot3d(pts, col="red", size=3)  
A = matrix(c(1, 0, 0, 1, 3, 4), 3, 2, byrow=T)  
var pts = A %*% Sigma %*% t(A)  
var pts  
round(cov(pts),2)  
eigen(var pts)  
eigen(cov(pts))
```

Loading required package: MASS



```

3   2   17
2   4   22
17  22  139

2.93   1.61   15.23
1.61   3.83   20.17
15.23  20.17  126.37

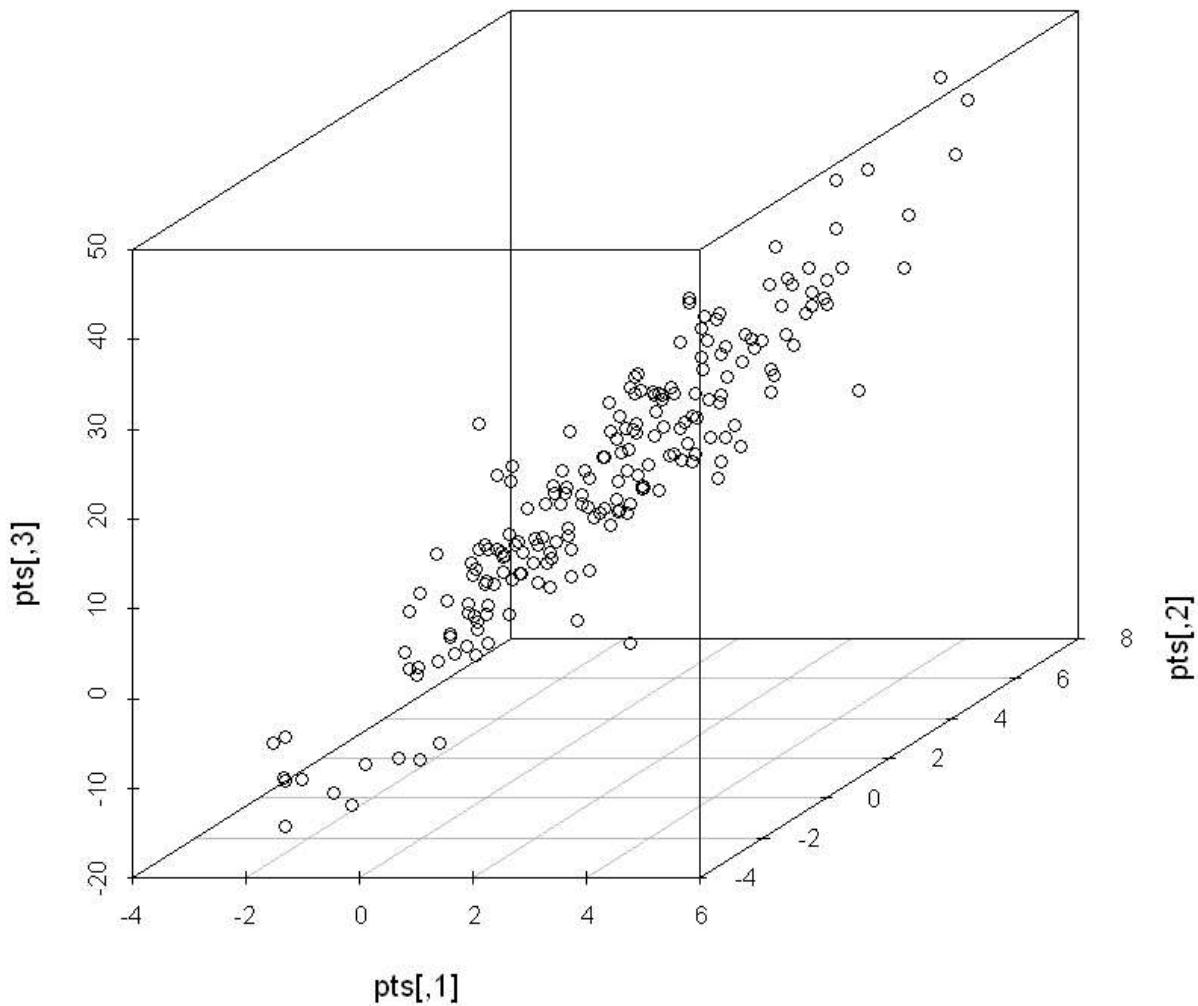
eigen() decomposition
$values
[1] 1.445612e+02 1.438837e+00 9.841374e-15

$vectors
[,1]      [,2]      [,3]
[1,] -0.1199493  0.799661372 -0.5883484
[2,] -0.1551822 -0.600444735 -0.7844645
[3,] -0.9805767 -0.002794823  0.1961161
eigen() decomposition

```

```
$values
[1] 1.314297e+02 1.706904e+00 -5.270794e-15

$vectors
[,1]      [,2]      [,3]
[1,] -0.1182110  0.79992019 -0.5883484
[2,] -0.1564869 -0.60010603 -0.7844645
[3,] -0.9805805 -0.00066356  0.1961161
```

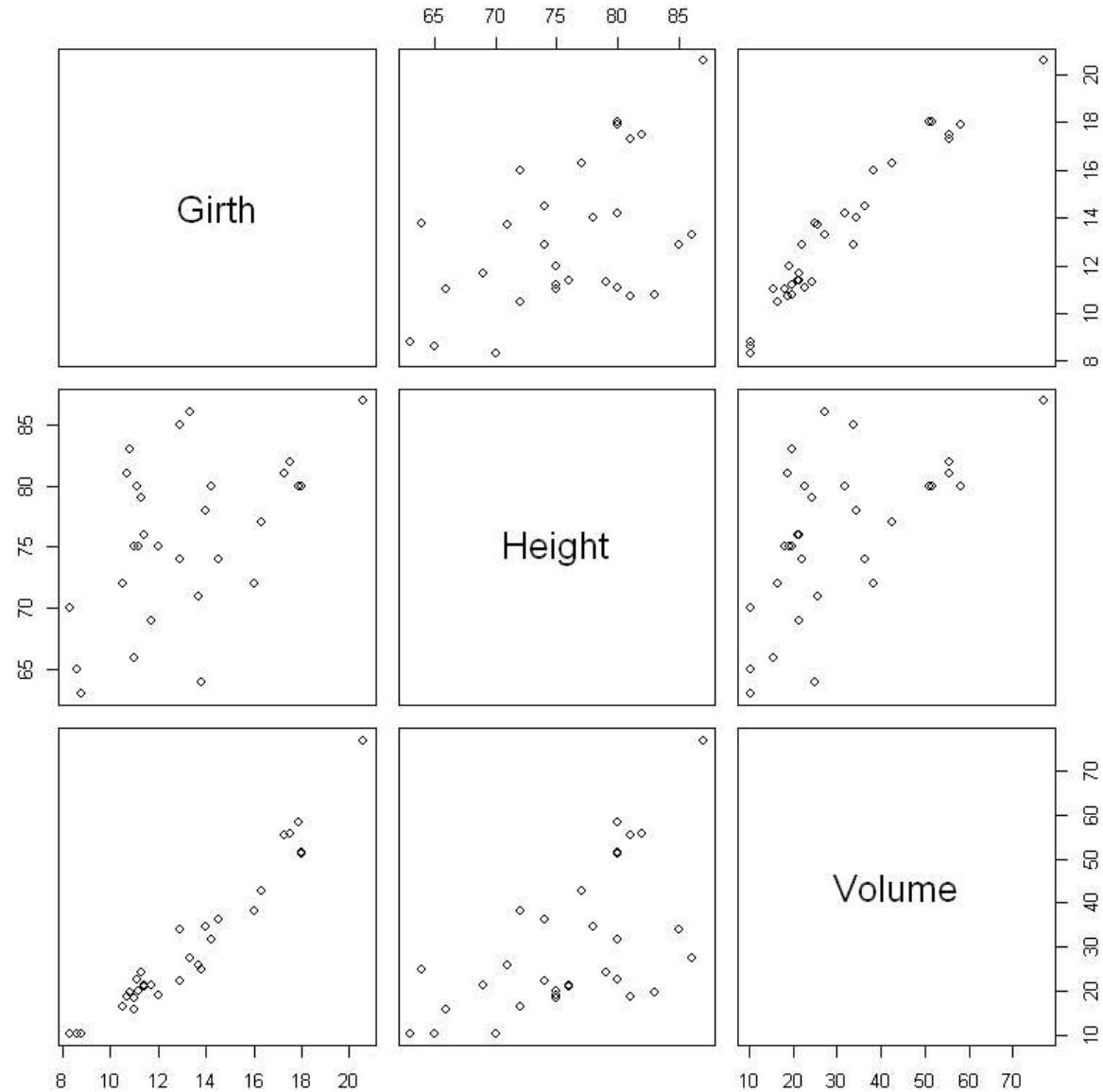


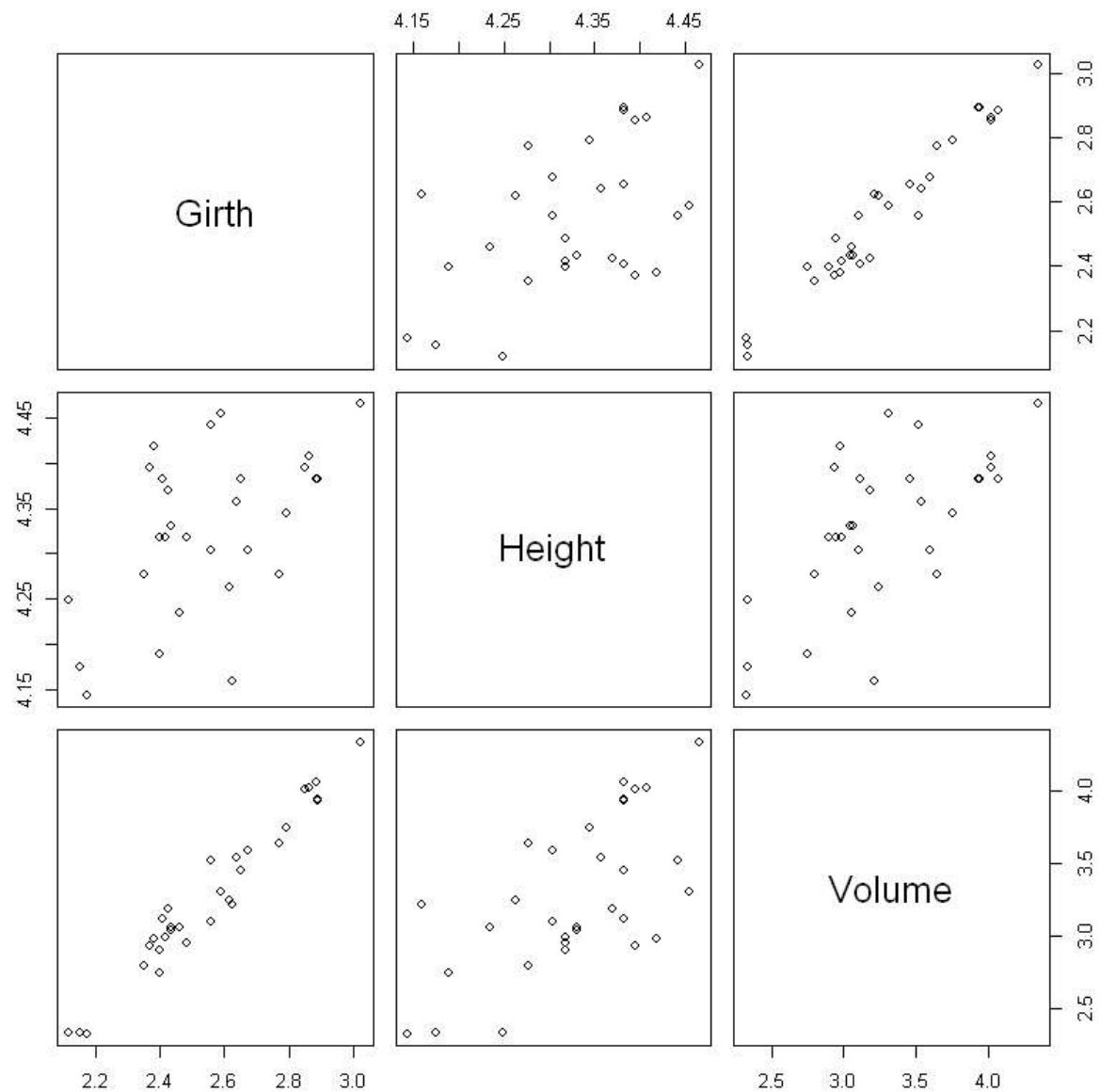
```
In [8]:
x <- rnorm(1000)
y <- rnorm(1000)
z <- 3 + 1.2*x - 1.2*y + rnorm(1000, sd=0.3)
d2 <- data.frame(x,y,z)
open3d()
plot3d(d2)
```

wgl: 2

```
In [3]: ?trees # girth = circunferencia
```

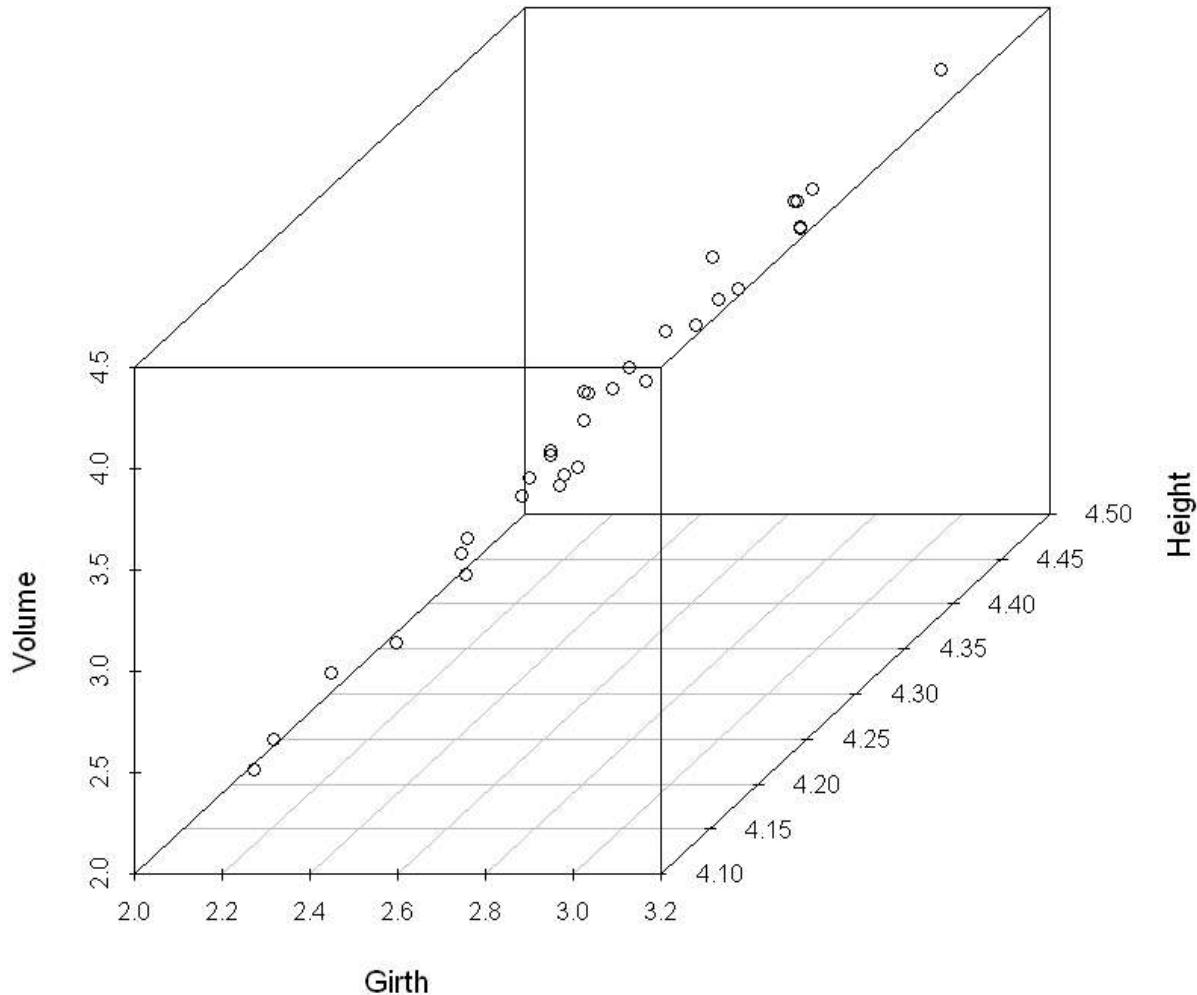
```
pairs(trees)
x= log(trees)
pairs(x)
scatterplot3d(x)
plot3d(x, col="red", size=3)
eigen(cov(x))
```





```
eigen() decomposition
$values
[1] 0.3323932158 0.0055253668 0.0009706266

$vectors
      [,1]      [,2]      [,3]
[1,] -0.39803570  0.4927144  0.7738217
[2,] -0.09514377 -0.8611453  0.4993761
[3,] -0.91242273 -0.1251452 -0.3896453
```



O vetor (X_1, X_2) possui distribuição normal bivariada com vetor esperado $\mu = (1, 2)$ e matriz de

$$\Sigma = \begin{bmatrix} 3 & 2 \\ 2 & 4 \end{bmatrix}$$

covariância

O vetor $X = (X_1, X_2, X_3)$ possui distribuição normal multivariada de dimensão 3 com vetor esperado

$$\mathbb{E}(X) = \mathbb{E}\left(\mathbf{A} \begin{pmatrix} X_1 \\ X_2 \end{pmatrix}\right) = \mathbf{A} \mathbb{E}\left(\begin{pmatrix} X_1 \\ X_2 \end{pmatrix}\right) = \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ 3 & 4 \end{bmatrix} \begin{pmatrix} 1 \\ 2 \end{pmatrix} = \begin{bmatrix} 1 \\ 2 \\ 11 \end{bmatrix}$$

e matriz de covariância dada por

$$\mathbf{A}\Sigma\mathbf{A}' = \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ 3 & 4 \end{bmatrix} \begin{bmatrix} 3 & 2 \\ 2 & 4 \end{bmatrix} \begin{bmatrix} 1 & 0 & 3 \\ 0 & 1 & 4 \end{bmatrix} = \begin{bmatrix} 3 & 2 & 17 \\ 2 & 4 & 22 \\ 17 & 22 & 139 \end{bmatrix} =$$

O objeto `cov(pts)` contém a matriz $\mathbf{A}\Sigma\mathbf{A}'$. O objeto `var.pts` contém uma estimativa empírica desta matriz, uma estimativa baseada nas 200 instâncias de dados que você gerou. Para a amostra de tamanho `nsims`, estas duas matrizes são similares. O comando `round(cov(pts),2)` calcula a estimativa empírica da matriz `var.pts = AΣA'`. Esta última matriz é fixa. A estimativa `cov(pts)` varia de amostra para amostra. Se `nsims` não for muito pequeno, `cov(pts)` e $\mathbf{A}\Sigma\mathbf{A}'$ devem ser parecidas, como é o caso neste exercício. Com a amostra gerada por mim, obtive `min(eigen(var.pts)$values)) igual a 9.841374 × 10 - 15` em `min(eigen(cov(pts))$values)` igual a $6.915267 \times 10 - 15$. Os valores são próximos, ambos próximos de zero. O menor autovalor de `cov.pts` é exatamente zero, e isto pode ser verificado se tentarmos fazer uma decomposição de Cholesky:

In [9]: `chol(var.pts)`

Error in `chol.default(var.pts)`: a submatriz de ordem 3 não é positiva definida
Traceback:

1. `chol(var.pts)`
2. `chol.default(var.pts)`

O algoritmo implementado em R para obter os autovalores de `var.pts` obtém apenas uma aproximação numérica para os reais autovalores e autovetores. De acordo com a página de help da função `eigen`, temos: Computing the eigendecomposition of a matrix is subject to errors on a real-world computer: the definitive analysis is Wilkinson (1965). All you can hope for is a solution to a problem suitably close to x. So even though a real asymmetric x may have an algebraic solution with repeated real eigenvalues, the computed solution may be of a similar matrix with complex conjugate pairs of eigenvalues. O segundo bloco de comandos gera também uma gaussiana tridimensional. Como `x1 = rnorm(1000)` e `x2 = rnorm(1000)` geram independentemente vetores gaussianos $N(0, 1)$ então $(X_1, X_2) \sim N_2(0, bsl2)$ onde $bsl2 = (0, 0)^T$ e I_2 é a matriz identidade 2×2 . O vetor (X_1, X_2, X_3) tem a distribuição de suas duas primeiras coordenadas já determinadas acima:

$$\begin{bmatrix} X_1 \\ X_2 \\ X_3 \end{bmatrix} \sim N_3 \left(\begin{bmatrix} 0 \\ 0 \\ \mu_3 \end{bmatrix}, \begin{bmatrix} 1 & 0 & \sigma_{13} \\ 0 & 1 & \sigma_{23} \\ \sigma_{31} & \sigma_{32} & \sigma_{33} \end{bmatrix} \right)$$

A terceira coordenada $X_3 = 3 + 1.2X_1 - 2.3X_2 + e$ onde $e \sim N(0, 0.32)$, independente de X_1 e X_2 . Assim, os elementos que faltam para determinar a distribuição de (X_1, X_2, X_3) são os seguintes:

$$\mathbb{E}(X_3) = \mathbb{E}(3 + 1.2X_1 - 2.3X_2 + \epsilon) = 3 + 1.2\mathbb{E}(X_1) + 2.3\mathbb{E}(X_2) + \mathbb{E}(\epsilon) = 3 + 0 + 0 + 0 = 3$$

e, pela independencia entre X1, X2 e e,

$$\begin{aligned}\sigma_{33} &= \mathbb{V}(X_3) = \mathbb{V}(3 + 1.2X_1 - 2.3X_2 + \epsilon) \\ &= (1.2)^2\mathbb{V}(X_1) + (-2.3)^2\mathbb{V}(X_2) + \mathbb{V}(\epsilon) \\ &= 1.44 + 5.29 + 1.0 = 7.73\end{aligned}$$

enquanto que

$$\begin{aligned}\sigma_{13} &= \sigma_{31} = \text{Cov}(X_1, X_3) \\ &= \text{Cov}(X_1, 3 + 1.2X_1 - 2.3X_2 + \epsilon) \\ &= (1.2)\text{Cov}(X_1, X_1) - 2.3\text{Cov}(X_1, X_2) + \text{Cov}(X_1, \epsilon) \\ &= 1.2\mathbb{V}(X_1) - 2.3 \times (0) + 0 = 1.2\end{aligned}$$

e

$$\begin{aligned}\sigma_{23} &= \sigma_{32} = \text{Cov}(X_2, X_3) \\ &= \text{Cov}(X_2, 3 + 1.2X_1 - 2.3X_2 + \epsilon) \\ &= 1.2\text{Cov}(X_2, X_1) - 2.3\text{Cov}(X_2, X_2) + \text{Cov}(X_2, \epsilon) \\ &= 1.2 \times 0 - 2.3\mathbb{V}(X_2) + 0 = -2.3\end{aligned}$$

Assim,

$$\begin{bmatrix} X_1 \\ X_2 \\ X_3 \end{bmatrix} \sim N_3 \left(\begin{bmatrix} 0 \\ 0 \\ 3 \end{bmatrix}, \begin{bmatrix} 1 & 0 & 1.2 \\ 0 & 1 & -2.3 \\ 1.2 & -2.3 & 7.73 \end{bmatrix} \right)$$

8.2 Análise Fatorial

- Neste exercicio, voce vai analisar os dados de uma analise química de vinhos. Voce vai ler uma matriz com 178 amostras de diferentes vinhos. Haverá uma linha para cada vinho. A primeira coluna indica o cultivar do vinho (entenda como o tipo de uva usada na fabricacao do vinho) tal como Sauvignon Blanc, Cabernet ou Chardonnay (rotulados como 1, 2 ou 3). As 13 colunas seguintes contem as concentrações de 13 diferentes compostos químicos na amostra. O

objetivo é diferenciar entre os 3 tipos de vinho com base na sua composicao química representada pelo vetor 13-dimensional X. Voce precisa criar uma regra para predizer o tipo de vinho (a primeira coluna) a partir das 13 variaveis de composicao química. Vamos verificar que, ao inves de usarmos as 13 variaveis, poderemos nos basear em dois indices, os dois primeiros PCAs, que resumem toda avariabilidade simultanea das 13 variaes. Estude o script R abaixo. De proposito, ele tem uma quantidade minima de comentarios. Procure identificar o que cada linha esta fazendo. WARNING: o help da funcao prcomp é confuso, misturando PCA e analise fatorial nas explicacoes.

In [3]:

```

arq = "http://archive.ics.uci.edu/ml/machine-learning-databases/wine/wine.data"
wine=read.table(arq, sep=",")
head(wine)
pairs(wine[,2:6])
round(100*cor(wine[,2:14]))
round(apply(wine[,2:14], 2, sd),2)
wine.pca = prcomp(wine[,2:14], scale. = TRUE)
summary(wine.pca)
wine.pca$sdev
sum((wine.pca$sdev)^2)
screeplot(wine.pca, type="lines")
# Barplot das variancias acumuladas
barplot(cumsum(wine.pca$sdev^2)/sum(wine.pca$sdev^2))
# os dois primeiros PCA's explicam aprox 60% da variancia total
# os 5 primeiros explicam aprox 80%
# Os autovetores
dim(wine.pca$rot)
# O 1o autovetor
wine.pca$rot[,1]
# O 2o autovetor
wine.pca$rot[,2]
# Coordenadas dos pontos ao Longo do primeiro componente
fscore1 = wine.pca$x[,1]
# Coordenadas dos pontos ao Longo do segundo componente
fscore2 = wine.pca$x[,2]
# plot dos pontos projetados
plot(fscore1, fscore2, pch="*", col=wine[,1]+8)

```

V1	V2	V3	V4	V5	V6	V7	V8	V9	V10	V11	V12	V13	V14
1	14.23	1.71	2.43	15.6	127	2.80	3.06	0.28	2.29	5.64	1.04	3.92	1065
1	13.20	1.78	2.14	11.2	100	2.65	2.76	0.26	1.28	4.38	1.05	3.40	1050
1	13.16	2.36	2.67	18.6	101	2.80	3.24	0.30	2.81	5.68	1.03	3.17	1185
1	14.37	1.95	2.50	16.8	113	3.85	3.49	0.24	2.18	7.80	0.86	3.45	1480
1	13.24	2.59	2.87	21.0	118	2.80	2.69	0.39	1.82	4.32	1.04	2.93	735
1	14.20	1.76	2.45	15.2	112	3.27	3.39	0.34	1.97	6.75	1.05	2.85	1450
V2	V3	V4	V5	V6	V7	V8	V9	V10	V11	V12	V13	V14	
100	9	21	-31	27	29	24	-16	14	55	-7	7	64	
9	100	16	29	-5	-34	-41	29	-22	25	-56	-37	-19	
21	16	100	44	29	13	12	19	1	26	-7	0	22	

	V2	V3	V4	V5	V6	V7	V8	V9	V10	V11	V12	V13	V14
V5	-31	29	44	100	-8	-32	-35	36	-20	2	-27	-28	-44
V6	27	-5	29	-8	100	21	20	-26	24	20	6	7	39
V7	29	-34	13	-32	21	100	86	-45	61	-6	43	70	50
V8	24	-41	12	-35	20	86	100	-54	65	-17	54	79	49
V9	-16	29	19	36	-26	-45	-54	100	-37	14	-26	-50	-31
V10	14	-22	1	-20	24	61	65	-37	100	-3	30	52	33
V11	55	25	26	2	20	-6	-17	14	-3	100	-52	-43	32
V12	-7	-56	-7	-27	6	43	54	-26	30	-52	100	57	24
V13	7	-37	0	-28	7	70	79	-50	52	-43	57	100	31
V14	64	-19	22	-44	39	50	49	-31	33	32	24	31	100
V2													0.81
V3													1.12
V4													0.27
V5													3.34
V6													14.28
V7													0.63
V8													1
V9													0.12
V10													0.57
V11													2.32
V12													0.23
V13													0.71
V14													314.91

Importance of components:

	PC1	PC2	PC3	PC4	PC5	PC6	PC7
Standard deviation	2.169	1.5802	1.2025	0.95863	0.92370	0.80103	0.74231
Proportion of Variance	0.362	0.1921	0.1112	0.07069	0.06563	0.04936	0.04239
Cumulative Proportion	0.362	0.5541	0.6653	0.73599	0.80162	0.85098	0.89337
	PC8	PC9	PC10	PC11	PC12	PC13	
Standard deviation	0.59034	0.53748	0.5009	0.47517	0.41082	0.32152	
Proportion of Variance	0.02681	0.02222	0.0193	0.01737	0.01298	0.00795	
Cumulative Proportion	0.92018	0.94240	0.9617	0.97907	0.99205	1.00000	

1. 2.16929717950087
2. 1.58018155077547
3. 1.2025273259733
4. 0.958631276222941
5. 0.923703512147875
6. 0.801034975203288
7. 0.74231281272859
8. 0.590336652503682
9. 0.537475527463961

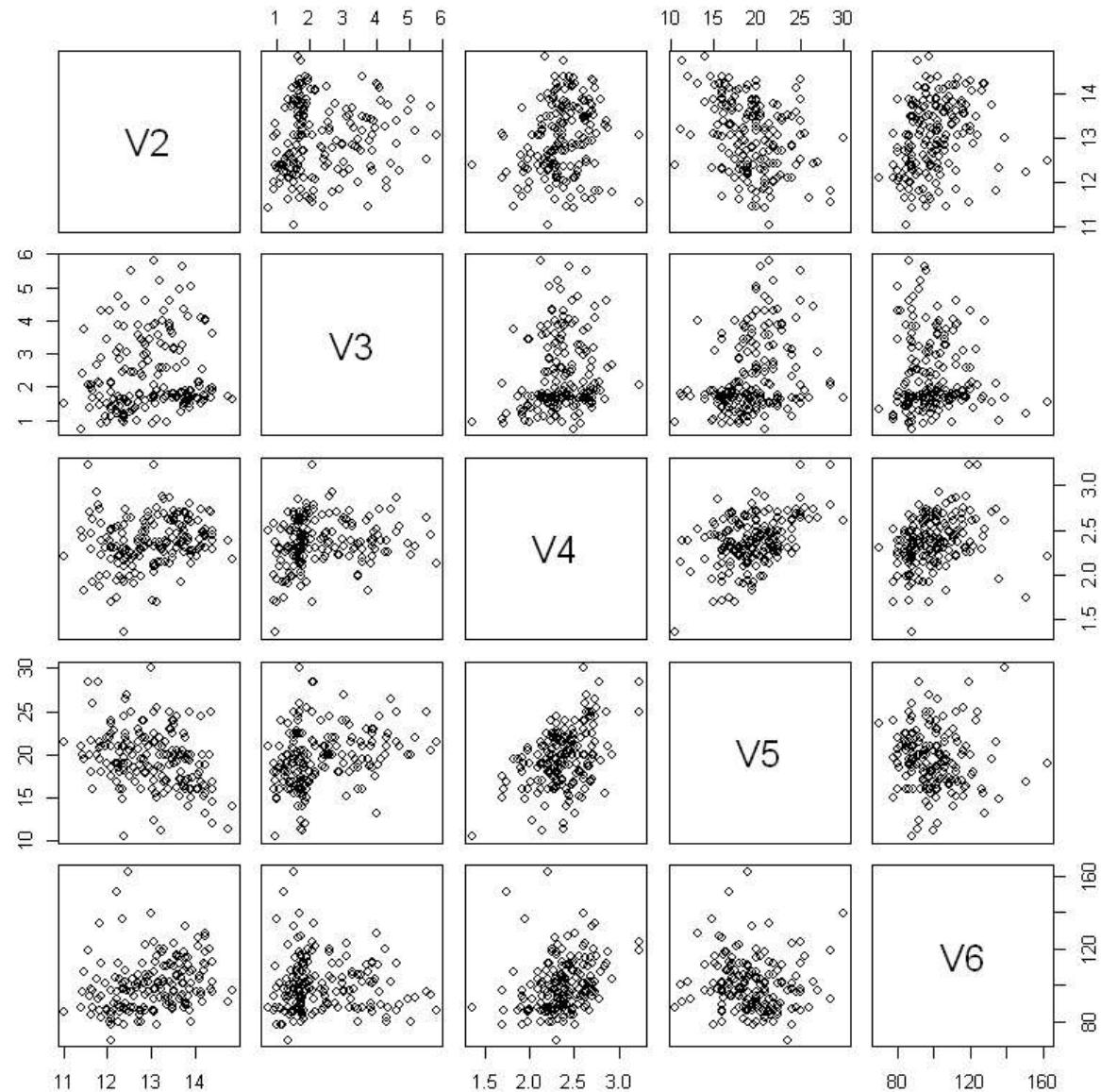
10. 0.500901669205374

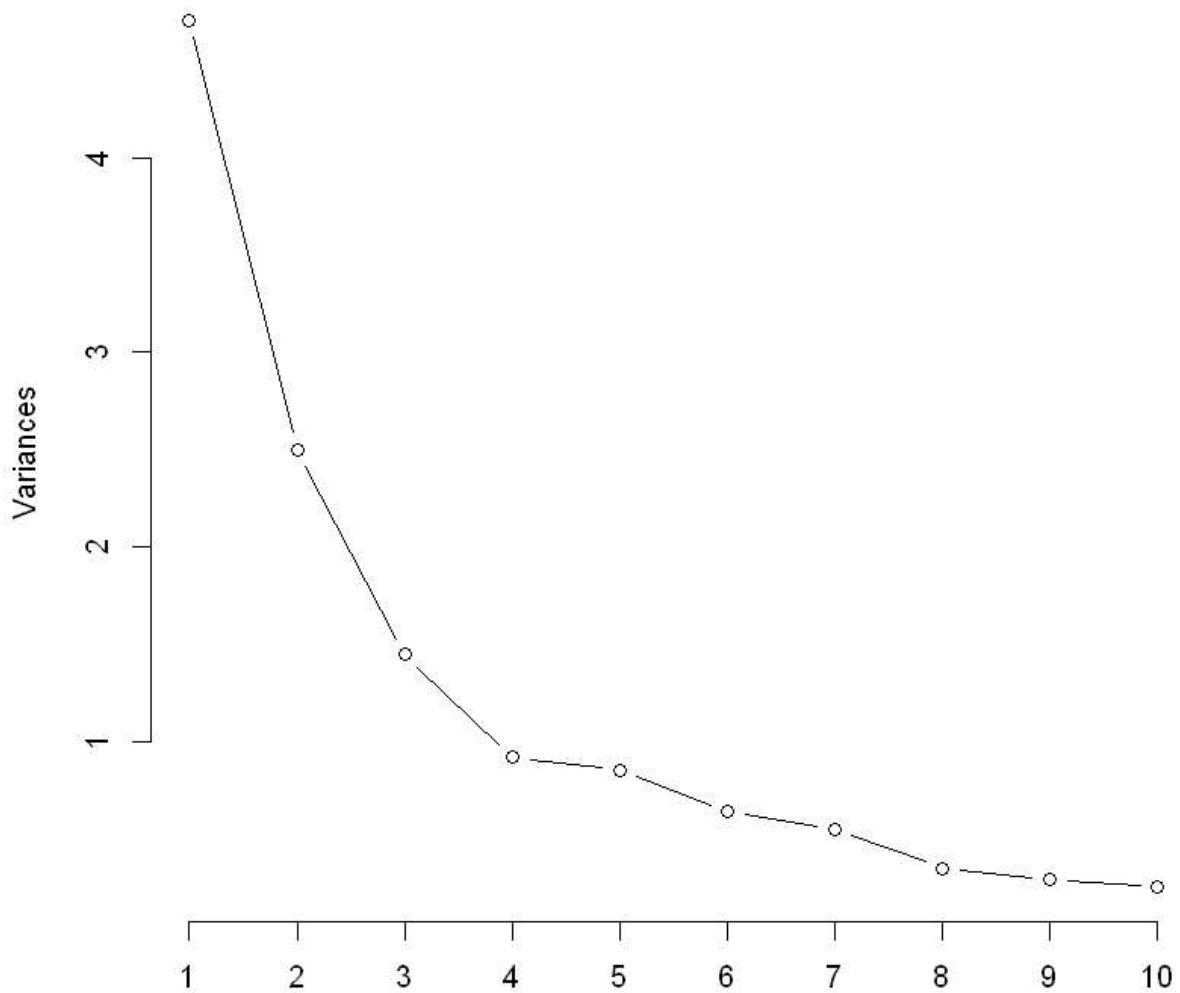
11. 0.475172221093246

12. 0.410816546439585

13. 0.321524393611011

13



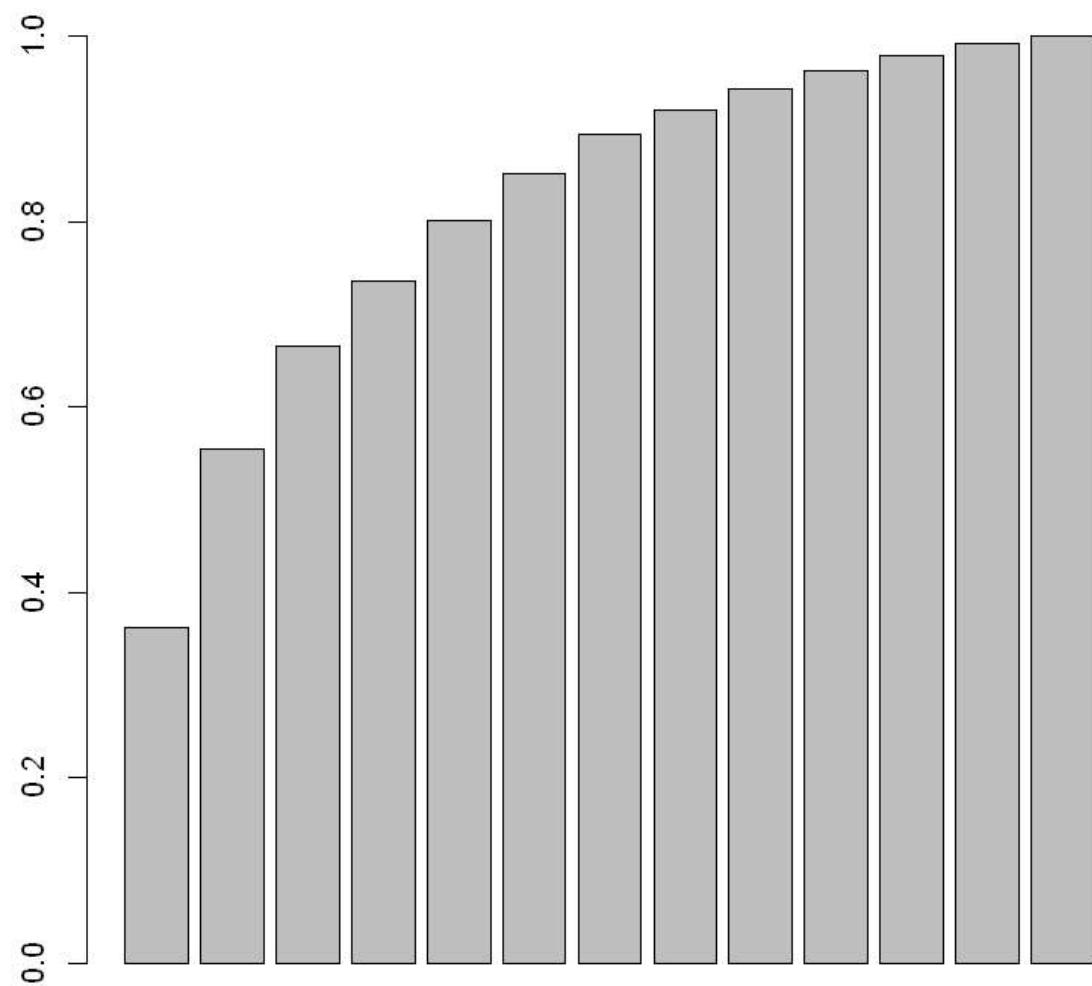
wine.pca

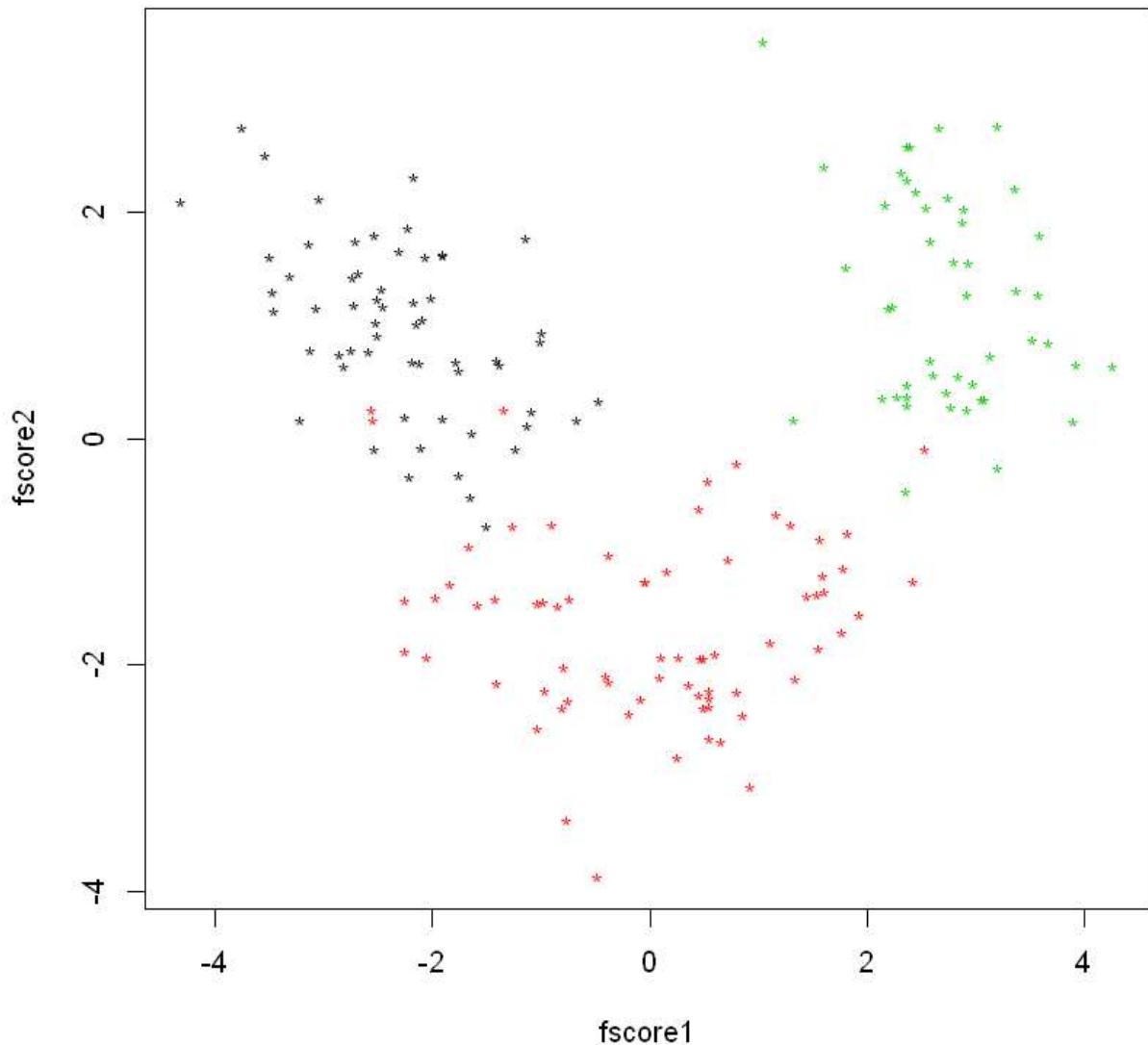
1.13

2.13

V2	-0.144329395406012
V3	0.245187580257221
V4	0.00205106144437095
V5	0.239320405487535
V6	-0.141992041952987
V7	-0.39466084506663
V8	-0.422934296710059
V9	0.298533102954715
V10	-0.313429488307689
V11	0.0886167047247227
V12	-0.296714563586381
V13	-0.376167410738713

V14	-0.286752226896805
V2	0.483651547817214
V3	0.224930934627845
V4	0.316068814025315
V5	-0.010590502288191
V6	0.299634003237862
V7	0.0650395118192793
V8	-0.00335981210030797
V9	0.0287794881129868
V10	0.0393017222897326
V11	0.529995672070044
V12	-0.279235147924282
V13	-0.164496192835785
V14	0.364902831798082





Seja $X_i = (X_{i1}, \dots, X_{i13})$ a linha i da matriz wine. Seja $Z_i = (Z_{i1}, \dots, Z_{i13})$ a linha i da matriz wine PADRONIZADA. Isto é, $Z_{ij} = (X_{ij} - \bar{x}_j) / s_j$ onde \bar{x}_i é a média aritmética e s_j é o desvio-padrão da coluna j da matriz wine. Esta matriz padronizada é obtida com o comando

```
In [4]: z = scale(wine[2:14])
round(apply(z, 2, mean), 5)
round(apply(z, 2, sd), 5)
```

V2	0
V3	0
V4	0
V5	0
V6	0
V7	0
V8	0
V9	0

V10	0
V11	0
V12	0
V13	0
V14	0
V2	1
V3	1
V4	1
V5	1
V6	1
V7	1
V8	1
V9	1
V10	1
V11	1
V12	1
V13	1
V14	1

Vamos considerar Z_i como um vetor-coluna 13-dimensional. Ao inves de usarmos o vetor Z_i , estamos usando apenas o vetor Y_i composto pelos dois indices formados pelos dois primeiros

$$\mathbf{Y}_i = \begin{bmatrix} Y_{i1} \\ Y_{i2} \end{bmatrix} = \begin{bmatrix} \mathbf{v}'_1 & \mathbf{Z}_i \\ \mathbf{v}'_2 & \mathbf{Z}_i \end{bmatrix}$$

componentes principais:

onde $v1$ e $v2$ sao os dois primeiros autovetores da matriz de correlacao de X .

- Preencha os locais com (??) com os valores numéricicos corretos (duas casa decimais apenas):

$$\begin{aligned} Y_{i1} &= (??)Z_{i1} + (??)Z_{i2} + (??)Z_{i3} + \dots + (??)Z_{i,13} \\ Y_{i2} &= (??)Z_{i1} + (??)Z_{i2} + (??)Z_{i3} + \dots + (??)Z_{i,13} \end{aligned}$$

- O último gráfico do acript R acima é um plot dos pontos \mathbf{Y}_i dos 178 vinhos. Identifique três regiões do plano Y_1, Y_2 que podem ser usadas para classificar futuras amostras de vinhos em uma das três categorias. Pode apenas esboçar grosseiramente no gráfico a mão livre.
- Suponha que uma nova amostra de vinho tem sua composição química medida e encontra-se

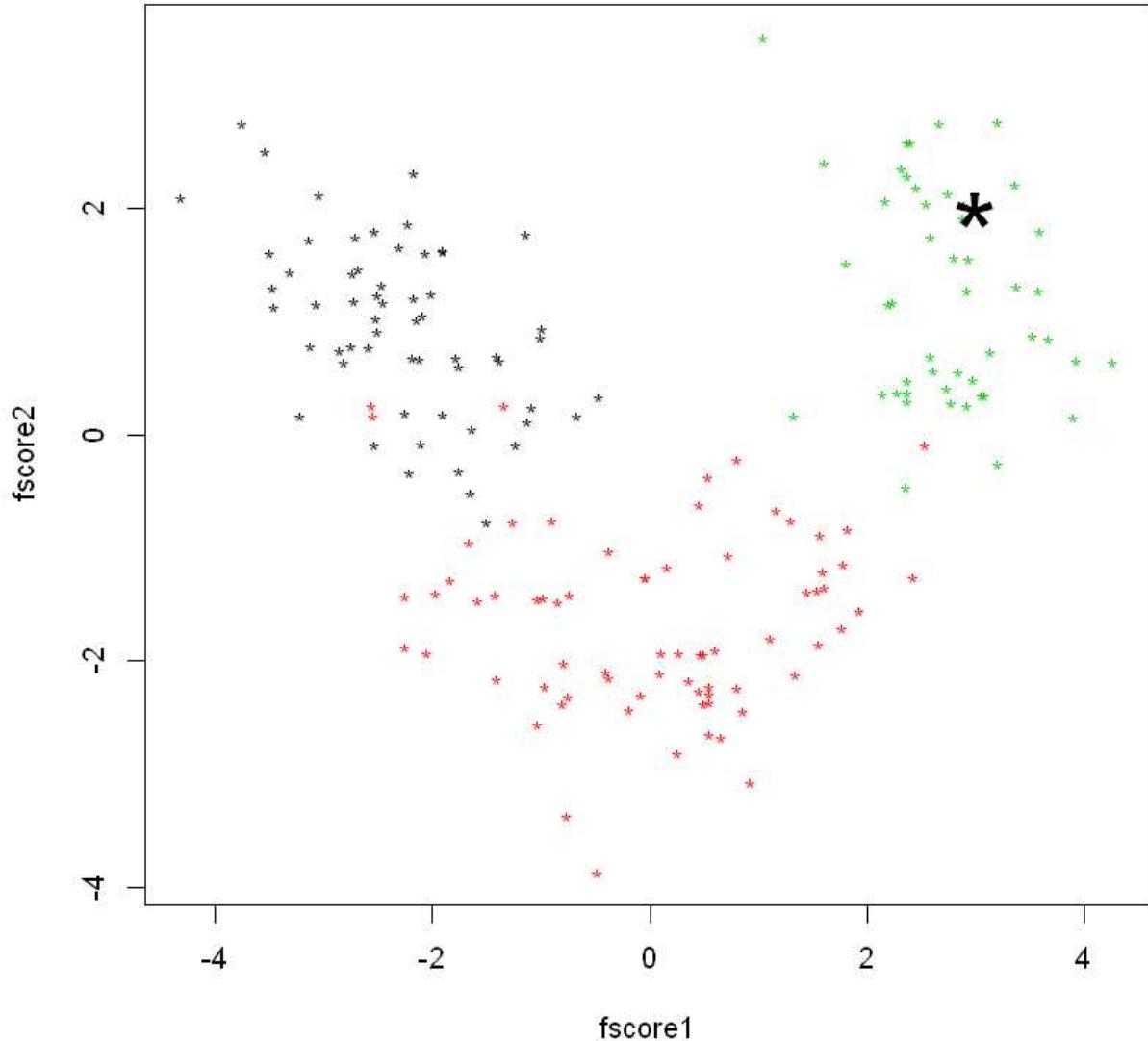
$$\mathbf{x} = (13.95, 3.65, 2.25, 18.4, 90.18, 1.55, 0.48, 0.5, 1.34, 10.2, 0.71, 1.48, 587.14)$$

Obtenha seu vetor \mathbf{z} , as suas coordenadas (y_1, y_2) e prediga o seu tipo. Confira sua resposta no final desta lista.

Solução:

In [5]:

```
x = c(13.95, 3.65, 2.25, 18.4, 90.18, 1.55, 0.48, 0.5, 1.34, 10.2, 0.71, 1.48, 587.14)
z = (x - apply(wine[,2:14], 2, mean))/apply(wine[,2:14], 2, sd)
y1 = sum( wine.pca$rot[,1] * z)
y2 = sum( wine.pca$rot[,2] * z)
plot(fscore1, fscore2, pch="*", col=wine[,1]+8)
points(y1, y2, pch="*", cex=4)
```



In []:

In []: