Exemplos do livro de Johnson e Wichern-Versão 1

Djalma Galvão Carneiro Pessoa (Consultor)- IBGE

16 de janeiro de 2008

Sumário

Introdução

Nessas notas reunimos o material utilizado em disciplinas de Análise Multivariada, que ministramos no programa de Mestrado da ENCE. O livro-texto adotado foi: `Johnson, R.A e Wichern, D.W., Applied Multivariate Statistical Analysis, 2002, 5ª edição`. Foram cobertos os Capítulos 1-9, 11 e 12 do livro.

Utilizamos nas aulas o sistema `R` de análise de dados. Para cada capítulo do livro, além da apresentarmos a teoria do assunto, implementamos, através do `R`, os exemplos contidos no livro-texto.

Reproduzimos aqui os *scripts* do `R` utilizados nas aulas, acompanhados de alguns comentários. Detalhes dos exemplos, imprescindíveis para o entendimento dos comandos, são apresentados nos exemplos correspondentes do livro-texto. Todos os dados utilizados estão contidos no CD que acompanha o livro.

Não tivemos a pretensão de esgotar os recursos, praticamente ilimitados, do `R` nas técnicas multivariadas, principalmente na parte de gráficos. Há vários recursos do `R` que não foram utilizados e que forneceriam outras análises interessantes. Optamos, apenas, por implementar técnicas descritas no livro-texto.

O objetivo foi tornar disponível material que talvez possa ser útil no aprendizado de Análise Multivariada. Agradecemos sugestões de melhorias.

Djalma G. C. Pessoa

Capítulo 1

Dados Multivariados

1.1 Exemplo 1.1

Uma matriz de dados.

```
> X1 <- c(42, 52, 48, 58)
> X2 <- c(4, 5, 4, 3)
> X <- cbind(X1, X2)
> X[1, 1]

[1] 42

> X[2, 1]

[1] 52

> X[3, 1]

[1] 48

> X[4, 1]

[1] 58

> X[1, 2]
```

```
[1] 4
```

```
> X[2, 2]
```

```
[1] 5
```

```
> X[3, 2]
```

```
[1] 4
```

```
> X[4, 2]
```

```
[1] 3
```

```
> X
```

```
     X1 X2
[1,] 42  4
[2,] 52  5
[3,] 48  4
[4,] 58  3
```

1.2 Exemplo 1.2

AS matrizes $\overline{\mathbf{x}}$, $\mathbf{S}_n$ e $\mathbf{R}$.

```
> n <- nrow(X)
> xbar <- colMeans(X)
> Sn <- round(cov(X) * (n - 1)/n, 1)
> R <- round(cor(X), 2)
```

Técnicas Gráficas

```
> library(graphics)
> X1 <- c(3, 4, 2, 6, 8, 2, 5)
> X2 <- c(5, 5.5, 4, 7, 10, 5, 7.5)
> X <- data.frame(X1 = X1, X2 = X2)
> nf <- layout(matrix(c(3, 1, 0, 2), 2, 2, byrow = TRUE),
```

```
+     c(1, 3), c(3, 1), TRUE)
> xrange <- with(X, c(min(X1), max(X1)))
> yrange <- with(X, c(min(X2), max(X2)))
> with(X, plot(X1, X2, xlim = xrange, ylim = yrange,
+     xlab = "", ylab = ""))
> par(mar = c(1, 3, 1, 1))
> stripchart(X$X1, method = "stack", offset = 1/2,
+     pch = 16)
> par(mar = c(3, 0, 1, 1))
> stripchart(X$X2, method = "stack", vertical = TRUE,
+     offset = 1/2, pch = 18)
```

Dados emparelhados diferentemente:

```
> X1 <- c(5, 4, 6, 2, 2, 8, 3)
> X2 <- c(5, 5.5, 4, 7, 10, 5, 7.5)
> X <- data.frame(X1 = X1, X2 = X2)
> nf <- layout(matrix(c(3, 1, 0, 2), 2, 2, byrow = TRUE),
+     c(1, 3), c(3, 1), TRUE)
> xrange <- with(X, c(min(X1), max(X1)))
> yrange <- with(X, c(min(X2), max(X2)))
> par(mar = c(3, 3, 1, 1))
> with(X, plot(X1, X2, xlim = xrange, ylim = yrange,
+     xlab = "", ylab = ""))
> par(mar = c(1, 3, 1, 1))
> stripchart(X$X1, method = "stack", offset = 1/2,
+     pch = 16)
> par(mar = c(3, 0, 1, 1))
> stripchart(X$X2, method = "stack", vertical = TRUE,
+     offset = 1/2, pch = 18)
```

1.3 Exemplo 1.3

Dados não disponíveis.

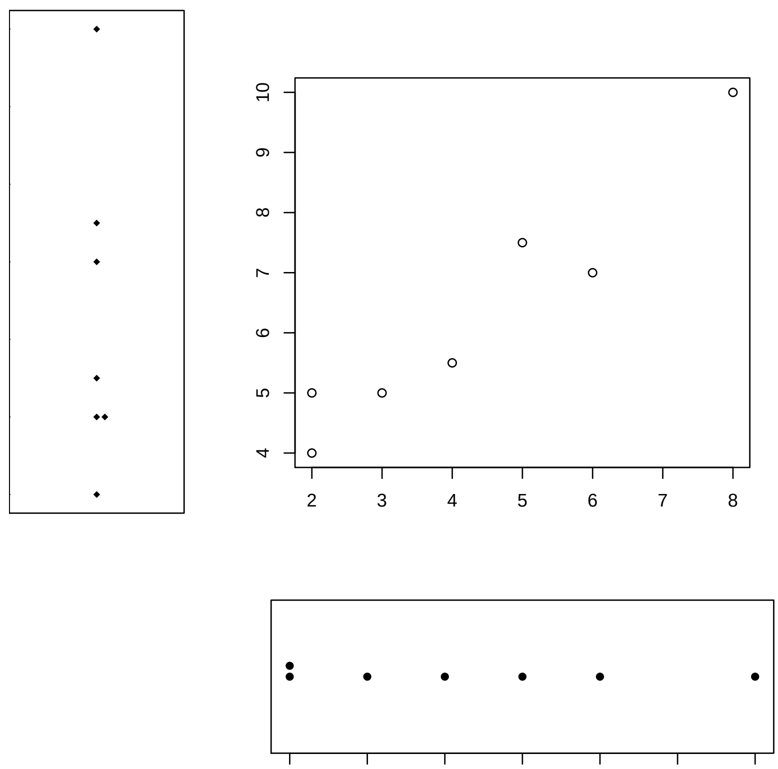


Figura 1.1: Diagrama de dispersão e diagramas de pontos marginais

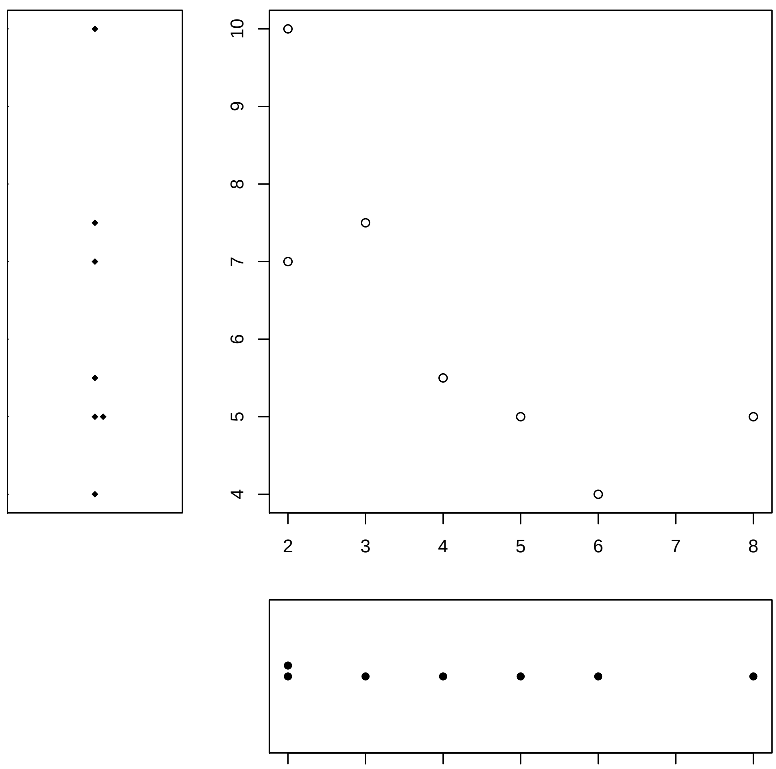


Figura 1.2: Diagrama de dispersão e diagramas de pontos marginais

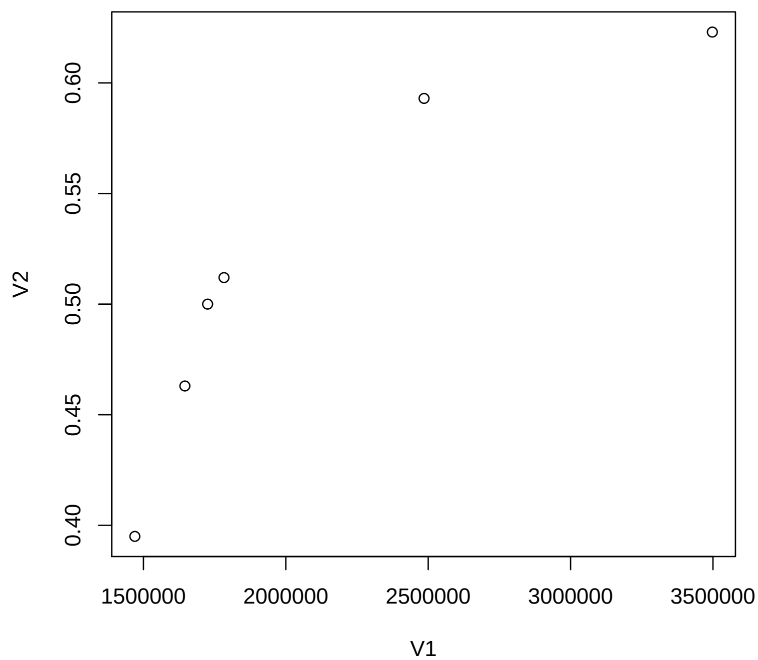


Figura 1.3: Salários vs % de "ganhos - perdas"da Tabela 1.1

1.4 Exemplo 1.4

Um diagrama de dispersão de dados de baseball.

V1: folha de pagamento do clube; V2: % ganho - % perda.

```
> Exemplo1.4 <- read.table("T1-1.DAT")
> with(Exemplo1.4, plot(V1, V2))
```

1.5 Exemplo 1.5

Diagrama de dispersão múltiplo para medidas de resistência de papel.

V1: densidade;
V2: resistência na direção da máquina;
V3: resistência na direção tranversal.
Colocar no gráfico pairs o histograma de cada variável na diagonal.

```
> Exemplo1.5 <- read.table("T1-2.DAT")
> panel.hist <- function(x, ...) {
+     usr <- par("usr")
+     on.exit(par(usr))
+     par(usr = c(usr[1:2], 0, 1.5))
+     h <- hist(x, plot = FALSE)
+     breaks <- h$breaks
+     nB <- length(breaks)
+     y <- h$counts
+     y <- y/max(y)
+     rect(breaks[-nB], 0, breaks[-1], y, col = "cyan",
+         ...)
+ }
> pairs(Exemplo1.5, panel = panel.smooth, cex = 1.5,
+     pch = 24, bg = "light blue", diag.panel = panel.hist,
+     cex.labels = 2, font.labels = 2)
> rownames(Exemplo1.5)[Exemplo1.5$V1 == max(Exemplo1.5$V1)]
```

```
[1] "25"
```

1.6 Exemplo 1.6 e 1.7

Busca de estrutura em dimensão mais baixa. Estrutura de grupo em três dimensões.

```
> Exemplo1.6 <- read.table("T1-3.DAT", col.names = c("Mass",
+     "SVL", "HLS"))
```

```
[1] "25"
```

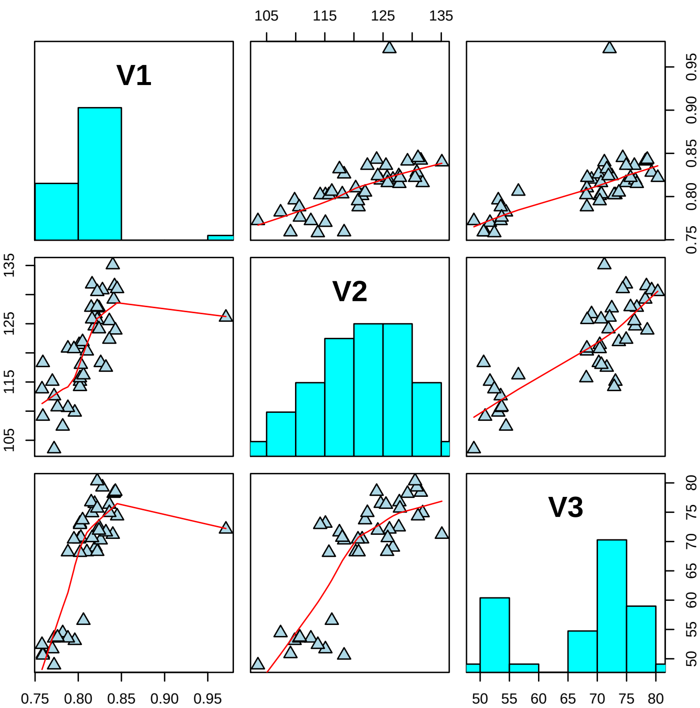


Figura 1.4: Gráficos de dispersão e suavização com histogramas de cada variável

```
> library(scatterplot3d)
> Exemplo1.6$sexo <- c("f", "m", "f", "f", "m", "f",
+     "m", "f", "m", "f", "m", "f", "m", "m", "m",
+     "m", "f", "m", "m", "m", "f", "f", "m", "f",
+     "f")
> cores <- with(Exemplo1.6, ifelse(sexo == "f", "red",
+     "blue"))
> scatterplot3d(Exemplo1.6[, 1:3], color = cores)
```

Figura 1.8

```
> with(Exemplo1.6, scatterplot3d(SVL, HLS, Mass, color = cores,
+     pch = 16, bty = "n"))
> legend("topleft", c("f", "m"), col = c("red", "blue"),
+     pch = 16, bty = "n")
```

1.7 Exemplo 1.10

Curvas de crescimento

```
> Exemplo1.10 <- read.table("T1-4.DAT")
> Exemplo1.10$urso <- 1:7
> Peso <- as.matrix(Exemplo1.10[, 1:4])
> library(stats)
> Peso.ts <- ts(t(Peso), start = 2, end = 5, names = c("U1",
+     "U2", "U3", "U4", "U5", "U6", "U7"))
> ts.plot(Peso.ts, gpars = list(ylab = "peso", lty = c(1:7)))
> legend("topleft", c("U1", "U2", "U3", "U4", "U5",
+     "U6", "U7"), lty = 1:7, bty = "n")
```

```
> Altura <- as.matrix(Exemplo1.10[, 5:8])
> Altura.ts <- ts(t(Altura), start = 2, end = 5, names = c("U1",
+     "U2", "U3", "U4", "U5", "U6", "U7"))
> ts.plot(Altura.ts, gpars = list(ylab = "altura",
+     lty = c(1:7)))
```

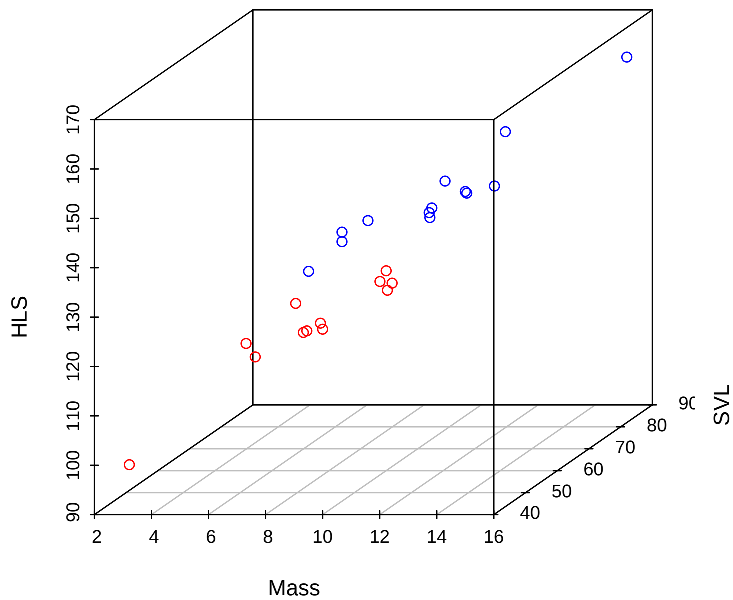


Figura 1.5: Gráfico de dispersão em três dimensões dos dados de crocodilhos fêmeas

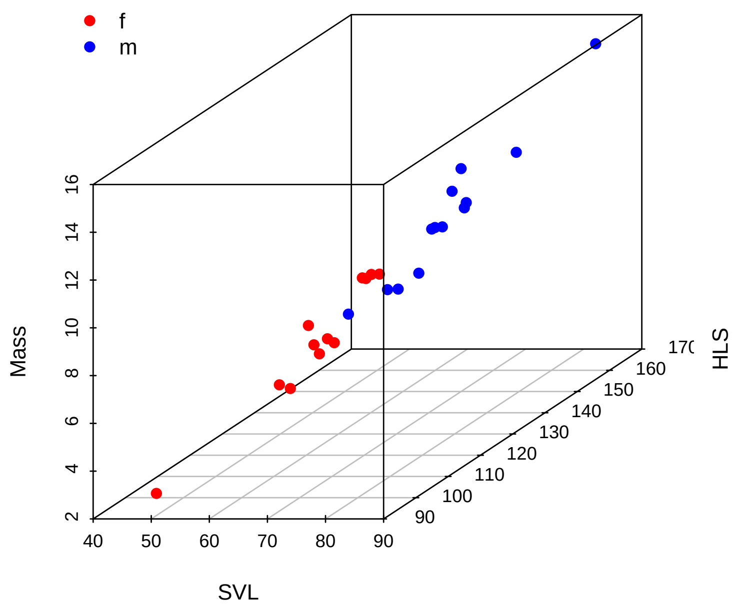


Figura 1.6: Gráficos de dispersão tridimensional dos dados de crocodilhos machos e fêmeas

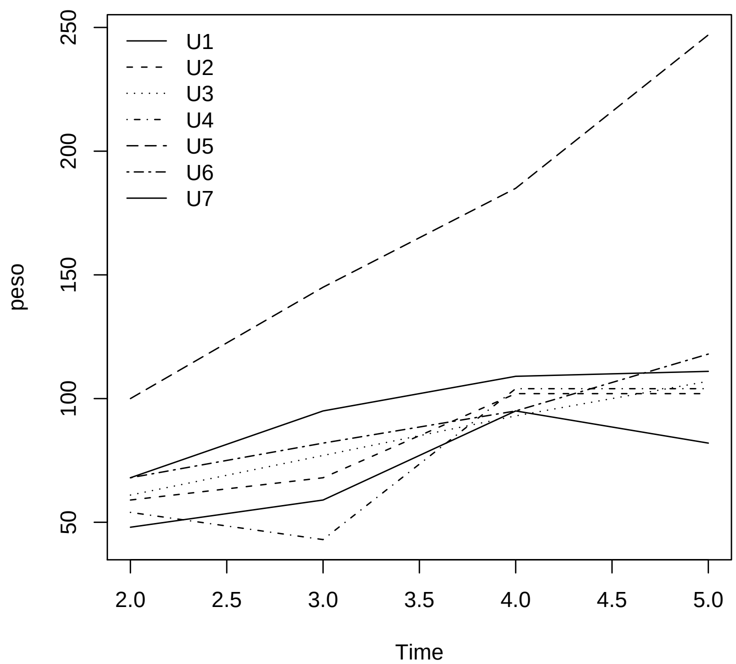


Figura 1.7: Curvas combinadas de crescimento de sete ursos fêmeas

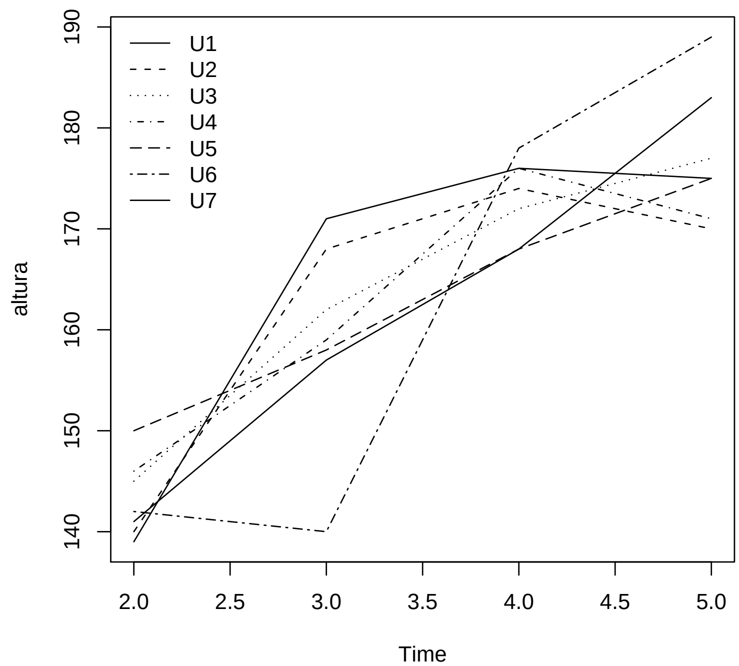


Figura 1.8: Gráficos individuais de crescimento de peso de sete ursos fêmeas

```
> legend("topleft", c("U1", "U2", "U3", "U4", "U5",
+     "U6", "U7"), lty = 1:7, bty = "n")
```

```
> plot(Altura.ts)
```

Outra forma de fazer o gráfico anterior:

```
> tab14 <- read.table("t1-4.dat")
> dim(tab14)
```

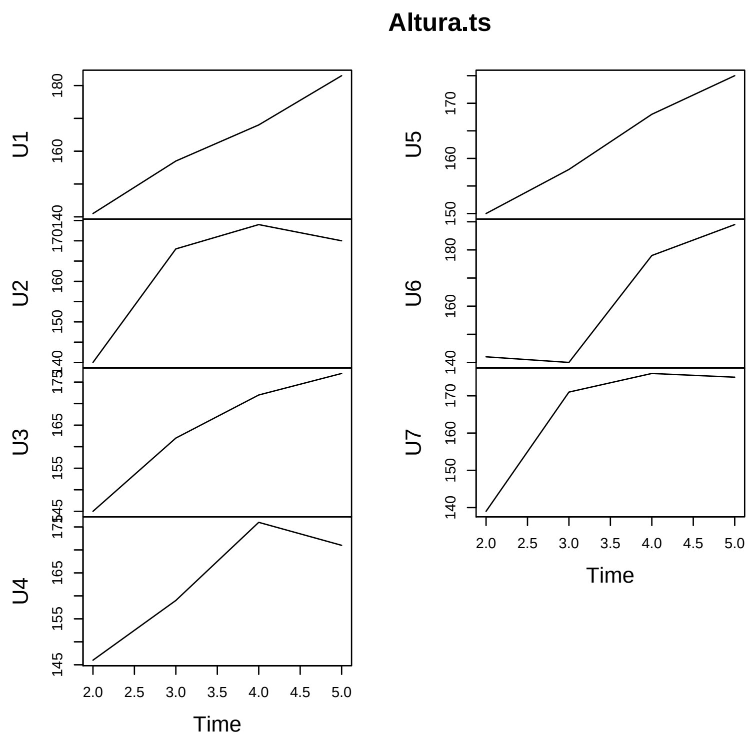


Figura 1.9: Gráficos individuais de crescimento de peso de sete ursos fêmeas

```
[1] 7 8

> names(tab14)

[1] "V1" "V2" "V3" "V4" "V5" "V6" "V7" "V8"

> names(tab14) <- c("W2", "W3", "W4", "W5", "L2", "L3",
+     "L4", "L5")
> Ursos.dat <- expand.grid(1:7, 2:5)
> Ursos.dat <- as.data.frame(Ursos.dat)
> names(Ursos.dat) <- c("Ursa", "Ano")
> Ursos.dat$Peso <- unlist(tab14[, 1:4])
> Ursos.dat$Comp <- unlist(tab14[, 5:8])
> with(Ursos.dat, interaction.plot(Ano, Ursa, Peso,
+     fun = mean))

> coplot(Peso ~ Ano | Ursa, data = Ursos.dat, panel = lines)

> coplot(Comp ~ Ano | Ursa, data = Ursos.dat, panel = lines)
```

1.8 Exemplo 1.14

Cálculo da distância estatística

```
> O <- c(0, 0)
> s11 <- 4
> s22 <- 1
> d2.OP <- expression(x1^2/s11 + x2^2/s22)
> x1 <- 0
> x2 <- 1
> eval(d2.OP)

[1] 1
```

```
[1] 7 8

[1] "V1" "V2" "V3" "V4" "V5" "V6" "V7" "V8"
```

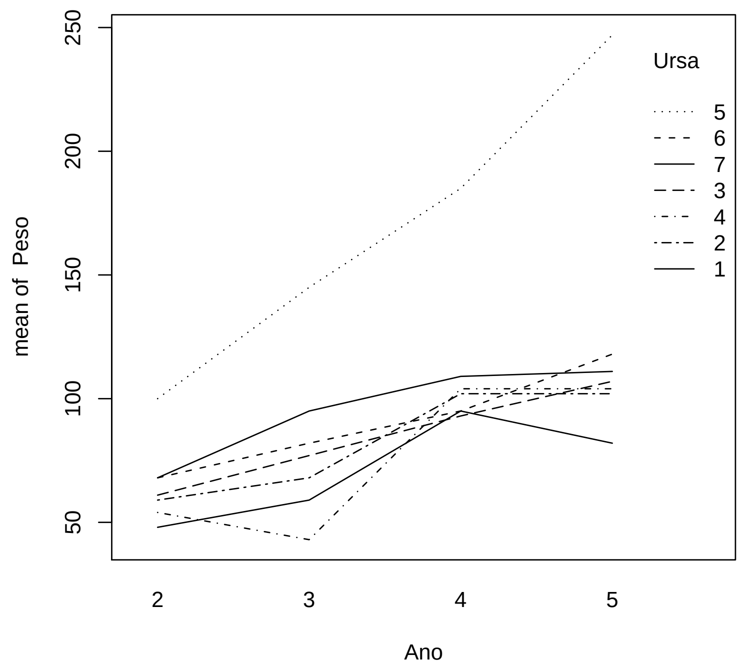


Figura 1.10: Gráficos individuais de crescimento de peso de sete ursos fêmeas

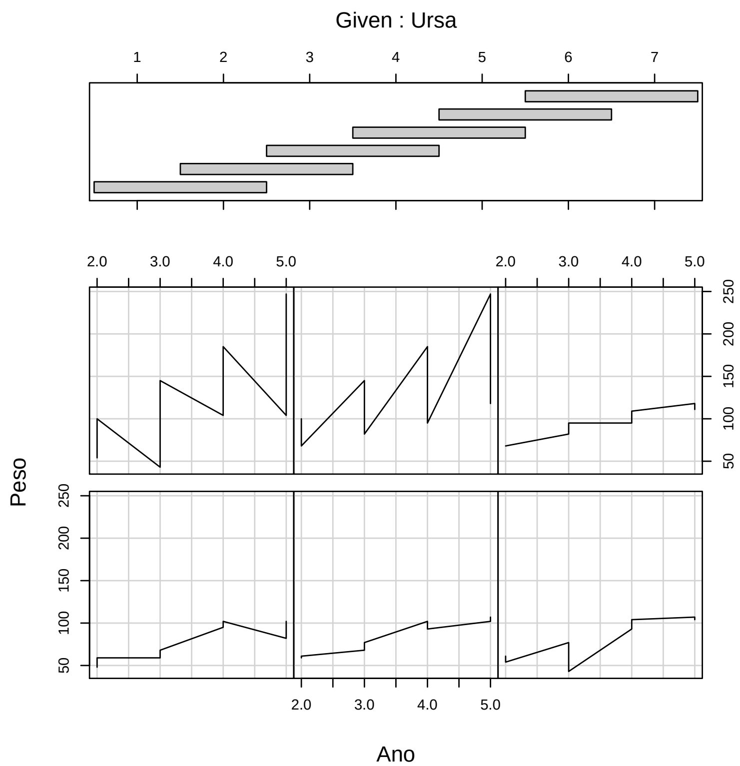


Figura 1.11: Gráficos individuais de crescimento de peso de 7 ursos fêmeas

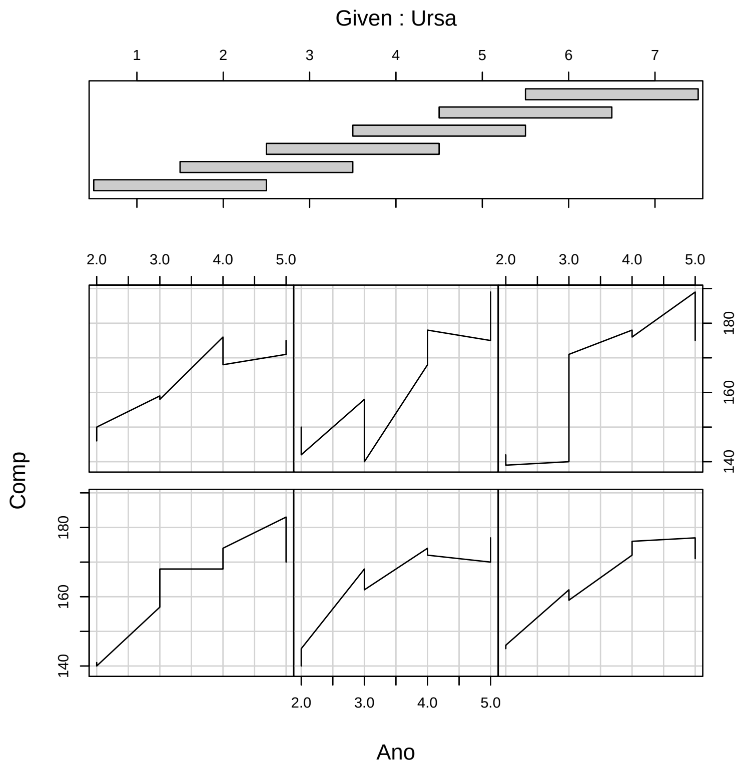


Figura 1.12: Gráficos individuais de crescimento de peso de 7 ursos fêmeas

```
> x1 <- 0
> x2 <- -1
> eval(d2.OP)
```

```
[1] 1
```

```
> x1 <- 2
> x2 <- 0
> eval(d2.OP)
```

```
[1] 1
```

```
> x1 <- 1
> x2 <- sqrt(3)/2
> eval(d2.OP)
```

```
[1] 1
```

1.9 Figura 1.22

Gráfico da elipse: $a_{11}x_1^2 + 2a_{12}x_1x_2 + a_{22}x_2^2 = r^2$

```
> w <- seq(0, 2 * pi, length = 1000)
> a11 <- 1/4
> a22 <- 1/1
> a12 <- 0
> r <- sqrt(1/(a11 * cos(w)^2 + 2 * a12 * sin(w) *
+     cos(w) + a22 * sin(w)^2))
> pontos <- cbind(r * cos(w), r * sin(w))
> plot(pontos, type = "l")
> abline(h = 0, v = 0)
> points(0, 1)
> points(0, -1)
> points(2, 0)
> points(1, sqrt(3)/2)
```

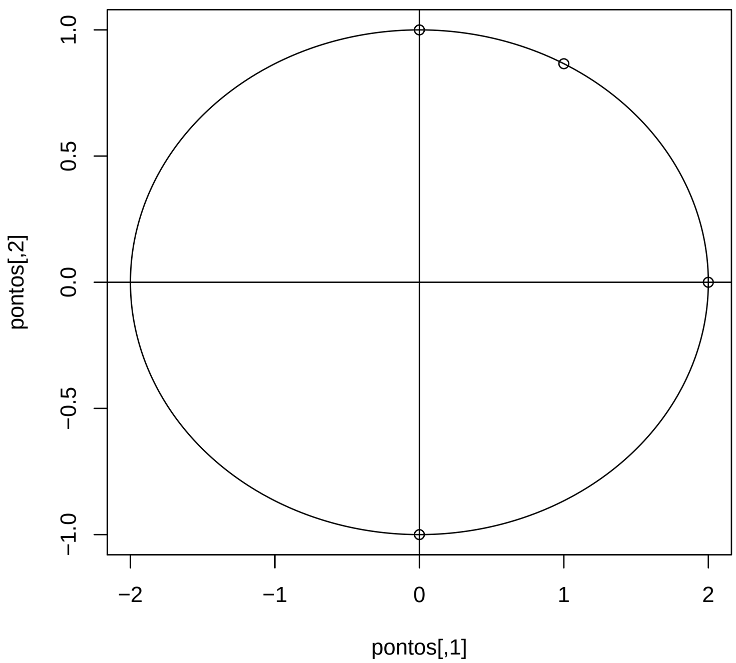


Figura 1.13: Gráfico de elipse

Capítulo 2

Vetores aletórios

2.1 Exemplo 2.1

Cálculo de comprimento de vetores e do ângulo entre eles.

```
> x <- matrix(c(1, 3, 2), ncol = 1)
> y <- matrix(c(-2, 1, -1), ncol = 1)
> 3 * x

     [,1]
[1,]    3
[2,]    9
[3,]    6

> x + y

     [,1]
[1,]   -1
[2,]    4
[3,]    1

> t(x) %*% x

     [,1]
[1,]   14
```

```
> t(y) %*% y

     [,1]
[1,]    6

> t(x) %*% y

     [,1]
[1,]   -1

> Lx <- sqrt(t(x) %*% x)
> Ly <- sqrt(t(y) %*% y)
> cos.teta <- t(x) %*% y/(Lx * Ly)
> round(cos.teta, 3)

       [,1]
[1,] -0.109

> round(acos(cos.teta) * 180/pi, 1)

     [,1]
[1,] 96.3

> L3x <- sqrt(t(3 * x) %*% (3 * x))
> all.equal(L3x, 3 * Lx)

[1] TRUE
```

2.2 Exemplo 2.2

Identificação de vetores linearmente independentes

```
> x1 <- matrix(c(1, 2, 1), ncol = 1)
> x2 <- matrix(c(1, 0, -1), ncol = 1)
> x3 <- matrix(c(1, -2, 1), ncol = 1)
> solve(cbind(x1, x2, x3), c(0, 0, 0))

[1] 0 0 0
```

x_1,x_2 e x_3 são linearmente independentes.

2.3 Exemplo 2.3

A transposta de uma matriz

```
> A <- matrix(c(3, 1, -1, 5, 2, 4), 2, 3)
> t(A)
```

```
     [,1] [,2]
[1,]    3    1
[2,]   -1    5
[3,]    2    4
```

2.4 Exemplo 2.4

Soma de duas matrizes e a multiplicação de uma matriz por uma constante.

```
> A <- matrix(c(0, 1, 3, -1, 1, 1), 2, 3)
> B <- matrix(c(1, 2, -2, 5, -3, 1), 2, 3)
> 4 * A
```

```
     [,1] [,2] [,3]
[1,]    0   12    4
[2,]    4   -4    4
```

```
> A + B
```

```
     [,1] [,2] [,3]
[1,]    1    1   -2
[2,]    3    4    2
```

2.5 Exemplo 2.5

Multiplicação de matrizes.

```
> A <- matrix(c(3, 1, -1, 5, 2, 4), 2, 3)
> B <- matrix(c(-2, 7, 9), ncol = 1)
> C1 <- matrix(c(2, 1, 0, -1), 2, 2)
> A %*% B
```

```
     [,1]
[1,]    5
[2,]   69
```

```
> C1 %*% A
```

```
     [,1] [,2] [,3]
[1,]    6   -2    4
[2,]    2   -6   -2
```

2.6 Exemplo 2.6

Alguns produtos típicos e suas dimensões.

```
> A <- matrix(c(1, 2, -2, 4, 3, -1), 2, 3)
> b <- matrix(c(7, -3, 6), ncol = 1)
> c1 <- matrix(c(5, 8, -4), ncol = 1)
> d <- matrix(c(2, 9), ncol = 1)
> A %*% b
```

```
     [,1]
[1,]   31
[2,]   -4
```

```
> t(b) %*% c1
```

```
     [,1]
[1,]  -13
```

```
> b %*% t(c1)
```

```
     [,1] [,2] [,3]
[1,]   35   56  -28
[2,]  -15  -24   12
[3,]   30   48  -24
```

```
> t(d) %*% A %*% b
```

```
     [,1]
[1,]   26
```

2.7 Exemplo 2.7

Uma matriz simétrica.

```
> A <- matrix(c(3, 5, 5, -2), nrow = 2)
> all.equal(A, t(A))

[1] TRUE
```

2.8 Exemplo 2.8

A existência de uma matriz inversa.

```
> A <- matrix(c(3, 4, 2, 1), nrow = 2)
> B <- matrix(c(-0.2, 0.8, 0.4, -0.6), nrow = 2)
> round(B %*% A, 4)

     [,1] [,2]
[1,]    1    0
[2,]    0    1

> solve(A, c(0, 0))

[1] 0 0
```

2.9 Exemplo 2.9

Verificação de autovalores e autovetores.

```
> A <- matrix(c(1, -5, -5, 1), nrow = 2)
> e <- matrix(c(1/sqrt(2), -1/sqrt(2)), ncol = 1)
> lambda <- 6
> all.equal(A %*% e, lambda * e)

[1] TRUE
```

2.10 Exemplo 2.10

A decomposição espectral de uma matriz.

```
> A <- matrix(c(13, -4, 2, -4, 13, -2, 2, -2, 10),
+     nrow = 3)
> A.eigen <- eigen(A)
> lambda1 <- A.eigen$values[1]
> lambda2 <- A.eigen$values[2]
> lambda3 <- A.eigen$values[3]
> e1 <- A.eigen$vectors[, 1, drop = F]
> e2 <- A.eigen$vectors[, 2, drop = F]
> e3 <- A.eigen$vectors[, 3, drop = F]
> sum(e1^2)

[1] 1

> sum(e2^2)

[1] 1

> sum(e3^2)

[1] 1

> sum(e1 * e2)

[1] 1.11e-16

> sum(e1 * e3)

[1] 0

> sum(e2 * e3)

[1] 0
```

Verificação do teorema espectral:

```
> all.equal(A, lambda1 * e1 %*% t(e1) + lambda2 * e2 %*%
+     t(e2) + lambda3 * e3 %*% t(e3))

[1] TRUE
```

2.11 Exemplo 2.11

Uma matriz definida positiva e forma quadrática.

```
> A <- matrix(c(3, -sqrt(2), -sqrt(2), 2), 2, 2)
> A.eigen <- eigen(A)
> e1 <- A.eigen$vectors[, 1, drop = F]
> e2 <- A.eigen$vectors[, 2, drop = F]
> lambda1 <- A.eigen$values[1]
> lambda2 <- A.eigen$values[2]
> all.equal(A, lambda1 * e1 %*% t(e1) + lambda2 * e2 %*%
+     t(e2))

[1] TRUE

> lambda1 > 0

[1] TRUE

> lambda2 > 0

[1] TRUE
```

2.12 Exemplo 2.12

Cálculo de valores esperados de variáveis aleatórias discretas.

```
> x1 <- c(-1, 0, 1)
> p1.x1 <- c(0.3, 0.3, 0.4)
> EX1 <- sum(x1 * p1.x1)
> x2 <- c(0, 1)
> p2.x2 <- c(0.8, 0.2)
> EX2 <- sum(x2 * p2.x2)
> EX <- matrix(c(EX1, EX2), ncol = 1)
```

2.13 Exemplo 2.13

Cálculo da matriz de covariãncia.

```
> x1 <- c(-1, 0, 1)
> p1.x1 <- c(0.3, 0.3, 0.4)
> x2 <- c(0, 1)
> p2.x2 <- c(0.8, 0.2)
> EX1 <- sum(x1 * p1.x1)
> EX2 <- sum(x2 * p2.x2)
> sigma11 <- sum((x1 - EX1)^2 * p1.x1)
> sigma22 <- sum((x2 - EX2)^2 * p2.x2)
> p12.x1x2 <- matrix(c(0.24, 0.16, 0.4, 0.06, 0.14,
+     0), ncol = 2)
> sigma12 <- sum(outer(x1 - EX1, x2 - EX2) * p12.x1x2)
> sigma21 <- sum(outer(x2 - EX2, x1 - EX1) * t(p12.x1x2))
> mu <- matrix(c(EX1, EX2), ncol = 1)
> sigma <- matrix(c(sigma11, sigma12, sigma21, sigma22),
+     2, 2)
```

2.14 Exemplo 2.14

Cálculo da matriz de correlação a partir da matriz de covariãncia.

```
> Sigma <- matrix(c(4, 1, 2, 1, 9, -3, 2, -3, 25),
+     nrow = 3)
> V.meio <- diag(sqrt(diag(Sigma)))
> solve(V.meio)
```

```
     [,1]  [,2] [,3]
[1,]  0.5 0.000  0.0
[2,]  0.0 0.333  0.0
[3,]  0.0 0.000  0.2
```

```
> solve(V.meio) %*% Sigma %*% solve(V.meio)
```

```
      [,1]   [,2] [,3]
[1,] 1.000  0.167  0.2
[2,] 0.167  1.000 -0.2
[3,] 0.200 -0.200  1.0
```

Capítulo 3

Geometria Amostral

3.1 Exemplo 3.1

Cálculo do vetor de médias.

```
> X <- matrix(c(4, -1, 3, 1, 3, 5), 3, 2)
> xbar <- matrix(colMeans(X), ncol = 1)
```

```
> plot(X[, 1], X[, 2])
> points(xbar[1, 1], xbar[2, 1], pch = 16)
```

3.2 Exemplo 3.2

Os dados como p vetores em n dimensões.

```
> library(scatterplot3d)
> ex32 <- scatterplot3d(X[1, ], X[2, ], X[3, ], pch = 16,
+     xlim = c(0, 6), ylim = c(-2, 6), zlim = c(0,
+         6))
> ex32$points3d(c(0, X[1, 1]), c(0, X[2, 1]), c(0,
+     X[3, 1]), type = "l")
> ex32$points3d(c(0, X[1, 2]), c(0, X[2, 2]), c(0,
+     X[3, 2]), type = "l")
```

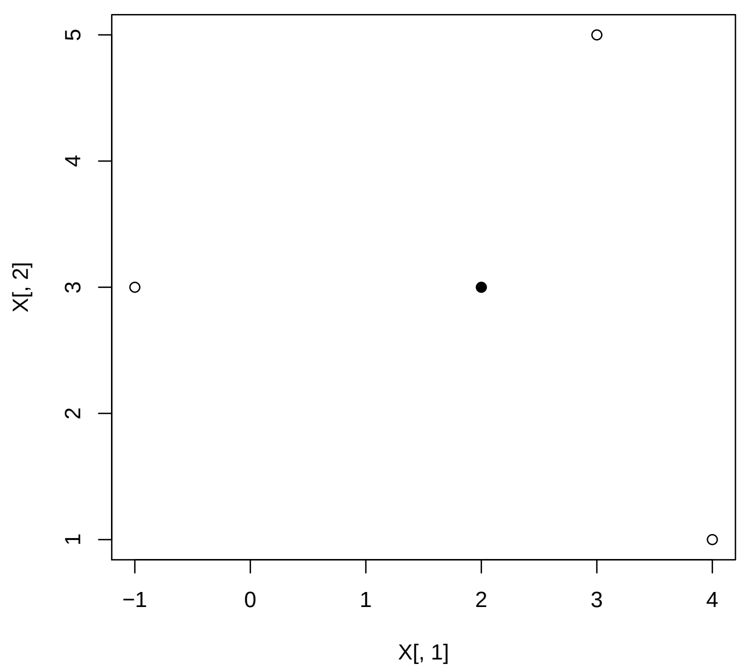


Figura 3.1: Um gráfico das linhas da matriz X com $n = 3$ e $p = 2$

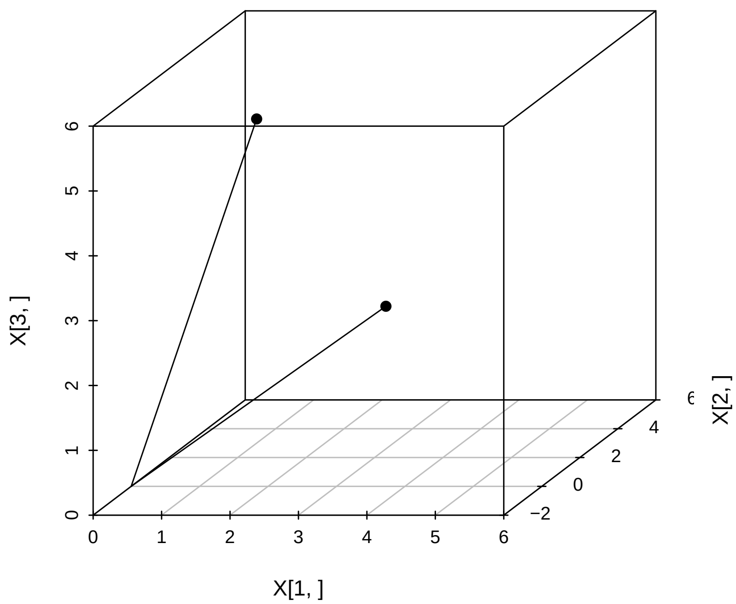


Figura 3.2: Um gráfico das colunas da matriz X com $n = 3$ e $p = 2$

3.3 Exemplo 3.3 e 3.4

Decomposição de um vetor nas suas componentes de média e de desvio. Cálculo de $\mathbf{S}_n$ e de $\mathbf{R}$ a partir do vetor de desvios.

```
> X <- matrix(c(4, -1, 3, 1, 3, 5), 3, 2)
> n <- 3
> y1 <- X[, 1, drop = FALSE]
> y2 <- X[, 2, drop = FALSE]
> x1bar <- colMeans(X)[1]
> x2bar <- colMeans(X)[2]
> med1 <- x1bar * matrix(rep(1, 3), ncol = 1)
> med2 <- x2bar * matrix(rep(1, 3), ncol = 1)
> d1 <- y1 - x1bar * matrix(rep(1, 3), ncol = 1)
> d2 <- y2 - x2bar * matrix(rep(1, 3), ncol = 1)
```

Ortogonalidade:

```
> t(med1) %*% d1

     [,1]
[1,]    0

> t(med2) %*% d2

     [,1]
[1,]    0
```

Decomposição:

```
> all.equal(y1, med1 + d1)

[1] TRUE

> all.equal(y2, med2 + d2)

[1] TRUE
```

Soma dos desvios quadráticos:

```
> t(d1) %*% d1

     [,1]
[1,]   14

> 3 * (2 * cov(X)[1, 1]/3)

[1] 14

> t(d2) %*% d2

     [,1]
[1,]    8

> 3 * (2 * cov(X)[2, 2]/3)

[1] 8
```

Soma dos produtos cruzados:

```
> t(d1) %*% d2

     [,1]
[1,]   -2
```

A correlação é o coseno:

```
> (t(d1) %*% d2)/(sqrt(t(d1) %*% d1) * sqrt(t(d2) %*%
+     d2))

       [,1]
[1,] -0.189

> cor(X)

       [,1]   [,2]
[1,]  1.000 -0.189
[2,] -0.189  1.000
```

3.4 Exemplo 3.7

Cálculo da variãncia generalizada.

```
> S <- matrix(c(252.04, -68.43, -68.43, 123.67), 2,
+     2)
```

Variância generalizada:

```
> det(S)
```

```
[1] 26487
```

3.5 Exemplo 3.8

Interpretação da inversa generalizada.
Vamos gerar nuvens de pontos e elipses de confiança para 3 distribuições:

```
> library(MASS)
> library(ellipse)
> S1 <- matrix(c(5, 4, 4, 5), 2, 2)
> r1 <- 0.8
> S2 <- matrix(c(3, 0, 0, 3), 2, 2)
> r2 <- 0
> S3 <- matrix(c(5, -4, -4, 5), 2, 2)
> r3 <- -0.8
> eigen(S1)
```

```
$values
[1] 9 1

$vectors
      [,1]   [,2]
[1,] 0.707 -0.707
[2,] 0.707  0.707
```

```
> eigen(S2)
```

```
$values
[1] 3 3

$vectors
     [,1] [,2]
[1,]    0   -1
[2,]    1    0
```

```
> eigen(S3)
```

```
$values
[1] 9 1

$vectors
       [,1]   [,2]
[1,] -0.707 -0.707
[2,]  0.707 -0.707
```

```
> det(S1)
```

```
[1] 9
```

```
> det(S2)
```

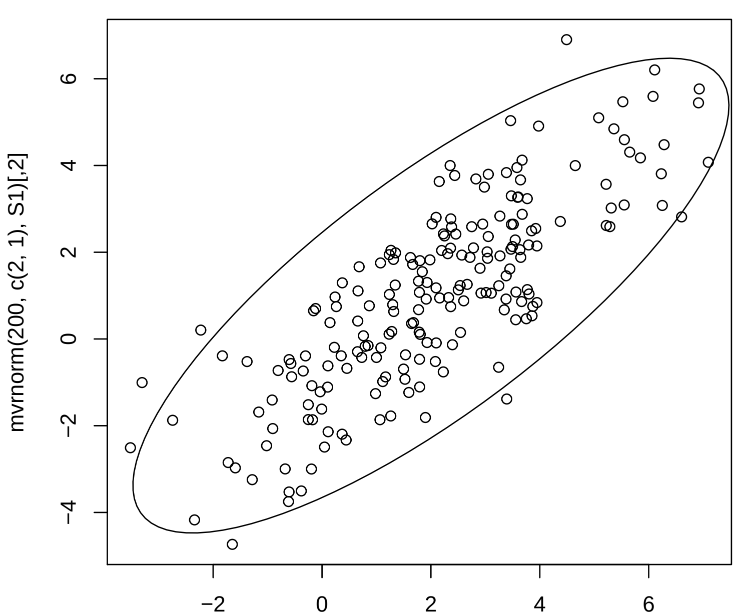
```
[1] 9
```

```
> det(S3)
```

```
[1] 9
```

```
> plot(mvrnorm(200, c(2, 1), S1))
> lines(ellipse(S1, centre = c(2, 1)), type = "l")

> plot(mvrnorm(200, c(2, 1), S2))
> lines(ellipse(S2, centre = c(2, 1)), type = "l")

> plot(mvrnorm(200, c(2, 1), S3))
> lines(ellipse(S3, centre = c(2, 1)), type = "l")
```

```
$values
[1] 9 1

$vectors
      [,1]   [,2]
[1,] 0.707 -0.707
[2,] 0.707  0.707

$values
[1] 3 3

$vectors
     [,1] [,2]
[1,]    0   -1
[2,]    1    0

$values
[1] 9 1

$vectors
       [,1]   [,2]
[1,] -0.707 -0.707
[2,]  0.707 -0.707

[1] 9

[1] 9

[1] 9
```



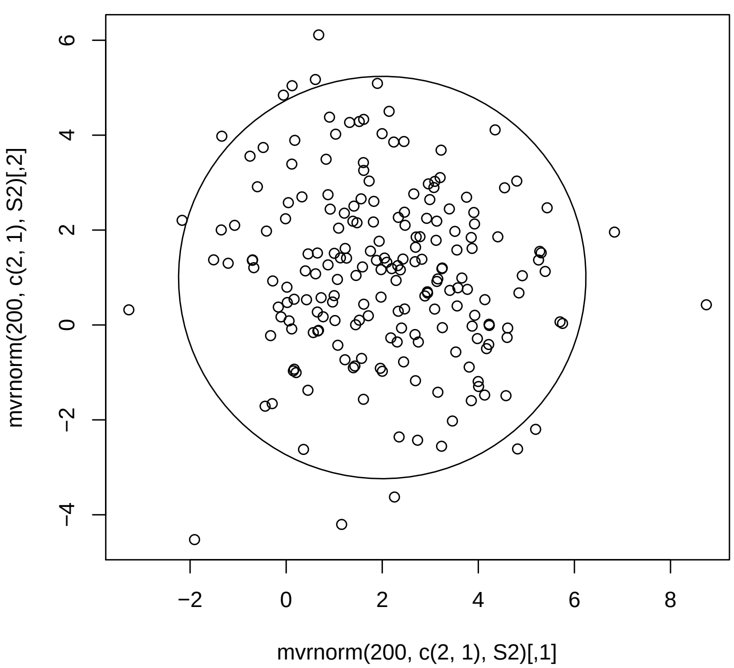


Figura 3.4: Nuvem de pontos e elipse de confiança

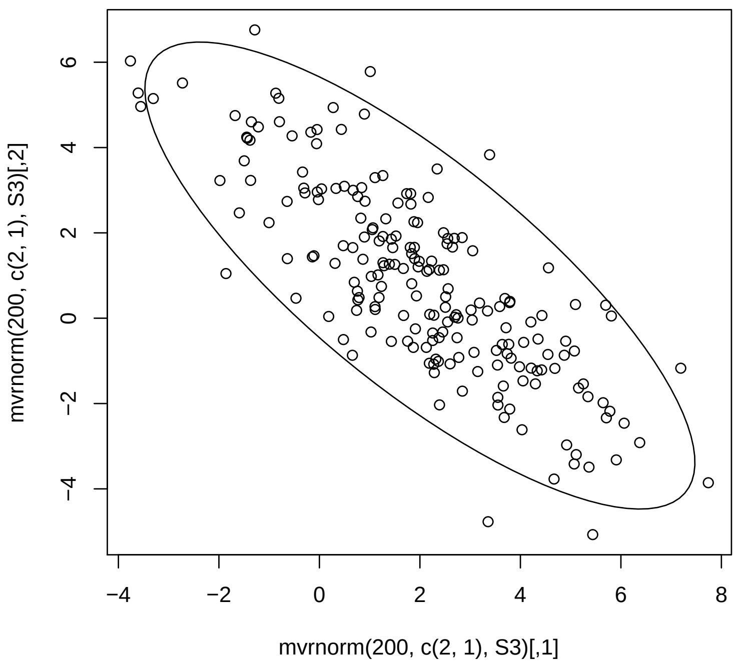


Figura 3.5: Nuvem de pontos e elipse de confiança

3.6 Exemplo 3.9

Um caso em que a variância generalizada é nula.

```
> X <- matrix(c(1, 4, 4, 2, 1, 0, 5, 6, 4), 3, 3)
```

Vetor de médias:

```
> xbar <- matrix(colMeans(X), ncol = 1)
```

Matriz de desvios em relação às médias:

```
> desv <- X - matrix(1, 3, 1) %*% t(xbar)
> all.equal(desv[, 1] + 2 * desv[, 2], desv[, 3])
```

```
[1] TRUE
```

```
> det(cov(X))
```

```
[1] 0
```

3.7 Exemplo 3.10

Criando novas variáveis que levam à variância generalizada nula.

```
> X <- matrix(c(1, 4, 2, 5, 3, 9, 12, 10, 8, 11, 10,
+     16, 12, 13, 14), ncol = 3)
```

Observe que terceira coluna é a soma das duas primeiras.
Matriz de desvios:

```
> desv <- X - matrix(1, NROW(X), 1) %*% t(matrix(colMeans(X),
+     ncol = 1))
> S <- cov(X)
> det(S)
```

```
[1] 0
```

Verificaçao da singularidade:

```
> all.equal(X[, 1] + X[, 2], X[, 3])

[1] TRUE

> 1 * desv[, 1] + 1 * desv[, 2] - 1 * desv[, 3]

[1] 0 0 0 0 0

> eigen(S)

$values
[1] 7.50e+00 2.50e+00 5.33e-15

$vectors
       [,1]      [,2]   [,3]
[1,] -0.408  7.07e-01  0.577
[2,] -0.408 -7.07e-01  0.577
[3,] -0.816  7.77e-16 -0.577

> S %*% eigen(S)$vectors[, 3]

          [,1]
[1,]  2.83e-15
[2,] -3.16e-15
[3,] -1.11e-16
```

3.8 Exemplo 3.11

Ilustração da relação entre $|\mathbf{S}|$ e $|\mathbf{R}|$.

```
> S <- matrix(c(4, 3, 1, 3, 9, 2, 1, 2, 1), nrow = 3)
> R <- diag(diag(S)^(-1/2)) %*% S %*% diag(diag(S)^(-1/2))
> det(S)

[1] 14

> det(R)
```

```
[1] 0.389
```

```
> det(S)
```

```
[1] 14
```

```
> prod(diag(S)) * det(R)
```

```
[1] 14
```

3.9 Exemplo 3.12

Cálculo da variância amostral total.

```
> S1 <- matrix(c(252.04, -68.43, -68.43, 123.67), 2,
+     2)
> sum(diag(S1))
```

```
[1] 376
```

```
> S2 <- matrix(c(3, -3/2, 0, -3/2, 1, 1/2, 0, 1/2,
+     1), 3, 3)
> sum(diag(S2))
```

```
[1] 5
```

3.10 Exemplo 3.13

Médias e covariâncias de combinações lineares.

```
> X <- matrix(c(1, 4, 4, 2, 1, 0, 5, 6, 4), 3, 3)
> l1 <- matrix(c(2, 2, -1), ncol = 1)
> l2 <- matrix(c(1, -1, 3), ncol = 1)
```

Observacões das combinações lineares:

```
> l1x1 <- t(l1) %*% matrix(X[1, ], ncol = 1)
> l1x2 <- t(l1) %*% matrix(X[2, ], ncol = 1)
> l1x3 <- t(l1) %*% matrix(X[3, ], ncol = 1)
> mean(c(l1x1, l1x2, l1x3))
```

```
[1] 3
```

```
> var(c(l1x1, l1x2, l1x3))
```

```
[1] 3
```

Para a segunda c.l.:

```
> l2x1 <- t(l2) %*% matrix(X[1, ], ncol = 1)
> l2x2 <- t(l2) %*% matrix(X[2, ], ncol = 1)
> l2x3 <- t(l2) %*% matrix(X[3, ], ncol = 1)
> mean(c(l2x1, l2x2, l2x3))
```

```
[1] 17
```

```
> var(c(l2x1, l2x2, l2x3))
```

```
[1] 13
```

```
> cov(c(l1x1, l1x2, l1x3), c(l2x1, l2x2, l2x3))
```

```
[1] 4.5
```

Outra forma de calcular:

```
> xbar <- matrix(colMeans(X), ncol = 1)
> S <- cov(X)
```

Médias:

```
> t(l1) %*% xbar
```

```
     [,1]
[1,]    3
```

```
> t(l2) %*% xbar

     [,1]
[1,]   17
```

Variâncias e covariância:

```
> t(l1) %*% S %*% l1

     [,1]
[1,]    3

> t(l2) %*% S %*% l2

     [,1]
[1,]   13

> t(l1) %*% S %*% l2

     [,1]
[1,]  4.5
```

Capítulo 4

Normal multivariada

4.1 Exemplo 4.1

Densidade normal multivariada

```
> library(mvtnorm)
> x <- seq(-3, 3, length = 40)
> y <- x
> xygrid <- expand.grid(x, y)
> sigma <- matrix(c(1, 0.75, 0.75, 1), 2, 2)
> z <- matrix(dmvnorm(xygrid, mean = c(0, 0), sigma),
+     40, 40)
> res <- persp(x, y, z, theta = 60, phi = 30, expand = 0.5,
+     col = "lightblue", ltheta = 120, shade = 0.75,
+     ticktype = "detailed", xlab = "X", ylab = "Y",
+     zlab = "Normal")
> round(res, 3)
```

```
      [,1]   [,2]   [,3]   [,4]
[1,] 0.167 -0.144  0.250 -0.250
[2,] 0.289  0.083 -0.144  0.144
[3,] 0.000  3.611  2.085 -2.085
[4,] 0.000 -0.433 -2.982  3.982
```

```
      [,1]   [,2]   [,3]   [,4]
[1,] 0.167 -0.144  0.250 -0.250
[2,] 0.289  0.083 -0.144  0.144
[3,] 0.000  3.611  2.085 -2.085
[4,] 0.000 -0.433 -2.982  3.982
```

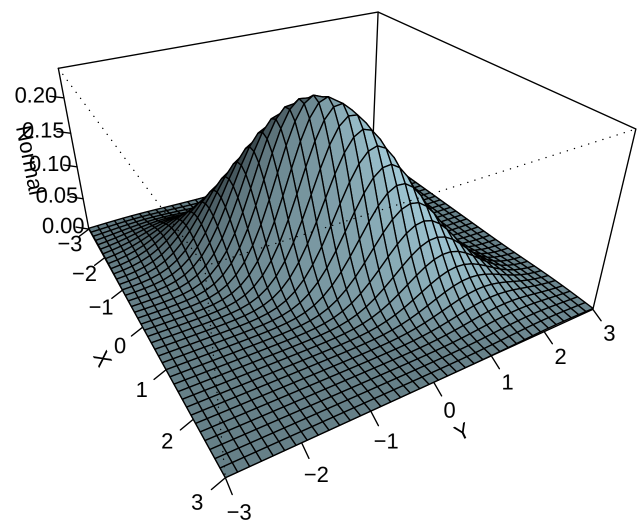


Figura 4.1: Densidade da distribuição normal bivariada

4.2 Exemplo 4.8

Combinações lineares de vetores aleatórios

```
> mu <- matrix(c(3, -1, 1), ncol = 1)
> sigma <- matrix(c(3, -1, 1, -1, 1, 0, 1, 0, 2), 3,
+     3)
> l1 <- matrix(c(1/2, 1/2, 1/2, 1/2), ncol = 1)
> l2 <- matrix(c(1, 1, 1, -3), ncol = 1)
```

Primeira combinação linear de vetores.Vetor de médias:

```
> sum(l1) * mu

     [,1]
[1,]    6
[2,]   -2
[3,]    2
```

Matriz de covariância:

```
> sum(l1^2) * sigma

     [,1] [,2] [,3]
[1,]    3   -1    1
[2,]   -1    1    0
[3,]    1    0    2
```

Segunda combinação linear de vetores.Vetor de médias:

```
> sum(l2) * mu

     [,1]
[1,]    0
[2,]    0
[3,]    0
```

Matriz de covariância:

```
> sum(l2^2) * sigma

     [,1] [,2] [,3]
[1,]   36  -12   12
[2,]  -12   12    0
[3,]   12    0   24
```

Covariância das duas combinações lineares:

```
> sum(l1 * l2) * sigma

     [,1] [,2] [,3]
[1,]    0    0    0
[2,]    0    0    0
[3,]    0    0    0
```

4.3 Exemplo 4.9

Construção de Q-Q plot.

```
> val <- c(-1, -0.1, 0.16, 0.41, 0.62, 0.8, 1.26, 1.54,
+     1.71, 2.3)
> niv.prob <- ((1:length(val)) - 1/2)/length(val)
> quant <- round(qnorm(niv.prob), 3)

> plot(quant, val)


> data.frame(valores = val, niv.prob = niv.prob, quantis = round(qnorm(niv.prob),
+     3))

   valores niv.prob quantis
1    -1.00     0.05  -1.645
2    -0.10     0.15  -1.036
3     0.16     0.25  -0.674
4     0.41     0.35  -0.385
5     0.62     0.45  -0.126
```

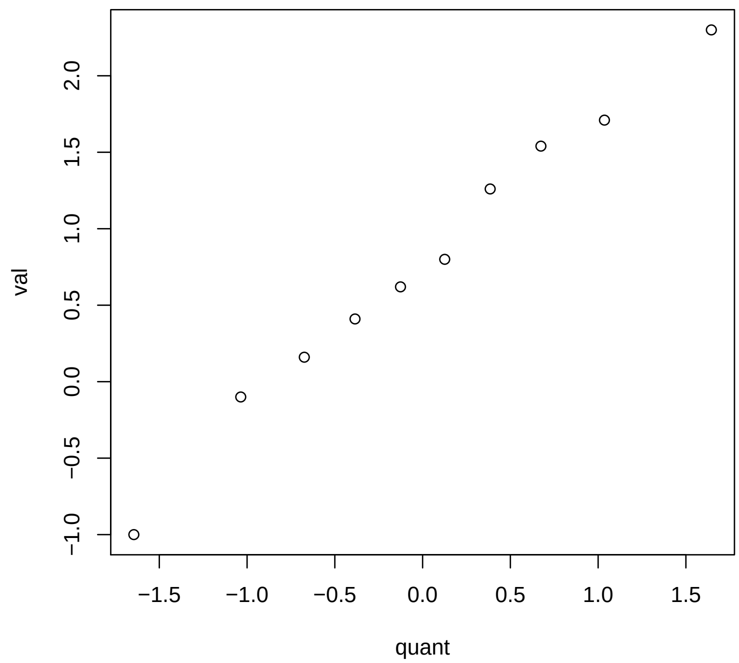


Figura 4.2: Q-Q plot dos dados do exemplo 4.9

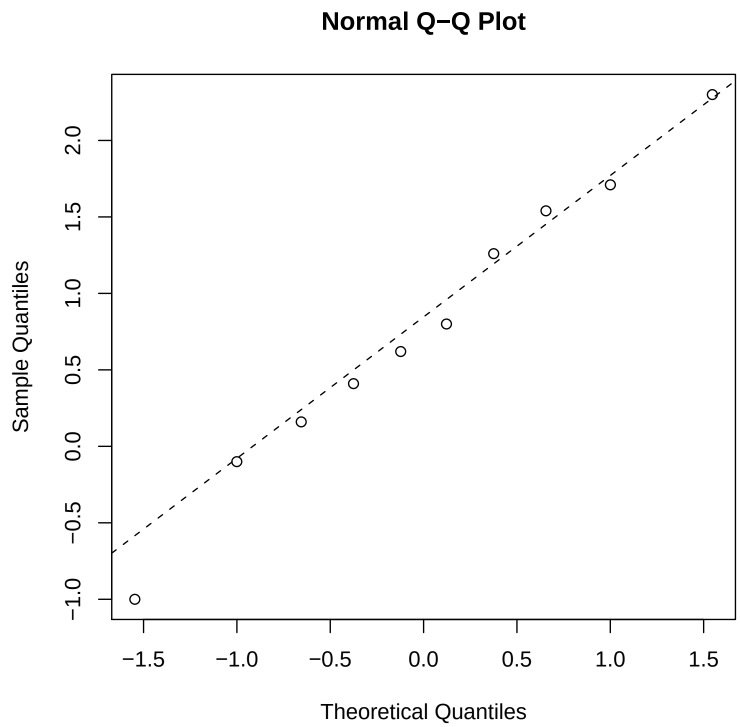


Figura 4.3: Q-Q plot normal dos dados do exemplo 4.9

```
6     0.80     0.55   0.126
7     1.26     0.65   0.385
8     1.54     0.75   0.674
9     1.71     0.85   1.036
10    2.30     0.95   1.645

> qqnorm(val)
> qqline(val, lty = 2)
```

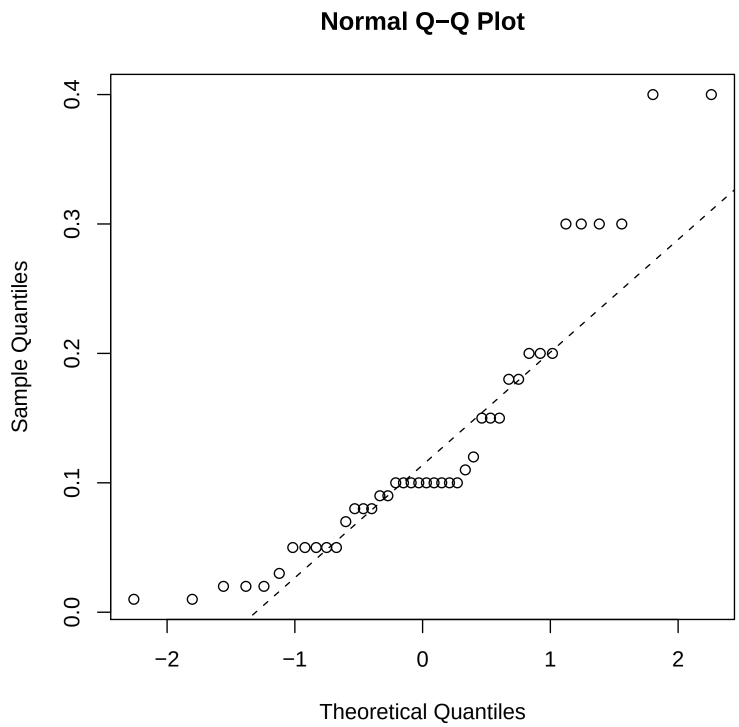


Figura 4.4: Q-Q plot normal dos dados do exemplo 4.10

4.4 Exemplo 4.10

Um Q-Q plot dos dados de radiação.

```
> tab4.1 <- read.table("T4-1.dat")
```

```
> qqnorm(tab4.1$V1)
> qqline(tab4.1$V1, lty = 2)
```

4.5 Exemplo 4.11

Um teste de coeficiente de correlação para a normalidade.

```
> alfa <- 0.1
> quantil <- qnorm(niv.prob)
> n <- length(quantil)
> cor(val, quantil)

[1] 0.994
```

Direto pelo R:

```
> shapiro.test(val)

        Shapiro-Wilk normality test

data:  val
W = 0.99, p-value = 0.9968
```

Não rejeita normalidade. Tamanho da amostra muito pequeno!

4.6 Exemplo 4.12

Verificação da normalidade bivariada:

```
> Prob1.4 <- read.table("P1-4.dat")
> n <- nrow(Prob1.4[, 1:2])
> p <- ncol(Prob1.4[, 1:2])
> xbar <- matrix(colMeans(Prob1.4[, 1:2]), ncol = 1)
> S <- cov(Prob1.4[, 1:2])
> x0 <- matrix(c(126.974, 4224), ncol = 1)
```

Verificar se x_0 cai dentro do contorno de 50%:

```
> t(x0 - xbar) %*% solve(S) %*% (x0 - xbar) <= qchisq(0.5,
+     2)
```

```
      [,1]
[1,] FALSE
```

```
> library(ellipse)
> plot(ellipse(S, centre = as.vector(xbar), level = 0.5),
+     type = "l")
> points(Prob1.4[, 1:2])
> points(xbar[1, 1], xbar[2, 1], pch = 16)
> segments(xbar[1, 1], 0, xbar[1, 1], xbar[2, 1], lty = 2)
> segments(0, xbar[2, 1], xbar[1, 1], xbar[2, 1], lty = 2)
```

Cálculo da proporção de pontos que cai dentro do contorno:

```
> sum(mahalanobis(as.matrix(Prob1.4[, 1:2]), xbar,
+     S) <= qchisq(0.5, 2))/n
```

```
[1] 0.7
```

A amostra é muito pequena para se afirmar qualquer coisa.

4.7 Exemplo 4.13

Função para desenhar um plot qui-quadrado:

```
> chisqPlot <- function(x) {
+     n <- nrow(x)
+     p <- ncol(x)
+     xbar <- colMeans(x)
+     S <- cov(x)
+     di <- mahalanobis(x, xbar, S)
+     index <- ((1:n) - 1/2)/n
+     quant <- qchisq(index, p)
+     plot(quant, sort(di), ylab = "Ordered distances",
+         xlab = "Chi-square quantile", lwd = 2, pch = 1)
+ }
```

Aplicação aos dados do Exemplo 1.4:

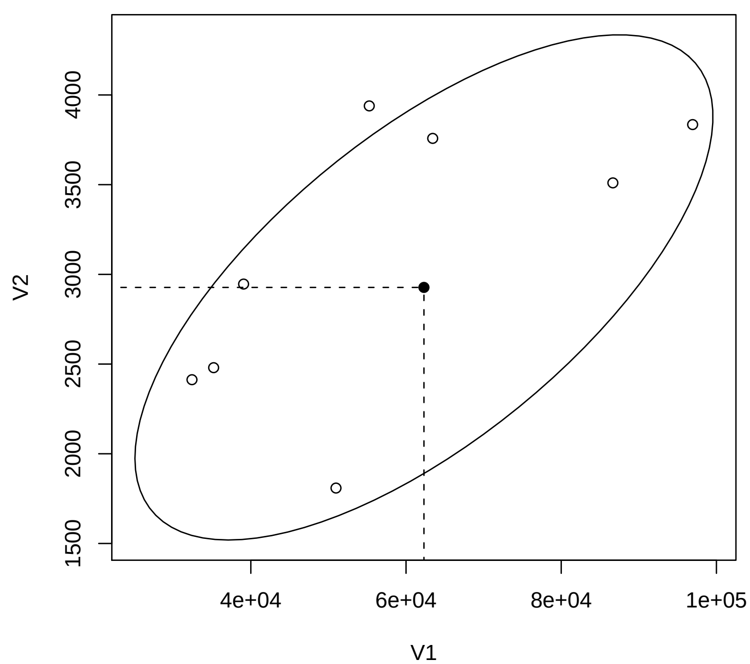


Figura 4.5: Gráfico do contorno de 50% de uma normal bivariada

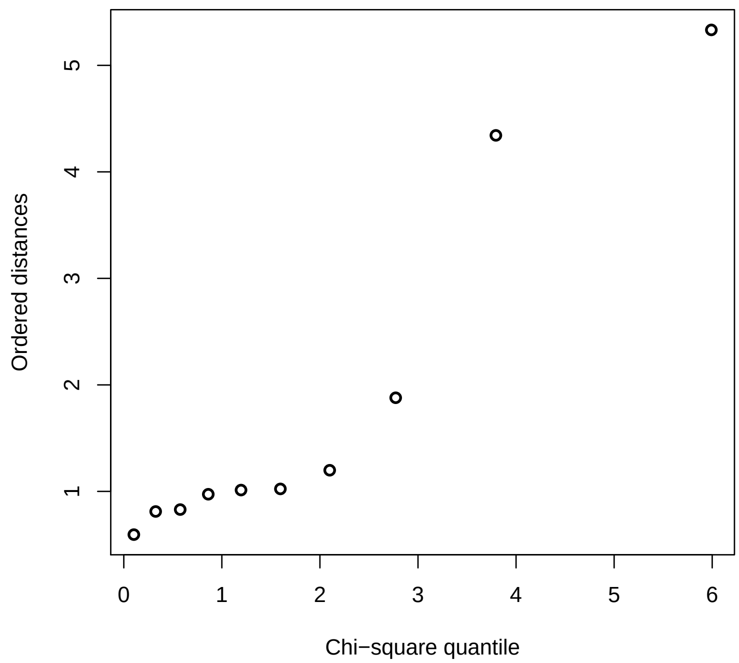


Figura 4.6: Plot qui-quadrado dos dados do exemplo 1.4

```
> chisqPlot(as.matrix(Prob1.4[, 1:2]))
```

Plot qui-quadradado para 2 amostras normais com n=30; p=4 (Figura 4.8)

```
> library(MASS)
> amost1 <- mvrnorm(30, rep(0, 4), diag(rep(1, 4)))
> amost2 <- mvrnorm(30, rep(0, 4), diag(rep(1, 4)))
> par(mfrow = c(1, 2))
> chisqPlot(amost1)
> chisqPlot(amost2)
```

4.8 Exemplo 4.14

Avaliando a normalidade multivariada de um conjunto de dados com quatro variáveis.

```
> Tab4.3 <- read.table("T4-3.dat", col.names = c("x1",
+     "x2", "x3", "x4", "d2"))
```

```
> qqnorm(Tab4.3[, 1])
```

Identificação de outlier:

```
> (1:nrow(Tab4.3))[Tab4.3[, 1] == max(Tab4.3[, 1])]
```

```
[1] 9
```

```
> qqnorm(Tab4.3[, 2])
```

Identificação de outlier:

```
> (1:nrow(Tab4.3))[Tab4.3[, 2] == max(Tab4.3[, 2])]
```

```
[1] 9
```

```
> qqnorm(Tab4.3[, 3])
```

Identificação de outliers:

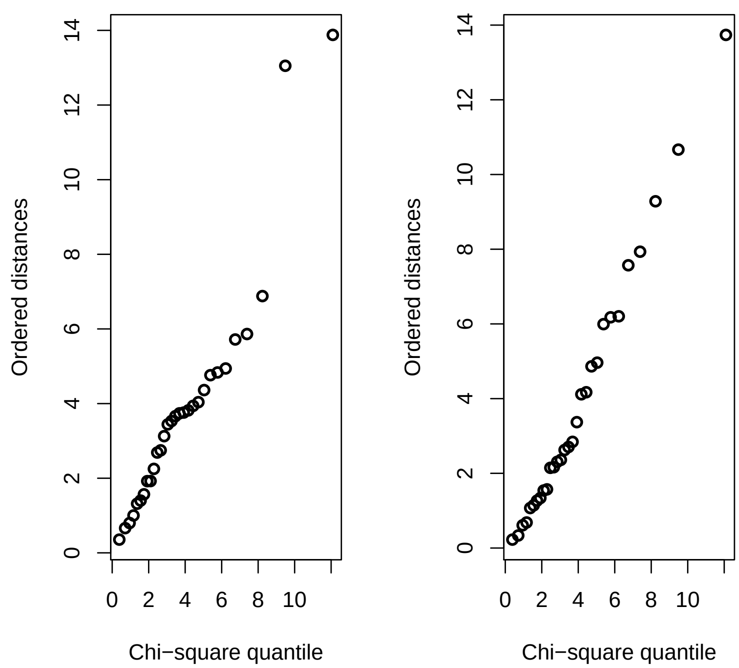


Figura 4.7: Plot qui-quadrado de duas amostras normais

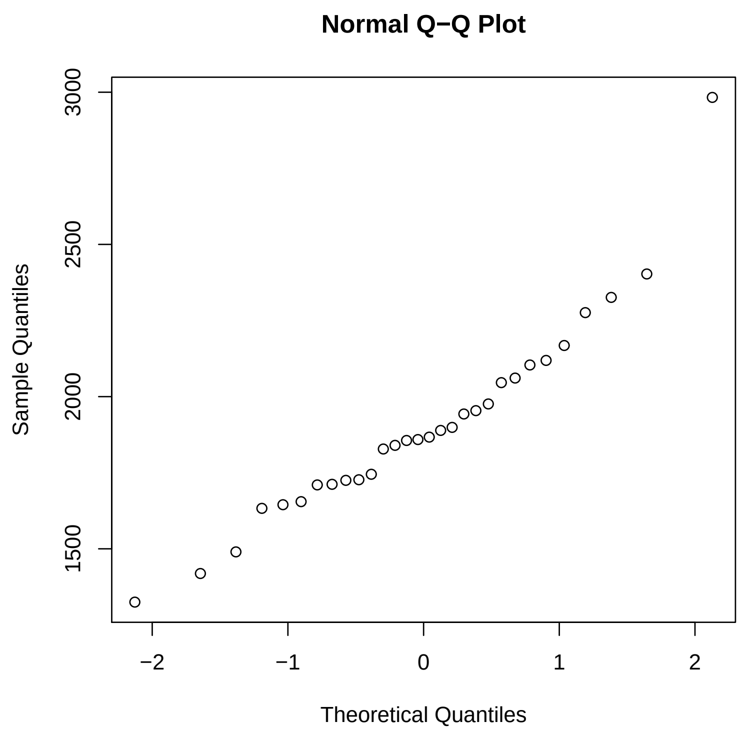


Figura 4.8: Plot normal dos dados da Tabela 4.3-Variável 1

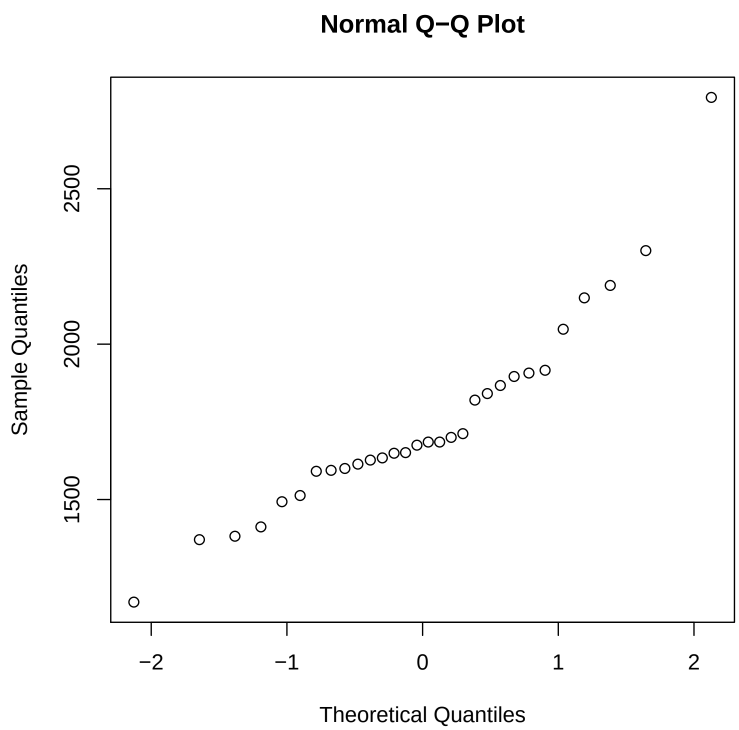


Figura 4.9: Plot normal dos dados da Tabela 4.3-Variável 2

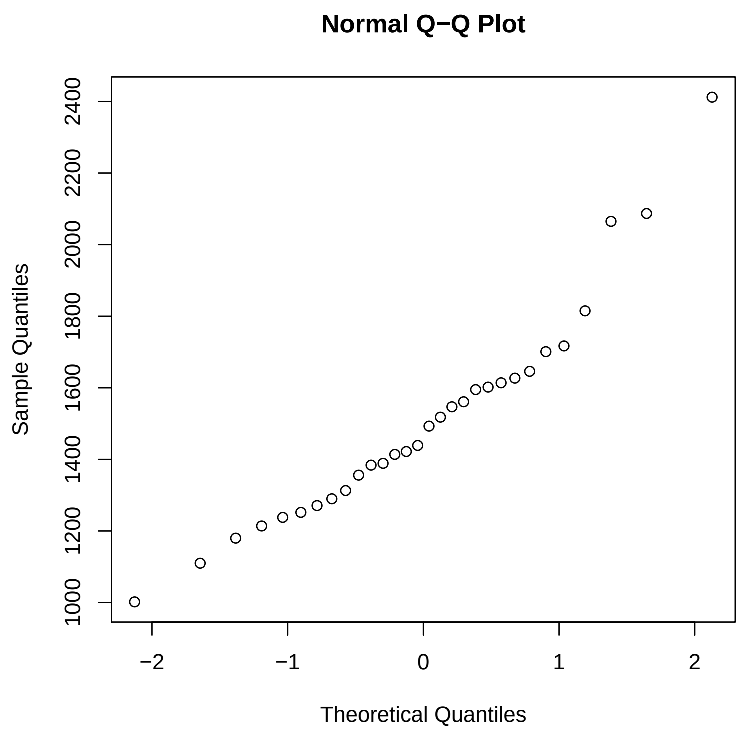


Figura 4.10: Plot normal dos dados da Tabela 4.3-Variável 3

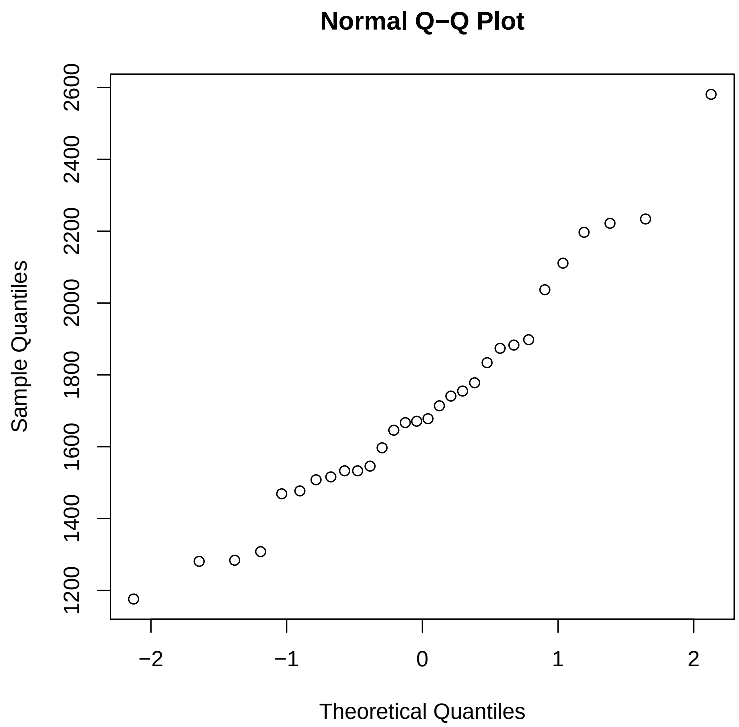


Figura 4.11: Plot normal dos dados da Tabela 4.3-Variável 4

```
> n <- nrow(Tab4.3)
> rownames(Tab4.3[order(Tab4.3[, 3]), ])[((n - 2):n)]

[1] "29" "2"  "9"

> qqnorm(Tab4.3[, 4])
```

Identificação de outliers:

```
> n <- nrow(Tab4.3)
> rownames(Tab4.3[order(Tab4.3[, 3]), ])[((n - 2):n)]
```

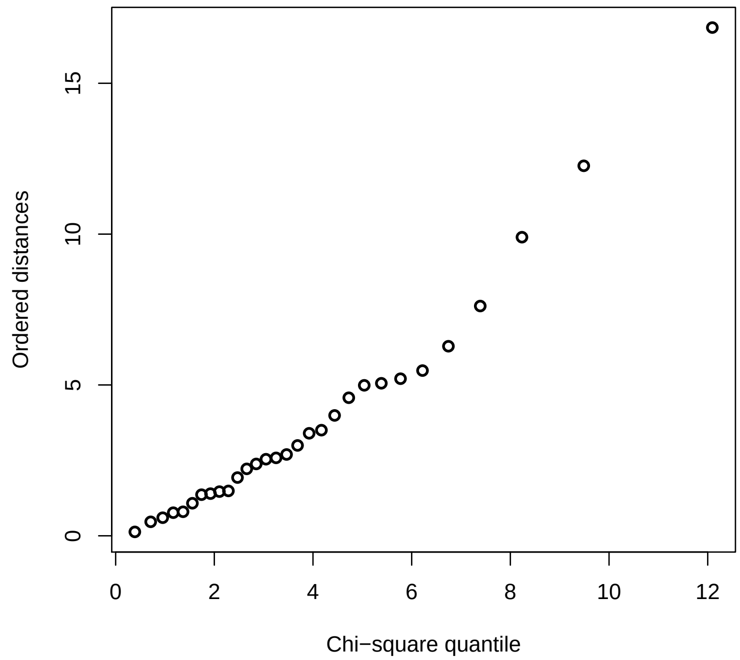


Figura 4.12: Plot qui-quadrado das variáveis da tabela 4.3

```
[1] "29" "2"  "9"

> (1:nrow(Tab4.3))[Tab4.3[, 4] == max(Tab4.3[, 4])]

[1] 9

> chisqPlot(as.matrix(Tab4.3[, 1:4]))
```

4.9 Exemplo 4.15

Detecção de outliers nos dados sobre tábuas.

```
> tab4.3 <- read.table("T4-3.dat", col.names = c("x1",
+     "x2", "x3", "x4", "d2"))
> tab4.3_scale <- scale(tab4.3[, 1:4])
> dimnames(tab4.3_scale)[[2]] <- c("z1", "z2", "z3",
+     "z4")
> tab4.3 <- cbind(tab4.3, tab4.3_scale)
```

Detecção de outliers:

```
> qchisq(0.005, 4, lower.tail = FALSE)

[1] 14.9

> boxplot(tab4.3$d2)


> n <- nrow(tab4.3)
> rot <- rownames(tab4.3[order(tab4.3$d2), ])[(n -
+     1):n]
> tab4.3$atip <- rep(0, n)
> tab4.3[rot, "atip"] <- 1
> pairs(tab4.3[, 1:4], pch = c(1, 16)[as.factor(tab4.3$atip)])
```

Identificar outliers no gráfico:

4.10 Exemplo 4.16

Determinação de uma transformação de potência para dados univariados.

```
> tab4.1 <- read.table("T4-1.dat")

> qqnorm(tab4.1$V1)
```

Transformação de Box=Cox:

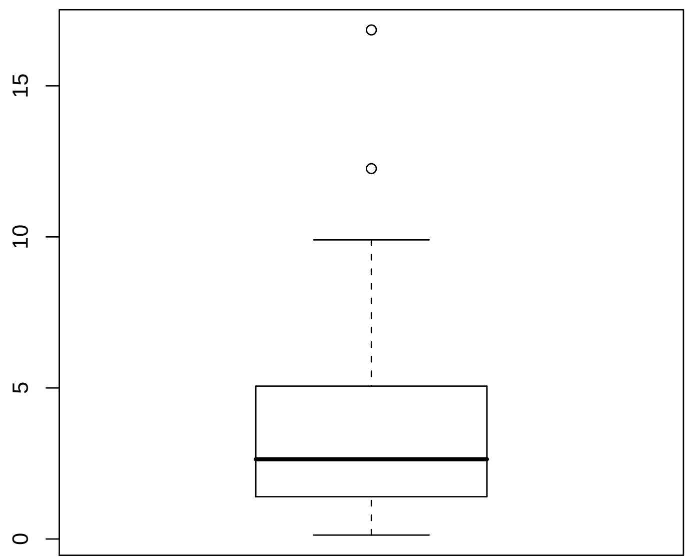


Figura 4.13: Plot normal dos dados da Tabela 4.3-Variável 1

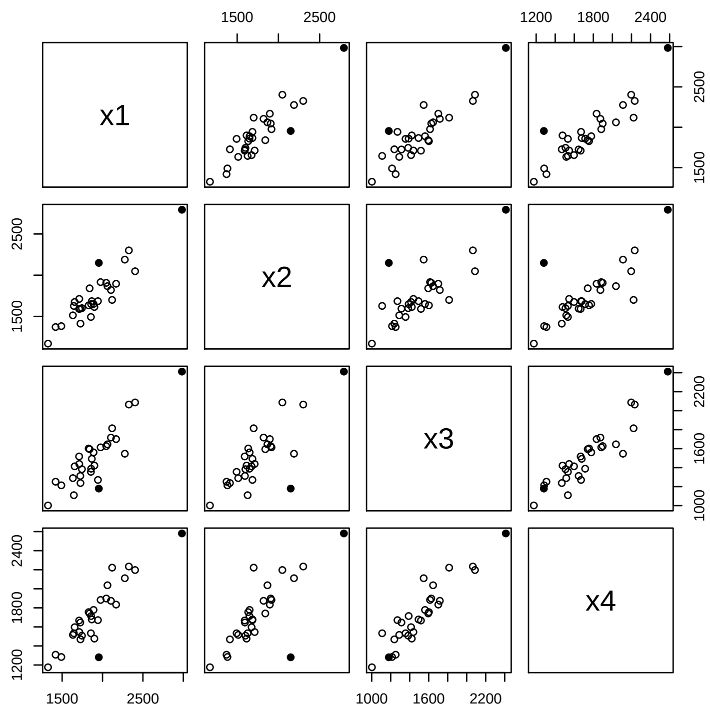


Figura 4.14: Diagramas de dispersão de dados de dureza de tábuas

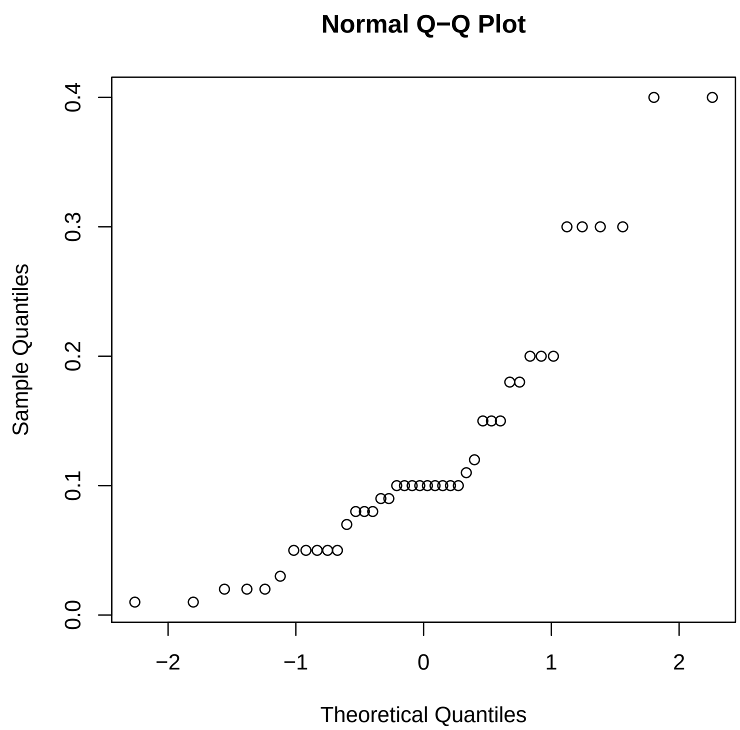


Figura 4.15: Plot normal da variável V1 da Tabela 4.1

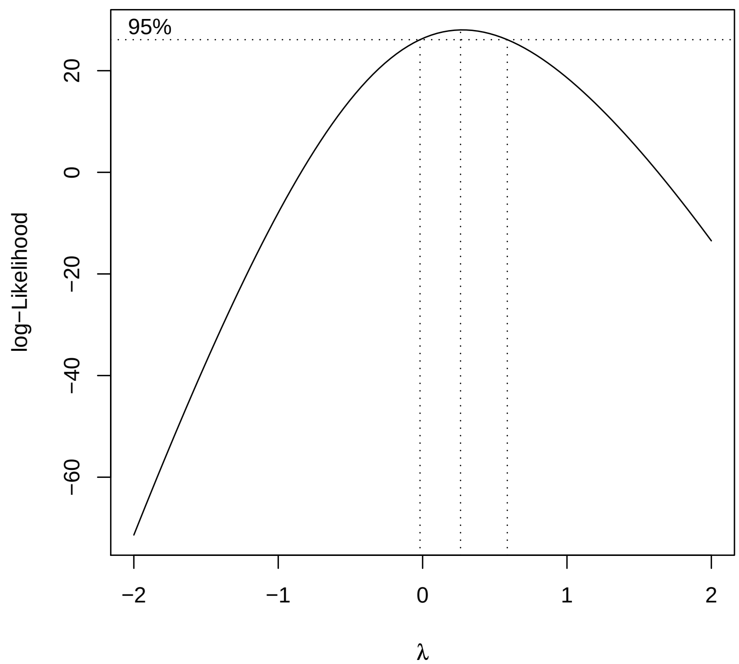


Figura 4.16: Gráfico para determinação de λ para a variável V1 da Tabela 4.1

```
> library(MASS)
> transf <- boxcox(V1 ~ 1, data = tab4.1, plotit = F)
> boxcox(V1 ~ 1, data = tab4.1, plotit = T)
```

```
> index <- (1:length(transf$y))[transf$y == max(transf$y)]
> lambda <- transf$x[index]
```

A transformação seria $(x^{.30} - 1)/.30$. No livro foi tomado $\lambda = .25$.

```
> tab4.1 <- transform(tab4.1, V1.T = V1^(1/4)/(1/4))
> qqnorm(tab4.1$V1.T)
```

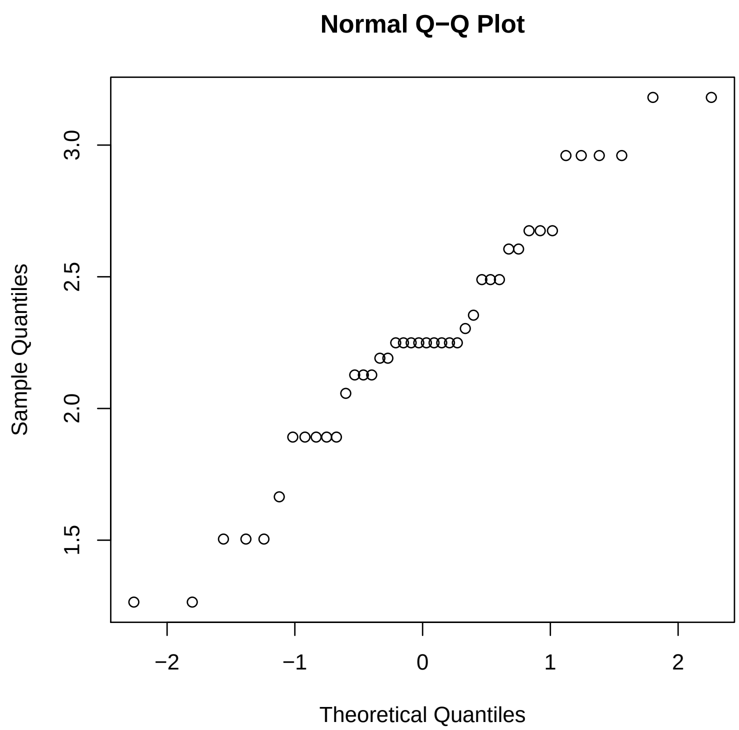


Figura 4.17: Plot normal da variável V1 transformada

Capítulo 5

Inferência sobre a média

5.1 Exemplo 5.1

Avaliação de T^2.

```
> X <- matrix(c(6, 10, 8, 9, 6, 3), 3, 2)
> mu0 <- matrix(c(9, 5), ncol = 1)
> n <- nrow(X)
> p <- ncol(X)
> xbar <- matrix(colMeans(X), ncol = 1)
> S <- cov(X)
> T2 <- 3 * t(xbar - mu0) %*% solve(S) %*% (xbar -
+     mu0)
> T2
```

```
      [,1]
[1,] 0.778
```

Comparar T^2 com:

```
> (((n - 1) * p)/(n - p)) * qf(0.05, p, (n - p), lower.tail = FALSE)
```

```
[1] 798
```

5.2 Exemplo 5.2

Teste de vetor de média multivariado com T^2

```
> tab5.1 <- read.table("t5-1.dat")
> n <- nrow(tab5.1)
> p <- ncol(tab5.1)
> mu0 <- c(4, 50, 10)
> args(mahalanobis)

function (x, center, cov, inverted = FALSE, ...)
NULL
```

Estatística T^2:

```
> T2 <- n * mahalanobis(mu0, colMeans(tab5.1), cov(tab5.1))
```

Valor crítico:

```
> Val.crit <- ((n - 1) * p/(n - 3)) * qf(0.1, p, n -
+     p, lower.tail = F)
> T2 > Val.crit

[1] TRUE
```

Função para o teste T^2 de Hotelling:

```
> T2.Hotteling <- function(mu0, data) {
+     S <- cov(data)
+     n <- nrow(data)
+     p <- ncol(data)
+     xbar <- colMeans(data)
+     T2 <- n * mahalanobis(mu0, xbar, S)
+     T2.f <- ((n - p)/((n - 1) * p)) * T2
+     pvalor <- 1 - pf(T2.f, p, (n - p))
+     return(list(estat = T2, p.value = pvalor))
+ }
> T2.Hotteling(mu0, tab5.1)
```

```
$estat
[1] 9.74

$p.value
[1] 0.0649
```

5.3 Exemplo 5.3

Construção de uma elipse de confiança para μ.

```
> PF <- read.table("t4-1.dat")
> PA <- read.table("t4-5.dat")
> tab4.1.5 <- cbind(PF, PA)
> names(tab4.1.5) <- c("PF", "PA")
```

Transformação dos dados:

```
> tab4.1.5$PF <- (tab4.1.5$PF)^(1/4)
> tab4.1.5$PA <- (tab4.1.5$PA)^(1/4)
> xbar <- matrix(colMeans(tab4.1.5), ncol = 1)
> S <- cov(tab4.1.5)
> n <- nrow(tab4.1.5)
> p <- ncol(tab4.1.5)

> library(ellipse)
> cte <- sqrt((((n - 1) * p)/(n * (n - p))) * qf(0.95,
+     p, (n - p)))
> AU <- eigen(S)
> P1 <- colMeans(tab4.1.5) + (sqrt(AU$values[1]) *
+     cte) * AU$vectors[, 1]
> P2 <- colMeans(tab4.1.5) - (sqrt(AU$values[1]) *
+     cte) * AU$vectors[, 1]
> Q1 <- colMeans(tab4.1.5) + (sqrt(AU$values[2]) *
+     cte) * AU$vectors[, 2]
> Q2 <- colMeans(tab4.1.5) - (sqrt(AU$values[2]) *
+     cte) * AU$vectors[, 2]
```

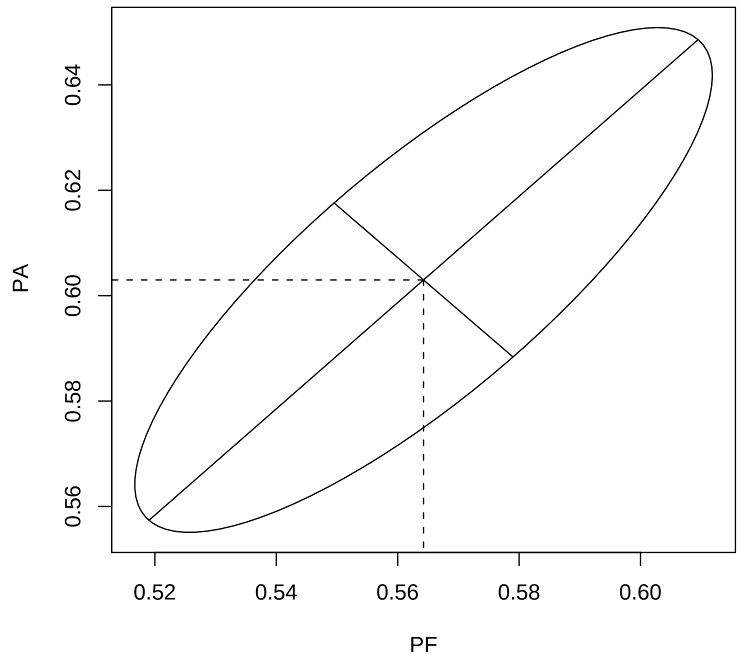


Figura 5.1: Elipse de confiança de 95%

```
> plot(ellipse(S, centre = colMeans(tab4.1.5), t = cte),
+     type = "l")
> segments(P1[1], P1[2], P2[1], P2[2])
> segments(Q1[1], Q1[2], Q2[1], Q2[2])
> segments(colMeans(tab4.1.5)[1], colMeans(tab4.1.5)[2],
+     0, colMeans(tab4.1.5)[2], lty = 2)
> segments(colMeans(tab4.1.5)[1], colMeans(tab4.1.5)[2],
+     colMeans(tab4.1.5)[1], 0, lty = 2)
```

Verificar se o ponto $x_0 = (.562, .589)$ cai dentro da elipse:

```
> mu <- matrix(c(0.562, 0.589), ncol = 1)
> n * t(xbar - mu) %*% solve(S) %*% (xbar - mu) <=
+     ((n - 1) * p/(n - p)) * qf(0.95, p, (n - p))

     [,1]
[1,] TRUE
```

Ponto na região não rejeitaria $\mu = (.562, .589)$ com $\alpha = 5\%$. Função para calcular I.C. simultâneos:

```
> Simul.Int <- function(a, data, alfa = 0.05, large.samp = FALSE) {
+     n <- nrow(data)
+     p <- ncol(data)
+     xbar <- colMeans(data)
+     S <- cov(data)
+     a <- matrix(a, ncol = 1)
+     xbar <- matrix(xbar, ncol = 1)
+     var.lin <- t(a) %*% cov(data) %*% a
+     if (large.samp) {
+         liminf <- t(a) %*% xbar - sqrt(qchisq(alfa,
+             p, lower.tail = FALSE) * sqrt(var.lin/n))
+         limsup <- t(a) %*% xbar + sqrt(qchisq(alfa,
+             p, lower.tail = FALSE) * sqrt(var.lin/n))
+         return(c(liminf, limsup))
+     }
+     cte <- (p * (n - 1))/(n * (n - p))
+     comp <- sqrt(cte * var.lin * qf(alfa, p, (n -
+         p), lower.tail = FALSE))
+     liminf <- t(a) %*% xbar - comp
+     limsup <- t(a) %*% xbar + comp
+     c(liminf, limsup)
+ }
```

5.4 Exemplo 5.4

Intervalos de confiança simultâneos como sombras do elipsóide de confiança.

```
> PF <- read.table("t4-1.dat")
> PA <- read.table("t4-5.dat")
> tab4.1.5 <- cbind(PF, PA)
> names(tab4.1.5) <- c("PF", "PA")
> tab4.1.5$PF <- (tab4.1.5$PF)^(1/4)
> tab4.1.5$PA <- (tab4.1.5$PA)^(1/4)
> CI.1 <- Simul.Int(a = c(1, 0), tab4.1.5)
> CI.2 <- Simul.Int(a = c(0, 1), tab4.1.5)
> S <- cov(tab4.1.5)

> library(ellipse)
> cte <- sqrt((((n - 1) * p)/(n * (n - p))) * qf(0.95,
+     p, (n - p)))
> plot(ellipse(S, centre = colMeans(tab4.1.5), t = cte),
+     type = "l")
> segments(CI.1[1], 0, CI.1[1], CI.2[2], lty = 2)
> segments(CI.1[2], 0, CI.1[2], CI.2[2], lty = 2)
> segments(0, CI.2[1], CI.1[2], CI.2[1], lty = 2)
> segments(0, CI.2[2], CI.1[2], CI.2[2], lty = 2)
```

5.5 Exemplo 5.5

Construção de intervalos de confiança simultâneos e elipses.

```
> tab5.2 <- read.table("t5-2.dat", col.names = c("CSH",
+     "VERBAL", "CIENCIA"))
> xbar <- matrix(colMeans(tab5.2), ncol = 1)
> S <- cov(tab5.2)
```

Intervalos de confiança simutâneos para μ_1, μ_2 e μ_3:

```
> n <- nrow(tab5.2)
> p <- ncol(tab5.2)
> cte <- ((p * (n - 1))/(n - p)) * qf(0.05, p, n -
+     p, lower.tail = FALSE)
```

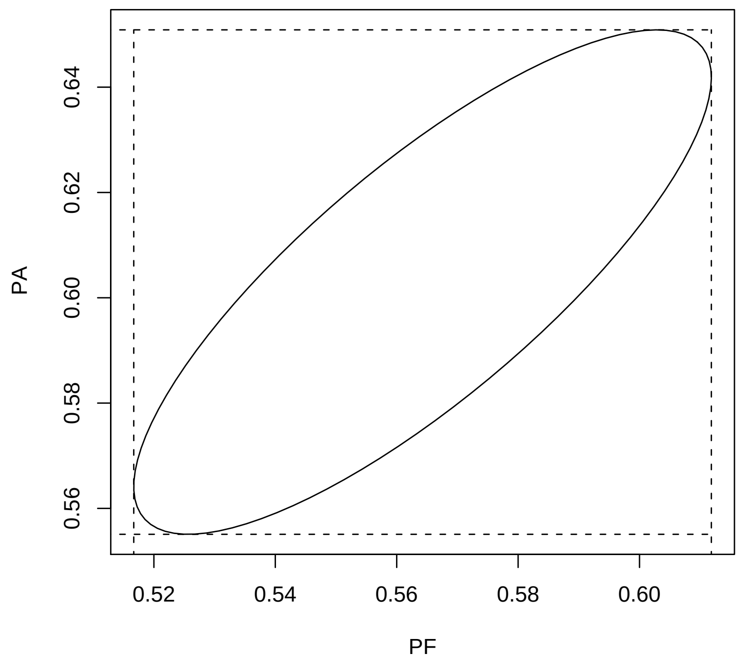


Figura 5.2: Elipse de confiança de 95%

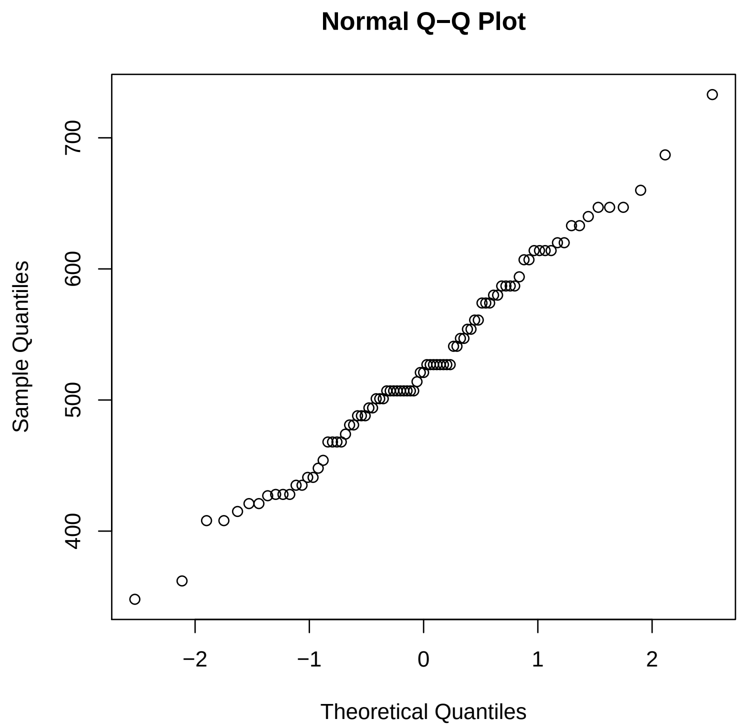


Figura 5.3: Q-Q plot normal da primeira coluna

Intervalos Simultâneos:

```
> CI.1 <- Simul.Int(a = c(1, 0, 0), tab5.2)
> CI.2 <- Simul.Int(a = c(0, 1, 0), tab5.2)
> CI.3 <- Simul.Int(a = c(0, 0, 1), tab5.2)
```

Dados são normais:

```
> qqnorm(tab5.2[, 1])
```

```
> qqnorm(tab5.2[, 2])
```

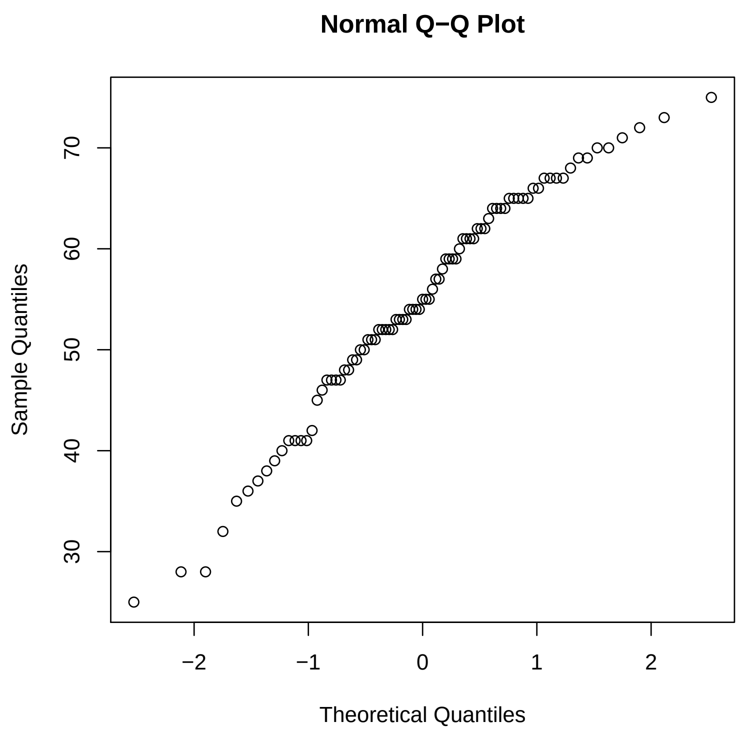


Figura 5.4: Q-Q plot normal da segunda coluna

Curvatura para escore verbal. Intervalo para a diferença de médias

```
> Simul.Int(a = c(0, 1, -1), tab5.2)

[1] 26.4 32.7

> cte <- sqrt((((n - 1) * p)/(n * (n - p))) * qf(0.95,
+     p, (n - p)))
> plot(ellipse(S[1:2, 1:2], centre = colMeans(tab5.2)[1:2],
+     t = cte), type = "l")
> segments(CI.1[1], 0, CI.1[1], CI.2[2], lty = 2)
> segments(CI.1[2], 0, CI.1[2], CI.2[2], lty = 2)
> segments(0, CI.2[1], CI.1[2], CI.2[1], lty = 2)
> segments(0, CI.2[2], CI.1[2], CI.2[2], lty = 2)
```

```
> plot(ellipse(S[c(1, 3), c(1, 3)], centre = colMeans(tab5.2)[c(1,
+     3)], t = cte), type = "l")
> segments(CI.1[1], 0, CI.1[1], CI.3[2], lty = 2)
> segments(CI.1[2], 0, CI.1[2], CI.3[2], lty = 2)
> segments(0, CI.3[1], CI.1[2], CI.3[1], lty = 2)
> segments(0, CI.3[1], CI.1[2], CI.3[2], lty = 2)
```

```
> plot(ellipse(S[2:3, 2:3], centre = colMeans(tab5.2)[2:3],
+     t = cte), type = "l")
> segments(CI.2[1], 0, CI.2[1], CI.3[2], lty = 2)
> segments(CI.2[2], 0, CI.2[2], CI.3[2], lty = 2)
> segments(0, CI.3[1], CI.2[2], CI.3[1], lty = 2)
> segments(0, CI.3[2], CI.2[2], CI.3[2], lty = 2)
```

5.6 Exemplo 5.6

Construção de intervalos de confiança simultâneos de Bonferroni e comparação deles com intervalos T^2.

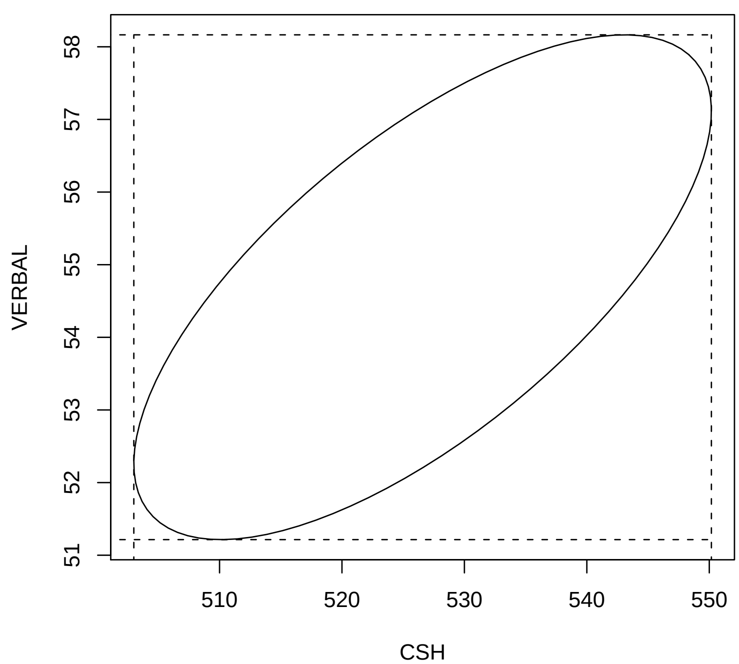


Figura 5.5: Região de Confiança para o par de médias μ_2, μ_3

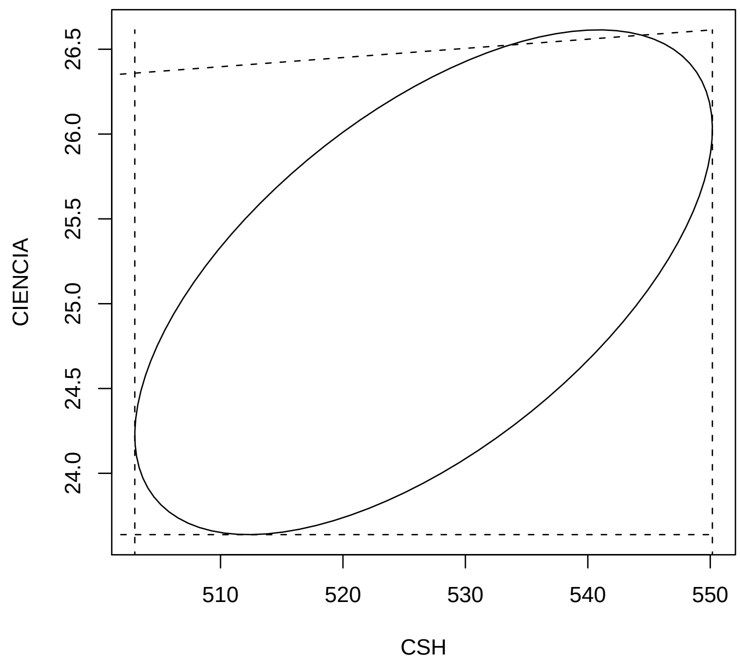


Figura 5.6: I.C. Simultâneos e Região de Confiança

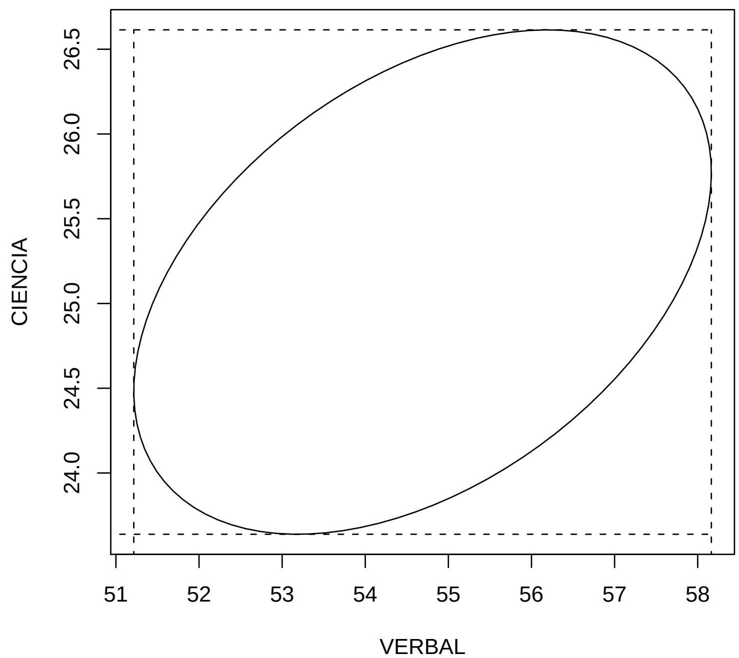


Figura 5.7: I.C. Simultâneos e Região de Confiança

```
> alfa <- 0.05
> n <- nrow(tab4.1.5)
> p <- ncol(tab4.1.5)
> xbar <- colMeans(tab4.1.5)
> S <- cov(tab4.1.5)
> ICB1.inf <- xbar[1] - qt(alfa/(2 * p), n - 1, lower.tail = FALSE) *
+     sqrt(S[1, 1]/n)
> ICB1.sup <- xbar[1] + qt(alfa/(2 * p), n - 1, lower.tail = FALSE) *
+     sqrt(S[1, 1]/n)
> ICB2.inf <- xbar[2] - qt(alfa/(2 * p), n - 1, lower.tail = FALSE) *
+     sqrt(S[2, 2]/n)
> ICB2.sup <- xbar[2] + qt(alfa/(2 * p), n - 1, lower.tail = FALSE) *
+     sqrt(S[2, 2]/n)
> c(ICB1.inf, ICB1.sup)

   PF    PF
0.521 0.607

> c(ICB2.inf, ICB2.sup)

   PA    PA
0.560 0.646
```

5.7 Figura 5.4

```
> CI.1 <- Simul.Int(a = c(1, 0), tab4.1.5)
> CI.2 <- Simul.Int(a = c(0, 1), tab4.1.5)
> S <- cov(tab4.1.5)

> library(ellipse)
> cte <- sqrt((((n - 1) * p)/(n * (n - p))) * qf(0.95,
+     p, (n - p)))
> plot(ellipse(S, centre = colMeans(tab4.1.5), t = cte),
+     type = "l")
> segments(CI.1[1], 0, CI.1[1], CI.2[2], lty = 2)
> segments(CI.1[2], 0, CI.1[2], CI.2[2], lty = 2)
```

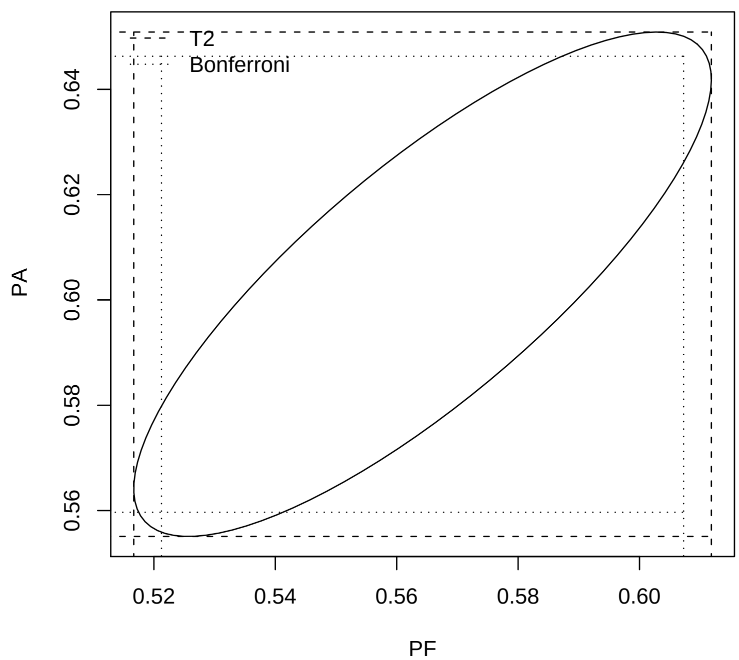


Figura 5.8: I.C. Simultâneos de Bonferroni e Região de Confiança

```
> segments(0, CI.2[1], CI.1[2], CI.2[1], lty = 2)
> segments(0, CI.2[2], CI.1[2], CI.2[2], lty = 2)
> segments(ICB1.inf, 0, ICB1.inf, ICB2.sup, lty = 3)
> segments(ICB1.sup, 0, ICB1.sup, ICB2.sup, lty = 3)
> segments(0, ICB2.inf, ICB1.sup, ICB2.inf, lty = 3)
> segments(0, ICB2.sup, ICB1.sup, ICB2.sup, lty = 3)
> legend("topleft", c("T2", "Bonferroni"), lty = c(2,
+     3), bty = "n")
```

5.8 Tabela 5.4

Razão entre comprimentos de intervalos de Bonferroni e T^2 para $\alpha = .95$ e $\alpha_i = .05/m$.

```
> alfa <- 0.05
> tab5.4 <- matrix(NA, 5, 3)
> n <- c(15, 25, 50, 100)
> p <- m <- c(2, 4, 10)
> for (i in 1:4) {
+     for (k in 1:3) {
+         comp.bonf <- qt(alfa/(2 * m[k]), (n[i] -
+             1), lower.tail = FALSE)
+         comp.T2 <- sqrt((((p[k] * (n[i] - 1))/(n[i] -
+             p[k])) * qf(alfa, p[k], (n[i] - p[k]),
+             lower.tail = FALSE))
+         tab5.4[i, k] <- comp.bonf/comp.T2
+     }
+ }
> for (k in 1:3) {
+     tab5.4[5, k] <- qnorm(alfa/(2 * m[k]), lower.tail = FALSE)/sqrt(qchisq(alfa
+         p[k], lower.tail = FALSE))
+ }
> dimnames(tab5.4) <- list(c("n=15", "n=25", "n=50",
+     "n=100", "n=inf"), c("m=2", "m=4", "m=10"))
> round(tab5.4, 2)
```

```
       m=2  m=4 m=10
n=15  0.88 0.69 0.29
n=25  0.89 0.75 0.48
n=50  0.91 0.78 0.58
n=100 0.91 0.80 0.62
n=inf 0.92 0.81 0.66
```

5.9 Exemplo 5.7

Construção de intervalos de confiança simultâneos para amostras grandes.

```
> result.mat <- matrix(0, 7, 14)
> med <- c(28.1, 26.6, 35.4, 34.2, 23.6, 22, 22.7)
> d.p <- c(5.76, 5.85, 3.82, 5.12, 3.76, 3.93, 4.03)
> n <- 96
> LINF <- med - sqrt(qchisq(0.1, length(med), lower.tail = FALSE)/n) *
+     d.p
> LSUP <- med + sqrt(qchisq(0.1, length(med), lower.tail = FALSE)/n) *
+     d.p
```

Testar $H_0 : \mu = c(31, 27, 34, 31, 23, 22, 22)$

```
> mu0 <- c(31, 27, 34, 31, 23, 22, 22)
> mu0 < LINF

[1] FALSE FALSE  TRUE  TRUE FALSE FALSE FALSE

> mu0 > LSUP

[1]  TRUE FALSE FALSE FALSE FALSE FALSE FALSE
```

5.10 Tabelas 5.5; 5.6 e 5.7

```
> for (i in 1:7) {
+     result.mat[i, 3] <- med[i] - qnorm(0.975) * d.p[i]/sqrt(96)
+     result.mat[i, 4] <- med[i] + qnorm(0.975) * d.p[i]/sqrt(96)
+     result.mat[i, 5] <- med[i] - qnorm(1 - 0.025/7) *
+         d.p[i]/sqrt(96)
+     result.mat[i, 6] <- med[i] + qnorm(1 - 0.025/7) *
+         d.p[i]/sqrt(96)
+     result.mat[i, 7] <- med[i] - sqrt(qchisq(0.95,
+         7)) * d.p[i]/sqrt(96)
+     result.mat[i, 8] <- med[i] + sqrt(qchisq(0.95,
+         7)) * d.p[i]/sqrt(96)
```

```
+     result.mat[i, 9] <- med[i] - qt(0.975, 95) *
+         d.p[i]/sqrt(96)
+     result.mat[i, 10] <- med[i] + qt(0.975, 95) *
+         d.p[i]/sqrt(96)
+     result.mat[i, 11] <- med[i] - qt(1 - 0.025/7,
+         95) * d.p[i]/sqrt(96)
+     result.mat[i, 12] <- med[i] + qt(1 - 0.025/7,
+         95) * d.p[i]/sqrt(96)
+     result.mat[i, 13] <- med[i] - sqrt((7 * 95/89) *
+         qf(0.95, 7, 89)) * d.p[i]/sqrt(96)
+     result.mat[i, 14] <- med[i] + sqrt((7 * 95/89) *
+         qf(0.95, 7, 89)) * d.p[i]/sqrt(96)
+ }
> nomes.linhas <- c("melodia", "harmonia", "tempo",
+     "metro", "frase", "equlibrio", "estilo")
> nomes.colunas <- c("medias", "dp", "liminf.indg",
+     "limsup.indg", "liminf.bonfg", "limsup.bonfg",
+     "liminf.simug", "limsup.simug", "liminf.indp",
+     "limsup.indp", "liminf.bonfp", "limsup.bonfp",
+     "liminf.simup", "limsup.simup")
> dimnames(result.mat) <- list(nomes.linhas, nomes.colunas)
> round(result.mat, 2)
```

```
          medias dp liminf.indg limsup.indg liminf.bonfg
melodia        0  0        26.9        29.2         26.5
harmonia       0  0        25.4        27.8         25.0
tempo          0  0        34.6        36.2         34.4
metro          0  0        33.2        35.2         32.8
frase          0  0        22.9        24.4         22.6
equlibrio      0  0        21.2        22.8         20.9
estilo         0  0        21.9        23.5         21.6
          limsup.bonfg liminf.simug limsup.simug liminf.indp
melodia           29.7         25.9         30.3        26.9
harmonia          28.2         24.4         28.8        25.4
tempo             36.5         33.9         36.9        34.6
metro             35.6         32.2         36.2        33.2
```

```
frase             24.6         22.2         25.0        22.8
equlibrio         23.1         20.5         23.5        21.2
estilo            23.8         21.2         24.2        21.9
          limsup.indp liminf.bonfp limsup.bonfp liminf.simup
melodia          29.3         26.5         29.7         25.8
harmonia         27.8         25.0         28.2         24.2
tempo            36.2         34.3         36.5         33.9
metro            35.2         32.8         35.6         32.1
frase            24.4         22.5         24.7         22.1
equlibrio        22.8         20.9         23.1         20.4
estilo           23.5         21.6         23.8         21.1
          limsup.simup
melodia           30.4
harmonia          29.0
tempo             37.0
metro             36.3
frase             25.1
equlibrio         23.6
estilo            24.3
```

5.11 Exemplo 5.13

Ilustração do Algoritmo EM

```
> X <- matrix(c(NA, 7, 5, NA, 0, 2, 1, NA, 3, 6, 2,
+     5), 4, 3)
> n <- nrow(X)
> p <- ncol(X)
```

Médias das variáveis calculadas com valores observados:

```
> mu1.til <- mean(X[, 1], na.rm = TRUE)
> mu2.til <- mean(X[, 2], na.rm = TRUE)
> mu3.til <- mean(X[, 3], na.rm = TRUE)
```

Imputar pela média:

```
> X[1, 1] <- mu1.til
> X[4, 1] <- mu1.til
> X[4, 2] <- mu2.til
```

Estimar parametros:

```
> sigma11.til <- var(X[, 1]) * (n - 1)/n
> sigma22.til <- var(X[, 2]) * (n - 1)/n
> sigma33.til <- var(X[, 3]) * (n - 1)/n
> sigma12.til <- cov(X[, 1], X[, 2]) * (n - 1)/n
> sigma13.til <- cov(X[, 1], X[, 3]) * (n - 1)/n
> sigma23.til <- cov(X[, 2], X[, 3]) * (n - 1)/n
> S <- cov(X) * (n - 1)/n
> S

     [,1] [,2] [,3]
[1,] 0.50 0.25 1.00
[2,] 0.25 0.50 0.75
[3,] 1.00 0.75 2.50
```

Passo de predição: Usar as estimativas iniciais para predizer as contribuições dos valores que faltantes nas estatísticas suficientes.
. Vamos utilizar diretamente a library norm do R para implementar esses passos e obter o EMV do vetor de médias e da matriz de covariância.

```
> library(norm)
> s <- prelim.norm(X)
> thetahat <- em.norm(s)

Iterations of EM:
1...2...

> getparam.norm(s, thetahat)

$mu
[1] 6 1 4
```

```
$sigma
     [,1] [,2] [,3]
[1,] 0.50 0.25 1.00
[2,] 0.25 0.50 0.75
[3,] 1.00 0.75 2.50

> Simul.Int.R <- function(a, object, alfa = 0.05, large.samp = FALSE) {
+     df.num <- anova(object, test = "Wilks")$num.DF[1]
+     df.den <- anova(object, test = "Wilks")$den.DF[1]
+     xbar <- S <- cov(data)
+     a <- matrix(a, ncol = 1)
+     xbar <- matrix(xbar, ncol = 1)
+     var.lin <- t(a) %*% cov(data) %*% a
+     if (large.samp) {
+         liminf <- t(a) %*% xbar - sqrt(qchisq(alfa,
+             p, lower.tail = FALSE) * sqrt(var.lin/n))
+         limsup <- t(a) %*% xbar + sqrt(qchisq(alfa,
+             p, lower.tail = FALSE) * sqrt(var.lin/n))
+         return(c(liminf, limsup))
+     }
+     cte <- (p * (n - 1))/(n * (n - p))
+     comp <- sqrt(cte * var.lin * qf(alfa, p, (n -
+         p), lower.tail = FALSE))
+     liminf <- t(a) %*% xbar - comp
+     limsup <- t(a) %*% xbar + comp
+     c(liminf, limsup)
+ }
```

Capítulo 6

Comparação de várias médias

6.1 Exemplo 6.1

Verificação da diferença de médias com observações emparelhadas.

```
> tab6.1 <- read.table("t6-1.dat", col.names = c("COM.BOD",
+     "COM.SS", "EST.BOD", "EST.SS"))
```

Testar se há diferença entre labs $H_0 : \mu_1 = \mu_2$:

```
> tab6.1 <- transform(tab6.1, D.BOD = COM.BOD - EST.BOD,
+     D.SS = COM.SS - EST.SS)
> n <- nrow(tab6.1[, c("D.BOD", "D.SS")])
> p <- ncol(tab6.1[, c("D.BOD", "D.SS")])
> dbar <- matrix(colMeans(tab6.1[, c("D.BOD", "D.SS")]),
+     ncol = 1)
> Sd <- cov(tab6.1[, c("D.BOD", "D.SS")])
```

Calcular a estatística T^2:

```
> T2 <- n * t(dbar) %*% solve(Sd) %*% dbar
> alfa <- 0.05
> val.crit <- (p * (n - 1)/(n - p)) * qf(0.05, p, n -
+     p, lower.tail = FALSE)
> T2 > val.crit
```

```
     [,1]
[1,] TRUE
```

Logo rejeitamos a hipótese de igualdade de médias entre os laboratórios. Uma forma alternativa é usa função `T2.Hotteling`:

```
> T2.Hotteling(c(0, 0), tab6.1[, c("D.BOD", "D.SS")])
```

```
$estat
[1] 13.6

$p.value
[1] 0.0208
```

O teste poderia ser feito diretamente através do R:

```
> tab6.1.lm <- lm(cbind(D.BOD, D.SS) ~ 1, data = tab6.1)
> anova(tab6.1.lm, test = "Wilks")
```

```
Analysis of Variance Table

            Df Wilks approx F num Df den Df Pr(>F)
(Intercept)  1  0.42     6.14      2      9  0.021 *
Residuals   10
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

Vamos agora obter intervalos de confiança simultâneos para as diferenças de médias das variáveis componentes:

```
> Simul.Int(c(1, 0), tab6.1[, c("D.BOD", "D.SS")])
```

```
[1] -22.45   3.73
```

```
> Simul.Int(c(0, 1), tab6.1[, c("D.BOD", "D.SS")])
```

```
[1] -5.7 32.2
```

Observe que o teste rejeita e todos os intervalos simultãneos contêm 0. Esses resultados não são contraditórios. Deve haver intervalo de confiança para alguma combinação linear das diferenças de médias que não contém 0.

6.2 Exemplo 6.2

Testar a igualdade de tratamentos num desenho de medidas repetidas:

```
> tab6.2 <- read.table("t6-2.dat", col.names = c("TRAT1",
+     "TRAT2", "TRAT3", "TRAT4"))
> n <- nrow(tab6.2)
> q1 <- ncol(tab6.2)
```

Vamos considerar os seguintes contrastes de interesse:$(T_3 + T_4) - (T_1 + T_2)$; $(T_1 + T_3) - (T_2 + T_4)$ e $(T1 + T4) - (T2 + T3)$. Para isso, vamos formar a matriz **C** de contrastes:

```
> C1 <- matrix(c(-1, -1, 1, 1, 1, -1, 1, -1, 1, -1,
+     -1, 1), 3, 4, byrow = TRUE)
> xbar <- matrix(colMeans(tab6.2), ncol = 1)
> S <- cov(tab6.2)
> C1 %*% xbar
```

```
      [,1]
[1,] 209.3
[2,] -60.1
[3,] -12.8
```

```
> C1 %*% S %*% t(C1)
```

```
     [,1] [,2] [,3]
[1,] 9432 1099  928
[2,] 1099 5196  915
[3,]  928  915 7557
```

Vamos calcular estatística T^2 de Hotteling e compará-la ao valor crítíco do teste no nivel $\alpha = 5\%$:

```
> T2 <- n * t(C1 %*% xbar) %*% solve(C1 %*% S %*% t(C1)) %*%
+     (C1 %*% xbar)
> val.crit <- (((n - 1) * (q1 - 1))/(n - q1 + 1)) *
+     qf(0.05, q1 - 1, n - q1 + 1, lower.tail = FALSE)
> T2 > val.crit
```

```
     [,1]
[1,] TRUE
```

O teste rejeita a hipótese $H_0 : \mathbf{C}\mu = \mathbf{0}$. Alternativamente, podemos efetuar o teste diretamente através da função `anova` do `R`, tranformando os dados:

```
> tab6.2.C <- as.matrix(tab6.2) %*% t(C1)
> anova(lm(tab6.2.C ~ 1))

Analysis of Variance Table

            Df Pillai approx F num Df den Df  Pr(>F)
(Intercept)  1    0.9     34.4      3     16 3.3e-07 ***
Residuals   18
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

Intervalos de confiança simultâneos para as componentes da média do efeito $(T_3 + T_4) - (T_1 + T_2)$:

```
> c1 <- matrix(C1[1, ], ncol = 1)
> t(c1) %*% xbar - sqrt(val.crit) * sqrt(t(c1) %*%
+     S %*% c1/n)

     [,1]
[1,]  136

> t(c1) %*% xbar + sqrt(val.crit) * sqrt(t(c1) %*%
+     S %*% c1/n)

     [,1]
[1,]  283
```

Para o efeito $(T_1 + T_3) - (T_2 + T_4)$:

```
> c2 <- matrix(C1[2, ], ncol = 1)
> t(c2) %*% xbar - sqrt(val.crit) * sqrt(t(c2) %*%
+     S %*% c2/n)
```

```
     [,1]
[1,] -115
```

```
> t(c2) %*% xbar + sqrt(val.crit) * sqrt(t(c2) %*%
+     S %*% c2/n)
```

```
      [,1]
[1,] -5.38
```

Para o efeito $(T_1 + T_4) - (T_2 + T_3)$:

```
> c3 <- matrix(C1[3, ], ncol = 1)
> t(c3) %*% xbar - sqrt(val.crit) * sqrt(t(c3) %*%
+     S %*% c3/n)
```

```
      [,1]
[1,] -78.7
```

```
> t(c3) %*% xbar + sqrt(val.crit) * sqrt(t(c3) %*%
+     S %*% c3/n)
```

```
     [,1]
[1,] 53.1
```

O Primeiro I.C. indica que há um efeito do Halotano: $(T_3 + T_4) - (T_1 + T_2)$

6.3 Exemplo 6.3

Construção de uma região de confiança para a diferença de médias de dois vetores.

```
> xbar1 <- matrix(c(8.3, 4.1), ncol = 1)
> S1 <- matrix(c(2, 1, 1, 6), 2, 2)
> xbar2 <- matrix(c(10.2, 3.9), ncol = 1)
> S2 <- matrix(c(2, 1, 1, 4), 2, 2)
> n1 <- n2 <- 50
> p <- 2
```

```
> Spool <- ((n1 - 1)/(n1 + n2 - 2)) * S1 + ((n2 - 1)/(n1 +
+     n2 - 2)) * S2
> centro <- as.vector(xbar1 - xbar2)
> tam <- sqrt((1/n1 + 1/n2) * ((n1 + n2 - 2) * p/(n1 +
+     n2 - p - 1)) * qf(0.05, p, n1 + n2 - p - 1, lower.tail = FALSE))
```

Tamanhos dos semi-eixos:

```
> AU <- eigen(Spool)
> P1 <- centro + (sqrt(AU$values[1]) * tam) * AU$vectors[,
+     1]
> P2 <- centro - (sqrt(AU$values[1]) * tam) * AU$vectors[,
+     1]
> P1 <- centro + (sqrt(AU$values[1]) * tam) * AU$vectors[,
+     1]
> P2 <- centro - (sqrt(AU$values[1]) * tam) * AU$vectors[,
+     1]
> Q1 <- centro + (sqrt(AU$values[2]) * tam) * AU$vectors[,
+     2]
> Q2 <- centro - (sqrt(AU$values[2]) * tam) * AU$vectors[,
+     2]
```

```
> library(ellipse)
> plot(ellipse(Spool, centre = centro, t = tam), type = "l",
+     xlim = c(-3, -1), ylim = c(-1, 1.5))
> axis(1, at = -3:1, labels = as.character(-3:1), pos = 0)
> axis(2, pos = 0)
> segments(P1[1], P1[2], P2[1], P2[2])
> segments(Q1[1], Q1[2], Q2[1], Q2[2])
```

6.4 Exemplo 6.4

Cálculo de intervalos de confiança simultâneos para os componentes da diferença de médias.

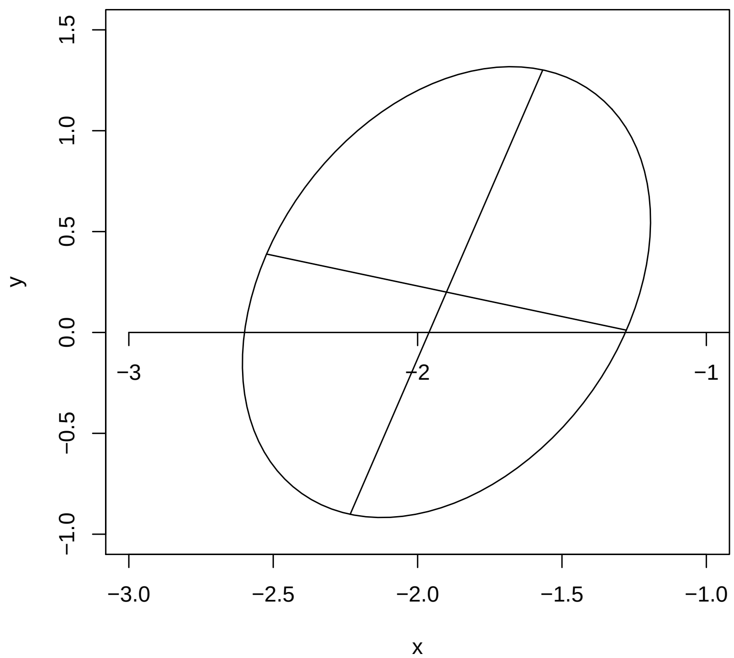


Figura 6.1: Elipse de confiança de 95% para $\mu_1 - \mu_2$

```
> xbar1 <- matrix(c(204.4, 556.6), ncol = 1)
> S1 <- matrix(c(13825.3, 23823.4, 23823.4, 73107.4),
+     2, 2)
> xbar2 <- matrix(c(130, 355), ncol = 1)
> S2 <- matrix(c(8632, 19616.7, 19616.7, 55964.5),
+     2, 2)
> n1 <- 45
> n2 <- 55
> p <- 2
> Spool <- ((n1 - 1)/(n1 + n2 - 2)) * S1 + ((n2 - 1)/(n1 +
+     n2 - 2)) * S2
> c2 <- ((n1 + n2 - 2) * p/(n1 + n2 - p - 1)) * qf(0.05,
+     p, n1 + n2 - p - 1, lower.tail = F)
```

Intervalos de confiança simutâneos para as diferenças das primeiras componentes do vetor de médias:

```
> xbar1[1, ] - xbar2[1, ] - sqrt(c2) * sqrt((1/n1 +
+     1/n2) * Spool[1, 1])

[1] 21.8

> xbar1[1, ] - xbar2[1, ] + sqrt(c2) * sqrt((1/n1 +
+     1/n2) * Spool[1, 1])

[1] 127
```

Intervalos de confiança simutâneos para as diferenças das segundas componentes do vetor de médias:

```
> xbar1[2, ] - xbar2[2, ] - sqrt(c2) * sqrt((1/n1 +
+     1/n2) * Spool[2, 2])

[1] 74.9

> xbar1[2, ] - xbar2[2, ] + sqrt(c2) * sqrt((1/n1 +
+     1/n2) * Spool[2, 2])
```

```
[1] 328
```

Vamos verificar se $c(0,0)$ está dentro da região de confiança usando a fórmula 6-24 do livro texto:

```
> t(xbar1 - xbar2) %*% solve((1/n1 + 1/n2) * Spool) %*%
+     (xbar1 - xbar2) <= c2

      [,1]
[1,] FALSE
```

Logo, rejeitamos a hipótese de igualdade de vetor de médias para o nível $\alpha = 5\%$. Vamos agora obter intervalos de confiança simultâneos pelo método de Bonferroni.

Para a diferença das primeiras componentes do vetor de médias:

```
> xbar1[1, ] - xbar2[1, ] - qt(0.05/(2 * p), n1 + n2 -
+     2, lower.tail = F) * sqrt((1/n1 + 1/n2) * Spool[1,
+     1])

[1] 26.5

> xbar1[1, ] - xbar2[1, ] + qt(0.05/(2 * p), n1 + n2 -
+     2, lower.tail = F) * sqrt((1/n1 + 1/n2) * Spool[1,
+     1])

[1] 122
```

Para a diferença das segundas componentes do vetor de médias:

```
> xbar1[2, ] - xbar2[2, ] - qt(0.05/(2 * p), n1 + n2 -
+     2, lower.tail = F) * sqrt((1/n1 + 1/n2) * Spool[2,
+     2])

[1] 86.2

> xbar1[2, ] - xbar2[2, ] + qt(0.05/(2 * p), n1 + n2 -
+     2, lower.tail = F) * sqrt((1/n1 + 1/n2) * Spool[2,
+     2])

[1] 317
```

6.5 Exemplo 6.5

Procedimentos de amostras grandes para inferências sobre a diferença de médias. Vamos agora utilizar procedimentos para amostras grandes decritos no Resultado 6.4 do livro texto:

```
> S <- (1/n1) * S1 + (1/n2) * S2
> a1 <- as.matrix(c(1, 0))
> a2 <- as.matrix(c(0, 1))
```

Intervalo de confiança para a primeira componente:

```
> t(a1) %*% (xbar1 - xbar2) - sqrt(qchisq(0.05, p,
+     lower.tail = F)) * sqrt(t(a1) %*% S %*% a1)

     [,1]
[1,] 21.7

> t(a1) %*% (xbar1 - xbar2) + sqrt(qchisq(0.05, p,
+     lower.tail = F)) * sqrt(t(a1) %*% S %*% a1)

     [,1]
[1,]  127
```

Intervalo de confiança para a segunda componente:

```
> t(a2) %*% (xbar1 - xbar2) - sqrt(qchisq(0.05, p,
+     lower.tail = F)) * sqrt(t(a2) %*% S %*% a2)

     [,1]
[1,] 75.8

> t(a2) %*% (xbar1 - xbar2) + sqrt(qchisq(0.05, p,
+     lower.tail = F)) * sqrt(t(a2) %*% S %*% a2)

     [,1]
[1,]  327
```

Estatistica T^2 de Hotelling para testar $H_0 : \mu_\mathbf{1} - \mu_\mathbf{2} = \mathbf{0}$:

```
> T2 <- t(xbar1 - xbar2) %*% solve(S) %*% (xbar1 -
+     xbar2)
```

Valor crítico aproximado no nível $\alpha = 5\%$:

```
> val.crit <- qchisq(0.05, p, lower.tail = F)
> T2 > val.crit
```

```
     [,1]
[1,] TRUE
```

Logo rejeitamos H_0

6.6 Exemplo 6.6

Decomposiçao de soma de quadrados para a ANOVA univariada:

```
> pop1 <- c(9, 6, 9)
> pop2 <- c(0, 2)
> pop3 <- c(3, 1, 2)
> n1 <- 3
> n2 <- 2
> n3 <- 3
> n <- n1 + n2 + n3
> ng <- 3
> grupo <- as.factor(c(rep(1, n1), rep(2, n2), rep(3,
+     n3)))
> resp <- c(9, 6, 9, 0, 2, 3, 1, 2)
> med.geral <- rep(mean(resp), n)
> med.trat <- rep(tapply(resp, grupo, mean), c(n1,
+     n2, n3))
> trat <- med.trat - med.geral
> res <- resp - med.trat
```

Verificação da identidade de soma de quadrados:

```
> SQG <- t(resp) %*% resp
> SQM <- t(med.geral) %*% med.geral
```

```
> SQT <- t(trat) %*% trat
> SQR <- t(res) %*% res
> SQG == SQM + SQT + SQR
```

```
     [,1]
[1,] TRUE
```

6.7 Exemplo 6.7

Uma tabela de ANOVA univariada e o teste F para efeitos de tratamentos.

```
> ANOVA <- matrix(NA, 3, 2)
> ANOVA[1, 1] <- SQT
> ANOVA[2, 1] <- SQR
> ANOVA[1, 2] <- ng - 1
> ANOVA[2, 2] <- n1 + n2 + n3 - ng
> ANOVA[3, 1] <- SQG - SQM
> ANOVA[3, 2] <- length(resp) - 1
> dimnames(ANOVA) <- list(c("Trat.", "Res.", "Total"),
+     c("SQ", "GL"))
> ANOVA
```

```
      SQ GL
Trat. 78  2
Res.  10  5
Total 88  7
```

Estatística de teste:

```
> F.val <- (ANOVA[1, 1]/ANOVA[1, 2])/(ANOVA[2, 1]/ANOVA[2,
+     2])
> val.crit <- qf(0.01, ANOVA[1, 2], ANOVA[2, 2], lower.tail = FALSE)
> F.val > val.crit
```

```
[1] TRUE
```

Alternativa: usar função `aov` do R para Análise de Variância:

```
> ex.6.6 <- data.frame(grupo, resp)
> summary(aov(resp ~ grupo, ex.6.6))

            Df Sum Sq Mean Sq F value Pr(>F)
grupo        2     78      39    19.5 0.0044 **
Residuals    5     10       2
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

6.8 Exemplo 6.8

Uma tabela Manova e lambda de Wilks para testar igualdade de três vetores de médias.

```
> n1 <- 3
> n2 <- 2
> n3 <- 3
> n <- n1 + n2 + n3
> ng <- 3
> grupo <- as.factor(c(rep(1, n1), rep(2, n2), rep(3,
+     n3)))
> resp.1 <- c(9, 6, 9, 0, 2, 3, 1, 2)
> resp.2 <- c(3, 2, 7, 4, 0, 8, 9, 7)
```

Decomposicão para a primeira variável (repete Exemplo 6.7):

```
> med.geral.1 <- rep(mean(resp.1), n)
> med.trat.1 <- rep(tapply(resp.1, grupo, mean), c(n1,
+     n2, n3))
> trat.1 <- med.trat.1 - med.geral.1
> res.1 <- resp.1 - med.trat.1
> SQG.1 <- t(resp.1) %*% resp.1
> SQM.1 <- t(med.geral.1) %*% med.geral.1
> SQT.1 <- t(trat.1) %*% trat.1
> SQR.1 <- t(res.1) %*% res.1
> SQG.1 == SQM.1 + SQT.1 + SQR.1
```

```
     [,1]
[1,] TRUE
```

Decomposicão para a segunda variável:

```
> med.geral.2 <- rep(mean(resp.2), n)
> med.trat.2 <- rep(tapply(resp.2, grupo, mean), c(n1,
+     n2, n3))
> trat.2 <- med.trat.2 - med.geral.2
> res.2 <- resp.2 - med.trat.2
> SQG.2 <- t(resp.2) %*% resp.2
> SQM.2 <- t(med.geral.2) %*% med.geral.2
> SQT.2 <- t(trat.2) %*% trat.2
> SQR.2 <- t(res.2) %*% res.2
> SQG.2 == SQM.2 + SQT.2 + SQR.2
```

```
     [,1]
[1,] TRUE
```

Somas de produtos cruzados:

```
> SPCG.12 <- t(resp.1) %*% resp.2
> SPCM.12 <- t(med.geral.1) %*% med.geral.2
> SPCT.12 <- t(trat.1) %*% trat.2
> SPCR.12 <- t(res.1) %*% res.2
> SPCG.12 == SPCM.12 + SPCT.12 + SPCR.12
```

```
     [,1]
[1,] TRUE
```

Matrizes de somas de quadrados e de produtos cruzados:

SQ de Tratamento:

```
> B <- matrix(c(SQT.1, SPCT.12, SPCT.12, SQT.2), 2,
+     2)
> gl.trt <- ng - 1
```

SQ Residual:

```
> W <- matrix(c(SQR.1, SPCR.12, SPCR.12, SQR.2), 2,
+     2)
> gl.res <- n1 + n2 + n3 - 3
```

SQ Total corrigida:

```
> MSQPRG <- matrix(c(SQG.1 - SQM.1, SPCG.12 - SPCM.12,
+     SPCG.12 - SPCM.12, SQG.2 - SQM.2), 2, 2)
> gl.tot <- n - 1
> all.equal(MSQPRG, B + W)

[1] TRUE
```

Cálculo do lambda de Wilks para testar igualdade de médias nos 3 grupos:

```
> L <- det(W)/det(B + W)
> val.est <- ((1 - sqrt(L))/sqrt(L)) * (n - ng - 1)/(ng -
+     1)
> val.crit <- qf(0.01, 2 * (ng - 1), 2 * (n - ng -
+     1), lower.tail = FALSE)
> val.est > val.crit

[1] TRUE
```

Pode ser feito diretamente através R:

```
> exe6.8.manova <- manova(cbind(resp.1, resp.2) ~ grupo)
> summary(exe6.8.manova, test = "Wilks")

          Df Wilks approx F num Df den Df Pr(>F)
grupo      2  0.04     8.20      4      8 0.0062 **
Residuals  5
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

6.9 Exemplo 6.9

Uma análise multivariada dos dados de creches de Wisconsin.

```
> n1 <- 271
> n2 <- 138
> n3 <- 107
> n <- n1 + n2 + n3
> g <- 3
> p <- 4
> xbar1 <- matrix(c(2.066, 0.48, 0.082, 0.36), ncol = 1)
> xbar2 <- matrix(c(2.167, 0.596, 0.124, 0.418), ncol = 1)
> xbar3 <- matrix(c(2.273, 0.521, 0.125, 0.383), ncol = 1)
```

Entrada de matriz simétrica

```
> mat.sim <- function(x, n) {
+     A <- matrix(0, n, n)
+     A[row(A) >= col(A)] <- x
+     A + t(A) - diag(diag(A))
+ }
> S1 <- mat.sim(c(0.291, -0.001, 0.002, 0.01, 0.011,
+     0, 0.003, 0.001, 0, 0.01), 4)
> S2 <- mat.sim(c(0.561, 0.011, 0.001, 0.037, 0.025,
+     0.004, 0.007, 0.005, 0.002, 0.019), 4)
> S3 <- mat.sim(c(0.261, 0.03, 0.003, 0.018, 0.017,
+     -0, 0.006, 0.004, 0.001, 0.013), 4)
> W <- (n1 - 1) * S1 + (n2 - 1) * S2 + (n3 - 1) * S3
> xbar <- (n1 * xbar1 + n2 * xbar2 + n3 * xbar3)/(n1 +
+     n2 + n3)
> B <- n1 * (xbar1 - xbar) %*% t(xbar1 - xbar) + n2 *
+     (xbar2 - xbar) %*% t(xbar2 - xbar) + n3 * (xbar3 -
+     xbar) %*% t(xbar3 - xbar)
```

Teste exato

```
> L <- det(W)/det(B + W)
> est.val <- ((n - p - 2)/p) * ((1 - sqrt(L))/sqrt(L))
> val.crit <- qf(0.01, 2 * p, 2 * (n - p - 2), lower.tail = FALSE)
```

Teste para grandes amostras:

```
> est.teste1 <- -(n - 1 - (p + g)/2) * log(det(W)/det(B +
+     W))
> val.crit1 <- qchisq(0.01, p * (g - 1), lower.tail = FALSE)
> est.teste1 > val.crit1
```

```
[1] TRUE
```

6.10 Exemplo 6.10

Intervalos simultâneos para diferenças de tratamentos- dados de creches

```
> tal1.hat <- as.vector(xbar1 - xbar)
> tal3.hat <- as.vector(xbar3 - xbar)
```

Intervalos Simultâneos de 95%:
Grupo 1 vs Grupo 3, variável 3:

```
> alfa <- 0.05
> tal1.hat[3] - tal3.hat[3] - qt(alfa/(p * g * (g -
+     1)), n - g, lower.tail = FALSE) * sqrt((W[3,
+     3]/(n - g)) * (1/n1 + 1/n3))
```

```
[1] -0.06
```

```
> tal1.hat[3] - tal3.hat[3] + qt(alfa/(p * g * (g -
+     1)), n - g, lower.tail = FALSE) * sqrt((W[3,
+     3]/(n - g)) * (1/n1 + 1/n3))
```

```
[1] -0.0260
```

Grupo 1 vs Grupo 2, variável 3:

```
> tal1.hat <- as.vector(xbar1 - xbar)
> tal2.hat <- as.vector(xbar2 - xbar)
> tal1.hat[3] - tal2.hat[3] - qt(alfa/(p * g * (g -
+     1)), n - g, lower.tail = FALSE) * sqrt((W[3,
+     3]/(n - g)) * (1/n1 + 1/n3))
```

```
[1] -0.059
```

```
> tal1.hat[3] - tal2.hat[3] + qt(alfa/(p * g * (g -
+     1)), n - g, lower.tail = FALSE) * sqrt((W[3,
+     3]/(n - g)) * (1/n1 + 1/n3))
```

```
[1] -0.0250
```

Grupo 2 vs grupo 3, variável 3:

```
> tal2.hat <- as.vector(xbar2 - xbar)
> tal3.hat <- as.vector(xbar3 - xbar)
> tal2.hat[3] - tal3.hat[3] - qt(alfa/(p * g * (g -
+     1)), n - g, lower.tail = FALSE) * sqrt((W[3,
+     3]/(n - g)) * (1/n1 + 1/n3))
```

```
[1] -0.0180
```

```
> tal2.hat[3] - tal3.hat[3] + qt(alfa/(p * g * (g -
+     1)), n - g, lower.tail = FALSE) * sqrt((W[3,
+     3]/(n - g)) * (1/n1 + 1/n3))
```

```
[1] 0.0160
```

6.11 Exemplo 6.11

Uma análise de variância multivariada de dois fatores de dados de filmes de plástico.

```
> tear <- c(6.5, 6.2, 5.8, 6.5, 6.5, 6.9, 7.2, 6.9,
+     6.1, 6.3, 6.7, 6.6, 7.2, 7.1, 6.8, 7.1, 7, 7.2,
+     7.5, 7.6)
> gloss <- c(9.5, 9.9, 9.6, 9.6, 9.2, 9.1, 10, 9.9,
+     9.5, 9.4, 9.1, 9.3, 8.3, 8.4, 8.5, 9.2, 8.8,
+     9.7, 10.1, 9.2)
> opacity <- c(4.4, 6.4, 3, 4.1, 0.8, 5.7, 2, 3.9,
+     1.9, 5.7, 2.8, 4.1, 3.8, 1.6, 3.4, 8.4, 5.2,
```

```
+     6.9, 2.7, 1.9)
> Y <- cbind(tear, gloss, opacity)
> rate <- factor(gl(2, 10), labels = c("Low", "High"))
> additive <- factor(gl(2, 5, len = 20), labels = c("Low",
+     "High"))
> fit <- manova(Y ~ rate * additive)
```

Tabela de ANOVA univariadas:

```
> summary.aov(fit)

 Response tear :
              Df Sum Sq Mean Sq F value Pr(>F)
rate           1  1.740   1.740    15.8 0.0011 **
additive       1  0.761   0.761     6.9 0.0183 *
rate:additive  1 0.0005  0.0005  0.0045 0.9471
Residuals     16  1.764   0.110
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

 Response gloss :
              Df Sum Sq Mean Sq F value Pr(>F)
rate           1  1.300   1.300    7.92  0.012 *
additive       1  0.613   0.613    3.73  0.071 .
rate:additive  1  0.544   0.544    3.32  0.087 .
Residuals     16  2.628   0.164
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

 Response opacity :
              Df Sum Sq Mean Sq F value Pr(>F)
rate           1    0.4     0.4    0.10   0.75
additive       1    4.9     4.9    1.21   0.29
rate:additive  1    4.0     4.0    0.98   0.34
Residuals     16   64.9     4.1
```

Tabela de ANOVA t de lambda de Wilks:

```
> summary.fit <- summary(fit, test = "Wilks")
```

Matrizes de somas de quadrados e de produtos cruzados

```
> summary.fit$SS

$rate
          tear gloss opacity
tear     1.740 -1.50   0.856
gloss   -1.504  1.30  -0.739
opacity  0.856 -0.74   0.421

$additive
         tear gloss opacity
tear    0.761 0.683    1.93
gloss   0.683 0.613    1.73
opacity 1.931 1.733    4.90

$`rate:additive`
          tear  gloss opacity
tear    0.0005 0.0165  0.0445
gloss   0.0165 0.5445  1.4685
opacity 0.0445 1.4685  3.9605

$Residuals
         tear  gloss opacity
tear     1.76  0.020  -3.070
gloss    0.02  2.628  -0.552
opacity -3.07 -0.552  64.924
```

Alguns testes:

```
> g <- 2
> b <- 2
> n <- 5
> p <- ncol(Y)
```

Teste de interação:

```
> SSP.int <- summary.fit$SS$"rate:additive"
> SSP.res <- summary.fit$SS$Residuals
> L.int <- det(SSP.res)/det(SSP.int + SSP.res)
> est.teste <- ((1 - L.int)/L.int) * ((g * b * (n -
+     1) - p + 1)/2)/((abs((g - 1) * (b - 1) - p) +
+     1)/2)
> nu1 <- (abs((g - 1) * (b - 1) - p)) + 1
> nu2 <- g * b * (n - 1) - p + 1
> valor.crit <- qf(0.05, nu1, nu2, lower.tail = FALSE)
> est.teste > valor.crit
```

```
[1] FALSE
```

Não rejeita efeito de nenhuma interação. Para outros teste ver p. 317 do livro. Estatísticas do Painel 6.1 na p.316:

```
> by(Y, rate, mean)
```

```
INDICES: Low
   tear   gloss opacity
   6.49    9.57    3.79
------------------------------------------------
INDICES: High
   tear   gloss opacity
   7.08    9.06    4.08
```

```
> by(Y, rate, sd)
```

```
INDICES: Low
   tear   gloss opacity
  0.420   0.298   1.854
------------------------------------------------
INDICES: High
   tear   gloss opacity
  0.322   0.576   2.182
```

```
> by(Y, additive, mean)
```

```
INDICES: Low
   tear   gloss opacity
   6.59    9.14    3.44
--------------------------------------------------
INDICES: High
   tear   gloss opacity
   6.98    9.49    4.43
```

```
> by(Y, additive, sd)
```

```
INDICES: Low
   tear   gloss opacity
  0.407   0.560   1.551
--------------------------------------------------
INDICES: High
   tear   gloss opacity
  0.473   0.428   2.301
```

6.12 Exemplo 6.12

Uma análise de perfil de dados de casamento e amor.

```
> p <- 4
> n1 <- n2 <- 30
> xbar1 <- matrix(c(6.833, 7.033, 3.967, 4.7), ncol = 1)
> xbar2 <- matrix(c(6.633, 7, 4, 4.533), ncol = 1)
> Spool <- mat.sim(c(0.606, 0.262, 0.066, 0.161, 0.637,
+     0.173, 0.143, 0.81, 0.029, 0.306), 4)
```

Testar paralelismo: $H_0 : \mathbf{C}\mu_{\mathbf{1}} = \mathbf{C}\mu_{\mathbf{2}}$

```
> C1 <- matrix(c(-1, 0, 0, 1, -1, 0, 0, 1, -1, 0, 0,
+     1), 3, 4)
```

Cálculo da estatística T^2 de teste:

```
> T2 <- t(xbar1 - xbar2) %*% t(C1) %*% solve((1/n1 +
+     1/n2) * C1 %*% Spool %*% t(C1)) %*% C1 %*% (xbar1 -
```

```
+     xbar2)
> alfa <- 0.05
> c2 <- ((n1 + n2 - 2) * (p - 1)/(n1 + n2 - p)) * qf(0.05,
+     p - 1, n1 + n2 - p, lower.tail = FALSE)
> T2 > c2

      [,1]
[1,] FALSE
```

Não rejeitamos a hipótese.

6.12.1 Figura 6.5

```
> plot(1:4, as.vector(xbar1), type = "n", xlab = "variável",
+     ylab = "média amostral")
> lines(1:4, as.vector(xbar1), type = "b", pch = 4,
+     lty = 1, col = "blue")
> lines(1:4, as.vector(xbar2), type = "b", pch = 1,
+     lty = 2, col = "red")
> legend("topleft", c("homens", "mulheres"), col = c("blue",
+     "red"), lty = c(1, 2), pch = c(4, 1))
```

6.13 Exemplo 6.13

Ajuste de uma curva quadrática de crescimento para a perda de cálcio.

```
> tab6.5.cont <- read.table("t6-5.dat")
> tab6.5.trat <- read.table("t6-6.dat")
> B <- matrix(c(1, 0, 0^2, 1, 1, 1^2, 1, 2, 2^2, 1,
+     3, 3^2), ncol = 3, byrow = TRUE)
> n1 <- nrow(tab6.5.cont)
> n2 <- nrow(tab6.5.trat)
> g <- 2
> q1 <- 2
> N <- n1 + n2
```

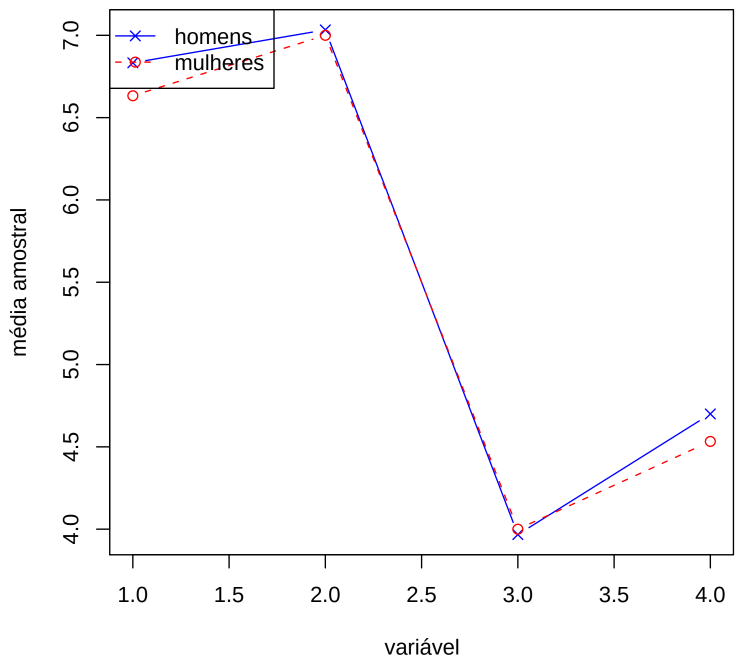


Figura 6.2: Perfís amostrais para respostas casamento-amor

```
> xbar1 <- as.matrix(colMeans(tab6.5.cont))
> xbar2 <- as.matrix(colMeans(tab6.5.trat))
> S1 <- cov(tab6.5.cont)
> S2 <- cov(tab6.5.trat)
> Spool <- (1/(N - g)) * ((n1 - 1) * S1 + (n2 - 1) *
+     S2)
> solve(t(B) %*% solve(Spool) %*% B)

       [,1]  [,2]   [,3]
[1,] 93.174 -5.84  0.218
[2,] -5.837  9.57 -3.024
[3,]  0.218 -3.02  1.105

> beta1 <- solve(t(B) %*% solve(Spool) %*% B) %*% t(B) %*%
+     solve(Spool) %*% xbar1
> beta2 <- solve(t(B) %*% solve(Spool) %*% B) %*% t(B) %*%
+     solve(Spool) %*% xbar2
> mat.coef <- cbind(beta1, beta2)
> k <- ((N - g) * (N - g - 1))/((N - g - p + q1) *
+     (N - g - p + q1 + 1))
> cov.beta1 <- (k/n1) * solve(t(B) %*% solve(Spool) %*%
+     B)
> cov.beta2 <- (k/n2) * solve(t(B) %*% solve(Spool) %*%
+     B)
> mat.coef <- cbind(beta1, sqrt(diag(cov.beta1)), beta2,
+     sqrt(diag(cov.beta2)))
> dimnames(mat.coef) <- list(c("int.", "t", "t^2"),
+     c("coef.cont", "dp.cont", "coef.trat", "dp.trat"))
> round(mat.coef, 2)

     coef.cont dp.cont coef.trat dp.trat
int.     73.07    2.58     70.14    2.50
t         3.64    0.83      4.09    0.80
t^2      -2.03    0.28     -1.85    0.27
```

Teste da nulidade de efeito quadrático:

```
> um1 <- matrix(1, nrow(tab6.5.cont), ncol = 1)
> um2 <- matrix(1, nrow(tab6.5.trat), ncol = 1)
> W2 <- t(as.matrix(tab6.5.cont) - um1 %*% t(B %*%
+     beta1)) %*% (as.matrix(tab6.5.cont) - um1 %*%
+     t(B %*% beta1)) + t(as.matrix(tab6.5.trat) -
+     um2 %*% t(B %*% beta2)) %*% (as.matrix(tab6.5.trat) -
+     um2 %*% t(B %*% beta2))
> W <- (N - g) * Spool
> L <- det(W)/det(W2)
> alfa <- 0.01
> val.est <- -(N - (1/2) * (p - q1 + g)) * log(L)
> val.crit <- qchisq(alfa, (p - q1 - 1) * g, lower.tail = FALSE)
> val.est > val.crit
```

```
[1] FALSE
```

Não rejeita a adequação de ajuste quadrático.

6.14 Exemplo 6.14

Comparação entre testes multivariados e univariados para a diferença de médias.

```
> x1 <- c(5, 4.5, 6, 6, 6.2, 6.9, 6.8, 5.3, 6.6, 7.3,
+     4.6, 4.9, 4, 3.8, 6.2, 5, 5.3, 7.1, 5.8, 6.8)
> x2 <- c(3, 3.2, 3.5, 4.6, 5.6, 5.2, 6, 5.5, 7.3,
+     6.5, 4.9, 5.9, 4.1, 5.4, 6.1, 7, 4.7, 6.6, 7.8,
+     8)
> grupo <- c(rep(1, 10), rep(2, 10))
```

Criar um gráfico de dispersão com gráficos marginais de pontos:

```
> nf <- layout(matrix(c(3, 1, 0, 2), 2, 2, byrow = TRUE),
+     c(1, 3), c(3, 1), TRUE)
> xrange <- c(min(x1), max(x1))
> yrange <- c(min(x2), max(x2))
```

```
> par(mar = c(3, 3, 1, 1))
> plot(x1, x2, xlim = xrange, ylim = yrange, type = "n",
+     xlab = "", ylab = "")
> points(x1[1:10], x2[1:10], pch = 1)
> points(x1[11:20], x2[11:20], pch = 4)
> par(mar = c(1, 3, 1, 1))
> stripchart(x1 ~ grupo, method = "stack", offset = 1/2,
+     pch = c(1, 4))
> par(mar = c(3, 0, 1, 1))
> stripchart(x2 ~ grupo, method = "stack", vertical = TRUE,
+     offset = 1/2, pch = c(1, 4))
```

6.15 Exemplo 6.15

Dados sobre lagartos que exigem teste bivariado para estabelecer um diferença nas médias. Estatísticas de resumo:

```
> tab6.7 <- read.table("t6-7.dat", col.names = c("Mass",
+     "SVL", "GRUPO"))
> tab6.7 <- transform(tab6.7, Mass = log(Mass), SVL = log(SVL))
```

```
> with(tab6.7, {
+     plot(SVL, Mass, type = "n", xlab = "ln(SVL)",
+         ylab = "ln(Mass)")
+     points(SVL[1:20], Mass[1:20], pch = 16)
+     points(SVL[21:60], Mass[21:60], pch = 1)
+     legend("topleft", c("C", "S"), pch = c(16, 1))
+ })
```

Intervalos de confiança individuais para as diferenças de médias, amostras grandes. Resultado 6.4 do livro texto:

```
> p <- 2
> n1 <- 20
> xbar1 <- matrix(c(2.24, 4.394), ncol = 1)
> S1 <- matrix(c(0.35305, 0.09417, 0.09417, 0.02595),
```

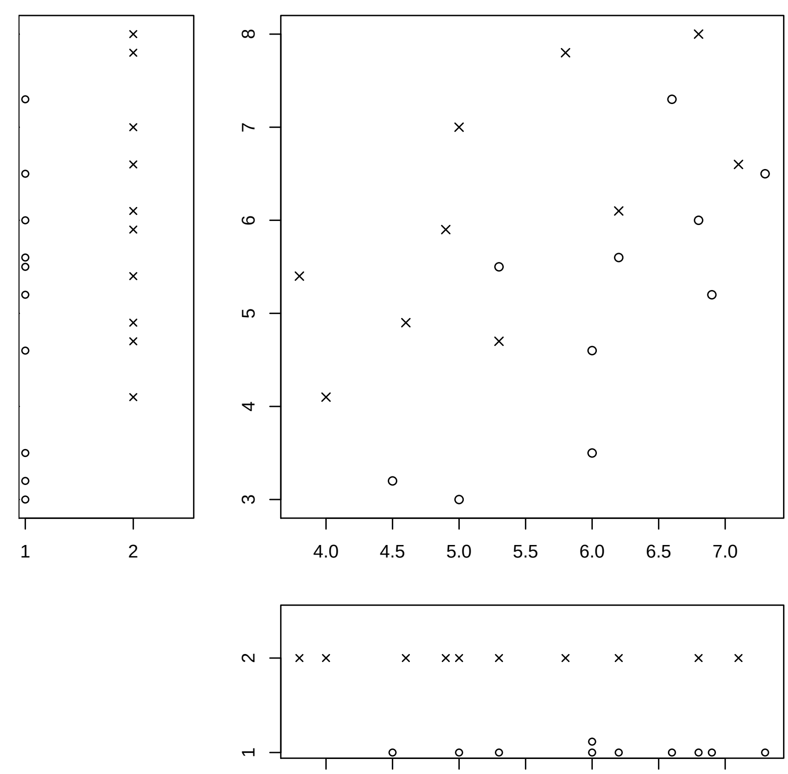


Figura 6.3: Pontos e marginais

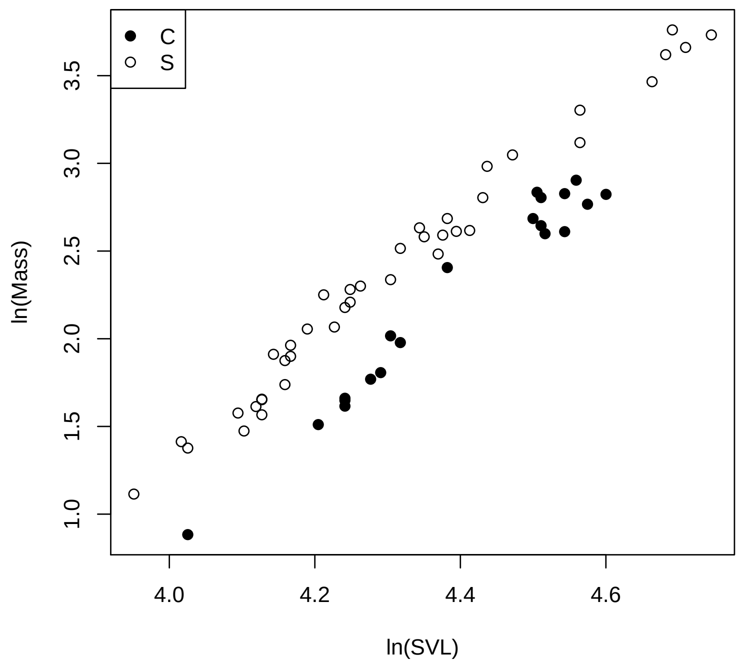


Figura 6.4: Figura 6.7

```
+     2, 2)
> n2 <- 40
> xbar2 <- matrix(c(2.368, 4.308), ncol = 1)
> S2 <- matrix(c(0.50684, 0.14539, 0.14539, 0.04255),
+     2, 2)
```

Intervalo de confiança t para a diferença de médias de ”Mass”nos 2 grupos:

```
> xbar1[1, 1] - xbar2[1, 1] - qt(0.05/2, n1 + n2 -
+     2, lower.tail = FALSE) * sqrt(S1[1, 1]/n1 + S2[1,
+     1]/n2)

[1] -0.477

> xbar1[1, 1] - xbar2[1, 1] + qt(0.05/2, n1 + n2 -
+     2, lower.tail = FALSE) * sqrt(S1[1, 1]/n1 + S2[1,
+     1]/n2)

[1] 0.221
```

Podemos usar diretamente o R:

```
> with(tab6.7, t.test(Mass[1:20], Mass[21:60]))

        Welch Two Sample t-test

data:  Mass[1:20] and Mass[21:60]
t = -0.736, df = 44.8, p-value = 0.4654
alternative hypothesis: true difference in means is not equal to 0
95 percent confidence interval:
 -0.479  0.223
sample estimates:
mean of x mean of y
     2.24      2.37
```

Intervalo de confiança t para diferença de médias de ”SVL”nos 2 grupos:

```
> xbar1[2, 1] - xbar2[2, 1] - qt(0.05/2, n1 + n2 -
+     2, lower.tail = FALSE) * sqrt(S1[2, 2]/n1 + S2[2,
+     2]/n2)
```

```
[1] -0.0113
```

```
> xbar1[2, 1] - xbar2[2, 1] + qt(0.05/2, n1 + n2 -
+     2, lower.tail = FALSE) * sqrt(S1[2, 2]/n1 + S2[2,
+     2]/n2)
```

```
[1] 0.183
```

Usando diretamente o R:

```
> with(tab6.7, t.test(SVL[1:20], SVL[21:60]))
```

```
        Welch Two Sample t-test

data:  SVL[1:20] and SVL[21:60]
t = 1.78, df = 47.4, p-value = 0.08205
alternative hypothesis: true difference in means is not equal to 0
95 percent confidence interval:
 -0.0114  0.1841
sample estimates:
mean of x mean of y
     4.39      4.31
```

Os dois intervalos contêm o 0. Pela Figura 6.7,a análise bivariada rejeita igualdade de médias. Vamos usar o resultado 6.4 do livro texto:

```
> T2 <- t(xbar1 - xbar2) %*% solve((S1/n1) + (S2/n2)) %*%
+     (xbar1 - xbar2)
> val.crit <- qchisq(0.05, p, lower.tail = FALSE)
> T2 > val.crit
```

```
     [,1]
[1,] TRUE
```

Logo, rejeitamos formente a hipótese nula.

Capítulo 7

Regressão linear multivariada

7.1 Exemplo 7.1

Ajuste de um modelo de regressão linear: $E(Y) = \beta_0 + \beta_1 z_1$.

```
> z1 <- c(0:4)
> y <- c(1, 4, 3, 8, 9)
> y <- matrix(y, ncol = 1)
> Z <- cbind(rep(1, length(z1)), z1)
```

7.2 Exemplo 7.2

A matriz de desenho para uma ANOVA de um fator como um modelo de regressão.
One-way ANOVA, 3 populações. Vamos definir as variáveis indicadoras das três populações z_1, z_2 e z_3, cada população com três observações:

```
> Y <- c(9, 6, 9, 0, 2, 3, 3, 1, 2)
> z0 <- rep(1, 9)
> z1 <- c(rep(c(1, 0), c(3, 6)))
> z2 <- c(rep(c(0, 1, 0), c(3, 3, 3)))
> z3 <- c(rep(c(0, 1), c(6, 3)))
> Y <- matrix(Y, ncol = 1)
> Z <- cbind(z0, z1, z2, z3)
```

7.3 Exemplo 7.3

Cálculo das estimativas de mínimos quadrados, resíduos e soma de quadrados dos resíduos.

```
> z1 <- c(0:4)
> y <- c(1, 4, 3, 8, 9)
> y <- matrix(y, ncol = 1)
> Z <- cbind(rep(1, length(z1)), z1)
> t(Z)

   [,1] [,2] [,3] [,4] [,5]
      1    1    1    1    1
z1    0    1    2    3    4

> y

     [,1]
[1,]    1
[2,]    4
[3,]    3
[4,]    8
[5,]    9

> t(Z) %*% Z

       z1
    5 10
z1 10 30

> solve(t(Z) %*% Z)

          z1
    0.6 -0.2
z1 -0.2  0.1

> t(Z) %*% y
```

```
    [,1]
      25
z1    70
```

```
> bhat <- solve(t(Z) %*% Z) %*% (t(Z) %*% y)
```

A equação ajustada é $\hat{y} = 1 + 2z$ e o vetor de valores ajustados é dado por:

```
> yhat <- Z %*% bhat
```

O vetor de resíduos e a soma de quadrados de resíduos são:

```
> ehat <- y - yhat
> t(ehat) %*% ehat
```

```
     [,1]
[1,]    6
```

Decomposição de soma de quadrados:

```
> (t(y) %*% y) == (t(yhat) %*% yhat + t(ehat) %*% ehat)
```

```
      [,1]
[1,] FALSE
```

A decomposicão para valores centrados na média:

```
> ybar <- matrix(rep(mean(y), length(y)), ncol = 1)
> t(y - ybar) %*% (y - ybar) == t(yhat - ybar) %*%
+     (yhat - ybar) + t(ehat) %*% ehat
```

```
      [,1]
[1,] FALSE
```

Cálculo de R^2:

```
> R2 <- 1 - (t(ehat) %*% ehat)/(t(y - ybar) %*% (y -
+     ybar))
```

Equivalente a:

```
> (t(yhat - ybar) %*% (yhat - ybar))/(t(y - ybar) %*%
+     (y - ybar))
```

```
     [,1]
[1,] 0.87
```

7.4 Exemplo 7.4

Ajuste de um modelo de regressão aos dados imobiliários.

```
> tab7.1 <- read.table("t7-1.dat", col.names = c("z1",
+     "z2", "Y"))
> Z <- cbind(rep(1, nrow(tab7.1)), as.matrix(tab7.1[,
+     1:2]))
> round(solve(t(Z) %*% Z), 4)

              z1      z2
    5.152  0.2544 -0.1463
z1  0.254  0.0512 -0.0172
z2 -0.146 -0.0172  0.0067

> bhat <- solve(t(Z) %*% Z) %*% (t(Z) %*% matrix(tab7.1[,
+     3], ncol = 1))
> yhat <- expression(z %*% bhat)
> z <- Z[1, , drop = F]
> eval(yhat)

     [,1]
[1,] 73.9
```

Usando a função lm do R, os resultados podem ser comparados com os do painel 7.1 do livro:

```
> tab7.1.lm <- lm(Y ~ z1 + z2, data = tab7.1)
> summary.lm(tab7.1.lm)

Call:
lm(formula = Y ~ z1 + z2, data = tab7.1)

Residuals:
    Min      1Q  Median      3Q     Max
-5.5894 -1.5411 -0.0718  1.3507  6.4605
```

```
Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  30.9666     7.8822    3.93   0.0011 **
z1            2.6344     0.7856    3.35   0.0038 **
z2            0.0452     0.2852    0.16   0.8760
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 3.47 on 17 degrees of freedom
Multiple R-Squared: 0.834,        Adjusted R-squared: 0.815
F-statistic: 42.8 on 2 and 17 DF,  p-value: 2.3e-07
```

```
> anova(tab7.1.lm)
```

```
Analysis of Variance Table

Response: Y
          Df Sum Sq Mean Sq F value  Pr(>F)
z1         1   1033    1033   85.63 4.8e-08 ***
z2         1    0.3     0.3    0.03    0.88
Residuals 17    205      12
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

Vamos calcular um intervalo de confiança de 95% para β_2:

```
> names(summary.lm(tab7.1.lm))
```

```
 [1] "call"          "terms"         "residuals"
 [4] "coefficients"  "aliased"       "sigma"
 [7] "df"            "r.squared"     "adj.r.squared"
[10] "fstatistic"    "cov.unscaled"
```

```
> cov.bhat <- (summary.lm(tab7.1.lm)$sigma^2) * summary.lm(tab7.1.lm)$cov.unscaled
> coef(tab7.1.lm)[3] - qt(0.025, summary(tab7.1.lm)$df[2],
+     lower.tail = FALSE) * sqrt(cov.bhat[3, 3])
```

```
    z2
-0.556
```

```
> coef(tab7.1.lm)[3] + qt(0.025, summary(tab7.1.lm)$df[2],
+     lower.tail = FALSE) * sqrt(cov.bhat[3, 3])
```

```
   z2
0.647
```

Como o intervalo inclue 0, podemos excluir a variável z_2 do modelo.

7.5 Exemplo 7.5

Teste da importância de preditores adicionais usando a abordagem da soma-extra de quadrados .

```
> tab7.2 <- read.table("t7-2.dat", col.names = c("Local",
+     "Sexo", "Y"))
> tab7.2 <- transform(tab7.2, Local = as.factor(Local),
+     Sexo = as.factor(Sexo))
> tab7.2.lm0 <- lm(Y ~ Local + Sexo + Local:Sexo, data = tab7.2)
> tab7.2.lm1 <- update(tab7.2.lm0, . ~ . - Local:Sexo)
> anova(tab7.2.lm1, tab7.2.lm0)
```

```
Analysis of Variance Table

Model 1: Y ~ Local + Sexo
Model 2: Y ~ Local + Sexo + Local:Sexo
  Res.Df  RSS Df Sum of Sq    F Pr(>F)
1     14 3419
2     12 2977  2       442 0.89   0.44
```

A interação entre Local e Sexo é não-significante. Vamos verificar a significância do efeito da variável Local:

```
> tab7.2.lm2 <- update(tab7.2.lm1, . ~ . - Local)
> anova(tab7.2.lm2, tab7.2.lm1)
```

```
Analysis of Variance Table

Model 1: Y ~ Sexo
Model 2: Y ~ Local + Sexo
  Res.Df  RSS Df Sum of Sq    F Pr(>F)
1     16 3666
2     14 3419  2       247 0.51   0.61
```

Concluímos que o fator Local não é significante. Vamos testar o efeito do fator sexo:

```
> tab7.2.lm3 <- update(tab7.2.lm1, . ~ . - Sexo)
> anova(tab7.2.lm3, tab7.2.lm1)
```

```
Analysis of Variance Table

Model 1: Y ~ Local
Model 2: Y ~ Local + Sexo
  Res.Df  RSS Df Sum of Sq    F Pr(>F)
1     15 7166
2     14 3419  1      3747 15.3 0.0015 **
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

Sexo é significante: Homens e mulheres não dão a mesma avaliação.

7.6 Exemplo 7.6

Estimativas de intervalos para uma resposta média e para uma resposta futura.

```
> tab7.3 <- read.table("t7-3.dat", col.names = c("z1",
+     "z2", "Y"))
> Z <- as.matrix(cbind(rep(1, nrow(tab7.3)), tab7.3[,
+     1:2]))
> t(Z) %*% Z
```

```
                     rep(1, nrow(tab7.3))     z1     z2
rep(1, nrow(tab7.3))                  7.0    912   24.8
z1                                  911.7 121006 3402.1
z2                                   24.8   3402  170.0
```

```
> n <- nrow(Z)
> r <- ncol(Z) - 1
> tab7.3.lm <- lm(Y ~ z1 + z2, data = tab7.3)
> round(coef(tab7.3.lm), 2)
```

```
(Intercept)          z1          z2
       8.42        1.08        0.42
```

```
> round(summary(tab7.3.lm)$sigma, 3)
```

```
[1] 1.20
```

```
> s1 <- summary(tab7.3.lm)$sigma
> betahat <- matrix(coef(tab7.3.lm), ncol = 1)
> z0 <- matrix(c(1, 130, 7.5), ncol = 1)
```

Valor predito:

```
> t(z0) %*% betahat
```

```
     [,1]
[1,]  152
```

Intervalo de confiança de 95% para a média:

```
> t(z0) %*% betahat - qt(0.025, n - r - 1, lower.tail = FALSE) *
+     s1 * sqrt(t(z0) %*% solve(t(Z) %*% Z) %*% z0)
```

```
     [,1]
[1,]  150
```

```
> t(z0) %*% betahat + qt(0.025, n - r - 1, lower.tail = FALSE) *
+     s1 * sqrt(t(z0) %*% solve(t(Z) %*% Z) %*% z0)
```

```
     [,1]
[1,]  154
```

Intervalo de confiança para o valor predito

```
> t(z0) %*% betahat - qt(0.025, n - r - 1, lower.tail = FALSE) *
+     s1 * sqrt(1 + t(z0) %*% solve(t(Z) %*% Z) %*%
+     z0)
```

```
     [,1]
[1,]  148
```

```
> t(z0) %*% betahat + qt(0.025, n - r - 1, lower.tail = FALSE) *
+     s1 * sqrt(1 + t(z0) %*% solve(t(Z) %*% Z) %*%
+     z0)
```

```
     [,1]
[1,]  156
```

7.7 Exemplo 7.7

Gráficos de resíduos.

```
> par(mfrow = c(2, 2))
> plot(tab7.3$z1, residuals(tab7.3.lm))
> plot(tab7.3$z2, residuals(tab7.3.lm))
> plot(predict(tab7.3.lm), residuals(tab7.3.lm))
> plot(tab7.3.lm)
> par(mfrow = c(1, 1))
```

7.8 Exemplo 7.8

Ajuste de regressão linear multivariada.

```
> z1 <- c(0:4)
> y1 <- c(1, 4, 3, 8, 9)
> y2 <- c(-1, -1, 2, 3, 2)
```

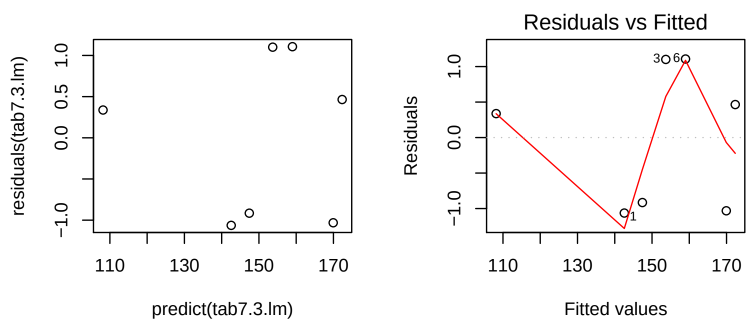


Figura 7.1: Gráficos de resíduos

Vamos reproduzir resultados contidos no PAINEL 7.2 do livro. Tomando y_1 como variável dependente:

```
> anova(lm(y1 ~ z1))

Analysis of Variance Table

Response: y1
          Df Sum Sq Mean Sq F value Pr(>F)
z1         1     40      40      20  0.021 *
Residuals  3      6       2
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

> summary(lm(y1 ~ z1))

Call:
lm(formula = y1 ~ z1)

Residuals:
        1         2         3         4         5
 6.88e-17  1.00e+00 -2.00e+00  1.00e+00 -1.33e-16

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)    1.000      1.095    0.91    0.429
z1             2.000      0.447    4.47    0.021 *
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 1.41 on 3 degrees of freedom
Multiple R-Squared: 0.87,       Adjusted R-squared: 0.826
F-statistic:   20 on 1 and 3 DF,  p-value: 0.0208

> residuals(lm(y1 ~ z1))

        1         2         3         4         5
 6.88e-17  1.00e+00 -2.00e+00  1.00e+00 -1.33e-16
```

```
> predict(lm(y1 ~ z1))

1 2 3 4 5
1 3 5 7 9
```

Tomando y_2 como variável dependente:

```
> anova(lm(y2 ~ z1))

Analysis of Variance Table

Response: y2
          Df Sum Sq Mean Sq F value Pr(>F)
z1         1  10.00   10.00     7.5  0.071 .
Residuals  3   4.00    1.33
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

> summary(lm(y2 ~ z1))

Call:
lm(formula = y2 ~ z1)

Residuals:
        1         2         3         4         5
 9.93e-17 -1.00e+00  1.00e+00  1.00e+00 -1.00e+00

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)   -1.000      0.894   -1.12    0.345
z1             1.000      0.365    2.74    0.071 .
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 1.15 on 3 degrees of freedom
Multiple R-Squared: 0.714,       Adjusted R-squared: 0.619
F-statistic:  7.5 on 1 and 3 DF,  p-value: 0.0714
```

```
> residuals(lm(y1 ~ z1))

        1         2         3         4         5
 6.88e-17  1.00e+00 -2.00e+00  1.00e+00 -1.33e-16

> predict(lm(y1 ~ z1))

1 2 3 4 5
1 3 5 7 9
```

Vamos agora considerar as duas variáveis y_1 e y_2 como dependentes:

```
> exe7.8.lm <- lm(cbind(y1, y2) ~ z1)
> summary(exe7.8.lm)

Response y1 :

Call:
lm(formula = y1 ~ z1)

Residuals:
        1         2         3         4         5
 6.88e-17  1.00e+00 -2.00e+00  1.00e+00 -1.33e-16

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)    1.000      1.095    0.91    0.429
z1             2.000      0.447    4.47    0.021 *
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 1.41 on 3 degrees of freedom
Multiple R-Squared: 0.87,        Adjusted R-squared: 0.826
F-statistic:   20 on 1 and 3 DF,  p-value: 0.0208


Response y2 :
```

```
Call:
lm(formula = y2 ~ z1)

Residuals:
        1          2          3          4          5
 9.93e-17 -1.00e+00  1.00e+00  1.00e+00 -1.00e+00

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)   -1.000      0.894   -1.12    0.345
z1             1.000      0.365    2.74    0.071 .
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 1.15 on 3 degrees of freedom
Multiple R-Squared: 0.714,        Adjusted R-squared: 0.619
F-statistic:  7.5 on 1 and 3 DF,  p-value: 0.0714

> anova(exe7.8.lm)

Analysis of Variance Table

            Df Pillai approx F num Df den Df Pr(>F)
(Intercept)  1   0.97    31.50      2      2  0.031 *
z1           1   0.94    15.00      2      2  0.062 .
Residuals    3
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

> anova(exe7.8.lm, test = "Wilks")

Analysis of Variance Table

            Df Wilks approx F num Df den Df Pr(>F)
(Intercept)  1  0.03    31.50      2      2  0.031 *
```

```
z1           1  0.06    15.00      2      2  0.062 .
Residuals    3
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

> anova(exe7.8.lm, test = "Hotelling-Lawley")

```
Analysis of Variance Table

            Df Hotelling-Lawley approx F num Df den Df Pr(>F)
(Intercept)  1             31.5     31.5      2      2  0.031 *
z1           1             15.0     15.0      2      2  0.062 .
Residuals    3
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

> anova(exe7.8.lm, test = "Roy")

```
Analysis of Variance Table

            Df  Roy approx F num Df den Df Pr(>F)
(Intercept)  1 31.5     31.5      2      2  0.031 *
z1           1 15.0     15.0      2      2  0.062 .
Residuals    3
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

> SSD(exe7.8.lm)

```
$SSD
   y1 y2
y1  6 -2
y2 -2  4

$call
lm(formula = cbind(y1, y2) ~ z1)
```

```
$df
[1] 3

attr(,"class")
[1] "SSD"

> predict(exe7.8.lm)

  y1        y2
1  1 -1.00e+00
2  3  2.22e-16
3  5  1.00e+00
4  7  2.00e+00
5  9  3.00e+00

> residuals(exe7.8.lm)

         y1        y2
1  6.88e-17  9.93e-17
2  1.00e+00 -1.00e+00
3 -2.00e+00  1.00e+00
4  1.00e+00  1.00e+00
5 -1.33e-16 -1.00e+00
```

Partição de SQ e de SP:

```
> Y <- cbind(y1, y2)
> Ytil <- predict(exe7.8.lm)
> Eps <- residuals(exe7.8.lm)
> all.equal(t(Y) %*% Y, t(Ytil) %*% Ytil + t(Eps) %*%
+     Eps)

[1] TRUE

> anova(lm(cbind(y1, y2) ~ 1), lm(cbind(y1, y2) ~ z1))
```

```
Analysis of Variance Table

Model 1: cbind(y1, y2) ~ 1
Model 2: cbind(y1, y2) ~ z1
  Res.Df Df Gen.var. Pillai approx F num Df den Df Pr(>F)
1      4        4.47
2      3 -1     1.49   0.94    15.00      2      2  0.062 .
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

O Exemplo 7.9 pode ser feito diretamente no R a partir dos dados originais.

7.9 Exemplo 7.10

Construção de uma elipse de confiança e uma elipse de predição para respostas bivariadas.

```
> y2 <- c(301.8, 396.1, 328.2, 307.4, 362.4, 369.5,
+     229.1)
> tab7.3.a <- cbind(tab7.3, y2)
> names(tab7.3.a)[3] <- "y1"
> tab7.3.a.lm <- lm(cbind(y1, y2) ~ z1 + z2, data = tab7.3.a)
> beta.hat <- coef(tab7.3.a.lm)
> z0 <- matrix(c(1, 130, 7.5), ncol = 1)
> Yhat.z0 <- predict(tab7.3.a.lm, data.frame(z1 = 130,
+     z2 = 7.5))
> SSD(tab7.3.a.lm)

$SSD
     y1    y2
y1 5.80  5.22
y2 5.22 12.57

$call
lm(formula = cbind(y1, y2) ~ z1 + z2, data = tab7.3.a)
```

```
$df
[1] 4

attr(,"class")
[1] "SSD"

> n <- nrow(tab7.3.a)
> r <- 2
> m <- 2
> fac <- tab7.3.a.lm$df.residual
```

Região de confiança para média e para valor predito:

```
> Z <- as.matrix(cbind(rep(1, n), tab7.3.a[, 1:2]))
> t1 <- sqrt(t(z0) %*% solve(t(Z) %*% Z) %*% z0) *
+     (m * n/(n - r - m)) * qf(0.05, m, n - r - m,
+     lower.tail = FALSE)
> t2 <- sqrt(1 + t(z0) %*% solve(t(Z) %*% Z) %*% z0) *
+     (m * n/(n - r - m)) * qf(0.05, m, n - r - m,
+     lower.tail = FALSE)
```

Elipse de confiança e de predição:

```
> library(ellipse)
> plot(ellipse((1/4) * SSD(tab7.3.a.lm)$SSD, centre = Yhat.z0,
+     t = t2), type = "l")
> lines(ellipse((1/4) * SSD(tab7.3.a.lm)$SSD, centre = Yhat.z0,
+     t = t1), type = "l")
> sigma.auto <- eigen((1/4) * (SSD(tab7.3.a.lm)$SSD))
> eixo1.1 <- as.vector(Yhat.z0) + t2 * sqrt(sigma.auto$values[1]) *
+     sigma.auto$vectors[, 1]
> eixo1.2 <- as.vector(Yhat.z0) - t2 * sqrt(sigma.auto$values[1]) *
+     sigma.auto$vectors[, 1]
> eixo2.1 <- as.vector(Yhat.z0) + t2 * sqrt(sigma.auto$values[2]) *
+     sigma.auto$vectors[, 2]
> eixo2.2 <- as.vector(Yhat.z0) - t2 * sqrt(sigma.auto$values[2]) *
+     sigma.auto$vectors[, 2]
```

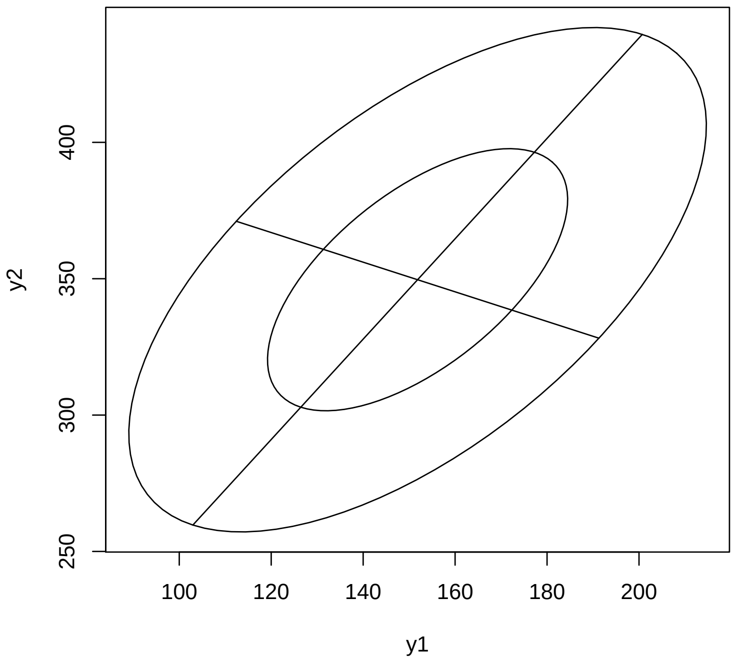


Figura 7.2: Elipse de confiança e de predição

```
> segments(eixo1.2[1], eixo1.2[2], eixo1.1[1], eixo1.1[2])
> segments(eixo2.2[1], eixo2.2[2], eixo2.1[1], eixo2.1[2])
```

7.10 Exemplo 7.11

Determinação do melhor predidor linear, seu erro médio quadrático e o coeficiente de correlação múltipla.

```
> mu <- matrix(c(5, 2, 0), ncol = 1)
> Sigma <- matrix(c(10, 1, -1, 1, 7, 3, -1, 3, 2),
+     3, 3)
> muY <- mu[1, 1]
> muZ <- mu[2:3, 1, drop = F]
> SigmaYY <- Sigma[1, 1]
> SigmaZZ <- Sigma[2:3, 2:3]
> SigmaZY <- Sigma[2:3, 1, drop = F]
```

Determinar melhor preditor linear:

```
> beta.12 <- solve(SigmaZZ) %*% SigmaZY
> beta.0 <- muY - t(beta.12) %*% muZ
```

Melhor preditor linear: $3 + Z_1 - 2Z_2$. Erro médio quadrático:

```
> SigmaYY - t(SigmaZY) %*% solve(SigmaZZ) %*% SigmaZY

     [,1]
[1,]    7
```

Coeficiente de correlação múltipla:

```
> ro.Y.Z <- sqrt((t(SigmaZY) %*% solve(SigmaZZ) %*%
+     SigmaZY)/SigmaYY)
```

Outra maneira de calcular o EMQ do preditor:

```
> SigmaYY * (1 - ro.Y.Z^2)

     [,1]
[1,]    7
```

7.11 Exemplo 7.12

Estimativa de máxima verossimilhança da função de regressão linear- resposta única.

```
> n <- nrow(tab7.3)
> muhat <- matrix(colMeans(tab7.3), ncol = 1)
> S <- cov(tab7.3)
> ybar <- muhat[3]
> zbar <- muhat[1:2, 1, drop = F]
> SZZ <- S[1:2, 1:2]
> SZY <- S[1:2, 3, drop = F]
> SYY <- S[3, 3]
```

EMV da funcão de regressão:

```
> betahat <- solve(SZZ) %*% SZY
> betahat0 <- ybar - t(betahat) %*% zbar
```

EMV do EMQ:

```
> ((n - 1)/n) * (SYY - t(SZY) %*% solve(SZZ) %*% SZY)

      Y
Y 0.828
```

7.12 Exemplo 7.13

Estimativas de máxima verossimilhança das funções de regressão- duas respostas.

```
> n <- nrow(tab7.3.a)
> muhat <- matrix(colMeans(tab7.3.a), ncol = 1)
> S <- cov(tab7.3.a)
> ybar <- muhat[3:4]
> zbar <- muhat[1:2, 1, drop = F]
> SZZ <- S[1:2, 1:2]
> SYZ <- S[3:4, 1:2, drop = F]
> SZY <- S[1:2, 3:4]
> SYY <- S[3:4, 3:4]
```

EMV da função de regressão:

```
> betahat <- SYZ %*% solve(SZZ)
> betahat0 <- ybar - betahat %*% zbar
> reg.expr <- expression(betahat0 + betahat %*% matrix(c(z1,
+     z2), ncol = 1))
> z1 <- 130
> z2 <- 3
> eval(reg.expr)
```

```
   [,1]
y1  150
y2  324
```

EMV dos Erros quadráticos esperados e da matriz de produtos cruzados:

```
> ((n - 1)/n) * (SYY - SYZ %*% solve(SZZ) %*% SZY)
```

```
      y1    y2
y1 0.828 0.746
y2 0.746 1.795
```

7.13 Exemplo 7.14

Cálculo de uma correlação parcial.
A partir dos dados de computadores no Exemplo 7.13:

```
> SYY.Z <- SYY - SYZ %*% solve(SZZ) %*% SZY
> round(SYY.Z, 3)
```

```
      y1   y2
y1 0.966 0.87
y2 0.870 2.09
```

```
> ry1y2.z <- SYY.Z[1, 2]/(sqrt(SYY.Z[1, 1]) * sqrt(SYY.Z[2,
+     2]))
> round(ry1y2.z, 2)
```

```
[1] 0.61
```

7.14 Exemplo 7.15

Duas abordagens fornecem o mesmo preditor linear.

7.15 Exemplo 7.16

Incorporar erros dependentes do tempo na regressão.

```
> tab7.4 <- read.table("t7-4.dat", col.names = c("Sendout",
+     "DHD", "DHDLag", "Windspeed", "Weekend"))
> tab7.4.lm <- lm(Sendout ~ DHD + DHDLag + Windspeed +
+     Weekend, data = tab7.4)
```

O valor de R^2 é:

```
> summary(tab7.4.lm)[[8]]
```

```
[1] 0.952
```

Função de autocorrelacão dos resíduos:

```
> acf(residuals(tab7.4.lm))
```

Valores das autocorrelações por lag:

```
> acf(residuals(tab7.4.lm), plot = F)
```

```
Autocorrelations of series ‘residuals(tab7.4.lm)’, by lag

     0      1      2      3      4      5      6      7      8
 1.000  0.515  0.276  0.259  0.277  0.105  0.265  0.351  0.092
     9     10     11     12     13     14     15     16     17
-0.046 -0.017 -0.030 -0.099 -0.020  0.038 -0.205 -0.262 -0.171
```

O próximo passo seria introduzir no modelo a autocorrelacão nos erros: modelo autoregressivo para os ruídos com N_j relacionado com anterior e o de 1 semana atrás.

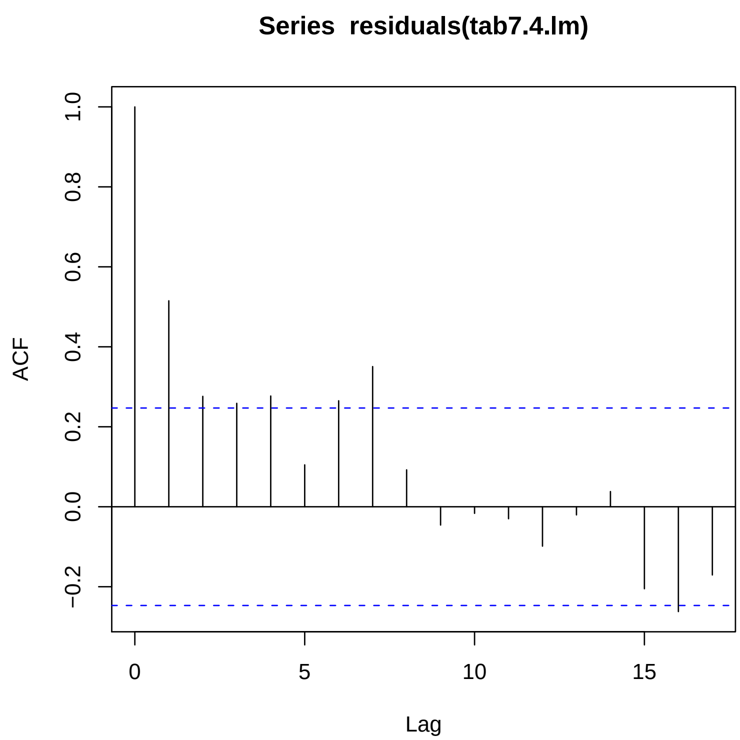


Figura 7.3: Função de autocorrelação dos resíduos

Capítulo 8

Componentes principais

8.1 Exemplo 8.1

Calcular componentes principais populacionais

```
> Sigma <- matrix(c(1, -2, 0, -2, 5, 0, 0, 0, 2), 3,
+     3)
> AU <- eigen(Sigma)
> Var.Y1 <- t(AU$vectors[, 1, drop = F]) %*% Sigma %*%
+     AU$vectors[, 1, drop = F]
> Var.Y1
```

```
     [,1]
[1,] 5.83
```

```
> abs(Var.Y1 - AU$values[1]) < 1e-10
```

```
     [,1]
[1,] TRUE
```

```
> Cov.Y1.Y2 <- t(AU$vectors[, 1, drop = F]) %*% Sigma %*%
+     AU$vectors[, 2, drop = F]
> Cov.Y1.Y2
```

```
     [,1]
[1,]    0
```

```
> abs(sum(diag(Sigma)) - sum(AU$values)) < 1e-10
```

```
[1] TRUE
```

Proporção de variância explicada:

```
> AU$values[1]/sum(AU$values)
```

```
[1] 0.729
```

```
> (AU$values[1] + AU$values[2])/sum(AU$values)
```

```
[1] 0.979
```

Correlação das componentes com as variáveis:

```
> ro.Y1.X1 <- AU$vectors[1, 1] * sqrt(AU$values[1])/sqrt(Sigma[1,
+     1])
> ro.Y1.X1
```

```
[1] -0.924
```

```
> ro.Y1.X2 <- AU$vectors[2, 1] * sqrt(AU$values[1])/sqrt(Sigma[2,
+     2])
> ro.Y1.X2
```

```
[1] 0.997
```

```
> ro.Y2.X1 <- AU$vectors[1, 2] * sqrt(AU$values[2])/sqrt(Sigma[1,
+     1])
> ro.Y2.X1
```

```
[1] 0
```

```
> ro.Y2.X2 <- AU$vectors[2, 2] * sqrt(AU$values[2])/sqrt(Sigma[2,
+     2])
> ro.Y2.X2
```

```
[1] 0
```

```
> ro.Y2.X3 <- AU$vectors[3, 2] * sqrt(AU$values[2])/sqrt(Sigma[3,
+     3])
> ro.Y2.X3
```

```
[1] 1
```

Figura 8.1

```
> ro.mat <- matrix(c(1, 0.75, 0.75, 1), 2, 2)
> library(ellipse)
> plot(ellipse(0.75), type = "l", axes = F)
> abline(h = 0)
> abline(v = 0)
> abline(coef = c(0, 1))
> abline(coef = c(0, -1))
> text(2.5, 0, "x1")
> text(0, 2.5, "x2")
> text(2.5, 2.5, "y1")
> text(-2.5, 2.5, "y2")
```

8.2 Exemplo 8.2

Componentes principais obtidas a partir da matriz de covariância e da matriz de correlação são diferentes.

```
> Sigma <- matrix(c(1, 4, 4, 100), 2, 2)
> ro <- matrix(c(1, 0.4, 0.4, 1), 2, 2)
```

Para Σ:

```
> Sigma.auto <- eigen(Sigma)
> ro.auto <- eigen(ro)
```

Variação explicada:

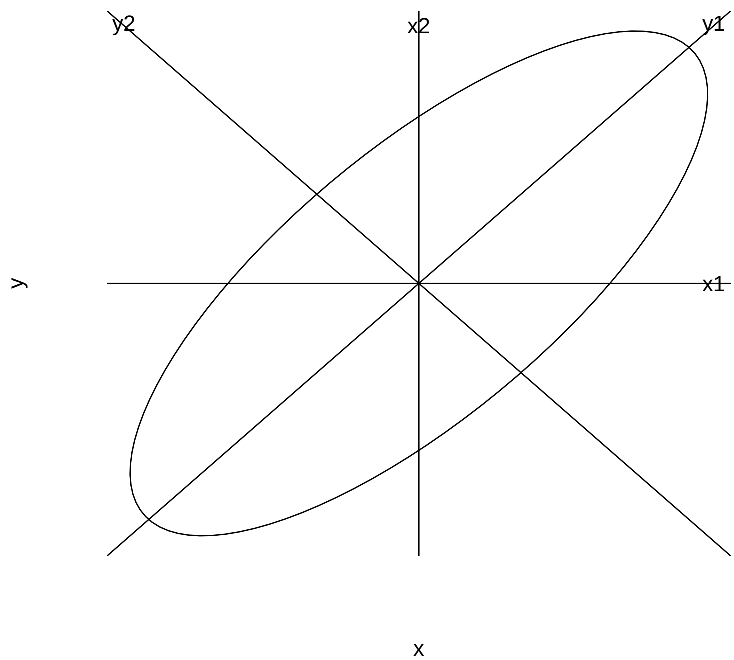


Figura 8.1: A elipse de densidade constante $x^t\Sigma^{-1}x = c^2$ e as componentes principais y_1 e y_2 para o vetor normal bivariado X com média 0

```
> Sigma.auto$values[1]/sum(Sigma.auto$values)

[1] 0.992
```

Para variáveis padronizadas:

```
> ro.Y1.Z1 <- ro.auto$vectors[1, 1] * sqrt(ro.auto$values[1])
> ro.Y1.Z2 <- ro.auto$vectors[1, 2] * sqrt(ro.auto$values[1])
> ro.auto$values[1]/sum(ro.auto$values)

[1] 0.7
```

8.3 Exemplo 8.3

Resumo da variabilidade amostral com duas componentes principais amostrais.

```
> tab8.5 <- read.table("t8-5.dat", col.names = c("POP",
+     "ESC", "EMP", "SAUD", "VRES"))
> xbar <- as.matrix(colMeans(tab8.5))
> S <- cov(tab8.5)
> tab8.5.prcomp <- prcomp(tab8.5)
> round(tab8.5.prcomp$rotation, 3)

        PC1    PC2    PC3    PC4    PC5
POP  -0.781  0.071 -0.004  0.542  0.302
ESC  -0.306  0.764  0.162 -0.545  0.009
EMP  -0.334 -0.083 -0.015  0.051 -0.937
SAUD -0.426 -0.579 -0.220 -0.636  0.172
VRES  0.054  0.262 -0.962  0.051 -0.025

> round((tab8.5.prcomp$sdev)^2, 3)

[1] 6.931 1.785 0.390 0.230 0.014
```

Tabela da página 440 do livro texto:

```
> e1.hat <- round(tab8.5.prcomp$rotation[, 1], 3)
> rY1.Xk <- round(tab8.5.prcomp$rotation[, 1] * tab8.5.prcomp$sdev[1]/sqrt(diag(S
+     2)
> e2.hat <- round(tab8.5.prcomp$rotation[, 2], 3)
> rY2.Xk <- round(tab8.5.prcomp$rotation[, 2] * tab8.5.prcomp$sdev[2]/sqrt(diag(S
+     2)
> e3.hat <- round(tab8.5.prcomp$rotation[, 3], 3)
> e4.hat <- round(tab8.5.prcomp$rotation[, 4], 3)
> e5.hat <- round(tab8.5.prcomp$rotation[, 5], 3)
> varian <- round(tab8.5.prcomp$sdev^2, 3)
> acum <- round(100 * cumsum(tab8.5.prcomp$sdev^2)/sum(tab8.5.prcomp$sdev^2),
+     1)
```

Primeira parte da Tabela da p. 440

```
> tab8.5.cp1 <- -cbind(e1.hat, rY1.Xk, e2.hat, rY2.Xk,
+     e3.hat, e4.hat, e5.hat)
> tab8.5.cp1
```

```
     e1.hat rY1.Xk e2.hat rY2.Xk e3.hat e4.hat e5.hat
POP   0.781   0.99 -0.071  -0.05  0.004 -0.542 -0.302
ESC   0.306   0.61 -0.764  -0.77 -0.162  0.545 -0.009
EMP   0.334   0.98  0.083   0.12  0.015 -0.051  0.937
SAUD  0.426   0.80  0.579   0.55  0.220  0.636 -0.172
VRES -0.054  -0.20 -0.262  -0.49  0.962 -0.051  0.025
```

Segunda parte da Tabela da p. 440

```
> tab8.5.cp2 <- rbind(varian, acum)
> dimnames(tab8.5.cp2)[[2]] <- c("Y1", "Y2", "Y3",
+     "Y4", "Y5")
> tab8.5.cp1
```

```
     e1.hat rY1.Xk e2.hat rY2.Xk e3.hat e4.hat e5.hat
POP   0.781   0.99 -0.071  -0.05  0.004 -0.542 -0.302
ESC   0.306   0.61 -0.764  -0.77 -0.162  0.545 -0.009
EMP   0.334   0.98  0.083   0.12  0.015 -0.051  0.937
SAUD  0.426   0.80  0.579   0.55  0.220  0.636 -0.172
VRES -0.054  -0.20 -0.262  -0.49  0.962 -0.051  0.025
```

8.4 Exemplo 8.4

Resumo da variabilidade amostral com uma componente principal amostral.

```
> tab6.9 <- read.table("t6-9.dat", col.names = c("COMP",
+     "LARG", "ALT", "SEXO"))
> tab6.9 <- transform(tab6.9, COMP = log(COMP), LARG = log(LARG),
+     ALT = log(ALT))
```

Observações para sexo feminino:

```
> tab6.9.M <- subset(tab6.9, SEXO == "male")
> xbar <- matrix(colMeans(tab6.9.M[, 1:3]), ncol = 1)
> S <- cov(tab6.9.M[, 1:3])
```

Tabela da página 442

```
> tab6.9M.prcomp <- prcomp(tab6.9.M[, 1:3])
> e1.hat <- round(tab6.9M.prcomp$rotation[, 1], 3)
> rY1.Xk <- round(tab6.9M.prcomp$rotation[, 1] * tab6.9M.prcomp$sdev[1]/sqrt(diag(S)),
+     2)
> e2.hat <- round(tab6.9M.prcomp$rotation[, 2], 3)
> e3.hat <- round(tab6.9M.prcomp$rotation[, 3], 3)
> varian <- round(tab6.9M.prcomp$sdev^2, 3)
> acum <- round(100 * cumsum(tab6.9M.prcomp$sdev^2)/sum(tab6.9M.prcomp$sdev^2),
+     1)
```

Primeira parte da Tabela da p. 442

```
> tab6.9M.cp1 <- cbind(e1.hat, rY1.Xk, e2.hat, e3.hat)
> tab6.9M.cp1

     e1.hat rY1.Xk e2.hat e3.hat
COMP  0.683   0.99 -0.159  0.713
LARG  0.510   0.97 -0.594 -0.622
ALT   0.523   0.97  0.788 -0.324
```

Segunda parte da Tabela da p. 442

```
> tab6.9M.cp2 <- rbind(varian, acum)
> dimnames(tab6.9M.cp2)[[2]] <- c("Y1", "Y2", "Y3")
> tab6.9M.cp2

           Y1     Y2  Y3
varian  0.023  0.001   0
acum   96.100 98.500 100
```

A Primeira C.P. explica 96% da variância total e tem uma interpretação prática. Ela é dada por:

$$\begin{aligned}\hat{y}_1 &= 0.683\ln(COMP) + 0.510\ln(LARG) + 0.523\ln(ALT)\\ &= \ln\left[(COMP)^{.683}(LARG)^{.510}(ALT)^{.523}\right].\end{aligned}$$

que pode ser interpretada como uma medida de volume.

```
> plot(tab6.9M.prcomp)
> screeplot(tab6.9M.prcomp, type = "lines")
```

8.5 Exemplo 8.5

Componentes principais amostrais a partir de dados padronizados.

```
> tab8.4 <- read.table("t8-4.dat", col.names = c("x1",
+     "x2", "x3", "x4", "x5"))
> xbar <- matrix(colMeans(tab8.4), ncol = 1)
> R <- cor(tab8.4)
```

Autovalores e autovetores de R:

```
> R.AU <- eigen(R)
> round(R.AU$values, 3)

[1] 2.856 0.809 0.540 0.451 0.343

> round(R.AU$vectors, 3)
```

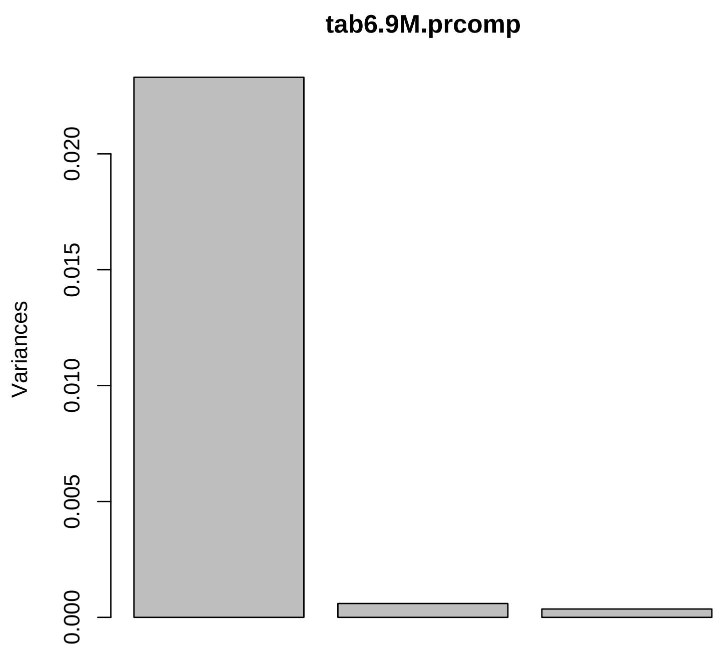


Figura 8.2: Gráfico scree para dados de tartarugas

```
      [,1]   [,2]   [,3]   [,4]   [,5]
[1,] 0.464  0.241  0.613 -0.381 -0.453
[2,] 0.457  0.509 -0.178 -0.211  0.675
[3,] 0.470  0.261 -0.337  0.664 -0.396
[4,] 0.422 -0.525 -0.539 -0.473 -0.179
[5,] 0.421 -0.582  0.434  0.381  0.387
```

```
> tab8.4.P <- scale(tab8.4)
> attributes(tab8.4.P) <- NULL
> attr(tab8.4.P, "dim") <- dim(tab8.4)
> dimnames(tab8.4.P) <- list(NULL, c("z1", "z2", "z3",
+     "z4", "z5"))
> e1hat <- R.AU$vectors[, 1]
> e2hat <- R.AU$vectors[, 2]
> y1hat <- expression(t(e1hat) %*% z)
> y2hat <- expression(t(e2hat) %*% z)
> cp1.amo <- apply(tab8.4.P, 1, function(t) {
+     z <- t
+     eval(y1hat)
+ })
> cp2.amo <- apply(tab8.4.P, 1, function(t) {
+     z <- t
+     eval(y2hat)
+ })
```

Quantidade de variação explicada:

```
> (R.AU$values[1] + R.AU$values[2])/sum(R.AU$values)
```

```
[1] 0.733
```

8.6 Exemplo 8.6

Componentes a partir de uma matriz de correlação com uma estrutura especial.

```
> xbar <- matrix(c(39.88, 45.08, 48.11, 49.95), ncol = 1)
> R <- matrix(c(1, 0.7501, 0.6329, 0.6363, 0.7501,
```

```
+     1, 0.6925, 0.7386, 0.6329, 0.6925, 1, 0.6625,
+     0.6363, 0.7386, 0.6625, 1), 4, 4)
> AU.R86 <- eigen(R)
> AU.R86$values
```

```
[1] 3.058 0.382 0.342 0.217
```

```
> p <- ncol(R)
> 1 + (p - 1) * mean(R[row(R) != col(R)])
```

```
[1] 3.06
```

Primeira CP:

```
> y1hat <- expression(t(z) %*% AU.R86$vectors[, 1,
+     drop = F])
```

Quantidade de variação explicada pela primeira CP:

```
> 100 * AU.R86$values[1]/sum(AU.R86$values)
```

```
[1] 76.5
```

8.7 Exemplo 8.7

Gráfico das CPs para dados de tartarugas.

```
> tab6.9M.cp.pred <- predict(tab6.9M.prcomp)
> y1hat <- tab6.9M.cp.pred[, 1]
> y2hat <- tab6.9M.cp.pred[, 2]
> y3hat <- tab6.9M.cp.pred[, 3]
```

```
> qqnorm(y2hat)
```

Há um ponto suspeito no gráfico.

```
> plot(y2hat, y1hat)
```

Os pontos não mostram afastamento da normalidade, exceto por um ponto.

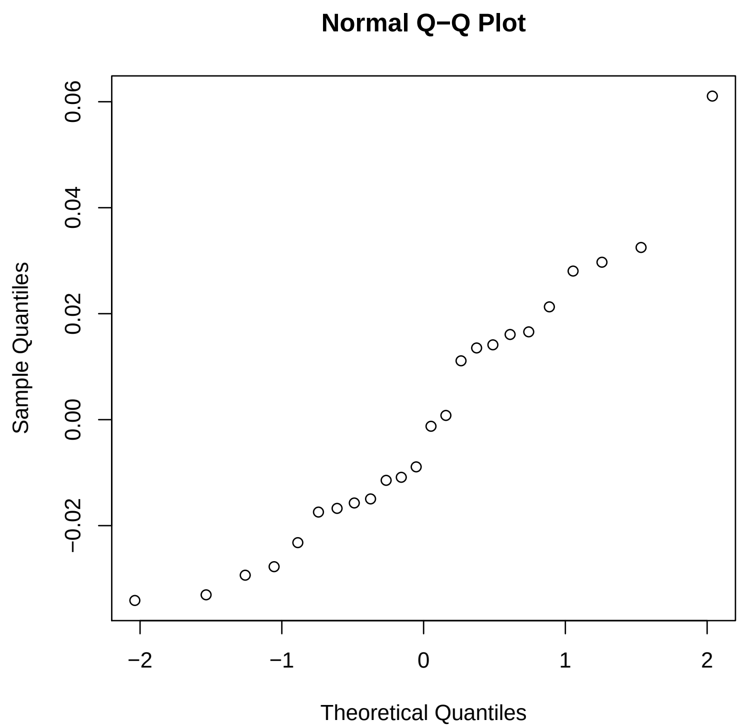


Figura 8.3: Um gráfico Q-Q da segunda CP $\hat{y}_2$ para os dados de tartarugas machos

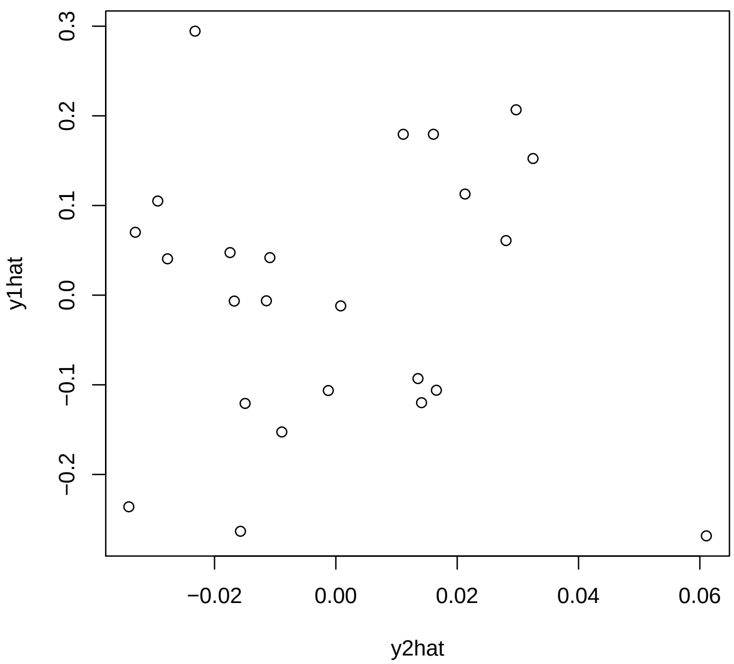


Figura 8.4: Gráfico de dispersão das CP's $\hat{y}_1$ e $\hat{y}_2$ dos dados de tartarugas machos

8.8 Exemplo 8.8

Construção de I.C. para λ_1.

```
> tab8.4 <- read.table("t8-4.dat")
> nn <- nrow(tab8.4)
> AU.tab8.4 <- eigen(cov(tab8.4))
> AU.tab8.4$values[1]

[1] 0.00360
```

Intervalo de confiança aproximado de 95% para λ_1:

```
> round(AU.tab8.4$values[1]/(1 + qnorm(0.025, lower.tail = F) *
+     sqrt(2/n)), 4)

[1] 0.0018

> round(AU.tab8.4$values[1]/(1 - qnorm(0.025, lower.tail = F) *
+     sqrt(2/n)), 4)

[1] -0.0755
```

8.9 Exemplo 8.9

Teste da estrutura de equi-correlação

```
> R <- matrix(c(1, 0.7501, 0.6329, 0.6363, 0.7501,
+     1, 0.6925, 0.7386, 0.6329, 0.6925, 1, 0.6625,
+     0.6363, 0.7386, 0.6625, 1), 4, 4)
> n <- 150
> p <- nrow(R)
> rbar <- numeric(p)
> for (i in 1:p) {
+     rbar[i] <- (1/(p - 1)) * sum(R[row(R) != i &
+         col(R) == i])
+ }
```

```
> r.med <- 2/(p * (p - 1)) * sum(R[row(R) < col(R)])
> gama.hat <- ((p - 1)^2 * (1 - (1 - r.med)^2))/(p -
+     (p - 2) * (1 - r.med)^2)
> T1 <- sum((R[row(R) < col(R)] - r.med)^2)
> T2 <- sum((rbar - r.med)^2)
> T.est <- ((n - 1)/(1 - r.med)^2) * (T1 - gama.hat *
+     T2)
> val.crit <- qchisq(0.05, (p + 1) * (p - 2)/2, lower.tail = F)
> T.est > val.crit
```

```
[1] TRUE
```

Há evidência contrária a H_0, mas não muito forte.

Capítulo 9

Análise fatorial

9.1 Exemplo 9.1

Verificação da relação $\mathbf{\Sigma} = \mathbf{L}\mathbf{L}^{\mathbf{t}} + \mathbf{\Psi}$

```
> Sigma <- matrix(c(19, 30, 2, 12, 30, 57, 5, 23, 2,
+     5, 38, 47, 12, 23, 47, 68), 4, 4)
> L <- matrix(c(4, 7, -1, 1, 1, 2, 6, 8), 4, 2)
> Psi <- diag(c(2, 4, 1, 3))
> all.equal(Sigma, L %*% t(L) + Psi)
```

```
[1] TRUE
```

Comunalidade de X_1:

```
> h1.2 <- L[1, 1]^2 + L[1, 2]^2
```

Decomposição da variância de X_1:

```
> all.equal(Sigma[1, 1], h1.2 + Psi[1, 1])
```

```
[1] TRUE
```

Supondo p=12 , m=2, número de elementos de Σ:

```
> p <- 12
> p * (p + 1)/2

[1] 78
```

Descrito em termos de

```
> m <- 2
> m * p + p

[1] 36
```

9.2 Exemplo 9.2

9.3 Exemplo 9.3

Análise fatorial de dados de preferência de consumidor.

```
> R <- matrix(c(1, 0.02, 0.96, 0.42, 0.01, 0.02, 1,
+     0.13, 0.71, 0.85, 0.96, 0.13, 1, 0.5, 0.11, 0.42,
+     0.71, 0.5, 1, 0.79, 0.01, 0.85, 0.11, 0.79, 1),
+     5, 5)
> dimnames(R) <- list(c("Sabor", "Barganha", "Gosto",
+     "Lanche", "Energia"), c("Sabor", "Barganha",
+     "Gosto", "Lanche", "Energia"))
> R.eigen <- eigen(R)
```

Vamos tomar os dois primeiros auto valores de R e estimar a proporção de variação explicada:

```
> round(c(sum(R.eigen$values[1])/nrow(R), sum(R.eigen$values[1:2])/nrow(R)),
+     3)

[1] 0.571 0.932
```

Vamos obter cargas fatoriais e comunalidades, usando o método de componentes principais.

```
> L <- matrix(NA, 5, 2)
> L[, 1] <- sqrt(R.eigen$values[1]) * R.eigen$vectors[,
+     1]
> L[, 2] <- sqrt(R.eigen$values[2]) * R.eigen$vectors[,
+     2]
> round(L, 2)
```

```
      [,1]  [,2]
[1,] -0.56  0.82
[2,] -0.78 -0.52
[3,] -0.65  0.75
[4,] -0.94 -0.10
[5,] -0.80 -0.54
```

No livro a primeira coluna de L aparece com os sinais trocados. As colunas são definidas a menos do sinal. A matriz de variâncias específicas é dada por:

```
> psi <- diag(diag(R - L %*% t(L)))
> round(psi, 2)
```

```
     [,1] [,2] [,3] [,4] [,5]
[1,] 0.02 0.00 0.00 0.00 0.00
[2,] 0.00 0.12 0.00 0.00 0.00
[3,] 0.00 0.00 0.02 0.00 0.00
[4,] 0.00 0.00 0.00 0.11 0.00
[5,] 0.00 0.00 0.00 0.00 0.07
```

Verificação: comparar R com $\tilde{\mathbf{L}}\tilde{\mathbf{L}}^{\mathbf{t}} + \tilde{\mathbf{\Psi}}$

```
> round(L %*% t(L) + psi, 2)
```

```
     [,1] [,2] [,3] [,4] [,5]
[1,] 1.00 0.01 0.97 0.44 0.00
[2,] 0.01 1.00 0.11 0.78 0.91
[3,] 0.97 0.11 1.00 0.53 0.11
[4,] 0.44 0.78 0.53 1.00 0.81
[5,] 0.00 0.91 0.11 0.81 1.00
```

Reproduz aproximadamente R.

9.4 Exemplo 9.4

Análise fatorial para dados de valor de ações.

```
> tab8.4 <- read.table("t8-4.dat", col.names = c("All.Chem",
+     "Du_Pont", "Union_Carbide", "Exxon", "Texaco"))
> xbar <- matrix(colMeans(tab8.4), ncol = 1)
> R <- cor(tab8.4)
```

Autovalores e autovetores de R:

```
> R.AU <- eigen(R)
> table9.2 <- matrix(NA, 5, 5, dimnames = list(names(tab8.4),
+     c("F1", "psi1", "F1", "F2", "psi1")))
> table9.2[, 1] <- (-1) * sqrt(R.AU$values[1]) * R.AU$vectors[,
+     1]
> table9.2[, 2] <- 1 - table9.2[, 1]^2
> table9.2[, 3] <- table9.2[, 1]
> table9.2[, 4] <- (-1) * sqrt(R.AU$values[2]) * R.AU$vectors[,
+     2]
> table9.2[, 5] <- 1 - table9.2[, 3]^2 - table9.2[,
+     4]^2
> round(table9.2, 3)
```

```
                  F1  psi1     F1     F2  psi1
All.Chem      -0.783 0.386 -0.783 -0.217 0.339
Du_Pont       -0.773 0.403 -0.773 -0.458 0.194
Union_Carbide -0.794 0.369 -0.794 -0.234 0.314
Exxon         -0.713 0.492 -0.713  0.472 0.269
Texaco        -0.712 0.493 -0.712  0.524 0.219
```

Proporção explicada acumulada:

```
> round(c(R.AU$values[1], sum(R.AU$values[1:2]))/nrow(R),
+     3)
```

```
[1] 0.571 0.733
```

Matriz residual para m=2 fatores

```
> L <- table9.2[, 3:4]
> round(R - L %*% t(L) - diag(table9.2[, 5]), 3)
```

```
              All.Chem Du_Pont Union_Carbide  Exxon Texaco
All.Chem         0.000  -0.128        -0.164 -0.069  0.018
Du_Pont         -0.128   0.000        -0.123  0.055  0.012
Union_Carbide   -0.164  -0.123         0.000 -0.019 -0.017
Exxon           -0.069   0.055        -0.019  0.000 -0.231
Texaco           0.018   0.012        -0.017 -0.231  0.000
```

Em geral $\mathbf{LL^t}$ produz valores maiores que os em R.

9.5 Exemplo 9.5

Análise fatorial dos dados de preços de ações usando o método de máxima verossimilhança.

```
> tab8.4.AF <- factanal(covmat = R, factors = 2, rotation = "varimax")
> table9.3 <- matrix(NA, 5, 3, dimnames = list(c("All.Chem",
+     "Du_Pont", "Union_Carbide", "Exxon", "Texaco"),
+     c("F1", "F2", "Psi")))
> table9.3[, 1:2] <- loadings(tab8.4.AF)
> table9.3[, 3] <- tab8.4.AF$uniquenesses
> round(table9.3, 3)
```

```
                 F1    F2   Psi
All.Chem      0.601 0.378 0.497
Du_Pont       0.849 0.165 0.252
Union_Carbide 0.643 0.336 0.474
Exxon         0.365 0.507 0.610
Texaco        0.207 0.884 0.176
```

Ver table9.2 e matriz de resíduos:

```
> round(R - loadings(tab8.4.AF) %*% t(loadings(tab8.4.AF)) -
+     diag(tab8.4.AF$uniquenesses), 3)
```

```
              All.Chem Du_Pont Union_Carbide  Exxon Texaco
All.Chem         0.000   0.005        -0.004 -0.024  0.004
Du_Pont          0.005   0.000        -0.003 -0.004  0.000
Union_Carbide   -0.004  -0.003         0.000  0.031 -0.004
Exxon           -0.024  -0.004         0.031  0.000  0.000
Texaco           0.004   0.000        -0.004  0.000  0.000
```

Os elementos da matriz de resíduos são muito menores que os obtidos pelo método de componentes principais.

9.6 Exemplo 9.6

Análise fatorial dos dados de decatlo olímpico.
Função para ler matriz simétrica:

```
> mat.sim <- function(x, n) {
+     A <- matrix(0, n, n)
+     A[row(A) >= col(A)] <- x
+     A + t(A) - diag(diag(A))
+ }
```

Vamos aplicar essa função aos dados da matriz R na página 495:

```
> R <- mat.sim(c(1, 0.59, 0.35, 0.34, 0.63, 0.4, 0.28,
+     0.2, 0.11, -0.07, 1, 0.42, 0.51, 0.49, 0.52,
+     0.31, 0.36, 0.21, 0.09, 1, 0.38, 0.19, 0.36,
+     0.73, 0.24, 0.44, -0.08, 1, 0.29, 0.46, 0.27,
+     0.39, 0.17, 0.18, 1, 0.34, 0.17, 0.23, 0.13,
+     0.39, 1, 0.32, 0.33, 0.18, 0, 1, 0.24, 0.34,
+     -0.02, 1, 0.24, 0.17, 1, -0, 1), 10)
> dimnames(R) <- list(c("100m", "SaltDist", "LancPeso",
+     "Alt", "400m", "100mBarr", "Disco", "SaltVara",
+     "Dardo", "1500m"), c("100m", "SaltDist", "LancPeso",
+     "Alt", "400m", "110mBarr", "Disco", "SaltVara",
+     "Dardo", "1500m"))
> R.eigen <- eigen(R)
> round(R.eigen$values, 2)
```

```
 [1] 3.79 1.52 1.11 0.91 0.72 0.59 0.53 0.38 0.24 0.21
```

Tabela para o método de componentes principais:

```
> Tab94a <- matrix(NA, 10, 4)
> for (i in 1:4) Tab94a[, i] <- sqrt(R.eigen$values[i]) *
+     R.eigen$vectors[, i]
> dimnames(Tab94a) <- list(c("100m", "SaltDist", "LancPeso",
+     "Alt", "400m", "100mBarr", "Disco", "SaltVara",
+     "Dardo", "1500m"), c("F1", "F2", "F3", "F4"))
> Psi <- 1 - rowSums(Tab94a^2)
> Tab94a <- cbind((-1) * Tab94a, Psi)
> round(Tab94a, 3)
```

```
            F1     F2     F3     F4   Psi
100m     0.691  0.217 -0.520  0.206 0.163
SaltDist 0.789  0.184 -0.193 -0.092 0.299
LancPeso 0.702 -0.535  0.047  0.175 0.189
Alt      0.674  0.134  0.139 -0.396 0.352
400m     0.620  0.551 -0.084  0.419 0.130
100mBarr 0.687  0.042 -0.161 -0.345 0.382
Disco    0.621 -0.521  0.109  0.234 0.276
SaltVara 0.538  0.087  0.411 -0.440 0.340
Dardo    0.434 -0.439  0.372  0.235 0.426
1500m    0.147  0.596  0.658  0.279 0.112
```

Proporção acumulada de variância explicada:

```
> round(cumsum(R.eigen$values[1:4])/10, 2)
```

```
[1] 0.38 0.53 0.64 0.73
```

Tabela para o método de máxima verossimilhança:

```
> Tab9.4.AF <- factanal(covmat = R, factors = 2)
> Tab94b <- matrix(NA, 10, 5)
> dimnames(Tab94b) <- list(c("100m", "SaltDist", "LancPeso",
```

```
+     "Alt", "400m", "110mBarr", "Disco", "SaltVara",
+     "Dardo", "1500m"), c("F1", "F2", "F3", "F4",
+     "Psi"))
> Tab94b[, 1:4] <- loadings(Tab9.4.AF)
> Tab94b[, 5] <- Tab9.4.AF$uniquenesses
> dimnames(Tab94b) <- list(c("100m", "SaltDist", "LancPeso",
+     "Alt", "400m", "110mBarr", "Disco", "SaltVara",
+     "Dardo", "1500m"), c("F1", "F2", "F3", "F4",
+     "Psi"))
> round(Tab94b, 3)
```

```
            F1     F2    F3     F4   Psi
100m     0.701  0.220 0.701  0.220 0.461
SaltDist 0.733  0.294 0.733  0.294 0.376
LancPeso 0.197  0.939 0.197  0.939 0.080
Alt      0.508  0.297 0.508  0.297 0.653
400m     0.723  0.045 0.723  0.045 0.475
110mBarr 0.539  0.281 0.539  0.281 0.631
Disco    0.164  0.741 0.164  0.741 0.424
SaltVara 0.378  0.192 0.378  0.192 0.820
Dardo    0.108  0.444 0.108  0.444 0.791
1500m    0.240 -0.135 0.240 -0.135 0.924
```

Proporção acumulada de variância explicada:

```
> round(c(sum(Tab94b[, 1]^2)/10, sum(Tab94b[, 1:2]^2)/10,
+     sum(Tab94b[, 1:3]^2)/10, sum(Tab94b[, 1:4]^2)/10),
+     2)
```

```
[1] 0.24 0.44 0.67 0.87
```

Matrizes de resíduos:
Componentes principais:

```
> round(R - Tab94a[, 1:4] %*% t(Tab94a[, 1:4]) - diag(Tab94a[,
+     5]), 3)
```

```
           100m SaltDist LancPeso    Alt   400m 110mBarr  Disco
100m      0.000   -0.075   -0.030 -0.001 -0.047   -0.096 -0.027
SaltDist -0.075    0.000   -0.010 -0.056 -0.077   -0.092 -0.041
LancPeso -0.030   -0.010    0.000  0.042 -0.020   -0.032 -0.031
Alt      -0.001   -0.056    0.042  0.000 -0.024   -0.122 -0.001
400m     -0.047   -0.077   -0.020 -0.024  0.000    0.022 -0.017
100mBarr -0.096   -0.092   -0.032 -0.122  0.022    0.000  0.014
Disco    -0.027   -0.041   -0.031 -0.001 -0.017    0.014  0.000
SaltVara  0.114   -0.042   -0.034 -0.215  0.067   -0.129  0.009
Dardo     0.051    0.042   -0.158 -0.022  0.036    0.041 -0.254
1500m    -0.016    0.017    0.056  0.020 -0.091    0.076  0.062
         SaltVara  Dardo  1500m
100m        0.114  0.051 -0.016
SaltDist   -0.042  0.042  0.017
LancPeso   -0.034 -0.158  0.056
Alt        -0.215 -0.022  0.020
400m        0.067  0.036 -0.091
100mBarr   -0.129  0.041  0.076
Disco       0.009 -0.254  0.062
SaltVara    0.000 -0.005 -0.109
Dardo      -0.005  0.000 -0.112
1500m      -0.109 -0.112  0.000
```

Máxima Verossimilhança:

```
> round(R - Tab94b[, 1:4] %*% t(Tab94b[, 1:4]) - diag(Tab94b[,
+     5]), 3)
```

```
           100m SaltDist LancPeso    Alt   400m 110mBarr  Disco
100m     -0.539   -0.566   -0.340 -0.503 -0.403   -0.479 -0.276
SaltDist -0.566   -0.624   -0.421 -0.410 -0.597   -0.435 -0.366
LancPeso -0.340   -0.421   -0.920 -0.378 -0.181   -0.381 -0.726
Alt      -0.503   -0.410   -0.378 -0.347 -0.472   -0.255 -0.337
400m     -0.403   -0.597   -0.181 -0.472 -0.525   -0.465 -0.135
100mBarr -0.479   -0.435   -0.381 -0.255 -0.465   -0.369 -0.274
Disco    -0.276   -0.366   -0.726 -0.337 -0.135   -0.274 -0.576
```

```
SaltVara -0.415   -0.307   -0.269 -0.109 -0.335   -0.186 -0.168
Dardo    -0.237   -0.210   -0.436 -0.204 -0.067   -0.187 -0.353
1500m    -0.346   -0.182    0.079  0.017  0.056   -0.182  0.102
         SaltVara  Dardo  1500m
100m       -0.415 -0.237 -0.346
SaltDist   -0.307 -0.210 -0.182
LancPeso   -0.269 -0.436  0.079
Alt        -0.109 -0.204  0.017
400m       -0.335 -0.067  0.056
100mBarr   -0.186 -0.187 -0.182
Disco      -0.168 -0.353  0.102
SaltVara   -0.180 -0.012  0.040
Dardo      -0.012 -0.209  0.068
1500m       0.040  0.068 -0.076
```

9.7 Exemplo 9.7

Teste para dois fatores comuns.

```
> tab8.4 <- read.table("t8-4.dat", col.names = c("All.Chem",
+     "Du_Pont", "Union_Carbide", "Exxon", "Texaco"))
> R <- cor(tab8.4)
> tab8.4.AF <- factanal(covmat = R, factors = 2, rotation = "varimax")
> table9.3 <- matrix(NA, 5, 3, dimnames = list(c("All.Chem",
+     "Du_Pont", "Union_Carbide", "Exxon", "Texaco"),
+     c("F1", "F2", "Psi")))
> table9.3[, 1:2] <- loadings(tab8.4.AF)
> table9.3[, 3] <- tab8.4.AF$uniquenesses
> estat <- det(table9.3[, 1:2] %*% t(table9.3[, 1:2]) +
+     diag(table9.3[, 3]))/det(R)
> n <- 100
> p <- 5
> m <- 2
> estat <- (n - 1 - (2 * p + 4 * m + 5)/6) * log(estat)
> (1/2) * ((p - m)^2 - p - m)
```

```
[1] 1
```

```
> estat > qchisq(0.05, 1, lower.tail = F)
```

```
[1] FALSE
```

```
> pchisq(estat, 1, lower.tail = F)
```

```
[1] 0.448
```

Teste não rejeita para $\alpha = 5\%$.

9.8 Exemplo 9.8

Uma primeira visão da rotação fatorial.

```
> R <- mat.sim(c(1, 0.439, 0.41, 0.288, 0.329, 0.248,
+     1, 0.351, 0.354, 0.32, 0.329, 1, 0.164, 0.19,
+     0.181, 1, 0.595, 0.47, 1, 0.464, 1), 6)
> dimnames(R) <- list(c("Galico", "Ingles", "Historia",
+     "Aritmetica", "Algebra", "Geometria"), c("Galico",
+     "Ingles", "Historia", "Aritmetica", "Algebra",
+     "Geometria"))
> exe97.AF <- factanal(covmat = R, factors = 2, rotation = "varimax")
```

```
> plot(loadings(exe97.AF))
> text(loadings(exe97.AF), dimnames(R)[[2]], cex = 0.6)
```

9.9 Exemplo 9.9

Cargas rotacionadas para os dados de preferência do consumidor.

```
> R <- matrix(c(1, 0.02, 0.96, 0.42, 0.01, 0.02, 1,
+     0.13, 0.71, 0.85, 0.96, 0.13, 1, 0.5, 0.11, 0.42,
+     0.71, 0.5, 1, 0.79, 0.01, 0.85, 0.11, 0.79, 1),
+     5, 5)
```

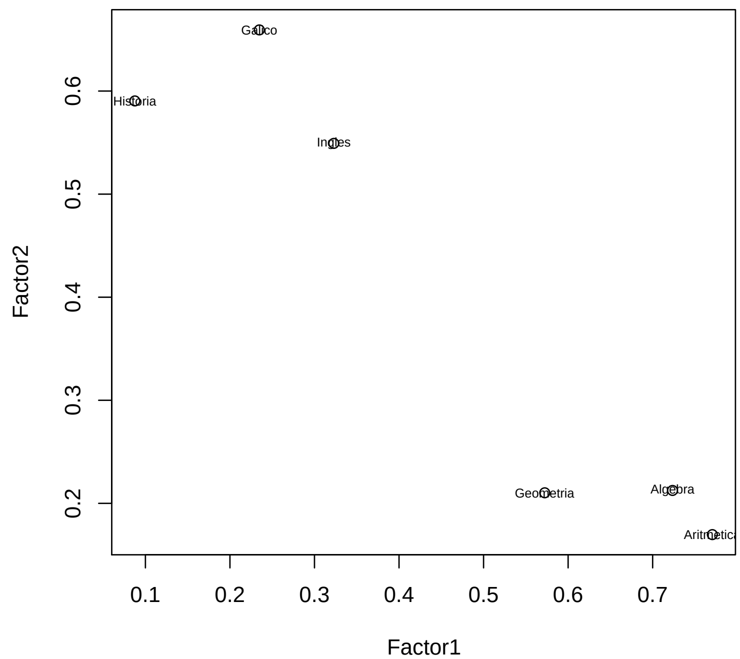


Figura 9.1: Gráficos das cargas fatoriais

```
> dimnames(R) <- list(c("Sabor", "Barganha", "Gosto",
+     "Lanche", "Energia"), c("Sabor", "Barganha",
+     "Gosto", "Lanche", "Energia"))
```

Ajuste inicial: componentes principais

```
> R.eigen <- eigen(R)
> R.eigen$values

[1] 2.8531 1.8063 0.2045 0.1024 0.0337
```

Proporção explicada:

```
> R.eigen$values/5

[1] 0.57062 0.36127 0.04090 0.02048 0.00674
```

Proporção acumulada:

```
> cumsum(R.eigen$values)/5

[1] 0.571 0.932 0.973 0.993 1.000
```

Dois fatores comuns explicam:

```
> PANEL9.1a <- matrix(NA, 5, 4, dimnames = list(c("Sabor",
+     "Barganha", "Gosto", "Lanche", "Energia"), c("F1",
+     "F2", "Comun", "Var.Esp")))
> PANEL9.1a[, 1] <- (-1) * sqrt(R.eigen$values[1]) *
+     R.eigen$vectors[, 1]
> PANEL9.1a[, 2] <- sqrt(R.eigen$values[2]) * R.eigen$vectors[,
+     2]
> PANEL9.1a[, 3] <- PANEL9.1a[, 1]^2 + PANEL9.1a[,
+     2]^2
> PANEL9.1a[, 4] <- 1 - PANEL9.1a[, 3]
> round(PANEL9.1a, 5)
```

```
            F1     F2 Comun Var.Esp
Sabor    0.560  0.816 0.979  0.0205
Barganha 0.777 -0.524 0.879  0.1211
Gosto    0.645  0.748 0.976  0.0241
Lanche   0.939 -0.105 0.893  0.1071
Energia  0.798 -0.543 0.932  0.0678
```

Rotação varimax:

```
> varimax(PANEL9.1a[, 1:2])

$loadings

Loadings:
         F1     F2
Sabor           0.989
Barganha  0.937
Gosto     0.130  0.979
Lanche    0.843  0.427
Energia   0.965

                  F1    F2
SS loadings    2.539 2.121
Proportion Var 0.508 0.424
Cumulative Var 0.508 0.932

$rotmat
       [,1]  [,2]
[1,]  0.837 0.548
[2,] -0.548 0.837
```

Continuação do PAINEL9.1:

```
> Exe99.AF <- factanal(covmat = R, factors = 2)
> Exe99.AF

Call:
factanal(factors = 2, covmat = R)
```

```
Uniquenesses:
   Sabor Barganha    Gosto   Lanche  Energia
   0.028    0.237    0.040    0.168    0.052

Loadings:
         Factor1 Factor2
Sabor             0.985
Barganha  0.873
Gosto     0.131   0.971
Lanche    0.817   0.405
Energia   0.973

               Factor1 Factor2
SS loadings      2.396   2.078
Proportion Var   0.479   0.416
Cumulative Var   0.479   0.895

The degrees of freedom for the model is 1 and the fit was 0.0233
```

```
> plot(loadings(Exe99.AF))
> text(loadings(Exe99.AF), dimnames(R)[[2]], pos = 3,
+     cex = 0.6)
```

9.10 Exemplo 9.10

Cargas rotacionadas para dados de preços de ações.

```
> R <- cor(tab8.4)
> TABLE9.8 <- matrix(NA, 5, 5, dimnames = list(dimnames(R)[[1]],
+     c("F1", "F2", "F1star", "F2star", "Psi")))
> TABLE9.8[, 1:2] <- loadings(factanal(covmat = R,
+     factors = 2, rotation = "none"))
> TABLE9.8[, 3:4] <- loadings(factanal(covmat = R,
+     factors = 2, rotation = "varimax"))
```

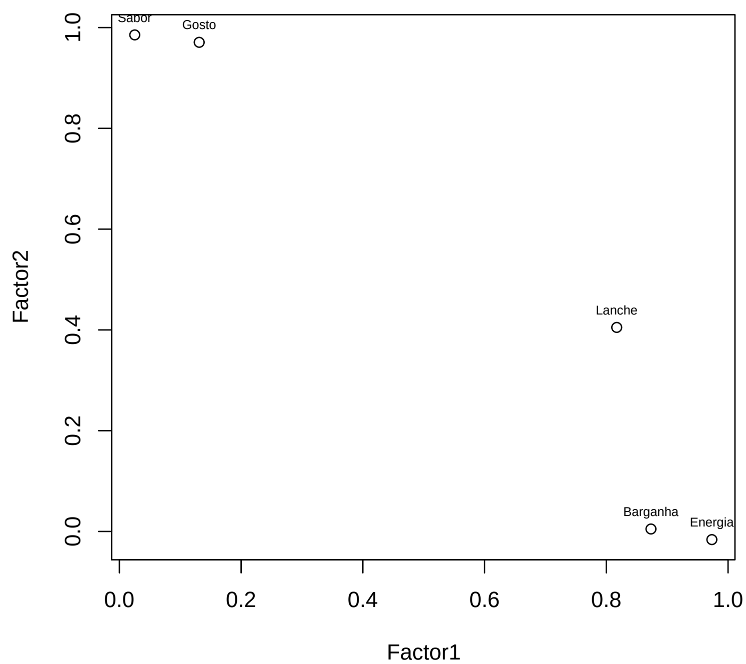


Figura 9.2: Gráficos das cargas fatoriais

```
> TABLE9.8[, 5] <- factanal(covmat = R, factors = 2,
+     rotation = "varimax")$uniquen
> round(TABLE9.8, 3)
```

```
                 F1     F2 F1star F2star   Psi
All.Chem      0.683  0.192  0.601  0.378 0.497
Du_Pont       0.692  0.519  0.849  0.165 0.252
Union_Carbide 0.680  0.251  0.643  0.336 0.474
Exxon         0.621 -0.070  0.365  0.507 0.610
Texaco        0.794 -0.439  0.207  0.884 0.176
```

```
> plot(TABLE9.8[, 3:4])
> text(TABLE9.8[, 3:4], dimnames(R)[[2]], pos = 3,
+     cex = 0.6)
```

9.11 Exemplo 9.11

Cargas rotacionadas para dados de decatlo olímpico.

```
> round(Tab94b, 3)
```

```
            F1     F2    F3     F4   Psi
100m     0.701  0.220 0.701  0.220 0.461
SaltDist 0.733  0.294 0.733  0.294 0.376
LancPeso 0.197  0.939 0.197  0.939 0.080
Alt      0.508  0.297 0.508  0.297 0.653
400m     0.723  0.045 0.723  0.045 0.475
110mBarr 0.539  0.281 0.539  0.281 0.631
Disco    0.164  0.741 0.164  0.741 0.424
SaltVara 0.378  0.192 0.378  0.192 0.820
Dardo    0.108  0.444 0.108  0.444 0.791
1500m    0.240 -0.135 0.240 -0.135 0.924
```

```
> plot(Tab94b[, 1:2])
> text(Tab94b[, 1:2], dimnames(Tab94b)[[1]], pos = 3,
+     cex = 0.6)
```

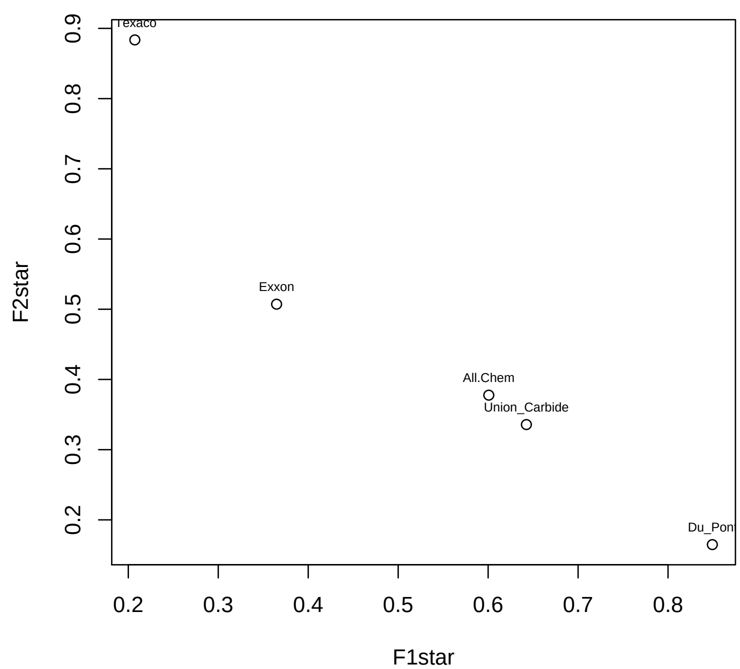


Figura 9.3: Gráficos das cargas fatoriais

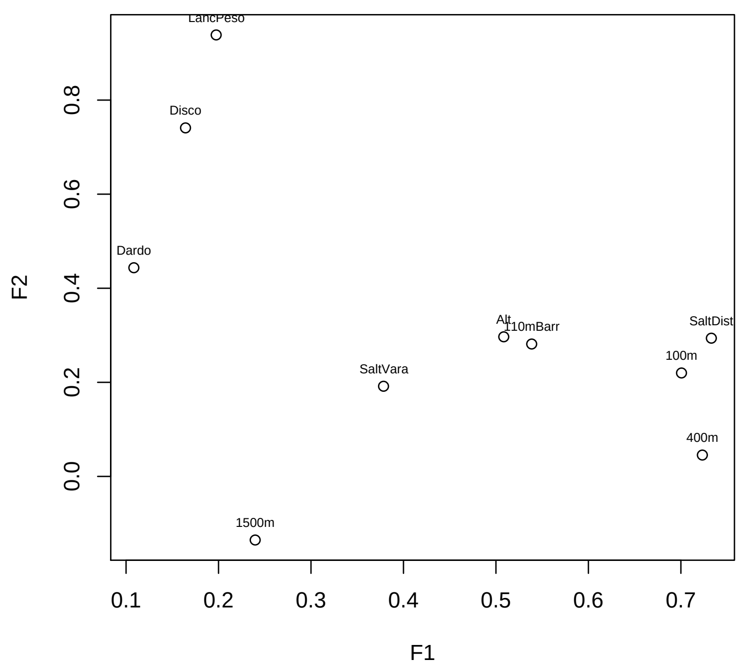


Figura 9.4: Gráficos das cargas fatoriais

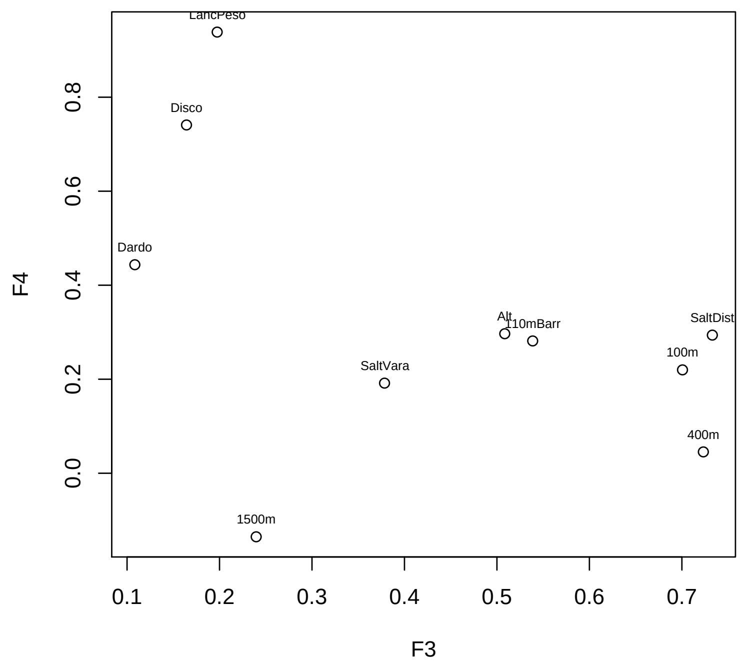


Figura 9.5: Gráficos das cargas fatoriais

```
> plot(Tab94b[, 3:4])
> text(Tab94b[, 3:4], dimnames(Tab94b)[[1]], pos = 3,
+     cex = 0.6)
```

9.12 Exemplo 9.12

Cálculo de escores fatoriais.

```
> tab8.4.AF <- factanal(~All.Chem + Du_Pont + Union_Carbide +
+     Exxon + Texaco, factors = 2, data = tab8.4, scores = "regression")
> Lzstar <- loadings(tab8.4.AF)
> Psizhat <- diag(tab8.4.AF$uniquen)
> z <- matrix(c(0.5, -1.4, -0.2, -0.7, 1.4), ncol = 1)
```

Mínimos quadrados ponderados:

```
> fhat <- solve(t(Lzstar) %*% solve(Psizhat) %*% Lzstar) %*%
+     t(Lzstar) %*% solve(Psizhat) %*% z
> round(fhat, 1)
```

```
        [,1]
Factor1 -1.8
Factor2  1.9
```

Regressão:

```
> round(t(Lzstar) %*% solve(tab8.4.AF$corr) %*% z,
+     2)
```

```
         [,1]
Factor1 -1.20
Factor2  1.41
```

```
> plot(tab8.4.AF$scores)
> abline(h = 0)
> abline(v = 0)
```

```
> dim(Lzstar)
```

```
[1] 5 2
```

9.13 Exemplo 9.14

Análise fatorial dos dados de ossos de galinha.

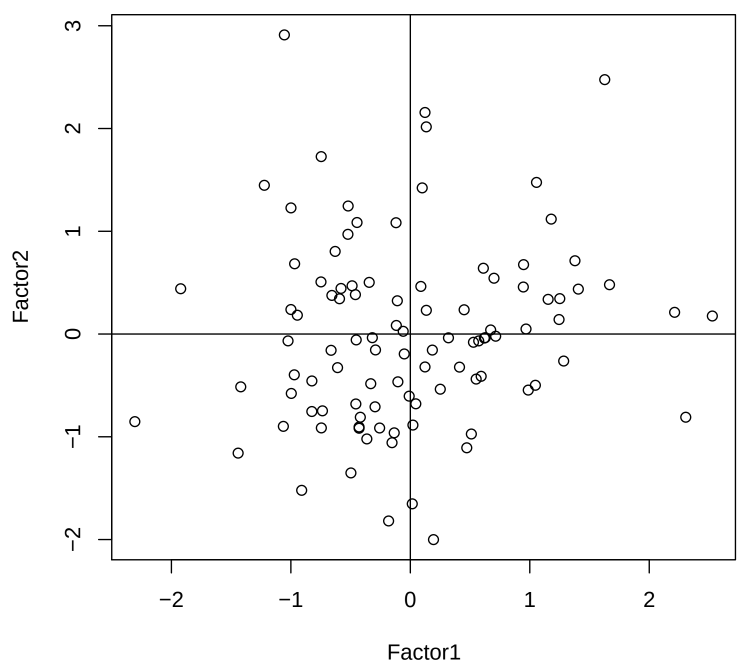


Figura 9.6: Gráficos das cargas fatoriais

```
> n <- 276
```

Dados não fornecidos no CD do livro, obtido na Internet

```
> chicken.dat <- read.table("cbbones.txt", header = F,
+     row.names = 1)
> names(chicken.dat) <- c("CompCran", "LargCran", "CompFemur",
+     "CompTibia", "CompHum", "CompUlna")
> Rmat <- cor(chicken.dat)
> R <- mat.sim(c(1, 0.505, 0.569, 0.602, 0.621, 0.603,
+     1, 0.422, 0.467, 0.482, 0.45, 1, 0.926, 0.877,
+     0.878, 1, 0.874, 0.894, 1, 0.937, 1), 6)
> dimnames(R) <- list(c("CompCran", "LargCran", "CompFemur",
+     "CompTibia", "CompHum", "CompUlna"), c("CompCran",
+     "LargCran", "CompFemur", "CompTibia", "CompHum",
+     "CompUlna"))
```

TABLE9.10.Componentes Principais:

```
> R.eigen <- eigen(R)
```

Componentes principais sem rotação:

```
> Tab9.10a <- matrix(NA, 6, 3, dimnames = list(dimnames(R)[[1]],
+     c("F1", "F2", "F3")))
> for (i in 1:3) Tab9.10a[, i] <- sqrt(R.eigen$values[i]) *
+     R.eigen$vectors[, i]
> Tab9.10a[, 1] <- (-1) * Tab9.10a[, 1]
> round(Tab9.10a, 3)
```

```
             F1     F2     F3
CompCran  0.741  0.350  0.573
LargCran  0.604  0.721 -0.340
CompFemur 0.929 -0.233 -0.075
CompTibia 0.943 -0.174 -0.067
CompHum   0.948 -0.143 -0.045
CompUlna  0.945 -0.189 -0.047
```

Proporção acumulada de variação explicada:

```
> round(cumsum(R.eigen$values[1:3])/6, 3)
```

```
[1] 0.743 0.873 0.950
```

Escores para método de CP:

```
> Z <- scale(chicken.dat)
> fhat <- Z %*% solve(Rmat) %*% Tab9.10a
```

Componentes principais com rotação varimax:

```
> varimax(Tab9.10a)
```

```
$loadings

Loadings:
          F1    F2    F3
CompCran  0.354 0.244 0.903
LargCran  0.234 0.949 0.211
CompFemur 0.921 0.166 0.218
CompTibia 0.903 0.214 0.252
CompHum   0.887 0.229 0.284
CompUlna  0.907 0.192 0.264

                  F1    F2    F3
SS loadings    3.454 1.123 1.120
Proportion Var 0.576 0.187 0.187
Cumulative Var 0.576 0.763 0.950

$rotmat
       [,1]   [,2]  [,3]
[1,]  0.855  0.339 0.394
[2,] -0.482  0.798 0.361
[3,] -0.192 -0.499 0.845
```

Especificidades:

```
> round(1 - rowSums(Tab9.10a^2), 2)
```

```
 CompCran  LargCran CompFemur CompTibia   CompHum  CompUlna
     0.00      0.00      0.08      0.08      0.08      0.07
```

Máxima Verossimilhança:

```
> Tab9.10b <- matrix(NA, 6, 3, dimnames = list(dimnames(R)[[1]],
+     c("F1", "F2", "F3")))
> Exe9.14.AF.1 <- factanal(~CompCran + LargCran + CompFemur +
+     CompTibia + CompHum + CompUlna, data = chicken.dat,
+     factors = 3, rotation = "none", scores = "regression")
```

Matriz de resíduos para Máxima Verossimilhança:

```
> round(R - loadings(Exe9.14.AF.1) %*% t(loadings(Exe9.14.AF.1)) -
+     diag(Exe9.14.AF.1$uniq), 3)
```

```
          CompCran LargCran CompFemur CompTibia CompHum CompUlna
CompCran     0.000   -0.078    -0.011     0.000  -0.001    0.009
LargCran    -0.078    0.000    -0.093    -0.081  -0.102   -0.075
CompFemur   -0.011   -0.093     0.000     0.000   0.000    0.000
CompTibia    0.000   -0.081     0.000     0.000   0.000    0.000
CompHum     -0.001   -0.102     0.000     0.000   0.000    0.000
CompUlna     0.009   -0.075     0.000     0.000   0.000    0.000
```

Figura 9.5: escores de M.V. com rotação

```
> plot(Exe9.14.AF.1$scores)
> abline(h = 0, v = 0)
```

Figura 9.6

```
> Exe9.14.CP.1 <- princomp(~CompCran + LargCran + CompFemur +
+     CompTibia + CompHum + CompUlna, data = chicken.dat)

> plot(c(-3.5, 3), c(-4, 3), type = "n")
> points(Exe9.14.AF.1$scores[, 1], fhat[, 1])
> abline(h = 0, v = 0)
```

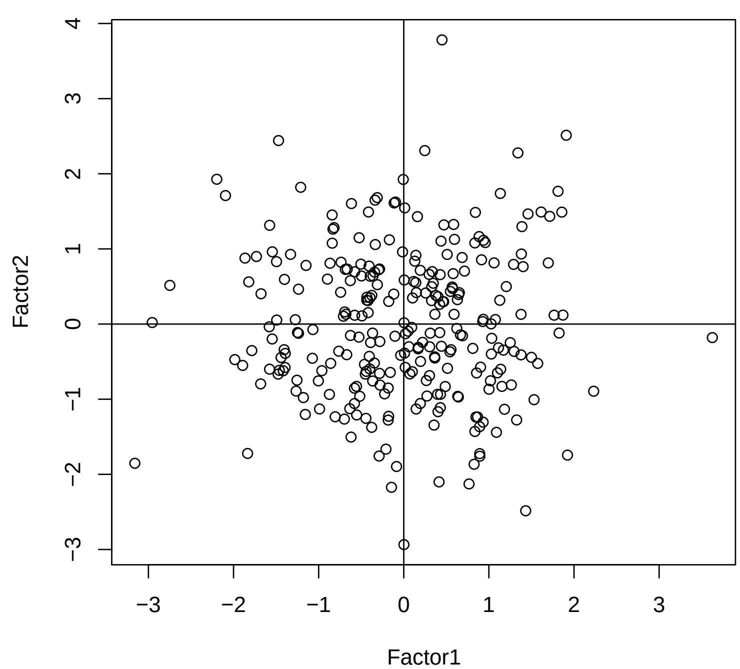


Figura 9.7: Gráficos das cargas fatoriais

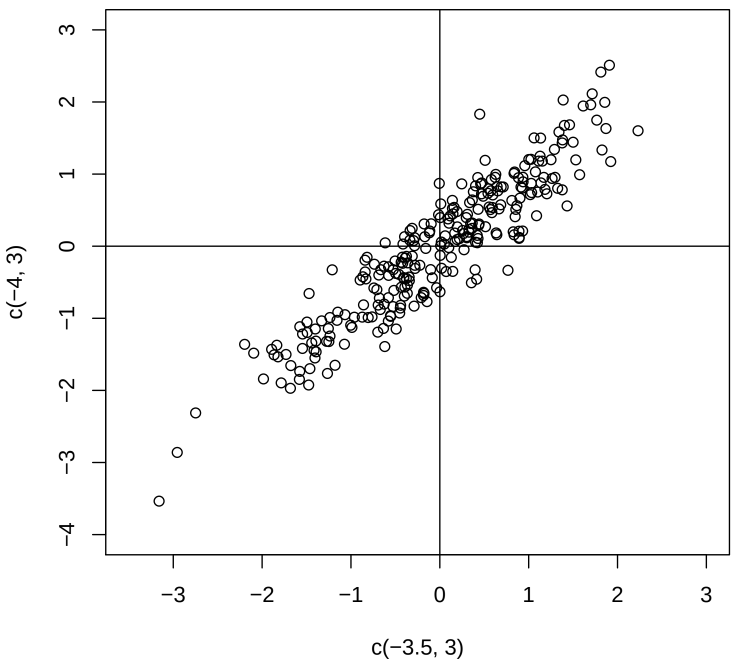


Figura 9.8: Gráficos das cargas fatoriais

Capítulo 10

Classificação

10.1 Exemplo 11.1

Discrminição de proprietários e não-proprietários de carrinhos de aparar grama.

```
> tab11.1 <- read.table("t11-1.dat", col.names = c("Income",
+     "Lote", "Grupo"))
> attach(tab11.1)

> plot(Income, Lote, type = "n", xlab = "Income", ylab = "Lote")
> points(Income[Grupo == 1], Lote[Grupo == 1], col = "red",
+     pch = 16)
> points(Income[Grupo == 2], Lote[Grupo == 2], col = "blue",
+     pch = 16)
> legend("topleft", "(x,y)", c("Prop.", "Nao-prop."),
+     col = c("red", "blue"), pch = 16, bty = "n")
```

10.2 Exemplo 11.3

Classificação com duas populações normais - Σ comum e custos iguais.
Este exemplo estuda a detecção de portadores de hemofilia A. Foram feitas medidas na variáveis $X_1 = log_{10}(atividadeAHF)$ e $X_21 = log_{10}(AntigenoAHF)$. AHF significa fator antihemofílico.

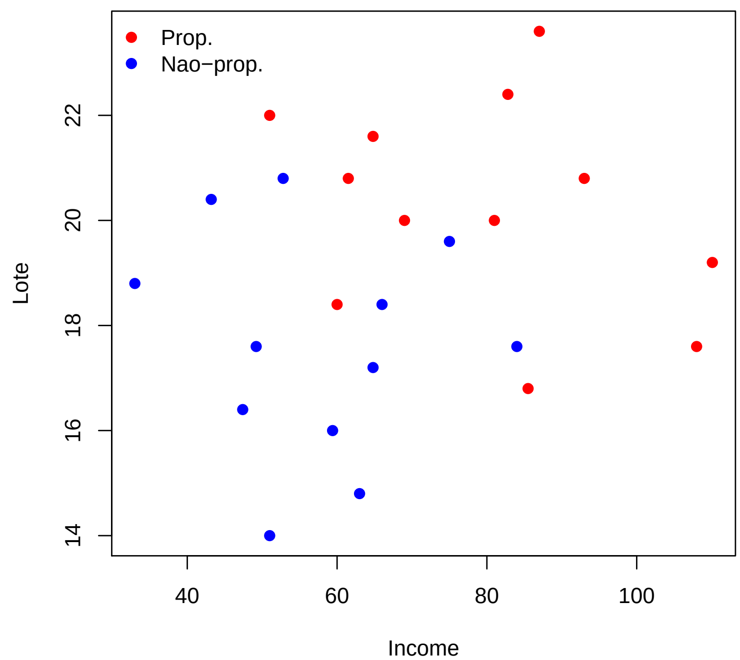


Figura 10.1: Renda vs tamanho do lote de proprietários e não-proprietários de carrinhos de cortar grama

```
> tab11.8 <- read.table("t11-8.dat", col.names = c("Grupo",
+     "atividade AHF", "Antigeno AHF"))
> x1bar <- matrix(c(-0.0065, -0.039), ncol = 1)
> x2bar <- matrix(c(-0.2483, 0.0262), ncol = 1)
> Spool.inv <- matrix(c(131.158, -90.423, -90.423,
+     108.147), 2, 2)
```

Custos iguais e priori uniforme:

```
> yhat <- expression(t(x1bar - x2bar) %*% Spool.inv %*%
+     x)
> x <- x1bar
> y1bar <- eval(yhat)
> x <- x2bar
> y2bar <- eval(yhat)
> c(y1bar, y2bar)

[1]   0.883 -10.096
```

Ponto médio dos y´s

```
> mhat <- (y1bar + y2bar)/2
> mhat

      [,1]
[1,] -4.61
```

Alocar nova observação $x_0 = c(-.210, -.044)$:

```
> x0 <- matrix(c(-0.21, -0.044), ncol = 1)
> x <- x0
> eval(yhat) >= mhat

      [,1]
[1,] FALSE
```

A mulher é classifica como π_2, ou seja portadora. Vamos supor que a distribuição a priori é conhecida: $p_1 = .75$ e $p_2 = .25$:

```
> w.hat <- expression(t(x1bar - x2bar) %*% Spool.inv %*%
+     x0 - 1/2 * t(x1bar - x2bar) %*% Spool.inv %*%
+     (x1bar + x2bar))
> p1 <- 0.75
> p2 <- 0.25
> eval(w.hat) < log(p2/p1)

     [,1]
[1,] TRUE
```

Logo classifica a mulher como portadora.

10.3 Exemplo 11.6

Cálculo de estimativa da taxa de erro usando o método *holdout*.

```
> X1 <- matrix(c(2, 4, 3, 12, 10, 8), 3, 2)
> X2 <- matrix(c(5, 3, 4, 7, 9, 5), 3, 2)
> x1bar <- as.matrix(colMeans(X1))
> x2bar <- as.matrix(colMeans(X2))
> S1 <- cov(X1)
> S2 <- cov(X2)
> Spool <- ((nrow(X1) - 1) * S1 + (nrow(X2) - 1) *
+     S2)/(nrow(X1) + nrow(X2) - 2)
```

Tabela 10.1: Tabela de Confusão

	π_1	π_2
π_1	2.00	1.00
π_2	1.00	2.00

```
> APER <- sum(conf.mat[row(conf.mat) != col(conf.mat)])/sum(conf.mat)
> APER
```

```
[1] 0.333
```

Procedimento holdout:

```
> class.pop1 <- rep(1, nrow(X1))
> for (i in 1:nrow(X1)) {
+     xH <- matrix(X1[i, ], ncol = 1)
+     S1H <- cov(X1[-i, ])
+     x1Hbar <- matrix(colMeans(X1[-i, ]), ncol = 1)
+     SH.pool <- ((nrow(X1) - 2) * S1H + (nrow(X2) -
+         1) * S2)/((nrow(X1) - 2) + (nrow(X2) - 1))
+     d1 <- t(xH - x1Hbar) %*% solve(SH.pool) %*% (xH -
+         x1Hbar)
+     d2 <- t(xH - x2bar) %*% solve(SH.pool) %*% (xH -
+         x2bar)
+     if (d2 < d1)
+         class.pop1[i] <- 2
+ }
> class.pop2 <- rep(2, nrow(X2))
> for (i in 1:nrow(X2)) {
+     xH <- matrix(X2[i, ], ncol = 1)
+     S2H <- cov(X2[-i, ])
+     x2Hbar <- matrix(colMeans(X2[-i, ]), ncol = 1)
+     SH.pool <- ((nrow(X2) - 2) * S2H + (nrow(X1) -
+         1) * S1)/((nrow(X2) - 2) + (nrow(X1) - 1))
+     d2 <- t(xH - x2Hbar) %*% solve(SH.pool) %*% (xH -
+         x2Hbar)
+     d1 <- t(xH - x1bar) %*% solve(SH.pool) %*% (xH -
+         x1bar)
+     if (d1 < d2)
+         class.pop2[i] <- 1
+ }
> TEAP.EST <- (sum(class.pop1 == 2) + sum(class.pop2 ==
+     1))/sum(nrow(X1) + nrow(X2))
> TEAP.EST
```

```
[1] 0.5
```

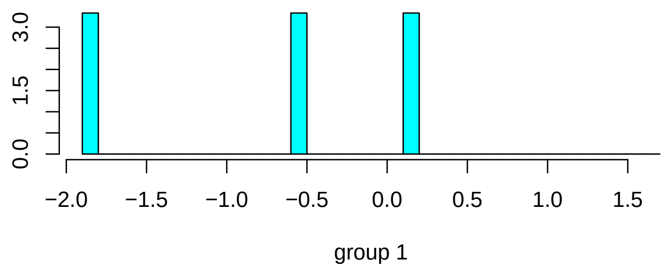


Figura 10.2: Histogramas do único discriminante linear de Fisher por grupo

```
> library(MASS)
> X <- rbind(X1, X2)
> X <- cbind(X, rep(1:2, c(3, 3)))
> X <- as.data.frame(X)
> names(X) <- c("V1", "V2", "Pop")
> exe116.lda <- lda(factor(Pop) ~ V1 + V2, data = X)

> plot(exe116.lda)
```

10.4 Exemplo 11.7

Classificação de salmão do Alasca e do Canadá.

```
> tab11.2 <- read.table("t11-2.dat", col.names = c("Group",
+     "Sexo", "Freshwater", "Marine"))
> tab11.2 <- transform(tab11.2, Group = as.factor(Group))
> lista <- split(tab11.2, tab11.2$Group)
> X1 <- lista[[1]]
> X2 <- lista[[2]]
> x1bar <- as.matrix(colMeans(X1[, 3:4]))
> x2bar <- as.matrix(colMeans(X2[, 3:4]))
> S1 <- cov(X1[, 3:4])
> S2 <- cov(X2[, 3:4])
> Spool <- ((nrow(X1) - 1) * S1 + (nrow(X2) - 1) *
+     S2)/(nrow(X1) + nrow(X2) - 2)
> ahat <- t(x1bar - x2bar) %*% solve(Spool)
> mhat <- ahat %*% (x1bar + x2bar)/2
> what <- expression(ahat %*% matrix(x, ncol = 1) -
+     mhat)
> w.vec1 <- apply(X1[, 3:4], 1, function(x) eval(what))
> w.vec2 <- apply(X2[, 3:4], 1, function(x) eval(what))
> what.result <- matrix(NA, 2, 3, dimnames = list(c("Alaskan",
+     "Canadian"), c("n", "Med", "DP")))
> what.result[, 1] <- c(nrow(X1), nrow(X2))
> what.result[, 2] <- c(mean(w.vec1), mean(w.vec2))
> what.result[, 3] <- c(sd(w.vec1), sd(w.vec2))
```

Estimação da taxa de erro: diretamente usando lda

```
> tab11.2.lda <- lda(Group ~ Freshwater + Marine, data = tab11.2)
> tab11.2.lda

Call:
lda(Group ~ Freshwater + Marine, data = tab11.2)

Prior probabilities of groups:
```

```
  1   2
0.5 0.5

Group means:
  Freshwater Marine
1       98.4    430
2      137.5    367

Coefficients of linear discriminants:
               LD1
Freshwater  0.0446
Marine     -0.0180
```

```
> plot(tab11.2.lda, dimen = 1)
```

```
> tab11.2.pred <- predict(tab11.2.lda)
```

Cálculo da Taxa de Erro Aparente:

```
> (sum(tab11.2.pred$class[1:50] == 2) + sum(tab11.2.pred$class[51:100] ==
+     1))/length(tab11.2.pred$class)
```

```
[1] 0.07
```

Para calcular taxa de erro pelo holdout usa lda com CV=T

```
> tab11.2.lda <- lda(as.factor(Group) ~ Freshwater +
+     Marine, data = tab11.2, CV = T)
> Tab.conf <- rbind(table(tab11.2.lda$class[1:50]),
+     table(tab11.2.lda$class[51:100]))
> dimnames(Tab.conf) <- list(c("Alaskan", "Canadian"),
+     c("Alaskan", "Canadian"))
> Tab.conf
```

```
         Alaskan Canadian
Alaskan       44        6
Canadian       1       49
```

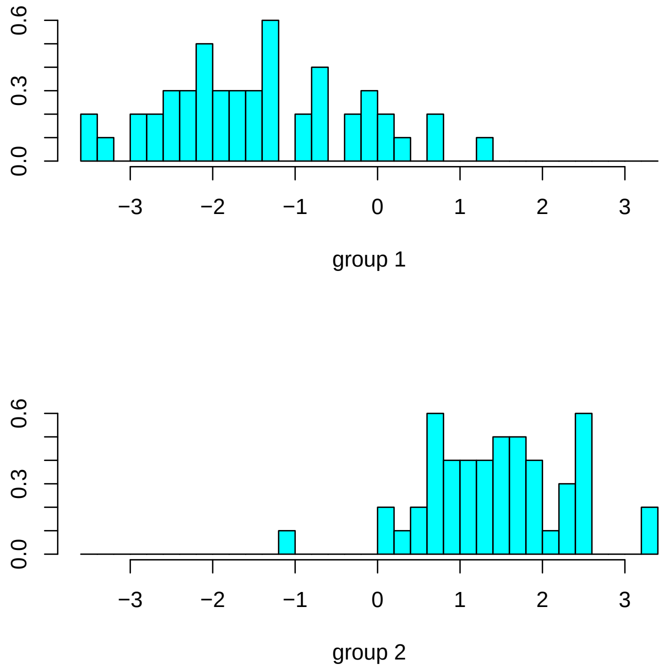


Figura 10.3: Histogramas do único discriminante linear de Fisher por grupo

Figura 11.7

```
> y1.vec <- as.matrix(X1[, 3:4]) %*% matrix(ahat, ncol = 1)
> y2.vec <- as.matrix(X2[, 3:4]) %*% matrix(ahat, ncol = 1)
> mean(y1.vec)

[1] 9.69

> sd(y1.vec)

[1] 3.25

> mean(y2.vec)

[1] 1.40

> sd(y2.vec)

[1] 2.45

> y1grid <- seq(mean(y1.vec) - 3 * sd(y1.vec), mean(y1.vec) +
+     3 * sd(y1.vec), by = 0.1)
> y2grid <- seq(mean(y2.vec) - 3 * sd(y2.vec), mean(y2.vec) +
+     3 * sd(y2.vec), by = 0.1)
> plot(c(y1grid, y2grid), c(dnorm(y1grid, mean = mean(y1.vec),
+     sd = sd(y1.vec)), dnorm(y2grid, mean = mean(y2.vec),
+     sd = sd(y2.vec))), type = "n", xlab = "y", ylab = "dens",
+     main = "Densidades de y para os dois grupos")
> lines(y1grid, dnorm(y1grid, mean = mean(y1.vec),
+     sd = sd(y1.vec)), type = "l")
> lines(y2grid, dnorm(y2grid, mean = mean(y2.vec),
+     sd = sd(y2.vec)), type = "l")
> abline(v = mhat)
```

```
[1] 9.69

[1] 3.25

[1] 1.40

[1] 2.45
```

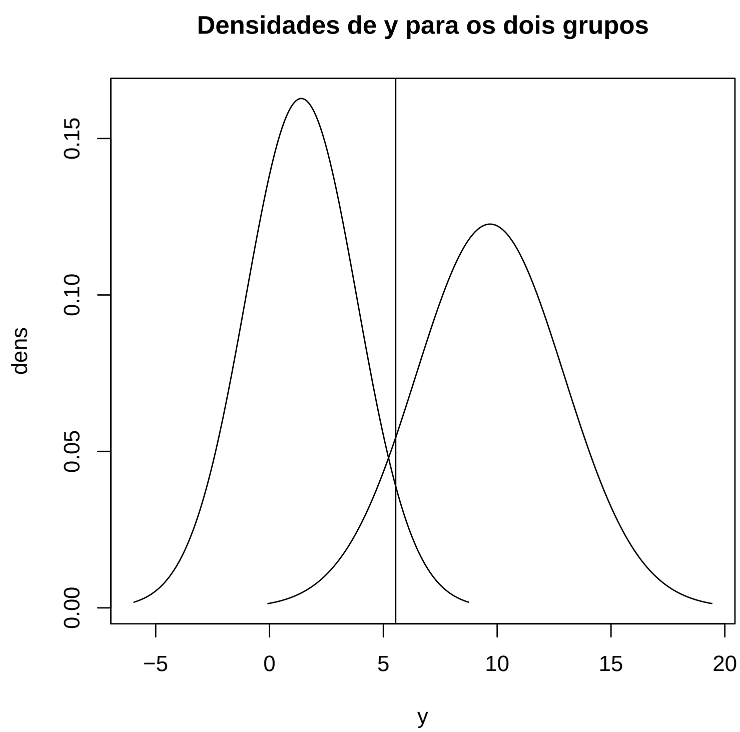


Figura 10.4: Probabilidades de classificação incorreta baseada em Y

10.5 Exemplo 11.8

Discriminante linear de Fisher para dados de hemofilia.

```
> x1bar <- matrix(c(-0.0065, -0.039), ncol = 1)
> x2bar <- matrix(c(-0.2483, 0.0262), ncol = 1)
> Spool.inv <- matrix(c(131.158, -90.423, -90.423,
+     108.147), 2, 2)
> yhat <- expression(t(x1bar - x2bar) %*% Spool.inv %*%
+     x)
```

Separação máxima:

```
> D2 <- t(x1bar - x2bar) %*% Spool.inv %*% (x1bar -
+     x2bar)
> D2
```

```
     [,1]
[1,] 11.0
```

10.6 Exemplo 11.9

Classificação de uma nova observação em uma de três populações conhecidas.

```
> Custos <- matrix(c(0, 10, 50, 500, 0, 200, 100, 50,
+     0), 3, 3, dimnames = list(c("pi1", "pi2", "pi3"),
+     c("pi1", "pi2", "pi3")))
> prior <- c(0.05, 0.6, 0.35)
> dens.x0 <- c(0.01, 0.85, 2)
```

Cálculo dos custos esperados:
Classificar como $k = 1$

```
> prior[2] * dens.x0[2] * Custos[1, 2] + prior[3] *
+     dens.x0[3] * Custos[1, 3]
```

```
[1] 325
```

Classificar como $k = 2$

```
> prior[1] * dens.x0[1] * Custos[2, 1] + prior[3] *
+     dens.x0[3] * Custos[2, 3]
```

```
[1] 35
```

Classificar como $k = 3$

```
> prior[1] * dens.x0[1] * Custos[3, 1] + prior[2] *
+     dens.x0[2] * Custos[3, 2]
```

```
[1] 102
```

O menor custo esperado de classificação incorreta obtido para k=2, se os custos fossem iguais:

```
> (prior * dens.x0)[1]
```

```
[1] 5e-04
```

```
> (prior * dens.x0)[2]
```

```
[1] 0.51
```

```
> (prior * dens.x0)[3]
```

```
[1] 0.7
```

Deve alocar em π_3.
Cálculo das posteriori

```
> (prior * dens.x0)[1]/sum(prior * dens.x0)
```

```
[1] 0.000413
```

```
> (prior * dens.x0)[2]/sum(prior * dens.x0)
```

```
[1] 0.421
```

```
> (prior * dens.x0)[3]/sum(prior * dens.x0)
```

```
[1] 0.578
```

10.7 Exemplo 11.10

Cálculo de escores discriminantes amostrais, supondo uma matriz de covariância comum.

```
> p <- c(0.25, 0.25, 0.5)
> x0 <- matrix(c(-2, -1), ncol = 1)
> X1 <- matrix(c(-2, 0, -1, 5, 3, 1), 3, 2)
> n1 <- nrow(X1)
> x1bar <- as.matrix(colMeans(X1))
> S1 <- cov(X1)
> X2 <- matrix(c(0, 2, 1, 6, 4, 2), 3, 2)
> n2 <- nrow(X2)
> x2bar <- as.matrix(colMeans(X2))
> S2 <- cov(X2)
> X3 <- matrix(c(1, 0, -1, -2, 0, -4), 3, 2)
> n3 <- nrow(X3)
> x3bar <- as.matrix(colMeans(X3))
> S3 <- cov(X3)
> Spool <- ((n1 - 1) * S1 + (n2 - 1) * S2 + (n3 - 1) *
+     S3)/(n1 + n2 + n3 - 3)
> Spool.inv <- solve(Spool)
```

Cálculo dos escores:

```
> d1.x0 <- log(p[1]) + t(x1bar) %*% Spool.inv %*% x0 -
+     (t(x1bar) %*% Spool.inv %*% x1bar)/2
> d2.x0 <- log(p[2]) + t(x2bar) %*% Spool.inv %*% x0 -
+     (t(x2bar) %*% Spool.inv %*% x2bar)/2
> d3.x0 <- log(p[3]) + t(x3bar) %*% Spool.inv %*% x0 -
+     (t(x3bar) %*% Spool.inv %*% x3bar)/2
> d1.x0

      [,1]
[1,] -1.94

> d2.x0
```

```
      [,1]
[1,] -8.16
```

```
> d3.x0
```

```
      [,1]
[1,] -0.35
```

Aloca em π_3.

10.8 Exemplo 11.11

Classificação de aluno potencial na pós-graduação da *business-school*.

```
> tab11.6 <- read.table("t11-6.dat", col.names = c("GPA",
+     "GMAT", "Grupo"))
> tab11.6 <- transform(tab11.6, Grupo = factor(Grupo,
+     labels = c("A", "B", "C")))
```

Gráfico dos dados:

```
> plot(tab11.6$GPA, tab11.6$GMAT, type = "n", xlab = "GPA",
+     ylab = "GMAT")
> text(tab11.6$GPA, tab11.6$GMAT, labels = tab11.6$Grupo,
+     cex = 0.8)
> legend("topleft", c("Admite", "N. Admite", "Borda"),
+     pch = c("A", "B", "C"))
```

Discriminação Linear:

```
> tab11.6.lda <- lda(Grupo ~ GPA + GMAT, data = tab11.6,
+     prior = c(1/3, 1/3, 1/3))
> names(tab11.6.lda)
```

```
 [1] "prior"   "counts"  "means"   "scaling" "lev"     "svd"
 [7] "N"       "call"    "terms"   "xlevels"
```

Teste de igualdade de médias:

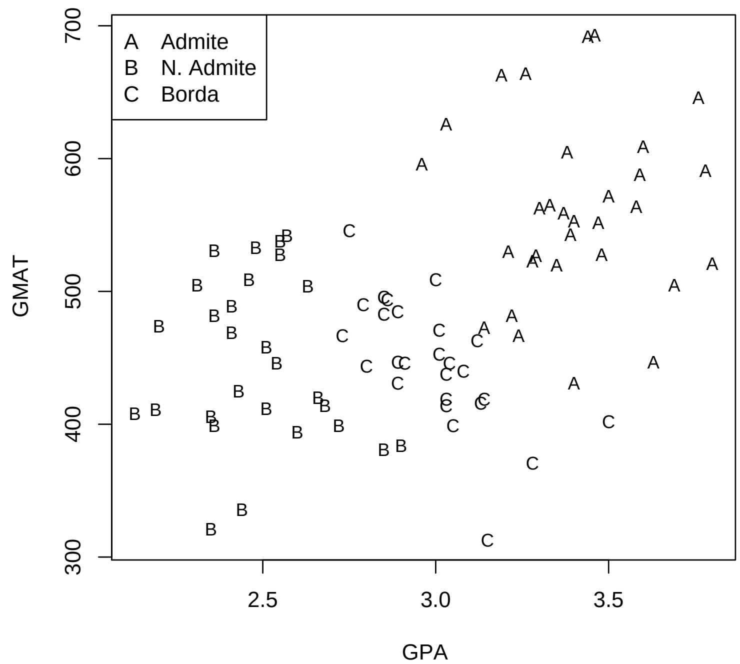


Figura 10.5: Diagrama de dispersão ($x_1 = GPA, x_2 = GMAT$) de candidatos à pós-graduação na school of business, classifcados como "admite", "não admite" ou "borda"

```
> by(tab11.6, tab11.6$Grupo, function(t) cov(t[, 1:2]))

tab11.6$Grupo: A
        GPA     GMAT
GPA  0.0436 5.81e-02
GMAT 0.0581 4.62e+03
------------------------------------------------
tab11.6$Grupo: B
         GPA    GMAT
GPA   0.0336   -1.19
GMAT -1.1920 3891.25
------------------------------------------------
tab11.6$Grupo: C
         GPA   GMAT
GPA   0.0297   -5.4
GMAT -5.4038 2246.9

> tab11.6.manova <- manova(cbind(GPA, GMAT) ~ Grupo,
+     data = tab11.6)
> summary(tab11.6.manova, test = "Wilks")

           Df Wilks approx F num Df den Df Pr(>F)
Grupo       2   0.1     73.4      4    162 <2e-16 ***
Residuals  82
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

> summary(tab11.6.manova, test = "Pillai")

           Df Pillai approx F num Df den Df Pr(>F)
Grupo       2    1.0     41.8      4    164 <2e-16 ***
Residuals  82
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

> summary(tab11.6.manova, test = "Hotelling")
```

```
          Df Hotelling-Lawley approx F num Df den Df Pr(>F)
Grupo      2              5.8    116.7      4    160 <2e-16 ***
Residuals  82
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

> summary(tab11.6.manova, test = "Roy")

          Df   Roy approx F num Df den Df Pr(>F)
Grupo      2   5.6    231.5      2     82 <2e-16 ***
Residuals 82
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

> summary.aov(tab11.6.manova)

 Response GPA :
            Df Sum Sq Mean Sq F value Pr(>F)
Grupo        2  12.50    6.25     173 <2e-16 ***
Residuals   82   2.96    0.04
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

 Response GMAT :
            Df Sum Sq Mean Sq F value  Pr(>F)
Grupo        2 258471  129236    35.4 8.5e-12 ***
Residuals   82 299784    3656
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

> tab11.6.lda.pred <- predict(tab11.6.lda)
> ind <- c(2, 3, 24, 31, 58, 59, 66)
> From <- tab11.6[ind, "Grupo"]
> Classif <- tab11.6.lda.pred$class[ind]
> round(tab11.6.lda.pred$posterior[c(2, 3, 24, 31,
+     58, 59, 66), ], 4)
```

```
        A      B     C
2  0.1202 0.0020 0.878
3  0.3654 0.0004 0.634
24 0.4766 0.0000 0.523
31 0.2964 0.0004 0.703
58 0.0001 0.2450 0.755
59 0.0001 0.1326 0.867
66 0.5336 0.0000 0.466
```

```
> table(tab11.6[, "Grupo"], tab11.6.lda.pred$class)
```

```
     A  B  C
  A 27  0  4
  B  0 26  2
  C  1  0 25
```

```
> result <- matrix(NA, 7, 6, dimnames = list(NULL,
+     c("Obs", "From", "Class", "admit", "border",
+         "notadmit")))
> result[, 1] <- ind
> result[, 2] <- From
> result[, 3] <- Classif
> result[, 4:6] <- round(tab11.6.lda.pred$posterior[c(2,
+     3, 24, 31, 58, 59, 66), ], 4)
> result
```

```
     Obs From Class  admit border notadmit
[1,]   2    1     3 0.1202 0.0020    0.878
[2,]   3    1     3 0.3654 0.0004    0.634
[3,]  24    1     3 0.4766 0.0000    0.523
[4,]  31    1     3 0.2964 0.0004    0.703
[5,]  58    2     3 0.0001 0.2450    0.755
[6,]  59    2     3 0.0001 0.1326    0.867
[7,]  66    3     1 0.5336 0.0000    0.466
```

10.9 Exemplo 11.12

Classificação efetiva com menos variáveis.

3 grupos: 1-Iris Setosa; 2-Iris versicolor; 3-Iris virginica
4 variáveis:X1- comp. sépala; X2 -largura de sépala; X4- comp. pétala; X4 -largura da pétala

```
> data(iris3)
> Iris <- data.frame(rbind(iris3[, , 1], iris3[, ,
+     2], iris3[, , 3]), Sp = rep(c("s", "c", "v"),
+     rep(50, 3)))
```

Separar 75 obs. ao acaso para treinamento:

```
> train <- sample(1:150, 75)
> table(Iris$Sp[train])

 c  s  v
26 25 24
```

Determinar regra de classificação usando amostra de treinamento:

```
> z <- lda(Sp ~ ., Iris, prior = c(1, 1, 1)/3, subset = train)
```

Predizer na amostra de teste:

```
> predict(z, Iris[-train, ])$class

 [1] s s s s s s s s s s s s s s s s s s s s s s s s s c c c c c
[31] c c c c c c c c c c c c c c c c c c c c v v v v v v v v v v
[61] v v v v v v v v v v v v v v v
Levels: c s v
```

Tabela na ordem: c-versicolor; s-setosa; v-virginica

```
> z0 <- lda(Sp ~ ., Iris, CV = T)
> mat.result <- matrix(NA, 3, 3)
> p.corr <- numeric(3)
```

```
> mat.result[1, ] <- table((z0$class)[51:100])
> mat.result[2, ] <- table((z0$class)[1:50])
> mat.result[3, ] <- table((z0$class)[101:150])
> p.corr[1] <- mat.result[1, 1]/sum(mat.result[1, ])
> p.corr[2] <- mat.result[2, 2]/sum(mat.result[2, ])
> p.corr[3] <- mat.result[3, 3]/sum(mat.result[3, ])
> mat.result <- cbind(mat.result, 100 * p.corr)
> dimnames(mat.result) <- list(c("Versicolor", "Setosa",
+     "Virginica"), c("Versicolor", "Setosa", "Virginica",
+     "p.corr"))
```

Estimativa do valor esperado da taxa de erro real:

```
> TER <- (sum(mat.result[1, c(2, 3)]) + sum(mat.result[2,
+     c(1, 3)]) + sum(mat.result[3, c(1, 2)]))/150

> z1 <- lda(Sp ~ Sepal.L., Iris, CV = T)$class
> z2 <- lda(Sp ~ Sepal.W., Iris, CV = T)$class
> z3 <- lda(Sp ~ Petal.L., Iris, CV = T)$class
> z4 <- lda(Sp ~ Petal.W., Iris, CV = T)$class
> z12 <- lda(Sp ~ Sepal.L. + Sepal.W., Iris, CV = T)$class
> z13 <- lda(Sp ~ Sepal.L. + Petal.L., Iris, CV = T)$class
> z14 <- lda(Sp ~ Sepal.L. + Petal.W., Iris, CV = T)$class
> z23 <- lda(Sp ~ Sepal.W. + Petal.L., Iris, CV = T)$class
> z24 <- lda(Sp ~ Sepal.W. + Petal.W., Iris, CV = T)$class
> z34 <- lda(Sp ~ Petal.L. + Petal.W., Iris, CV = T)$class
```

Cálculo das taxas de erro:

```
> erro.exp <- expression(round(sum(M[row(M) != col(M)])/150,
+     3))
> listao <- list(z1, z2, z3, z4, z12, z13, z14, z23,
+     z24, z34)
> sapply(listao, function(z) {
+     M <- table(Iris$Sp, z)
+     eval(erro.exp)
+ })
```

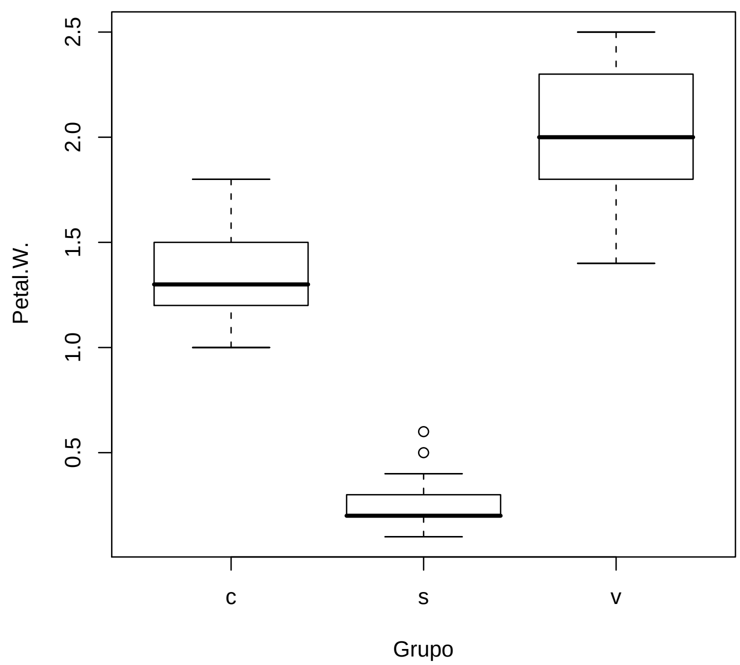


Figura 10.6: Box-plot da largura da pétala para os três espécies

```
 [1] 0.253 0.480 0.067 0.040 0.207 0.040 0.047 0.047 0.040 0.040
```

Variável X_4 funciona bem, ganha pouco introduzindo mais variáveis.

```
> with(Iris, plot(as.factor(Sp), Petal.W., xlab = "Grupo",
+     ylab = "Petal.W."))
```

10.10 Exemplo 11.13

Cálculo de discriminantes lineares de Fisher para três populações.

```
> X1 <- matrix(c(-2, 0, -1, 5, 3, 1), 3, 2)
> n1 <- nrow(X1)
> x1bar <- as.matrix(colMeans(X1))
> S1 <- cov(X1)
> X2 <- matrix(c(0, 2, 1, 6, 4, 2), 3, 2)
> n2 <- nrow(X2)
> x2bar <- as.matrix(colMeans(X2))
> S2 <- cov(X2)
> X3 <- matrix(c(1, 0, -1, -2, 0, -4), 3, 2)
> n3 <- nrow(X3)
> x3bar <- as.matrix(colMeans(X3))
> S3 <- cov(X3)
> X <- rbind(X1, X2, X3)
> grupo <- rep(1:3, rep(3, 3))
> X <- cbind(X, grupo)
> X <- as.data.frame(X)
> names(X) <- c("V1", "V2", "grupo")
> lda(grupo ~ V1 + V2, data = X)

Call:
lda(grupo ~ V1 + V2, data = X)

Prior probabilities of groups:
    1     2     3
0.333 0.333 0.333

Group means:
  V1 V2
1 -1  3
2  1  4
3  0 -2

Coefficients of linear discriminants:
      LD1    LD2
V1 -0.386 -0.938
V2 -0.495  0.112
```

```
Proportion of trace:
 LD1  LD2
0.76 0.24
```

Comparar com os coeficientes na pág.632 do livro.

10.11 Exemplo 11.14

Discriminantes de Fisher para dados de óleo crú.

```
> tab11.7 <- read.table("t11-7.dat", col.names = c("Van.",
+     "Iron", "Ber.", "Hidr.", "ArHidr.", "Grupo"))
> tab11.7 <- transform(tab11.7, Iron = sqrt(Iron),
+     Ber. = sqrt(Ber.), Hidr. = 1/Hidr.)
> by(tab11.7, tab11.7$Grupo, function(t) colMeans(t[1:4]))

tab11.7$Grupo: SubMuli
 Van.  Iron  Ber. Hidr.
4.445 5.667 0.344 0.157
------------------------------------------------
tab11.7$Grupo: Upper
 Van.  Iron  Ber. Hidr.
7.226 4.634 0.598 0.223
------------------------------------------------
tab11.7$Grupo: Wilhelm
 Van.  Iron  Ber. Hidr.
3.229 6.586 0.303 0.150

> xbar <- round(matrix(colMeans(tab11.7[, -6]), ncol = 1),
+     3)
```

Discriminantes lineares de Fisher centrados na média:

```
> tab11.7.lda <- lda(Grupo ~ . - Grupo, data = tab11.7)
> tab11.7.lda$scaling
```

```
           LD1    LD2
Van.     0.312 -0.169
Iron    -0.710  0.245
Ber.     2.764  2.046
Hidr.   11.809 24.453
ArHidr. -0.235  0.378
```

```
> tab11.7.ld <- predict(tab11.7.lda, dimen = 2)$x
```

```
> Grp <- tab11.7$Grupo
> eqscplot(tab11.7.ld, type = "n", xlab = "DL1", ylab = "DL2")
> points(tab11.7.ld[Grp == "Wilhelm", ], col = "red",
+     pch = 18)
> points(tab11.7.ld[Grp == "SubMuli", ], col = "blue",
+     pch = 18)
> points(tab11.7.ld[Grp == "Upper", ], col = "green",
+     pch = 18)
> legend("topleft", c("Wilhelm", "SubMuli", "Upper"),
+     col = c("red", "blue", "green"), pch = 18, bty = "n")
```

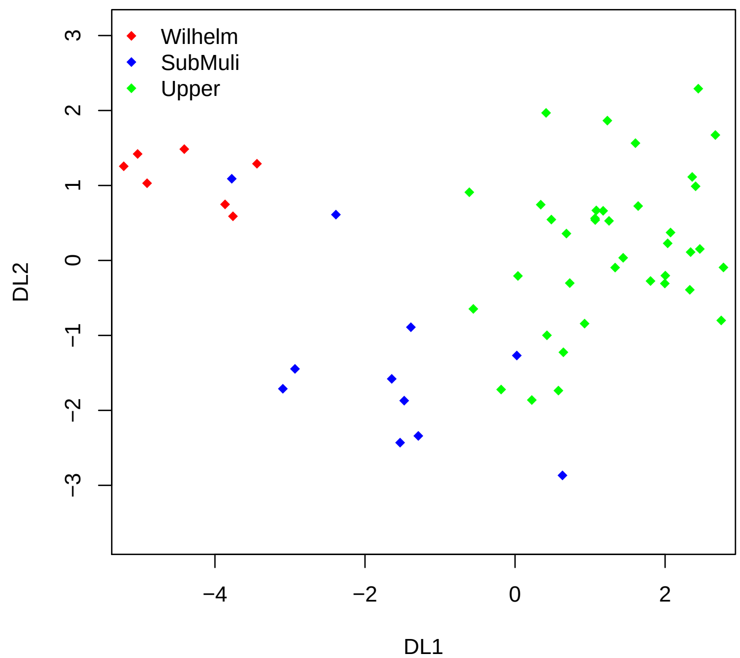


Figura 10.7: Amostras de óleo crú no espaço discriminante

Capítulo 11

Análise de conglomerados

11.1 Exemplo 12.1

Conglomeração por sombra uma matriz de distância.

```
> tab12.5 <- read.table("t12-5.dat")
> names(tab12.5)[9] <- "Empresas"
```

Mudança de escala dos dados:

```
> mat.dat <- as.matrix(tab12.5[, 1:8])
> mat.dat <- scale(mat.dat)
```

Calcula distância euclidiana - TABLE 12.1

```
> dist.mat.empr <- dist(mat.dat, diag = T)
> round(dist.mat.empr, 2)
```

```
      1    2    3    4    5    6    7    8    9   10   11   12
1  0.00
2  3.10 0.00
3  3.68 4.92 0.00
4  2.46 2.16 4.11 0.00
5  4.12 3.85 4.47 4.13 0.00
6  3.61 4.22 2.99 3.20 4.60 0.00
7  3.90 3.45 4.22 3.97 4.60 3.35 0.00
8  2.74 3.89 4.99 3.69 5.16 4.91 4.36 0.00
9  3.25 3.96 2.75 3.75 4.49 3.73 2.80 3.59 0.00
10 3.10 2.71 3.93 1.49 4.05 3.83 4.51 3.67 3.57 0.00
11 3.49 4.79 5.90 4.86 6.46 6.00 6.00 3.46 5.18 5.08 0.00
12 3.22 2.43 4.03 3.50 3.60 3.74 1.66 4.06 2.74 3.94 5.21 0.00
13 3.96 3.43 4.39 2.58 4.76 4.55 5.01 4.14 3.66 1.41 5.31 4.50
14 2.11 4.32 2.74 3.23 4.82 3.47 4.91 4.34 3.82 3.61 4.32 4.34
15 2.59 2.50 5.16 3.19 4.26 4.07 2.93 3.85 4.11 4.26 4.74 2.33
16 4.03 4.84 5.26 4.97 5.82 5.84 5.04 2.20 3.63 4.53 3.43 4.62
```

```
17 4.40 3.62 6.36 4.89 5.63 6.10 4.58 5.43 4.90 5.48 4.75 3.50
18 1.88 2.90 2.72 2.65 4.34 2.85 2.95 3.24 2.43 3.07 3.95 2.45
19 2.41 4.63 3.18 3.46 5.13 2.58 4.52 4.11 4.11 4.13 4.52 4.41
20 3.17 3.00 3.73 1.82 4.39 2.91 3.54 4.09 2.95 2.05 5.35 3.43
21 3.45 2.32 5.09 3.88 3.64 4.63 2.68 3.98 3.74 4.36 4.88 1.38
22 2.51 2.42 4.11 2.58 3.77 4.03 4.00 3.24 3.21 2.56 3.44 3.00
     13   14   15   16   17   18   19   20   21   22
1
2
3
4
5
6
7
8
9
10
11
12
13 0.00
14 4.39 0.00
15 5.10 4.24 0.00
16 4.41 5.17 5.18 0.00
17 5.61 5.56 3.40 5.56 0.00
18 3.78 2.30 3.00 3.97 4.43 0.00
19 5.01 1.88 4.03 5.23 6.09 2.47 0.00
20 2.23 3.74 3.78 4.82 4.87 2.92 3.90 0.00
21 4.94 4.93 2.10 4.57 3.10 3.19 4.97 4.15 0.00
22 2.74 3.51 3.35 3.46 3.63 2.55 3.97 2.62 3.01 0.00
```

11.2 Exemplo 12.3

Medidas de similaridade de 11 línguas.

```
> tab12.4 <- scan("t12-4.dat")
> tab.dist <- matrix(NA, 11, 11)
> tab.dist[row(tab.dist) <= col(tab.dist)] <- tab12.4
> tab.dist[row(tab.dist) > col(tab.dist)] <- 0
> tab.dist <- tab.dist + t(tab.dist) - diag(diag(tab.dist))
```

11.3 Exemplo 12.4

Conglomeração usando single linkage.

```
> Dist <- mat.sim(c(0, 9, 3, 6, 11, 0, 7, 5, 10, 0,
+     9, 2, 0, 8, 0), 5)
> D.dist <- as.dist(Dist, diag = T)
> Dist.clus <- hclust(D.dist, method = "single")

> Dist.clus$merge
```

```
     [,1] [,2]
[1,]   -3   -5
[2,]   -1    1
[3,]   -2   -4
[4,]    2    3
```

```
> Dist.clus$height
```

```
[1] 2 3 5 6
```

```
> Dist.clus$order
```

```
[1] 1 3 5 2 4
```

Ordem de fusões:

			1,2,3,4,5
1	-3	-5	1,2,(3,5),4
2	-1	1	(1,3,5),2,4
3	-2	-4	(1,3,5),(2,4)
4	2	3	(1,3,5,2,4)

11.4 Exemplo 12.5

Conglomeração por *single linkage* de 11 línguas.

```
> Dist.lang <- 10 - tab.dist
> dimnames(Dist.lang) <- list(c("E", "N", "Da", "Du",
+     "G", "Fr", "Sp", "I", "P", "H", "Fi"), c("E",
+     "N", "Da", "Du", "G", "Fr", "Sp", "I", "P", "H",
+     "Fi"))
> Lang.dist <- as.dist(Dist.lang, diag = T)
> Lang.clus <- hclust(Lang.dist, method = "single")
> Lang.clus$merge
```

```
      [,1] [,2]
 [1,]   -2   -3
```

```
 [2,]   -6   -8
 [3,]   -7    2
 [4,]   -1    1
 [5,]   -9    3
 [6,]   -5    4
 [7,]   -4    6
 [8,]    5    7
 [9,]  -10    8
[10,]  -11    9
```

```
> plot(Lang.clus)
```

11.5 Exemplo 12.6

Conglomeração usando *complete linkage*.

```
> Dist <- mat.sim(c(0, 9, 3, 6, 11, 0, 7, 5, 10, 0,
+     9, 2, 0, 8, 0), 5)
> D.dist <- as.dist(Dist, diag = T)
> Dist.clus <- hclust(D.dist, method = "complete")
> Dist.clus$merge
```

```
     [,1] [,2]
[1,]   -3   -5
[2,]   -2   -4
[3,]   -1    2
[4,]    1    3
```

```
> Dist.clus$height
```

```
[1]  2  5  9 11
```

```
> Dist.clus$order
```

```
[1] 3 5 1 2 4
```

```
> plot(Dist.clus)
```

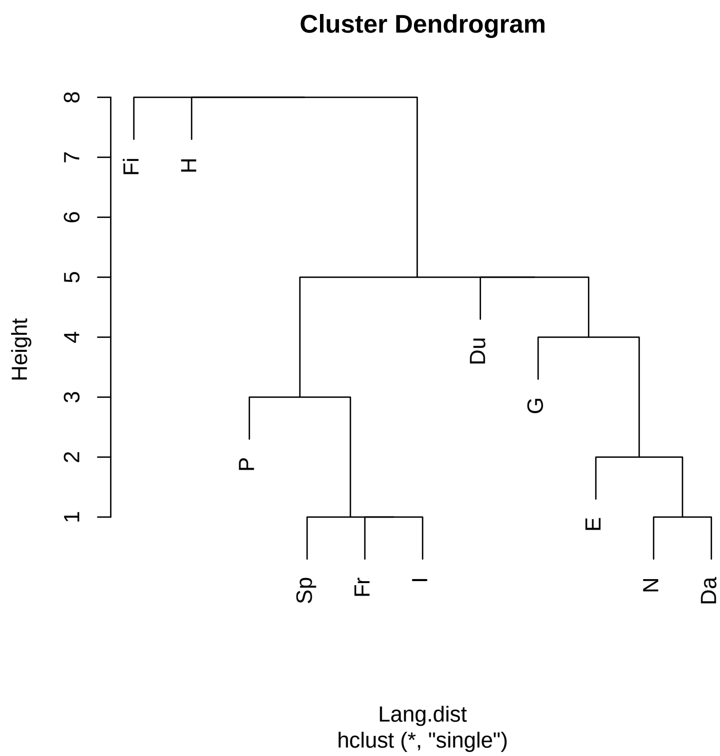


Figura 11.1: Gráficos de resíduos

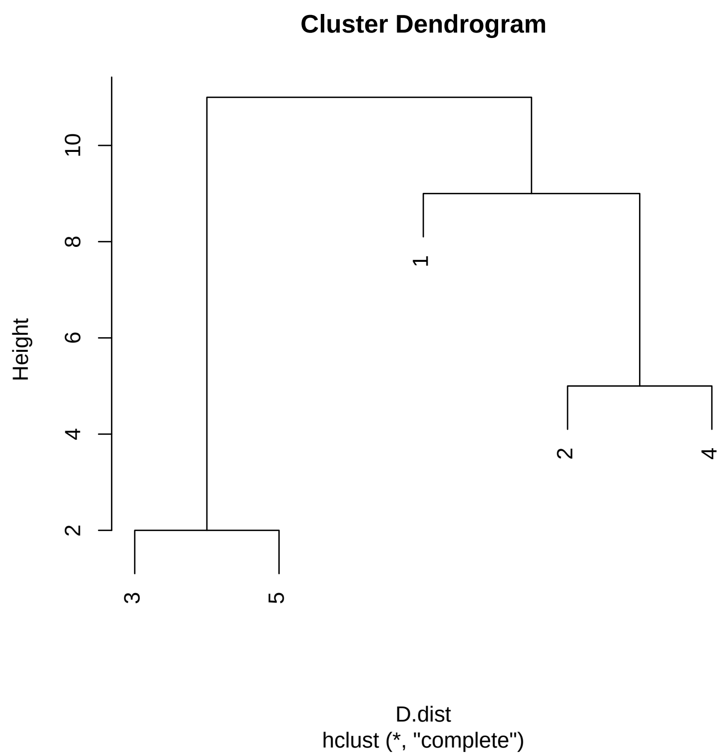


Figura 11.2: Gráficos de resíduos

11.6 Exemplo 12.7

Conglomeração de *complete linkage* de 11 línguas.

```
> Lang.clus <- hclust(Lang.dist, method = "complete")
> Lang.clus$merge
```

```
      [,1] [,2]
 [1,]   -2   -3
 [2,]   -6   -8
 [3,]   -1    1
 [4,]   -7    2
 [5,]   -4   -5
 [6,]   -9    4
 [7,]    3    5
 [8,]  -10  -11
 [9,]    7    8
[10,]    6    9
```

```
> plot(Lang.clus)
```

Ordem de fusões:

			1,2,3,4,5,6,7,8,9,10,11
1	-2	-3	1,(2,3),4,5,6,7,8,9,10,11
2	-6	-8	1,(2,3),4,5,(6,8),7,9,10,11
3	-1	1	(1,2,3),4,5,(6,8),7,9,10,11
4	-7	2	(1,2,3),4,5,(6,8,7),9,10,11
5	-4	-5	(1,2,3),(4,5),(6,8,7),9,10,11
6	-9	4	(1,2,3),(4,5),(6,8,7,9),10,11
7	3	5	(1,2,3,4,5),(6,8,7,9),10,11
8	-10	-11	(1,2,3,4,5),(6,8,7,9),(10,11)
9	7	8	(1,2,3,4,5,10,11),(6,8,7,9)
10	6	9	(1,2,3,4,5,10,11,6,8,7,9)

11.7 Exemplo 12.8

Variáveis de conglomeração usando *complete linkage*.

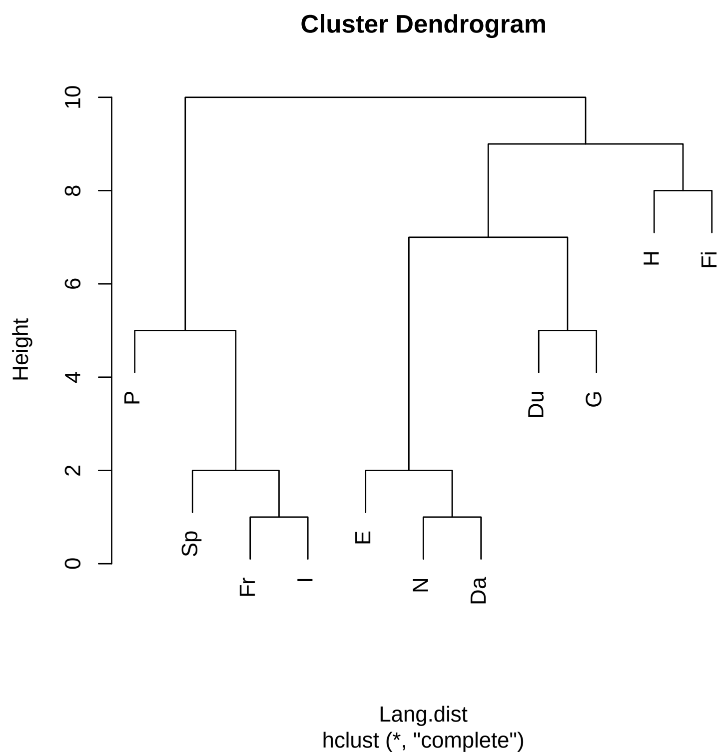


Figura 11.3: Gráficos de resíduos

```
> tab12.5 <- read.table("t12-5.dat")
> names(tab12.5)[9] <- "Empresas"
> empr.mat.cor <- cor(tab12.5[, -9])
> dd <- as.dist((1 - cor(USJudgeRatings))/2)
> empr.dist <- as.dist((1 - empr.mat.cor)/2)
> round(empr.dist, 3)
```

```
      V1    V2    V3    V4    V5    V6    V7
V2 0.179
V3 0.551 0.674
V4 0.541 0.543 0.450
V5 0.630 0.630 0.282 0.483
V6 0.576 0.505 0.486 0.644 0.412
V7 0.478 0.394 0.443 0.582 0.510 0.687
V8 0.507 0.664 0.497 0.257 0.504 0.780 0.593
```

```
> empr.clus <- hclust(empr.dist, method = "complete")
```

```
> plot(empr.clus)
```

11.8 Exemplo 12.9

Conglomeração de *average linkage* de 11 línguas.

```
> Lang.clus <- hclust(Lang.dist, method = "average")
> Lang.clus$merge
```

```
     [,1] [,2]
 [1,]   -2   -3
 [2,]   -6   -8
 [3,]   -7    2
 [4,]   -1    1
 [5,]   -9    3
 [6,]   -5    4
 [7,]   -4    6
```

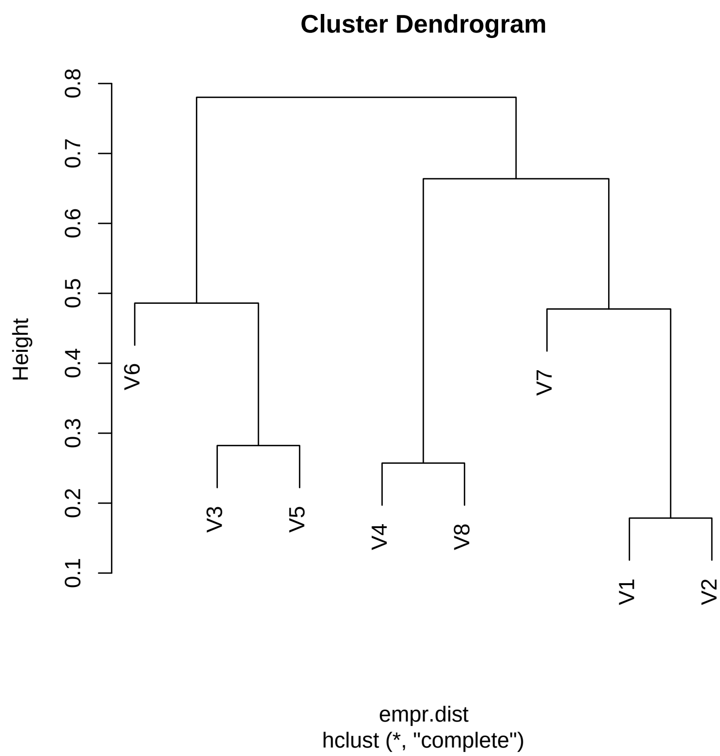


Figura 11.4: Gráficos de resíduos

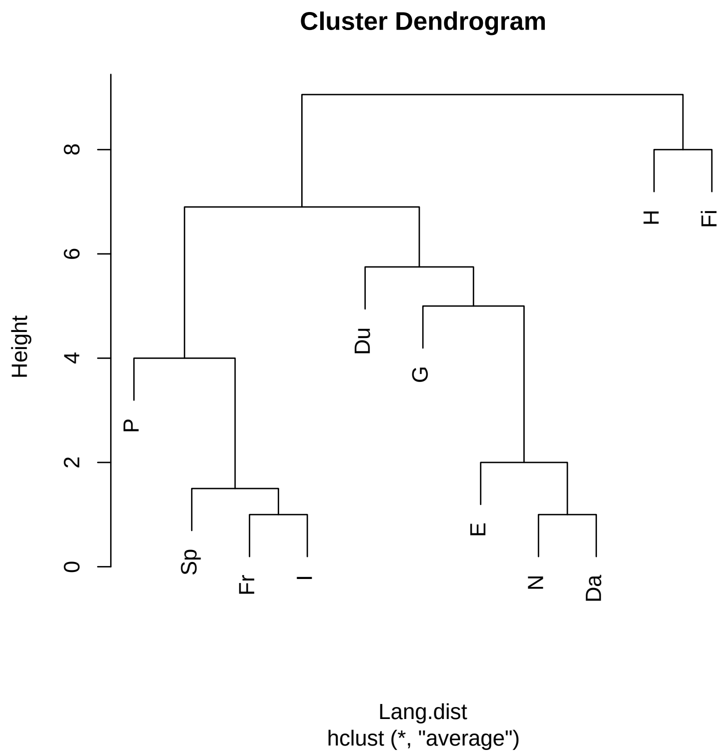


Figura 11.5: Gráficos de resíduos

```
 [8,]    5    7
 [9,]  -10  -11
[10,]    8    9

> plot(Lang.clus)
```

11.9 Exemplo 12.10

Conglomeração de *average linkage* de dados de empresas de utilidades públicas.

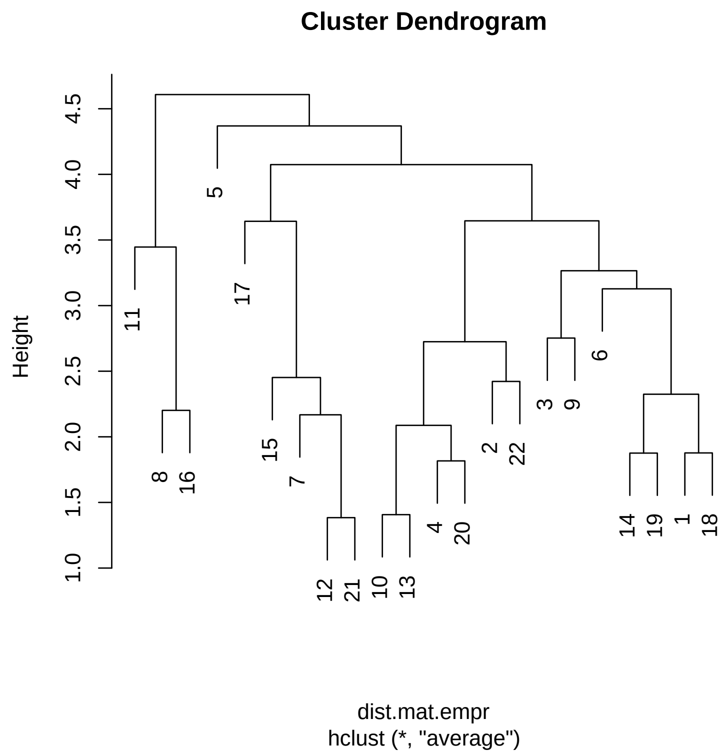


Figura 11.6: Gráficos de resíduos

```
> empr.clus <- hclust(dist.mat.empr, method = "average")
> plot(empr.clus)
```

11.10 Exemplo 12.11

Conglomeração de dados uísque de malte puro. Dados não disponíveis.

11.11 Exemplo 12.12

Conglomeração usando método de K-médias.

```
> Exe12.12 <- matrix(c(5, -1, 1, -3, 3, 1, -2, -2),
+     ncol = 2)
> dimnames(Exe12.12) <- list(c("A", "B", "C", "D"),
+     c("x1", "x2"))
> Exe12.12.clu <- kmeans(Exe12.12, centers = 2)
> Exe12.12.clu$cluster

A B C D
2 1 1 1

> Exe12.12.clu$centers

  x1 x2
1 -1 -1
2  5  3

> Exe12.12.clu$withinss

[1] 14  0

> Exe12.12.clu$size

[1] 3 1

> plot(Exe12.12, col = Exe12.12.clu$cluster)
> points(Exe12.12.clu$centers, col = 1:2, pch = 8,
+     cex = 2)
```

11.12 Exemplo 12.13

Conglomeração de K-médias de dados de utilidades públicas.

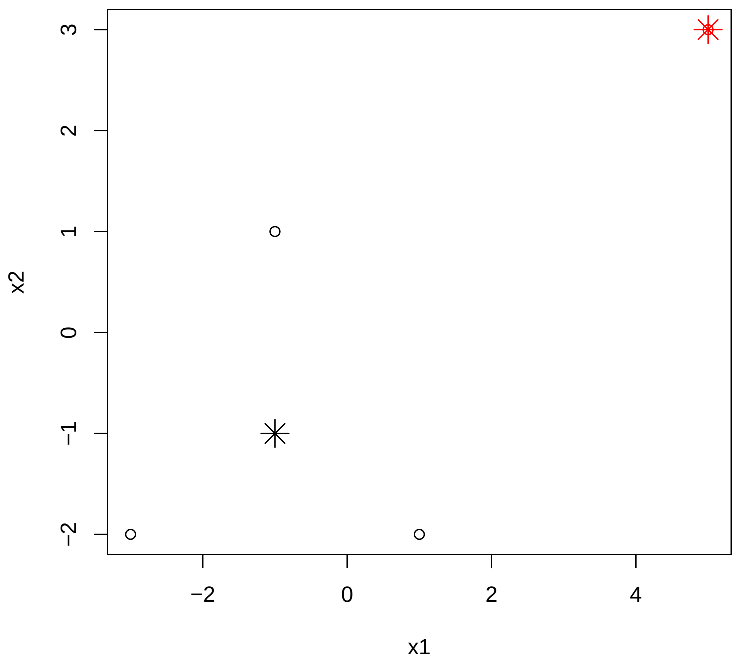


Figura 11.7: Gráficos de resíduos

```
> Exe12.13 <- as.matrix(scale(tab12.5[, -9]))
> dimnames(Exe12.13)[[1]] <- tab12.5[, 9]
> Exe12.13.clu4 <- kmeans(Exe12.13, 4)
> names(Exe12.13.clu4$cluster)[Exe12.13.clu4$cluster ==
+     1]

[1] "Boston"   "Consolid" "Hawaiian" "NewEngla" "Pacific"
[6] "SanDiego" "United"

> names(Exe12.13.clu4$cluster)[Exe12.13.clu4$cluster ==
+     2]

[1] "Arizona"  "Central"  "Florida"  "Kentucky" "Oklahoma"
[6] "Southern" "Texas"

> names(Exe12.13.clu4$cluster)[Exe12.13.clu4$cluster ==
+     3]

[1] "Common"   "Madison"  "Northern" "Wisconsi" "Virginia"

> names(Exe12.13.clu4$cluster)[Exe12.13.clu4$cluster ==
+     4]

[1] "Idaho"  "Nevada" "Puget"

> Exe12.13.clu5 <- kmeans(Exe12.13, 5)
> names(Exe12.13.clu5$cluster)[Exe12.13.clu5$cluster ==
+     1]

[1] "Idaho"  "Nevada" "Puget"

> names(Exe12.13.clu5$cluster)[Exe12.13.clu5$cluster ==
+     2]

[1] "Common"   "Madison"  "Northern" "Wisconsi" "Virginia"

> names(Exe12.13.clu5$cluster)[Exe12.13.clu5$cluster ==
+     3]
```

```
[1] "Consolid"

> names(Exe12.13.clu5$cluster)[Exe12.13.clu5$cluster ==
+     4]

[1] "Arizona"  "Central"  "Florida"  "Kentucky" "Oklahoma"
[6] "Southern" "Texas"

> names(Exe12.13.clu5$cluster)[Exe12.13.clu5$cluster ==
+     5]

[1] "Boston"   "Hawaiian" "NewEngla" "Pacific"  "SanDiego"
[6] "United"
```