Distanze

1. Si consideri la seguente matrice `X` di dimensioni 10×2. Calcolare la matrice di distanze tra le n unità statistiche con il comando `dist()` utilizzando la distanza Euclidea.

```
X <- matrix(c(2,3,3,4,4,5,6,6,7,8,7,8,10,6,8,10,12,13,11,12),nrow=10,ncol=2)
n <- nrow(X)
p <- ncol(X)

( D2 = dist(X, method="euclidean") )
```

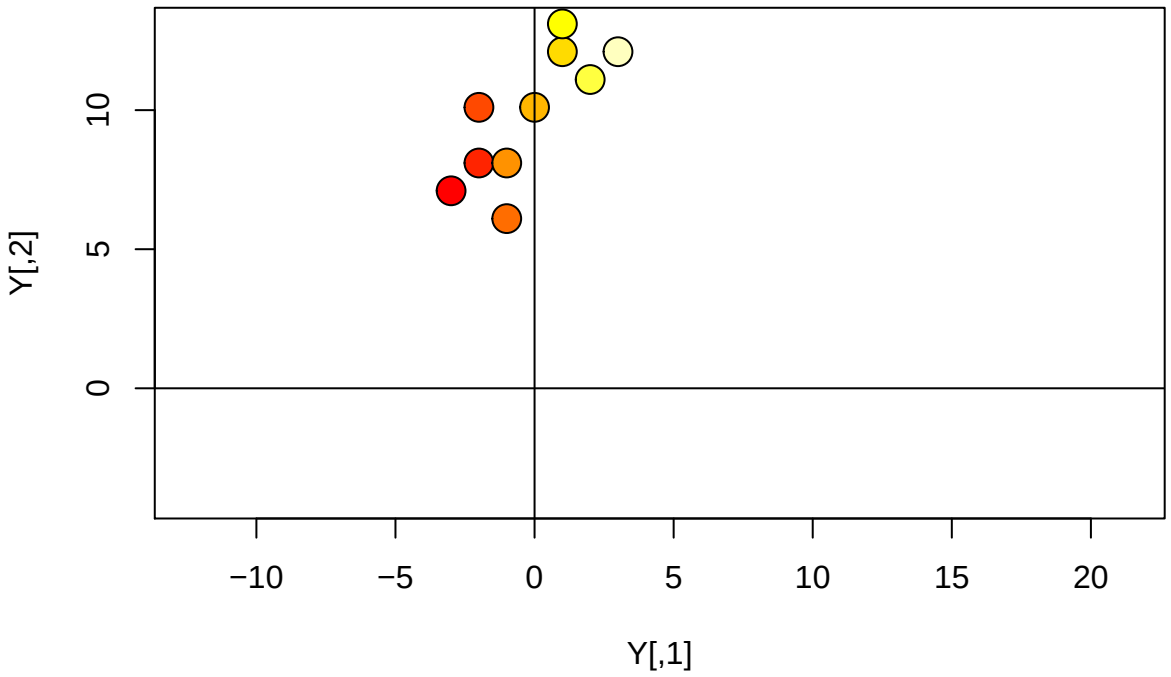
```
##           1        2        3        4        5        6        7        8
## 2  1.414214
## 3  3.162278 2.000000
## 4  2.236068 2.236068 4.123106
## 5  2.236068 1.000000 2.236068 2.000000
## 6  4.242641 2.828427 2.000000 4.123106 2.236068
## 7  6.403124 5.000000 3.605551 6.324555 4.472136 2.236068
## 8  7.211103 5.830952 4.242641 7.280110 5.385165 3.162278 1.000000
## 9  6.403124 5.000000 4.123106 5.830952 4.242641 2.236068 1.414214 2.236068
## 10 7.810250 6.403124 5.385165 7.211103 5.656854 3.605551 2.000000 2.236068
##           9
## 2
## 3
## 4
## 5
## 6
## 7
## 8
## 9
## 10 1.414214
```

2. Si consideri la traslazione $Y = X + 1b'$ con

$$b = \begin{bmatrix} -5 \\ 0.1 \end{bmatrix}$$

che sottrae 5 a tutti i valori della prima colonna e somma 0.1 a quelli della seconda. Si costruisca il diagramma di dispersione e si verifichi che la matrice di distanze di Minkowski di ordine $m \geq 1$ (ad esempio $m = \sqrt{2}$) non cambia (è invariante rispetto alle traslazioni).

```
b = matrix(c(-5,0.1),ncol=1)
one.n = matrix(rep(1,n),ncol=1)
Y = X + one.n%*%t(b)

plot(Y,xlim=c(-4,13),ylim=c(-4,13),asp=1,
     bg=heat.colors(n),pch=21,cex=2)
abline(h=0)
abline(v=0)
```




```
# verifico che è 0
m = sqrt(2)
sum( dist(Y, method="minkowski", p=m) - dist(X, method="minkowski", p=m) )
```

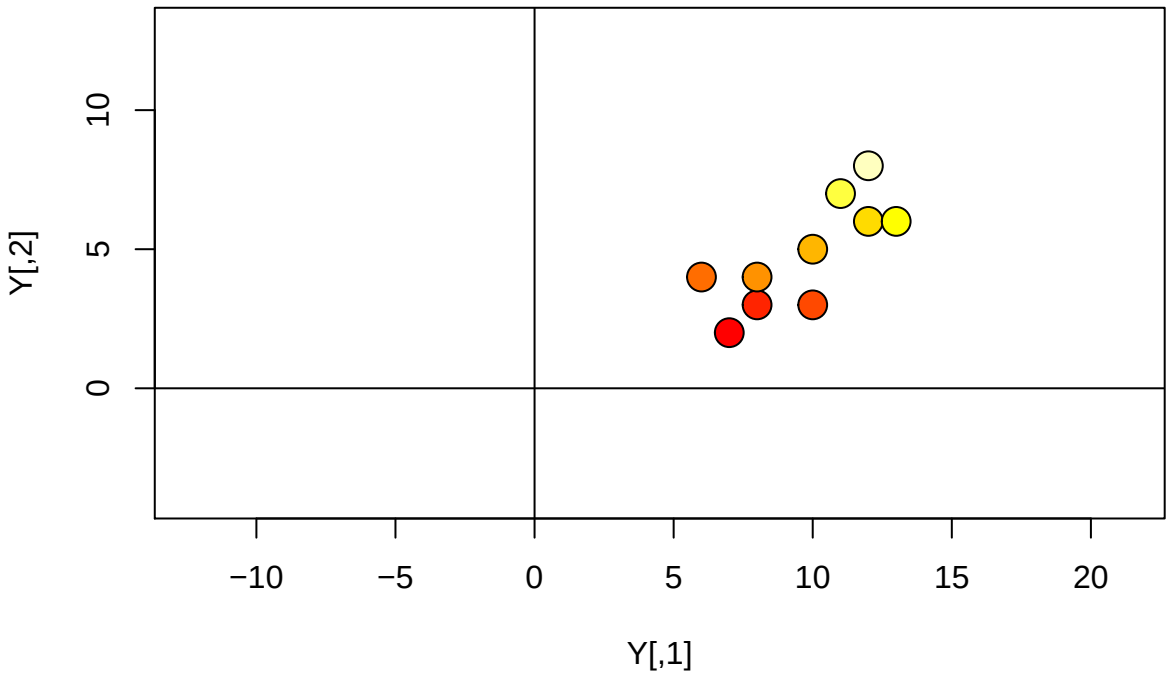
```
## [1] 0
```

3. Si consideri la trasformazione ortogonale $Y = XA'$ con

$$A = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$$

matrice di permutazione che scambia le due colonne di X. Si costruisca il diagramma di dispersione e si verifichi che la matrice di distanze Euclidee è invariante rispetto alle permutazioni.

```
A=matrix(c(0,1,1,0),2,2)
Y = X%*%t(A)

plot(Y,xlim=c(-4,13),ylim=c(-4,13),asp=1,
     bg=heat.colors(n),pch=21,cex=2)
abline(h=0)
abline(v=0)
```




```
# verifico che è 0
sum( dist(Y, method="euclidean") - D2 )
```

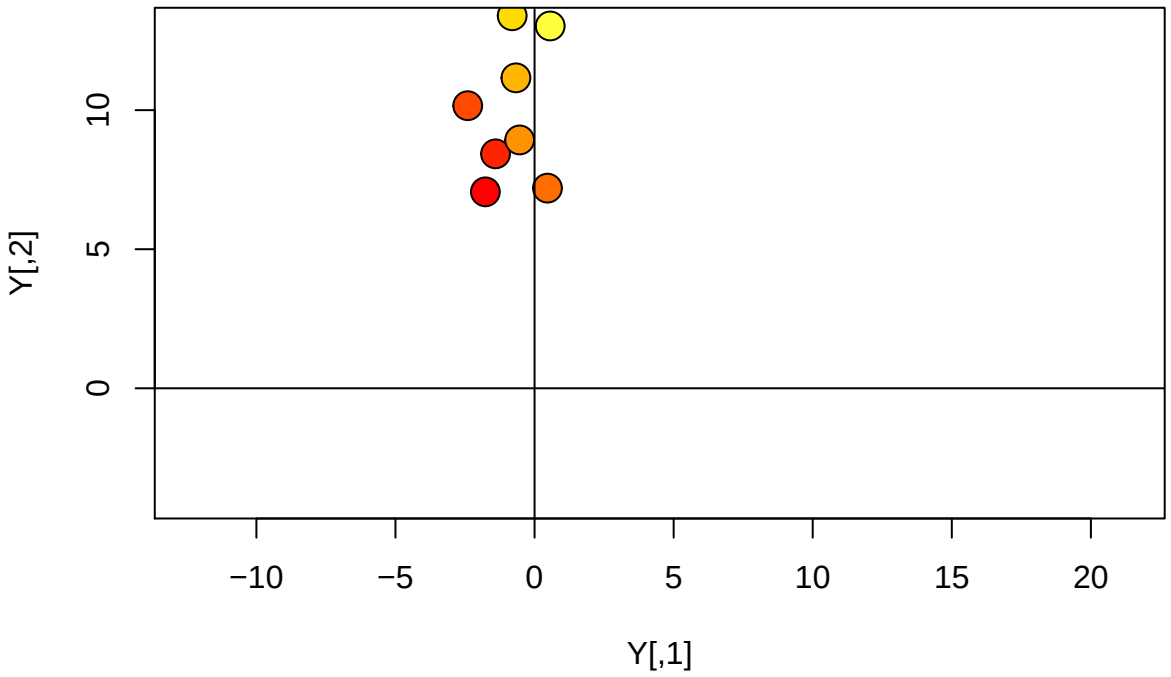
```
## [1] 0
```

4. Si consideri la trasformazione ortogonale $Y = XA'$ con

$$A = \begin{bmatrix} \cos(\pi/6) & -\sin(\pi/6) \\ \sin(\pi/6) & \cos(\pi/6) \end{bmatrix}$$

matrice di rotazione che comporta una rotazione antioraria di angolo $\theta = \pi/6$ radianti (30 gradi) intorno all'origine. Si costruisca il diagramma di dispersione e si verifichi che la matrice di distanze Euclidee è invariante rispetto alle rotazioni, mentre la matrice di distanze Manhattan no.

```
gradi = 30
theta = (pi/180)*gradi
A = matrix(c(cos(theta), -sin(theta), sin(theta), cos(theta)),byrow=T,2,2)

Y = X%*%t(A)

plot(Y,xlim=c(-4,13),ylim=c(-4,13),asp=1,
     bg=heat.colors(n),pch=21,cex=2)
abline(h=0)
abline(v=0)
```




```
# verifico che è 0
sum( dist(Y, method="euclidean") - D2 )
```

```
## [1] 1.265654e-14
```

```
# verifico che non è  0
sum( dist(Y, method="manhattan") - dist(X, method="manhattan")  )
```

```
## [1] -19.42889
```

Distanza di Mahalanobis e outliers

1. Importare i dati `Animals` presenti nella libreria `MASS`. Trasformare le misurazioni in scala logatitmica ed escludere le righe 6, 16 e 26 corrispondenti alle specie estinte Brachiosaurus, Triceratops e Dipliodocus. Costruire il diagramma di dispersione per le due variabili `log(body)` e `log(brain)` e aggiungere l'ellisse (comando `ellipse()`) con distanza costante $t = 2.447$ dal baricentro.

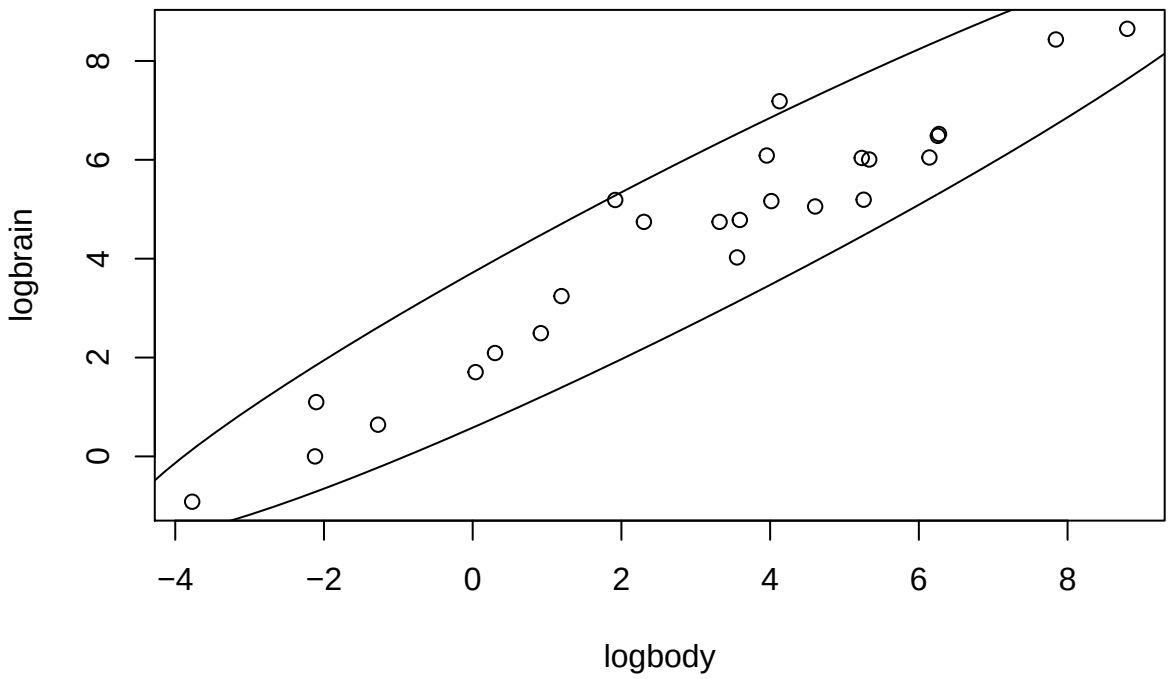
```
require("MASS")
```

```
## Loading required package: MASS
```

```
X = as.matrix( log(Animals[-c(6,16,26),]) )
colnames(X) = c("logbody","logbrain")
n = nrow(X)
p = ncol(X)

# vettore medie
xbar = matrix(colMeans(X), nrow=p, ncol=1)
# matrice di var/cov
S = var(X) * ((n-1)/n)

# ellisse
require(ellipse)
```

```
## Loading required package: ellipse
```

```
plot(X)
lines(ellipse(S,centre = t(xbar), t = 2.447))
```

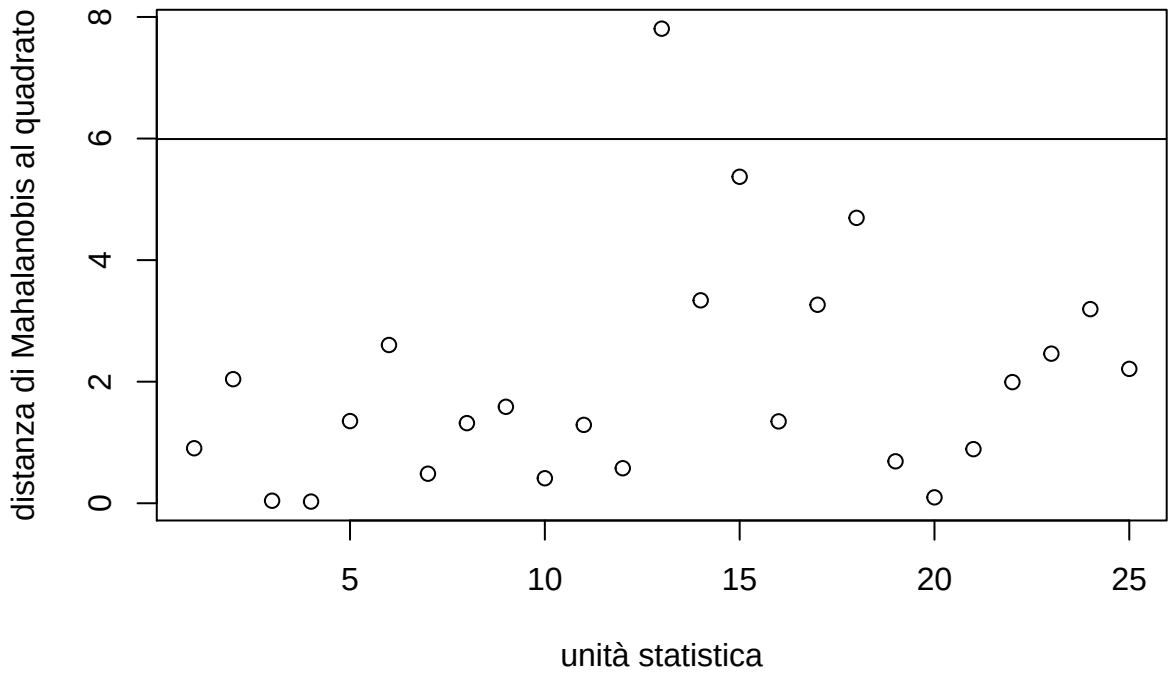


2. Assumendo che le n osservazioni misurate sulle due variabili `log(body)` e `log(brain)` siano realizzazioni i.i.d. da una Normale bivariata, verificare numericamente la presenza di valori anomali calcolando il quadrato della distanza di Mahalanobis dal baricentro $d^2_M(u_i, \bar{x}) = (u_i - \bar{x})' S^{-1} (u_i - \bar{x})$ e confrontandola con $q_{0.95} = 5.9915$ (il quantile 0.95 di un χ^2_2), e commentare.

```
# matrice inversa
InvS = solve(S)

# distanza di Mahalanobis al quadrato per la prima osservazione
t(X[1,] - xbar) %*% InvS %*% (X[1,] - xbar)
```

```
##           [,1]
## [1,] 0.9049929
```

```
# quadrato della distanza di Mahalanobis per le n osservazioni
dM2 = apply(X,MARGIN=1, function(u) t(u-xbar) %*% InvS %*% (u - xbar) )

# quantile 0.95 di una distribuzione Chi-quadrato con p=2 gradi di libertà
q.95 = qchisq(0.95, df=2)

# grafico
plot(dM2, xlab="unità statistica", ylab="distanza di Mahalanobis al quadrato")
abline(h=q.95)
```




```
# individuo la riga corrispondente all'outlier
which(dM2 > q.95)
```

```
## Human
##    13
```

```
# valore atteso di outliers
n * 0.05
```

```
## [1] 1.25
```

Commento.

3. Verificare che il quadrato della distanza di Mahalanobis dal baricentro calcolata sui dati originali è uguale al quadrato della distanza Euclidea dall'origine calcolata sui dati ortogonalizzati.

```
# dati centrati
Xtilde <- scale(X,center=TRUE,scale=FALSE)

# S^(1/2)
eigenS = eigen(S)
InvSqrtS = eigenS$vectors %*% diag(eigenS$values^(-1/2)) %*% t(eigenS$vectors)

# dati ortogonalizzati
Ztilde = Xtilde %*% InvSqrtS

# quadrato della distanza Euclidea dall'origine
dE2.Ztilde = apply(Ztilde,MARGIN=1, function(u) t(u) %*% u )

# verifico che la somma delle differenze è 0
sum(dE2.Ztilde  - dM2)
```

```
## [1] 5.4904e-14
```

4. Verificare che il quadrato della distanza di Mahalanobis dal baricentro non cambia (è invariante) se trasformo il peso del corpo da Kg a grammi (trasformazione di scala) prima di effettuare la trasformazione

al logaritmo.

```
Y = X
Y[,"logbody"] = X[,"logbody"] + log(1000)
n = nrow(Y)
p = ncol(Y)

# vettore medie
ybar = matrix(colMeans(Y), nrow=p, ncol=1)
# matrice di var/cov
S.Y = var(Y) * ((n-1)/n)

# matrice inversa
InvS.Y = solve(S.Y)

# distanza di Mahalanobis al quadrato per tutte le osservazioni
dM2.Y = apply(Y,MARGIN=1, function(u) t(u-ybar) %*% InvS.Y %*% (u - ybar) )

# verifico che la somma delle differenze è 0
sum(dM2.Y - dM2)
```

```
## [1] -1.075529e-14
```