
```
Exercice d'inversion, matrices non singulières et singulières

==========

Inversion d'une matrice non singulière par 'solve'

mat2=matrix(c(1,2,3,4,5,6,7,8,10),3,3,byrow=TRUE)
mat2
     [,1] [,2] [,3]
[1,]    1    2    3
[2,]    4    5    6
[3,]    7    8   10

# La matrice est-elle singulière?
det(mat2)
[1] -3

# Calcul de l'inverse (fonction 'solve')
mat2.solve=solve(mat2)
mat2.solve
           [,1]      [,2] [,3]
[1,] -0.6666667 -1.333333    1
[2,] -0.6666667  3.666667   -2
[3,]  1.0000000 -2.000000    1

==========

Inversion d'une matrice singulière
par décomposition en valeurs singulières (fonction 'svd')

# Création d'une matrice singulière
mat1=matrix(c(1,2,3,4,5,1,2,3,4,5,5,2,4,3,1),5,3)
mat1
     [,1] [,2] [,3]
[1,]    1    1    5
[2,]    2    2    2
[3,]    3    3    4
[4,]    4    4    3
[5,]    5    5    1

# Calcul de la matrice de covariance
mat1.cov=cov(mat1)
mat1.cov
      [,1]  [,2]  [,3]
[1,]  2.50  2.50 -1.75
[2,]  2.50  2.50 -1.75
[3,] -1.75 -1.75  2.50

# La matrice est-elle singulière?
det(mat1.cov)
[1] 0

# Calcul de l'inverse (fonction 'solve')
mat1.solve=solve(mat1.cov)
Erreur dans solve.default(mat1.cov) : sous-programme Lapack dgesv :
le système est exactement singulier
```

```
# Décomposition en valeurs singulières
mat1.svd=svd(mat1.cov)
mat1.svd
$d
[1] 6.522634e+00 9.773659e-01 5.223681e-16

$u
           [,1]       [,2]          [,3]
[1,] -0.6022530 -0.3705284  7.071068e-01
[2,] -0.6022530 -0.3705284 -7.071068e-01
[3,]  0.5240063 -0.8517144  1.821078e-16

$v
           [,1]       [,2]          [,3]
[1,] -0.6022530 -0.3705284 -7.071068e-01
[2,] -0.6022530 -0.3705284  7.071068e-01
[3,]  0.5240063 -0.8517144  1.307369e-18

# Deux premières colonnes de l'élément $u
mat1.svd$u[,1:2]
# Deux premières valeurs propres dans une matrice diagonale
diag(mat1.svd$d[1:2])

# Inversion (equation 2.32)
mat1.inv=mat1.svd$u[,1:2] %*% diag(mat1.svd$d[1:2]^-1) %*% t(mat1.svd$u[,1:2])
mat1.inv
          [,1]      [,2]      [,3]
[1,] 0.1960784 0.1960784 0.2745098
[2,] 0.1960784 0.1960784 0.2745098
[3,] 0.2745098 0.2745098 0.7843137

# Vérification
mat1.cov %*% mat1.inv

==========

Vérification: inversion par 'ginv' (bibliothèque 'MASS')

library(MASS)
mat1.ginv=ginv(mat1.cov)
mat1.ginv
          [,1]      [,2]      [,3]
[1,] 0.1960784 0.1960784 0.2745098
[2,] 0.1960784 0.1960784 0.2745098
[3,] 0.2745098 0.2745098 0.7843137

==========
```

```
# Utilisation de l'inversion en régression multiple

# Les deux premières variables de X sont colinéaires
X = matrix(c(1,2,3,4,5,1,2,3,4,5,5,2,4,3,1),5,3)
X
X.cen = apply(X,2,scale,center=TRUE,scale=FALSE)
y = matrix(rnorm(5,0,1),5,1)
y

# Utilisation de 'ginv' sur les 3 variables de X
XprX = t(X.cen) %*% X.cen
det(XprX)
[1] 0          # Le déterminant est 0: matrice singulière

# Inversion par ginv(XprX)  (bibliothèque 'MASS')

b = ginv(XprX) %*% t(X.cen) %*% y
b
            [,1]
[1,] -0.06398446
[2,] -0.06398446
[3,] -0.15571786

proj = X.cen %*% ginv(XprX) %*% t(X.cen)
y.fit = proj %*% y
y.fit
            [,1]
[1,] -0.50654078
[2,]  0.33912492
[3,] -0.29619765
[4,] -0.04292727
[5,]  0.50654078

# Utilisation de 'solve' sur les variables 2 et 3 de X
X.2 = X.cen[,2:3]
XprX.2 = t(X.2) %*% X.2
det(XprX.2)
[1] 51          # Le déterminant est > 0: matrice non singulière

# Inversion par solve(XprX.2)

b2 = solve(XprX.2) %*% t(X.2) %*% y
b2
           [,1]
[1,] -0.1279689
[2,] -0.1557179

proj.2 = X.2 %*% solve(XprX.2) %*% t(X.2)
y.fit.2 = proj.2 %*% y
y.fit.2
            [,1]
[1,] -0.50654078
[2,]  0.33912492
[3,] -0.29619765
[4,] -0.04292727
[5,]  0.50654078
```