ejercicios_ecuaciones_lineales

Oscar Omar

12/5/2021

```
A <- rbind(c(1,1,2),c(2,4,-3),c(3,6,-5))
b <- c(9,1,0)
AB <- cbind(A,b)
AB
```

```
##              b
## [1,] 1 1  2 9
## [2,] 2 4 -3 1
## [3,] 3 6 -5 0
```

```
qr(A)$rank==qr(AB)$rank
```

```
## [1] TRUE
```

```
solve(A,b)
```

```
## [1] 1 2 3
```

Ejemplo 2 en R

```
A <- rbind(c(2,2),c(-1,1))
b <- c(1,2)
AB <- cbind(A,b)
R(A)
```

```
## [1] 2
```

```
R(AB)
```

```
## [1] 2
```

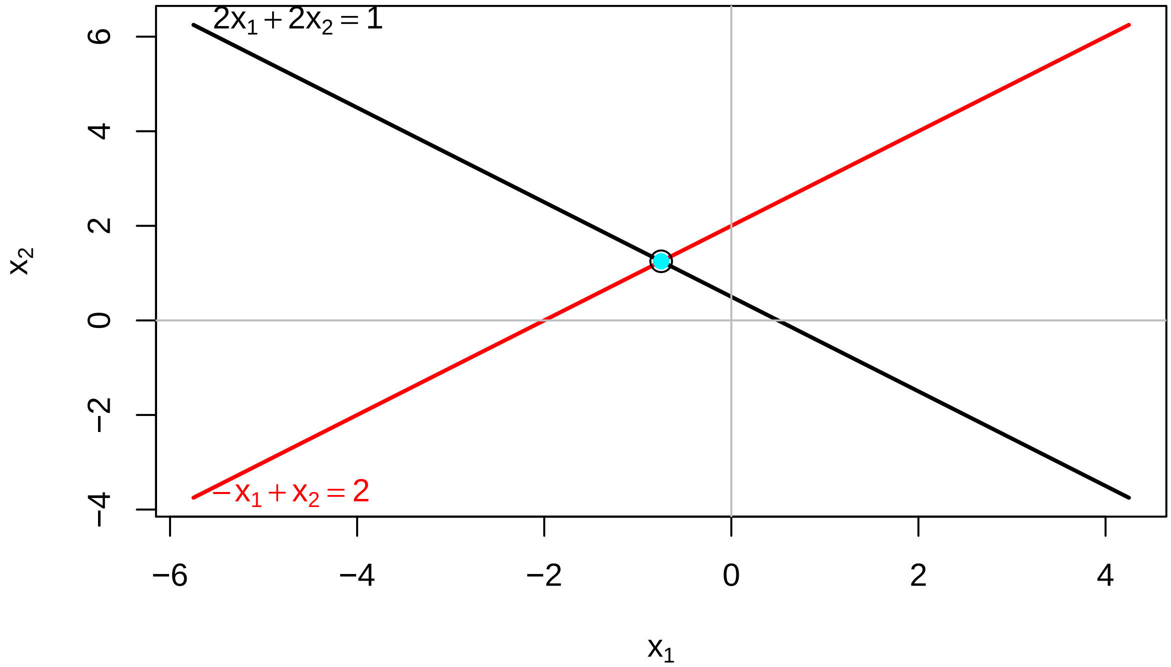
```
Solve(A, b, fractions = TRUE)
```

```
## x1    =  -3/4
##   x2  =   5/4
```

```
plotEqn(A,b)
```

```
## 2*x[1] + 2*x[2]  =  1
##  -x[1]   + x[2]  =  2
```

```
points(-3/4,5/4, col="turquoise1", pch=19)
```



## Ejemplo 3 en R

```
A <- rbind(c(4,2),c(1,-2),c(3,4))
b <- c(3,2,1)
AB <- cbind(A,b)
showEqn(A,b)
```

```
## 4*x1 + 2*x2  =  3
## 1*x1 - 2*x2  =  2
## 3*x1 + 4*x2  =  1
```

```
R(A)
```

```
## [1] 2
```

```
R(AB)
```

```
## [1] 2
```

```
all.equal(R(A),R(AB))
```

```
## [1] TRUE
```

```
Solve(A,b, fractions = T)
```

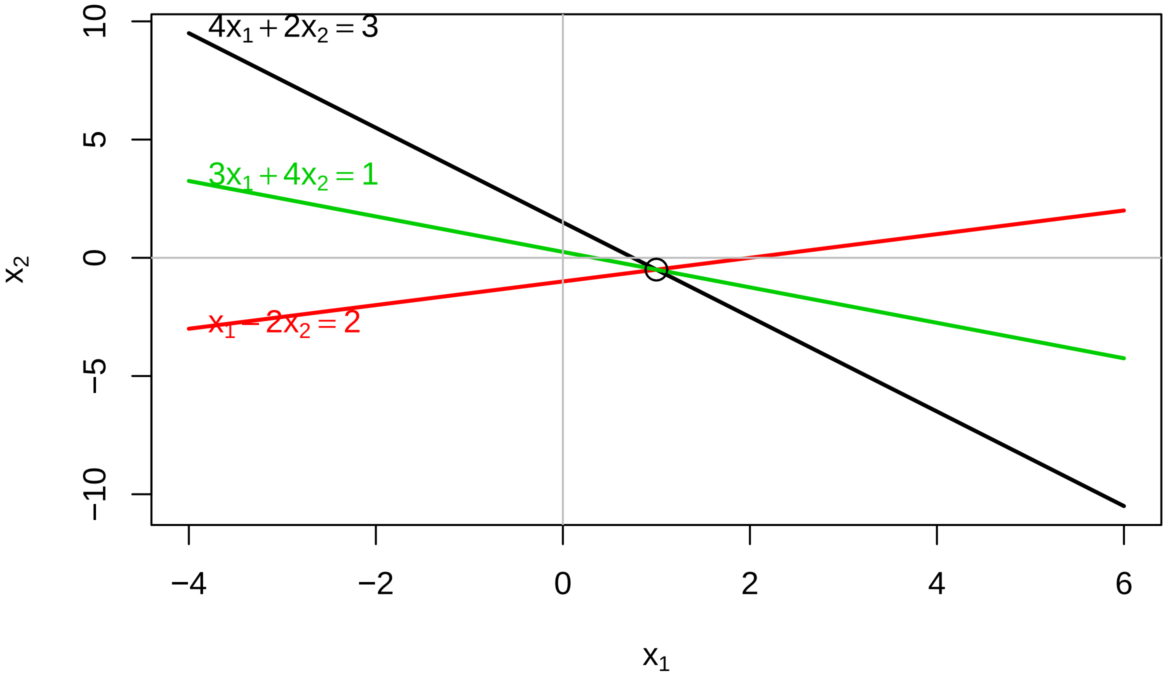
```
## x1    =     1
##   x2  =  -1/2
##    0  =     0
```

```
s = c(1, -1/2)
A%*%s == b
```

```
##      [,1]
## [1,] TRUE
## [2,] TRUE
## [3,] TRUE
```

```
plotEqn(A,b)
```

```
## 4*x[1] + 2*x[2]  =  3
##   x[1] - 2*x[2]  =  2
## 3*x[1] + 4*x[2]  =  1
```



```
A <- rbind(c(1,1,2),c(2,4,-3),c(3,6,-5))
b <- c(9,1,0)
AB <- cbind(A,b)
AB
```

```
##              b
## [1,] 1 1  2 9
## [2,] 2 4 -3 1
## [3,] 3 6 -5 0
```

```
qr(A)$rank==qr(AB)$rank
```

```
## [1] TRUE
```

```
solve(A,b)
```

```
## [1] 1 2 3
```

```r
echelon(AB, verbose = T, fractions = T)
```

```
##
## Initial matrix:
##               b
## [1,]  1  1  2  9
## [2,]  2  4 -3  1
## [3,]  3  6 -5  0
##
## row: 1
##
##  exchange rows 1 and 3
##               b
## [1,]  3  6 -5  0
## [2,]  2  4 -3  1
## [3,]  1  1  2  9
##
##  multiply row 1 by 1/3
##                     b
## [1,]    1    2 -5/3    0
## [2,]    2    4   -3    1
## [3,]    1    1    2    9
##
##  multiply row 1 by 2 and subtract from row 2
##                     b
## [1,]    1    2 -5/3    0
## [2,]    0    0  1/3    1
## [3,]    1    1    2    9
##
##  subtract row 1 from row 3
##                     b
## [1,]    1    2 -5/3    0
## [2,]    0    0  1/3    1
## [3,]    0   -1 11/3    9
##
## row: 2
##
##  exchange rows 2 and 3
##                     b
## [1,]    1    2 -5/3    0
## [2,]    0   -1 11/3    9
## [3,]    0    0  1/3    1
##
##  multiply row 2 by -1
##                        b
## [1,]     1     2  -5/3     0
## [2,]     0     1 -11/3    -9
## [3,]     0     0   1/3     1
##
##  multiply row 2 by 2 and subtract from row 1
##                        b
## [1,]     1     0  17/3    18
## [2,]     0     1 -11/3    -9
## [3,]     0     0   1/3     1
```

```
##
## row: 3
##
##  multiply row 3 by 3
##                         b
## [1,]     1     0  17/3    18
## [2,]     0     1 -11/3    -9
## [3,]     0     0     1     3
##
##  multiply row 3 by 17/3 and subtract from row 1
##                         b
## [1,]     1     0     0     1
## [2,]     0     1 -11/3    -9
## [3,]     0     0     1     3
##
##  multiply row 3 by 11/3 and add to row 2
##            b
## [1,] 1 0 0 1
## [2,] 0 1 0 2
## [3,] 0 0 1 3
```

```
a <- matrix(c(1,1,-1,1,-1,1,3,1,-1), byrow = TRUE, nrow = 3, ncol = 3)
b <- c(2,1,5)
ab = cbind(a,b)
c(R(a),R(ab))
```

```
## [1] 2 2
```

```
echelon(ab)
```

```
##               b
## [1,] 1 0  0 1.5
## [2,] 0 1 -1 0.5
## [3,] 0 0  0 0.0
```

```
Solve(a,b, fractions = T)
```

```
## x1           =  3/2
##   x2 - 1*x3  =  1/2
##           0  =    0
```

```
A <- cbind(c(1,1,2),c(1,-1,1))
b <- c(2,1,3)
AB <- cbind(A,b)
AB
```

```
##           b
## [1,] 1  1 2
## [2,] 1 -1 1
## [3,] 2  1 3
```

```
c(R(A), R(AB))
```

```
## [1] 2 3
```

```
Solve(A,b, fractions = T)
```

```
## x1    =  4/3
##   x2  =  1/3
##    0  =  1/3
```

```
all.equal(R(A), R(AB))
```

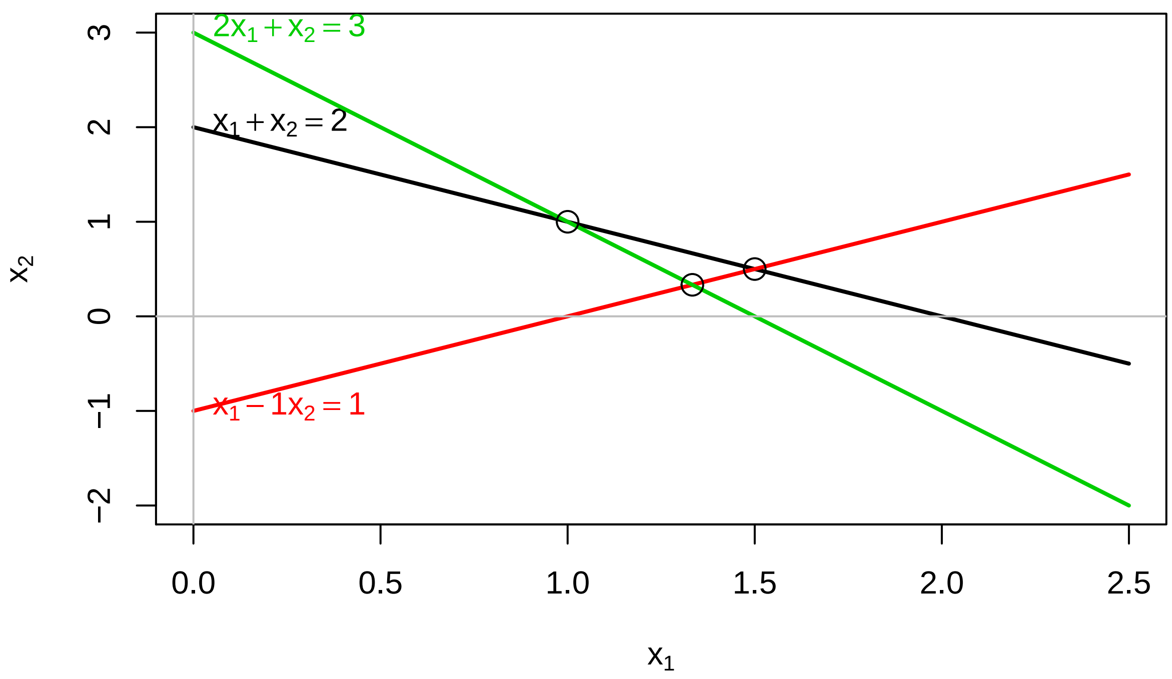
```
## [1] "Mean relative difference: 0.5"
```

```
echelon(AB)
```

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

```
plotEqn(A,b)
```

```
##   x[1]   + x[2]  =  2
##   x[1] - 1*x[2]  =  1
## 2*x[1]   + x[2]  =  3
```



## Ejemplo 6 R

```
A <- rbind(c(0,4),c(2,1))
B <- rbind(c(1,-1),c(2,3))
C <- rbind(c(1,2),c(3,-2))
D <- rbind(c(-2,1),c(-1,1))
I <- diag(1, nrow = 2, ncol = 2)
M <- A - (C + D)
N <- 3*(D-B)+10*I
X <- solve(M,N)
X
```

```
##      [,1] [,2]
## [1,]  5.5    4
## [2,] -4.5    2
```

```
A%*%X + 3*B == (C+D)%*%X+3*D+10*I
```

```
##      [,1] [,2]
## [1,] TRUE TRUE
## [2,] TRUE TRUE
```

Ejemplo 1 Python

```
import numpy as np
A = np.array([[1,1,2],[2,4,-3],[3,6,-5]])
b = np.array([9,1,0])
AB = np.array([[1,1,2,9],[2,4,-3,1],[3,6,-5,0]])
np.linalg.matrix_rank(A) == np.linalg.matrix_rank(AB)
```

```
## True
```

```
np.linalg.matrix_rank(A)
```

```
## 3
```

```
np.linalg.solve(A, b)
```

```
## array([1., 2., 3.])
```

Ejemplo 2 Python

```
from sympy import *
from sympy.solvers.solveset import linsolve
x,y,z = symbols('x,y,z')
x1,x2,x3 = symbols('x1,x2,x3')
linsolve([2*x1 + 2*x2 - 1, -x1 + x2 - 2], (x1,x2))
```

```
## FiniteSet((-3/4, 5/4))
```

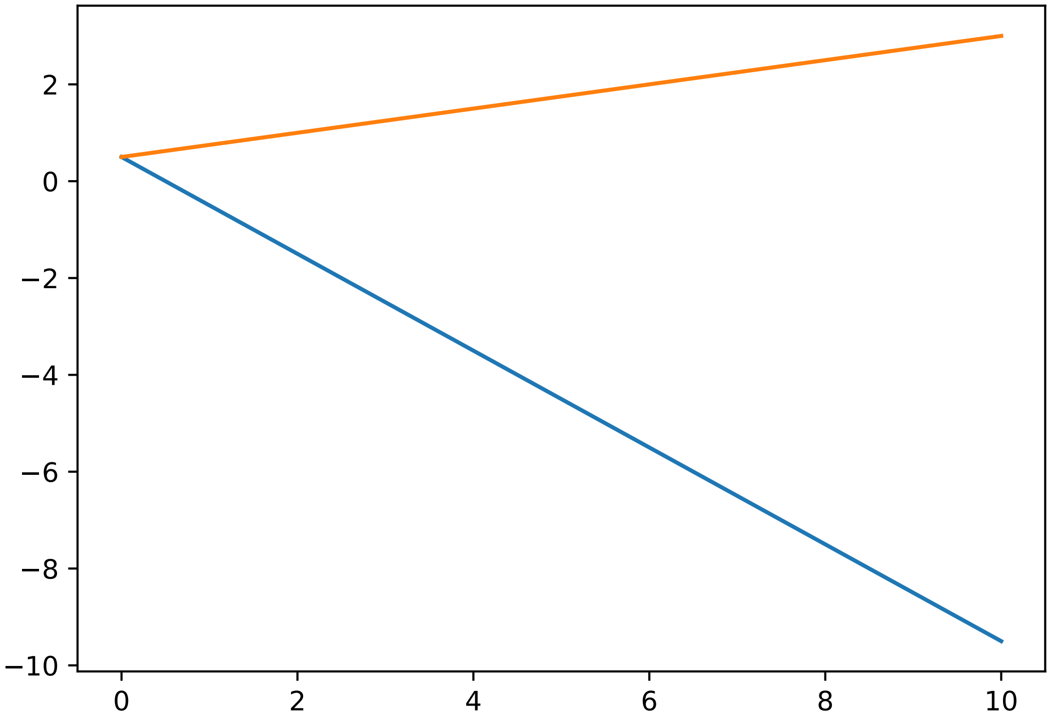
```
linsolve(Matrix(([2,2,1],[-1,1,2])), (x1,x2))
```

```
## FiniteSet((-3/4, 5/4))
```

```
import matplotlib.pyplot as plt
x1 = np.linspace(0,10,100)
plt.plot(x1, 1/2-x1, x1, (2+x1)/4)
```

```
## [<matplotlib.lines.Line2D object at 0x000000003743BBA8>, <matplotlib.lines.Line2D object at 0x0000000
```

```
plt.show()
```

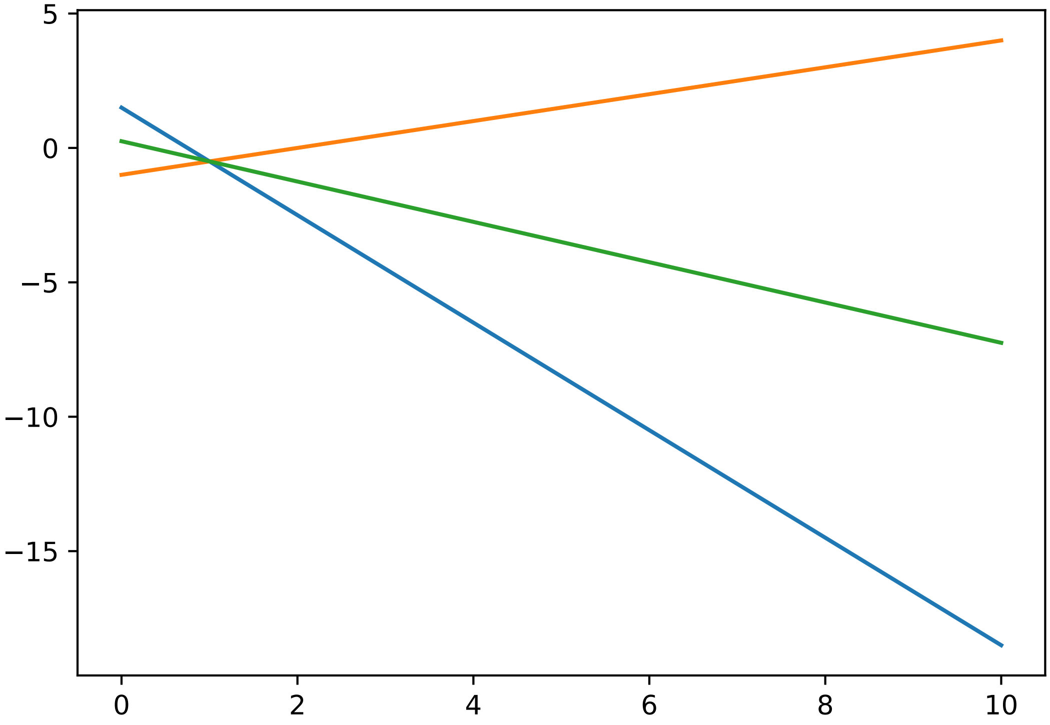


Ejemplo 3 `Python`

```
import matplotlib.pyplot as plt
x1 = np.linspace(0,10,100)
plt.plot(x1, 3/2-2*x1, x1, (x1/2)-1, x1, (1-3*x1)/4)
```

```
## [<matplotlib.lines.Line2D object at 0x00000000373FE7B8>, <matplotlib.lines.Line2D object at 0x0000000
```

```
plt.show()
```



Ejemplo 4 `Python`

```
from mpl_toolkits.mplot3d import Axes3D

point1 = np.array([0,0,9/2])
n1 = np.array([1,1,2])

point2 = np.array([0,0,-1/3])
n2 = np.array([2,4,-3])

point3 = np.array([0,0,0])
n3 = np.array([3,6,-5])

D1 = -9
D2 = -1
D3 = 0

X, Y = np.meshgrid(range(30), range(30))

Z1 = (-n1[0]*X - n1[1]*Y - D1)*1./n1[2]
Z2 = (-n2[0]*X - n2[1]*Y - D2)*1./n2[2]
Z3 = (-n3[0]*X - n3[1]*Y - D3)*1./n3[2]

plot3d = plt.figure().gca(projection='3d')
plot3d.plot_surface(X,Y,Z1, color='red')
```

```
## <mpl_toolkits.mplot3d.art3d.Poly3DCollection object at 0x000000002948C1D0>
```

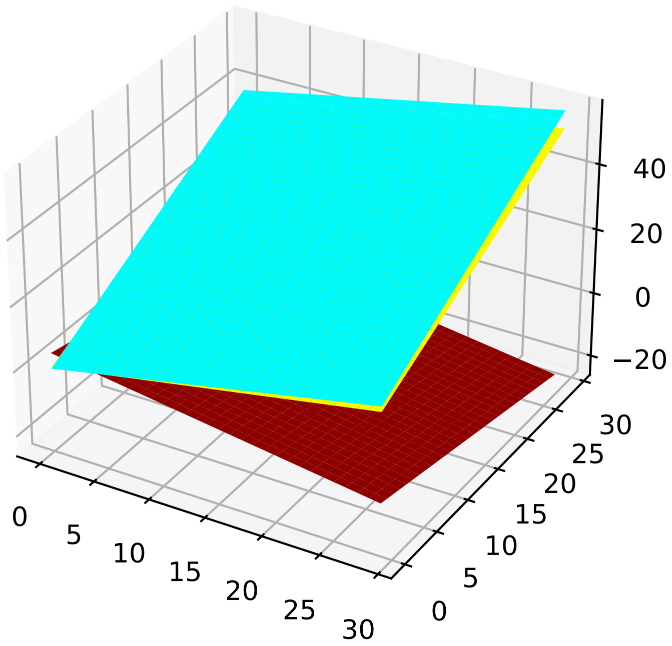
```
plot3d.plot_surface(X,Y,Z2, color='cyan')
```

```
## <mpl_toolkits.mplot3d.art3d.Poly3DCollection object at 0x000000002948C4A8>
```

```
plot3d.plot_surface(X,Y,Z3, color='yellow')
```

```
## <mpl_toolkits.mplot3d.art3d.Poly3DCollection object at 0x000000002948C780>
```

```
plt.show()
```



Ejemplo 5 Python

```
A = np.array([[1,1,-1],[1,-1,1],[3,1,-1]])
B = np.array([2,1,5])
AB = np.array([[1,1,-1,2],[1,-1,1,1],[3,1,-1,5]])
```