# ALGEBRA LINEAL

Muchos cálculos implican la resolución de sistemas de ecuaciones lineales. En muchos casos, te resultará conveniente escribir las ecuaciones explícitamente y luego resolverlas con **Solve**.

En algunos casos, sin embargo, puede que prefiera convertir el sistema de ecuaciones lineales en una ecuación matricial y luego aplicar operaciones de manipulación de matrices para resolverlo. Este enfoque suele ser útil cuando el sistema de ecuaciones surge como parte de un algoritmo general y no se sabe de antemano cuántas variables estarán involucradas.

Un sistema de ecuaciones lineales se puede expresar en forma matricial como **m.x == b**, donde x es el vector de variables.

Tenga en cuenta que si su sistema de ecuaciones es disperso, de modo que la mayoría de las entradas en la matriz m son cero, lo mejor es representar la matriz como un objeto **SparseArray.** Como se explica en “Matrices Dispersas: Álgebra Lineal”, puede convertir ecuaciones simbólicas a objetos **SparseArray** mediante **CoefficientArrays**. Todas las funciones descritas aquí funcionan tanto con objetos **SparseArray** como con **matrices ordinarias**.

## Resolviendo sistemas lineales

***LinearSolve [m, b]***
Un vector x que resuelve la ecuación matricial: **m.x == b**.

***NullSpace [m]***
Una lista de vectores linealmente independientes cuyas combinaciones lineales abarcan todas las soluciones de la ecuación matricial m.x == 0

***MatrixRank [m]***
Da el número de filas o columnas linealmente independientes de m.

***RowReduce [m]***
Da una forma simplificada de **m** obtenida al realizar combinaciones lineales de filas.

### Ejemplos:

In[ ]:=
```
Clear["Global`*"]
```
borra

Definimos una matriz de 2 por 2:

In[ ]:=
```
m = {{1, 5}, {2, 1}};
```

In[ ]:=
```
MatrixForm[m]
```
forma de matriz

Out[ ]//MatrixForm=

$$\begin{pmatrix} 1 & 5 \\ 2 & 1 \end{pmatrix}$$

Al aplicar la ecuación matricial **m.x == b** nos da dos ecuaciones lineales:

In[•]:=
```
m.{x, y} == {a, b}
```

Out[•]=

$$\{x + 5\,y,\ 2\,x + y\} == \{a,\ b\}$$

Podemos resolver las ecuaciones directamente usando el comando **Solve**:

In[•]:=
```
Solve[%, {x, y}]
```
resuelve

Out[•]=

$$\left\{\left\{x \to \frac{1}{9}\,(-a + 5\,b),\ y \to \frac{1}{9}\,(2\,a - b)\right\}\right\}$$

El comando **Solve** aplica al sistema de ecuaciones.

También puedes obtener el vector de soluciones mediante **LinearSolve**. El resultado es equivalente al que se obtiene con **Solve**:

In[•]:=
```
LinearSolve[m, {a, b}]
```
resuelve ecuación lineal

Out[•]=

$$\left\{\frac{1}{9}\,(-a + 5\,b),\ \frac{1}{9}\,(2\,a - b)\right\}$$

El comando **LinearSolve** aplica a la matriz **m** para resolver la ecuación matricial.

Otra forma de resolver las ecuaciones es invertir la matriz **m** y luego multiplicar **{a,b}** por la inversa. Esto no es tan eficiente como usar **LinearSolve**:

In[•]:=
```
Inverse[m].{a, b}
```
matriz inversa

Out[•]=

$$\left\{-\frac{a}{9} + \frac{5\,b}{9},\ \frac{2\,a}{9} - \frac{b}{9}\right\}$$

**RowReduce** realiza una versión de eliminación gaussiana y también se puede utilizar para resolver las ecuaciones:

In[•]:=
```
RowReduce[{{1, 5, a}, {2, 1, b}}]
```
reduce filas

Out[•]=

$$\left\{\left\{1,\ 0,\ \frac{1}{9}\,(-a + 5\,b)\right\},\ \left\{0,\ 1,\ \frac{1}{9}\,(2\,a - b)\right\}\right\}$$

Si se tiene una matriz cuadrada **m** con determinante distinto de cero, siempre se puede encontrar una solución única para la ecuación matricial **m.x == b** para cualquier b. Sin embargo, si la matriz **m** tiene determinante cero, entonces puede no haber ningún vector, o un número infinito de vectores x que satisfagan **m.x == b** para un b particular. Esto ocurre cuando las ecuaciones lineales incorporadas en m no son independientes.

Cuando **m** tiene determinante cero, siempre es posible encontrar vectores x distintos de cero que satisfagan **m.x = 0**. El conjunto de vectores x que satisfacen esta ecuación forma el espacio nulo o núcleo de la matriz **m**. Cualquiera de estos vectores puede expresarse como una combinación lineal de un conjunto particular de vectores base, que se obtiene utilizando **EspacioNulo[m]**.

## Ejemplos:

```
In[•]:= Clear["Global`*"]
```
borra

Aquí hay una matriz simple, correspondiente a dos ecuaciones lineales idénticas:

```
In[•]:= m = {{1, 2}, {1, 2}}
Out[•]=
{{1, 2}, {1, 2}}
```

```
In[•]:= MatrixForm[%]
```
forma de matriz

```
Out[•]//MatrixForm=
(1 2)
(1 2)
```

**LinearSolve** no puede encontrar una solución a la ecuación **m.x == b** en este caso:

```
In[•]:= LinearSolve[m, {a, b}]
```
resuelve ecuación lineal

LinearSolve : Linear equation encountered that has no solution.

```
Out[•]=
LinearSolve[{{1, 2}, {1, 2}}, {a, b}]
```

Hay un único vector base para el espacio nulo de m:

```
In[•]:= NullSpace[m]
```
espacio nulo

```
Out[•]=
{{-2, 1}}
```

Al multiplicar el vector base del espacio nulo por **m** se obtiene el vector cero:

```
In[•]:= m.%[[1]]
Out[•]=
{0, 0}
```

Solo hay una fila linealmente independiente en m:

```
In[•]:= MatrixRank[m]
```
rango matricial

```
Out[•]=
1
```

**NullSpace** y **MatrixRank** deben determinar si determinadas combinaciones de elementos de la matriz son cero. Para matrices numéricas aproximadas, se puede usar la opción **Tolerance** para especificar qué tan cerca de cero se considera suficiente. Para matrices simbólicas exactas, a veces puede ser necesario especificar algo como **ZeroTest->(FullSimplify[#]==0&)** para forzar un análisis más exhaustivo y comprobar si las expresiones simbólicas son cero.

Aquí hay una matriz simbólica simple con determinante cero:

```
In[•]:= m = {{a, b, c}, {2 a, 2 b, 2 c}, {3 a, 3 b, 3 c}}
Out[•]=
{{a, b, c}, {2 a, 2 b, 2 c}, {3 a, 3 b, 3 c}}
```

La base del espacio nulo de m contiene dos vectores:

In[ ]:=
```
NullSpace[m]
```
espacio nulo

Out[ ]=

$$\left\{\left\{-\frac{c}{a}, 0, 1\right\}, \left\{-\frac{b}{a}, 1, 0\right\}\right\}$$

Multiplicar m por cualquier combinación lineal de estos vectores da cero:

In[ ]:=
```
Simplify[m.(x %[[1]] + y %[[2]])]
```
simplifica

Out[ ]=
```
{0, 0, 0}
```

## Matrices rectangulares

Una característica importante de funciones como **LinearSolve** y **NullSpace** es que funcionan con matrices rectangulares y cuadradas.

Cuando se representa un sistema de ecuaciones lineales mediante una ecuación matricial de la forma **m.x == b**, el número de columnas en **m** indica el número de variables, y el número de filas indica el número de ecuaciones. Existen varios casos.

***Indeterminado (Undetermined)***
*Cuando el* número de ecuaciones es menor que el número de variables; pueden existir muchas soluciones o ninguna.

***Sobredeterminado (Undetermined)***
*Cuando el número de ecuaciones independientes es mayor que el número de variables; las soluciones pueden existir o no.*

***No singular (Nonsingular)***
*Cuando número de ecuaciones independientes igual al número de variables y determinante distinto de cero; existe una solución única.*

***Consistente (Consistent)***
*Existe al menos una solución.*

***Inconsistente (Inconsistent)***
*No existen soluciones.*

## Ejemplos:

Esto pide la solución al conjunto inconsistente de ecuaciones x == 1 y x == 0:

In[ ]:=
```
LinearSolve[{{1}, {1}}, {1, 0}]
```
resuelve ecuación lineal

LinearSolve : Linear equation encountered that has no solution.

Out[ ]=
```
LinearSolve[{{1}, {1}}, {1, 0}]
```

La siguiente matriz representa dos ecuaciones para tres variables:

In[ ]:=
```
m = {{1, 3, 4}, {2, 1, 3}}
```

Out[ ]=
```
{{1, 3, 4}, {2, 1, 3}}
```

```
In[ ]:= MatrixForm[%]
        forma de matriz
Out[ ]//MatrixForm=
        (1 3 4)
        (2 1 3)
```

**LinearSolve** proporciona una de las posibles soluciones a este conjunto subdeterminado de ecuaciones:

```
In[ ]:= v = LinearSolve[m, {1, 1}]
          resuelve ecuación lineal
Out[ ]=
        {2/5, 1/5, 0}
```

Cuando una matriz representa un sistema de ecuaciones indeterminado, la matriz tiene un espacio nulo no trivial. En este caso, el espacio nulo está abarcado por un único vector:

```
In[ ]:= NullSpace[m]
        espacio nulo
Out[ ]=
        {{-1, -1, 1}}
```

Si tomas la solución que obtienes de **LinearSolve** y agregas cualquier combinación lineal de los vectores base para el espacio nulo, aún obtienes una solución:

```
In[ ]:= m.(v + 4 %[[1]])
Out[ ]=
        {1, 1}
```

El número de ecuaciones independientes es el rango de la matriz **MatrixRank[m]**. El número de ecuaciones redundantes es **Length[NullSpace[m]]**. Nótese que la suma de estas cantidades siempre es igual al número de columnas en **m**.

```
In[ ]:= MatrixRank[m]
        rango matricial
Out[ ]=
        2

In[ ]:= Length[NullSpace[m]]
        longitud  espacio nulo
Out[ ]=
        1
```

***LinearSolve [m]***

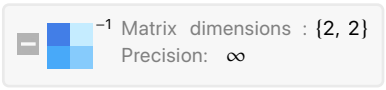
Genera una función para resolver ecuaciones en forma matricial: **m.x == b**.

En algunas aplicaciones, querrás resolver ecuaciones de la forma m.x == b varias veces con la misma **m**, pero con un valor de **b** diferente. Puedes hacerlo eficientemente en Wolfram Language usando **LinearSolve[m]** para crear una única función **LinearSolveFunction** que puedas aplicar a tantos vectores como desees.

Esto crea la **LinearSolveFuntion**:

In[•]:=
```
f = LinearSolve[{{1, 4}, {2, 3}}]
```
resuelve ecuación lineal

Out[•]=

LinearSolveFunction[  ]


Se puede aplicar esto al vector:

In[•]:=
```
f[{5, 7}]
```

Out[•]=

$$\left\{\frac{13}{5}, \frac{3}{5}\right\}$$

Obtendrás el mismo resultado al dar el vector como un segundo argumento explícito a LinearSolve:

In[•]:=
```
LinearSolve[{{1, 4}, {2, 3}}, {5, 7}]
```
resuelve ecuación lineal

Out[•]=

$$\left\{\frac{13}{5}, \frac{3}{5}\right\}$$

Podemos aplicar f a cualquier vector que quieras:

In[•]:=
```
f[{-5, 9}]
```

Out[•]=

$$\left\{\frac{51}{5}, -\frac{19}{5}\right\}$$

***LeastSquares [m, b]***

Da un vector x que resuelve el problema de mínimos cuadrados m.x == b

El siguiente sistema lineal es inconsistente:

In[•]:=
```
LinearSolve[{{1, 2}, {3, 4}, {5, 6}}, {-1, 0, 2}]
```
resuelve ecuación lineal

LinearSolve : Linear equation encountered that has no solution.

Out[•]=

```
LinearSolve[{{1, 2}, {3, 4}, {5, 6}}, {-1, 0, 2}]
```

**LeastSquares** encuentra un vector x que minimiza m.x-b en el sentido de mínimos cuadrados:

In[•]:=
```
LeastSquares[{{1, 2}, {3, 4}, {5, 6}}, {-1, 0, 2}]
```
mínimos cuadrados

Out[•]=

$$\left\{\frac{8}{3}, -\frac{23}{12}\right\}$$