# Resum-SageMath.pdf

user_2397943

Espais Vectorials

1º Grado en Ingeniería de Datos

Escuela de Ingeniería
Universidad Autónoma de Barcelona

# Matrius

## Creació Matrius

In [22]:
```
A=Matrix(QQ,10,10,[1/(j+k) for j in range(1,11) for k in range(1,11)])
show(A)
```

$$\begin{pmatrix} \frac{1}{2} & \frac{1}{3} & \frac{1}{4} & \frac{1}{5} & \frac{1}{6} & \frac{1}{7} & \frac{1}{8} & \frac{1}{9} & \frac{1}{10} & \frac{1}{11} \\ \frac{1}{3} & \frac{1}{4} & \frac{1}{5} & \frac{1}{6} & \frac{1}{7} & \frac{1}{8} & \frac{1}{9} & \frac{1}{10} & \frac{1}{11} & \frac{1}{12} \\ \frac{1}{4} & \frac{1}{5} & \frac{1}{6} & \frac{1}{7} & \frac{1}{8} & \frac{1}{9} & \frac{1}{10} & \frac{1}{11} & \frac{1}{12} & \frac{1}{13} \\ \frac{1}{5} & \frac{1}{6} & \frac{1}{7} & \frac{1}{8} & \frac{1}{9} & \frac{1}{10} & \frac{1}{11} & \frac{1}{12} & \frac{1}{13} & \frac{1}{14} \\ \frac{1}{6} & \frac{1}{7} & \frac{1}{8} & \frac{1}{9} & \frac{1}{10} & \frac{1}{11} & \frac{1}{12} & \frac{1}{13} & \frac{1}{14} & \frac{1}{15} \\ \frac{1}{7} & \frac{1}{8} & \frac{1}{9} & \frac{1}{10} & \frac{1}{11} & \frac{1}{12} & \frac{1}{13} & \frac{1}{14} & \frac{1}{15} & \frac{1}{16} \\ \frac{1}{8} & \frac{1}{9} & \frac{1}{10} & \frac{1}{11} & \frac{1}{12} & \frac{1}{13} & \frac{1}{14} & \frac{1}{15} & \frac{1}{16} & \frac{1}{17} \\ \frac{1}{9} & \frac{1}{10} & \frac{1}{11} & \frac{1}{12} & \frac{1}{13} & \frac{1}{14} & \frac{1}{15} & \frac{1}{16} & \frac{1}{17} & \frac{1}{18} \\ \frac{1}{10} & \frac{1}{11} & \frac{1}{12} & \frac{1}{13} & \frac{1}{14} & \frac{1}{15} & \frac{1}{16} & \frac{1}{17} & \frac{1}{18} & \frac{1}{19} \\ \frac{1}{11} & \frac{1}{12} & \frac{1}{13} & \frac{1}{14} & \frac{1}{15} & \frac{1}{16} & \frac{1}{17} & \frac{1}{18} & \frac{1}{19} & \frac{1}{20} \end{pmatrix}$$

### Rang d'una matriu

In [40]:
```
A.rank()
```

Out[40]: 10

## Matriu + Matriu

In [25]:
```
var('a')
C=matrix(4,4,[1,0,4,4,
             1,2,0,3,
             0,1,-2,-1,
              1,1,a,3])
show(C)

T=matrix(4,1,[1,0,a-1,a])
show(T)

S=block_matrix(1,2,[C,T])
show(S)
```

$$\begin{pmatrix} 1 & 0 & 4 & 4 \\ 1 & 2 & 0 & 3 \\ 0 & 1 & -2 & -1 \\ 1 & 1 & a & 3 \end{pmatrix}$$

$$\begin{pmatrix} 1 \\ 0 \\ a-1 \\ a \end{pmatrix}$$

$$\left(\begin{array}{cccc|c} 1 & 0 & 4 & 4 & 1 \\ 1 & 2 & 0 & 3 & 0 \\ 0 & 1 & -2 & -1 & a-1 \\ 1 & 1 & a & 3 & a \end{array}\right)$$

## Matriu identitat

In [23]:
```
identitat = identity_matrix(5)
show(identitat)
```

$$\begin{pmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{pmatrix}$$

### Matriu zero

In [3]:
```
zeros = zero_matrix(4,5)
show(zeros)
```

$$\begin{pmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$

### Diagonal

In [21]:
```
diagonal = diagonal_matrix([7,2,0,4,5,6])
show(diagonal)
```

$$\begin{pmatrix} 7 & 0 & 0 & 0 & 0 & 0 \\ 0 & 2 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 4 & 0 & 0 \\ 0 & 0 & 0 & 0 & 5 & 0 \\ 0 & 0 & 0 & 0 & 0 & 6 \end{pmatrix}$$

## Vectors

### Creació Vectors

In [6]:
```
v = vector(QQ,[11,12,..16])
show(v)
```

$$(11, 12, 13, 14, 15, 16)$$

### Construcció s.e.v - e.v generat per uns quants vectors

In [10]:
```
V=span(QQ,[[1,2,1],[2,0,1],[-1,2,0],[3,2,2]])
show(V) # s'eliminen generadors innecessaris - s'ensenya la reducció G-J
aux = matrix(QQ,[[1,2,1],[2,0,1],[-1,2,0],[3,2,2]])
show(aux)
show(aux.extended_echelon_form())
```

$$\mathrm{RowSpan}_{\mathbf{Q}} \begin{pmatrix} 1 & 0 & \frac{1}{2} \\ 0 & 1 & \frac{1}{4} \end{pmatrix}$$

$$\begin{pmatrix} 1 & 2 & 1 \\ 2 & 0 & 1 \\ -1 & 2 & 0 \\ 3 & 2 & 2 \end{pmatrix}$$

$$\begin{pmatrix} 1 & 0 & \frac{1}{2} & 0 & 0 & -\frac{1}{4} & \frac{1}{4} \\ 0 & 1 & \frac{1}{4} & 0 & 0 & \frac{3}{8} & \frac{1}{8} \\ 0 & 0 & 0 & 1 & 0 & -\frac{1}{2} & -\frac{1}{2} \\ 0 & 0 & 0 & 0 & 1 & \frac{1}{2} & -\frac{1}{2} \end{pmatrix}$$

In [16]:
```
v=vector([1,2,3])
show(v)
v in V #per saber si v és subespai vectorial de V
w=vector([20,-4,9])
show(w)
w in V
w1=V.basis() #base més senzilla (a partir de GJ) -- una família independent de generadors
show(w1)
show(V.dimension()) #nombre de vectors d'una base
```

$(1,\ 2,\ 3)$

$(20,\ -4,\ 9)$

$\left[\left(1,\ 0,\ \frac{1}{2}\right),\left(0,\ 1,\ \frac{1}{4}\right)\right]$

$2$

## Matriu + Vector

In [22]:
```
v = vector(QQ,[11,12,..16])
sistema = diagonal.augment(v,subdivide=True)
show(sistema)
show(sistema.echelon_form())

w = matrix(QQ,6,1,[11,12,..16])
dw = block_matrix(1,2,[diagonal, w])
show(dw.extended_echelon_form())
```

$$\left(\begin{array}{rrrrrr|r} 7 & 0 & 0 & 0 & 0 & 0 & 11 \\ 0 & 2 & 0 & 0 & 0 & 0 & 12 \\ 0 & 0 & 0 & 0 & 0 & 0 & 13 \\ 0 & 0 & 0 & 4 & 0 & 0 & 14 \\ 0 & 0 & 0 & 0 & 5 & 0 & 15 \\ 0 & 0 & 0 & 0 & 0 & 6 & 16 \end{array}\right)$$

$$\left(\begin{array}{rrrrrr|r} 7 & 0 & 0 & 0 & 0 & 0 & 11 \\ 0 & 2 & 0 & 0 & 0 & 0 & 12 \\ 0 & 0 & 0 & 4 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 5 & 0 & 2 \\ 0 & 0 & 0 & 0 & 0 & 6 & 3 \\ 0 & 0 & 0 & 0 & 0 & 0 & 13 \end{array}\right)$$

$$\left(\begin{array}{rrrrrrrrrrrrr} 1 & 0 & 0 & 0 & 0 & 0 & 0 & \frac{1}{7} & 0 & -\frac{11}{91} & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & \frac{1}{2} & -\frac{6}{13} & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & -\frac{7}{26} & \frac{1}{4} & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & -\frac{3}{13} & 0 & \frac{1}{5} & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & -\frac{8}{39} & 0 & 0 & \frac{1}{6} \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & \frac{1}{13} & 0 & 0 & 0 \end{array}\right)$$

## Canvi de files

In [21]:
```
sistema.swap_rows(1,2)
show(sistema)
```

$$\left(\begin{array}{rrrrrr|r} 7 & 0 & 0 & 0 & 0 & 0 & 11 \\ 0 & 2 & 0 & 0 & 0 & 0 & 12 \\ 0 & 0 & 0 & 0 & 0 & 0 & 13 \\ 0 & 0 & 0 & 4 & 0 & 0 & 14 \\ 0 & 0 & 0 & 0 & 5 & 0 & 15 \\ 0 & 0 & 0 & 0 & 0 & 6 & 16 \end{array}\right)$$

**A.add_multiple_of_row(i,j,a) = a*(row j) + row i**

In [24]:
```
sistema.add_multiple_of_row(3,1,1000)
show(sistema)
```

$$\left(\begin{array}{rrrrrr|r} 7 & 0 & 0 & 0 & 0 & 0 & 11 \\ 0 & 2 & 0 & 0 & 0 & 0 & 12 \\ 0 & 0 & 0 & 0 & 0 & 0 & 13 \\ 0 & 2000 & 0 & 4 & 0 & 0 & 12014 \\ 0 & 0 & 0 & 0 & 5 & 0 & 15 \\ 0 & 0 & 0 & 0 & 0 & 6 & 16 \end{array}\right)$$

## A.rescale_row(i,a) = a*(row i)

In [18]:
```
sistema.rescale_row(4,10)
show(sistema)
```

$$\left(\begin{array}{rrrrrr|r} 7 & 0 & 0 & 0 & 0 & 0 & 11 \\ 0 & 2 & 0 & 0 & 0 & 0 & 12 \\ 0 & 0 & 0 & 0 & 0 & 0 & 13 \\ 0 & 4000 & 0 & 4 & 0 & 0 & 24014 \\ 0 & 0 & 0 & 0 & 50 & 0 & 150 \\ 0 & 0 & 0 & 0 & 0 & 6 & 16 \end{array}\right)$$

# Substitució variable

In [83]:
```
S2 = S.subs(a=2)
show(S)
show(S2)
```

$$\left(\begin{array}{rrrr|r} 1 & 0 & 4 & 4 & 1 \\ 1 & 2 & 0 & 3 & 0 \\ 0 & 1 & -2 & -1 & a-1 \\ 1 & 1 & a & 3 & a \end{array}\right)$$

$$\left(\begin{array}{rrrrr} 1 & 0 & 4 & 4 & 1 \\ 1 & 2 & 0 & 3 & 0 \\ 0 & 1 & -2 & -1 & 1 \\ 1 & 1 & 2 & 3 & 2 \end{array}\right)$$

# Gauss-Jordan

In [89]:
```
# show(S2.jordan_form(transformation=True)) -- NOMÉS VA B SI MATRIU QUADRADA
show(S2.echelon_form()) # Gauss-Jordan
show(S2.extended_echelon_form()) # Gauss-Jordan amb matriu informació

S2_extended = copy(S2.extended_echelon_form())
S2_extended.subdivide([], [5])
show(S2_extended)
```

$$\left(\begin{array}{rrrrr} 1 & 0 & 4 & 0 & 13 \\ 0 & 1 & -2 & 0 & -2 \\ 0 & 0 & 0 & 1 & -3 \\ 0 & 0 & 0 & 0 & 0 \end{array}\right)$$

$$\left(\begin{array}{rrrrrrrrr} 1 & 0 & 4 & 0 & 13 & 0 & -4 & 3 & 5 \\ 0 & 1 & -2 & 0 & -2 & 0 & 1 & 0 & -1 \\ 0 & 0 & 0 & 1 & -3 & 0 & 1 & -1 & -1 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & -1 \end{array}\right)$$

$$\left(\begin{array}{rrrrr|rrrr} 1 & 0 & 4 & 0 & 13 & 0 & -4 & 3 & 5 \\ 0 & 1 & -2 & 0 & -2 & 0 & 1 & 0 & -1 \\ 0 & 0 & 0 & 1 & -3 & 0 & 1 & -1 & -1 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & -1 \end{array}\right)$$

**Millor manera per veure G-J**

In [8]:
```
var('a')
C=matrix(4,4,[1,0,4,4,
             1,2,0,3,
             0,1,-2,-1,
              1,1,a,3])
I = identity_matrix(4)
S=block_matrix(1,2,[C,I])
show(S)
show(C.extended_echelon_form())
show(S.echelon_form())
```

$$\left(\begin{array}{rrrr|rrrr} 1 & 0 & 4 & 4 & 1 & 0 & 0 & 0 \\ 1 & 2 & 0 & 3 & 0 & 1 & 0 & 0 \\ 0 & 1 & -2 & -1 & 0 & 0 & 1 & 0 \\ 1 & 1 & a & 3 & 0 & 0 & 0 & 1 \end{array}\right)$$

$$\left(\begin{array}{rrrrrrrr} 1 & 0 & 0 & 0 & \frac{4}{a-2}+5 & -4 & \frac{4}{a-2}+8 & -\frac{4}{a-2} \\ 0 & 1 & 0 & 0 & -\frac{2}{a-2}-1 & 1 & -\frac{2}{a-2}-1 & \frac{2}{a-2} \\ 0 & 0 & 1 & 0 & -\frac{1}{a-2} & 0 & -\frac{1}{a-2} & \frac{1}{a-2} \\ 0 & 0 & 0 & 1 & -1 & 1 & -2 & 0 \end{array}\right)$$

$$\left(\begin{array}{rrrr|rrrr} 1 & 0 & 0 & 0 & \frac{4}{a-2}+5 & -4 & \frac{4}{a-2}+8 & -\frac{4}{a-2} \\ 0 & 1 & 0 & 0 & -\frac{2}{a-2}-1 & 1 & -\frac{2}{a-2}-1 & \frac{2}{a-2} \\ 0 & 0 & 1 & 0 & -\frac{1}{a-2} & 0 & -\frac{1}{a-2} & \frac{1}{a-2} \\ 0 & 0 & 0 & 1 & -1 & 1 & -2 & 0 \end{array}\right)$$

**Per quedar-nos la matriu informació**

In [41]:
```
P=S2.extended_echelon_form().matrix_from_columns([5..8])
show(P)
```

$$\left(\begin{array}{rrrr} 0 & -4 & 3 & 5 \\ 0 & 1 & 0 & -1 \\ 0 & 1 & -1 & -1 \\ 1 & 0 & 1 & -1 \end{array}\right)$$

## Pseudoinversa / Inversa generalitzada

In [23]:
```
A = Matrix(ZZ,5,5,[1,0,-1,0,1,0,1,0,0,0,0,-1,0,1,0,0,0,-1,0,-1,0,1,0,-1,0])
show(A)
```

$$\left(\begin{array}{rrrrr} 1 & 0 & -1 & 0 & 1 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & -1 & 0 & 1 & 0 \\ 0 & 0 & -1 & 0 & -1 \\ 0 & 1 & 0 & -1 & 0 \end{array}\right)$$

In [25]:
```
# S'ha de complir AGA = A

G = A.pseudoinverse() # sempre existeix un G inversa generalitzada
show(G)
show(A*G*A)
show(A)
```

$$\left(\left(\begin{array}{rr|rrr} 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ \hline 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 \end{array}\right), \left(\begin{array}{rrrrr} 2 & 0 & \frac{2}{3} & \frac{2}{3} & 1 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & -1 & \frac{1}{3} & 0 & 0 \\ 0 & 0 & 0 & \frac{1}{3} & 0 \\ 0 & 1 & -\frac{1}{3} & 0 & -\frac{1}{3} \end{array}\right)\right)$$

$$\left(\begin{array}{rrrrr} \frac{1}{3} & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ -\frac{1}{3} & 0 & 0 & -\frac{1}{2} & 0 \\ 0 & 1 & \frac{1}{2} & 0 & -\frac{1}{2} \\ \frac{1}{3} & 0 & 0 & -\frac{1}{2} & 0 \end{array}\right)$$

$$\left(\begin{array}{rrrrr} 1 & 0 & -1 & 0 & 1 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & -1 & 0 & 1 & 0 \\ 0 & 0 & -1 & 0 & -1 \\ 0 & 1 & 0 & -1 & 0 \end{array}\right)$$

$$\left(\begin{array}{rrrrr} 1 & 0 & -1 & 0 & 1 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & -1 & 0 & 1 & 0 \\ 0 & 0 & -1 & 0 & -1 \\ 0 & 1 & 0 & -1 & 0 \end{array}\right)$$

### PAQ reducció

In [33]:
```
"""
RECORDATORI PAQ-reducció

1r// (A|I filesA) ~ (GJ A|P)
2n// ((GJ A)**t|I columnA) ~((PAQ)**t|Q**t)


Com trobar PAQ reducció amb Sage i una P i una Q?

A és 5x5
"""
GJ_ns = A.jordan_form() # no dona GJ de A
GJ = A.echelon_form()
show(GJ)
show(GJ_ns)
GJtransp = GJ.transpose()
trPAQ = GJtransp.extended_echelon_form()
trQ = trPAQ.matrix_from_columns([0..4])
PAQ = trPAQ.transpose() # és una matriu amb la matriu identitat amb mida rang matriu i la resta 0
Q = trQ.transpose()

# G = Q*trPAQ*P
```

$$\left(\begin{array}{rrrrr} 1 & 0 & 0 & 0 & 2 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{array}\right)$$

$$\left(\begin{array}{rr|rrr} 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ \hline 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 \end{array}\right)$$

In [53]:
```
show(A.diagonal())
```

$$[1, 1, 0, 0, 0]$$

## Resolució de sistemes

In [66]:
```
C = Matrix(ZZ, [[1, 4, 1],[4, 13, 7],[7, 22, 13]])
b = Matrix(ZZ, 3, 1, [0, 0, 0])
show(b)
show(C)
show(C.solve_right(b))
show(C.column_space()) # rang de columnes
print("Gauss-Jordan")
show(C.echelon_form())
```

$$\begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}$$

$$\begin{pmatrix} 1 & 4 & 1 \\ 4 & 13 & 7 \\ 7 & 22 & 13 \end{pmatrix}$$

$$\begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}$$

$$\mathrm{RowSpan}_{\mathbf{Z}} \begin{pmatrix} 1 & 1 & 1 \\ 0 & 3 & 6 \end{pmatrix}$$

```
Gauss-Jordan
```

$$\begin{pmatrix} 1 & 1 & 4 \\ 0 & 3 & -3 \\ 0 & 0 & 0 \end{pmatrix}$$

### Solucions d'un sist. d'eq homogeni

In [71]:
```
var('y z t')
equacions=[
    x-y+t==0,
    x+y+z+t==0
]
show(equacions)
coeficients=[
    [
        eq.lhs().coefficient(v)
        for v in [x,y,z,t]
    ]
    for eq in equacions
]
show(coeficients)
A=matrix(QQ,coeficients)
show(A)
```

$$[t + x - y = 0, t + x + y + z = 0]$$

$$[[1, -1, 0, 1], [1, 1, 1, 1]]$$

$$\begin{pmatrix} 1 & -1 & 0 & 1 \\ 1 & 1 & 1 & 1 \end{pmatrix}$$

In [73]:
```
W=A.right_kernel() #és un e.v
show(W) #ens passa la solució del sist. fins arribar a GJ
```

$$\mathrm{RowSpan}_{\mathbf{Q}} \begin{pmatrix} 1 & 0 & 0 & -1 \\ 0 & 1 & -2 & 1 \end{pmatrix}$$

### Sumes i interseccions de subespais vectorials

In [90]:
```
v1=vector(QQ,[1,1,-1]) # si afegim un núm a v1 i a v2, a la comanda S=F+G ens dona error
v2=vector(QQ,[2,0,-1]) # si traiem les QQ tmb error
w1=vector(QQ,[1,0,-1])
w2=vector(QQ,[2,3,0])
w3=vector(QQ,[4,3,-2])
F=span([v1,v2])
G=span([w1,w2,w3])
show(F)
show(G)
```

$$\mathrm{RowSpan}_{\mathbf{Q}}\begin{pmatrix} 1 & 0 & -\frac{1}{2} \\ 0 & 1 & -\frac{1}{2} \end{pmatrix}$$

$$\mathrm{RowSpan}_{\mathbf{Q}}\begin{pmatrix} 1 & 0 & -1 \\ 0 & 1 & \frac{2}{3} \end{pmatrix}$$

In [93]:
```
# Suma i intersecció

S=F+G
show(S)
show(S.dimension())

"""
dimensió F = 2
dim G = 2
dim F+G = 0 --> dim F intersecció G = 1
"""

Intrs=F.intersection(G)
show(Intrs.basis())
show(Intrs)
show(Intrs.dimension())
```

$$\mathrm{RowSpan}_{\mathbf{Q}}\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

3

$$\left[\left(1, \frac{3}{7}, -\frac{5}{7}\right)\right]$$

$$\mathrm{RowSpan}_{\mathbf{Q}}\begin{pmatrix} 1 & \frac{3}{7} & -\frac{5}{7} \end{pmatrix}$$

1

## Definició subespai de R format per x files de la matriu A

Definiu el subespai $F$ de $\mathbb{Q}^7$ format per les primeres 4 files de la matriu $A$.

In [ ]:
```
F=(QQ^7).span(A.matrix_from_rows([0..3]))
show(F)

# per fer agafar det columnes fer la transposada
```

## Canvi de coordenades

In [95]:
```
v1=vector(QQ,[1,-1,1])
v2=vector(QQ,[2,3,-1])
E=span(QQ,[v1,v2]) # espai ambient, definit amb comb. linelas dels vectors, agafant els eixos més senzills
Ev=(QQ^3).subspace_with_basis([v1,v2])
show(E)
show(Ev)
show("Son iguals?")
show(E==Ev)
v=vector(QQ,[4,1,1])
show(v)
v in E
v in Ev

Ev.coordinates(v)
2*v1+v2,v
E.coordinates(v)
E.0 # eix q agafa E predeterminadament
E.1 # eix q agafa E predeterminadament
```

$$\mathrm{RowSpan}_{\mathbb{Q}}\begin{pmatrix} 1 & 0 & \frac{2}{5} \\ 0 & 1 & -\frac{3}{5} \end{pmatrix}$$

$$\mathrm{RowSpan}_{\mathbb{Q}}\begin{pmatrix} 1 & -1 & 1 \\ 2 & 3 & -1 \end{pmatrix}$$

Son iguals?

True

$(4, 1, 1)$

Out[95]: (0, 1, -3/5)

## Polinomi característic

In [101]:
```
A=matrix(QQ,4,4,[3, -6, -8, 6,
                 2, -3, -4, 4,
                 -2, 3, 3, -3,
                 -2, 1, -2, 0])
show(A)
show(A.charpoly())
show((A-x*identity_matrix(4)).det())
show(A.charpoly('t')) # per diferenciar variables en cas d'haver una x a la matriu
show("descomposició: ", A.charpoly().factor())
```

$$\begin{pmatrix} 3 & -6 & -8 & 6 \\ 2 & -3 & -4 & 4 \\ -2 & 3 & 3 & -3 \\ -2 & 1 & -2 & 0 \end{pmatrix}$$

$x^4 - 3x^3 + x^2 + 3x - 2$

$x^4 - 3\,x^3 + x^2 + 3\,x - 2$

$t^4 - 3t^3 + t^2 + 3t - 2$

descomposició:$(x-2)\cdot(x+1)\cdot(x-1)^2$

In [ ]:
```
var('a')
diag = diagonal_matrix([-a,-a,-a])
mat_x = aux + diag
substitucio = mat_x.subs(a=92.6112784468182)
```

In [98]:
```
A.is_diagonalizable() # problema d'aproximació
```

Out[98]: True

## Polinomi mínim

In [106]:
```
show(A.minimal_polynomial())
```

$x^3 - 2x^2 - x + 2$

### valors propis

In [102]:
```
show(A.eigenvalues())
```

$[2, -1, 1, 1]$

### vectors propis

In [104]:
```
show(A.eigenspaces_right()) # vectors per la dreta, iguala la matriu nucli a 0

# ens retorna (vap, base de vap)
```

$$\left[\left(2, \mathrm{RowSpan}_{\mathbf{Q}}\begin{pmatrix}1 & \frac{2}{3} & -\frac{1}{2} & -\frac{1}{6}\end{pmatrix}\right), \left(-1, \mathrm{RowSpan}_{\mathbf{Q}}\begin{pmatrix}1 & 1 & -1 & -1\end{pmatrix}\right), \left(1, \mathrm{RowSpan}_{\mathbf{Q}}\begin{pmatrix}1 & 0 & -\frac{1}{2} & -1 \\ 0 & 1 & 0 & 1\end{pmatrix}\right)\right]$$

In [107]:
```
# llista de tripletes formades pel valor propi, una base de
# l'espai de vectors propis associats i la multiplicitat.
show(A.eigenvectors_right())
```

$$\left[\left(2, \left[\left(1, \frac{2}{3}, -\frac{1}{2}, -\frac{1}{6}\right)\right], 1\right), \left(-1, \left[(1, 1, -1, -1)\right], 1\right), \left(1, \left[\left(1, 0, -\frac{1}{2}, -1\right), (0, 1, 0, 1)\right], 2\right)\right]$$

#### impressió VAPs i VEPs

In [109]:
```
VP = A.eigenspaces_right() # injectiva (nucli vector 0) --> NO ho és pq té VAP = 0
show(VP)

for vp in VP:
    show("valor propi: ")
    show(vp[0])
    show("base:")
    show(vp[1].basis())
    print("-------")
```

$$\left[\left(2, \mathrm{RowSpan}_{\mathbf{Q}}\begin{pmatrix}1 & \frac{2}{3} & -\frac{1}{2} & -\frac{1}{6}\end{pmatrix}\right), \left(-1, \mathrm{RowSpan}_{\mathbf{Q}}\begin{pmatrix}1 & 1 & -1 & -1\end{pmatrix}\right), \left(1, \mathrm{RowSpan}_{\mathbf{Q}}\begin{pmatrix}1 & 0 & -\frac{1}{2} & -1 \\ 0 & 1 & 0 & 1\end{pmatrix}\right)\right]$$

valor propi:

$2$

base:

$$\left[\left(1, \frac{2}{3}, -\frac{1}{2}, -\frac{1}{6}\right)\right]$$

-------

valor propi:

$-1$

base:

$[(1, 1, -1, -1)]$

-------

valor propi:

$1$

base:

$$\left[\left(1, 0, -\frac{1}{2}, -1\right), (0, 1, 0, 1)\right]$$

-------

## Matriu canvi de base

In [111]:
```
[D,M]=A.eigenmatrix_right()
show(D)
show(M) # M = P
```

$$\begin{pmatrix} 2 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

$$\begin{pmatrix} 1 & 1 & 1 & 0 \\ \frac{2}{3} & 1 & 0 & 1 \\ -\frac{1}{2} & -1 & -\frac{1}{2} & 0 \\ -\frac{1}{6} & -1 & -1 & 1 \end{pmatrix}$$

In [112]:
```
show(M.inverse()*A*M)
show(D)
```

$$\begin{pmatrix} 2 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

$$\begin{pmatrix} 2 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

In [113]:
```
show(M*D*M.inverse())
show(A)
```

$$\begin{pmatrix} 3 & -6 & -8 & 6 \\ 2 & -3 & -4 & 4 \\ -2 & 3 & 3 & -3 \\ -2 & 1 & -2 & 0 \end{pmatrix}$$

$$\begin{pmatrix} 3 & -6 & -8 & 6 \\ 2 & -3 & -4 & 4 \\ -2 & 3 & 3 & -3 \\ -2 & 1 & -2 & 0 \end{pmatrix}$$

## Gram-Schmidt

In [23]:
```
v1 = vector(RR,[1,1,1,1])
v2 = vector(RR,[-1,4,4,-1])
v3 = vector(RR,[4,-2,2,0])
B = [v1,v2,v3]
F = (RR^4).subspace_with_basis(B)
show(F)
type(F)
```

$$\mathrm{RowSpan}_{\mathbf{R}}\begin{pmatrix} 1.00000000000000 & 1.00000000000000 & 1.00000000000000 & 1.00000000000000 \\ -1.00000000000000 & 4.00000000000000 & 4.00000000000000 & -1.00000000000000 \\ 4.00000000000000 & -2.00000000000000 & 2.00000000000000 & 0.000000000000000 \end{pmatrix}$$

Out[23]:
```
<class 'sage.modules.free_module.FreeModule_submodule_with_basis_field_with_category'>
```

In [3]:
```
BO = [] # on afegim els vectors de la base ortogonal
w1 = B[0]/sqrt(B[0]*B[0]) # vector entre la seva norma (1r vector de la base)
BO.append(w1)
print(BO[0]," té norma ",sqrt(BO[0]*BO[0]))
```

```
(0.500000000000000, 0.500000000000000, 0.500000000000000, 0.500000000000000)  té norma  1.00000000000000
```

In [4]:
```
w2temp = B[1] - (B[1]*BO[0])*BO[0]
w2 = w2temp/sqrt(w2temp*w2temp)  # segon vector
BO.append(w2)
print(BO)
print("Els dos primers vectors de B i BO generen el mateix: ",span(B[0:2]) == span(BO))
print(BO[1]," té norma ",sqrt(BO[1]*BO[1]))
print("El producte escalar entre ",BO[0]," i ",BO[1]," és ",BO[0]*BO[1])
```

```
[(0.500000000000000, 0.500000000000000, 0.500000000000000, 0.500000000000000), (-0.500000000000000, 0.50000000000000
0, 0.500000000000000, -0.500000000000000)]
Els dos primers vectors de B i BO generen el mateix:  True
(-0.500000000000000, 0.500000000000000, 0.500000000000000, -0.500000000000000)  té norma  1.00000000000000
El producte escalar entre  (0.500000000000000, 0.500000000000000, 0.500000000000000, 0.500000000000000)  i  (-0.50000
0000000000, 0.500000000000000, 0.500000000000000, -0.500000000000000)  és  0.000000000000000
```

In [6]:
```
w3temp = B[2] - sum([(B[2]*BO[j])*BO[j] for j in range(2)])
w3 = w3temp/sqrt(w3temp*w3temp)
BO.append(w3)
print(BO)
print(span(RR,B[0:3]) == span(RR,BO))
print(BO[2]," té norma ",sqrt(BO[2]*BO[2]))
for j in range(3):
    for k in range(j,3):
        print("El producte escalar entre ",BO[j]," i ",BO[k]," és ",BO[j]*BO[k])
```

```
[(0.500000000000000, 0.500000000000000, 0.500000000000000, 0.500000000000000), (-0.500000000000000, 0.50000000000000
0, 0.500000000000000, -0.500000000000000), (0.500000000000000, -0.500000000000000, 0.500000000000000, -0.500000000000
000), (0.500000000000000, -0.500000000000000, 0.500000000000000, -0.500000000000000)]
True
(0.500000000000000, -0.500000000000000, 0.500000000000000, -0.500000000000000)  té norma  1.00000000000000
El producte escalar entre  (0.500000000000000, 0.500000000000000, 0.500000000000000, 0.500000000000000)  i  (0.500000
000000000, 0.500000000000000, 0.500000000000000, 0.500000000000000)  és  1.00000000000000
El producte escalar entre  (0.500000000000000, 0.500000000000000, 0.500000000000000, 0.500000000000000)  i  (-0.50000
0000000000, 0.500000000000000, 0.500000000000000, -0.500000000000000)  és  0.000000000000000
El producte escalar entre  (0.500000000000000, 0.500000000000000, 0.500000000000000, 0.500000000000000)  i  (0.500000
000000000, -0.500000000000000, 0.500000000000000, -0.500000000000000)  és  0.000000000000000
El producte escalar entre  (-0.500000000000000, 0.500000000000000, 0.500000000000000, -0.500000000000000)  i  (-0.500
000000000000, 0.500000000000000, 0.500000000000000, -0.500000000000000)  és  1.00000000000000
El producte escalar entre  (-0.500000000000000, 0.500000000000000, 0.500000000000000, -0.500000000000000)  i  (0.5000
00000000000, -0.500000000000000, 0.500000000000000, -0.500000000000000)  és  0.000000000000000
El producte escalar entre  (0.500000000000000, -0.500000000000000, 0.500000000000000, -0.500000000000000)  i  (0.5000
00000000000, -0.500000000000000, 0.500000000000000, -0.500000000000000)  és  1.00000000000000
```

## Diagonalització per matrius simètriques

In [7]:
```
"""
Si tenim A quadrada i simètrica sempre diagonalitza. És a dir existeix una matriu P invertible i D diagonal on A=PDP^
1.
Però a més existeix P1 invertible on A = P1*D*P1^-1 i ORTOGONAL (=> P^-1 = P^transposta)
"""
A=matrix(QQ,3,3,[1,2,3,2,1,2,3,2,1]);
show(A)
```

$$\begin{pmatrix} 1 & 2 & 3 \\ 2 & 1 & 2 \\ 3 & 2 & 1 \end{pmatrix}$$

In [8]:
```
D,P=A.eigenmatrix_right();
show(P); # són els vectors de cada VAP posats ens columnes
show(D); # ens dona una aproximació de la diagonal => tenim 3 valors propis
```

$$\begin{pmatrix} 1 & 1 & 1 \\ 0 & -2.350781059358213? & 0.8507810593582122? \\ -1 & 1 & 1 \end{pmatrix}$$

$$\begin{pmatrix} -2 & 0 & 0 \\ 0 & -0.7015621187164243? & 0 \\ 0 & 0 & 5.701562118716425? \end{pmatrix}$$

In [9]:
```
"""
Si fem P*Ptransposada i no surt MId --> P NO ORTOGONAL
"""
P2=P.transpose();
show(P2*P)
```

$$\begin{pmatrix} 2 & 0 & 0 \\ 0 & 7.526171589037319? & 0.?e-18 \\ 0 & 0.?e-18 & 2.723828410962682? \end{pmatrix}$$

In [10]:
```
show(P2.gram_schmidt()[0]); # ho fa en files
R=P2.gram_schmidt()[0].transpose(); # fer transposta i després normes
show(R)
```

$$\begin{pmatrix} 1 & 0 & -1 \\ 1 & -2.350781059358213? & 1 \\ 1.000000000000000? & 0.8507810593582122? & 1.000000000000000? \end{pmatrix}$$

$$\begin{pmatrix} 1 & 1 & 1.000000000000000? \\ 0 & -2.350781059358213? & 0.8507810593582122? \\ -1 & 1 & 1.000000000000000? \end{pmatrix}$$

### funció normalització columnes

In [12]:
```
def normalize_columns(M):
    return column_matrix([v/v.norm() for v in M.columns()]);
P3=normalize_columns(R);
show(P3)
show(P3.transpose()*P3)
```

$$\begin{pmatrix} 0.7071067811865475? & 0.3645129333556568? & 0.6059128001754497? \\ 0 & -0.8568900996623581? & 0.515499134011970? \\ -0.7071067811865475? & 0.3645129333556568? & 0.6059128001754497? \end{pmatrix}$$

$$\begin{pmatrix} 1.000000000000000? & 0.?e-18 & 0.?e-18 \\ 0.?e-18 & 1.000000000000000? & 0.?e-18 \\ 0.?e-18 & 0.?e-18 & 1.000000000000000? \end{pmatrix}$$

In [13]:
```
show(P3*D*P3.transpose()); # ENS HA DE SORTIR LA M ORTOGONAL BUSCADA
show(A);
```

$$\begin{pmatrix} 1.000000000000000? & 2.000000000000000? & 3.000000000000000? \\ 2.000000000000000? & 1.000000000000000? & 2.000000000000000? \\ 3.000000000000000? & 2.000000000000000? & 1.000000000000000? \end{pmatrix}$$

$$\begin{pmatrix} 1 & 2 & 3 \\ 2 & 1 & 2 \\ 3 & 2 & 1 \end{pmatrix}$$

### funció orotgonalitació

In [14]:
```
def ortogonalitzacioGM(B):
    # Omplir el codi aquí
    # hauríem de posar un if per veure si els vectors són LI
    # es podria fer una matriu i mirar el rang
    BO = []
    for i,v in enumerate(B):
        wt = v - sum([(v*BO[j])*BO[j] for j in range(i)])
        w = wt/sqrt(wt*wt)
        BO.append(w)
    return BO # retorna els vecotrs de la BOrtogonal
"""
Si hem creat un espai com a continuació, hem de passar a la funció B.basis() pq no hi hagi divisió 0
"""
ex4 = matrix(QQ,4,4,[1,2,3,4,2,3,4,5,3,4,5,6,1,1,2,1])
F=(QQ^4).span(ex4.matrix_from_rows([0..3]))
show(ex4)
U,X=matrix(F.basis()).gram_schmidt()
show(U)
show(X)
show([u/u.norm() for u in U.rows()])
show(ortogonalitzacioGM(F.basis()))
```

## Factorització de Valors Singulars

Hem vist a classe la factorització en valors singulars on A=USigmaV^t on U,V matrius ortogonals i Sigma sol té valors no zeros en les posicions (i,i) de la matriu, corresponent als valors singulars de A de més gran a més petits

In [17]:
```
C=matrix(CDF, [[1,2,3],[1,4,7]]); # CDF és un aproximació molt precisa
show(C)
Uu,Sig,Vv=C.SVD(); # Singular Value Decomposition
show(Sig);show(Uu);show(Vv);
show(Uu*Sig*Vv.transpose()); # surt igual que C
```

$$\begin{pmatrix} 1.0 & 2.0 & 3.0 \\ 1.0 & 4.0 & 7.0 \end{pmatrix}$$

$$\begin{pmatrix} 8.927422163343595 & 0.0 & 0.0 \\ 0.0 & 0.5487563370400236 & 0.0 \end{pmatrix}$$

$$\begin{pmatrix} -0.4153730020007066 & -0.909651179963463 \\ -0.909651179963463 & 0.4153730020007066 \end{pmatrix}$$

$$\begin{pmatrix} -0.14842181289519157 & -0.9007243189734793 & 0.4082482904638637 \\ -0.5006317212382568 & -0.28757818593098095 & -0.816496580927726 \\ -0.8528416295813221 & 0.3255679471115185 & 0.40824829046386274 \end{pmatrix}$$

$$\begin{pmatrix} 0.9999999999999993 & 1.9999999999999998 & 2.9999999999999996 \\ 0.9999999999999996 & 4.0 & 7.000000000000001 \end{pmatrix}$$

In [ ]:
```
show(Uu) # es troben a partir de cada valor singular de A*At --> (1/v.sing)*(S)*(vector de v.sing)
show(Sig) # són els VAPs de A
show(Vv) # són els VEPs (en base ortonormal) de A*Atransformada - VAP(Id)
```

In [18]:
```
CtranspC = C.transpose()*C
show(CtranspC)
# busquem valors propis i base de A^t*A
(CtranspC.eigenvalues())
sqrt(CtranspC.eigenvalues()[1])
```

$$\begin{pmatrix} 2.0 & 6.0 & 10.0 \\ 6.0 & 20.0 & 34.0 \\ 10.0 & 34.0 & 58.0 \end{pmatrix}$$

Out[18]: 0.5487563370400209

In [19]:
```
# no ho sé si va aquí
D, P = CtranspC.eigenmatrix_right()
show(D)
show(P)
show(P*P.transpose())
# ens surt gairebé tot és 0 -- és gairebé ortogonal
show(normalize_columns(P)) # surt igual que la Vv
```

$$\begin{pmatrix} 79.69886648255844 & 0.0 & 0.0 \\ 0.0 & 0.30113351744158107 & 0.0 \\ 0.0 & 0.0 & 7.881163252378632 \times 10^{-16} \end{pmatrix}$$

$$\begin{pmatrix} 0.14842181289519124 & 0.9007243189734789 & -0.4082482904638529 \\ 0.5006317212382569 & 0.28757818593098083 & 0.8164965809277293 \\ 0.852841629581322 & -0.32556794711151865 & -0.40824829046386674 \end{pmatrix}$$

$$\begin{pmatrix} 0.999999999999991 & 6.938893903907228 \times 10^{-15} & -3.247402347028583 \times 10^{-15} \\ 6.938893903907228 \times 10^{-15} & 1.0000000000000056 & -4.551914400963142 \times 10^{-15} \\ -3.247402347028583 \times 10^{-15} & -4.551914400963142 \times 10^{-15} & 1.000000000000003 \end{pmatrix}$$

$$\begin{pmatrix} 0.14842181289519124 & 0.9007243189734792 & -0.4082482904638529 \\ 0.5006317212382569 & 0.2875781859309809 & 0.8164965809277293 \\ 0.852841629581322 & -0.3255679471115187 & -0.40824829046386674 \end{pmatrix}$$

## Solució mínims quadrats

In [24]:
```
ex3 = matrix(3,3,[1,2,3,2,3,4,3,4,5])
v = matrix(3,1,[1,5,-2])
# show(ex3.solve_right(v)) -- ens dona error pq SI

# apliquem A*At*(x,y,z) = At*v
i = ex3*ex3.transpose()
show(i)
show(j)
j = (ex3.transpose()*v)

k = i.augment(j,subdivide=True)
show(k)
show(k.echelon_form())
show(i.solve_right(j))
show(i.right_kernel(j)) # si SCD ens dona u*, si no ho és ns com
```

$$\begin{pmatrix} 14 & 20 & 26 \\ 20 & 29 & 38 \\ 26 & 38 & 50 \end{pmatrix}$$

2

$$\left(\begin{array}{ccc|c} 14 & 20 & 26 & 5 \\ 20 & 29 & 38 & 9 \\ 26 & 38 & 50 & 13 \end{array}\right)$$

$$\begin{pmatrix} 2 & 2 & 2 & -3 \\ 0 & 3 & 6 & 13 \\ 0 & 0 & 0 & 0 \end{pmatrix}$$

$$\begin{pmatrix} -\frac{35}{6} \\ \frac{13}{3} \\ 0 \end{pmatrix}$$

$$\mathrm{RowSpan}_{\mathbf{Z}} \begin{pmatrix} 1 & -2 & 1 \end{pmatrix}$$

## Injectiva o exhaustiva

In [ ]:
```
print('Es exhaustiva? ',f.is_surjective())
print('Es injectiva? ',f.is_injective())
```

## Base subespai nucli

In [25]:
```
f = linear_transformation(A[0:4].transpose())
show(f)
show(f.kernel().basis())
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
<ipython-input-25-f7af87b97bf8> in <module>
----> 1 show(f.kernel().basis())

NameError: name 'f' is not defined
```

## MIRAR lo de dalt

</font>

## Base subespai imatge

In [ ]:
```
show(f.image().basis())
```