MAT 305: Mathematical Computing

John Perry

Vectors and Vector Spaces

Matrices

How matrices can be useful

Animation and graphic design

Eigenvalues, eigenvectors

Summary

MAT 305: Mathematical Computing

Linear algebra

John Perry

University of Southern Mississippi

Spring 2017

Outline

1. Vectors and Vector Spaces
2. Matrices
3. How matrices can be useful
    - Animation and graphic design
    - Eigenvalues, eigenvectors
4. Summary

Outline

1. Vectors and Vector Spaces
2. Matrices
3. How matrices can be useful
   - Animation and graphic design
   - Eigenvalues, eigenvectors
4. Summary

Vectors

`vector(`*ring*`,` *entries*`)` where

- *ring* is base algebraic ring of *entries* (a list)
- default ring: appropriate to entries ($\mathbb{Z}$ for integers)

Vectors

`vector(`*ring*`,` *entries*`)` where

- *ring* is base algebraic ring of *entries* (a list)
- default ring: appropriate to entries ($\mathbb{Z}$ for integers)

Example

```
sage: u = vector([0, 2, 2, 0])
sage: v = vector([1, 3, -1, 2])
sage: u + v
(1, 5, 1, 2)
sage: u*v
4
sage: u.norm()
2*sqrt(2)
```

Dot product!

You can plot vectors!

`v.plot()`, with optional arguments:

- `plot_type`: `'arrow'`, `'point'`, `'step'`
- `start`: tuple, list, or vector

You can plot vectors!

`v.plot()`, with optional arguments:

- `plot_type`: `'arrow'`, `'point'`, `'step'`
- `start`: tuple, list, or vector

Example

Illustration of vector arithmetic:

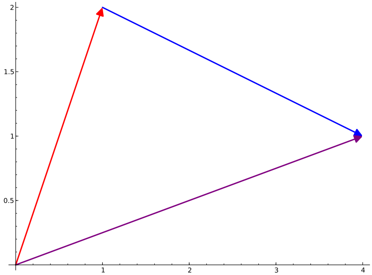
```
sage:  u = vector([1,2])
sage:  v = vector([3,-1])
sage:  u.plot(color='red')
         + v.plot(color='blue',start=u)
         + (u+v).plot(color='purple')
```

You can plot vectors!

Example

Illustration of vector arithmetic:

```
sage:  u = vector([1,2])
sage:  v = vector([3,-1])
sage:  u.plot(color='red')
         + v.plot(color='blue',start=u)
         + (u+v).plot(color='purple')
```



Outline

1. Vectors and Vector Spaces
2. Matrices
3. How matrices can be useful
   - Animation and graphic design
   - Eigenvalues, eigenvectors
4. Summary

The `matrix()` command

`matrix(` *ring*, *#rows*, *#cols*, entries`)` where

- *ring* (optional) an appropriate algebraic ring
- *#rows*, *#cols* (optional) number of rows and columns (default depends on *entries*; no *entries* $\Longrightarrow$ 0×0 matrix)
- *entries* (optional) is one of
  - a list of entries, from northwest corner to southeast (if *#rows*, *#cols* specified)
  - a list of row vectors
  - none specified? all entries `0`

Example matrices

```
sage: MZ = matrix(ZZ,3,3)
sage: MZ
[0 0 0]
[0 0 0]
[0 0 0]
```

Example matrices

```
sage:  MZ = matrix(ZZ,3,3)
sage:  MZ
[0 0 0]
[0 0 0]
[0 0 0]
sage:  MR = matrix(RR,[[1,2,3],[3,2,1],[1,1,2]])
sage:  MR
[1.00000000000000 2.00000000000000 3.00000000000000]
[3.00000000000000 2.00000000000000 1.00000000000000]
[1.00000000000000 1.00000000000000 2.00000000000000]
```

Example matrices

```
sage: MZ = matrix(ZZ,3,3)
sage: MZ
[0 0 0]
[0 0 0]
[0 0 0]
sage: MR = matrix(RR,[[1,2,3],[3,2,1],[1,1,2]])
sage: MR
[1.00000000000000 2.00000000000000 3.00000000000000]
[3.00000000000000 2.00000000000000 1.00000000000000]
[1.00000000000000 1.00000000000000 2.00000000000000]
sage: MS = matrix(SR,[[x**2 + 1, 0, 0],
      [x + I, 1, 0]])
sage: MS
[x^2 + 1       0       0]
[  x + I       1       0]
```

Help yourself read

Good idea to put rows in different lines

```
sage:  MR = matrix(RR, [
        [1,2,3],
        [3,2,1],
        [1,1,2]
      ])
sage:  MR
[1.00000000000000 2.00000000000000 3.00000000000000]
[3.00000000000000 2.00000000000000 1.00000000000000]
[1.00000000000000 1.00000000000000 2.00000000000000]
```

Accessing matrix entries

Matrix a list of lists $\Longrightarrow$ `M[i,j]` $= M_{i,j}$

Accessing matrix entries

Matrix a list of lists $\Longrightarrow$ `M[i,j]` $= M_{i,j}$

Example

```
sage: MS[1,0]
x+I
sage: MS[0,2] = x - I
sage: MS
[x^2 + 1        0        x - I]
[  x + I        1            0]
```

(counting starts from 0)

Submatrices

- `M.submatrix(`i, j, m, n`)` gives
  - $m \times n$ submatrix of M
  - whose northwest corner is in row i, column j

- `M.augment(`A`)` gives $(M|A)$

Submatrices

- `M.submatrix(`i, j, m, n`)` gives
  - $m \times n$ submatrix of M
  - whose northwest corner is in row i, column j
- `M.augment(`A`)` gives $(M|A)$

Example

```
sage:  MZ[1,1] = 1
sage:  MZ.submatrix(1,1,2,2)
[ 1 0]
[ 0 0]
```

Basic matrix operations

"dot" command	mathematics
`M.det()`	determinant
`M.inverse()`	
`M.transpose()`	
`M.eigenvalues()`	
`M.eigenvectors_right()`	right eigenvectors*
`M.eigenvectors_left()`	left eigenvectors*
`M.echelon_form()`	echelon formof unchanged M
`M.echelonize()`	change M to echelon form
`M.ncols()`	number of columns
`M.nrows()`	number of rows

*"right eigenvectors" are usual "eigenvectors"

Row arithmetic

"dot" command	mathematics
`M.set_row_to_multiple_of_row(i,j,a)`	set row i to a times row j*
`M.add_multiple_of_row(i,j,a)`	add a times row j to row i*
`M.swap_rows(i,j)`	swap rows i, j
`M.swap_columns(i,j)`	swap columns i, j

*row i changes; row j remains the same

Example: find inverse of matrix

Sage has a `.inverse()` command, but suppose you want to see steps...?

Example: find inverse of matrix

Sage has a `.inverse()` command, but suppose you want to see steps…?

Algorithm from High School Algebra II!

algorithm Compute inverse
  inputs
    M, an invertible matrix over a field
  outputs
    M^{-1}
  do
    Let $n = \dim(M)$
    Let A be augmented matrix $(M \mid I_n)$
    Triangularize A
    return rightmost $n \times n$ submatrix of A

Try it!

```
sage: MZ = matrix([[1, 2, 3, 4], [0, 2, 2, 3],
          [8, 3, 1, 2], [0, 1, 2, 3]])
```

Augment `MZ` *by* I_4

Try it!

```
sage:  MZ = matrix([[1, 2, 3, 4], [0, 2, 2, 3],
                    [8, 3, 1, 2], [0, 1, 2, 3]])
```

Augment `MZ` *by* I_4

To create I_4*, can set diagonal entries of zero matrix to 1...*

```
sage:  I4 = matrix(4,4)
sage:  for i in range(4):
         I4[i,i] = 1
sage:  I4
[1 0 0 0]
[0 1 0 0]
[0 0 1 0]
[0 0 0 1]
```

Try it!

```
sage: MZ = matrix([[1, 2, 3, 4], [0, 2, 2, 3],
                   [8, 3, 1, 2], [0, 1, 2, 3]])
```

Augment `MZ` *by* I_4

…*or use* `identity_matrix()` *command*

```
sage: I4 = identity_matrix(4)
[1 0 0 0]
[0 1 0 0]
[0 0 1 0]
[0 0 0 1]
```

Try it!

```
sage: MZ = matrix([[1, 2, 3, 4], [0, 2, 2, 3],
                   [8, 3, 1, 2], [0, 1, 2, 3]])
```

Augment `MZ` *by* I_4

…*or use* `identity_matrix()` *command*

```
sage: I4 = identity_matrix(4)
[1 0 0 0]
[0 1 0 0]
[0 0 1 0]
[0 0 0 1]
sage: A = MZ.augment(I4)
sage: A
[1 2 3 4 1 0 0 0]
[0 2 2 3 0 1 0 0]
[8 3 1 2 0 0 1 0]
[0 1 2 3 0 0 0 1]
```

Try it!

```
sage:  MZ = matrix([[1, 2, 3, 4], [0, 2, 2, 3],
                   [8, 3, 1, 2], [0, 1, 2, 3]])
```

First column: eliminate non-zero in row 3

Try it!

```
sage: MZ = matrix([[1, 2, 3, 4], [0, 2, 2, 3],
          [8, 3, 1, 2], [0, 1, 2, 3]])
```

First column: eliminate non-zero in row 3

```
sage: A.add_multiple_of_row(2,0,-8)
sage: A
[   1   2   3   4   1   0   0   0]
[   0   2   2   3   0   1   0   0]
[   0 -13 -23 -30  -8   0   1   0]
[   0   1   2   3   0   0   0   1]
```

Try it!

```
sage:  MZ = matrix([[1, 2, 3, 4], [0, 2, 2, 3],
               [8, 3, 1, 2], [0, 1, 2, 3]])
```

First column: eliminate non-zero in row 3

```
sage:  A.add_multiple_of_row(2,0,-8)
sage:  A
[   1   2   3   4   1   0   0   0]
[   0   2   2   3   0   1   0   0]
[   0 -13 -23 -30  -8   0   1   0]
[   0   1   2   3   0   0   0   1]
```

Second column: swap row w/pivot to row 2, eliminate other non-zeros

Try it!

```
sage: MZ = matrix([[1, 2, 3, 4], [0, 2, 2, 3],
          [8, 3, 1, 2], [0, 1, 2, 3]])
```

Second column: swap row w/pivot to row 2, eliminate other non-zeros

Try it!

```
sage:  MZ = matrix([[1, 2, 3, 4], [0, 2, 2, 3],
              [8, 3, 1, 2], [0, 1, 2, 3]])
```

Second column: swap row w/pivot to row 2, eliminate other non-zeros

```
sage:  A.swap_rows(1,3)
sage:  A.add_multiple_of_row(0,1,-2)
sage:  A.add_multiple_of_row(2,1,13)
sage:  A.add_multiple_of_row(3,1,-2)
sage:  A
[  1   0  -1  -2   1   0   0  -2]
[  0   1   2   3   0   0   0   1]
[  0   0   3   9  -8   0   1  13]
[  0   0  -2  -3   0   1   0  -2]
```

Try it!

```
sage: MZ = matrix([[1, 2, 3, 4], [0, 2, 2, 3],
          [8, 3, 1, 2], [0, 1, 2, 3]])
```

Second column: swap row w/pivot to row 2, eliminate other non-zeros

```
sage: A.swap_rows(1,3)
sage: A.add_multiple_of_row(0,1,-2)
sage: A.add_multiple_of_row(2,1,13)
sage: A.add_multiple_of_row(3,1,-2)
sage: A
[  1   0  -1  -2   1   0   0  -2]
[  0   1   2   3   0   0   0   1]
[  0   0   3   9  -8   0   1  13]
[  0   0  -2  -3   0   1   0  -2]
```

Third column: need pivot multiply row 3 by 1/3

Try it!

```
sage:  MZ = matrix([[1, 2, 3, 4], [0, 2, 2, 3],
                    [8, 3, 1, 2], [0, 1, 2, 3]])
```

Third column: need pivot
multiply row 3 by 1/3

Try it!

```
sage:  MZ = matrix([[1, 2, 3, 4], [0, 2, 2, 3],
                    [8, 3, 1, 2], [0, 1, 2, 3]])
```

Third column: need pivot
multiply row 3 by 1/3

```
sage:  A.set_row_to_multiple_of_row(2,2,1/3)
...
TypeError:  Multiplying row by Rational Field
element cannot be done over Integer Ring, use
change_ring or with_row_set_to_multiple_of_row
instead.
```

Try it!

```
sage: MZ = matrix([[1, 2, 3, 4], [0, 2, 2, 3],
                   [8, 3, 1, 2], [0, 1, 2, 3]])
```

Third column: need pivot
multiply row 3 by 1/3

```
sage: A.set_row_to_multiple_of_row(2,2,1/3)
...
TypeError: Multiplying row by Rational Field
element cannot be done over Integer Ring, use
change_ring or with_row_set_to_multiple_of_row
instead.
```

Uh-oh! No multiplicative inverses in default ring! ($\mathbb{Z}$)
Change to $\mathbb{Q}$ *and proceed.*

Try it!

```
sage:  MZ = matrix([[1, 2, 3, 4], [0, 2, 2, 3],
                    [8, 3, 1, 2], [0, 1, 2, 3]])
```

Uh-oh! No multiplicative inverses in default ring! ($\mathbb{Z}$)
Change to $\mathbb{Q}$ *and proceed.*

```
sage:  A = A.change_ring(QQ)
sage:  A
[   1   0  -1  -2   1   0   0  -2]
[   0   1   2   3   0   0   0   1]
[   0   0   3   9  -8   0   1  13]
[   0   0  -2  -3   0   1   0  -2]
```

Looks the same, but it's not.
Return to regularly-scheduled programming.

Try it!

```
sage:  MZ = matrix([[1, 2, 3, 4], [0, 2, 2, 3],
                    [8, 3, 1, 2], [0, 1, 2, 3]])
```

Third column: need pivot
multiply row 3 by 1/3

Try it!

```
sage:  MZ = matrix([[1, 2, 3, 4], [0, 2, 2, 3],
                    [8, 3, 1, 2], [0, 1, 2, 3]])
```

Third column: need pivot
multiply row 3 by 1/3

```
sage:  A.set_row_to_multiple_of_row(2,2,1/3)
sage:  A
[    1     0    -1    -2     1     0     0    -2]
[    0     1     2     3     0     0     0     1]
[    0     0     1     3  -8/3     0   1/3  13/3]
[    0     0    -2    -3     0     1     0    -2]
```

Try it!

```
sage:  MZ = matrix([[1, 2, 3, 4], [0, 2, 2, 3],
               [8, 3, 1, 2], [0, 1, 2, 3]])
```

Third column: need pivot
multiply row 3 by 1/3

```
sage:  A.set_row_to_multiple_of_row(2,2,1/3)
sage:  A
[    1    0   -1   -2    1    0    0   -2]
[    0    1    2    3    0    0    0    1]
[    0    0    1    3 -8/3    0  1/3 13/3]
[    0    0   -2   -3    0    1    0   -2]
```

Third column: eliminate other non-zeros

Try it!

```
sage: MZ = matrix([[1, 2, 3, 4], [0, 2, 2, 3],
           [8, 3, 1, 2], [0, 1, 2, 3]])
```

Third column: eliminate other non-zeros

Try it!

```
sage:  MZ = matrix([[1, 2, 3, 4], [0, 2, 2, 3],
                    [8, 3, 1, 2], [0, 1, 2, 3]])
```

Third column: eliminate other non-zeros

```
sage:  A.add_multiple_of_row(0,2,1)
sage:  A.add_multiple_of_row(1,2,-2)
sage:  A.add_multiple_of_row(3,2,2)
sage:  A
[     1     0     0     1  -5/3     0   1/3   7/3]
[     0     1     0    -3  16/3     0  -2/3 -23/3]
[     0     0     1     3  -8/3     0   1/3  13/3]
[     0     0     0     3 -16/3     1   2/3  20/3]
```

Try it!

```
sage:  MZ = matrix([[1, 2, 3, 4], [0, 2, 2, 3],
                    [8, 3, 1, 2], [0, 1, 2, 3]])
```

Third column: eliminate other non-zeros

```
sage:  A.add_multiple_of_row(0,2,1)
sage:  A.add_multiple_of_row(1,2,-2)
sage:  A.add_multiple_of_row(3,2,2)
sage:  A
[     1     0     0     1  -5/3     0   1/3   7/3]
[     0     1     0    -3  16/3     0  -2/3 -23/3]
[     0     0     1     3  -8/3     0   1/3  13/3]
[     0     0     0     3 -16/3     1   2/3  20/3]
```

Fourth column: need pivot
multiply row 4 by 1/3

Try it!

```
sage:  MZ = matrix([[1, 2, 3, 4], [0, 2, 2, 3],
                    [8, 3, 1, 2], [0, 1, 2, 3]])
```

Fourth column: need pivot
multiply row 4 by 1/3

Try it!

```
sage:  MZ = matrix([[1, 2, 3, 4], [0, 2, 2, 3],
              [8, 3, 1, 2], [0, 1, 2, 3]])
```

Fourth column: need pivot
multiply row 4 by 1/3

```
sage:  A.set_row_to_multiple_of_row(3,3,1/3)
sage:  A
[     1      0      0      1  -5/3      0    1/3    7/3]
[     0      1      0     -3  16/3      0   -2/3  -23/3]
[     0      0      1      3  -8/3      0    1/3   13/3]
[     0      0      0      1 -16/9    1/3    2/9   20/9]
```

Try it!

```
sage:  MZ = matrix([[1, 2, 3, 4], [0, 2, 2, 3],
              [8, 3, 1, 2], [0, 1, 2, 3]])
```

Fourth column: need pivot
multiply row 4 by 1/3

```
sage:  A.set_row_to_multiple_of_row(3,3,1/3)
sage:  A
[     1      0      0      1  -5/3      0    1/3    7/3]
[     0      1      0     -3  16/3      0   -2/3  -23/3]
[     0      0      1      3  -8/3      0    1/3   13/3]
[     0      0      0      1 -16/9    1/3    2/9   20/9]
```

Fourth column: eliminate other non-zeros

Try it!

```
sage:  MZ = matrix([[1, 2, 3, 4], [0, 2, 2, 3],
                    [8, 3, 1, 2], [0, 1, 2, 3]])
```

Fourth column: eliminate other non-zeros

Try it!

```
sage: MZ = matrix([[1, 2, 3, 4], [0, 2, 2, 3],
           [8, 3, 1, 2], [0, 1, 2, 3]])
```

Fourth column: eliminate other non-zeros

```
sage: A.add_multiple_of_row(0,3,-1)
sage: A.add_multiple_of_row(1,3,3)
sage: A.add_multiple_of_row(2,3,-3)
sage: A
[    1     0     0     0   1/9  -1/3   1/9   1/9]
[    0     1     0     0  16/3     1     0    -1]
[    0     0     1     0  -8/3    -1  -1/3  -7/3]
[    0     0     0     1 -16/9   1/3   2/9  20/9]
```

Try it!

```
sage:  MZ = matrix([[1, 2, 3, 4], [0, 2, 2, 3],
                  [8, 3, 1, 2], [0, 1, 2, 3]])
```

Fourth column: eliminate other non-zeros

```
sage:  A.add_multiple_of_row(0,3,-1)
sage:  A.add_multiple_of_row(1,3,3)
sage:  A.add_multiple_of_row(2,3,-3)
sage:  A
[     1      0      0      0    1/9   -1/3    1/9    1/9]
[     0      1      0      0   16/3      1      0     -1]
[     0      0      1      0   -8/3     -1   -1/3   -7/3]
[     0      0      0      1  -16/9    1/3    2/9   20/9]
```

Have inverse! extract, test

Try it!

```
sage: MZ = matrix([[1, 2, 3, 4], [0, 2, 2, 3],
                   [8, 3, 1, 2], [0, 1, 2, 3]])
```

Have inverse! extract, test

Try it!

```
sage:  MZ = matrix([[1, 2, 3, 4], [0, 2, 2, 3],
                    [8, 3, 1, 2], [0, 1, 2, 3]])
```

Have inverse! extract, test

```
sage:  Minv = A.submatrix(0,4,4,4)
sage:  Minv * M
[1 0 0 0]
[0 1 0 0]
[0 0 1 0]
[0 0 0 1]
```

Other tools

Need another computation w/M? Remember:

- `M.<tab>` states all tools for M
- `M.<command>?` states help for `command`
- `M.<command>??` lists source code for `command`

Outline

1 Vectors and Vector Spaces

2 Matrices

3 How matrices can be useful
 - Animation and graphic design
 - Eigenvalues, eigenvectors

4 Summary

Outline

1. Vectors and Vector Spaces
2. Matrices
3. How matrices can be useful
   - Animation and graphic design
   - Eigenvalues, eigenvectors
4. Summary

Manipulating graphics

Can manipulate point (x, y) using matrix arithmetic:

- let $\mathbf{v} = (x, y)$ be vector
- let M be matrix of special form
- $M\mathbf{v}$ gives new point

Useful matrix forms

type	scaling	rotation	reflection
form	$\begin{pmatrix} s & 0 \\ 0 & s \end{pmatrix}$	$\begin{pmatrix} \cos\alpha & \sin\alpha \\ -\sin\alpha & \cos\alpha \end{pmatrix}$	$\begin{pmatrix} \cos\beta & \sin\beta \\ \sin\beta & -\cos\beta \end{pmatrix}$
effect	rescales (x,y) to (sx,sy)	rotates (x,y) through angle α	reflects (x,y) across line w/slope ${}^{1-\cos\beta}/_{\sin\beta}$

Example; rotate a polygon

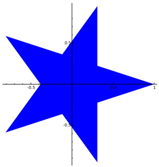
Let's rotate an off-kilter star

```
sage: V = [
          vector((1,0)),
          vector((cos(4*pi/5),sin(4*pi/5))),
          vector((cos(8*pi/5),sin(8*pi/5))),
          vector((cos(2*pi/5),sin(2*pi/5))),
          vector((cos(6*pi/5),sin(6*pi/5)))
        ]
sage: polygon(U)
```

Example; rotate a polygon

Let's rotate an off-kilter star

```
sage: V = [
           vector((1,0)),
           vector((cos(4*pi/5),sin(4*pi/5))),
           vector((cos(8*pi/5),sin(8*pi/5))),
           vector((cos(2*pi/5),sin(2*pi/5))),
           vector((cos(6*pi/5),sin(6*pi/5)))
          ]
sage: polygon(U)
```



A little off… $\pi/10$ maybe?

Use a rotation matrix

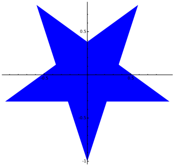
- try $\pi/10$
- easier w/comprehension

```
sage:  M = matrix((
                 (cos(pi/10), sin(pi/10)),
                 (-sin(pi/10), cos(pi/10))
               ))
sage:  U = [M*v for v in V]
sage:  polygon(U)
```

Use a rotation matrix

- try $\pi/10$
- easier w/comprehension

```
sage:  M = matrix((
              (cos(pi/10), sin(pi/10)),
              (-sin(pi/10), cos(pi/10))
            ))
sage:  U = [M*v for v in V]
sage:  polygon(U)
```



Oops! upside-down!

Rotate the *other* way

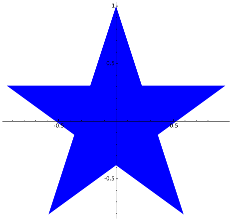
- M rotated clockwise
- try $-\pi/10$

```
sage: M = matrix((
                (cos(-pi/10), sin(-pi/10)),
                (-sin(-pi/10), cos(-pi/10))
              ))
sage: U = [M*v for v in V]
sage: polygon(U)
```

Rotate the *other* way

- M rotated clockwise
- try- $-\pi/10$

```
sage:  M = matrix((
                 (cos(-pi/10), sin(-pi/10)),
                 (-sin(-pi/10), cos(-pi/10))
               ))
sage:  U = [M*v for v in V]
sage:  polygon(U)
```



Outline

1. Vectors and Vector Spaces
2. Matrices
3. How matrices can be useful
   - Animation and graphic design
   - Eigenvalues, eigenvectors
4. Summary

Eigenvectors and eigenvalues

An **eigenvector** $\mathbf{x}$ of a matrix M with **eigenvalue** λ satisfies

$$M\mathbf{x} = \lambda\mathbf{x}$$

Eigenvectors and eigenvalues

An **eigenvector x** of a matrix M with **eigenvalue** λ satisfies

$$M\mathbf{x} = \lambda\mathbf{x}$$

Example

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

verification in Sage:

```
sage:  M = matrix(2,2,[0,2,2,0])
sage:  v = vector([1,-1])
sage:  M*v
(-2, 2)
```

Easy to find in Sage

```
sage:  M = matrix(2,2,[0,2,2,0])
sage:  M.eigenvectors_left()
[(2, [ (1, 1) ], 1), (-2, [ (1, -1) ], 1)]
```

What does this tell us?

- $\mathbf{e}_1 = (1,1)$ is eigenvec w/eigenval 2, mult 1
- $\mathbf{e}_2 = (1,-1)$ is eigenvec, w/eigenval -2, mult 1

Easy to find in Sage

```
sage:  M = matrix(2,2,[0,2,2,0])
sage:  M.eigenvectors_left()
[(2, [ (1, 1) ], 1), (-2, [ (1, -1) ], 1)]
```

What does this tell us?

- $\mathbf{e}_1 = (1,1)$ is eigenvec w/eigenval 2, mult 1
- $\mathbf{e}_2 = (1,-1)$ is eigenvec, w/eigenval -2, mult 1

In other words,

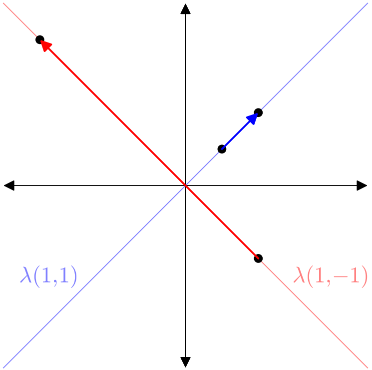
- $M\mathbf{e}_1 = 2\mathbf{e}_1$
- $M\mathbf{e}_2 = -2\mathbf{e}_2$

Verify in Sage

Geometric interpretation

$M\mathbf{x} = \lambda\mathbf{x}$

- $\lambda\mathbf{x}$ on same *line through origin* as $\mathbf{x}$
  - $\lambda > 0$? same direction
  - $\lambda < 0$? opposite direction
- $\lambda\mathbf{x}$ different *size* from $\mathbf{x}$



Outline

1. Vectors and Vector Spaces
2. Matrices
3. How matrices can be useful
   - Animation and graphic design
   - Eigenvalues, eigenvectors
4. Summary

Summary

- Sage does matrices
  - over fields *and* rings
  - symbolic ring! explore!
  - can change base ring
- You can solve some sophisticated problems using matrices on Sage