Sage and Linear Algebra Worksheet

FCLA Section SD

Robert Beezer
Department of Mathematics and Computer Science
University of Puget Sound

Spring 2021

1 Similarity

We manufacture two matrices that are similar, and use Sage to check. A "unimodular" matrix is one with determinant 1. A unimodular matrix with integer entries will have an inverse with integer entries (that is a theorem, and Exercise PDM.M20).

```
A = random_matrix(ZZ, 10, x = -9, y = 9).change_ring(QQ)
S = random_matrix(QQ, 10, algorithm='unimodular',
    upper_bound=9)
B = S.inverse()*A*S
A, B
```

This next command might be broken, and might even just hang. My fault. It will be fixed, using rational canonical form, for Sage 7.6. See Trac ticket #18505 for the details.

```
A.is_similar(B)
```

2 Diagonalization

These two matrices are from the earlier demo for Section EE. First is diagonalizable, second is not. The easiest way to see the difference is with the `eigenmatrix` commands.

Demonstration 1 Diagonalize the matrix A.

```
A = matrix(QQ, [
    [-31, -23, -16,  12, 120, -17],
    [ -3,   7,   0, -12,  60, -21],
    [-28, -14,  -9,  -4, 152, -30],
    [-36, -20, -16,  -1, 192, -32],
    [ -9,  -5,  -4,   0,  47,  -8],
    [ -1,   1,   0,  -4,  20,  -3]
    ])
A
```

`S`, the matrix whose columns are eigenvectors, will "diagonalize" `A`.

```
D, S = A.eigenmatrix_right()
D, S
```

```
S.inverse()*A*S == D
```

Here is an equivalent formulation.

```
A*S == S*D
```

Demonstration 2 Now, in contrast, a matrix that is not diagonalizable. Try to diagonalize the matrix C.

```
C = matrix(QQ, [
    [128,   20,  44, -50,  236, -18, -330, -565],
    [ -23, -16,  -5,   6,  -40,  27,   62,  128],
    [ -44, -12, -15,  16,  -78,  20,  110,  207],
    [ -2,   10,  -4,   3,  -10, -23,   20,   -9],
    [ -61,   5, -25,  27, -116, -26,  153,  225],
    [ -12, -12,  -1,   2,  -20,  24,   34,   82],
    [ -23,  -3,  -8,   9,  -42,   2,   57,   99],
    [  13,   6,   3,  -4,   23, -12,  -35,  -72]
    ])
C
```

```
D, S = C.eigenmatrix_right()
D, S
```

The zero columns in `S` tell us that at least one eigenvalue has a geometric multiplicity strictly less than the algebraic multiplicity of the eigenvalue. So by Theorem DMFE the matrix `C` is not diagonalizable.

A second consequence of the zero columns of `S` is that it will not be an invertible matrix. But the output from Sage still obeys a fundamental relationship.

```
C*S == S*D
```

Perhaps simpler is the built-in function `.is_diagonalizable()`.

```
A.is_diagonalizable()
```

```
C.is_diagonalizable()
```

3 Block Diagonal

This section uses results about generalized eigenspaces, a topic from Section IS, which is only available in the Late Determinants version. We continue with the matrix `C` from above.

We compute a generalized eigenspace for each of the two eigenvalues, by using $n = 8$ as the power of $C - \lambda I$ in the null space computation, according to Theorem GNES. The choice of a pivot basis provides slightly cleaner results later, but Sage's default basis would work equally well.

```
G1 = ((C-1)^8).right_kernel(basis='pivot')
G1.dimension()
```

```
G2 = ((C+2)^8).right_kernel(basis='pivot')
G2.dimension()
```

Notice that the dimensions of the two generalized eigenspaces sum to $n = 8$. Even better, the union of the two bases is a linearly independent set, and hence a basis of $\mathbf{C}^8$. None of this is an accident, and is the content of the upcoming Theorem GEB.

We grab a basis of each generalized eigenspace, form their union (a "sum" of lists in Python syntax), and make a (nonsingular) matrix with the basis vectors as columns.

```
B1 = G1.basis()
B2 = G2.basis()
S = column_matrix(B1+B2)
S
```

We form a similar matrix, which has a **block diagonal** form, indicated by the subdivisions we have added. The blocks are a direct consequence, and manifestation, of the fact that the generalized eigenspaces are invariant subspaces of C.

```
BD = S.inverse()*C*S
BD.subdivide([3],[3])
BD
```

4 Nearly Diagonalizable

A matrix that is not diagonalizable will always be similar to a matrix that is *almost* diagonalizable. The "nearly diagonal" matrix is called the **Jordan canonical form** of the matrix.

Demonstration 3 While beyond the scope of this course, use Sage to compute the Jordan canonical form for the matrix `C`. Notice the eigenvalues of `C` on the diagonal and the 1's on the **super-diagonal**.

Peculiarly, the similarity matrix need not be computed to get the form, and it is a significant computational expense. So we ask for it explicitly.

```
J, T = C.jordan_form(transformation=True)
J, T
```

The transformation matrix, `T`, is invertible and will "almost diagonalize" `C`.

```
T.inverse()*C*T == J
```

Demonstration 4 Rational canonical form is another interesting version of this idea, try `.rational_form()` on `C`. And if you do, then execute the following two cells and see if the coefficients look familiar. Learn more about **companion matrices** if this makes you curious.

```
C.rational_form()
```

```
C.fcp()
```

```
((x-1)^3*(x+2)^3).expand()
```