Computer Project 6
Eigenvalues, Eigenvectors, and Diagonalization of Matrices in Python

DUE: January 25, 2023

Introduction: Python is capable of computing most quantities of interest for matrices: including determinants, eigenvalues, and eigenvectors. Python can also give us the Jordan Canonical Form of a matrix (which includes the special case of diagonalizing a matrix when that is possible). This computer project will introduce you to the syntax needed to work with matrices in Python. You will need to add the following import statement to the beginning of your code.

```
from sympy import Matrix
```

To input a matrix, we use the `Matrix` function giving it a list of the rows (themselves being lists of entries). As an example, we can enter the matrix

$$M = \begin{bmatrix} 1 & 2 & 3 \\ 4 & 3 & 2 \\ 1 & 0 & -1 \end{bmatrix}$$

into Python using the following code.

```
M = Matrix([[1,2,3],[4,3,2],[1,0,-1]])
```

Once the matrix has been declared, we can work with the matrix using a number of methods from the `sympy` package.

- `M.det()` will compute the determinant of the matrix.
- `M.eigenvals()` will compute the eigenvalues of `M`. The return type is a dictionary of entries with items of the form `eigenvalue: algebraic_multiplicity`.
- `M.eigenvects()` will compute both the eigenvalues and corresponding eigenvectors of `M`. The return type is a list of tuples for each eigenvalue. Each tuple has the form

  `(eigenvalue, algebraic_multiplicity, list_of_eigenvalues)`.

  Notice that the list may contain *fewer* eigenvectors than the algebraic multiplicity (if the matrix has a deficiency).

- `B, D = M.diagonalize()` returns a pair of matrices if `M` is diagonalizable. `B` is the invertible matrix whose columns consist of eigenvectors of `M` while `D` is the corresponding diagonal matrix similar to `M`: `M = B D B`$^{-1}$.
- `B, J = M.jordan_form()` returns a pair of matrices. `J` is the Jordan matrix similar to `M` and `B` is the invertible matrix whose columns consist of the corresponding generalized eigenvectors: `M = B J B`$^{-1}$.

If we were working with more than one matrix, the standard matrix operations are defined as you would expect.

- Addition: `M + N` (assuming the matrices are the same shape)
- Scalar Multiplication: `2*M`
- Matrix Multiplication: `M*N` (assuming dimensions match)
- Matrix Powers: `M**2` (assuming a square matrix)
- Inverses: `M**(-1)` (assuming the matrix is invertible)

More information about matrix manipulation in `sympy` is available at the following URL.

https://docs.sympy.org/latest/tutorials/intro-tutorial/matrices.html#

Instructions: For the following matrices, use Python to compute the determinant, eigenvalues, and corresponding eigenvectors. If a matrix is diagonalizable, explicitly state that fact and find the change of basis matrix B and diagonal matrix D that satisfies $M = BDB^{-1}$. If M is not diagonalizable, find the Jordan matrix J and change of basis matrix B so that $M = BJB^{-1}$.

1) $M_1 = \begin{bmatrix} 3 & -1 & 1 \\ -1 & 5 & -1 \\ 1 & -1 & 3 \end{bmatrix}$

2) $M_2 = \begin{bmatrix} 9 & 0 & -6 \\ 12 & 3 & -12 \\ 12 & 0 & -9 \end{bmatrix}$

3) $M_3 = \begin{bmatrix} -6 & 4 & 3 & 4 \\ -7 & 5 & 3 & 4 \\ -8 & 5 & 4 & 5 \\ 3 & -2 & -1 & -1 \end{bmatrix}$

4) $M_4 = \begin{bmatrix} 2 & 2 & 4 & 3 & 1 \\ -2 & 7 & 2 & 0 & 2 \\ 2 & 2 & 7 & 0 & -2 \\ -2 & -2 & 2 & 9 & 2 \\ -5 & 4 & 2 & 3 & 8 \end{bmatrix}$