Max Kramer

I affirm that I have adhered to the honor code on this assignment

Hello again, scientist! I'll write in italics, and problems for you will always be in ***bold****. As a general rule, I expect you to do at least as much writing as I do. Code should be part of your solution, but I expect variables to be clear and explanation to involve complete sentences. Cite your sources; if you work with someone in the class on a problem, that's an extremely important source.*

Problem 11.1.

MATLAB is fantastic at computing eigenvectors, but it's good to do it by hand... once. Here's a matrix:

```
B = [2 -1 -1; 1 4 1; 1 1 4]
syms lambda;
```

```
B =

     2    -1    -1
     1     4     1
     1     1     4
```

Using the command det(), find the characteristic polynomial of B. *There's a charpoly() command; please don't use it on this one.*

```
char_poly = det(B - lambda * eye(3))
```

```
char_poly =

- lambda^3 + 10*lambda^2 - 33*lambda + 36
```

The above command calculates det(B - lambda * I). The scalar equation det(B - lambda * I) = 0 is the characteristic polynomial of B.

Using the command factor(), find the eigenvalues of B.

```
factor(char_poly)
```

```
ans =

[ -1, lambda - 4, lambda - 3, lambda - 3]
```

Your results should match this one.

```
eig(B)
```

```
ans =

    3.0000
    4.0000
    3.0000
```

Without doing any more calculations, **is invertible? Explain.** *(Is 0 an eigenvalue? Why is that relevant?)*

The invertible matrix theorem states that an n x n matrix for which 0 is NOT an eigenvalue is invertible. As 0 is NOT an eigenvalue of the matrix, A is invertible.

Without doing any more calculations, ***what are the possible dimensions of the 4-eigenspace?***

As the eigenvalue 4 has multiplicity 1, the dimension of the 4-eigenspace must be less than or equal to 1.

Find a basis for the 4-eigenspace. *Here's a start:*

```
K = B-4*eye(3);
sys = horzcat(K,[0;0;0]);
rref(sys)
```

```
ans =

     1     0     1     0
     0     1    -1     0
     0     0     0     0
```

The general solution to the above problem is [-1;1;1] which is an eigenvector corresponding to the eigenvalue 4.

Check that the vector(s) you found in the previous part are actually eigenvectors with eigenvalue 4 by multiplying them by B.

```
eigenv = [-1;1;1];
B * eigenv
```

```
ans =

    -4
     4
     4
```

The above computation demonstrates that eigenv multiplied by B is 4 * eigenv, so v is an eigenvector corresponding to eigenvalue 4.

Without doing any more calculations, what are the possible dimensions of the 3-eigenspace?

As the eigenvalue 3 has multiplicity 2, the dimension of the 3-eigenspace must be less than or equal to 2.

Find a basis for the 3-eigenspace. Does R^3 have a basis of eigenvectors for B?

```
L = B-3*eye(3);
```

```
sys2 = horzcat(L,[0;0;0]);
rref(sys2)

eigenv1 = [-1;0;1];
eigenv2 = [-1;1;0];

B * eigenv1
B * eigenv2


ans =

     1     1     1     0
     0     0     0     0
     0     0     0     0


ans =

    -3
     0
     3


ans =

    -3
     3
     0
```

The above computation demonstrates that the equation (A-3I) = 0 has a general solution with the vectors [-1;0;1] and [-1;1;0]. These vectors multiplied by B produce 3 times the vectors. This demonstrates that these vectors are eigenvectors corresponding to an eigenvalue of 3. There are three linearly independent vectors betwen the eigenvectors for the 3-eigenspace and 4-eigenspace, so R^3 does have a basis of eigenvectors for B.

The eig() command is weird. It's overloaded, meaning it does different things depending on how you call it. Check it out.

```
[V,D] = eig(B)


V =

   -0.8165   -0.5774   -0.1447
    0.4082    0.5774   -0.6236
    0.4082    0.5774    0.7682


D =

    3.0000         0         0
         0    4.0000         0
         0         0    3.0000
```

If you did everything right, the 4-eigenspace should be pretty clearly the same as the one you found, but the 3-eigenspace will look a bit odd. That's because MATLAB uses a greedy algorithm to compute eigenvectors; the first one it finds is usually pretty clear, but they tend to get weirder.

Published with MATLAB® R2019b