Tutorial 4: Eigenvalues and diagonalization

In this Tutorial you will learn how to compute eigenvalues and eigenvectors in Matlab and how to diagonalize a symmetric matrix.

Computing eigenvalues and eigenvectors

You can use eig to compute the eigenvalues and eigenvectors of a matrix. Use help to find out about the eig command.

```
help eig
```

Now you are ready to use the command with a matrix **A**

```
A = [2 -1  0
    -1  2 -1
     0  -1  2];
[V, L]=eig(A);
```

The 3x3 matrix **V** contains the three eigenvectors of **A** as its columns. The diagonal 3x3 matrix **L** contains the three eigenvalues of **A** on its diagonal.

Recall that eigenvalues and eigenvectors satisfy:

$$\mathbf{Av} = \lambda\mathbf{v}$$

You can check these relations by computing $\mathbf{Av} - \lambda\mathbf{v}$, which should be zero:

```
A*V(:,1)-L(1,1)*V(:,1)
```

Here, V(:,1) is the first eigenvector of **A** (first column of **V**), L(1,1) is the first eigenvalue of A (first diagonal element of **L**). Note that we see again the familiar numerical error, i.e., $\mathbf{Av} - \lambda\mathbf{v}$ is not exactly zero, but very nearly zero.

You can check that all three eigenvalues/eigenvectors satisfy this relation:

```
A*V(:,1)-L(1,1)*V(:,1)
A*V(:,2)-L(2,2)*V(:,2)
A*V(:,3)-L(3,3)*V(:,3)
```

You can also write all three relations in matrix form:

$$\mathbf{AV} = \mathbf{VL}$$

Check this numerically:

```
A*V-V*L
```

Diagonalization

To transpose a matrix, use the '. For example, to get the transpose of A, simply write:

```
A'
```

If you multiply **V** and its transpose, you get the identity matrix:

```
V'*V
V*V'
```

Note that, suprisingly, here $\mathbf{V}^T\mathbf{V} = \mathbf{V}\mathbf{V}^T$. Thus, the inverse of **V** is its transpose, $\mathbf{V}^{-1} = \mathbf{V}^T$. Multiplying $\mathbf{AV} = \mathbf{VL}$ from the left by $\mathbf{V}^T$ thus gives

$$\mathbf{V}^T\mathbf{AV} = \mathbf{L}$$

Note that A is not diagonal, but after multiplying from the left and right by $\mathbf{V}^T$ and **V**, we obtain the diagonal matrix **L**. In this sense, the matrix **V**, diagonalizes the matrix **A**.

You can diagonalize in Matlab as follows:

```
V'*A*V
```

Normalization of eigenvectors

Recall from class that eigenvectors are only known up to an arbitrary multiplicative constant. That is, if **v** is an eigenvector, the $\alpha\mathbf{v}$ is also an eigenvector for any non-zero scalar α. You can thus create other eigenvectors of **A**:

```
b1 = 5*V(:,1);
b2 = -0.2*V(:,2);
b3 = 10*V(:,3);
B = [b1 b2 b3]; % the 3x3 matrix B has the three vectors b1, b2 and b3 as its columns
```

You can check that b1, b2 and b3 are indeed eigenvectors of **A**:

```
A*b1-L(1,1)*b1
A*b2-L(2,2)*b2
A*b3-L(3,3)*b3
```

Or in matrix form:

```
A*B-B*L
```

The inverse of **B**, however, is no longer its transpose:

```
B'*B
B*B'
```

But since $\mathbf{B}^T\mathbf{B}$ is still a diagonal matrix, **B** also diagonalizes **A**:

```
B'*A*B
```

But note that $\mathbf{B}^T\mathbf{AB}$ is no longer equal to **L**. It is equal to $\mathbf{B}^T\mathbf{BL}$

```
B'*B*L
```

Since eigenvectors are only defined up to a multiplicative constant, it is common practice to "normalize" them so that $\mathbf{v}^T\mathbf{v} = 1$. This is what Matlab's eig command automatically does.

You can get the eigenvectors $\mathbf{v}$ that you computed using Matlab's back from the eigenvectors b1, b2, b3 by normalizing the eigenvectors such that $\mathbf{b}^T\mathbf{b} = 1$. You can achieve this by dividing $\mathbf{b}$ by $\sqrt{\mathbf{b}^T\mathbf{b}}$:

```
b1 = b1/sqrt(b1'*b1);
b1'*b1

b2 = b2/sqrt(b2'*b2);
b2'*b2

b3 = b3/sqrt(b3'*b3);
b3'*b3
```

After this last step, the eigenvectors b1, b2, b3 are exactly the same as the eigenvectors contained in the matrix $\mathbf{V}$. There is, however, one more issue: the sign of and eigenvector is still not uniquely defined, even when one enforces that $\mathbf{v}^T\mathbf{v} = 1$. Define an eigenvector $\mathbf{u} = -\mathbf{v}$.

```
u=-V(:,1);
```

Then $\mathbf{u}^T\mathbf{u} = 1$ and $\mathbf{v}^T\mathbf{v} = 1$, but the vectors are clearly not the same.

```
u'*u
V(:,1)'*V(:,1)
u-V(:,1)
```

Nonetheless, none of these non-uniqueness issues are critical to diagonalization. Do you know why?

Exercises

1. Why is $\mathbf{B}^T\mathbf{BL}$ (see above for definitions) a diagonal matrix?

2 The determinant of a matrix **A** is equal to the product of the eigenvalues. Write a code that uses only the eig command and basic operations (such as multiplication and addition) to compute the determinant of the matrix

```
A= [ 2 -1  0
```

```
    -1  2 -1
     0 -1  2];
```

You can check your answer by using Matlab's det command. As usual, find out how to use the det command by using the help command.

3. Compute the eigenvalues of the 1000x1000 matrix

$$\mathbf{A} = \begin{pmatrix} 2 & -1 & 0 & \cdots & & 0 \\ -1 & 2 & -1 & 0 & \cdots & 0 \\ 0 & -1 & 2 & -1 & 0 & \vdots \\ & \ddots & \ddots & \ddots & \ddots & \ddots \\ & & 0 & -1 & 2 & -1 \\ & & & 0 & -1 & 2 \end{pmatrix}$$

You can use this code to create **A**

```
clear % clear old variables
n = 100;
A = 2*eye(n,n)-diag(ones(n-1,1),1)-diag(ones(n-1,1),-1);
```

You can visualize such a large matrix by plotting the values of the elements as colors:

```
figure
imagesc(A)
```

Use the eig command to compute the eigenvalues and eigenvectors of **A**. Make a plot of the eigenvalues by plotting the diagonal elements of the matrix **L** that you obtain fromt the eig command and that contains the eigenvalues of **A** on the diagonal. You can do this by using this command:

```
figure
plot(diag(L))
```

Diagonalize the matrix **A**. Make a plot of the diagonalized **A**.