Eigenvalues and eigenvectors using Python

Eigenvalues and eigenvectors

Eigenvalues

Consider the matrix

$$\mathbf{A} = \begin{bmatrix} 2 & 4 \\ 1 & 5 \end{bmatrix}$$

We input this into Python:

```
A = Matrix([[2,4],[1,5]])
```

Eigenvalues are calculated using

```
A.eigenvals()
```

This returns pairs of information: `eigenvalue: algebraic multiplicity`. The first value is the eigenvalue. The second is the algebraic multiplicity, which you don't need to worry about provided it is 1.

For example, our matrix A returns

```
{1: 1, 6: 1}
```

so the eigenvalues are 1 and 6 (the first value of each pair).

You can get just the eigenvalues in a way you can access using the more complicated command:

```
eig = [A.eigenvects()[i][0] for i in range(len(A.eigenvects()))]
```

Now you can access the eigenvalues as `eig[0]` and `eig[1]` (and `eig[2]` and so on if there were more of them).

Eigenvectors

To get the eigenvectors, use

```
A.eigenvects()
```

Here the output is more complicated. In each set of round brackets there are three items: `eigenvalue, algebraic multiplicity, eigenvector`.

You can access just the eigenvectors of A using a more complicated command:

```
eigv = [v[2][0] for v in A.eigenvects()]
```

Now you can access the eigenvectors as `eigv[0]` and `eigv[1]` (and `eigv[2]` and so on if there were more of them).

Diagonalisation

To diagonalise the matrix, use `diagonalize` (note the American spelling). This code sets `U` to be the matrix of eigenvectors and `D` to be the diagonal matrix of eigenvalues.

```
U,D = A.diagonalize()
```

You can check the diagonliation has worked by calculating UDU^{-1} and seeing if you get back the original matrix.

```
U*D*U**-1
```