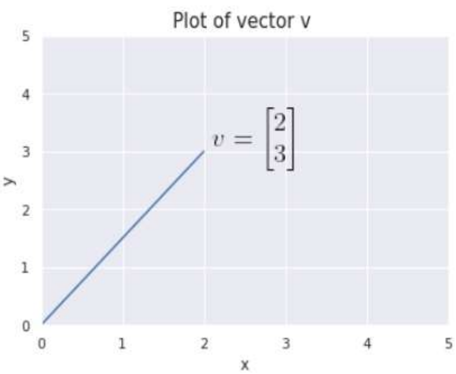
Eigenvectors and Eigenvalues

Consider that we have a vector **V** and a matrix **A**.

If we multiply matrix **A** with vector **V**, we obtain a new, transformed vector.

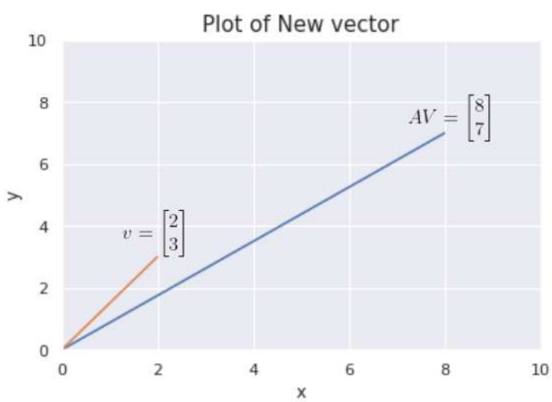


$$A = \begin{bmatrix} 1 & 2 \\ 2 & 1 \end{bmatrix}$$

$$AV = \begin{bmatrix} 1 & 2 \\ 2 & 1 \end{bmatrix} \begin{bmatrix} 2 \\ 3 \end{bmatrix}$$

$$= \begin{bmatrix} 8 \\ 7 \end{bmatrix}$$

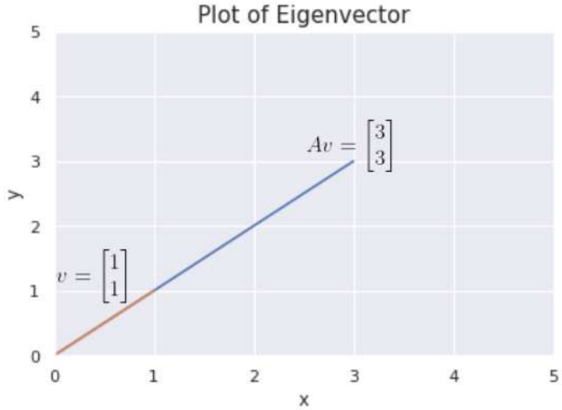
The below figure shows that multiplying by the matrix **A** has scaled the vector **V** to a new vector, with a different magnitude and a slightly different direction.



As we know, vectors have both a magnitude and a direction. The new vector **AV** seems to have a different direction as well as magnitude in comparison to the old vector **V**.

In linear algebra, the operation above is known as a **linear transformation.** It is not only restricted to scaling - linear transformations can be used for flipping, rotating, shearing and other mathematical operations.

Let’s now consider a different vector **V** and multiply it with the same matrix **A**:



$$A = \begin{bmatrix}1 & 2\\2 & 1\end{bmatrix} \quad v = \begin{bmatrix}1\\1\end{bmatrix}$$

$$Av = \begin{bmatrix}1 & 2\\2 & 1\end{bmatrix}\begin{bmatrix}1\\1\end{bmatrix}$$

$$Av = \begin{bmatrix}3\\3\end{bmatrix} = 3\begin{bmatrix}1\\1\end{bmatrix}$$

In this example, we notice that something different has happened.

We see that in this instance, while the magnitude of the new vector is certainly different, **its direction has not changed.**

These special vectors are known in linear algebra as **Eigenvectors**.

As illustrated in the example above, given the corresponding matrix A, eigenvectors are directionally-invariant when multiplied with that matrix i.e. they don’t change their direction on multiplication with matrix A **-** they merely get scaled in terms of magnitude. The value with which the eigenvector gets scaled, is known as the **Eigenvalue,** denoted by the symbol lambda.

In our example, the eigenvector is $\begin{pmatrix}1\\1\end{pmatrix}$ and the eigenvalue is 3.

So to summarize, the eigenvector of a matrix is a vector whose direction does not change when a linear transformation (matrix multiplication) is performed on it.

Mathematically, the equation is represented as:

$$Av = \lambda v$$

- **A -** Transformation matrix
- **v -** Eigenvector
- λ **-** Eigenvalue

Taking λv to the left side:

$$Av - \lambda v = 0$$

Lambda is a scalar, so taking **v** out in common:

$$(A - \lambda I)v = 0$$

An eigenvector is a non-zero vector. So, **v** cannot be zero.

Hence, to satisfy the right-hand side condition, $(A - \lambda I)$ needs to be zero.

If we were to multiply by the inverse of the matrix on both sides:

$$(A - \lambda I)^{-1}(A - \lambda I)v = (A - \lambda I)^{-1} \times 0$$

Since the product of a matrix and its inverse is I (the identity matrix), equating the left and right sides, we get:

$$v = 0$$

This is contradictory, as we have mentioned above that the Eigenvector cannot be zero.

So that means we cannot actually use the inverse of $(A - \lambda I)$, as it is not an invertible matrix.

In linear algebra, **if a matrix is not invertible then the determinant of the matrix is equal to zero.**

That means, we only need to solve the equation $det(A - \lambda I) = 0$, to get the eigenvalues and through them, the eigenvectors.

Due to Numpy in Python, we do do not need to perform these operations by hand. Numpy has functions to find the eigenvalues and eigenvectors of a matrix for us.

Why are Eigenvectors important?

The directional invariance of Eigenvectors turns out to be incredibly important, and is utilized by many applications - Principal Component Analysis (PCA), for example, is a highly popular data projection technique that can be used to reduce the dimensionality of a dataset and visualize it in lower dimensions.

This technique will be discussed in the first lecture of the week.

To understand the concept of Eigenvectors and Eigenvalues in some more detail, check out this video from 3 Blue 1 Brown:

Eigenvectors and Eigenvalues