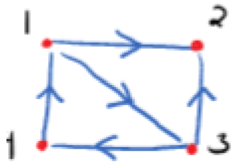
Powers of matrices

Imagine traversing a graph and wanting to find all paths of length two.

We can represent the graph with a matrix:



We can write this out as

$$\begin{bmatrix} 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 \\ 1 & 0 & 0 & 0 \end{bmatrix}$$

Which represents connections between each node and the nodes around it. We can manually count the number of steps, or we can multiply A by itself. Woah! Why does this work? Each column $\times$ each row is asking "how many paths exist when we do this thing". A^2 is asking for the length 2 paths.

For three step paths, we compute
A^3

$$= \begin{bmatrix} 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 \\ 1 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 \\ 1 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 \\ 1 & 0 & 0 & 0 \end{bmatrix}$$

This scales to be annoying really fast to do by hand.

But we can apply this in the real world. IE. how many ways are there to traverse a city efficiently?

$A^3 =$

$$\begin{bmatrix} 1 & 4 & 4 & 1 \\ 4 & 2 & 4 & 4 \\ 4 & 0 & 2 & 4 \\ 1 & 4 & 4 & 1 \end{bmatrix}$$

lets look at the numbers corresponding to

$A \to D, C \to B, \text{ and } C \to C$
They match the number of possible paths starting and ending in those places!

What is a nice way to compute this? Hint: We can use eigenvectors to simplify this.

We can take any vector $\vec{v}$ and represent it as a linear combination of those eigenvectors. We can now rewrite taking powers to be far more computationally efficient.

$A^n\vec{v}$ is hard, but

$$a\lambda_1^n\vec{w} + b\lambda_2\vec{w}_2$$

is easy!