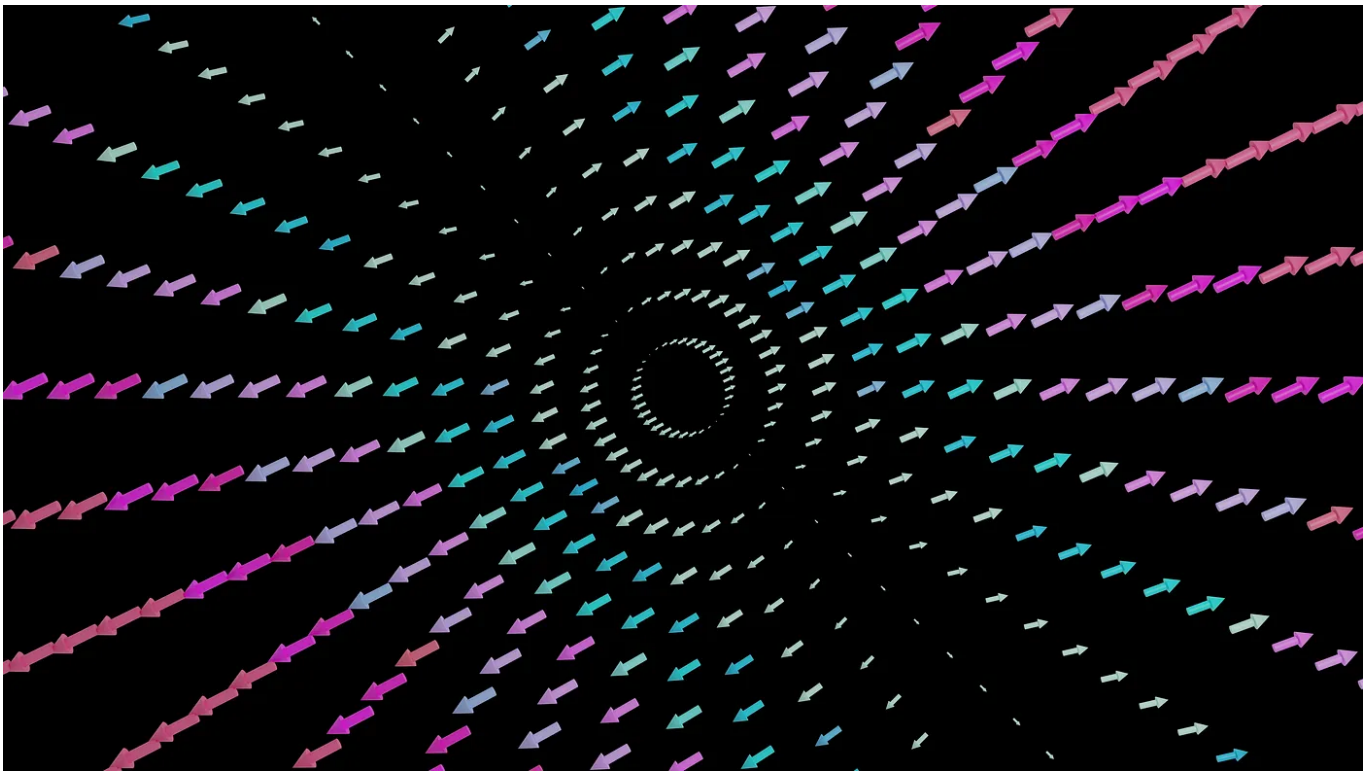
Member-only story

Eigenvalues and Eigenvectors

Joseph Mellor · Following
16 min read · Sep 4

Eigenvalues and eigenvectors are the most powerful tools in Linear Algebra.



This article is part 7 in the How to Discover Finite Fields series.

We've talked about matrices and linear transformations in the previous article in the context of solving systems of equations, but there are often many cases where we have a linear transformation of a space to itself. In cases like that, we can repeatedly apply the linear transformation.

- Previous Article
- Next Article
- All Articles

Repeatedly applying linear transformations can be quite time consuming. Furthermore, while the columns of a matrix indicate where we send the basis vectors, it's difficult to interpret what's going on with the matrices. The solution to both of these problems involves rewriting everything in the eigenbasis.

. . .

Prerequisites

While this is part of a series, you don't need to know anything from the previous articles except what was covered in the previous article.

. . .

Motivation

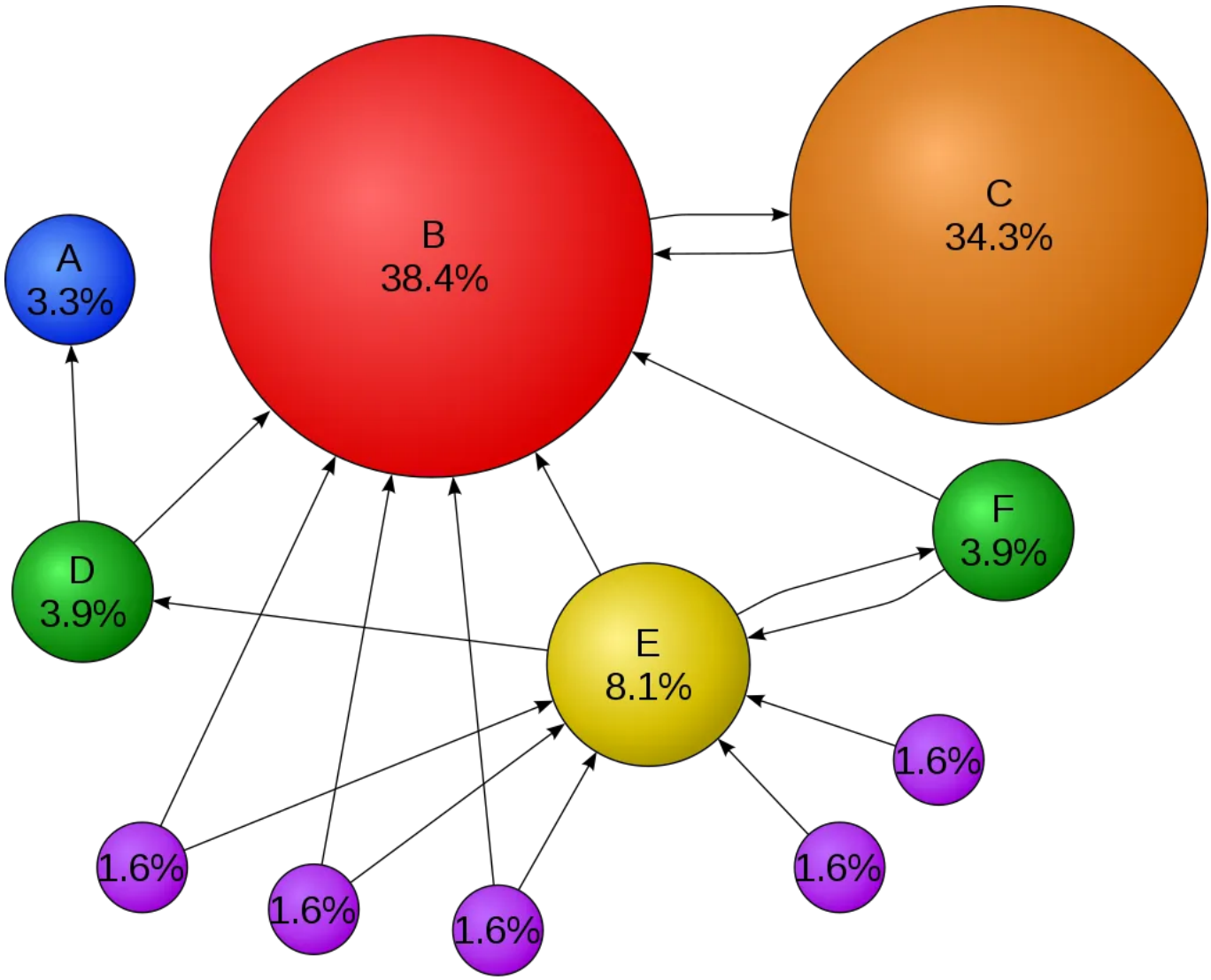
While I've given some general motivation in the intro to the article, the motivation will become more clear if we give some examples.

Markov Chains

Say you live in a city where the weather on one day depends on the weather of the previous day with the following rules:

- If it's sunny today, there's a 75% chance it will be sunny tomorrow and a 25% chance it will be cloudy tomorrow.
- If it's cloudy today, there's a 50% chance it will be sunny tomorrow and a 50% chance it will be cloudy tomorrow.
- Seasons don't affect the weather. I'm making this assumption for simplicity, not because we have to make the assumption. If you want, you can come up with a day-by-day change in the probabilities.

On a scale of Phoenix, Arizona to Seattle, Washington, how sunny would you say the city is? To approach this problem, you'll want to start with a 2D vector whose rows correspond to the probability that a given day will be sunny and the probability that a given day will be cloudy. All the elements of the vector add up to 1. If you're stuck, consider how this probability vector would change if the probability that it would be sunny today is 100%.



PageRank

Markov chains show up in other places too. Say you want to make a search engine where users give you a list of terms and you give the users a list of documents with those terms. Such a search engine would be decent, but it would be better if you showed users the most relevant documents first. While you might be able to do some natural language processing to figure out how much a given document fits a specific query, there's also the issue of credibility. In principle, people prefer to read things that match reality, but it's easy to write a lot of incoherent, inconsistent, or otherwise incorrect nonsense that sounds right. If you want your search engine to not be evil, you're going to have to in some way incorporate credibility into your

recommendation system. To do so, you either need expertise in everything or you need to know what experts consider to be reliable sources.

Finding Experts

It turns out that many of the experts have webpages with links to sources they consider to be reliable. Those sources often have links to other webpages considered to be reliable. So what you can do is come up with a list of experts, look at the links on their webpages, and use that to consider which webpages are credible. A webpage is more credible if there are a lot of experts recommending it and less credible if there are few experts recommending it.

The Problem with Finding Experts

There's a problem, though. Figuring out which people are experts in a topic requires that you be an expert in the topic itself. While you might be able to cheat a little by looking at professional scientists in various universities and companies, there are many topics that universities don't care about, such as the best training spots on Gorod Krovi, how to pick a guitar that's the right size for you, or the origin of the Roblox "oof" sound.

They won't teach you this in college.

So you're back at square one. You need to be enough of an expert in everything to find the experts in everything so that you can look at what they consider to be good resources on everything. There's a trick, though. Everyone's like a half-expert in at least one thing and people that are experts tend to make webpages about the thing. So, instead of finding a set of webpages specifically from a list of fixed experts, you start by considering every webpage kind of an expert in something. Then, you let every webpage lend some of its credibility to every webpage it links to.

If you repeat this process long enough, you would expect that some webpages end up with a lot of credibility while others end up with little credibility. But is that guaranteed to happen? If so, how long will it take? If

not, what would have to change for you to make that guarantee? In either case, what would change if webpages change what they link to?

Principal Component Analysis

One of the major goals of Data Science is to discover which variables can be used to predict outcomes. For example, say you want to predict something like what kinds of wine are most likely win a wine-tasting contest. In that case, you would measure a lot of different variables like color, alcohol content, how long it's been aged, etc. There's a bit of a problem because a lot of these variables are correlated with each other, some of the variables don't matter as much, the important variables could be implicit in something you've measured (e.g. tannin content is implicitly related to the color), and it might be difficult to find which equation best fits the data. The obvious question to ask would be "Is there some way to find out which of the variables we should consider?" Such a method would ideally give us independent variables in some sense and give us a measure for how much info each variable gives us.

Numerical Integration and Stability

I really need to do the *Numerical Methods* series. In the meantime, check out these videos.

As a brief summary, you can view each iteration as a linear transformation. You can then check the stability of the transformation by checking what happens to vectors over time. Stable linear transformations tend to keep

vectors roughly the same length or make them smaller while unstable linear transformations make the vectors bigger. So how do we tell if a linear transformation makes vectors bigger or smaller over time?

Physics

See *The Road to Quantum Mechanics*, particularly starting with the fourth article in the series. Also, the eigenvalues of certain operators in Quantum Mechanics correspond to the values you could get from a measurement, which include things like energy levels in atoms. Lastly, there are some computational benefits of working in orthogonal bases, such as working with inertia tensors in the Principal Axis frame (I'll have an article and maybe a video on that soon.).

. . .

Change of Basis Recap

The fact that we can represent matrices in different bases means we can choose the basis that best fits the problem. For many cases, we're looking for the **eigenbasis**, a basis made up of **eigenvectors**. To understand why, let's say that we want to repeatedly apply a linear transformation L to a vector $\boldsymbol{v}$.

$$\left(\hat{L}\left(\hat{L}\left(\ldots\left(\hat{L}\vec{v}\right)\ldots\right)\right)\right)$$

Let's make everything 2D so I don't develop carpal tunnel syndrome typing out everything, though this argument will work in arbitrary dimensions.

Since we can write all linear transformations as a matrix multiplication, we're going to replace the repeated application of a the linear transformation L with a matrix M being raised to some power.

$$\left(\hat{L}\left(\hat{L}\left(\ldots\left(\hat{L}\vec{v}\right)\ldots\right)\right)\right) = M^n\vec{v}$$

Since matrix multiplication is only defined when the number of columns of the first matrix is equal to the number of rows of the second matrix, we can only take powers of a square matrix, so M must be a square matrix. Since I specified that we're working in 2D, M must have the form

$$M = \begin{bmatrix} m_{11} & m_{12} \\ m_{21} & m_{22} \end{bmatrix}$$

Diagonal Matrices

Currently, when we apply M to a basis vector, it spits out a linear combination of all the basis vectors.

$$M \begin{bmatrix} 1 \\ 0 \end{bmatrix} = \begin{bmatrix} m_{11} & m_{12} \\ m_{21} & m_{22} \end{bmatrix} \begin{bmatrix} 1 \\ 0 \end{bmatrix}$$

$$= \begin{bmatrix} m_{11} \\ m_{21} \end{bmatrix}$$

$$= m_{11} \begin{bmatrix} 1 \\ 0 \end{bmatrix} + m_{21} \begin{bmatrix} 0 \\ 1 \end{bmatrix}$$

$$M \begin{bmatrix} 0 \\ 1 \end{bmatrix} = \begin{bmatrix} m_{11} & m_{12} \\ m_{21} & m_{22} \end{bmatrix} \begin{bmatrix} 0 \\ 1 \end{bmatrix}$$

$$= \begin{bmatrix} m_{12} \\ m_{22} \end{bmatrix}$$

$$= m_{12} \begin{bmatrix} 1 \\ 0 \end{bmatrix} + m_{22} \begin{bmatrix} 0 \\ 1 \end{bmatrix}$$

This fact makes computing high powers difficult. To make it simpler, we'd have to make the linear combination only contain one vector. For this to happen, we need all but one element of each column in the matrix to be zero. In this form, we have a few possibilities.

$$D_1 = \begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{bmatrix}$$

$$D_2 = \begin{bmatrix} 0 & \lambda_1 \\ \lambda_2 & 0 \end{bmatrix}$$

$$D_3 = \begin{bmatrix} \lambda_1 & \lambda_2 \\ 0 & 0 \end{bmatrix}$$

While the first two are fine, the last one is not. This matrix would send two different basis vectors to the same basis vector, which means our span decreases and the matrix is not invertible. If M is invertible, representing it in a different basis shouldn't change that. In other words, we need there to be one element per column and per row. This fact leaves us with a few choices. For 2D vectors, our choices are

$$D_1 \begin{bmatrix} 1 \\ 0 \end{bmatrix} = \lambda_1 \begin{bmatrix} 1 \\ 0 \end{bmatrix}$$

$$D_1 \begin{bmatrix} 0 \\ 1 \end{bmatrix} = \lambda_2 \begin{bmatrix} 0 \\ 1 \end{bmatrix}$$

$$D_2 \begin{bmatrix} 1 \\ 0 \end{bmatrix} = \lambda_2 \begin{bmatrix} 0 \\ 1 \end{bmatrix}$$

$$D_2 \begin{bmatrix} 0 \\ 1 \end{bmatrix} = \lambda_1 \begin{bmatrix} 1 \\ 0 \end{bmatrix}$$

The first one has the nice property that applying it to a basis vector scales the basis vector and does nothing else. The other ones scale the basis vectors and reassigns them. While it's still fine to deal with, it's quite annoying, so

we're going to go with the first one. Since it can only have non-zero values only on the diagonal, we call it a **diagonal matrix**.

Eigenvalues and Eigenvectors

Now, let's rewrite $\boldsymbol{v}$ in terms of the basis vectors for the diagonalized version of L (i.e. D) then apply powers of D to $\boldsymbol{v}$.

$$\vec{v} = \begin{bmatrix} a \\ b \end{bmatrix}$$

$$D\vec{v} = D\left(a \begin{bmatrix} 1 \\ 0 \end{bmatrix} + b \begin{bmatrix} 0 \\ 1 \end{bmatrix} \right)$$

$$= aD \begin{bmatrix} 1 \\ 0 \end{bmatrix} + bD \begin{bmatrix} 0 \\ 1 \end{bmatrix}$$

$$= a\lambda_1 \begin{bmatrix} 1 \\ 0 \end{bmatrix} + b\lambda_2 \begin{bmatrix} 0 \\ 1 \end{bmatrix}$$

$$D^2\vec{v} = a\lambda_1^2 \begin{bmatrix} 1 \\ 0 \end{bmatrix} + b\lambda_2^2 \begin{bmatrix} 0 \\ 1 \end{bmatrix}$$

$$D^3\vec{v} = a\lambda_1^3 \begin{bmatrix} 1 \\ 0 \end{bmatrix} + b\lambda_2^3 \begin{bmatrix} 0 \\ 1 \end{bmatrix}$$

As you can see, we've turned the process from a matrix exponentiation to scalar exponentiation. Furthermore, we can calculate powers of the matrix easily since we're just calculating powers of numbers. We call these numbers **eigenvalues**, this basis an **eigenbasis**, and the vectors in the eigenbasis **eigenvectors**.

Why Eigen-?

The German prefix *eigen-* means "same" in the context of math. It refers to the fact that applying a matrix to an eigenvector gets you a scaled version of the eigenvector without changing the direction of the eigenvector. Everything else with eigen- attached refers to either some other part of the process (e.g. an eigenvalue is how much an eigenvector gets scaled) or it refers to the part of the process in a specific context (e.g. an **eigenstate** refers to an eigenvector in the Hilbert space of a Quantum Mechanical state).

The Algorithm for Large Powers of Matrices

The full process consists of

1. Convert your vector from its current basis to the eigenbasis.
2. Calculate all the required powers of the eigenvalues.
3. Multiply each coefficient of each eigenvector by its corresponding eigenvalue raised to the necessary power.
4. Convert your vector from the eigenbasis to its original basis.

To convert your vector from its current basis to the eigenbasis, you have to solve the equation $Py = x$, where x is the vector in its current basis, y is the vector in the eigenbasis, and P is a matrix **with each column corresponding to an eigenvector**. The solution is given by $y = P^{-1}x$. In this basis, M acts like a diagonal matrix D, where each element on the diagonal corresponds to an eigenvector. To find the powers of M, you take powers of D. Lastly, to get it back into the original basis, you multiply by P. These transformations become

$$M = PDP^{-1}$$

$$M^n = PD^nP^{-1}$$

Since calculating powers of diagonal matrices is fast, this process can give a significant speed boost. You have to balance it against calculating the eigenvectors and P^{-1}, though. For small matrices and small powers, it might be more efficient to use the original value of M.

Orthonormal Eigenbasis

As you can imagine, choosing a basis can be quite important in Math and Physics. Choosing an **orthogonal basis** in which all vectors point in totally different directions allows you to get rid of a lot of cross terms and it means it's easy to rewrite the vectors in the orthogonal basis with some dot products. Furthermore, choosing a **normal basis** means that all the basis vectors have unit length which means you don't have to deal with vectors of different lengths. You can make any basis a normal basis by normalizing all the vectors in your basis or dividing them by their length, so you can make

any orthogonal basis into an **orthonormal basis**. The last way you can improve your basis involves choosing an eigenbasis, which makes applying the linear operators easy. If you can find a set of orthonormal eigenvectors that span the entire space, you have an **orthonormal eigenbasis**.

The Problem

Orthonormal eigenbases only exist for Hermitian operators. Granted, those operators are pretty important in Physics, but generally you have to solve a system of linear equations to get your vectors into the eigenbasis.

Degeneracy

It's possible to have multiple eigenvectors that all have the same eigenvalue. We call the eigenvalue **degenerate** and we describe this situation as **degeneracy**. If we're in this scenario, we have an infinite number of choices for our eigenvalues as any non-zero linear combination of eigenvectors with the same eigenvalue is also an eigenvector. In that case, we should try to choose our eigenvectors so that they are orthogonal to each other. We can use the **Gram-Schmidt process** to find these vectors.

Finding the Eigenvalues

So now, we have the question of how to find the eigenvalues. Luckily for me, this process is everywhere on the internet and in textbooks since it's all people talk about. You can see the whole process worked out in the video below.

A Better Method

From a practical standpoint, this method is pretty inefficient (more specifically, calculating the determinant for an $n \times n$ matrix using the Laplace expansion is $O(n!)$). To speed it up, you often use row reduction (which is $O(n^3)$) to put it in a form where calculating the determinant is simple.

There are a million other tricks for approximating determinants, eigenvalues, and eigenvectors that I'm going to cover in my upcoming *Numerical Methods* series, so be sure to follow me so you see it when it comes out. For this series, we'll need the exact characteristic polynomial, so we're stuck with these two methods (The actual algorithm I'm using is whatever `sympy` uses, but this algorithm should be fine.).

The Actual Method We'll Use

The matrices we're using have some restrictions that we can use to figure out the eigenvalues with some tricks.

The Eigenvalue Approach to Invertibility

For this section, we're going to assume that all our matrices are diagonalizable. We might talk about the general case in a later article, though. Let's say we have a matrix M and we diagonalize it to get a new matrix D. The eigenvalues are given by λ_1 to λ_n, so the diagonal matrix can be written as

$$\left[\begin{array}{ccc|ccc} \lambda_1 & \dots & 0 & 1 & \dots & 0 \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ 0 & \dots & \lambda_n & 0 & \dots & 1 \end{array}\right]$$

To get its inverse, we can look at the reduced row echelon form of the augmented matrix. Doing so will get us

$$\left[\begin{array}{ccc|ccc} 1 & \dots & 0 & \frac{1}{\lambda_1} & \dots & 0 \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ 0 & \dots & 1 & 0 & \dots & \frac{1}{\lambda_n} \end{array}\right]$$

For this inverse to exist, all of the eigenvalues must have a multiplicative inverse. Since we're working in a field, the only way for an eigenvalue to not have a multiplicative inverse is if it's zero. At the end of this thought process, we get that **a matrix is invertible if and only if it does not have zero as an eigenvalue.**

Determinant Approach to Invertibility

Since the determinant is the product of all the eigenvalues, **a matrix is invertible if and only if its determinant is nonzero.**

Geometric Approach to Invertibility

The determinant and basis approach to invertibility are closely related to the geometric approach to invertibility, in which **a matrix is invertible if and only if the zero vector is the only vector in the null space.** If other vectors are in the null space, then you have multiple vectors mapped to the same output, which leads to some outputs having multiple possible inputs (underdetermined) and others having no inputs (inconsistent). You can see both the determinant and geometric explanations in the 3blue1brown video below.

. . .

Applications

So now, we're going to go back through the applications mentioned in the **Motivation** section and discuss how they apply to each problem.

Markov Chains

If you run a Markov chain long enough, you'll find that it tends towards a steady state value. For example, if we say it's sunny today and we assume the Markov chain presented in the **Motivation** section is an accurate model, then the probability of it being sunny after 365 days is about 2/3 and the probability of it being cloudy after 365 days is about 1/3. The whole process can be written mathematically as

$$\begin{bmatrix} P\Big(\text{Weather After 365 Days} = \text{Sunny}\Big) \\ P\Big(\text{Weather After 365 Days} = \text{Cloudy}\Big) \end{bmatrix} = \begin{bmatrix} \frac{3}{4} & \frac{1}{2} \\ \frac{1}{4} & \frac{1}{2} \end{bmatrix}^{365} \begin{bmatrix} 1 \\ 0 \end{bmatrix} \approx \begin{bmatrix} \frac{2}{3} \\ \frac{1}{3} \end{bmatrix}$$

To get equality, we can take the limit after as the number of days approaches infinity.

$$\lim_{n\to\infty} \begin{bmatrix} P\Big(\text{Weather After } n \text{ Days} = \text{Sunny}\Big) \\ P\Big(\text{Weather After } n \text{ Days} = \text{Cloudy}\Big) \end{bmatrix} = \lim_{n\to\infty} \begin{bmatrix} \frac{3}{4} & \frac{1}{2} \\ \frac{1}{4} & \frac{1}{2} \end{bmatrix}^{n} \begin{bmatrix} a \\ b \end{bmatrix} = \begin{bmatrix} \frac{2}{3} \\ \frac{1}{3} \end{bmatrix}$$

It turns out that the resulting vector is approximately the eigenvector of the transition matrix T

$$T = \begin{bmatrix} \frac{3}{4} & \frac{1}{2} \\ \frac{1}{4} & \frac{1}{2} \end{bmatrix} = \begin{bmatrix} -1 & 2 \\ 1 & 1 \end{bmatrix} \begin{bmatrix} \frac{1}{4} & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} -\frac{1}{3} & \frac{2}{3} \\ \frac{1}{3} & \frac{1}{3} \end{bmatrix}$$

corresponding to an eigenvalue equal to 1. To see why, we can rewrite our vectors in the eigenbasis. We have two eigenvalues: 1 and 1/4. Since 1/4 is less than 1, the part of any initial state in the direction of the corresponding eigenvector will eventually go to zero while the part in the direction of the eigenvector corresponding to 1 will remain.

$$\begin{aligned} \lim_{n\to\infty} D^n \vec{v}_0 &= \lim_{n\to\infty} D^n \left(a\hat{e}_1 + b\hat{e}_2 \right) \\ &= \lim_{n\to\infty} aD^n\hat{e}_1 + bD^n\hat{e}_2 \\ &= \lim_{n\to\infty} a(1)^n\hat{e}_1 + b\left(\frac{1}{4}\right)^n\hat{e}_2 \\ &= \lim_{n\to\infty} a\hat{e}_1 + b\left(\frac{1}{4}\right)^n\hat{e}_2 \\ &= a\hat{e}_1 + b(0)\hat{e}_2 \\ &= a\hat{e}_1 \end{aligned}$$

In other words, the eigenspace corresponding to an eigenvalue with an absolute value of 1 describes the steady state parts of the system while all the eigenvalues with absolute values less than 1 will correspond to transient parts of the system.

PageRank

The flow of credibility can be measured using a Markov process. You start by picking a random webpage and following links at random. You also count the number of times you end up on a webpage. Credible websites will have a higher count since they have a lot of credible sources linking to them. This count is an approximation of the steady state probability distribution. Granted, you have to do a bit more to deal with a few bad faith actors and a preference for older stuff over newer stuff, but it's a good start.

How do We Know It Will Converge?

To prove that there is a steady state distribution, we need an eigenvalue with an absolute value of 1. You can find a proof that we always have 1 as an eigenvalue in the Definition and Properties section of this Wikipedia article.

Principal Component Analysis

As you might expect, finding the best variables for the system involves finding variables that are independent from each other.

Covariance

To measure the independence of variables, you look at their **covariance**.

$$\operatorname{cov}\Big(X,Y\Big) = E\Big[\Big(X - E\big[X\big]\Big)\Big(Y - E\big[Y\big]\Big)\Big]$$

Here, the E represents the **expected value** of the stuff inside the square brackets. You can think of the magnitude of the covariance as a measure of how accurately you can guess one of the variables given the other variable. For example, wine that has a lot of tannin tends to be more red, more bitter, etc., so the covariance between tannin content and bitterness/redness should be high. On the other hand, there are plenty of bitter wines that are white and plenty of sweet wines that are red, so the covariance between bitterness and redness should be close to zero.

Variance

We can also define the **variance** of a variable as the covariance of it with itself. Since you want to predict the value of the variable without knowing any of the other variables, your best guess is to pick the average of the variable. Now, the variance measures how accurate your guess for the variable will be.

Diagonalizing the Covariance Matrix

The concept of covariance allows us to greatly simplify our problem of finding independent components. We just need to find a bunch of linear combinations of our current variables that span the whole space and have zero covariance. If we put our variances and covariances into a matrix, this condition becomes the requirement that our covariance matrix is diagonal.

Missing Eigenbasis

You often don't have an eigenbasis of a system because you often have a many to many or a many to one function. In either case, the dimensions

don't add up. You can, however, still write your system as a matrix and then find its **singular value decomposition**. Once you find the independent variables (or principal components), you can tell just how much each one affects the system by looking at the corresponding singular values (which are closely related to the eigenvalues). You can then get rid of variables that don't contribute much to the outcome in a process called **dimensionality reduction**.

Numerical Integration and Stability

Our condition for stability is that vectors tend not to get longer. To make sure these vectors don't grow, we need all our eigenvalues to have an absolute value less than or equal to one. The standard Euler method is pretty awful partly because it tends to have eigenvalues with a magnitude greater than one while methods like the Semi-Implicit Euler method tend to have eigenvalues of exactly one, which roughly corresponds to making sure certain quantities in the system are approximately conserved.

With that being said, you could also look for other methods that constrain the eigenvalues like in the video below.

What's Next

There are a bunch of fun and interesting problems that I've left out in finding the eigenbasis (most notably, the Jordan Normal Form and degenerate eigenvectors). As we go further in this series, we'll end up filling most of the gaps I've left. In the next article, we're going to start applying the concepts we've set up in this series to the initial problem of finding cycle lengths and delay lengths in a specific cellular automata.

Self-Promotion

If you liked this article, you probably know someone else who will. It would help me out if you could share this article with them. If you really liked this article or any of my other articles, you can help me write them by donating to my ko-fi account. If you're not already a Medium member and you like the

articles on the website, you can name me as your referred member and a portion of your monthly fee will help support me. Lastly, if you know of a cool application or idea that relies on topics covered in my articles, let me know in a response, DM, death threat, etc.

Math Mathematics Science

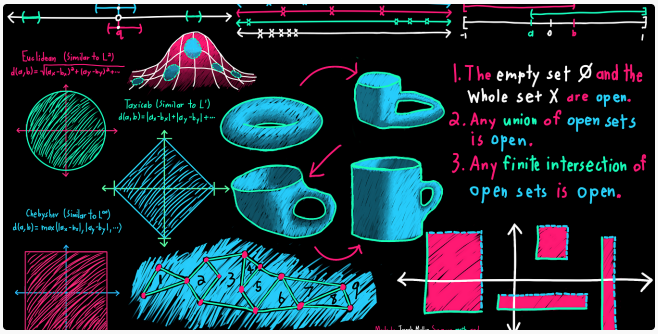
Written by Joseph Mellor

Following

3.3K Followers

BS in Physics, Math, and CS with a minor in High-Performance Computing. You can find all my articles at https://josephmellor.xyz/articles/.

More from Joseph Mellor



Joseph Mellor in Cantor's Paradise

An Intro to Topology

Topology is the study of how you can squish and stretch things without gluing or cutting,...

✦ · 22 min read · Nov 16, 2022

491 7

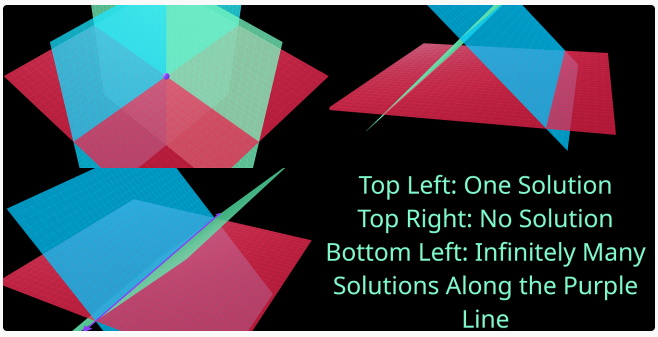
Joseph Mellor

Stop Saying Entropy is Disorder

We misunderstand entropy because we're taught it's a measure of disorder, but the real...

✦ · 15 min read · Oct 4, 2020

164 3



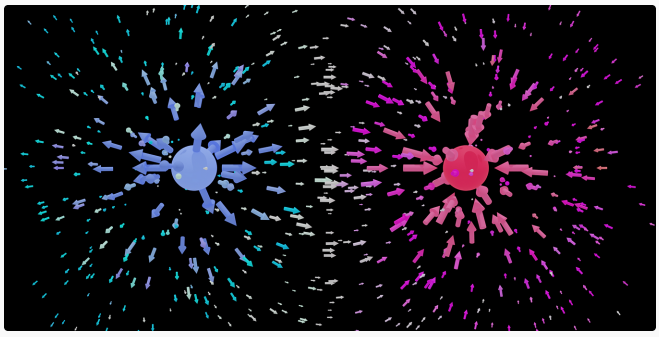
Joseph Mellor in Cantor's Paradise

An Intro to Abstract Linear Algebra

Linear Algebra is a lot more than just a way to move lines around in space.

✦ · 21 min read · Nov 24, 2022

213



Joseph Mellor

Update: I'm Still Alive and Making Content

I haven't published an article in a few months because I've been incredibly busy, but I'm...

✦ · 3 min read · Sep 3

20 3

See all from Joseph Mellor

Recommended from Medium

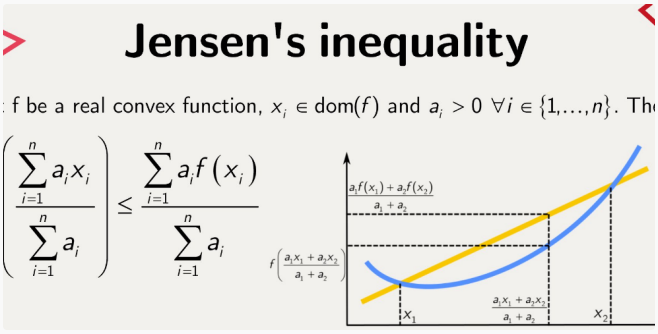
Ali

The Lifelong Learner's Toolkit: 50 Useful Websites for Continuous...

Hey there! If you're anything like me, you love finding useful websites that make a differenc...

✦ · 43 min read · Sep 10

1.1K 12



Hossam Hamdy in The Modern Scientist

Jensen's Inequality (Part I): proof

Imagine you're a photographer and you're trying to take a picture of a beautiful...

7 min read · Jul 4

81

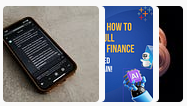
Lists

6 Science-Backed Health Stories on Covid, Sleep, and...

6 stories · 38 saves

ChatGPT

21 stories · 156 saves



ChatGPT prompts

24 stories · 396 saves

What is ChatGPT?

9 stories · 175 saves

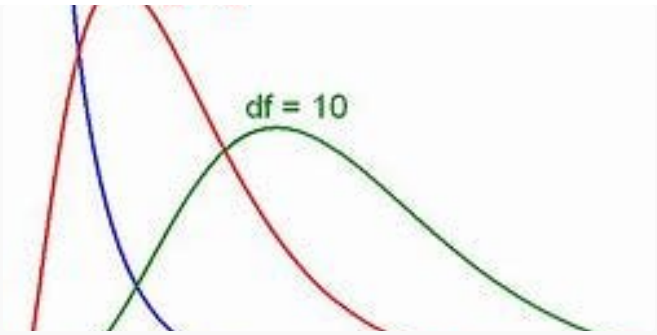
Wes O'Donnell

Ukraine is Using its Leopard Tanks at Night as 'Nocturnal Predators'

The incredible advantage of night fighting in war

✦ · 7 min read · 5 days ago

3.2K 23



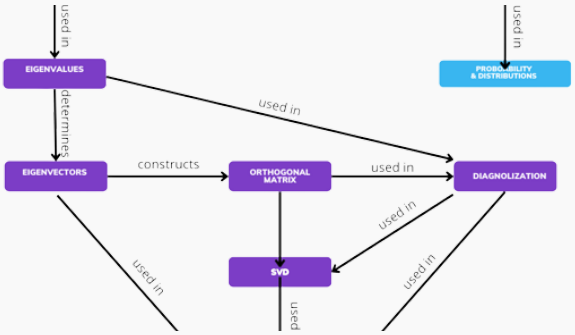
ajaymehta

"Unlocking the Power of Chi Square: A Guide to Statistical...

Chi Square Distribution

14 min read · Apr 14

55



Haythamcheikhrouhou

Don't Worry About it (Part 1) : Linear Algebra

Welcome to the "Don't Worry About It" series, where we're here to explain the mathematica...

7 min read · Aug 14

169 2

Franciszek Szewczyk

Rope Simulator in C++

For starters, we're going to implement a simple rope simulator. We will be using C++,...

7 min read · Sep 11

26 1

See more recommendations