Data 345

Applied Linear Algebra for Statistical Learning

Class 1 (Aug. 25, 2025)

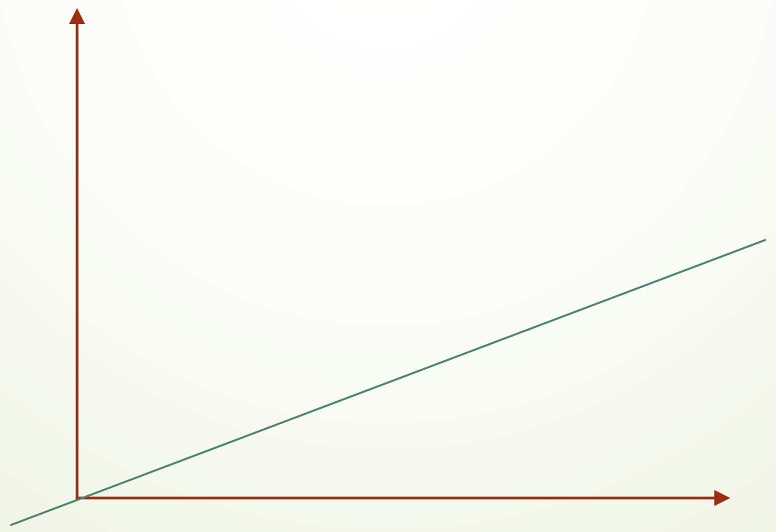
Data 345

- **Catalog Description:** Students will apply matrix theory to the study and implementation of linear models to the problems in data science. Topics include basic matrix theory with applications to optimization, and machine learning.
- Focus: Application and implementation (no proofs!*)
- Basic building blocks: vectors, matrices, dot products, matrix factorizations
- Use custom `data345` Python library, which we will build over the semester

* okay maybe very basic soft proofs please don't be mad

What *is* Linear Algebra?

- Fundamentally, linear algebra is the study of linear transformations between "linear spaces."
- Basic example: $f: \mathbb{R} \to \mathbb{R}$ given by $f(x) = \frac{1}{2}x$



What *is* Linear Algebra?

- In the case of real-valued functions of a single variable, linear functions are precisely those functions whose graphs are lines passing through the origin.
- These linear functions are completely characterized by the equation $f(x) = cx$, where c is some real number.
- Obviously, there must be more to this story, else the study of linear algebra would be extraordinarily boring.
- In algebra classes, we are predisposed to solving equations. Linear algebra is not different in this respect.

Linear Equations

A **linear equation** in a single variable x is any equation of the form

$$ax + b = 0,$$

Where $a \neq 0$ and b are real numbers. To **solve** such an equation means to determine the set of all x for which the equation $ax + b = 0$ evaluates as true. These are easy to solve (spoiler: $x = -\frac{b}{a}$).

A **linear equation** in two variables x and y is any equation of the form

$$ax + by + c = 0,$$

Where $a \neq 0, b \neq 0$, and c are real numbers. To **solve** this equation, again, means to determine the set of all (x, y) that satisfy this equation. Solving is super easy:

$$y = -\frac{a}{b}x - \frac{c}{b}$$

Linear Equations

A single point forms the solution set for a linear equation in a single variable, and a line forms the solution set for a linear equation in two variables.

Similarly, a plane is a solution for a linear equation in three variables, and so on. Basically, if you have n variables in a single linear equation, you have what we call $n - 1$ "degrees of freedom," in that you must choose $n - 1$ values before all n values of the variable are determined.

This is hard to imagine geometrically in higher dimensions, but as a matter of procedure, it's not too difficult to generate a solution to a linear equation.

Simultaneous Linear Equations

It is not often that we will find ourselves concerned with solving a single linear equation at a time.

Linear algebra as a mathematical discipline was developed to solve **multiple** linear equations **simultaneously**.

This sort of problem has been of human interest since ancient times. Take, for example, the Chinese text *Chiu-chang Suan-shu* (*Nine Chapters on the Mathematical Art*), written by generations of Chinese scholars from the 10th to the 1st century BCE. The eighth chapter "Fangcheng" ("the two-sided reference") focused explicitly on solving simultaneous equations.

Simultaneous Linear Equations

"Now given 3 bundles of top grade paddy, 2 bundles of medium grade paddy, [and] 1 bundle of low grade paddy. Yield: 39 dou of grain. 2 bundles of top grade paddy, 3 bundles of medium grade paddy, [and] 1 bundle of low grade paddy, yield 34 dou. 1 bundle of top grade paddy, 2 bundles of medium grade paddy, [and] 3 bundles of low grade paddy, yield 26 dou. Tell: how much paddy does one bundle of each grain yield?"

-Problem 1, Chapter VIII, *Chiu-chang Suan-shu*

Simultaneous Linear Equations

Let's let T represent the yield of a "top grade" bundle of rice paddy, M represent the yield of a "middle grade" bundle of rice paddy, and L represent the yield of a "low grade" bundle of rice paddy.

Even if we are not familiar with the units or how rice paddy is graded, the overall structure of the problem is familiar!

$$3T + 2M + L = 39$$
$$2T + 3M + L = 34$$
$$T + 2M + 3L = 26$$

This is a **system of linear equations** in three unknowns!

Solving a System of Linear Equations

A few observations about how “=“ works:

If we know that $a = b$, and c is a nonzero real number, we also know that $ac = bc$. This means multiplying both sides of an equation by a number does not affect the truth of equality.

Similarly, if we know $a = b$ and $c = d$, then we could also observe that $a + c = b + d$. In other words, we can add together the left- and right-hand sides of two separate equations and end up with another true equation.

Solving a System of Linear Equations

$$\begin{cases} 3T + 2M + L = 39 \\ 2T + 3M + L = 34 \\ T + 2M + 3L = 26 \end{cases}$$

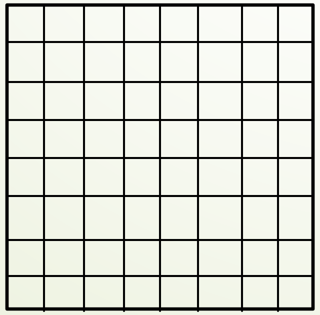
$$\rightarrow \quad \begin{array}{cccc} T & M & L & RHS \end{array}$$
$$\begin{bmatrix} 3 & 2 & 1 & 39 \\ 2 & 3 & 1 & 34 \\ 1 & 2 & 3 & 26 \end{bmatrix}$$

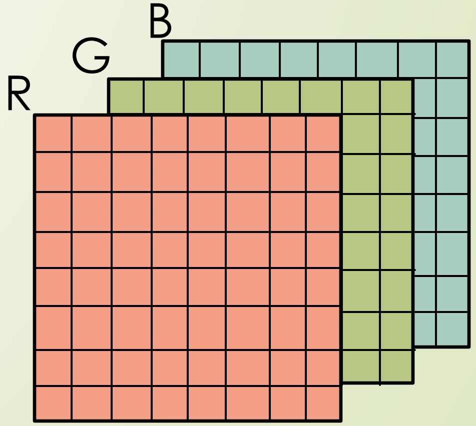
Linear Algebra in the Modern Age

- Solving linear equations may have been the genesis of linear algebra, but its utility is by no means limited to equation solving (although this would be enough on its own!)
- Principles of linear algebra form the **backbone of data science methods**, whether we realize it or not.

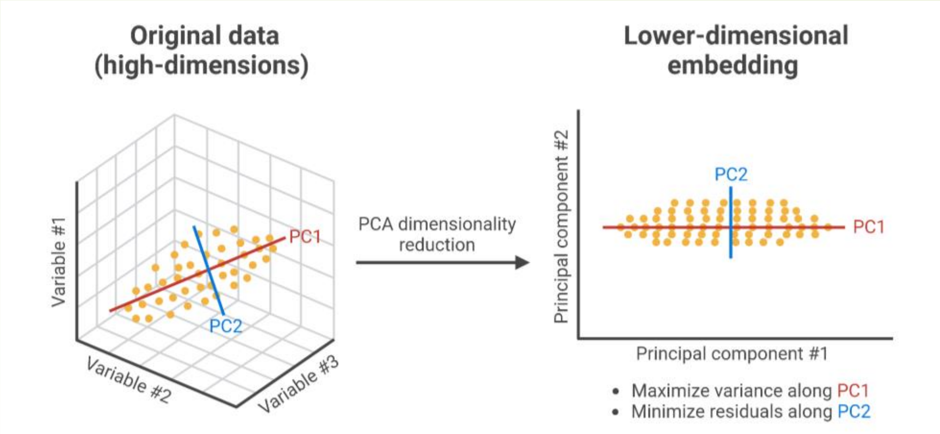
Data Representation

- In Machine Learning (ML) and so-called Deep Learning (DL), data is very often represented as <u>vectors</u> (1-dimensional arrays) or <u>matrices</u> (2-dimensional arrays).
- Tools from linear algebra give us the language and machinery to manipulate these data structures.
- Example: Grayscale image to matrix of pixel intensities
- Example: Color image to 3D tensor (height-width-color channels)
- Example: Dataset with n samples and d features, as an $n \times d$ matrix



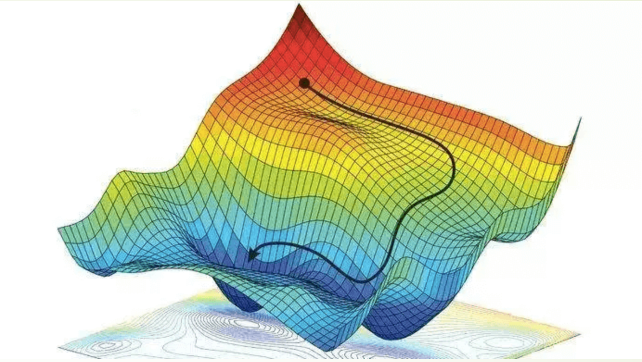


Transformations



- Many ML methods incorporate <u>linear transformations</u>, whether in scaling/cleaning/preparing the data or in the implementation of a particular algorithm.
- Operations that modify data, like scaling, rotating, or projecting, can be written as matrix operations on some vector space.
- Example: Principal Component Analysis (PCA) is a technique to reduce the number of features you're using; it uses eigenvalues and eigenvectors to project your data onto a lower-dimensional subspace.
- Neural network (NN) layers utilize matrix multiplication plus a nonlinear term to predict outputs on unseen training data. ($y = f(Wx + b)$ where W is a matrix of weights, x is an input vector, and f is an activation function)

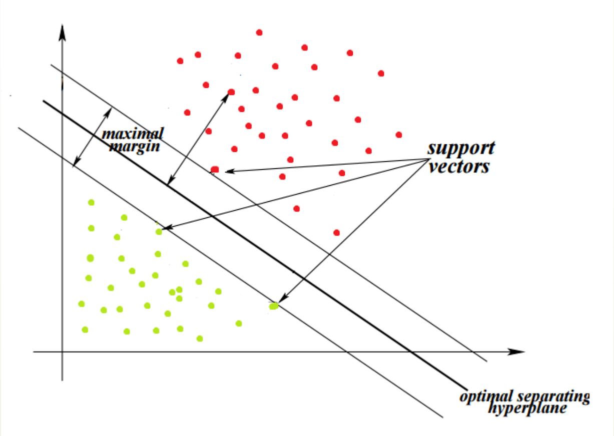
Optimization



- Generally "optimization" refers to finding the largest (i.e., "maximizing") or smallest (i.e., "minimizing") value of interest.
- In statistics and data science, particularly in ML, the goal is often to predict an output based on training data. Something called a loss function quantifies the difference between the *predicted* output of a ML algorithm and the *actual* target values. (Think: something like an error margin.)
- To make models as good as possible we want this loss function to be minimized.
- In calculus I, basic optimization procedure requires you to take a derivative and set it equal to 0.
- When multiple variables are involved, you still "take the derivative and set it equal to 0," but the steps are more complicated. In multiple variables derivatives are represented with matrices, and methods like gradient descent iteratively use multivariate derivatives to minimize loss.

Core Methods of Machine Learning

- The classic "ole reliable" of machine learning is something called *linear regression*, which is where you model data points with a linear function; that linear function is (or can be) determined easily with linear algebra.
- Solving $\hat{y} = X\beta$ would give $\hat{\beta} = (X^T X)^{-1} X^T y$ which allows us to determine the coefficients of the linear regression.
- The method of support vector machines (SVM) for classifying data essentially tries to find a hyperplane with which the data can be separated, and depends on things like the inner (dot) product of vectors.



Computational Efficiency

- Modern ML/DL often requires doing operations on many different elements of arrays. Rather than use loops to step through each array the operations are often *vectorized* and done in parallel.
- Particularly in interest for DL applications, computer graphics processing units (GPUs) are optimized for doing linear algebra operations, like matrix multiplications, inner products, tensor contractions, etc.
- DL frameworks like TensorFlow or PyTorch are extremely well-optimized linear algebra libraries with differentiation functionality.

A Look Ahead

- Basic data structures for linear algebra (vectors / matrices)
- Vector / matrix operations (addition / scalar multiplication / dot product)
- Row operations for matrices
- Solve systems of equations with Gaussian elimination
- Matrix rank & inverses
- Linear independence and basis
- Orthogonality & projections
- Least squares / linear regression
- Eigenvalues and eigenvectors
- PCA
- SVD & Applications