Linear Algebra and Calculus

Siju Swamy

2025-08-11

Table of contents

1 Introduction to Linear ALgebra and Calculus

2 Introduction to Linear ALgebra and Calculus

2.1 Why Are You Here?

Welcome. You are here because you want to build the future. You want to design the next generation of intelligent devices, write the code that powers artificial intelligence, and create the communication systems that connect the world. My goal in this course is not to just teach you mathematics, but to give you the fundamental language and toolkit you will use to achieve those goals.

Over the next five years, the fields of electronics and computer science will be dominated by machine learning, robotics, next-generation wireless communication (5G and 6G), and incredibly complex integrated circuits. The code you'll write, the systems you'll design—they all speak a language. That language is a beautiful combination of linear algebra and calculus.

This course is your Rosetta Stone. We will move beyond memorizing formulas and instead focus on three questions for every topic:

1. **What is the core idea?** (The Intuition)
2. **Why does it work?** (The Theory)
3. **What can I build with it?** (The Application)

Let's look at the journey ahead and connect it directly to the technologies you will be creating.

2.2 The Syllabus: A 4-Year Technology Roadmap

2.2.1 Module I: Systems of Linear Equations — The Language of Problems

- **The Math:** You'll learn to solve systems of equations, `Ax = b`. We'll explore this through the "Row Picture" (intersecting planes) and the "Column Picture" (combining vectors).
- **The 4-Year Horizon:**

- **VLSI Chip Design:** A modern processor has billions of transistors. Analyzing the voltages and currents across this massive network is a linear algebra problem on an unimaginable scale. The methods we start with here are the foundation for the software that designs and verifies the chips in your phone and computer.
- **Network Analysis:** How does Google balance traffic across its millions of servers? How does data flow through the internet? These are gigantic systems of linear equations, where the variables are data rates and server loads.
- **Machine Learning Models:** Training a simple model can involve solving for thousands of parameters simultaneously. This is `Ax = b` in disguise.

2.2.2 Module II: Eigenvalues & Eigenvectors — The DNA of a System

- **The Math:** We will find the "special" vectors of a matrix, the ones that don't change direction when transformed (`Ax = x`). These are the eigenvectors, and they reveal the deepest properties of a system.
- **The 5-Year Horizon:**
  - **Principal Component Analysis (PCA):** How does your phone recognize your face so quickly? It uses PCA to reduce a high-resolution image to its most important features—its "principal components." These are, quite literally, the eigenvectors of the data's covariance matrix. This is essential for data compression and machine learning.
  - **Quantum Computing:** The state of a qubit is a vector. The possible measurement outcomes are the eigenvalues of its operator matrix. The fundamental principles of quantum mechanics, which will power the next computing revolution, are expressed in the language of eigenvalues.
  - **Robotics and Control Systems:** When you design a robot arm or a drone, you need it to be stable. The eigenvalues of its control system matrix tell you if it will shake itself apart or smoothly return to equilibrium. A positive eigenvalue could mean disaster!

2.2.3 Modules III & IV: Multivariable Calculus — The Landscape of Optimization

- **The Math:** We'll explore functions of multiple variables, finding their peaks and valleys using partial derivatives (the gradient `f`) and calculating volumes under surfaces with multiple integrals ( ).
- **The 4-Year Horizon:**
  - **Gradient Descent (The Engine of AI):** This is the single most important application of calculus for you. How does a neural network learn? It calculates an "error landscape" (a high-dimensional surface) and uses the gradient to find

the direction to "descend" towards the lowest error. Every time you hear about a model being "trained," you are hearing about gradient descent in action. This is the engine that drives the entire AI revolution.
  - **Computer Graphics & AR/VR:** How does a GPU render realistic lighting and shadows on a 3D object in a game? It calculates surface normals (using gradients) and performs integrals over surfaces to determine how light reflects. This is multivariable calculus happening in real-time.
  - **Robotics Path Planning:** A robot navigating a complex terrain uses calculus to find the optimal, most energy-efficient path—essentially finding a "valley" in a "cost landscape."

2.2.4 Module V: Series Representation — The Art of Approximation

- **The Math:** We will learn to approximate complex functions with simpler building blocks: polynomials (Taylor Series) and sine/cosine waves (Fourier Series).
- **The 4-Year Horizon:**
  - **Signal Processing (5G, Wi-Fi, Audio):** Your phone receives a messy, complex radio wave. The **Fourier Transform** breaks this signal down into its constituent frequencies, separating the data from the noise. This is the absolute heart of all modern digital communications and audio/video compression (like MP3 and JPEG). The future of wireless technology is built on Fourier analysis.
  - **Embedded Systems & IoT:** A tiny sensor in an IoT device doesn't have the processing power to calculate `sin(x)` precisely. Instead, it uses the first few terms of a Taylor series—a simple polynomial—to get an answer that is "good enough" while saving precious battery life and clock cycles.

2.3 How We'll Learn: Building Computational Intuition

To make these connections real, we won't just use pen and paper. We will use Python, the language of modern scientific computing and AI. You will learn to use libraries like NumPy, Matplotlib, and Plotly to see these concepts in action.

For example, when we say that solving `Ax=b` is about finding the right combination of column vectors, what does that *look* like? It looks like this:

```
import numpy as np
import matplotlib.pyplot as plt

# The column vectors from our system
v1 = np.array([2, 1])     # First column
```

```
v2 = np.array([-1, 1])    # Second column

# The target vector on the right-hand side
b = np.array([1, 5])      # Should equal 2*v1 + 3*v2

# The solution we will learn to find is x=2, y=3
x, y = 2, 3

# --- Visualization ---
plt.figure(figsize=(8, 8))
ax = plt.gca()

# Plot the basis vectors
ax.quiver(0, 0, v1[0], v1[1], angles='xy', scale_units='xy', scale=1,
          color='blue', label=r'Column 1: $v_1$')
ax.quiver(0, 0, v2[0], v2[1], angles='xy', scale_units='xy', scale=1,
          color='green', label=r'Column 2: $v_2$')

# Plot the linear combination step by step
ax.quiver(0, 0, (x*v1)[0], (x*v1)[1], angles='xy', scale_units='xy', scale=1,
          color='lightblue', alpha=0.8, label=r'$2v_1$')
ax.quiver((x*v1)[0], (x*v1)[1], (y*v2)[0], (y*v2)[1], angles='xy', scale_units='xy', scale=1
          color='lightgreen', alpha=0.8, label=r'$3v_2$')

# Plot the target vector b
ax.quiver(0, 0, b[0], b[1], angles='xy', scale_units='xy', scale=1,
          color='red', label=r'Target Vector $b$')

# Formatting
ax.set_xlim(-2, 6)
ax.set_ylim(-1, 7)
ax.grid(True)
ax.set_title("Visualizing the Solution to Ax=b")
ax.legend()
plt.show()
```

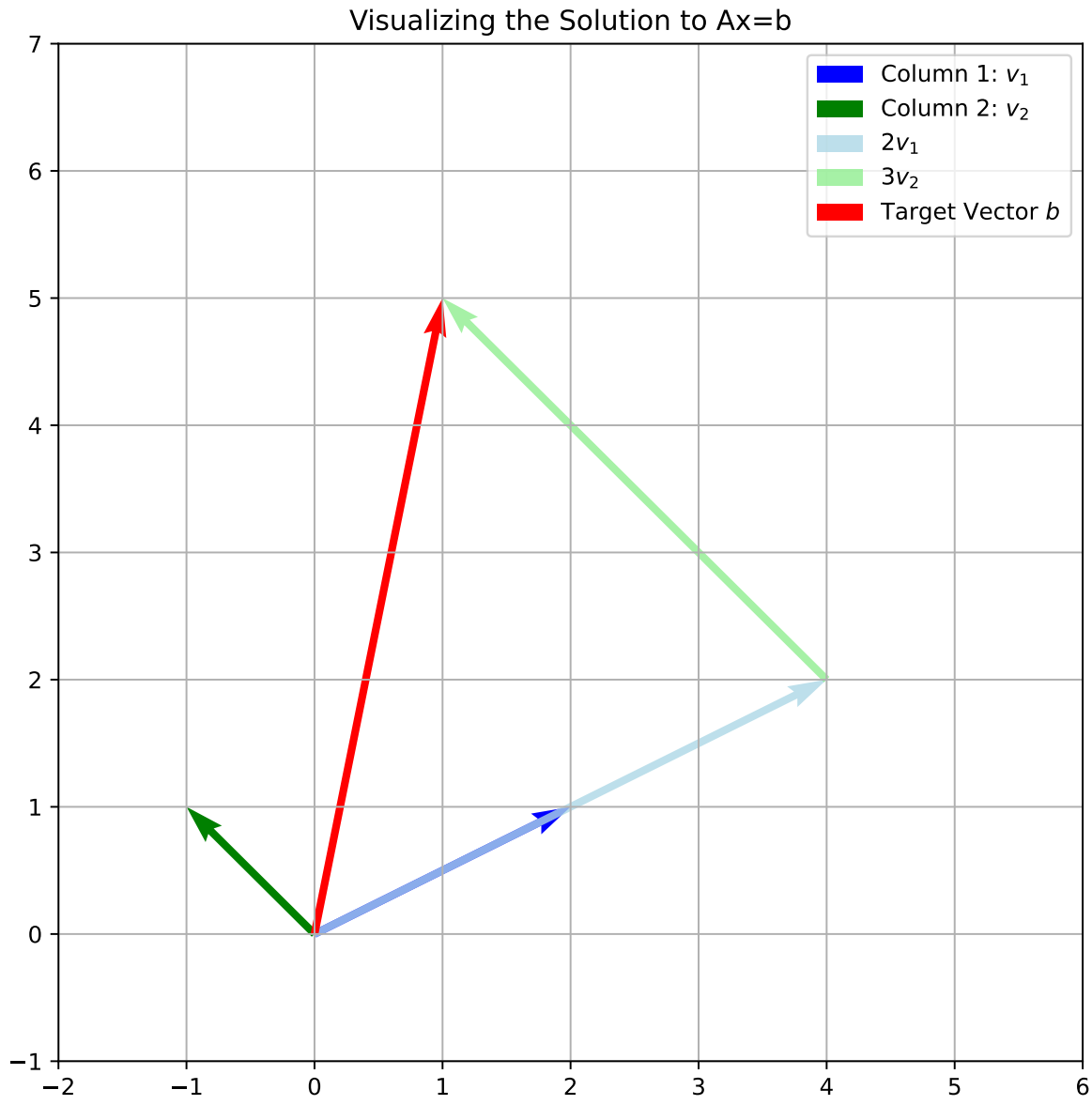

Figure 2.1: The Column Picture: Using Python to see that 2 of the blue vector plus 3 of the green vector builds the red target vector.

By the end of this course, you won't just be able to solve these problems. You will have a deep, visual, and computational intuition for them. You will see a problem in your own field and recognize the mathematical tools needed to solve it.

This course is your first and most important step towards becoming a true architect of future technology. Let's get started.

3 Module-1: Linear Systems, Properties, and its Solution

Syllabus: System of linear equations - Solution by Gauss elimination - Row Echelon form and Rank of a matrix - Fundamental theorem for linear systems of homogeneous and nonhomogeneous (statement only) - Homogeneous linear system - Non-homogeneous linear system