Linear Algebra

Guide to AI and Data Courses · Part V: Mathematical Foundations

Vectors and matrices | Linear systems | Eigenvalues | SVD and PCA | Where in AI | Neighboring courses

Sercan KÜLCÜ · Giresun University · Computer Engineering

© Sercan Külcü. All rights reserved. · Interactive HTML version of these slides: https://sercankulcu.github.io/files/ai_courses/slides/Chapter_31_Linear_Algebra.html

INTRODUCTION

What Question Does It Ask?

> **How do vectors and matrices represent data and transformations?**

Linear algebra studies vectors, matrices and **linear transformations**. In AI and data courses almost everything is a vector: an image, a word, a user. A neural network layer is, to a large extent, a matrix multiplication.

At a glance

APPROACH vector spaces and transformations	**INPUT** vectors, matrices	**OUTPUT** solution, decomposition, transformation
SUCCESS MEASURE accuracy, numerical stability	**MATH LOAD** the theory itself	**DATA NEEDS** none

A Short History

1844
Grassmann sets out the idea of a multidimensional vector space systematically for the first time.

1858
Cayley treats matrices as algebraic objects in their own right; matrix multiplication is defined.

1901
Pearson finds the lines and planes that best fit data: principal component analysis (PCA).

1936
Eckart and Young show that the SVD gives the best low-rank approximation of a matrix.

1965
Golub and Kahan give a numerically stable way to compute the SVD on a computer.

1979
BLAS is published: the basic matrix operations beneath today's scientific and AI libraries.

THE COURSE

Core Concepts

Vectors and spaces
Turning data into a point in many dimensions.

Matrix multiplication
Applying a transformation or a network layer.

Inner product and norm
Measuring similarity (cosine) and distance.

Rank and basis
How many independent directions the data really spans.

Linear systems
Ax = b; the least squares solution.

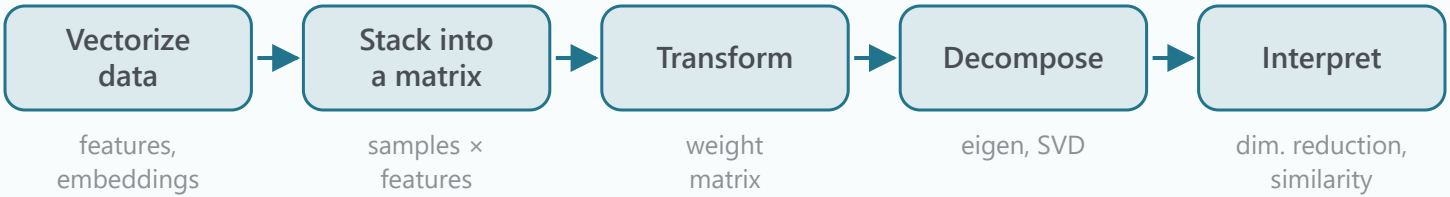
Eigenvalues, eigenvectors
Vectors whose direction a map leaves unchanged.

SVD
Splitting any matrix into rotate, scale, rotate.

Projection
The closest point to a vector in a subspace.

THE COURSE

A Typical Workflow



Example: In face recognition each face image is a long vector of pixels. The faces are stacked into a matrix, the mean is subtracted, SVD finds a few most important directions (eigenfaces), and each face is represented and compared by its weights on those few directions.

THE COURSE

What Does the Course Cover?

1. Vectors, linear combinations and dot products
2. Linear systems and Gaussian elimination
3. Matrix operations, inverses and LU factorization
4. Vector spaces and the four fundamental subspaces
5. Linear independence, basis, dimension and rank
6. Orthogonality, projections and least squares
7. Gram-Schmidt and QR factorization
8. Determinants and their properties
9. Eigenvalues, eigenvectors and diagonalization
10. Symmetric and positive definite matrices
11. Singular value decomposition (SVD) and PCA
12. Linear transformations and change of basis

Skills you will gain

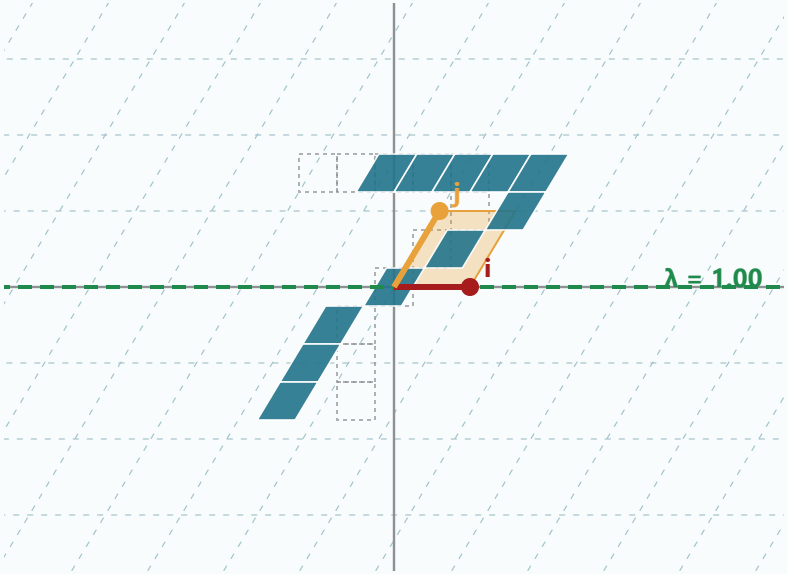
matrix computation with NumPy | solving systems with scipy.linalg | dimensionality reduction with SVD | least squares fitting | writing short proofs

A typical semester; the order and coverage vary between universities.

TRY IT

Augmenting a Digit Image

A handwritten digit recognizer has few examples; making new ones by rotating and shearing each image (data augmentation) means multiplying every pixel by a 2×2 matrix. Change the matrix and watch the "7".



a = **1.00**

b = **0.60**

c = **0.00**

d = **1.00**

Rotate 30° | Mirror | **Shear** | Stretch | Squash | Identity

Transform: **shear**
det = **1.00**: area ×1.00
Eigenvalues: **1.00** and **1.00** (green dashed lines keep their direction)

The determinant tells how much area is scaled; eigenvectors give the directions the map does not bend. The same eigenvector idea finds the directions of greatest spread in PCA.

CONTEXT

Prerequisites

Courses worth knowing before this one, and why they are needed.

Course	Why it is needed
High school mathematics	solving equations, functions and the coordinate plane
Programming	arrays and loops: trying matrix operations in code

CONTEXT

Are You Ready for This Course?

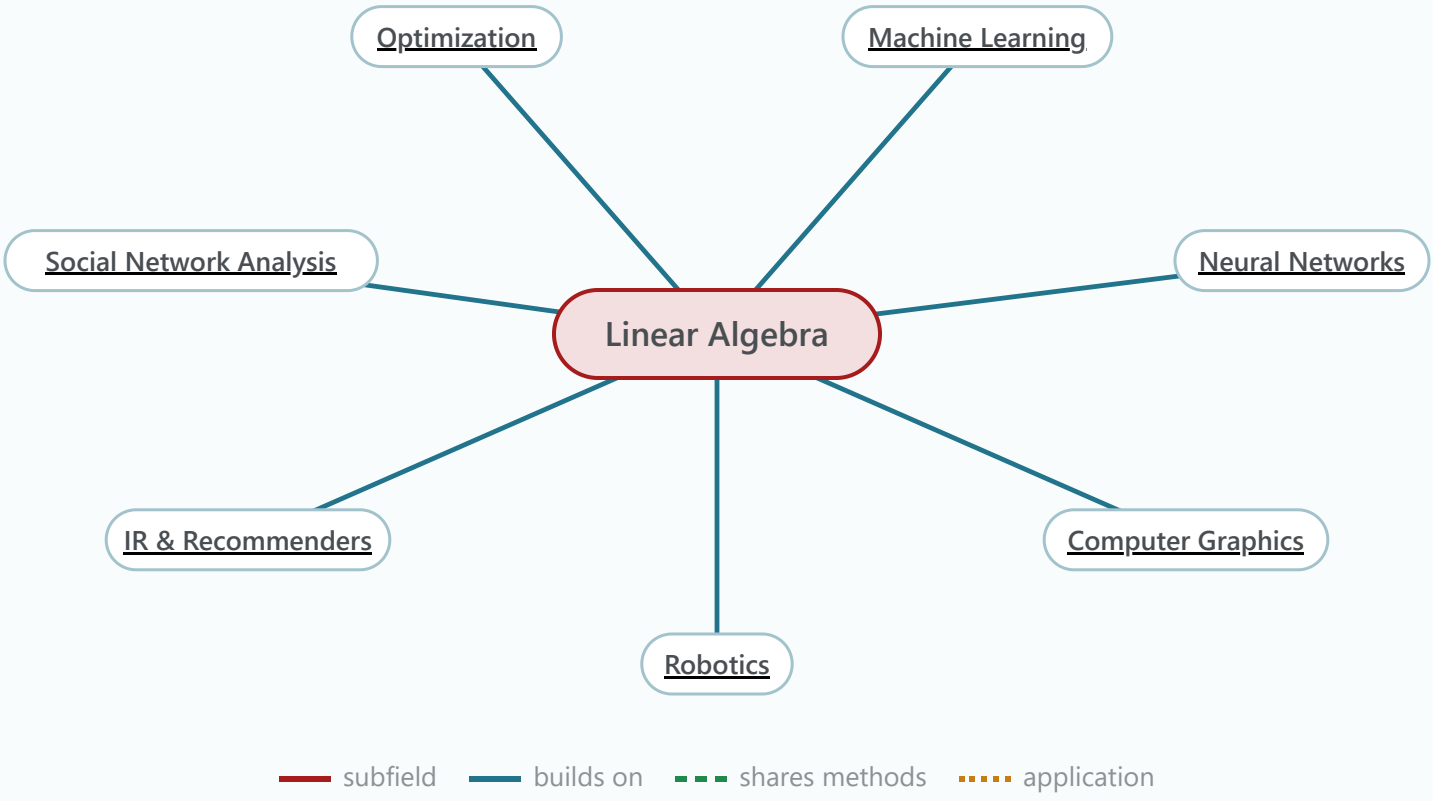
Ask yourself; for every "no", the right-hand column shows where to start.

Ask yourself	If not, start with
Can you solve a system of two equations in two unknowns by elimination?	Systems of equations from high school maths
Can you find the distance between two points in the plane?	Coordinate geometry from high school maths
Can you find an angle of a right triangle with sine and cosine?	High school trigonometry
Can you multiply two Python lists element by element in a loop and sum them?	An intro programming course, then a short NumPy tutorial

If you can say "yes" to most of these, you are ready; one or two gaps can be closed during the semester.

CONTEXT

Neighbouring Courses



Overlapping Courses

Relation	Course	Shared ground
builds on	Machine Learning builds on this course	feature vectors, matrix multiplication, PCA
builds on	Artificial Neural Networks builds on this course	weight matrices, vectorized computation
builds on	Computer Graphics builds on this course	transformation matrices, projection
builds on	Robotics and Robot Vision builds on this course	coordinate transforms, kinematics
builds on	Information Retrieval and Recommender Systems builds on this course	vector space model, matrix factorization
builds on	Social Network Analysis builds on this course	adjacency matrix, spectral methods
builds on	Optimization builds on this course	gradient, Hessian, convexity

Where It Differs (1/2)

Criterion	Linear Algebra	Optimization	Probability & Statistics
Core question	How do vectors and matrices represent data and transformations?	How do we find the values that make an objective function best?	How is uncertainty measured, and how do we draw reliable conclusions from limited data?
Approach	vector spaces and transformations	mathematical model of objective and constraints	probability models and inference
Input	vectors, matrices	objective function and constraints	sample data
Output	solution, decomposition, transformation	optimal or near-optimal solution	estimate, confidence interval, test result
Success measure	accuracy, numerical stability	objective value, convergence rate	unbiasedness, power, significance
Math load	the theory itself	the theory itself	the theory itself
Data needs	none	none	a sample

- **Optimization:** Optimization searches for the best values; linear algebra supplies the gradients, Hessians and system solves used in that search.
- **Probability & Statistics:** Probability measures uncertainty; linear algebra represents and decomposes the data and the covariance matrix.

Where It Differs (2/2)

Criterion	Linear Algebra	Machine Learning	Computer Graphics
Core question	How do vectors and matrices represent data and transformations?	How does a program generalize from examples?	How is a realistic image produced from the description of a scene?
Approach	vector spaces and transformations	statistical learning	geometry and light simulation
Input	vectors, matrices	labeled or unlabeled examples	scene: models, lights, camera
Output	solution, decomposition, transformation	a model that makes predictions	image or animation
Success measure	accuracy, numerical stability	error on unseen data	realism, frame rate
Math load	the theory itself	high: linear algebra, probability, optimization	high: linear algebra, geometry
Data needs	none	moderate to large	none

- **Machine Learning:** ML learns a model from data; linear algebra is the language of the computation inside that model and learns nothing itself.
- **Computer Graphics:** Computer graphics applies 3D transformation matrices; this course gives the general theory of those matrices.

PRACTICE

Applications

Neural networks
Layer = matrix product plus activation.

PCA
Dimensionality reduction and visualization.

Recommender systems
Factorizing the user-item matrix.

Word embeddings
Similar meaning = nearby vectors.

PageRank
Dominant eigenvector of the link matrix.

Image processing
Transforms, compression and filters.

PRACTICE

Risks and Ethics

Ill-conditioning

With a large condition number, a tiny input error is magnified many times; the answer fills up with meaningless digits.

The normal equations trap

Forming A^TA for least squares squares the condition number; solving with QR or SVD is safer.

Silent shape errors

NumPy broadcasting combines mismatched shapes without an error; the result is wrong but looks fine.

Misreading PCA

On unscaled features PCA favours the variable with the largest units; a component is not always a meaningful factor.

In depth: Explainable AI and AI Ethics.

PRACTICE

Tools and Datasets

Tools and libraries

NumPy | SciPy (scipy.linalg) | PyTorch | MATLAB | GNU Octave | LAPACK and BLAS

Classic example problems

Line fitting by least squares | Eigenfaces | PageRank eigenvector | Image compression with SVD | Netflix Prize matrix factorization | Markov chain steady state

Matrix operations are not coded by hand: they run on **LAPACK and BLAS**, the product of decades of work, which NumPy and PyTorch also call.

PRACTICE

Common Misconceptions

Common belief	In fact
Linear algebra is just solving equations.	Its real power is understanding transformations and decompositions: PCA, SVD and embeddings are built on them.
Computing the inverse and multiplying is the right way.	In practice you do not invert; you solve with LU, QR or Cholesky, which is faster and more stable.
High-dimensional space behaves like 3D.	In high dimensions points sit at similar distances from each other; intuition misleads (curse of dimensionality).
With libraries around, theory is unnecessary.	Rank deficiency, ill-conditioning and shape errors make sense only if you know the theory.

NEXT STEPS

After This Course

Courses that build on or extend this one

Machine Learning | Artificial Neural Networks | Computer Graphics | Robotics and Robot Vision | Information Retrieval and Recommender Systems | Social Network Analysis | Optimization

Related careers

Machine learning engineer | Data scientist | Computer graphics developer | Numerical computing specialist

Linear algebra is a prerequisite for almost every course in the guide. After it you can move comfortably to optimization, machine learning, neural networks and computer graphics. Numerical linear algebra and GPU programming teach how to do the same computations fast at scale.

NEXT STEPS

Further Reading

- Strang, *Introduction to Linear Algebra*, 6th ed., Wellesley-Cambridge Press, 2023.
- Strang, *Linear Algebra and Learning from Data*, Wellesley-Cambridge Press, 2019.
- Trefethen and Bau, *Numerical Linear Algebra*, SIAM, 1997.
- Deisenroth, Faisal and Ong, *Mathematics for Machine Learning*, Cambridge University Press, 2020.

The End

Chapter 31: Linear Algebra

← Previous chapter · Map of the field · Next chapter →