CS111 Notes

Spring 2025

UCSB

Preface

These notes have been prepared for CS111: Introduction to Computational Science. They highlight the pivotal role that computers play in solving complex mathematical models and demonstrate how advanced mathematical techniques can be harnessed to tackle real-world problems.

These notes draw inspiration from the CS192A: Matrix Analysis and Computation lecture notes by Prof. Shivkumar Chandrasekaran, a course that covers graduate-level linear algebra for Engineering and Computer Science at UCSB. I have incorporated many of the concepts from those notes and adapted them to be more accessible for upper-undergraduate students, with a particular emphasis on practical applications in linear algebra.

The primary goal of these notes is to bridge the gap between rigorous mathematical theory and computational practice. By emphasizing real-world applications in areas such as machine learning and bioinformatics, we illustrate how fundamental mathematical ideas underpin the algorithms and models that drive modern computational science.

For those seeking further insight and a deeper dive into scientific computing, I highly recommend exploring the excellent notes by David Bindel.

Contents

1 Linear Algebra Review

1.1 Vectors and Matrices

Scientific computing involves numerically approximating mathematical models. To build these approximations, we first define the fundamental building blocks—scalars, vectors, and matrices—which later serve as the groundwork for more advanced concepts in linear algebra and beyond.

$\mathbb{R}$ The set of all real numbers. Real numbers form the backbone of most numerical computations.

$\mathbb{Z}$ The set of all integers. Integers are useful when dealing with countable or discrete quantities.

$\mathbb{C}$ The set of all complex numbers. Although our focus will primarily be on real scalars, complex numbers are essential in many advanced topics, such as signal processing and quantum mechanics.

Scalars Elements of number spaces (e.g., $\mathbb{R}$, $\mathbb{C}$, $\mathbb{Z}$) are called **scalars**. They represent single quantities and serve as the fundamental units from which vectors and matrices are constructed.

Scalars are the atoms of numerical computation. In many advanced fields, such as functional analysis or numerical linear algebra, understanding scalar properties is crucial when extending these ideas to infinite-dimensional spaces or complex systems.

Vectors A **vector** is a one-dimensional array of scalars. Vectors can represent points, directions, or even more abstract quantities in a vector space.

NOTE A vector can be arranged as a row or a column. Although both representations carry the same information, in advanced linear algebra, the distinction is critical when discussing dual spaces and linear maps.

Examples

$$\begin{pmatrix} 1 & 2 & 3 \end{pmatrix}, \quad \begin{pmatrix} 1 \\ 2 \\ 3 \end{pmatrix}, \quad \begin{pmatrix} 1+2i \\ 7i \\ 3 \end{pmatrix}, \ldots$$

Just as scalars reside in number spaces like $\mathbb{R}$ and $\mathbb{C}$, vectors inhabit **vector spaces**. Later chapters will formalize the notion of a vector space, but for now, consider it simply as a collection of vectors that share common properties.

$\mathbb{R}^n$ The set of all real vectors of size n. When we write $x \in \mathbb{R}^n$, we mean that x is a vector with n real entries. Typically, we express vectors in one of two forms:

$$\boldsymbol{x} \in \mathbb{R}^{n\times 1} = \begin{pmatrix} x_1 \\ \vdots \\ x_n \end{pmatrix}, \quad \text{(column vector)}$$

$$\boldsymbol{x} \in \mathbb{R}^{1\times n} = \begin{pmatrix} x_1 & \cdots & x_n \end{pmatrix}, \quad \text{(row vector)}$$

For these notes, we adopt the column vector form as our default.

NOTE In these notes, we focus on finite-dimensional vector spaces. In more advanced studies, infinite-dimensional spaces—central to functional analysis—play a major role in understanding differential equations, quantum mechanics, and more.

Matrices A **matrix** is a two-dimensional (or higher-dimensional) array of scalars. A matrix $\mathbf{A} \in \mathbb{R}^{m\times n}$ is written as

$$\mathbf{A} = \begin{pmatrix} a_{11} & \cdots & a_{1n} \\ \vdots & \ddots & \vdots \\ a_{m1} & \cdots & a_{mn} \end{pmatrix}.$$

Matrices are used to represent linear transformations, systems of equations, and more complex data structures.

Examples We will mainly focus on two-dimensional matrices in these notes:

$$\begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{pmatrix}, \quad \begin{pmatrix} 1 & 2 \\ 4 & 5 \\ 6 & 7 \end{pmatrix}, \quad \text{etc.}$$

NOTE Matrices can also be thought of as a vector whose entries are themselves vectors. For example,

$$\mathbf{A} = \begin{pmatrix} \text{——} & \boldsymbol{r}_1 & \text{——} \\ & \vdots & \\ \text{——} & \boldsymbol{r}_n & \text{——} \end{pmatrix} = \begin{pmatrix} | & & | \\ \boldsymbol{c}_1 & \cdots & \boldsymbol{c}_n \\ | & & | \end{pmatrix},$$

where:

1. $r_i = \mathbf{A}[i]$ is the *row vector* representing the i^{th} row of $\mathbf{A}$.
2. $c_i = \mathbf{A}[:, i]$ is the *column vector* representing the i^{th} column of $\mathbf{A}$.

This perspective becomes increasingly important when exploring advanced topics such as block matrices, tensor decompositions, and matrix factorization methods.

Now that we have covered the basic definitions and notations for scalars, vectors, and matrices, we will soon explore their arithmetic operations. These operations not only form the basis of numerical computing but also provide a glimpse into deeper linear algebraic structures encountered in graduate-level studies.

1.2 Matrix Arithmetic

We are all familiar with scalar arithmetic, including operations like addition, subtraction, and multiplication, as well as fundamental properties such as associativity and distributivity. In the following sections, we extend these familiar concepts to vectors and matrices, exploring how these operations are defined and applied in higher dimensions.

1.2.1 Adding and Subtracting Matrices

Matrix addition is performed element-wise. Note that matrices can only be added (or subtracted) if they have the same dimensions.

Matrix Sum Let $\mathbf{A}, \mathbf{B} \in \mathbb{R}^{m \times n}$ be matrices. Their sum is defined as

$$\mathbf{A} + \mathbf{B} = \begin{pmatrix} a_{11} & \cdots & a_{1n} \\ \vdots & \ddots & \vdots \\ a_{n1} & \cdots & a_{nn} \end{pmatrix} + \begin{pmatrix} b_{11} & \cdots & b_{1n} \\ \vdots & \ddots & \vdots \\ b_{n1} & \cdots & b_{nn} \end{pmatrix} = \begin{pmatrix} (a+b)_{11} & \cdots & (a+b)_{1n} \\ \vdots & \ddots & \vdots \\ (a+b)_{n1} & \cdots & (a+b)_{nn} \end{pmatrix},$$

where each entry satisfies $(a+b)_{ij} = a_{ij} + b_{ij}$ for $1 \leq i \leq m$ and $1 \leq j \leq n$.

NOTE For two matrices $\mathbf{A} \in \mathbb{R}^{m \times n}$ and $\mathbf{B} \in \mathbb{R}^{p \times q}$, the sum $\mathbf{A} + \mathbf{B}$ is defined if and only if $m = p$ and $n = q$.

Exercise Prove that matrix addition is commutative and associative.

1.2.2 Multiplication

Matrix operations include multiplication by a scalar and matrix multiplication. In the following, we review these operations.

Scalar Multiplication Let $\mathbf{A} \in \mathbb{R}^{m \times n}$ and $c \in \mathbb{R}$. Then scalar multiplication is defined as

$$c\mathbf{A} = c \begin{pmatrix} a_{11} & \cdots & a_{1n} \\ \vdots & \ddots & \vdots \\ a_{n1} & \cdots & a_{nn} \end{pmatrix} = \begin{pmatrix} ca_{11} & \cdots & ca_{1n} \\ \vdots & \ddots & \vdots \\ ca_{n1} & \cdots & ca_{nn} \end{pmatrix},$$

where each entry satisfies $(ca)_{ij} = c \cdot a_{ij}$.

Exercise Using the definitions of matrix sum and scalar multiplication, define matrix subtraction.

Hint: Express $\mathbf{A} - \mathbf{B}$ as $\mathbf{A} + (-1)\mathbf{B}$.

Dot Product

For vectors $\boldsymbol{a}, \boldsymbol{b} \in \mathbb{R}^n$, the dot product (or inner product) is defined as

$$\boldsymbol{a} \cdot \boldsymbol{b} = \boldsymbol{a}^t \boldsymbol{b} = \begin{pmatrix} a_1 & \cdots & a_n \end{pmatrix} \begin{pmatrix} b_1 \\ \vdots \\ b_n \end{pmatrix} = \sum_{i=1}^{n} a_i b_i. \tag{1}$$

Matrix Product Let $\mathbf{A} \in \mathbb{R}^{m \times n}$ and $\mathbf{B} \in \mathbb{R}^{n \times p}$. Their product $\mathbf{AB}$ is an $m \times p$ matrix defined by

$$(\mathbf{AB})_{ij} = \sum_{k=1}^{n} a_{ik} b_{kj},$$

or equivalently, if we consider the rows of $\mathbf{A}$ as vectors $\boldsymbol{a}_i$ and the columns of $\mathbf{B}$ as vectors $\boldsymbol{b}_j$, then

$$\mathbf{AB} = \begin{pmatrix} \boldsymbol{a}_1 \cdot \boldsymbol{b}_1 & \cdots & \boldsymbol{a}_1 \cdot \boldsymbol{b}_p \\ \vdots & \ddots & \vdots \\ \boldsymbol{a}_m \cdot \boldsymbol{b}_1 & \cdots & \boldsymbol{a}_m \cdot \boldsymbol{b}_p \end{pmatrix}.$$

NOTE Matrix multiplication is defined only when the number of columns of $\mathbf{A}$ equals the number of rows of $\mathbf{B}$. Moreover, it is associative and distributive over addition, but in general it is not commutative; that is, $\mathbf{AB} \neq \mathbf{BA}$ for most matrices.

Exercise Prove that if $\mathbf{A} \in \mathbb{R}^{m \times n}$ and $\mathbf{B} \in \mathbb{R}^{p \times q}$, then the product $\mathbf{AB}$ is defined if and only if $n = p$.

Exercise Prove the distributive property of matrix multiplication: show that $\mathbf{A}(\mathbf{B}+\mathbf{C}) = \mathbf{AB} + \mathbf{AC}$.

Exercise Prove that scalar multiplication distributes over matrix addition, i.e., $\alpha(\mathbf{A}+\mathbf{B}) = \alpha\mathbf{A} + \alpha\mathbf{B}$.

Exercise Investigate the statement: "$\mathbf{AB} = \mathbf{BA}$ if and only if $\mathbf{A} = \mathbf{B}$." Prove or provide a counterexample.

1.2.3 Matrices as Vectors and Linear Maps

Many properties of scalar arithmetic extend naturally to both vectors and matrices. In fact, matrices can be viewed as elements of a higher-dimensional vector space, and matrix multiplication can be interpreted as the composition of linear transformations. While the notations for vectors and matrices are similar, their roles in linear algebra differ:

- Vectors typically represent points or directions in space.
- Matrices often represent linear maps or transformations between vector spaces.

This conceptual distinction becomes crucial when discussing topics such as vector spaces, eigenvalues, and linear transformations.

1.2.4 Matrix Inverse

Matrix Inverse For a square matrix $\mathbf{A} \in \mathbb{R}^{n \times n}$, the inverse of $\mathbf{A}$, denoted $\mathbf{A}^{-1}$, is defined as the unique matrix satisfying

$$\mathbf{A}\mathbf{A}^{-1} = \mathbf{A}^{-1}\mathbf{A} = \mathbf{I},$$

where $\mathbf{I}$ is the $n \times n$ identity matrix. A matrix that has an inverse is called *invertible* or *nonsingular*.

NOTE A square matrix $\mathbf{A}$ is invertible if and only if $\det(\mathbf{A}) \neq 0$, which also implies that $\mathbf{A}$ has full rank.

Exercise Prove that if $\mathbf{A}$ is invertible, then its inverse is unique.

Left Inverse For a matrix $\mathbf{A} \in \mathbb{R}^{m \times n}$ with $m \geq n$ and full column rank, a **left inverse** $\mathbf{A}^{-L}$ is an $n \times m$ matrix satisfying

$$\mathbf{A}^{-L}\mathbf{A} =_n .$$

Right Inverse For a matrix $\mathbf{A} \in \mathbb{R}^{m \times n}$ with $m \leq n$ and full row rank, a **right inverse** $\mathbf{A}^{-R}$ is an $n \times m$ matrix satisfying

$$\mathbf{A}\mathbf{A}^{-R} =_m .$$

In the case where $\mathbf{A}$ is rectangular or rank-deficient, the Moore-Penrose pseudoinverse $\mathbf{A}^{+}$ is defined uniquely to satisfy four specific properties, yielding the best least-squares solution to $\mathbf{A}\boldsymbol{x} = \boldsymbol{b}$.

Exercise Given a rank-r matrix $\mathbf{A} \in \mathbb{R}^{m \times n}$ with SVD $\mathbf{A} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^{T}$, derive the explicit expression for its Moore-Penrose pseudoinverse

$$\mathbf{A}^{+} = \mathbf{V}\mathbf{\Sigma}^{+}\mathbf{U}^{T},$$

where $\mathbf{\Sigma}^{+}$ is formed by taking the reciprocal of each nonzero singular value and transposing the matrix. Discuss how $\mathbf{A}^{+}$ serves as a left inverse when $\mathbf{A}$ is tall and as a right inverse when $\mathbf{A}$ is wide.

1.3 Vector Spaces

Vector Spaces A **vector space** $\mathcal{V}$ is a collection of *vectors* along with two operations—vector addition and scalar multiplication—that satisfy the following axioms. For any vectors $\boldsymbol{x}, \boldsymbol{y}, \in \mathcal{V}$ and any scalars $\alpha, \beta \in \mathbb{R}$, the axioms are:

1. **Commutativity:** $\boldsymbol{x} + \boldsymbol{y} = \boldsymbol{y} + \boldsymbol{x}$.
2. **Associativity:** $(\boldsymbol{x} + \boldsymbol{y}) + = \boldsymbol{x} + (\boldsymbol{y} +)$.
3. **Additive Identity:** There exists a zero vector $\mathbf{0} \in \mathcal{V}$ such that $\boldsymbol{x} + \mathbf{0} = \boldsymbol{x}$.
4. **Additive Inverse:** For every $\boldsymbol{x} \in \mathcal{V}$, there exists a vector $-\boldsymbol{x} \in \mathcal{V}$ with $\boldsymbol{x} + (-\boldsymbol{x}) = \mathbf{0}$.
5. **Distributivity:** $\alpha(\boldsymbol{x} + \boldsymbol{y}) = \alpha\boldsymbol{x} + \alpha\boldsymbol{y}$ and $(\alpha + \beta)\boldsymbol{x} = \alpha\boldsymbol{x} + \beta\boldsymbol{x}$.
6. **Scalar Multiplicative Identity:** $1 \cdot \boldsymbol{x} = \boldsymbol{x}$.

These axioms, familiar from $\mathbb{R}^n$, form the foundation for both finite and infinite-dimensional spaces.

NOTE Just like of a number space is a collection of numbers with a predefined set of rules (addition, subtraction etc.). A vector space is simply an higher dimensional abstraction to numberspaces.

Subspace A **subspace** of a vector space $\mathcal{V}$ is a subset $W \subseteq \mathcal{V}$ that is itself a vector space under the operations inherited from $\mathcal{V}$. In particular, W must satisfy:

- The zero vector of $\mathcal{V}$ is in W.
- W is closed under vector addition: for all $\boldsymbol{x}, \boldsymbol{y} \in W$, $\boldsymbol{x} + \boldsymbol{y} \in W$.
- W is closed under scalar multiplication: for all $\boldsymbol{x} \in W$ and all $\alpha \in \mathbb{R}$, $\alpha\boldsymbol{x} \in W$.

Exercise Verify that the set $W = \{(x, y) \in \mathbb{R}^2 : y = 2x\}$ is a subspace of $\mathbb{R}^2$. What is its dimension?

Linear Combination A **linear combination** of vectors $\{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n\}$ is any vector $\boldsymbol{y}$ that can be written as

$$\boldsymbol{y} = \alpha_1\boldsymbol{x}_1 + \alpha_2\boldsymbol{x}_2 + \cdots + \alpha_n\boldsymbol{x}_n,$$

where $\alpha_1, \ldots, \alpha_n \in \mathbb{R}$.

NOTE View each vector as an ingredient; a linear combination mixes these ingredients in various proportions to produce new vectors.

Linear Dependence A set of vectors $\{\boldsymbol{x}_1, \dots, \boldsymbol{x}_n\}$ is **linearly dependent** if there exist scalars $\alpha_1, \dots, \alpha_n$, not all zero, such that

$$\alpha_1 \boldsymbol{x}_1 + \alpha_2 \boldsymbol{x}_2 + \cdots + \alpha_n \boldsymbol{x}_n = \mathbf{0}.$$

If the only solution is $\alpha_1 = \cdots = \alpha_n = 0$, the vectors are **linearly independent**.

NOTE Linear dependence indicates redundancy; one or more vectors in the set can be expressed as a combination of the others.

Linear Span The **linear span** of a set of vectors $\{\boldsymbol{x}_1, \dots, \boldsymbol{x}_n\}$, denoted $\operatorname{span}\{\boldsymbol{x}_1, \dots, \boldsymbol{x}_n\}$, is the set of all linear combinations of these vectors:

$$\operatorname{span}\{\boldsymbol{x}_1, \dots, \boldsymbol{x}_n\} = \{\alpha_1 \boldsymbol{x}_1 + \cdots + \alpha_n \boldsymbol{x}_n : \alpha_i \in \mathbb{R}\}.$$

This is the smallest subspace of $\mathcal{V}$ that contains the given vectors.

Exercise Determine a basis for $\operatorname{span}\{(1,2), (3,4)\}$ in $\mathbb{R}^2$. Are the vectors $(1,2)$ and $(3,4)$ linearly independent?

Basis A **basis** for a vector space $\mathcal{V}$ is a set of vectors that is both linearly independent and spanning. Every vector in $\mathcal{V}$ can be uniquely expressed as a linear combination of the basis vectors. The number of basis vectors is the **dimension** of $\mathcal{V}$.

Exercise Find a basis for the subspace $W = \{(x, y) \in \mathbb{R}^2 : y = 3x\}$. What is the dimension of W?

Orthogonal Subspace (Orthogonal Complement) Given a vector space $\mathcal{V}$ equipped with an inner product $\langle \cdot, \cdot \rangle$ and a subspace $W \subseteq \mathcal{V}$, the **orthogonal complement** of W, denoted $W^\perp$, is defined as

$$W^\perp = \{ \in \mathcal{V} : \langle , \rangle = 0 \text{ for all } \in W\}.$$

$W^\perp$ is itself a subspace of $\mathcal{V}$.

NOTE $W^\perp$ consists of all vectors that are "perpendicular" to every vector in W. In $\mathbb{R}^2$, if W is a line through the origin, then $W^\perp$ is the line orthogonal to W.

Exercise For the subspace $W = \{(x, 2x) \in \mathbb{R}^2 : x \in \mathbb{R}\}$, find the orthogonal complement $W^\perp$. Hint: Determine all vectors $(a, b) \in \mathbb{R}^2$ satisfying $ax + b(2x) = 0$ for all x.

Direct Sum If W_1 and W_2 are subspaces of $\mathcal{V}$ such that every vector $\in \mathcal{V}$ can be uniquely written as $=_1 +_2$ with $_1 \in W_1$ and $_2 \in W_2$, then $\mathcal{V}$ is the **direct sum** of W_1 and W_2, denoted by

$$\mathcal{V} = W_1 \oplus W_2.$$

NOTE The direct sum decomposition splits the vector space into two non-overlapping parts. For instance, in $\mathbb{R}^2$, any vector can be uniquely expressed as the sum of a vector from a given line W and a vector from its orthogonal complement $W^\perp$.

Exercise Show that $\mathbb{R}^2 = W \oplus W^\perp$ for the subspace $W = \{(x, 2x) \in \mathbb{R}^2 : x \in \mathbb{R}\}$. Provide a detailed proof.

Orthogonal Projection Given a subspace W of a vector space $\mathcal{V}$ with inner product, the **orthogonal projection** of a vector $\in \mathcal{V}$ onto W is the unique vector $\in W$ such that $- \in W^\perp$. This projection is denoted by $\mathrm{proj}_W()$.

NOTE The orthogonal projection of onto W is the "shadow" of on the subspace W when light is cast perpendicular to W.

Exercise Let W be the subspace of $\mathbb{R}^2$ spanned by $(1, 2)$. Compute the orthogonal projection of the vector $(3, 4)$ onto W.

1.4 Linearity

Linear Systems A **system of linear equations** is a collection of equations in which each equation is a linear combination of the unknown variables. Such a system can be written in the form

$$\begin{array}{rcl} a_{11}x_1 + a_{12}x_2 + & \cdots & + a_{1n}x_n = b_1, \\ a_{21}x_1 + a_{22}x_2 + & \cdots & + a_{2n}x_n = b_2, \\ & \vdots & \\ a_{m1}x_1 + a_{m2}x_2 + & \cdots & + a_{mn}x_n = b_m, \end{array}$$

where each a_{ij} (with $1 \le i \le m$ and $1 \le j \le n$) is a known constant and $x_1, \dots, x_n$ are the unknowns. The objective is to find values for the x_j that simultaneously satisfy all the equations. In matrix-vector form, this system is succinctly expressed as

$$\mathbf{A}\boldsymbol{x} = \boldsymbol{b},$$

where

$$\mathbf{A} = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mn} \end{bmatrix}, \quad \boldsymbol{x} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}, \quad \boldsymbol{b} = \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_m \end{bmatrix}.$$

NOTE The mapping $T : \mathbb{R}^n \to \mathbb{R}^m$ defined by $T(\boldsymbol{x}) = \mathbf{A}\boldsymbol{x}$ is a linear transformation. That is, for any vectors $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^n$ and scalar $c \in \mathbb{R}$, we have:

$$T(\boldsymbol{x} + \boldsymbol{y}) = \mathbf{A}(\boldsymbol{x} + \boldsymbol{y}) = \mathbf{A}\boldsymbol{x} + \mathbf{A}\boldsymbol{y} = T(\boldsymbol{x}) + T(\boldsymbol{y}),$$

$$T(c\boldsymbol{x}) = \mathbf{A}(c\boldsymbol{x}) = c\,\mathbf{A}\boldsymbol{x} = c\,T(\boldsymbol{x}).$$

These properties, known as additivity and homogeneity, are the defining features of linear maps.
Existence and Uniqueness For a square system (i.e., when $m = n$), the linear system

$$\mathbf{A}\boldsymbol{x} = \boldsymbol{b}$$

has a unique solution if and only if the coefficient matrix $\mathbf{A}$ is invertible, which is equivalent to $\det(\mathbf{A}) \neq 0$. If $\mathbf{A}$ is singular (i.e., noninvertible, $\det(\mathbf{A}) = 0$), then the system either has no solution or has infinitely many solutions. In the latter case, the set of solutions forms an affine subspace of $\mathbb{R}^n$.

Homogeneous Systems A *homogeneous system* is a special case of a linear system where the constant vector is zero:

$$\mathbf{A}\boldsymbol{x} = \mathbf{0}.$$

Such a system always has at least the trivial solution $\boldsymbol{x} = \mathbf{0}$. Moreover, if there exists any nontrivial solution, then the complete set of solutions forms a linear subspace of $\mathbb{R}^n$, called the null space (or kernel) of $\mathbf{A}$.

Superposition Principle For a homogeneous linear system $\mathbf{A}\boldsymbol{x} = \mathbf{0}$, if $\boldsymbol{x}_1$ and $\boldsymbol{x}_2$ are solutions, then any linear combination

$$c_1\boldsymbol{x}_1 + c_2\boldsymbol{x}_2, \quad \text{with } c_1, c_2 \in \mathbb{R},$$

is also a solution. This principle, known as the **superposition principle**, reflects the fact that the set of all solutions to a homogeneous system forms a vector subspace of $\mathbb{R}^n$.

Examples Consider the system

$$\begin{aligned} 2x + 3y &= 5, \\ x - 4y &= -2. \end{aligned}$$

In matrix form, it is expressed as

$$\mathbf{A}\boldsymbol{x} = \boldsymbol{b} \quad \text{with} \quad \mathbf{A} = \begin{bmatrix} 2 & 3 \\ 1 & -4 \end{bmatrix}, \quad \boldsymbol{x} = \begin{bmatrix} x \\ y \end{bmatrix}, \quad \boldsymbol{b} = \begin{bmatrix} 5 \\ -2 \end{bmatrix}.$$

The determinant of the coefficient matrix is calculated as

$$\det \begin{bmatrix} 2 & 3 \\ 1 & -4 \end{bmatrix} = 2(-4) - 3(1) = -8 - 3 = -11 \neq 0.$$

Since the determinant is nonzero, $\mathbf{A}$ is invertible, and by the Existence and Uniqueness Theorem, the system has a unique solution.

NOTE For non-square systems or singular matrices, the solution set can be more intricate. For example, when $\mathbf{A}$ is not of full rank, the homogeneous system has infinitely many solutions forming a subspace whose dimension is given by the nullity of $\mathbf{A}$ (as described by the Rank-Nullity Theorem). Understanding these scenarios is essential in many applications, such as in solving least squares problems.

2 Numerical Linear Algebra

Numerical Linear Algebra forms the core of scientific computing, enabling the efficient and stable solution of linear systems, the decomposition of large-scale matrices, and the extraction of essential features—such as eigenvalues and singular values—from complex data. In this section, we present a detailed exploration of the central ideas and methods in numerical linear algebra. Building on fundamental linear algebraic principles, we extend these ideas to cover algorithmic and computational considerations. We begin by discussing the transition from analytical to numerical methods, then revisit LU Decomposition and its pivoting variants, proceed through other matrix factorizations (Cholesky, QR), and conclude with the Eigenvalue and Singular Value Decompositions.

2.1 From Analytical to Numerical

Analytical Solutions In an idealized mathematical setting, a square and invertible matrix $\mathbf{A} \in \mathbb{R}^{n\times n}$ admits a closed-form solution for the linear system $\mathbf{A}\boldsymbol{x} = \boldsymbol{b}$, given by

$$\boldsymbol{x} = \mathbf{A}^{-1}\boldsymbol{b}.$$

This expression is exact and demonstrates the theoretical uniqueness of the solution provided that $\mathbf{A}^{-1}$ exists.

Limitations of Analytical Methods Despite the clarity of the analytical expression $\boldsymbol{x} = \mathbf{A}^{-1}\boldsymbol{b}$, computing $\mathbf{A}^{-1}$ directly is rarely practical. For large n or when $\mathbf{A}$ is nearly singular, direct inversion is computationally expensive (typically $O(n^3)$ operations) and can be numerically unstable due to the propagation of round-off errors.

Numerical Methods Numerical linear algebra circumvents direct inversion by employing alternative strategies. Instead of forming $\mathbf{A}^{-1}$ explicitly, one uses matrix factorizations (such as LU, QR, or SVD) or iterative schemes (such as Jacobi, Gauss-Seidel, or Conjugate Gradient) to approximate the solution $\hat{\boldsymbol{x}}$ of $\mathbf{A}\boldsymbol{x} = \boldsymbol{b}$. These methods are designed to exploit the structure or sparsity of $\mathbf{A}$ and to control the amplification of numerical errors.

Iterative Methods Iterative methods generate a sequence of approximations $\{\boldsymbol{x}^{(k)}\}_{k=0}^{\infty}$ starting from an initial guess $\boldsymbol{x}^{(0)}$. Under suitable conditions, this sequence converges to the true solution $\boldsymbol{x}$. The convergence rate and overall efficiency of iterative methods depend critically on properties of $\mathbf{A}$, such as its spectral radius and, notably, its condition number.

Condition Number The condition number of $\mathbf{A}$ with respect to a chosen matrix norm $\|\cdot\|$ is defined as

$$\kappa(\mathbf{A}) = \|\mathbf{A}\|\,\|\mathbf{A}^{-1}\|.$$

This scalar quantifies the sensitivity of the solution $\boldsymbol{x}$ to perturbations in $\mathbf{A}$ or $\boldsymbol{b}$. A large $\kappa(\mathbf{A})$ indicates that the system is *ill-conditioned*, meaning that even minute errors in the data may be significantly amplified in the computed solution.

NOTE For example, consider the matrix

$$\mathbf{A} = \begin{pmatrix} 1 & 0 \\ 0 & 10^{-6} \end{pmatrix}.$$

Its singular values are 1 and 10^{-6}, yielding $\kappa_2(\mathbf{A}) = 10^6$. Such a high condition number suggests that even a relative error of 10^{-8} in the data could be magnified to a relative error of approximately 10^{-2} in the solution.

Exercise Take the matrix

$$\mathbf{A} = \begin{pmatrix} 1 & 0 \\ 0 & 10^{-6} \end{pmatrix},$$

and compute its condition number using the 2-norm. Then, perturb the right-hand side vector $\boldsymbol{b}$ by a small relative error (e.g., add 10^{-8} times a random vector) and solve for $\boldsymbol{x}$. Compare the relative error in $\boldsymbol{x}$ with the product $\kappa(\mathbf{A})$ times the relative error in $\boldsymbol{b}$. Discuss your findings and explore other examples of matrices with high condition numbers.

2.2 Floating Point Error

Absolute and Relative Error

When a real number A is approximated by a computed value $\hat{A}$, the **absolute error** is defined as

$$e = \hat{A} - A.$$

The **relative error** is given by

$$\varepsilon = \frac{\hat{A} - A}{A},$$

so that the computed value can be expressed as

$$\hat{A} = A + e \quad \text{or} \quad \hat{A} = A(1 + \varepsilon).$$

Relative error is dimensionless and generally more informative than absolute error, especially when the scale of A is significant.

Exercise Consider approximating $\sqrt{175}$ by 13. Compute both the absolute and relative error of this approximation. How does the relative error provide insight into the accuracy independent of the scale of $\sqrt{175}$?

Rounding Error

Computers represent real numbers using floating point arithmetic, which approximates real numbers by a finite number of bits. Let $\mathrm{fl}(x)$ denote the floating point representation of a real number x. The **rounding error** is defined as

$$\mathrm{fl}(x) - x,$$

and when normalized, the relative rounding error satisfies

$$\left| \frac{\mathrm{fl}(x) - x}{x} \right| \leq \varepsilon_{\text{mach}},$$

where $\varepsilon_{\text{mach}}$ (machine epsilon) is the upper bound on the relative error due to rounding.

NOTE For example, in IEEE single precision, $\varepsilon_{\text{mach}} \approx 6 \times 10^{-8}$; in double precision, $\varepsilon_{\text{mach}} \approx 10^{-16}$. These values indicate the smallest relative difference that can be distinguished by the floating point format.

Exercise Determine the machine epsilon for IEEE double precision arithmetic by considering the distance between 1 and the next representable floating point number. Explain why this value is significant in numerical computations.

Cancellation **Cancellation** occurs when subtracting two nearly equal numbers. Although the absolute error in each number may be small, their difference can lead to a significant loss of significant digits. For instance, if $A = B - C$ where B and C are close in value, then the computed difference $\hat{A} = \hat{B} - \hat{C}$ may have a relative error much larger than that of $\hat{B}$ or $\hat{C}$ individually. This phenomenon is known as **catastrophic cancellation**.

NOTE Consider $B \approx 2.38 \times 10^5$ and $C \approx 2.33 \times 10^5$. The leading digits cancel in the subtraction, leaving A with only one or two significant digits. Even if B and C are known to high precision, the cancellation drastically reduces the accuracy of A.

Exercise Take two numbers B and C with high precision and compute $A = B - C$. Experiment with values where the first several digits of B and C agree. Quantify the loss in relative accuracy for A compared to the original numbers. Suggest an alternative formulation that could reduce the effect of cancellation.

Floating Point Representation Floating point numbers are represented in computers using a format similar to scientific notation. A normalized floating point number has the form

$$x = (-1)^s \times m \times 2^e,$$

where:

- s is the sign bit,

- m is the mantissa (with $1 \leq m < 2$), and
- e is the exponent.

The number of bits allocated to the mantissa and exponent determines the precision and range of representable numbers. For example, IEEE single precision allocates 24 bits (including an implicit leading 1) to the mantissa and 8 bits to the exponent.

NOTE Not all real numbers can be exactly represented in this format. For example, numbers like 1/3 have a non-terminating binary expansion, which leads to a rounding error when stored as a floating point number.

Exercise Explain why the fraction 1/3 cannot be represented exactly in binary floating point format. Estimate the resulting rounding error in IEEE single precision.

Denormalized Numbers and Underflow When the magnitude of a real number is smaller than the smallest normalized floating point number (denoted $2^{e_{\min}}$), the number is represented as a **denormalized number**. In this format, the leading digit is assumed to be zero, allowing for a gradual underflow. Although denormalized numbers provide a way to represent very small values, they come with a reduced precision compared to normalized numbers.

NOTE The use of denormalized numbers avoids a sudden jump to zero when numbers become very small, thereby improving the accuracy of computations that involve underflow. However, operations involving denormalized numbers can be slower on some hardware.

Exercise Find the smallest positive normalized and denormalized numbers in IEEE single precision. Discuss the trade-offs in precision and performance when dealing with denormalized numbers.

Exceptions in Floating Point Arithmetic The IEEE standard also defines special values for representing exceptional cases: $+\infty$ and $-\infty$ are used to denote overflow, and NaN (Not a Number) is used to represent undefined or unrepresentable results, such as the result of 0/0 or $\sqrt{-1}$. These exceptions are crucial for robust numerical algorithms, ensuring that computations yield consistent and interpretable results even in edge cases.

NOTE When a floating point operation produces a result outside the representable range, the result is set to infinity, and further arithmetic operations propagate this infinity. Similarly, any operation involving NaN typically results in NaN, alerting the programmer to an undefined condition.

Exercise Consider performing the operation $\sqrt{-1}$ in a programming environment that follows the IEEE standard. What is the expected output? Discuss how the propagation of NaN in subsequent computations helps in debugging numerical algorithms.

Understanding floating point error is essential for the design and analysis of numerical algorithms. Rounding errors, cancellation, and the limitations of finite precision arithmetic are intrinsic to computer computations. By carefully modeling these errors and employing techniques to mitigate their effects, one can design algorithms that yield reliable and accurate results even in the presence of unavoidable numerical imperfections.

2.3 LU Factorization

For a square matrix $\mathbf{A} \in \mathbb{R}^{n\times n}$, the LU factorization expresses $\mathbf{A}$ as the product of a lower triangular matrix $\mathbf{L}$ (usually unit lower triangular) and an upper triangular matrix $\mathbf{U}$. This factorization simplifies the solution of $\mathbf{A}\boldsymbol{x} = \boldsymbol{b}$ by reducing it to two sequential triangular solves, which are computationally inexpensive.

LU Factorization

Let $\mathbf{A} \in \mathbb{R}^{n\times n}$ be a square matrix. An **LU Factorization** of $\mathbf{A}$ is a decomposition of the form

$$\mathbf{A} = \mathbf{L}\,\mathbf{U},$$

where $\mathbf{L} \in \mathbb{R}^{n\times n}$ is a lower triangular matrix with ones on its diagonal (a unit lower triangular matrix), and $\mathbf{U} \in \mathbb{R}^{n\times n}$ is an upper triangular matrix.

NOTE

The key idea behind LU factorization is to systematically eliminate the subdiagonal entries of $\mathbf{A}$ by applying Gaussian elimination. Instead of performing elimination separately for each system, one factors $\mathbf{A}$ into $\mathbf{L}$ and $\mathbf{U}$ once, so that solving $\mathbf{A}\boldsymbol{x} = \boldsymbol{b}$ reduces to solving $\mathbf{L}\boldsymbol{y} = \boldsymbol{b}$ (via forward substitution) and then $\mathbf{U}\boldsymbol{x} = \boldsymbol{y}$ (via backward substitution).
To illustrate, suppose

$$\mathbf{A} = \begin{pmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n1} & a_{n2} & \cdots & a_{nn} \end{pmatrix}.$$

The goal is to eliminate the entries below the main diagonal. For $i = 2, \ldots, n$, the first step is to eliminate a_{i1} by setting

$$\ell_{i1} = \frac{a_{i1}}{a_{11}},$$

and replacing row i with

$$\mathrm{row}_i \leftarrow \mathrm{row}_i - \ell_{i1}\,\mathrm{row}_1.$$

After this step, the first column of the modified matrix has zeros below a_{11}, and the new entries in row i are given by

$$a_{ij}^{(1)} = a_{ij} - \ell_{i1}\, a_{1j}, \quad j = 2, \dots, n.$$

This process is then repeated on the $(n-1) \times (n-1)$ submatrix obtained by deleting the first row and first column.

NOTE In the factorization, the multipliers ℓ_{ij} (for $i > j$) are stored in the corresponding entries of $\mathbf{L}$ (with $\ell_{jj} = 1$), and the resulting coefficients after elimination become the entries of $\mathbf{U}$. Thus, $\mathbf{L}$ captures the elimination steps, while $\mathbf{U}$ is the upper triangular matrix produced by Gaussian elimination.

Exercise Prove by induction on n that if all the leading principal minors of $\mathbf{A}$ are nonzero, then $\mathbf{A}$ has an LU factorization without row interchanges. (Hint: Start with $n = 1$ and assume the result for an $(n-1) \times (n-1)$ matrix; then show how to construct $\mathbf{L}$ and $\mathbf{U}$ for an $n \times n$ matrix.)
More explicitly, one may write

$$\mathbf{L} = \begin{pmatrix} 1 & 0 & \cdots & 0 \\ \ell_{21} & 1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ \ell_{n1} & \ell_{n2} & \cdots & 1 \end{pmatrix}, \quad \mathbf{U} = \begin{pmatrix} u_{11} & u_{12} & \cdots & u_{1n} \\ 0 & u_{22} & \cdots & u_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & u_{nn} \end{pmatrix}.$$

The entries of $\mathbf{U}$ and the multipliers ℓ_{ij} are determined by the recurrences:

$$u_{1j} = a_{1j}, \quad j = 1, \dots, n,$$

$$\ell_{i1} = \frac{a_{i1}}{u_{11}}, \quad i = 2, \dots, n,$$

and for $k = 2, \dots, n$:

$$u_{kj} = a_{kj} - \sum_{s=1}^{k-1} \ell_{ks}\, u_{sj}, \quad j = k, \dots, n,$$

$$\ell_{ik} = \frac{1}{u_{kk}}\Big(a_{ik} - \sum_{s=1}^{k-1} \ell_{is}\, u_{sk}\Big), \quad i = k+1, \dots, n.$$

NOTE These formulas summarize the Gaussian elimination process: at each step, the pivot u_{kk} is used to eliminate the entries below it, and the multipliers are recorded in $\mathbf{L}$. The computation of u_{kj} reflects the updated entries after previously performed eliminations.

Exercise

Consider the 3×3 matrix

$$\mathbf{A} = \begin{pmatrix} 2 & 3 & 1 \\ 4 & 7 & 3 \\ -2 & -3 & 1 \end{pmatrix}.$$

Perform the LU factorization manually by computing the multipliers ℓ_{ij} and the resulting entries of $\mathbf{U}$. Verify that $\mathbf{A} = \mathbf{L}\,\mathbf{U}$.

NOTE In practice, if a pivot u_{kk} is zero or nearly zero, row interchanges (pivoting) become necessary to ensure numerical stability. In such cases, the LU factorization is modified to the LUP factorization, where a permutation matrix is incorporated. However, when all leading principal minors are nonzero, as assumed here, an LU factorization without pivoting exists.

2.4 Pivoting and LUP Decomposition

Pivoting and LUP Decomposition When a matrix $\mathbf{A}$ contains zeros or very small entries in prospective pivot positions, row (or column) interchanges are necessary to maintain numerical stability. These interchanges are captured by a permutation matrix , yielding the factorization

$$\mathbf{A} = \mathbf{L}\mathbf{U}.$$

This variant is known as the *LUP Decomposition*, where $\mathbf{L}$ is unit lower triangular and $\mathbf{U}$ is upper triangular.

Partial pivoting involves interchanging rows to select the largest available pivot element in absolute value, thereby reducing the risk of division by small numbers and limiting error growth. Complete pivoting, which also swaps columns, may be employed for additional stability, albeit at a higher computational cost.

NOTE The permutation matrix is orthogonal, satisfying $^{T} =$, and it geometrically represents the reordering of the rows (or columns) of $\mathbf{A}$.

Exercise Given the matrix

$$\mathbf{A} = \begin{pmatrix} 0 & 2 & 1 \\ 2 & 1 & 0 \\ 1 & 0 & 2 \end{pmatrix},$$

determine a suitable permutation matrix so that $\mathbf{A}$ has a nonzero (and preferably large) pivot in the $(1,1)$ position. Then, perform the LU Decomposition on $\mathbf{A}$ and verify that $\mathbf{A} = \mathbf{L}\mathbf{U}$. Additionally, prove that for any permutation matrix , it holds that $^{-1} =^{T}$.

2.5 Cholesky Decomposition

Cholesky Decomposition For a symmetric positive definite (SPD) matrix $\mathbf{A} \in \mathbb{R}^{n\times n}$, the Cholesky Decomposition expresses $\mathbf{A}$ as

$$\mathbf{A} = \mathbf{L}\mathbf{L}^T,$$

where $\mathbf{L}$ is a unique lower triangular matrix with strictly positive diagonal entries.

SPD matrices, which satisfy $\boldsymbol{x}^T\mathbf{A}\boldsymbol{x} > 0$ for all nonzero $\boldsymbol{x}$, are prevalent in optimization, statistics, and finite element methods. The Cholesky factorization is both computationally efficient and numerically stable since it avoids the need for pivoting.

NOTE The uniqueness of the Cholesky Decomposition stems from the positivity of the leading principal minors of an SPD matrix, ensuring that each pivot in the elimination process is nonzero.

Exercise Prove that if $\mathbf{A}$ is SPD, then all its leading principal minors are positive. Use this result to demonstrate that the Cholesky factorization $\mathbf{A} = \mathbf{L}\mathbf{L}^T$ is unique. As an application, compute the Cholesky factorization of

$$\mathbf{A} = \begin{pmatrix} 4 & 2 \\ 2 & 3 \end{pmatrix},$$

and verify your result.

2.6 QR Decomposition and Orthogonal Transformations

QR Decomposition For a matrix $\mathbf{A} \in \mathbb{R}^{m\times n}$ with $m \geq n$, the QR Decomposition factors $\mathbf{A}$ as

$$\mathbf{A} = \mathbf{Q}\mathbf{R},$$

where $\mathbf{Q}$ is an orthogonal matrix (i.e., $\mathbf{Q}^T\mathbf{Q} =$) and $\mathbf{R}$ is an upper triangular matrix.

The QR factorization is particularly effective in solving least squares problems. Given $\mathbf{A} = \mathbf{Q}\mathbf{R}$, one can minimize $\|\mathbf{A}\boldsymbol{x} - \boldsymbol{b}\|$ by solving the simpler upper triangular system $\mathbf{R}\boldsymbol{x} = \mathbf{Q}^T\boldsymbol{b}$, as orthogonal transformations preserve Euclidean norms.

NOTE Several algorithms exist to compute the QR decomposition, including the classical Gram-Schmidt process, Householder reflections, and Givens rotations. Among these, Householder and Givens methods are favored for their superior numerical stability.

Exercise

Using a simple 3 × 2 matrix, implement the classical Gram-Schmidt process to compute its QR decomposition. Then, analyze the orthogonality of the resulting $\mathbf{Q}$ and comment on any potential numerical issues that may arise in practice.

2.6.1 Householder Transformations

Householder Reflection

A Householder reflection is an orthogonal transformation defined by

$$= -2\,\frac{{}^{T}}{\|\|^{2}},$$

where is a nonzero vector in $\mathbb{R}^m$. The matrix is symmetric and orthogonal.

Householder reflections are used to eliminate subdiagonal elements in a matrix. By applying a series of such reflections, one transforms $\mathbf{A}$ into an upper triangular matrix $\mathbf{R}$. The product of the Householder matrices forms the orthogonal factor $\mathbf{Q}^T$ in the QR decomposition.

Exercise

Prove that the product of Householder matrices ${}_k \cdots {}_2\,{}_1$ is orthogonal. Then, for a given vector $\boldsymbol{x} \in \mathbb{R}^m$, construct a Householder reflector that zeros out all but the first component of $\boldsymbol{x}$. Provide a step-by-step derivation.

2.6.2 Givens Rotations

Givens Rotation

A Givens rotation (i, j, θ) is an orthogonal transformation that rotates the (i, j)-plane by an angle θ. It is defined by inserting a 2 × 2 rotation matrix

$$\begin{pmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{pmatrix}$$

into the identity matrix at the ith and jth positions.

Givens rotations are especially useful for eliminating individual elements of a matrix, making them well-suited for sparse matrices. By applying an appropriate sequence of Givens rotations, one can compute the QR decomposition while preserving the sparsity pattern.

Exercise

Develop an algorithm using Givens rotations to compute the QR decomposition of a tridiagonal matrix. Explain how the rotations are chosen to eliminate subdiagonal elements, and analyze the algorithm's computational complexity.

2.7 Eigenvalue Decomposition

Eigenvalue Decomposition

If $\mathbf{A} \in \mathbb{R}^{n \times n}$ is diagonalizable with n linearly independent eigenvectors, then it can be expressed as

$$\mathbf{A} = \mathbf{V}\mathbf{V}^{-1},$$

where $\mathbf{V}$ is the matrix of eigenvectors and is a diagonal matrix containing the corresponding eigenvalues.

Eigenvalue decompositions are fundamental in understanding the spectral properties of matrices. They play a central role in stability analysis, vibrations, and many iterative methods. When $\mathbf{A}$ is symmetric, the eigenvalue decomposition simplifies to $\mathbf{A} = \mathbf{Q}\mathbf{Q}^T$ with orthogonal $\mathbf{Q}$, which is particularly amenable to numerical computation.

Exercise

Show that if $\mathbf{A}$ is symmetric positive definite, then all its eigenvalues are positive. Use this result to analyze the convergence of the Conjugate Gradient method for solving $\mathbf{A}\boldsymbol{x} = \boldsymbol{b}$.

2.8 Singular Value Decomposition and the Four Fundamental Subspaces

Singular Value Decomposition (SVD)

For any matrix $\mathbf{A} \in \mathbb{R}^{m \times n}$, the **Singular Value Decomposition (SVD)** expresses $\mathbf{A}$ as

$$\mathbf{A} = \mathbf{U}\boldsymbol{\Sigma}\mathbf{V}^T,$$

where $\mathbf{U} \in \mathbb{R}^{m \times m}$ and $\mathbf{V} \in \mathbb{R}^{n \times n}$ are orthogonal matrices, and $\boldsymbol{\Sigma} \in \mathbb{R}^{m \times n}$ is a diagonal (or more precisely, a rectangular diagonal) matrix whose diagonal entries $\sigma_1 \geq \sigma_2 \geq \cdots \geq 0$ are called the *singular values* of $\mathbf{A}$.

NOTE

The SVD is often regarded as the culmination of linear algebra concepts. It not only factors the matrix into rotations and scalings but also reveals the intrinsic geometric structure of $\mathbf{A}$. In particular, the SVD explicitly identifies the four fundamental subspaces associated with $\mathbf{A}$, which are essential for understanding solutions to linear systems, least squares problems, and the behavior of $\mathbf{A}$ under perturbations.

The Four Fundamental Subspaces

Given a matrix $\mathbf{A} \in \mathbb{R}^{m \times n}$, the four fundamental subspaces are:

1. **Column Space (Range):** $\mathcal{R}(\mathbf{A}) = \{\mathbf{A}\boldsymbol{x} : \boldsymbol{x} \in \mathbb{R}^n\} \subseteq \mathbb{R}^m$. This subspace consists of all vectors that can be expressed as $\mathbf{A}\boldsymbol{x}$.

2. **Null Space (Kernel):** $\mathcal{N}(\mathbf{A}) = \{\boldsymbol{x} \in \mathbb{R}^n : \mathbf{A}\boldsymbol{x} = \mathbf{0}\}$. This subspace comprises all solutions to the homogeneous equation.

3. **Row Space:** $\mathcal{R}(\mathbf{A}^T) = \{\mathbf{A}^T\boldsymbol{y} : \boldsymbol{y} \in \mathbb{R}^m\} \subseteq \mathbb{R}^n$. It is the span of the rows of $\mathbf{A}$ and is the orthogonal complement of $\mathcal{N}(\mathbf{A})$ in $\mathbb{R}^n$.

4. **Left Null Space:** $\mathcal{N}(\mathbf{A}^T) = \{\boldsymbol{y} \in \mathbb{R}^m : \mathbf{A}^T\boldsymbol{y} = \mathbf{0}\}$. This is the set of all vectors in $\mathbb{R}^m$ that are orthogonal to every column of $\mathbf{A}$.

NOTE

Link to SVD: In the SVD $\mathbf{A} = \mathbf{U}\boldsymbol{\Sigma}\mathbf{V}^T$, the columns of $\mathbf{U}$ corresponding to nonzero singular values form an orthonormal basis for the column space $\mathcal{R}(\mathbf{A})$, while the remaining columns of $\mathbf{U}$ form a basis for the left null space $\mathcal{N}(\mathbf{A}^T)$. Similarly, the columns of $\mathbf{V}$ corresponding to nonzero singular values span the row space $\mathcal{R}(\mathbf{A}^T)$, and those corresponding to zero singular values span the null space $\mathcal{N}(\mathbf{A})$.

Link to Solutions of Linear Equations

Consider the linear system $\mathbf{A}\boldsymbol{x} = \boldsymbol{b}$. The existence and uniqueness of solutions depend on the relationship between $\boldsymbol{b}$ and the fundamental subspaces:

- A solution exists if and only if $\boldsymbol{b} \in \mathcal{R}(\mathbf{A})$.
- When $\boldsymbol{b} \in \mathcal{R}(\mathbf{A})$, the general solution can be expressed as the sum of a particular solution and any vector in the null space $\mathcal{N}(\mathbf{A})$.
- In the context of least squares, when $\boldsymbol{b}$ is not in $\mathcal{R}(\mathbf{A})$, one seeks a solution $\hat{\boldsymbol{x}}$ such that the residual $\|\mathbf{A}\hat{\boldsymbol{x}} - \boldsymbol{b}\|$ is minimized. The SVD provides a natural framework to compute the minimum-norm solution via the pseudoinverse.

Exercise

For a given matrix $\mathbf{A}$, use its SVD to:

1. Determine orthonormal bases for $\mathcal{R}(\mathbf{A})$, $\mathcal{N}(\mathbf{A})$, $\mathcal{R}(\mathbf{A}^T)$, and $\mathcal{N}(\mathbf{A}^T)$.
2. Analyze the conditions under which the linear system $\mathbf{A}\boldsymbol{x} = \boldsymbol{b}$ has a unique solution, infinitely many solutions, or no solution.
3. Show that if $\boldsymbol{b} \notin \mathcal{R}(\mathbf{A})$, the least squares solution is given by $\hat{\boldsymbol{x}} = \mathbf{A}^+\boldsymbol{b}$, where $\mathbf{A}^+$ is the Moore-Penrose pseudoinverse derived from the SVD.

NOTE

The SVD not only decomposes the matrix into its fundamental components but also organizes its action into rotations (via $\mathbf{U}$ and $\mathbf{V}$) and scalings (via $\boldsymbol{\Sigma}$). This clear separation allows us to understand how errors propagate in the solution of linear systems and how the geometry of $\mathbf{A}$ influences the solvability and stability of $\mathbf{A}\boldsymbol{x} = \boldsymbol{b}$.

2.9 Rank Truncation via the SVD: Full Mathematical Explanation

Singular Value Decomposition

For any matrix $\mathbf{A} \in \mathbb{R}^{m\times n}$, the Singular Value Decomposition (SVD) is given by

$$\mathbf{A} = \mathbf{U}\boldsymbol{\Sigma}\mathbf{V}^T,$$

where:

- $\mathbf{U} \in \mathbb{R}^{m\times m}$ is an orthogonal matrix whose columns $\{_1,_2,\ldots,_m\}$ are the left singular vectors.
- $\mathbf{V} \in \mathbb{R}^{n\times n}$ is an orthogonal matrix whose columns $\{_1,_2,\ldots,_n\}$ are the right singular vectors.
- $\boldsymbol{\Sigma} \in \mathbb{R}^{m\times n}$ is a diagonal matrix with nonnegative entries $\sigma_1, \sigma_2, \ldots, \sigma_p$ (with $p = \min(m,n)$) arranged in descending order:

$$\boldsymbol{\Sigma} = \begin{pmatrix} \sigma_1 & & & 0 \\ & \sigma_2 & & \\ & & \ddots & \\ 0 & & & \sigma_p \end{pmatrix}.$$

NOTE **Intuition:** The SVD decomposes the matrix $\mathbf{A}$ into three parts: the rotations $\mathbf{U}$ and $\mathbf{V}^T$ and the scaling $\boldsymbol{\Sigma}$. This factorization exposes the intrinsic rank and geometric features of $\mathbf{A}$.

Rank Truncation Let r be an integer with $0 \le r \le p$. The **rank-r approximation** of $\mathbf{A}$ is defined as

$$\mathbf{A}_r = \sum_{i=1}^{r} \sigma_{i}{}_{i}{}_{i}^{T},$$

or equivalently, if we partition $\mathbf{U}$, $\boldsymbol{\Sigma}$, and $\mathbf{V}$ as

$$\mathbf{U} = \begin{pmatrix} \mathbf{U}_r & \mathbf{U}_{r,\perp} \end{pmatrix}, \quad \boldsymbol{\Sigma} = \begin{pmatrix} \boldsymbol{\Sigma}_r & 0 \\ 0 & \boldsymbol{\Sigma}_{r,\perp} \end{pmatrix}, \quad \mathbf{V} = \begin{pmatrix} \mathbf{V}_r & \mathbf{V}_{r,\perp} \end{pmatrix},$$

then

$$\mathbf{A}_r = \mathbf{U}_r\boldsymbol{\Sigma}_r\mathbf{V}_r^T.$$

Here, $\boldsymbol{\Sigma}_r \in \mathbb{R}^{r\times r}$ contains the largest r singular values, while $\mathbf{U}_r$ and $\mathbf{V}_r$ contain the corresponding singular vectors.

NOTE **Eckart-Young Theorem:** This truncated SVD, $\mathbf{A}_r$, is the best approximation to $\mathbf{A}$ among all matrices of rank at most r. Specifically, for any matrix $\mathbf{B}$ of rank at most r,

$$\|\mathbf{A} - \mathbf{A}_r\|_2 \le \|\mathbf{A} - \mathbf{B}\|_2 \quad \text{and} \quad \|\mathbf{A} - \mathbf{A}_r\|_F \le \|\mathbf{A} - \mathbf{B}\|_F.$$

Moreover, the errors are given by

$$\|\mathbf{A} - \mathbf{A}_r\|_2 = \sigma_{r+1}, \quad \text{and} \quad \|\mathbf{A} - \mathbf{A}_r\|_F = \sqrt{\sum_{i=r+1}^{p} \sigma_i^2}.$$

Exercise Prove the Eckart-Young Theorem for the 2-norm case. In particular, show that for any matrix $\mathbf{B}$ with $\text{rank}(\mathbf{B}) \leq r$,

$$\|\mathbf{A} - \mathbf{B}\|_2 \geq \sigma_{r+1}.$$

(Hint: Consider the singular value decomposition of $\mathbf{A}$ and use the unitary invariance of the spectral norm.)

To see the derivation, observe that the SVD of $\mathbf{A}$ gives

$$\mathbf{A} = \sum_{i=1}^{p} \sigma_{i}{}_{i}{}_{i}^{T}.$$

Truncating the series at r terms results in

$$\mathbf{A}_r = \sum_{i=1}^{r} \sigma_{i}{}_{i}{}_{i}^{T},$$

and the remainder is

$$\mathbf{A} - \mathbf{A}_r = \sum_{i=r+1}^{p} \sigma_{i}{}_{i}{}_{i}^{T}.$$

Since the spectral norm (or 2-norm) is given by the largest singular value, it follows that

$$\|\mathbf{A} - \mathbf{A}_r\|_2 = \sigma_{r+1}.$$

Any other rank-r approximation $\mathbf{B}$ must incur an error at least as large as this, establishing the optimality of $\mathbf{A}_r$.

NOTE **Exercise Hint:** The singular values of $\mathbf{A} - \mathbf{B}$ cannot be made smaller than those in the truncated tail $\{\sigma_{r+1}, \ldots, \sigma_p\}$ due to the unitary invariance of the norm.

Exercise For a given 4×4 matrix with singular values $\sigma_1, \sigma_2, \sigma_3, \sigma_4$, construct the rank-2 approximation $\mathbf{A}_2$ and compute $\|\mathbf{A} - \mathbf{A}_2\|_2$ and $\|\mathbf{A} - \mathbf{A}_2\|_F$. Compare your results with the theoretical error bounds.

2.10 Conditioning and Stability Revisited

Throughout this discussion, the notion of conditioning pops up ubiquitously. Although numerical algorithms can be designed to be backward stable—meaning the computed solution is the exact solution of a slightly perturbed problem—the inherent sensitivity of a problem is dictated by the condition number $\kappa(\mathbf{A})$. If $\mathbf{A}$ is ill-conditioned, even the most stable algorithm may yield a solution with significant relative error.

NOTE Backward stable algorithms do not improve the condition number of the problem; they merely ensure that the numerical method does not introduce additional error beyond what is implied by $\kappa(\mathbf{A})$. For instance, LU factorization with partial pivoting is backward stable for most matrices, yet if $\kappa(\mathbf{A})$ is large, the final solution may still be inaccurate.

Exercise Consider a matrix $\mathbf{A}$ with an extremely large condition number (for example, one of the matrices you explored earlier in this section). Introduce a small relative perturbation ϵ to the right-hand side $\boldsymbol{b}$, and solve $\mathbf{A}\boldsymbol{x} = \boldsymbol{b}$ using a stable LU decomposition with partial pivoting. Compare the relative error in the computed solution $\hat{\boldsymbol{x}}$ to $\kappa(\mathbf{A})\epsilon$. Provide a detailed discussion on how the condition number impacts the accuracy of the solution and propose methods to mitigate these effects.

NOTE The choice of method depends critically on the structure of the matrix (e.g., symmetry, sparsity, bandedness) and the nature of the problem (e.g., direct solution versus iterative refinement). Advanced topics such as preconditioning and randomized algorithms for SVD extend these ideas into the realm of large-scale, high-performance computing.

Exercise Explore the implementation of these matrix decompositions in a standard numerical library (e.g., LAPACK, MATLAB, NumPy, or Julia). Compare the theoretical operation counts with empirical performance data for large-scale problems. Discuss how design choices—such as pivoting, blocking, or parallelization strategies—affect both the computational efficiency and the numerical stability of the algorithms.

2.11 Overdetermined and Underdetermined Systems

NOTE Not all systems have the same structure. An **overdetermined system** (more equations than unknowns, $m > n$) typically has no exact solution, necessitating least squares techniques. An **underdetermined system** ($m < n$) has infinitely many solutions, and additional criteria (such as minimum norm) are used to select a unique solution.

Exercise Discuss the differences between overdetermined and underdetermined systems. Show how the SVD and the concept of the pseudoinverse provide natural ways to derive solutions in both cases.

2.12 Conditioning and Stability Revisited

Throughout this discussion, the inherent sensitivity of a problem, as measured by the condition number $\kappa(\mathbf{A})$, plays a crucial role. Even with backward stable algorithms, if $\mathbf{A}$ is ill-conditioned, the computed solution may exhibit significant relative errors.

NOTE Backward stable methods ensure that the computed solution is exact for a nearby problem, but they do not reduce the condition number. For example, LU factorization with partial pivoting is backward stable, yet for a matrix with a high $\kappa(\mathbf{A})$, the final error in $\hat{\boldsymbol{x}}$ can be amplified.

Exercise Consider a matrix $\mathbf{A}$ with a very large condition number. Introduce a small relative perturbation ϵ to the right-hand side $\boldsymbol{b}$, and solve $\mathbf{A}\boldsymbol{x} = \boldsymbol{b}$ using a stable LU decomposition with partial pivoting. Compare the relative error in $\hat{\boldsymbol{x}}$ with $\kappa(\mathbf{A})\epsilon$. Discuss strategies such as preconditioning to mitigate the effects of high condition numbers.

2.13 Least Squares

Least Squares Problem Consider an overdetermined linear system

$$\mathbf{A}\boldsymbol{x} = \boldsymbol{b},$$

where $\mathbf{A} \in \mathbb{R}^{m\times n}$ with $m > n$ and $\boldsymbol{b} \in \mathbb{R}^m$. In general, no exact solution $\boldsymbol{x}$ exists that satisfies the system. The **least squares problem** seeks an approximate solution $\hat{\boldsymbol{x}}$ that minimizes the residual error measured in the 2-norm:

$$\hat{\boldsymbol{x}} =_{\boldsymbol{x}\in\mathbb{R}^n} \|\mathbf{A}\boldsymbol{x} - \boldsymbol{b}\|_2.$$

In other words, we wish to find the vector $\hat{\boldsymbol{x}}$ that minimizes the function

$$f(\boldsymbol{x}) = \|\mathbf{A}\boldsymbol{x} - \boldsymbol{b}\|_2^2.$$

NOTE Geometrically, the least squares solution corresponds to projecting the vector $\boldsymbol{b}$ onto the column space of $\mathbf{A}$. The computed $\hat{\boldsymbol{x}}$ yields the point $\mathbf{A}\hat{\boldsymbol{x}}$ in the column space that is closest (in the Euclidean norm) to $\boldsymbol{b}$.

We begin by expanding the objective function. Notice that

$$f(\boldsymbol{x}) = \|\mathbf{A}\boldsymbol{x} - \boldsymbol{b}\|_2^2 = (\mathbf{A}\boldsymbol{x} - \boldsymbol{b})^T(\mathbf{A}\boldsymbol{x} - \boldsymbol{b}).$$

Expanding this quadratic form, we have

$$f(\boldsymbol{x}) = \boldsymbol{x}^T\mathbf{A}^T\mathbf{A}\boldsymbol{x} - 2\boldsymbol{b}^T\mathbf{A}\boldsymbol{x} + \boldsymbol{b}^T\boldsymbol{b}.$$

Since $\boldsymbol{b}^T\boldsymbol{b}$ is constant with respect to $\boldsymbol{x}$, minimizing $f(\boldsymbol{x})$ is equivalent to minimizing the quadratic function

$$g(\boldsymbol{x}) = \boldsymbol{x}^T\mathbf{A}^T\mathbf{A}\boldsymbol{x} - 2\boldsymbol{b}^T\mathbf{A}\boldsymbol{x}.$$

To find the minimum, we compute the gradient of $g(\boldsymbol{x})$ with respect to $\boldsymbol{x}$. Using standard results from calculus, the gradient is given by

$$\nabla_{\boldsymbol{x}} g(\boldsymbol{x}) = 2\mathbf{A}^T\mathbf{A}\boldsymbol{x} - 2\mathbf{A}^T\boldsymbol{b}.$$

Setting the gradient equal to zero (a necessary condition for optimality), we obtain the **normal equations**:

$$2\mathbf{A}^T\mathbf{A}\boldsymbol{x} - 2\mathbf{A}^T\boldsymbol{b} = \mathbf{0} \quad \Longrightarrow \quad \mathbf{A}^T\mathbf{A}\boldsymbol{x} = \mathbf{A}^T\boldsymbol{b}.$$

NOTE The normal equations represent a system of n equations in n unknowns. Under the assumption that $\mathbf{A}$ has full column rank (i.e., $\text{rank}(\mathbf{A}) = n$), the matrix $\mathbf{A}^T\mathbf{A}$ is invertible, and the least squares solution is unique.

Normal Equations and the Least Squares Solution

Assuming that $\mathbf{A}^T\mathbf{A}$ is invertible, the unique least squares solution is given by

$$\hat{\boldsymbol{x}} = (\mathbf{A}^T\mathbf{A})^{-1}\mathbf{A}^T\boldsymbol{b}.$$

This formula is central to the theory and practice of least squares, providing an explicit expression for the solution.

NOTE An important consequence of the least squares derivation is that the residual $\boldsymbol{r} = \boldsymbol{b} - \mathbf{A}\hat{\boldsymbol{x}}$ is orthogonal to the column space of $\mathbf{A}$. In fact, multiplying the normal equations by $\hat{\boldsymbol{x}}$ shows that

$$\mathbf{A}^T(\boldsymbol{b} - \mathbf{A}\hat{\boldsymbol{x}}) = \mathbf{0}.$$

This orthogonality property is key to many theoretical and practical aspects of least squares analysis.

Exercise Show that if $\hat{\boldsymbol{x}} = (\mathbf{A}^T\mathbf{A})^{-1}\mathbf{A}^T\boldsymbol{b}$ is the least squares solution of $\mathbf{A}\boldsymbol{x} = \boldsymbol{b}$, then the residual $\boldsymbol{r} = \boldsymbol{b} - \mathbf{A}\hat{\boldsymbol{x}}$ is orthogonal to every column of $\mathbf{A}$; that is, prove that $\mathbf{A}^T\boldsymbol{r} = \mathbf{0}$.

An alternative viewpoint is to express the least squares solution as the orthogonal projection of $\boldsymbol{b}$ onto the column space of $\mathbf{A}$. Define the projection matrix

$$= \mathbf{A}(\mathbf{A}^T\mathbf{A})^{-1}\mathbf{A}^T.$$

Then, the projection of $\boldsymbol{b}$ onto the column space of $\mathbf{A}$ is given by

$$\hat{\boldsymbol{b}} = \boldsymbol{b} = \mathbf{A}\hat{\boldsymbol{x}}.$$

The matrix has the properties of being symmetric ($^T =$) and idempotent ($^2 =$), and it plays a fundamental role in both theoretical analyses and practical algorithms for least squares problems.

Exercise Prove that the projection matrix $= \mathbf{A}(\mathbf{A}^T\mathbf{A})^{-1}\mathbf{A}^T$ is both symmetric and idempotent.

While the normal equations provide a direct way to obtain the least squares solution, they are not always the most numerically stable approach. The computation of $(\mathbf{A}^T\mathbf{A})^{-1}$ can lead to loss of precision, particularly when $\mathbf{A}$ is ill-conditioned. For this reason, alternative methods—such as the QR Decomposition or the Singular Value Decomposition (SVD)—are often employed in practice to solve least squares problems.

QR-Based Least Squares Suppose $\mathbf{A}$ has a QR factorization $\mathbf{A} = \mathbf{QR}$ where $\mathbf{Q} \in \mathbb{R}^{m\times m}$ is orthogonal and $\mathbf{R} \in \mathbb{R}^{m\times n}$ is upper triangular (with the last $m - n$ rows being zero if $m > n$). Then, the least squares problem becomes

$$\min_{\boldsymbol{x}} \|\mathbf{QR}\boldsymbol{x} - \boldsymbol{b}\|_2 = \min_{\boldsymbol{x}} \|\mathbf{R}\boldsymbol{x} - \mathbf{Q}^T\boldsymbol{b}\|_2,$$

since $\mathbf{Q}$ preserves the 2-norm. The solution is obtained by solving the triangular system

$$\mathbf{R}_1\boldsymbol{x} = \mathbf{Q}_1^T\boldsymbol{b},$$

where $\mathbf{R}_1$ is the upper $n \times n$ portion of $\mathbf{R}$ and $\mathbf{Q}_1$ consists of the first n columns of $\mathbf{Q}$.

NOTE This approach avoids forming $\mathbf{A}^T\mathbf{A}$ directly, thereby mitigating the potential numerical instability associated with squaring the condition number.

Exercise Given a full-rank matrix $\mathbf{A} \in \mathbb{R}^{m\times n}$ with $m > n$ and its QR decomposition $\mathbf{A} = \mathbf{QR}$, derive the reduced form of the least squares solution by showing that solving $\mathbf{R}_1\boldsymbol{x} = \mathbf{Q}_1^T\boldsymbol{b}$ is equivalent to solving the normal equations.

SVD-Based Least Squares The Singular Value Decomposition (SVD) offers yet another robust method for solving least squares problems. If $\mathbf{A} = \mathbf{U\Sigma V}^T$ is the SVD of $\mathbf{A}$, then the least squares solution is given by

$$\hat{\boldsymbol{x}} = \mathbf{V\Sigma}^+\mathbf{U}^T\boldsymbol{b},$$

where $\mathbf{\Sigma}^+$ is the pseudoinverse of the diagonal matrix $\mathbf{\Sigma}$. This method is particularly advantageous when $\mathbf{A}$ is rank-deficient or nearly singular.

NOTE

The SVD-based approach naturally provides insight into the rank and conditioning of $\mathbf{A}$. Small singular values indicate directions in which $\mathbf{A}$ nearly loses rank, and they contribute disproportionately to the error in $\hat{\boldsymbol{x}}$.

Exercise For a given matrix $\mathbf{A}$ with known singular values, use the SVD-based formula $\hat{\boldsymbol{x}} = \mathbf{V}\boldsymbol{\Sigma}^{+}\mathbf{U}^{T}\boldsymbol{b}$ to compute the least squares solution. Analyze how the presence of very small singular values affects the solution, and discuss strategies (such as truncation) to mitigate these effects.

In summary, the least squares method is a fundamental tool for solving overdetermined systems. It minimizes the error between the observed vector $\boldsymbol{b}$ and the predicted vector $\mathbf{A}\boldsymbol{x}$ in a least-squares sense. The derivation via the normal equations provides a clear and concise formulation, while alternative approaches based on QR Decomposition and SVD offer improved numerical stability in practical applications.

Index