Linear Algebra

Philipp W.

April 18, 2025

Contents

1 Introduction

2 Vectors

Let $\mathbf{u}$ and $\mathbf{v}$ be vectors in $\mathbb{R}^n$, and let α be a scalar (real number). We write

$$\mathbf{u} = \begin{pmatrix} u_1 \\ u_2 \\ \vdots \\ u_n \end{pmatrix}, \quad \mathbf{v} = \begin{pmatrix} v_1 \\ v_2 \\ \vdots \\ v_n \end{pmatrix}.$$

Running Example (in $\mathbb{R}^2$): Throughout these subsections, we illustrate computations with the following vectors:

$$\mathbf{u} = \begin{pmatrix} 2 \\ 3 \end{pmatrix}, \quad \mathbf{v} = \begin{pmatrix} 1 \\ -1 \end{pmatrix}, \quad \text{and sometimes a third vector } \mathbf{w} = \begin{pmatrix} 3 \\ -2 \end{pmatrix}.$$

All arithmetic shown in the examples is specific to these choices.

2.1 Operations on Vectors

2.1.1 Vector Addition and Scalar Multiplication

$$\mathbf{u} + \mathbf{v} = \begin{pmatrix} u_1 + v_1 \\ u_2 + v_2 \\ \vdots \\ u_n + v_n \end{pmatrix}, \quad \alpha\mathbf{u} = \begin{pmatrix} \alpha u_1 \\ \alpha u_2 \\ \vdots \\ \alpha u_n \end{pmatrix}.$$

- *Commutativity:* $\mathbf{u} + \mathbf{v} = \mathbf{v} + \mathbf{u}$.
- *Associativity:* $(\mathbf{u} + \mathbf{v}) + \mathbf{w} = \mathbf{u} + (\mathbf{v} + \mathbf{w})$.
- *Distributivity over vector addition:* $\alpha(\mathbf{u} + \mathbf{v}) = \alpha\mathbf{u} + \alpha\mathbf{v}$.
- *Distributivity over scalar addition:* $(\alpha + \beta)\mathbf{u} = \alpha\mathbf{u} + \beta\mathbf{u}$.

Example (in $\mathbb{R}^2$):

$$\mathbf{u} = \begin{pmatrix} 2 \\ 3 \end{pmatrix}, \quad \mathbf{v} = \begin{pmatrix} 1 \\ -1 \end{pmatrix}.$$

Then

$$\mathbf{u} + \mathbf{v} = \begin{pmatrix} 2 + 1 \\ 3 + (-1) \end{pmatrix} = \begin{pmatrix} 3 \\ 2 \end{pmatrix}, \quad 2\mathbf{u} = \begin{pmatrix} 4 \\ 6 \end{pmatrix}.$$

2.1.2 Dot (Inner) Product

$$\mathbf{u} \cdot \mathbf{v} \;=\; \mathbf{u}^\top \mathbf{v} \;=\; \sum_{i=1}^{n} u_i\, v_i.$$

- *Commutative:* $\mathbf{u} \cdot \mathbf{v} = \mathbf{v} \cdot \mathbf{u}$.
- *Distributive over addition:* $\mathbf{u} \cdot (\mathbf{v} + \mathbf{w}) = \mathbf{u} \cdot \mathbf{v} + \mathbf{u} \cdot \mathbf{w}$.
- *Bilinear:* $(\alpha\mathbf{u}) \cdot \mathbf{v} = \alpha(\mathbf{u} \cdot \mathbf{v})$.
- *Relation to vector length:*

$$\|\mathbf{u}\|^2 = \mathbf{u} \cdot \mathbf{u} \;=\; \sum_{i=1}^{n} u_i^2.$$

Example (in $\mathbb{R}^2$):

$$\mathbf{u} \cdot \mathbf{v} = \begin{pmatrix} 2 \\ 3 \end{pmatrix} \cdot \begin{pmatrix} 1 \\ -1 \end{pmatrix} = 2 \cdot 1 + 3 \cdot (-1) = 2 - 3 = -1.$$

Also,

$$\|\mathbf{u}\| = \sqrt{\mathbf{u} \cdot \mathbf{u}} = \sqrt{2^2 + 3^2} = \sqrt{4 + 9} = \sqrt{13}.$$

2.1.3 Outer Product

$$\mathbf{u}\,\mathbf{v}^\top = \begin{pmatrix} u_1 \\ u_2 \\ \vdots \\ u_n \end{pmatrix} \begin{pmatrix} v_1 & v_2 & \cdots & v_n \end{pmatrix}.$$

- The result is an $n \times n$ matrix.
- *Non-commutative:* $\mathbf{u}\mathbf{v}^\top \neq \mathbf{v}^\top\mathbf{u}$ in general (the latter is a 1×1 matrix, i.e. a scalar).

Example (in $\mathbb{R}^2$):

$$\mathbf{u} = \begin{pmatrix} 2 \\ 3 \end{pmatrix}, \quad \mathbf{v} = \begin{pmatrix} 1 \\ -1 \end{pmatrix}.$$

Then

$$\mathbf{u}\mathbf{v}^\top = \begin{pmatrix} 2 \\ 3 \end{pmatrix} \begin{pmatrix} 1 & -1 \end{pmatrix} = \begin{pmatrix} 2 \cdot 1 & 2 \cdot (-1) \\ 3 \cdot 1 & 3 \cdot (-1) \end{pmatrix} = \begin{pmatrix} 2 & -2 \\ 3 & -3 \end{pmatrix}.$$

2.1.4 Cross Product

The *cross product* is a binary operation defined for vectors in three-dimensional space:

$$\times : \mathbb{R}^3 \times \mathbb{R}^3 \to \mathbb{R}^3,$$

which takes two vectors $\mathbf{a}, \mathbf{b} \in \mathbb{R}^3$ and returns a new vector $\mathbf{a} \times \mathbf{b}$ that is orthogonal to both $\mathbf{a}$ and $\mathbf{b}$.

$$\mathbf{a} \times \mathbf{b} \;=\; \begin{vmatrix} \mathbf{e}_1 & \mathbf{e}_2 & \mathbf{e}_3 \\ a_1 & a_2 & a_3 \\ b_1 & b_2 & b_3 \end{vmatrix} \;=\; \begin{pmatrix} a_2 b_3 - a_3 b_2 \\ a_3 b_1 - a_1 b_3 \\ a_1 b_2 - a_2 b_1 \end{pmatrix},$$

where $\mathbf{e}_1, \mathbf{e}_2, \mathbf{e}_3$ are the standard basis vectors in $\mathbb{R}^3$, and

$$\mathbf{a} = \begin{pmatrix} a_1 \\ a_2 \\ a_3 \end{pmatrix}, \quad \mathbf{b} = \begin{pmatrix} b_1 \\ b_2 \\ b_3 \end{pmatrix}.$$

- **Orthogonality:** $\mathbf{a} \times \mathbf{b}$ is always *orthogonal* to both $\mathbf{a}$ *and* $\mathbf{b}$. Formally,

  $$(\mathbf{a} \times \mathbf{b}) \cdot \mathbf{a} \;=\; 0, \quad (\mathbf{a} \times \mathbf{b}) \cdot \mathbf{b} \;=\; 0.$$

- **Magnitude:** The magnitude of $\mathbf{a} \times \mathbf{b}$ is given by

  $$\|\mathbf{a} \times \mathbf{b}\| \;=\; \|\mathbf{a}\|\,\|\mathbf{b}\|\,\sin(\theta),$$

  where θ is the angle between $\mathbf{a}$ and $\mathbf{b}$ (cf. Section on *Angle Between Vectors*). Thus, $\|\mathbf{a} \times \mathbf{b}\|$ equals the *area* of the parallelogram spanned by $\mathbf{a}$ and $\mathbf{b}$.

- **Direction (Right-hand rule):** The direction of $\mathbf{a} \times \mathbf{b}$ follows the *right-hand rule*: if you point the index finger of your right hand along $\mathbf{a}$ and your middle finger along $\mathbf{b}$, your thumb points in the direction of $\mathbf{a} \times \mathbf{b}$. Equivalently,

  $$\mathbf{a} \times \mathbf{b} = -\,(\mathbf{b} \times \mathbf{a}).$$

- **Anticommutativity:**

  $$\mathbf{a} \times \mathbf{b} \;=\; -\,(\mathbf{b} \times \mathbf{a}), \quad \mathbf{a} \times \mathbf{a} = \mathbf{0}.$$

- **Distributivity over addition:**

  $$\mathbf{a} \times (\mathbf{b} + \mathbf{c}) \;=\; \mathbf{a} \times \mathbf{b} \;+\; \mathbf{a} \times \mathbf{c}.$$

- **Scalar multiplication:**

  $$(\alpha\mathbf{a}) \times \mathbf{b} \;=\; \alpha(\mathbf{a} \times \mathbf{b}), \quad \mathbf{a} \times (\beta\mathbf{b}) \;=\; \beta(\mathbf{a} \times \mathbf{b}).$$

Example in $\mathbb{R}^3$

Suppose

$$\mathbf{a} = \begin{pmatrix}1\\2\\3\end{pmatrix}, \quad \mathbf{b} = \begin{pmatrix}2\\1\\4\end{pmatrix}.$$

Then

$$\mathbf{a}\times\mathbf{b} = \begin{pmatrix}(2)(4)-(3)(1)\\(3)(2)-(1)(4)\\(1)(1)-(2)(2)\end{pmatrix} = \begin{pmatrix}8-3\\6-4\\1-4\end{pmatrix} = \begin{pmatrix}5\\2\\-3\end{pmatrix}.$$

Indeed,

$$(\mathbf{a}\times\mathbf{b})\cdot\mathbf{a} = 5\cdot 1 + 2\cdot 2 + (-3)\cdot 3 = 5+4-9 = 0,$$

confirming orthogonality with $\mathbf{a}$, and similarly one checks $(\mathbf{a}\times\mathbf{b})\cdot\mathbf{b} = 0$.

Scalar Triple Product and Volume Interpretation

For vectors $\mathbf{a},\mathbf{b},\mathbf{c}\in\mathbb{R}^3$, the *scalar triple product* is defined as

$$\mathbf{a}\cdot(\mathbf{b}\times\mathbf{c}),$$

and it gives the (signed) volume of the parallelepiped determined by the three vectors. Hence,

$$\left|\mathbf{a}\cdot(\mathbf{b}\times\mathbf{c})\right|$$

represents the volume of that parallelepiped.

- **Cyclic permutations:**
$$\mathbf{a}\cdot(\mathbf{b}\times\mathbf{c}) \;=\; \mathbf{b}\cdot(\mathbf{c}\times\mathbf{a}) \;=\; \mathbf{c}\cdot(\mathbf{a}\times\mathbf{b}).$$
The scalar triple product is invariant under any cyclic permutation of $\mathbf{a}$, $\mathbf{b}$, $\mathbf{c}$.
- **Geometric interpretation:** $\|\mathbf{b}\times\mathbf{c}\|$ is the area of the parallelogram spanned by $\mathbf{b}$ and $\mathbf{c}$, and then the dot product with $\mathbf{a}$ effectively multiplies this area by the perpendicular height from $\mathbf{a}$. The sign indicates orientation (via the right-hand rule).

In summary, the cross product is a key operation in 3D vector calculus and geometry, giving a new vector orthogonal to two given vectors. It does *not* generalize in a straightforward way to higher-dimensional spaces (the notion of a *wedge product* does generalize, but behaves differently). However, in $\mathbb{R}^3$, the cross product is crucial for many geometric and physical applications (e.g. torque, angular momentum, electromagnetic fields).

2.1.5 Element-wise (Hadamard) Product

For $\mathbf{u},\mathbf{v}\in\mathbb{R}^n$ their *Hadamard product* is the vector obtained by multiplying matching components:

$$\mathbf{u}\odot\mathbf{v} \;=\; \begin{pmatrix}u_1v_1\\u_2v_2\\\vdots\\u_nv_n\end{pmatrix}.$$

- **Commutative & associative** (because real multiplication is).
- **Distributes** over component-wise addition: $\mathbf{u}\odot(\mathbf{v}+\mathbf{w}) = \mathbf{u}\odot\mathbf{v} + \mathbf{u}\odot\mathbf{w}$.
- Shows up in statistics (variance of element-wise products), machine learning (attention mechanisms), and digital signal processing.

Example (in $\mathbb{R}^2$).

$$\begin{pmatrix}2\\3\end{pmatrix} \odot \begin{pmatrix}1\\-1\end{pmatrix} = \begin{pmatrix}2\cdot 1\\3\cdot(-1)\end{pmatrix} = \begin{pmatrix}2\\-3\end{pmatrix}.$$

2.1.6 Kronecker (Tensor) Product of Vectors

For $\mathbf{u} \in \mathbb{R}^m$ and $\mathbf{v} \in \mathbb{R}^n$, their *Kronecker product* is the mn-dimensional vector

$$\mathbf{u} \otimes \mathbf{v} \;=\; \begin{pmatrix}u_1\,\mathbf{v}\\u_2\,\mathbf{v}\\\vdots\\u_m\,\mathbf{v}\end{pmatrix} = \begin{pmatrix}u_1v_1\\u_1v_2\\\cdots\\u_1v_n\\u_2v_1\\u_2v_2\\\cdots\\u_2v_n\\\vdots\\u_mv_1\\u_mv_2\\\cdots\\u_mv_n\end{pmatrix}.$$

- Widely used in signal processing, quantum computing, and to express the covariance of $\mathrm{vec}(X)$ for random matrices X.
- Not commutative but satisfies a *mixed-product* property with ordinary matrix multiplication: $(A \otimes B)(C \otimes D) = (AC) \otimes (BD)$ whenever the products make sense.

Example ($m = n = 2$).

$$\begin{pmatrix}2\\3\end{pmatrix} \otimes \begin{pmatrix}1\\-1\end{pmatrix} = \begin{pmatrix}2\cdot 1\\2\cdot(-1)\\3\cdot 1\\3\cdot(-1)\end{pmatrix} = \begin{pmatrix}2\\-2\\3\\-3\end{pmatrix}.$$

2.1.7 Vector Triple Product (Lagrange Identity)

For $\mathbf{a}, \mathbf{b}, \mathbf{c} \in \mathbb{R}^3$ the *vector triple product* expands as

$$\mathbf{a} \times (\mathbf{b} \times \mathbf{c}) \;=\; (\mathbf{a}\cdot\mathbf{c})\,\mathbf{b} \;-\; (\mathbf{a}\cdot\mathbf{b})\,\mathbf{c}.$$

Swapping the first two vectors flips the sign:

$$(\mathbf{a} \times \mathbf{b}) \times \mathbf{c} \;=\; (\mathbf{a}\cdot\mathbf{c})\,\mathbf{b} \;-\; (\mathbf{b}\cdot\mathbf{c})\,\mathbf{a}.$$

Geometric use-case. The formula resolves the component of $\mathbf{c}$ lying in the plane spanned by $\mathbf{a}$ and $\mathbf{b}$; it is also the basis for derivations of torque and magnetic forces in physics.

2.1.8 Vector Stack (Concatenation)

Given two column vectors $\mathbf{u} \in \mathbb{R}^m$ and $\mathbf{v} \in \mathbb{R}^n$, the *vertical stack* (or *concatenation*) is the $(m+n)$-dimensional vector

$$\operatorname{stack}(\mathbf{u}, \mathbf{v}) \;=\; \begin{pmatrix} \mathbf{u} \\ \mathbf{v} \end{pmatrix} = \begin{pmatrix} u_1 \\ \vdots \\ u_m \\ v_1 \\ \vdots \\ v_n \end{pmatrix}.$$

If both vectors are rows, the same notation yields a longer row. For horizontal concatenation of columns one writes $\left(\mathbf{u}^\top \,|\, \mathbf{v}^\top\right)$, which is a $1 \times (m+n)$ row.

Example (in $\mathbb{R}^{2+2}$).

$$\mathbf{u} = \begin{pmatrix} 2 \\ 3 \end{pmatrix}, \; \mathbf{v} = \begin{pmatrix} 1 \\ -1 \end{pmatrix} \quad \Longrightarrow \quad \operatorname{stack}(\mathbf{u}, \mathbf{v}) = \begin{pmatrix} 2 \\ 3 \\ 1 \\ -1 \end{pmatrix}.$$

2.2 Properties of Vectors

2.2.1 Vector Spaces

A set V is called a **vector space** (or **linear space**) over the field $\mathbb{R}$ if it is closed under:

- **Vector addition:** $+ : V \times V \to V$
- **Scalar multiplication:** $\cdot : \mathbb{R} \times V \to V$

That is, for all $\mathbf{v}_1, \mathbf{v}_2 \in V$ and all scalars $\alpha, \beta \in \mathbb{R}$, the combination $\alpha\mathbf{v}_1 + \beta\mathbf{v}_2 \in V$.

With respect to addition, V forms a *commutative group*:

- Associativity: $(\mathbf{u} + \mathbf{v}) + \mathbf{w} = \mathbf{u} + (\mathbf{v} + \mathbf{w})$
- Commutativity: $\mathbf{u} + \mathbf{v} = \mathbf{v} + \mathbf{u}$
- Existence of identity: $\exists \mathbf{0} \in V$ such that $\mathbf{u} + \mathbf{0} = \mathbf{u}$
- Existence of inverses: For all $\mathbf{u} \in V$, $\exists - \mathbf{u} \in V$ such that $\mathbf{u} + (-\mathbf{u}) = \mathbf{0}$

Scalar multiplication must satisfy:

- Compatibility: $\alpha(\beta\mathbf{u}) = (\alpha\beta)\mathbf{u}$
- Identity: $1 \cdot \mathbf{u} = \mathbf{u}$
- Distributivity over vector addition: $\alpha(\mathbf{u} + \mathbf{v}) = \alpha\mathbf{u} + \alpha\mathbf{v}$
- Distributivity over scalar addition: $(\alpha + \beta)\mathbf{u} = \alpha\mathbf{u} + \beta\mathbf{u}$

Example: $\mathbb{R}^n$ is a vector space, and a typical vector is written as $\mathbf{v} = (x_1, \ldots, x_n)^\top$.

A subset $W \subseteq V$ is called a **subspace** if $0 \in W$ and W is closed under both addition and scalar multiplication (i.e., $\forall \alpha \in \mathbb{R}, \mathbf{u}, \mathbf{v} \in W$, we have $\alpha\mathbf{u} + \mathbf{v} \in W$).

2.2.2 Vector Norms

A *norm* on $\mathbb{R}^n$ is a function $\|\cdot\| : \mathbb{R}^n \to \mathbb{R}$ that assigns a nonnegative real number to each vector, and satisfies the following properties for all $\mathbf{u}, \mathbf{v} \in \mathbb{R}^n$ and scalars $\alpha \in \mathbb{R}$:

1. **Non-negativity:** $\|\mathbf{u}\| \geq 0$, and $\|\mathbf{u}\| = 0$ if and only if $\mathbf{u} = \mathbf{0}$.
2. **Homogeneity (absolute scalability):** $\|\alpha\,\mathbf{u}\| = |\alpha|\,\|\mathbf{u}\|$.
3. **Triangle inequality:** $\|\mathbf{u} + \mathbf{v}\| \leq \|\mathbf{u}\| + \|\mathbf{v}\|$.

2.2.3 Distances Induced by Norms

Given a norm $\|\cdot\|$ on $\mathbb{R}^n$, we can define a corresponding *distance function* (or *metric*) d by

$$d(\mathbf{x}, \mathbf{y}) \;=\; \|\mathbf{x} - \mathbf{y}\|.$$

This distance measures how "far apart" two points (vectors) $\mathbf{x}$ and $\mathbf{y}$ are under the chosen norm. It satisfies the usual distance axioms:

- $d(\mathbf{x}, \mathbf{y}) \geq 0$, and $d(\mathbf{x}, \mathbf{y}) = 0$ if and only if $\mathbf{x} = \mathbf{y}$.
- $d(\mathbf{x}, \mathbf{y}) = d(\mathbf{y}, \mathbf{x})$ (symmetry).
- $d(\mathbf{x}, \mathbf{z}) \leq d(\mathbf{x}, \mathbf{y}) + d(\mathbf{y}, \mathbf{z})$ (triangle inequality).

Examples of Norms and Distances

- $\boldsymbol{L^1}$ Norm (Manhattan / Taxicab Norm).

  $$\|\mathbf{u}\|_1 \;=\; \sum_{i=1}^{n} |u_i|.$$

  Distance induced by $\|\cdot\|_1$:

  $$d_1(\mathbf{x}, \mathbf{y}) \;=\; \|\mathbf{x} - \mathbf{y}\|_1 \;=\; \sum_{i=1}^{n} |x_i - y_i|.$$

  This corresponds to traveling along coordinate axes (like navigating city blocks).

- $\boldsymbol{L^2}$ Norm (Euclidean Norm).

  $$\|\mathbf{u}\|_2 \;=\; \sqrt{\mathbf{u} \cdot \mathbf{u}} \;=\; \sqrt{\sum_{i=1}^{n} u_i^2}.$$

  Distance induced by $\|\cdot\|_2$:

  $$d_2(\mathbf{x}, \mathbf{y}) \;=\; \|\mathbf{x} - \mathbf{y}\|_2 \;=\; \sqrt{\sum_{i=1}^{n} (x_i - y_i)^2}.$$

  This is the usual Euclidean distance.

- $\boldsymbol{L^p}$ Norm (General $\boldsymbol{p}$-Norm). For $1 \leq p < \infty$, the *p-norm* is

  $$\|\mathbf{u}\|_p \;=\; \Big(\sum_{i=1}^{n} |u_i|^p\Big)^{1/p}.$$

  Distance induced by $\|\cdot\|_p$:

  $$d_p(\mathbf{x}, \mathbf{y}) \;=\; \|\mathbf{x} - \mathbf{y}\|_p \;=\; \Big(\sum_{i=1}^{n} |x_i - y_i|^p\Big)^{1/p}.$$

  Both $\|\cdot\|_1$ and $\|\cdot\|_2$ are special cases of this general form.

- $\boldsymbol{L^\infty}$ Norm (Max Norm).
$$\|\mathbf{u}\|_\infty \;=\; \max_{1\le i\le n} |u_i|.$$
Distance induced by $\|\cdot\|_\infty$:
$$d_\infty(\mathbf{x},\mathbf{y}) \;=\; \|\mathbf{x}-\mathbf{y}\|_\infty \;=\; \max_{1\le i\le n} |x_i-y_i|.$$
Geometrically, this measures the *largest* coordinate difference between $\mathbf{x}$ and $\mathbf{y}$.

Equivalence of Norms in Finite Dimensions In $\mathbb{R}^n$, all of these norms (and indeed *any* norms) are *equivalent* in the sense that if you pick any two norms $\|\cdot\|_a$ and $\|\cdot\|_b$, there exist constants $c_1, c_2 > 0$ (depending on n but not on the vector itself) such that
$$c_1\|\mathbf{u}\|_a \;\le\; \|\mathbf{u}\|_b \;\le\; c_2\|\mathbf{u}\|_a,$$
for all $\mathbf{u}\in\mathbb{R}^n$. This means they induce the same notion of *topology* (i.e. the same idea of which points are "close" to which), and there is no norm that fundamentally changes the notion of "distance" in finite-dimensional spaces.

Example in $\mathbb{R}^2$ with $\mathbf{u}=(2,3)$ and $\mathbf{v}=(1,-1)$

- $\|\mathbf{u}\|_1 = |2|+|3| = 5.$
- $\|\mathbf{u}\|_2 = \sqrt{2^2+3^2} = \sqrt{13}.$
- $\|\mathbf{u}\|_\infty = \max\{\,|2|,\ |3|\} = 3.$

Distances between $\mathbf{u}$ and $\mathbf{v}$:
$$d_1(\mathbf{u},\mathbf{v}) = \|\mathbf{u}-\mathbf{v}\|_1 = |2-1|+|3-(-1)| = 1+4 = 5,$$
$$d_2(\mathbf{u},\mathbf{v}) = \|\mathbf{u}-\mathbf{v}\|_2 = \sqrt{(2-1)^2+(3+1)^2} = \sqrt{1^2+4^2} = \sqrt{17},$$
$$d_\infty(\mathbf{u},\mathbf{v}) = \|\mathbf{u}-\mathbf{v}\|_\infty = \max\{|2-1|,\ |3-(-1)|\} = \max\{1,4\} = 4.$$
All three metrics satisfy the distance axioms, but they measure "distance" in different ways.

2.2.4 Orthogonal and Orthonormal Vectors

- Two vectors $\mathbf{u}$ and $\mathbf{v}$ in $\mathbb{R}^n$ are *orthogonal* if $\mathbf{u}\cdot\mathbf{v}=0$.
- A set of vectors $\{\mathbf{u}_1,\mathbf{u}_2,\ldots,\mathbf{u}_k\}$ is *orthogonal* if each pair of distinct vectors is orthogonal:
$$\mathbf{u}_i\cdot\mathbf{u}_j = 0 \quad\text{for}\quad i\ne j.$$
- A set of vectors is *orthonormal* if it is orthogonal and each vector has norm 1:
$$\|\mathbf{u}_i\| = 1 \quad\text{for all } i, \quad\text{and}\quad \mathbf{u}_i\cdot\mathbf{u}_j = 0 \quad (i\ne j).$$

Example (in $\mathbb{R}^2$): Notice that
$$\mathbf{u}\cdot\mathbf{w} = (2)(3)+(3)(-2) = 6-6 = 0,$$
so $\mathbf{u}$ and $\mathbf{w}$ are orthogonal in $\mathbb{R}^2$. Neither is a unit vector, though. We can normalize each to obtain an orthonormal pair:
$$\widehat{\mathbf{u}} = \frac{1}{\sqrt{13}}\begin{pmatrix}2\\3\end{pmatrix}, \quad \widehat{\mathbf{w}} = \frac{1}{\sqrt{13}}\begin{pmatrix}3\\-2\end{pmatrix}.$$
It is then straightforward to verify:
$$\widehat{\mathbf{u}}\cdot\widehat{\mathbf{w}} = 0, \quad \|\widehat{\mathbf{u}}\| = 1, \quad \|\widehat{\mathbf{w}}\| = 1.$$

2.2.5 Angle Between Vectors

If $\mathbf{u}$ and $\mathbf{v}$ are nonzero vectors, the angle θ between them is given by

$$\mathbf{u} \cdot \mathbf{v} = \|\mathbf{u}\|\|\mathbf{v}\| \cos\theta,$$

which implies

$$\theta \;=\; \arccos\!\left(\frac{\mathbf{u} \cdot \mathbf{v}}{\|\mathbf{u}\|\|\mathbf{v}\|}\right).$$

- $\theta = 0$ when $\mathbf{u}$ and $\mathbf{v}$ point in the same direction.
- $\theta = \pi/2$ (i.e. 90°) when $\mathbf{u} \perp \mathbf{v}$ (orthogonal).
- $0 \le \theta \le \pi$ in $\mathbb{R}^n$.

Example (in $\mathbb{R}^2$):

$$\mathbf{u} \cdot \mathbf{v} = -1, \quad \|\mathbf{u}\| = \sqrt{13}, \quad \|\mathbf{v}\| = \sqrt{2}.$$

Hence

$$\theta = \arccos\!\left(\frac{-1}{\sqrt{13}\sqrt{2}}\right) = \arccos\!\left(\frac{-1}{\sqrt{26}}\right).$$

Numerically, θ lies between 90° and 180° since the dot product is negative.

2.2.6 Vector Projection

The *projection* of a vector $\mathbf{u}$ onto a nonzero vector $\mathbf{v}$ is defined as

$$\operatorname{proj}_{\mathbf{v}}(\mathbf{u}) \;=\; \left(\frac{\mathbf{u} \cdot \mathbf{v}}{\mathbf{v} \cdot \mathbf{v}}\right)\mathbf{v}.$$

- **Geometric Interpretation:** $\operatorname{proj}_{\mathbf{v}}(\mathbf{u})$ is the component of $\mathbf{u}$ in the direction of $\mathbf{v}$. In other words, it is the *closest point* (in the sense of Euclidean distance) on the line spanned by $\mathbf{v}$ to $\mathbf{u}$.
- **If $\mathbf{v}$ is a unit vector:** If $\mathbf{v}$ has length 1, denoted $\widehat{\mathbf{v}}$, then

$$\operatorname{proj}_{\widehat{\mathbf{v}}}(\mathbf{u}) \;=\; (\mathbf{u} \cdot \widehat{\mathbf{v}})\,\widehat{\mathbf{v}}.$$

- **Rejection:** The *rejection* of $\mathbf{u}$ from $\mathbf{v}$ (the component of $\mathbf{u}$ orthogonal to $\mathbf{v}$) is given by

$$\mathbf{u} - \operatorname{proj}_{\mathbf{v}}(\mathbf{u}).$$

Example (in $\mathbb{R}^2$):

$$\mathbf{u} \cdot \mathbf{v} = -1, \quad \mathbf{v} \cdot \mathbf{v} = 1^2 + (-1)^2 = 2.$$

So

$$\operatorname{proj}_{\mathbf{v}}(\mathbf{u}) = \left(\frac{-1}{2}\right)\mathbf{v} = \begin{pmatrix} -\frac{1}{2} \\ \frac{1}{2} \end{pmatrix}.$$

Hence the rejection is

$$\mathbf{u} - \operatorname{proj}_{\mathbf{v}}(\mathbf{u}) = \begin{pmatrix} 2 \\ 3 \end{pmatrix} - \begin{pmatrix} -\frac{1}{2} \\ \frac{1}{2} \end{pmatrix} = \begin{pmatrix} 2 + \frac{1}{2} \\ 3 - \frac{1}{2} \end{pmatrix} = \begin{pmatrix} \frac{5}{2} \\ \frac{5}{2} \end{pmatrix}.$$

2.2.7 Linear Independence, Span, and Basis

- **Linear Independence:** A set of vectors $\{\mathbf{u}_1, \dots, \mathbf{u}_k\}$ in $\mathbb{R}^n$ is said to be *linearly independent* if the only solution to
$$\alpha_1\mathbf{u}_1 + \alpha_2\mathbf{u}_2 + \cdots + \alpha_k\mathbf{u}_k = \mathbf{0}$$
is $\alpha_1 = \alpha_2 = \cdots = \alpha_k = 0$. Equivalently, no vector in the set can be expressed as a linear combination of the others.

- **Span:** The *span* of $\{\mathbf{u}_1, \dots, \mathbf{u}_k\}$ is the set of all linear combinations
$$\alpha_1\mathbf{u}_1 + \cdots + \alpha_k\mathbf{u}_k,$$
for all scalars $\alpha_1, \dots, \alpha_k$. The span represents all possible points you can reach by scaling and adding these vectors.

- **Basis:** A set of vectors in $\mathbb{R}^n$ is called a *basis* if it is linearly independent *and* its span is the entire space $\mathbb{R}^n$.
  - The number of vectors in any basis of $\mathbb{R}^n$ is n, which is called the *dimension* of $\mathbb{R}^n$.

- **Orthonormal Basis:** If a set of n linearly independent vectors in $\mathbb{R}^n$ is also orthonormal (i.e. each pair of distinct vectors is orthogonal, and each has norm 1), then it forms an *orthonormal basis* for $\mathbb{R}^n$. Such bases simplify many computations because dot products, norms, and projections have especially simple forms under orthonormal sets.

Example (in $\mathbb{R}^2$):

- *Linear independence:* The vectors $\mathbf{u} = \begin{pmatrix}2\\3\end{pmatrix}$ and $\mathbf{v} = \begin{pmatrix}1\\-1\end{pmatrix}$ are linearly independent because no scalar multiple of one can produce the other.

- *Span:* The span of $\{\mathbf{u}, \mathbf{v}\}$ in $\mathbb{R}^2$ is all of $\mathbb{R}^2$ if $\mathbf{u}$ and $\mathbf{v}$ are not collinear. Indeed, any vector $\mathbf{x} = (x_1, x_2)^\top \in \mathbb{R}^2$ can be written as a combination of $\mathbf{u}$ and $\mathbf{v}$ if their determinant (thinking of them as columns) is nonzero.

- *Basis:* In $\mathbb{R}^2$, any pair of linearly independent vectors forms a basis for $\mathbb{R}^2$. Since $\{\mathbf{u}, \mathbf{v}\}$ are independent, they form a basis of $\mathbb{R}^2$.

- *Orthonormal basis:* The normalized vectors $\widehat{\mathbf{u}}$ and $\widehat{\mathbf{w}}$ above (where $\mathbf{u} \perp \mathbf{w}$) form an orthonormal set. If you had two such orthonormal, independent vectors in $\mathbb{R}^2$, they would automatically form an orthonormal basis for $\mathbb{R}^2$.

2.3 Vector Bases

2.3.1 Standard Basis

The *standard basis* for $\mathbb{R}^n$ is the set of n vectors

$$\mathbf{e}_1 = \begin{pmatrix}1\\0\\\vdots\\0\end{pmatrix}, \quad \mathbf{e}_2 = \begin{pmatrix}0\\1\\\vdots\\0\end{pmatrix}, \quad \dots \quad, \mathbf{e}_n = \begin{pmatrix}0\\0\\\vdots\\1\end{pmatrix}.$$

Each $\mathbf{e}_i$ has a 1 in the i-th coordinate and 0 elsewhere. The collection $\{\mathbf{e}_1, \mathbf{e}_2, \dots, \mathbf{e}_n\}$ has several important properties:

- **Any vector as a combination:** Any $\mathbf{u} \in \mathbb{R}^n$ can be written uniquely as

$$\mathbf{u} \;=\; u_1\mathbf{e}_1 + u_2\mathbf{e}_2 + \cdots + u_n\mathbf{e}_n.$$

  In coordinate form, $\mathbf{u} = (u_1, u_2, \ldots, u_n)^\top$. This reflects the fact that each $\mathbf{e}_i$ "picks out" the i-th coordinate of $\mathbf{u}$.

- **Relationship to the identity matrix:** If we place the vectors $\mathbf{e}_1, \ldots, \mathbf{e}_n$ as the columns of an $n \times n$ matrix, we obtain the *identity matrix* I_n:

$$I_n \;=\; \begin{pmatrix} | & | & & | \\ \mathbf{e}_1 & \mathbf{e}_2 & \cdots & \mathbf{e}_n \\ | & | & & | \end{pmatrix}.$$

  Consequently, multiplying any vector $\mathbf{u}$ by I_n leaves $\mathbf{u}$ unchanged, illustrating the "identity" property in matrix form.

- **Orthonormality:** The standard basis vectors form an *orthonormal* set because $\mathbf{e}_i \cdot \mathbf{e}_j = 0$ for $i \neq j$ and $\|\mathbf{e}_i\| = 1$ for all i. Consequently, the standard basis is a key example of an orthonormal basis for $\mathbb{R}^n$.

- **Dimension:** Because $\{\mathbf{e}_1, \ldots, \mathbf{e}_n\}$ spans $\mathbb{R}^n$ (as above) and contains n vectors, it constitutes a *basis*, and also tells us that the dimension of $\mathbb{R}^n$ is n.

- **Coordinate extraction:** Note that the i-th coordinate of $\mathbf{u}$ can be found via the dot product:

$$u_i \;=\; \mathbf{u} \cdot \mathbf{e}_i.$$

  In effect, each $\mathbf{e}_i$ "samples" exactly the i-th component of any vector in $\mathbb{R}^n$.

Example (in $\mathbb{R}^2$):

$$\mathbf{e}_1 = \begin{pmatrix} 1 \\ 0 \end{pmatrix}, \quad \mathbf{e}_2 = \begin{pmatrix} 0 \\ 1 \end{pmatrix}.$$

Any vector $\mathbf{u} = (u_1, u_2)^\top$ can be written as

$$\mathbf{u} = u_1\mathbf{e}_1 + u_2\mathbf{e}_2 = \begin{pmatrix} u_1 \\ u_2 \end{pmatrix}.$$

This construction extends naturally to $\mathbb{R}^n$, where each vector is a sum of coordinate-scaled basis vectors.

2.3.2 Other Notable Bases

While the *standard basis* is often the first (and simplest) example for $\mathbb{R}^n$, there are many other important bases in linear algebra. Below are some commonly encountered alternatives, each with its own purpose and advantages.

Eigenbasis (Diagonalizing a Matrix) If A is an $n \times n$ matrix that has n linearly independent eigenvectors, then those eigenvectors form what is called an *eigenbasis* of $\mathbb{R}^n$. Concretely, suppose A has n distinct (or at least enough) eigenvectors $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n$ such that

$$A\,\mathbf{x}_j = \lambda_j \mathbf{x}_j \quad \text{for } j = 1, 2, \ldots, n,$$

where each λ_j is the corresponding eigenvalue. If you arrange these eigenvectors as columns of a matrix P, i.e.

$$P = \begin{pmatrix} | & | & & | \\ \mathbf{x}_1 & \mathbf{x}_2 & \cdots & \mathbf{x}_n \\ | & | & & | \end{pmatrix},$$

then $P^{-1}AP = D$ is a diagonal matrix whose diagonal entries are the eigenvalues $\lambda_1, \lambda_2, \ldots, \lambda_n$. In this basis, the matrix A acts particularly simply:

$$A\left(\alpha_1\,\mathbf{x}_1 + \cdots + \alpha_n\mathbf{x}_n\right) = \alpha_1\lambda_1\mathbf{x}_1 + \cdots + \alpha_n\lambda_n\mathbf{x}_n.$$

Advantages:

- Diagonalizing a matrix greatly simplifies many operations, like taking powers A^k.
- Each component in the new coordinate system (the eigenbasis) evolves independently.

Note: Not every matrix is diagonalizable. In such cases, one may use the Jordan normal form, leading to a *Jordan basis* instead (discussed briefly below).

Jordan Basis (Jordan Normal Form) When A is not diagonalizable but still has a full set of n eigenvalues (counting multiplicities) over the complex field, one can put it into a *Jordan normal form.* This involves creating a *Jordan basis* that consists of generalized eigenvectors, arranged so that A takes the Jordan block structure when viewed in this basis:

$$P^{-1}AP = \begin{pmatrix} J_1 & 0 & \cdots & 0 \\ 0 & J_2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & J_k \end{pmatrix},$$

where each J_i is a Jordan block corresponding to one eigenvalue λ_i. Each Jordan block is almost diagonal, except for 1's on the superdiagonal.
Advantages:

- Allows one to understand the structure of a matrix that is not diagonalizable.
- Provides a standard form J for similarity classes of matrices.

Orthonormal Bases in $\mathbb{R}^n$ (via Gram-Schmidt) Any basis $\{\mathbf{u}_1, \ldots, \mathbf{u}_n\}$ of $\mathbb{R}^n$ can be *orthonormalized* by the *Gram-Schmidt* procedure, leading to a set of vectors $\{\mathbf{q}_1, \ldots, \mathbf{q}_n\}$ such that:

$$\mathbf{q}_i \cdot \mathbf{q}_j = 0 \quad (i \neq j), \quad \text{and} \quad \|\mathbf{q}_i\| = 1 \quad \text{for all } i.$$

This new basis diagonalizes many transformations (especially symmetric matrices) in a more geometric manner and simplifies dot-product-related computations.
Advantages:

- Dot products and projections are very straightforward in an orthonormal basis.
- Symmetric matrices are particularly easy to handle: if A is real symmetric, one can always find an orthonormal eigenbasis (via the *Spectral Theorem*).

Bases for Function Spaces (e.g., Fourier Basis) Beyond $\mathbb{R}^n$, one can consider infinite-dimensional vector spaces, like spaces of functions. For instance:

- **Fourier Basis:** In the space of square-integrable functions on $[0, 2\pi]$, $\big\{\sin(kx), \cos(kx)\big\}_{k=0}^{\infty}$ (plus constants) can form a basis (technically, an *orthonormal set* in a Hilbert space).
- **Polynomial Spaces:** For the space of all polynomials up to degree n, the set $\{1, x, x^2, \ldots, x^n\}$ is the standard basis. One could also use *Legendre polynomials* or other orthogonal polynomials as a basis, which simplifies certain integrals.

Advantages:

- Often, changing to an appropriate basis (e.g. a Fourier basis) turns complicated differential equations or integrals into simpler forms.
- Orthonormal polynomials are extremely useful for approximations and in solution methods for boundary value problems.

Other Coordinate Systems in $\mathbb{R}^n$ Even in finite-dimensional spaces, one might pick a basis that aligns with certain geometric or problem-specific features:

- **Rotated / shifted bases:** Instead of aligning with the $x_1, x_2, \ldots, x_n$ axes, we might pick directions that match the geometry of a problem or the constraints of a system.
- **Basis for subspaces:** In many applications (e.g. solutions to a system of linear equations), one is more interested in a *subspace* (like the row space, column space, or null space of a matrix). Finding a basis for such a subspace is often the key step.

In summary, *any* set of n linearly independent vectors in $\mathbb{R}^n$ forms a valid basis, but choosing a **good** basis—eigenvectors, orthonormal vectors, special polynomials, etc.—can greatly simplify the problem at hand.

2.4 Special Vectors

2.4.1 Unit Vectors

A *unit vector* is any vector of length 1. Any nonzero vector $\mathbf{u}$ can be *normalized* to a unit vector by

$$\widehat{\mathbf{u}} \;=\; \frac{\mathbf{u}}{\|\mathbf{u}\|}.$$

- $\|\widehat{\mathbf{u}}\| = 1$.
- The direction of $\widehat{\mathbf{u}}$ is the same as that of $\mathbf{u}$, but its length is 1.

Example (in $\mathbb{R}^2$):

$$\mathbf{u} = \begin{pmatrix} 2 \\ 3 \end{pmatrix}, \quad \|\mathbf{u}\| = \sqrt{13}, \quad \widehat{\mathbf{u}} = \frac{1}{\sqrt{13}} \begin{pmatrix} 2 \\ 3 \end{pmatrix}.$$

Then $\|\widehat{\mathbf{u}}\| = 1$.

2.4.2 Eigenvalues and Eigenvectors

Let A be an $n \times n$ matrix over $\mathbb{R}$. A (nonzero) vector $\mathbf{x} \in \mathbb{R}^n$ is called an *eigenvector* of A if there exists a scalar $\lambda \in \mathbb{R}$, called the *eigenvalue*, such that

$$A\,\mathbf{x} = \lambda\,\mathbf{x}.$$

Equivalently, $\mathbf{x}$ is an eigenvector if it satisfies

$$(A - \lambda I)\mathbf{x} = \mathbf{0},$$

where I is the $n \times n$ identity matrix. For $\mathbf{x}$ to be an eigenvector, it must be nonzero, and λ must make the above equation hold.

- **Eigenvalue equation:** $A\,\mathbf{x} = \lambda\mathbf{x}$.
  - $\mathbf{x}$ is the eigenvector (required to be nonzero).
  - λ is the corresponding eigenvalue.
- **Characteristic polynomial:** To find possible eigenvalues λ, one typically solves

  $$\det(A - \lambda I) = 0.$$

  This determinant (a polynomial in λ) is called the *characteristic polynomial* of A.
- **Geometric interpretation:** If $\mathbf{x}$ is an eigenvector of A with eigenvalue λ, then A acts on $\mathbf{x}$ by simply *stretching* or *shrinking* it (and possibly reversing its direction if $\lambda < 0$), without changing its direction (except possibly reversing when λ is negative).

- **Eigenspace:** For each eigenvalue λ, the set of all eigenvectors with that eigenvalue, together with the zero vector, forms a *subspace* of $\mathbb{R}^n$, known as the *eigenspace* corresponding to λ:

$$E_\lambda = \{\mathbf{x} \in \mathbb{R}^n : A\,\mathbf{x} = \lambda\mathbf{x}\}.$$

Example (in $\mathbb{R}^2$): Consider the matrix

$$A = \begin{pmatrix} 3 & 2 \\ 2 & 3 \end{pmatrix}.$$

To find its eigenvalues, solve

$$\det\bigl(A - \lambda I\bigr) = \det\begin{pmatrix} 3-\lambda & 2 \\ 2 & 3-\lambda \end{pmatrix} = (3-\lambda)(3-\lambda) - 2\cdot 2 = (3-\lambda)^2 - 4 = 0.$$

Hence, $(3-\lambda)^2 = 4$, so $3 - \lambda = \pm 2$. This yields two eigenvalues:

$$\lambda_1 = 1, \quad \lambda_2 = 5.$$

To find an eigenvector for $\lambda_1 = 1$, solve $(A - I)\mathbf{x} = \mathbf{0}$:

$$\begin{pmatrix} 3-1 & 2 \\ 2 & 3-1 \end{pmatrix}\begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} 2 & 2 \\ 2 & 2 \end{pmatrix}\begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}.$$

This implies $2x + 2y = 0$, or $x = -\,y$. We can choose (for instance) $x = 1$, $y = -1$, so

$$\mathbf{x}_1 = \begin{pmatrix} 1 \\ -1 \end{pmatrix}$$

is an eigenvector for $\lambda_1 = 1$.

Similarly, for $\lambda_2 = 5$, solve $(A - 5I)\mathbf{x} = \mathbf{0}$:

$$\begin{pmatrix} 3-5 & 2 \\ 2 & 3-5 \end{pmatrix} = \begin{pmatrix} -2 & 2 \\ 2 & -2 \end{pmatrix}.$$

This matrix enforces $-2x + 2y = 0$, or $x = y$. Choosing $x = 1$, $y = 1$ gives

$$\mathbf{x}_2 = \begin{pmatrix} 1 \\ 1 \end{pmatrix}$$

as an eigenvector for $\lambda_2 = 5$.

Thus, the eigenvalues of A are 1 and 5, with corresponding eigenvectors $\begin{pmatrix} 1 \\ -1 \end{pmatrix}$ and $\begin{pmatrix} 1 \\ 1 \end{pmatrix}$, respectively.

3 Matrices

Let A be an $m \times n$ matrix and B be an $n \times p$ matrix. Then

$$A = \begin{pmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mn} \end{pmatrix},$$

$$B = \begin{pmatrix} b_{11} & b_{12} & \cdots & b_{1p} \\ b_{21} & b_{22} & \cdots & b_{2p} \\ \vdots & \vdots & \ddots & \vdots \\ b_{n1} & b_{n2} & \cdots & b_{np} \end{pmatrix}.$$

3.1 Operations on Matrices

3.1.1 Matrix Addition and Scalar Multiplication

$$A + C = \begin{pmatrix} a_{11}+c_{11} & \cdots & a_{1n}+c_{1n} \\ \vdots & \ddots & \vdots \\ a_{m1}+c_{m1} & \cdots & a_{mn}+c_{mn} \end{pmatrix}, \quad \alpha A = \begin{pmatrix} \alpha a_{11} & \cdots & \alpha a_{1n} \\ \vdots & \ddots & \vdots \\ \alpha a_{m1} & \cdots & \alpha a_{mn} \end{pmatrix}.$$

- A and C must be the same dimension for $A + C$ to be defined.
- Commutative: $A + C = C + A$.

3.1.2 Matrix Multiplication

$$AB = \begin{pmatrix} \sum_{k=1}^n a_{1k}b_{k1} & \sum_{k=1}^n a_{1k}b_{k2} & \cdots & \sum_{k=1}^n a_{1k}b_{kp} \\ \sum_{k=1}^n a_{2k}b_{k1} & \sum_{k=1}^n a_{2k}b_{k2} & \cdots & \sum_{k=1}^n a_{2k}b_{kp} \\ \vdots & \vdots & \ddots & \vdots \\ \sum_{k=1}^n a_{mk}b_{k1} & \sum_{k=1}^n a_{mk}b_{k2} & \cdots & \sum_{k=1}^n a_{mk}b_{kp} \end{pmatrix}.$$

- AB is defined only if the number of columns of A equals the number of rows of B.
- *Not commutative in general*: $AB \neq BA$ (unless A and B have special forms or dimensions).
- *Associative*: $(AB)C = A(BC)$.

3.1.3 Matrix–Vector Multiplication

Let A be an $m \times n$ matrix and $\mathbf{x}$ an $n \times 1$ column vector:

$$A\mathbf{x} = \begin{pmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mn} \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{pmatrix} = \begin{pmatrix} a_{11}x_1 + a_{12}x_2 + \cdots + a_{1n}x_n \\ a_{21}x_1 + a_{22}x_2 + \cdots + a_{2n}x_n \\ \vdots \\ a_{m1}x_1 + a_{m2}x_2 + \cdots + a_{mn}x_n \end{pmatrix}.$$

- The result is an $m \times 1$ vector.
- $(A\mathbf{x})^\top = \mathbf{x}^\top A^\top$.

3.2 Properties of Matrices

3.2.1 Trace of a Matrix

The *trace* of a square matrix $A \in \mathbb{R}^{n\times n}$, denoted $\operatorname{tr}(A)$ (or just $\operatorname{tr}A$ if the parentheses are clear), is the sum of its diagonal elements:

$$\operatorname{tr}(A) = \sum_{i=1}^{n} a_{ii}.$$

The trace has the following properties:

- *Symmetry:* For $A \in \mathbb{R}^{n\times n}$, $\operatorname{tr}(A) = \operatorname{tr}(A^\top)$.
- *Linearity:* For $A, B \in \mathbb{R}^{n\times n}$, $\operatorname{tr}(A+B) = \operatorname{tr}(A) + \operatorname{tr}(B)$, and for any scalar α, $\operatorname{tr}(\alpha A) = \alpha \operatorname{tr}(A)$.
- *Cyclicity* for two factors: If AB is square, then $\operatorname{tr}(AB) = \operatorname{tr}(BA)$.
- *Cyclicity* for three (or more) factors: If ABC is square, then $\operatorname{tr}(ABC) = \operatorname{tr}(BCA) = \operatorname{tr}(CAB)$, and similarly for products of any number of matrices.

Example proof of $\operatorname{tr}(AB) = \operatorname{tr}(BA)$:

Suppose $A \in \mathbb{R}^{m\times n}$ and $B \in \mathbb{R}^{n\times m}$, so that $AB \in \mathbb{R}^{m\times m}$ and $BA \in \mathbb{R}^{n\times n}$ are both square. Then

$$\operatorname{tr}(AB) \;=\; \sum_{i=1}^{m}(AB)_{ii} \;=\; \sum_{i=1}^{m}\sum_{j=1}^{n} A_{ij}\, B_{ji} \;=\; \sum_{j=1}^{n}\sum_{i=1}^{m} B_{ji}\, A_{ij} \;=\; \sum_{j=1}^{n}(BA)_{jj} \;=\; \operatorname{tr}(BA).$$

Thus we see $\operatorname{tr}(AB) = \operatorname{tr}(BA)$.

3.2.2 Matrix Transpose

$$A^\top = \begin{pmatrix} a_{11} & a_{21} & \cdots & a_{m1} \\ a_{12} & a_{22} & \cdots & a_{m2} \\ \vdots & \vdots & \ddots & \vdots \\ a_{1n} & a_{2n} & \cdots & a_{mn} \end{pmatrix}.$$

- $(A^\top)^\top = A$.
- $(AB)^\top = B^\top A^\top$ (reversing the order of multiplication when transposing a product).

3.2.3 Matrix Inverse

If A is an $n \times n$ invertible (non-singular) matrix, A^{-1} is defined by:

$$AA^{-1} = A^{-1}A = I_n,$$

where I_n is the $n \times n$ identity matrix.

- Not every square matrix is invertible (must have nonzero determinant).
- *Inverse of a Product:* $(AB)^{-1} = B^{-1}A^{-1}$ (again, note how the order is reversed).
- $(A^{-1})^\top = (A^\top)^{-1}$ (transpose and inverse also reverse).

Inverse of a 2x2 matrix: $A = \begin{pmatrix} a & b \\ c & d \end{pmatrix}, \quad A^{-1} = \dfrac{1}{ad-bc}\begin{pmatrix} d & -b \\ -c & a \end{pmatrix}.$

General formula via adjugate: $A^{-1} = \dfrac{1}{\det(A)}\operatorname{adj}(A),$

where $\operatorname{adj}(A)$ is the *adjugate* (or classical adjoint) of A, obtained by taking the transpose of the cofactor matrix of A.

3.2.4 Symmetric & Hermitian Matrices

A matrix $A \in \mathbb{R}^{n\times n}$ is *symmetric* if $A = A^\top$. Over the complex field, A is *Hermitian* if $A = A^*$ (the conjugate transpose).

- Symmetric real matrices have real eigenvalues and can be diagonalized by an orthogonal matrix (spectral theorem).
- Covariance matrices in statistics are by definition symmetric (and positive semidefinite).

3.2.5 Skew-Symmetric (Antisymmetric) Matrices

A square matrix $A \in \mathbb{R}^{n\times n}$ is called *skew–symmetric* (or *antisymmetric*) if

$$A^\top = -A.$$

Equivalently, every diagonal element of a real skew–symmetric matrix is zero and the entries satisfy $a_{ij} = -a_{ji}$.

Example (2×2).

$$A = \begin{pmatrix} 0 & -a \\ a & 0 \end{pmatrix}, \quad a \in \mathbb{R}.$$

Key properties

- **Purely imaginary eigenvalues.** All eigenvalues of a real skew–symmetric matrix occur in conjugate pairs $\pm i\lambda$ ($\lambda \in \mathbb{R}$). Hence $\det(A) \geq 0$, and if n is odd then $\det(A) = 0$.
- **Orthogonal exponentials.** For any real $A = -A^\top$, the matrix exponential $Q(t) = e^{tA}$ is orthogonal and has determinant 1 (it lies in $\mathrm{SO}(n)$). Thus skew–symmetric matrices are infinitesimal generators of rotations.
- **Zero quadratic form.** For every $x \in \mathbb{R}^n$, $x^\top Ax = 0$. Consequently skew–symmetric matrices are always *indefinite* in the sense of quadratic forms.
- **Dimensionality.** The vector space of $n \times n$ real skew–symmetric matrices has dimension $\frac{n(n-1)}{2}$, exactly the number of distinct pairs (i, j) with $i < j$.
- **Pfaffian.** When n is even, $\det(A) = \mathrm{pf}(A)^2$, where $\mathrm{pf}(A)$ is the *Pfaffian* of A.
- **Relation to cross / wedge products.** In $\mathbb{R}^3$, every $A = -A^\top$ corresponds to a unique vector ω such that $Ax = \omega \times x$ for all x; conversely, $\omega \mapsto [\omega]_\times$ embeds $\mathbb{R}^3$ into the space of 3×3 skew–symmetric matrices.

3.2.6 Definite Matrices

We define the following properties for symmetric matrices $A \in \mathbb{S}^n$:

- **Positive Definite (PD):** If for all non-zero vectors $x \in \mathbb{R}^n$, $x^TAx > 0$. Denoted as $A \succ 0$ or $A > 0$. The set of all positive definite matrices is often denoted $\mathbb{S}^n_{++}$.
- **Positive Semidefinite (PSD):** If for all vectors $x^TAx \geq 0$. Denoted as $A \succeq 0$ or $A \geq 0$. The set is often denoted $\mathbb{S}^n_{+}$.
- **Negative Definite (ND):** If for all non-zero $x \in \mathbb{R}^n$, $x^TAx < 0$. Denoted as $A \prec 0$ or $A < 0$.
- **Negative Semidefinite (NSD):** If for all $x \in \mathbb{R}^n$, $x^TAx \leq 0$. Denoted as $A \preceq 0$ or $A \leq 0$.
- **Indefinite:** If neither PSD nor NSD - i.e., if there exist $x_1, x_2 \in \mathbb{R}^n$ such that $x_1^TAx_1 > 0$ and $x_2^TAx_2 < 0$.

Properties of Definite Matrices

- If A is positive definite, then $-A$ is negative definite and vice versa.
- If A is positive semidefinite, then $-A$ is negative semidefinite and vice versa.
- If A is indefinite, then $-A$ is also indefinite.
- Positive definite and negative definite matrices are always full rank and invertible.
- For any matrix $A \in \mathbb{R}^{m\times n}$, the Gram matrix $G = A^T A$ is always positive semidefinite.
- If $m \geq n$ and A is full rank, then $G = A^T A$ is positive definite.

Proof of Full Rank for Definite Matrices To see why definite matrices are full rank, suppose matrix $A \in \mathbb{R}^{n\times n}$ is not full rank. Then some column j can be expressed as a linear combination of other columns:

$$a_j = \sum_{i\neq j} x_i a_i$$

for some $x_1, \ldots, x_{j-1}, x_{j+1}, \ldots, x_n \in \mathbb{R}$. Setting $x_j = -1$, we have:

$$Ax = \sum_{i=1}^{n} x_i a_i = 0$$

But this implies $x^T Ax = 0$ for some non-zero vector x, so A must be neither positive definite nor negative definite.

3.2.7 Quadratic Forms

Given a square matrix $A \in \mathbb{R}^{n\times n}$ and a vector $x \in \mathbb{R}^n$, the scalar value $x^T Ax$ is called a quadratic form. Written explicitly:

$$x^T Ax = \sum_{i=1}^{n} x_i (Ax)_i = \sum_{i=1}^{n} x_i \left(\sum_{j=1}^{n} A_{ij} x_j \right) = \sum_{i=1}^{n} \sum_{j=1}^{n} A_{ij} x_i x_j .$$

Note that:

$$x^T Ax = (x^T Ax)^T = x^T A^T x = x^T \left(\frac{1}{2} A + \frac{1}{2} A^T \right) x,$$

where the first equality follows from the fact that the transpose of a scalar is equal to itself, and the second equality follows from averaging two equal quantities. From this, we can conclude that only the symmetric part of A contributes to the quadratic form.

3.2.8 Orthogonal (or Orthonormal) Matrices

An *orthogonal matrix* $Q \in \mathbb{R}^{n\times n}$ satisfies $Q^\top Q = QQ^\top = I_n$. In complex spaces, the analogous property is for *unitary* matrices U where $U^* U = I_n$.

- Orthogonal matrices preserve vector norms: $\|Q\mathbf{x}\|_2 = \|\mathbf{x}\|_2$.
- The inverse of an orthogonal matrix is its transpose ($Q^{-1} = Q^\top$).

3.2.9 Linear Independence and Rank

A set of vectors $\{x_1, x_2, \ldots, x_n\} \subset \mathbb{R}^m$ is said to be (linearly) independent if no vector can be represented as a linear combination of the remaining vectors. Conversely, if one vector belonging to the set can be represented as a linear combination of the remaining vectors, then the vectors are said to be (linearly) dependent. That is, if

$$x_n = \sum_{i=1}^{n-1} \alpha_i x_i$$

for some scalar values $\alpha_1, \ldots, \alpha_{n-1} \in \mathbb{R}$, then we say that the vectors $x_1, \ldots, x_n$ are linearly dependent; otherwise, the vectors are linearly independent. For example, the vectors

$$x_1 = \begin{pmatrix} 1 \\ 2 \\ 3 \end{pmatrix}, \quad x_2 = \begin{pmatrix} 4 \\ 1 \\ 5 \end{pmatrix}, \quad x_3 = \begin{pmatrix} 2 \\ -3 \\ -1 \end{pmatrix}$$

are linearly dependent because $x_3 = -2x_1 + x_2$.

The column rank of a matrix $A \in \mathbb{R}^{m \times n}$ is the size of the largest subset of columns of A that constitute a linearly independent set. With some abuse of terminology, this is often referred to simply as the number of linearly independent columns of A. In the same way, the row rank is the largest number of rows of A that constitute a linearly independent set.

For any matrix $A \in \mathbb{R}^{m \times n}$, it turns out that the column rank of A is equal to the row rank of A (though we will not prove this), and so both quantities are referred to collectively as the rank of A, denoted as $\text{rank}(A)$. The following are some basic properties of the rank:

- For $A \in \mathbb{R}^{m \times n}$, $\text{rank}(A) \leq \min(m, n)$. If $\text{rank}(A) = \min(m, n)$, then A is said to be full rank.
- For $A \in \mathbb{R}^{m \times n}$, $\text{rank}(A) = \text{rank}(A^T)$.
- For $A \in \mathbb{R}^{m \times n}, B \in \mathbb{R}^{n \times p}$, $\text{rank}(AB) \leq \min(\text{rank}(A), \text{rank}(B))$.
- For $A, B \in \mathbb{R}^{m \times n}$, $\text{rank}(A + B) \leq \text{rank}(A) + \text{rank}(B)$.

3.2.10 Rank Preserving Operations

In linear algebra, **rank preserving operations** are operations that do **not change the rank** of a matrix. The *rank* of a matrix is the dimension of its column space (or row space), i.e., the maximum number of linearly independent rows or columns.

1. **Elementary Row Operations**
   These operations are used in Gaussian elimination and do not change the rank:
   - Swapping two rows.
   - Multiplying a row by a non-zero scalar.
   - Adding a scalar multiple of one row to another.

   Row operations preserve the *row rank*, and since row rank equals column rank, they preserve the overall rank.

2. **Multiplication by an Invertible Matrix**
   If A is a matrix and P, Q are invertible matrices, then:

   $$\text{rank}(PA) = \text{rank}(A), \quad \text{rank}(AQ) = \text{rank}(A)$$

   Left multiplication changes the row space basis, and right multiplication changes the column space basis. Neither affects the rank.

3. **Transposition**
   Transposing a matrix does not change its rank:
   $$\text{rank}(A^T) = \text{rank}(A)$$
4. **Similarity Transformation (for square matrices)**
   If P is invertible and A is a square matrix:
   $$\text{rank}(P^{-1}AP) = \text{rank}(A)$$

3.2.11 Non-Rank-Preserving Operations

For contrast, here are some operations that **can change the rank**:

- Multiplying by a non-invertible matrix.
- Zeroing out rows or columns.
- Projecting onto a lower-dimensional subspace.

3.3 Special Matrices

3.3.1 Identity Matrix

The *identity matrix* I_n is an $n \times n$ matrix with 1's on the main diagonal and 0's elsewhere:

$$I_n = \begin{pmatrix} 1 & 0 & \cdots & 0 \\ 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 1 \end{pmatrix}.$$

- For any $n \times n$ matrix A, $AI_n = I_nA = A$.
- The identity matrix is its own inverse: $I_n^{-1} = I_n$.

3.3.2 All-Ones (Unit) Matrix

Sometimes referred to as the *unit matrix of ones* (not to be confused with the identity matrix), the $m \times n$ all-ones matrix is denoted by $J_{m\times n}$ or simply J when dimensions are clear:

$$J_{m\times n} = \begin{pmatrix} 1 & 1 & \cdots & 1 \\ 1 & 1 & \cdots & 1 \\ \vdots & \vdots & \ddots & \vdots \\ 1 & 1 & \cdots & 1 \end{pmatrix}_{m\times n}.$$

- $J_{m\times n}$ has every entry equal to 1.
- When $m = n$, $J_{n\times n}$ can be an eigenvector/eigenvalue example: one eigenvalue is n (with eigenvector $\mathbf{1} = (1, 1, \ldots, 1)^\top$), and the others are 0.

3.3.3 Diagonal Matrix

A matrix $D \in \mathbb{R}^{n\times n}$ is called *diagonal* if $d_{ij} = 0$ for all $i \neq j$:

$$D = \begin{pmatrix} d_1 & 0 & \cdots & 0 \\ 0 & d_2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & d_n \end{pmatrix}.$$

- Diagonal matrices commute: $D_1D_2 = D_2D_1$ because multiplication is elementwise for the diagonal entries.
- Inverses and powers of a diagonal matrix are easy to compute (assuming diagonal entries are nonzero).

3.3.4 Jacobian Matrix

If $\mathbf{f} : \mathbb{R}^n \to \mathbb{R}^m$ is a vector-valued function $\mathbf{f}(\mathbf{x}) = (f_1(\mathbf{x}), \ldots, f_m(\mathbf{x}))^\top$, then the *Jacobian matrix* $J(\mathbf{f})(\mathbf{x})$ is an $m \times n$ matrix whose (i, j)th entry is $\partial f_i/\partial x_j$. Symbolically,

$$J(\mathbf{f})(\mathbf{x}) = \begin{pmatrix} \frac{\partial f_1}{\partial x_1} & \cdots & \frac{\partial f_1}{\partial x_n} \\ \vdots & \ddots & \vdots \\ \frac{\partial f_m}{\partial x_1} & \cdots & \frac{\partial f_m}{\partial x_n} \end{pmatrix}.$$

3.3.5 Hessian Matrix

If $f : \mathbb{R}^n \to \mathbb{R}$ is a scalar-valued function, the *Hessian matrix* $H_f(\mathbf{x})$ is the $n \times n$ matrix of second partial derivatives:

$$H_f(\mathbf{x}) = \begin{pmatrix} \frac{\partial^2 f}{\partial x_1^2} & \frac{\partial^2 f}{\partial x_1 \partial x_2} & \cdots & \frac{\partial^2 f}{\partial x_1 \partial x_n} \\ \frac{\partial^2 f}{\partial x_2 \partial x_1} & \frac{\partial^2 f}{\partial x_2^2} & \cdots & \frac{\partial^2 f}{\partial x_2 \partial x_n} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial^2 f}{\partial x_n \partial x_1} & \frac{\partial^2 f}{\partial x_n \partial x_2} & \cdots & \frac{\partial^2 f}{\partial x_n^2} \end{pmatrix}.$$

Under suitable smoothness conditions (Schwarz's theorem), mixed partials are equal, so $H_f(\mathbf{x})$ is symmetric.

3.3.6 Covariance Matrix

A covariance matrix $C \in \mathbb{R}^{n \times n}$ is defined for a random vector $\mathbf{X} \in \mathbb{R}^n$ by

$$C = \mathbb{E}\big[(\mathbf{X} - \mathbb{E}[\mathbf{X}])(\mathbf{X} - \mathbb{E}[\mathbf{X}])^\top\big].$$

- C is always symmetric positive semidefinite.
- If all eigenvalues of C are strictly positive, C is positive definite.
- Frequently appears in statistics, Gaussian distributions, and kernel methods in ML.

3.3.7 Toeplitz Matrices

A *Toeplitz matrix* has constant diagonals:

$$T = \begin{pmatrix} t_0 & t_{-1} & t_{-2} & \cdots & t_{-(n-1)} \\ t_1 & t_0 & t_{-1} & \cdots & t_{-(n-2)} \\ t_2 & t_1 & t_0 & \cdots & t_{-(n-3)} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ t_{n-1} & t_{n-2} & t_{n-3} & \cdots & t_0 \end{pmatrix}$$

- Arise in signal processing and time series analysis
- Can be multiplied by a vector in $O(n \log n)$ time using FFT

3.3.8 Circulant Matrices

A *circulant matrix* is a special Toeplitz matrix where each row is a cyclic shift of the previous row:

$$C = \begin{pmatrix} c_0 & c_{n-1} & c_{n-2} & \cdots & c_1 \\ c_1 & c_0 & c_{n-1} & \cdots & c_2 \\ c_2 & c_1 & c_0 & \cdots & c_3 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ c_{n-1} & c_{n-2} & c_{n-3} & \cdots & c_0 \end{pmatrix}$$

- Diagonalized by the DFT matrix
- Used in circular convolution

3.3.9 Vandermonde Matrices

A *Vandermonde matrix* has geometric progression in its columns:

$$V = \begin{pmatrix} 1 & x_1 & x_1^2 & \cdots & x_1^{n-1} \\ 1 & x_2 & x_2^2 & \cdots & x_2^{n-1} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & x_n & x_n^2 & \cdots & x_n^{n-1} \end{pmatrix}$$

- Determinant: $\det(V) = \prod_{1 \leq i < j \leq n}(x_j - x_i)$
- Used in polynomial interpolation

3.3.10 Block Matrices

A *block matrix* is partitioned into submatrices:

$$M = \begin{pmatrix} A & B \\ C & D \end{pmatrix}$$

- Block matrix multiplication follows the same rules as regular matrix multiplication
- Schur complement: $M/A = D - CA^{-1}B$ (if A is invertible)
- Determinant formula: $\det(M) = \det(A)\det(M/A)$

3.3.11 Hankel Matrices

A *Hankel matrix* has constant anti-diagonals:

$$H = \begin{pmatrix} h_0 & h_1 & h_2 & \cdots & h_{n-1} \\ h_1 & h_2 & h_3 & \cdots & h_n \\ h_2 & h_3 & h_4 & \cdots & h_{n+1} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ h_{n-1} & h_n & h_{n+1} & \cdots & h_{2n-2} \end{pmatrix}$$

- Used in system identification
- Related to Toeplitz matrices via permutation

4 Tensors

A *tensor* is an N-dimensional array that generalizes scalars (0-D), vectors (1-D), and matrices (2-D). We denote an order-N tensor by calligraphic letters $(\mathcal{X}, \mathcal{Y}, \dots)$ and write

$$\mathcal{X} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}, \qquad x_{i_1, i_2, \dots, i_N} \in \mathbb{R}.$$

Running Example (order 3)

Throughout this chapter we use

$$\mathcal{X} \in \mathbb{R}^{2\times 2\times 2}, \quad \begin{array}{l} \mathcal{X}(:,:,1) = \begin{pmatrix} 1 & 3 \\ 2 & 4 \end{pmatrix}, \\ \mathcal{X}(:,:,2) = \begin{pmatrix} 5 & 7 \\ 6 & 8 \end{pmatrix}. \end{array}$$

Individual entries are $x_{1,2,1} = 3,\ x_{2,2,2} = 8$, etc. (See Figure **??** for a sketch if desired.)

4.1 Operations on Tensors

4.1.1 Element-wise Operations and Scalar Multiplication

For $\mathcal{X}, \mathcal{Y} \in \mathbb{R}^{I_1 \times \cdots \times I_N}$ and $\alpha \in \mathbb{R}$

$$(\mathcal{X} + \mathcal{Y})_{i_1,\dots,i_N} = x_{i_1,\dots,i_N} + y_{i_1,\dots,i_N}, \qquad (\alpha\mathcal{X})_{i_1,\dots,i_N} = \alpha\, x_{i_1,\dots,i_N}.$$

All vector-space axioms (commutativity, associativity, distributivity, ...) carry over index-wise.

Example. With the running $\mathcal{X}$ and $\mathcal{Y}(:,:,1) = \left(\begin{smallmatrix} 0 & 1 \\ 1 & 0 \end{smallmatrix}\right)$, $\mathcal{Y}(:,:,2) = \left(\begin{smallmatrix} 1 & 0 \\ 0 & 1 \end{smallmatrix}\right)$,

$$(\mathcal{X} + \mathcal{Y})(:,:,1) = \begin{pmatrix} 1 & 4 \\ 3 & 4 \end{pmatrix}, \qquad 2\mathcal{X}(:,:,2) = \begin{pmatrix} 10 & 14 \\ 12 & 16 \end{pmatrix}.$$

4.1.2 Fibers, Slices, and Matricization

- **Mode-n fiber:** Fix all indices except i_n. Example: the column-mode fiber $\mathbf{x}_{:11} = \big(x_{1,1,1},\, x_{2,1,1}\big)^\top = (1,2)^\top$ in the running tensor.
- **Slice:** Fix all but two indices, e.g. $\mathcal{X}(:,:,2)$ above.
- **Matricization / unfolding:** Rearrange $\mathcal{X}$ into a matrix $X_{(n)} \in \mathbb{R}^{I_n \times (I_1 \cdots I_{n-1} I_{n+1} \cdots I_N)}$ by stacking all mode-n fibers as columns. In the example

$$X_{(1)} = \begin{pmatrix} 1 & 5 & 3 & 7 \\ 2 & 6 & 4 & 8 \end{pmatrix}.$$

4.1.3 Mode-n (Matrix) Multiplication

For $\mathcal{X} \in \mathbb{R}^{I_1 \times \cdots \times I_N}$ and $A^{(n)} \in \mathbb{R}^{J \times I_n}$, the mode-$n$ product is

$$\mathcal{Y} = \mathcal{X} \times_n A^{(n)} \iff Y_{(n)} = A^{(n)}\, X_{(n)}.$$

The result lives in $I_1 \times \cdots \times I_{n-1} \times J \times I_{n+1} \times \cdots \times I_N$.

Example (mode 1). With $A^{(1)} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \\ 1 & 1 \end{pmatrix} \in \mathbb{R}^{3\times 2}$,

$$\mathcal{Y} = \mathcal{X} \times_1 A^{(1)} \in \mathbb{R}^{\boxed{3}\times 2\times 2}, \qquad Y_{(1)} = A^{(1)} X_{(1)} = \begin{pmatrix} 1 & 5 & 3 & 7 \\ 2 & 6 & 4 & 8 \\ 3 & 11 & 7 & 15 \end{pmatrix}.$$

4.1.4 Tensor Contraction and the Frobenius Inner Product

If $\mathcal{X}, \mathcal{Y} \in \mathbb{R}^{I_1 \times \cdots \times I_N}$ share the same shape, their *Frobenius (inner) product* is

$$\langle \mathcal{X}, \mathcal{Y} \rangle = \sum_{i_1,\ldots,i_N} x_{i_1,\ldots,i_N}\, y_{i_1,\ldots,i_N},$$

yielding the *Frobenius norm* $\|\mathcal{X}\|_F = \sqrt{\langle \mathcal{X}, \mathcal{X} \rangle}$.

Example. $\langle \mathcal{X}, \mathcal{X} \rangle = 1^2 + 2^2 + \cdots + 8^2 = 204, \quad \|\mathcal{X}\|_F = \sqrt{204}.$

More general *tensor contractions* sum over *chosen* index pairs and encompass matrix multiplication, trace, and dot products as special cases.

4.1.5 Outer Product and Rank-1 Tensors

For vectors $\mathbf{a}^{(1)} \in \mathbb{R}^{I_1}, \ldots, \mathbf{a}^{(N)} \in \mathbb{R}^{I_N}$ define the *outer product*

$$\mathbf{a}^{(1)} \circ \cdots \circ \mathbf{a}^{(N)} \;\in\; \mathbb{R}^{I_1 \times \cdots \times I_N}, \qquad x_{i_1,\ldots,i_N} = a^{(1)}_{i_1} \cdots a^{(N)}_{i_N}.$$

Such a tensor has (CP) *rank 1*.

4.1.6 Tensor Rank and CP Decomposition

The **rank** of $\mathcal{X}$ is the minimal R so that

$$\mathcal{X} = \sum_{r=1}^{R} \mathbf{a}^{(1)}_r \circ \mathbf{a}^{(2)}_r \circ \cdots \circ \mathbf{a}^{(N)}_r \quad \text{(CANDECOMP/PARAFAC)}.$$

Unlike matrices, tensor rank is *NP-hard* to compute and may exceed $\min I_n$.

Example sketch. For the running $\mathcal{X}$ one finds $R = 2$ with suitable component vectors (exercise).

4.1.7 Tucker Decomposition and HOSVD

Tucker writes a tensor as a (usually small) *core* multiplied along each mode:

$$\mathcal{X} = \mathcal{G} \times_1 U^{(1)} \times_2 U^{(2)} \times_3 U^{(3)}.$$

Choosing $U^{(n)}$ orthogonal and $\mathcal{G}$ all-orthogonal gives the **Higher-Order SVD (HOSVD)**. Tucker separates *inter-mode* (core) from *intra-mode* (factor) interactions—an analogue of the matrix SVD.

4.1.8 Kronecker and Khatri–Rao Products

For matrices $A \in \mathbb{R}^{I \times J}$, $B \in \mathbb{R}^{K \times L}$:

$$A \otimes B = \begin{pmatrix} a_{11}B & \cdots & a_{1J}B \\ \vdots & \ddots & \vdots \\ a_{I1}B & \cdots & a_{IJ}B \end{pmatrix}, \text{ size } IK \times JL.$$

The **Khatri–Rao** product $A \odot B$ keeps matching columns: $(A \odot B)_{:,j} = A_{:,j} \otimes B_{:,j}$. These appear in vectorized tensor equations, e.g. $\operatorname{vec}\big(\mathcal{X} \times_3 U^{(3)}\big) = (U^{(3)} \otimes I_{I_2})\, \operatorname{vec}(X_{(3)})$.

4.2 Applications in Machine Learning and Data Science

- **Deep-learning back-ends.** Frameworks like PyTorch treat `torch.Tensor` as the primary data object; convolutional feature maps are 4-D tensors ($N \times C \times H \times W$).
- **Recommender systems.** User–item–time data $\Rightarrow$ CP or Tucker compression for temporal recommendation.
- **Graph representation learning.** Multi-relational graphs (subject, relation, object) encode as 3-way tensors; factorization yields TransE, DistMult, RESCAL, . . .
- **Computer vision.** Color videos are 4-D tensors (height, width, channel, frame); low-rank approximations reduce memory in video codecs.
- **Scientific computing.** Quantum many-body states (tensor-network methods), uncertainty quantification with polynomial-chaos tensors, and solution operators for high-dimensional PDEs.

With these ingredients the tensor chapter now mirrors the structure, notation, and example-driven approach of the vector chapter. Happy compiling!

5 Special Operations and Rules

5.1 Ordering Rules

Take-away: *Transposing or inverting a product reverses the order of its factors.*

Transpose

$$(AB)^\top = B^\top A^\top, \tag{1}$$
$$(\mathbf{u}\,\mathbf{v}^\top)^\top = \mathbf{v}\,\mathbf{u}^\top, \tag{2}$$
$$(A^{-1})^\top = (A^\top)^{-1}. \tag{3}$$

Inverse

$$(AB)^{-1} = B^{-1}A^{-1}, \tag{4}$$
$$(A_1A_2\cdots A_k)^{-1} = A_k^{-1}\cdots A_2^{-1}A_1^{-1}. \tag{5}$$

Why? The rules follow directly from the defining properties of transpose and inverse:

- $(AB)^\top$ is the unique matrix satisfying $\langle AB\mathbf{x}, \mathbf{y}\rangle = \langle \mathbf{x}, (AB)^\top\mathbf{y}\rangle$.
- $(AB)^{-1}$ is the unique matrix such that $(AB)(AB)^{-1} = I$; multiply out and use associativity.

Example 1 (Transpose of a product). Let $A \in \mathbb{R}^{m\times n}$ and $B \in \mathbb{R}^{n\times p}$. Then

$$(AB)^\top = B^\top A^\top \in \mathbb{R}^{p\times m}.$$

Example 2 (Inverse of a product). For $A, B \in \mathrm{GL}(m)$ we have $(AB)^{-1} = B^{-1}A^{-1}$.

Example 3 (Associativity vs. commutativity). Given $\mathbf{x} \in \mathbb{R}^p$,

$$(AB)\mathbf{x} = A(B\mathbf{x}) \quad \text{but} \quad \mathbf{x}(AB) \text{ is undefined.}$$

5.2 Trace Tricks

The trace operator is *cyclic* and *linear*:

$$\mathrm{tr}(ABC) = \mathrm{tr}(BCA) = \mathrm{tr}(CAB), \tag{6}$$
$$\mathrm{tr}(A^\top) = \mathrm{tr}(A), \tag{7}$$
$$\mathrm{tr}(\alpha A + \beta B) = \alpha\ \mathrm{tr}(A) + \beta\ \mathrm{tr}(B). \tag{8}$$

These identities shorten many derivations in machine-learning proofs, e.g. when reducing nested summations to a single line.

5.3 Vectorisation and the Kronecker Product

For $A \in \mathbb{R}^{m\times n}$, define

$$\mathrm{vec}(A) := \begin{bmatrix} a_{:,1} \\ \vdots \\ a_{:,n} \end{bmatrix} \in \mathbb{R}^{mn}.$$

A cornerstone identity is

$$\mathrm{vec}(ABC) = (C^\top \otimes A)\ \mathrm{vec}(B),$$

where $\otimes$ denotes the Kronecker product. It converts matrix equations into large linear systems and underpins many second-order optimisation methods.

5.4 Associativity Reminder

Matrix multiplication is associative:

$$(AB)C = A(BC),$$

provided the dimensions match. Remember, *associative* $\neq$ *commutative.* Missing this distinction is a common source of bugs in code.

5.5 Determinant Rules

Take-away: *Determinants turn products into products of scalars and ignore transposes.*

Key identities

$$\det(AB) = \det(A)\,\det(B), \tag{9}$$

$$\det(A^\top) = \det(A), \tag{10}$$

$$\det(A^{-1}) = \det(A)^{-1}, \tag{11}$$

$$\det(A + \mathbf{u}\mathbf{v}^\top) = \det(A)\big(1 + \mathbf{v}^\top A^{-1}\mathbf{u}\big) \quad \text{(matrix-determinant lemma)}. \tag{12}$$

Why? Multiplicativity of det follows from the Leibniz formula and multilinearity of rows. The lemma is a one-rank update proven by expanding a block determinant.

Worked Example

For $A \in \mathrm{GL}(n)$ and vectors $\mathbf{u}, \mathbf{v} \in \mathbb{R}^n$,

$$\det\!\big(A + \mathbf{u}\mathbf{v}^\top\big) = \det(A)\big(1 + \mathbf{v}^\top A^{-1}\mathbf{u}\big),$$

useful in Gaussian-process log-likelihoods where A is large but $A^{-1}\mathbf{u}$ is cached.

5.6 Rank, Trace, and Eigenvalue Facts

Take-away: *Rank bounds, trace, and determinant encode eigenvalue information.*

Snapshot

$$\operatorname{rank}(AB) \le \min\!\big(\operatorname{rank} A, \operatorname{rank} B\big), \tag{13}$$

$$\operatorname{rank}(A + \mathbf{u}\mathbf{v}^\top) = \operatorname{rank}(A) + \begin{cases} 0, & \mathbf{v}^\top A^\dagger \mathbf{u} = -1, \\ 1, & \text{otherwise}, \end{cases} \tag{14}$$

$$\operatorname{tr}(A) = \sum_{i=1}^{n} \lambda_i, \tag{15}$$

$$\det(A) = \prod_{i=1}^{n} \lambda_i, \tag{16}$$

$$\lambda(AB) = \lambda(BA) \quad \text{(same multiset)}. \tag{17}$$

Why? Rank is the dimension of the image; composing two linear maps cannot enlarge it. Trace and determinant coincide with the first and last elementary symmetric polynomials of the eigenvalues. AB and BA share the same characteristic polynomial because $\det(\lambda I - AB) = \det(\lambda I - BA)$.

5.7 Norm Equivalences

Take-away: *The Frobenius norm is a trace; the spectral norm is an eigenvalue.*

Formulas

$$\|A\|_F^2 = \operatorname{tr}\bigl(A^\top A\bigr), \tag{18}$$

$$\|A\|_2 = \sqrt{\lambda_{\max}\bigl(A^\top A\bigr)}, \tag{19}$$

$$\|A\|_F = \|\operatorname{vec}(A)\|_2. \tag{20}$$

Why? Diagonalise $A^\top A = Q\Lambda Q^\top$ with Q orthogonal. The eigenvalues λ_i are squares of singular values σ_i, so $\|A\|_F^2 = \sum_i \sigma_i^2 = \operatorname{tr}(\Lambda)$ and $\|A\|_2 = \max_i \sigma_i$.

5.8 Projection and Idempotent Matrices

Take-away: *Orthogonal projections are symmetric idempotents.*

Given $A \in \mathbb{R}^{m\times n}$ with full column rank, the matrix

$$P := A\bigl(A^\top A\bigr)^{-1}A^\top$$

satisfies

$$P^2 = P, \qquad P^\top = P.$$

Why? Both properties follow by straightforward multiplication and transpose rules. P projects any vector onto the column space of A along the orthogonal complement.

Worked Example

In least squares, the fitted values are $\hat{\mathbf{y}} = P\mathbf{y}$; idempotence gives $\hat{\hat{\mathbf{y}}} = \hat{\mathbf{y}}$, a sanity check for regression code.

5.9 Pseudoinverse Relations

Take-away: *The Moore–Penrose pseudoinverse behaves like an inverse on the range and kernel.*

$$\bigl(A^\dagger\bigr)^\dagger = A, \tag{21}$$

$$(A^\top)^\dagger = \bigl(A^\dagger\bigr)^\top, \tag{22}$$

$$AA^\dagger A = A, \qquad A^\dagger A A^\dagger = A^\dagger. \tag{23}$$

Why? All four Moore–Penrose conditions are met by construction via $A = U\Sigma V^\top$ and $A^\dagger = V\Sigma^\dagger U^\top$ in an SVD.

5.10 Block-Matrix Tricks

Take-away: *Schur complements turn block inversion into one small inverse.*

For a block matrix

$$M = \begin{bmatrix} A & B \\ C & D \end{bmatrix}, \qquad A \in \mathbb{R}^{k\times k},$$

with A invertible, the inverse is

$$M^{-1} = \begin{bmatrix} A^{-1} + A^{-1}BS^{-1}CA^{-1} & -A^{-1}BS^{-1} \\ -S^{-1}CA^{-1} & S^{-1} \end{bmatrix}, \quad S := D - CA^{-1}B \quad \text{(Schur complement)}.$$

Why? Multiply the candidate inverse by M and collect block products; the off-diagonal blocks vanish thanks to S.

5.11 SVD Cheatsheet

Take-away: *Singular values unify rank, norms, and conditioning.*

If $A = U\Sigma V^\top$ with singular values $\sigma_1 \geq \cdots \geq \sigma_r > 0$,

$$\operatorname{rank}(A) = r, \tag{24}$$

$$\|A\|_F^2 = \sum_{i=1}^{r} \sigma_i^2, \tag{25}$$

$$\kappa_2(A) = \frac{\sigma_1}{\sigma_r} \quad \text{(spectral condition number).} \tag{26}$$

6 Matrix Calculus and Gradients

6.1 Gradient with Respect to a Vector

Let $f(\mathbf{x})$ be a scalar function, where $\mathbf{x} \in \mathbb{R}^n$. Then the gradient $\nabla_{\mathbf{x}} f(\mathbf{x})$ is a column vector in $\mathbb{R}^n$ whose i-th entry is $\frac{\partial f}{\partial x_i}$:

$$\nabla_{\mathbf{x}} f(\mathbf{x}) = \begin{pmatrix} \frac{\partial f}{\partial x_1} \\ \frac{\partial f}{\partial x_2} \\ \vdots \\ \frac{\partial f}{\partial x_n} \end{pmatrix}.$$

Chain Rule Example. If $\mathbf{y} = A\mathbf{x}$ for $A \in \mathbb{R}^{m \times n}$, and $f(\mathbf{x}) = g(\mathbf{y}) = g(A\mathbf{x})$, then

$$\nabla_{\mathbf{x}} f(\mathbf{x}) = \nabla_{\mathbf{x}}\, g(A\mathbf{x}) = A^\top \nabla_{\mathbf{y}} g(\mathbf{y}).$$

6.2 Gradient with Respect to a Matrix

Let $F(A)$ be a scalar function of the matrix $A \in \mathbb{R}^{m \times n}$. The gradient of F with respect to A is another $m \times n$ matrix whose (i, j)-th entry is $\frac{\partial F}{\partial a_{ij}}$:

$$(\nabla_A F)_{ij} = \frac{\partial F}{\partial a_{ij}}.$$

A useful identity is:

$$\nabla_A \operatorname{tr}(BA) = B^\top,$$

where $\operatorname{tr}(\cdot)$ denotes the trace operator.

6.3 Jacobian

For a vector-valued function $\mathbf{f} : \mathbb{R}^n \to \mathbb{R}^m$, the *Jacobian* is defined as the $m \times n$ matrix whose (i, j)-th entry is $\frac{\partial f_i}{\partial x_j}$. That is,

$$J_{\mathbf{f}}(\mathbf{x}) = \begin{pmatrix} \frac{\partial f_1}{\partial x_1} & \cdots & \frac{\partial f_1}{\partial x_n} \\ \vdots & \ddots & \vdots \\ \frac{\partial f_m}{\partial x_1} & \cdots & \frac{\partial f_m}{\partial x_n} \end{pmatrix}.$$

In linear algebra terms, the Jacobian represents the best linear approximation to $\mathbf{f}$ around a given point $\mathbf{x}$. It is fundamental in multivariate calculus, numerical methods, and sensitivity analysis.

6.4 Hessian

For a scalar function $f : \mathbb{R}^n \to \mathbb{R}$, the *Hessian* is the $n \times n$ matrix of second partial derivatives:

$$H_f(\mathbf{x}) = \begin{pmatrix} \frac{\partial^2 f}{\partial x_1^2} & \cdots & \frac{\partial^2 f}{\partial x_1 \partial x_n} \\ \vdots & \ddots & \vdots \\ \frac{\partial^2 f}{\partial x_n \partial x_1} & \cdots & \frac{\partial^2 f}{\partial x_n^2} \end{pmatrix}.$$

If f is sufficiently smooth, $H_f(\mathbf{x})$ is symmetric. The Hessian captures second-order information about curvature, which is central to second-order optimization techniques (e.g., Newton's method). In linear algebra, analyzing the Hessian helps to identify whether a critical point is a local maximum, local minimum, or saddle point (by examining eigenvalues).

7 Matrix Decompositions

Matrix (or factorization) decompositions are fundamental tools in linear algebra. They allow us to rewrite matrices in ways that reveal their key properties, such as rank, eigenvalues, or condition numbers. Many of these decompositions have direct applications in machine learning, numerical methods, and data analysis.

7.1 Preliminaries and Definitions

- **Eigenvalues and Eigenvectors:** For a square matrix $A \in \mathbb{R}^{n\times n}$, a scalar $\lambda \in \mathbb{R}$ (or $\mathbb{C}$ in the complex case) is an *eigenvalue* if there exists a nonzero vector $\mathbf{v}$ (the *eigenvector*) such that

  $$A\mathbf{v} = \lambda\mathbf{v}.$$

  The set of all eigenvalues forms the *spectrum* (or *spectral set*) of A.

- **Orthogonal (Orthonormal) Basis:** A collection of vectors $\{\mathbf{q}_1, \dots, \mathbf{q}_n\}$ in $\mathbb{R}^n$ is orthonormal if $\mathbf{q}_i^\top \mathbf{q}_j = \delta_{ij}$ (the Kronecker delta). A matrix Q whose columns form an orthonormal set satisfies $Q^\top Q = I$.

- **Diagonal Matrix:** A matrix Λ is *diagonal* if $\Lambda_{ij} = 0$ whenever $i \neq j$. When diagonal entries are real and nonnegative, Λ is sometimes used to list singular values or eigenvalues on the main diagonal.

- **Triangular Matrices:** A matrix L is *lower triangular* if all entries above the main diagonal are zero. Similarly, R (or U) is *upper triangular* if all entries below the main diagonal are zero.

In what follows, we assume basic familiarity with determinants, rank, and vector spaces. Each decomposition has its own requirements (such as symmetry or invertibility).

7.2 Eigen-Decomposition (Diagonalization)

[Eigen-Decomposition Theorem] Let $A \in \mathbb{R}^{n\times n}$ be a square matrix with *n linearly independent* eigenvectors. Then A can be decomposed as

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

where

- V is the $n \times n$ *eigenvector matrix*, whose columns are the eigenvectors of A,
- Λ is the $n \times n$ *diagonal matrix* of eigenvalues (each eigenvalue λ_i appears on the diagonal).

Suppose

$$A = \begin{pmatrix} 4 & 1 \\ 0 & 3 \end{pmatrix}.$$

The eigenvalues are $\lambda_1 = 4, \lambda_2 = 3$. Corresponding eigenvectors might be $\mathbf{v}_1 = (1, 0)^\top$ and $\mathbf{v}_2 = (1, 1)^\top$. Then

$$V = \begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix}, \quad \Lambda = \begin{pmatrix} 4 & 0 \\ 0 & 3 \end{pmatrix}, \quad A = V\Lambda V^{-1}.$$

Symmetric matrices ($A = A^\top$): If A is *real symmetric*, the eigen-decomposition simplifies to an *orthogonal diagonalization*:

$$A = Q\Lambda Q^\top,$$

where Q is an orthogonal matrix ($Q^\top Q = I$) and Λ is diagonal with real eigenvalues. This is particularly important in machine learning for covariance/correlation matrices (e.g. PCA).

7.3 Singular Value Decomposition (SVD)

[Singular Value Decomposition] Any $m \times n$ matrix M (real or complex) can be decomposed as

$$M = U\Sigma V^\top,$$

where

- $U \in \mathbb{R}^{m\times m}$ is orthogonal (i.e. $U^\top U = I_m$),
- $V \in \mathbb{R}^{n\times n}$ is orthogonal (i.e. $V^\top V = I_n$),
- $\Sigma \in \mathbb{R}^{m\times n}$ is *diagonal* in the sense that only the entries along the main diagonal can be nonzero (these entries are the *singular values*).

Usage in ML:

- *Dimensionality Reduction (PCA):* By taking the top r singular values (and corresponding vectors), we obtain a best rank-r approximation to M.
- *Collaborative Filtering:* Large user–item matrices can be approximated by low-rank decompositions.
- *Model Compression:* Weight matrices in neural networks can sometimes be truncated to reduce the number of parameters.

7.4 QR Decomposition

[QR Factorization] If $M \in \mathbb{R}^{m\times n}$ has full column rank (i.e. rank $n \le m$), then M can be written as

$$M = QR,$$

where

- $Q \in \mathbb{R}^{m\times n}$ has orthonormal columns, and
- $R \in \mathbb{R}^{n\times n}$ is upper triangular.

Usage in ML:

- *Least Squares*: Solving $M\mathbf{x} \approx \mathbf{b}$ can be done stably via Q and R.
- *Numerical Optimization*: Many iterative methods for linear or non-linear problems rely on QR for improved numerical stability.

7.5 Cholesky Decomposition

[Cholesky Factorization] A *symmetric positive definite* matrix $A \in \mathbb{R}^{n\times n}$ can be uniquely decomposed as

$$A = LL^\top,$$

where L is lower-triangular with strictly positive diagonal entries. **Usage in ML:**

- *Covariance Matrices*: Cholesky is commonly used to invert or factor covariance matrices in Gaussian processes and Bayesian inference.
- *Sampling*: To sample from $\mathcal{N}(\mathbf{0}, \Sigma)$, we can set $\Sigma = LL^\top$ and let $\mathbf{z} = L\mathbf{x}$ where $\mathbf{x} \sim \mathcal{N}(\mathbf{0}, I)$.

7.6 LU (or LU) Decomposition

[LU Decomposition] If $M \in \mathbb{R}^{n\times n}$ is a square matrix that can be reduced to an upper-triangular form without row swapping, then there exists a decomposition

$$M = LU,$$

where L is lower triangular (with 1's on the diagonal, in one common convention) and U is upper triangular.

- If row interchanges are needed, one introduces a permutation matrix P to get $PM = LU$.
- *Usage:* LU factorization is another approach to solving systems $M\mathbf{x} = \mathbf{b}$ or computing determinants.

7.7 Schur Decomposition

[Schur Decomposition] Any square matrix $A \in \mathbb{C}^{n\times n}$ can be written as

$$A = QTQ^*,$$

where Q is a unitary matrix ($Q^*Q = I$) and T is an upper triangular matrix. The diagonal entries of T are the eigenvalues of A. This result is often used as an intermediate step toward the Jordan normal form or for proving other matrix decompositions.

7.8 Polar Decomposition

Any invertible matrix $A \in \mathbb{R}^{n\times n}$ (or $\mathbb{C}^{n\times n}$) can be written as

$$A = QH,$$

where Q is orthogonal (or unitary) and H is symmetric positive definite (Hermitian positive definite in the complex case).

- *Usage:* The polar decomposition is analogous to converting a complex number z into $|z|e^{i\theta}$. In ML, it sometimes appears in manifold optimization and shape analysis.

7.9 Additional Examples

[2x2 Cholesky] A simple 2×2 positive definite matrix:

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

Its Cholesky decomposition is

$$L = \begin{pmatrix} 2 & 0 \\ 1 & \sqrt{2} \end{pmatrix}, \quad LL^\top = A.$$

[SVD of a rank-1 matrix] Let

$$M = \begin{pmatrix} 2 & 2 \\ 2 & 2 \\ 2 & 2 \end{pmatrix}.$$

This is clearly rank-1 (all rows are multiples of $(2, 2)$). An SVD reveals a single nonzero singular value:

$$\Sigma = \begin{pmatrix} \sqrt{12} & 0 \\ 0 & 0 \\ 0 & 0 \end{pmatrix}, \quad U, V \text{ suitably chosen orthogonal matrices.}$$

[LU Example]

$$M = \begin{pmatrix} 2 & 4 & 2 \\ 4 & 8 & 6 \\ 2 & 6 & 9 \end{pmatrix}.$$

One can find

$$L = \begin{pmatrix} 1 & 0 & 0 \\ 2 & 1 & 0 \\ 1 & \frac{1}{2} & 1 \end{pmatrix}, \quad U = \begin{pmatrix} 2 & 4 & 2 \\ 0 & 0 & 2 \\ 0 & 0 & 4 \end{pmatrix}$$

such that $M = LU$.

7.10 Quick-Reference Guide

Take-away: *Know when a factorisation exists and what it costs.*

Factorisation	Form	Key condition(s)	Typical flop count[†]
Eigen	$A = V\Lambda V^{-1}$	n indep. eigenvectors	$O(n^3)$
Sym. eigen	$A = Q\Lambda Q^\top$	$A = A^\top$ (SPD $\Rightarrow \Lambda > 0$)	$O(n^3)$
Schur	$A = QTQ^*$	always (complex)	$O(n^3)$
SVD (thin)	$A = U_r \Sigma_r V_r^\top$	always	$O(mn \min\{m, n\})$
QR (econ.)	$A = QR$	rank $n \le m$	$O(2mn^2)$
CPQR	$A\Pi = QR$	always	$O(2mn^2)$
Cholesky	$A = LL^\top$	SPD	$O(n^3/3)$
LDL$^\top$	$A = LDL^\top$	$A = A^\top$ (indef. allowed)	$O(n^3/3)$
LU (with P)	$PA = LU$	nonsingular	$O(2n^3/3)$
Hessenberg	$Q^\top AQ = H$	always	$O(10n^3/3)$
Polar	$A = QH$	invertible	$O(n^3)$

[†]Dense matrices, classical algorithms; Krylov or randomised variants can reduce cost.

7.11 LDL$^\top$ Decomposition (Symmetric Indefinite)

Take-away: *Like Cholesky, but works even when A is not positive-definite.*

Factorisation

For $A = A^\top \in \mathbb{R}^{n\times n}$,

$$PAP^\top = LDL^\top, \qquad L \text{ unit lower-triangular, } D \text{ block-diag. with } 1\times 1 \text{ or } 2\times 2 \text{ pivots,}$$

where P is a permutation chosen for numerical stability.

Why? A variant of Gaussian elimination on A preserves symmetry by eliminating both the k-th row and column simultaneously, producing either a 1×1 or 2×2 pivot in D.

Worked Example

In sparse optimisation solvers (e.g. interior-point methods), $K = \nabla^2 f + \mathbf{A}^\top W \mathbf{A}$ is symmetric yet indefinite; an $LDL^\top$ factorisation with pivoting yields a stable Newton step without square-roots.

7.12 Economy-Size and Pivoted QR

Take-away: *"Thin" QR stores only what you need; column pivoting picks the best columns.*

$$\text{(economy)}\ A \in \mathbb{R}^{m\times n},\ m \ge n: \quad A = Q_{m\times n} R_{n\times n}; \tag{27}$$

$$\text{(pivoted)}\ A\Pi = QR, \quad \Pi \text{ permutation s.t. } |R_{11}| \ge |R_{22}| \ge \dots. \tag{28}$$

Why? Pivoting reveals a well-conditioned basis of the column space (rank-revealing QR), crucial in least-squares with ill-conditioned data.

7.13 Truncated SVD and the Eckart–Young–Mirsky Theorem

Take-away: *Keeping the top r singular values minimises $\|A - \tilde{A}\|_2$ and $\|A - \tilde{A}\|_F$.*

Let $A = U\Sigma V^\top$ and define

$$\tilde{A}_r := U_{(:,1:r)}\Sigma_{1:r,1:r}V_{(:,1:r)}^\top.$$

Then for any rank-r matrix B,

$$\|A - \tilde{A}_r\|_2 = \min_{\operatorname{rank}(B)\le r} \|A - B\|_2 \quad \text{and} \quad \|A - \tilde{A}_r\|_F = \min_{\operatorname{rank}(B)\le r} \|A - B\|_F.$$

Why? Both norms are unitarily invariant; removing the smallest singular values reduces energy optimally.

Usage

Dimensionality reduction, latent-semantic analysis, autoencoder weight initialisation, and compressing vision transformers where $r \ll \min\{m, n\}$.

7.14 Hessenberg Decomposition

Take-away: *One Householder sweep brings any matrix nearly triangular—ideal for QR iteration.*

For $A \in \mathbb{R}^{n\times n}$, there exists orthogonal Q such that

$$Q^\top AQ = H, \qquad H_{ij} = 0 \text{ for } i > j + 1.$$

Why? Successively reflect the sub-diagonal entries below the first sub-diagonal; H keeps the same eigenvalues as A but QR steps on H cost only $O(n^2)$ instead of $O(n^3)$.

7.15 Jordan Canonical Form (Conceptual)

Take-away: *Every square matrix is "almost" diagonal—just tolerate the ones on the super-diagonal.*

Over $\mathbb{C}$, $A = VJV^{-1}$ with $J = \operatorname{diag}(J_{k_1}(\lambda_1), \dots)$ and

$$J_k(\lambda) = \begin{bmatrix} \lambda & 1 & & 0 \\ & \lambda & \ddots & \\ & & \ddots & 1 \\ 0 & & & \lambda \end{bmatrix}.$$

Why (but seldom computed)? Gives the most refined structure of the nilpotent part; used in control theory proofs and the definition of the matrix exponential. Numerically, the Schur form is preferred.

7.16 Woodbury Identity (Low-Rank Updates)

Take-away: *Invert a giant matrix by inverting a small one.*

If $A \in \mathbb{R}^{n\times n}$ is invertible and $U \in \mathbb{R}^{n\times k}$, $V \in \mathbb{R}^{k\times n}$,

$$\big(A + UV\big)^{-1} = A^{-1} - A^{-1}U\big(I_k + VA^{-1}U\big)^{-1}VA^{-1}.$$

Why? Apply the block-matrix inversion formula to $\left[\begin{smallmatrix} A & U \\ -V & I \end{smallmatrix}\right]$ and read off the $(1,1)$ block.

Worked Example

In Gaussian-process regression with $n \approx 10^5$ data points, adding $k (\ll n)$ inducing points costs $O(nk^2)$ instead of $O(n^3)$ thanks to Woodbury.

7.17 Polar Decomposition—Computation Tip

Take-away: *Iterative refinement via SVD or Newton makes Q orthogonal to machine precision.*

A practical algorithm: iterate $A_{k+1} = \frac{1}{2}\big(A_k + (A_k^\top)^{-1}\big)$ until convergence; then $Q = A_k$ and $H = Q^\top A$.

8 Numerical Linear Algebra

8.1 Condition Numbers

The *condition number* of a matrix A measures how sensitive the solution of $A\mathbf{x} = \mathbf{b}$ is to small changes in A or $\mathbf{b}$. Formally, we define

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

A large value of $\kappa(A)$ indicates an ill-conditioned problem, meaning that even minor perturbations in A or $\mathbf{b}$ can cause significant changes in the solution $\mathbf{x}$. In the special case of the 2-norm, the condition number $\kappa_2(A)$ can be expressed in terms of the largest and smallest singular values, $\sigma_{\max}$ and $\sigma_{\min}$:

$$\kappa_2(A) = \frac{\sigma_{\max}}{\sigma_{\min}}.$$

8.2 Stability of Algorithms

When we implement numerical algorithms in finite-precision arithmetic, we distinguish between different notions of stability:

- **Backward Stability**: The computed solution is the exact solution of a slightly perturbed version of the original problem. In other words, the algorithm might introduce small changes to A or $\mathbf{b}$, but once those changes are accounted for, the final result is mathematically exact.
- **Forward Stability**: The computed solution itself is close to the exact solution of the original (unperturbed) system.
- **Mixed Stability**: Combines aspects of backward and forward stability, allowing some internal steps to be backward stable while also maintaining forward accuracy where it matters most.

8.3 Floating-Point Arithmetic

All numerical computations must account for the limitations of finite-precision arithmetic. The IEEE 754 standard dictates how real numbers are stored and manipulated, including rules for rounding and handling of overflow/underflow. One key parameter is the *machine epsilon* ϵ, which is the smallest number such that $1 + \epsilon > 1$ in floating-point representation. Because of rounding, small errors can accumulate during matrix operations, sometimes leading to *catastrophic cancellation* in subtraction of nearly equal values. This underscores the importance of algorithmic choices that minimize opportunities for numerical instability.

8.4 Iterative Methods

For many large-scale systems, iterative solvers are preferred over direct methods because they better exploit sparse matrices and can be more memory-efficient.

Stationary Methods. Classical methods such as Jacobi iteration, Gauss–Seidel, and Successive Over-Relaxation (SOR) update the solution vector in place based on the most recent values. For example, the Jacobi method updates each component x_i by isolating it from the rest of the system:

$$x_i^{(k+1)} = \frac{1}{a_{ii}}\Big(b_i - \sum_{j \neq i} a_{ij} x_j^{(k)}\Big).$$

Gauss–Seidel refines each component sequentially using updated values immediately, while SOR introduces a relaxation parameter to potentially speed up convergence.

Krylov Subspace Methods. More advanced iterative methods build their search directions within a sequence of subspaces generated by repeated multiplication of the system matrix. Conjugate Gradient (CG) is a prototypical example for symmetric positive-definite systems, while GMRES and BiCGSTAB address more general, possibly nonsymmetric problems. These approaches can yield faster convergence, especially when combined with effective *preconditioning*.

8.5 Preconditioning

Preconditioning aims to transform a linear system $A\mathbf{x} = \mathbf{b}$ into a more tractable form for iterative solvers. One common strategy is to solve

$$M^{-1}A\mathbf{x} = M^{-1}\mathbf{b},$$

where $M \approx A$ but is easier to invert or factorize. This transformation can significantly reduce the number of iterations needed. Popular preconditioners include incomplete LU (ILU) factorizations, diagonal (Jacobi) scaling, and multigrid methods.

8.6 Parallel Computing Considerations

Finally, large-scale linear algebra computations often run on parallel systems. Designing parallel algorithms requires:

- Distributing data among processing elements,
- Minimizing communication overhead,
- Balancing the computational load,
- Optimizing cache usage.

These considerations help maintain high efficiency and numerical stability at scale.

9 Computational Complexity

9.1 Matrix Multiplication

9.1.1 Classical Algorithm

The standard matrix multiplication algorithm for $n \times n$ matrices has complexity $O(n^3)$:

$$C_{ij} = \sum_{k=1}^{n} A_{ik} B_{kj}$$

9.1.2 Strassen's Algorithm

Strassen's algorithm reduces complexity to $O(n^{\log_2 7}) \approx O(n^{2.81})$:

- Divides matrices into 2x2 blocks
- Uses 7 multiplications instead of 8
- Recursively applies the same strategy

9.1.3 Coppersmith-Winograd Algorithm

Theoretical complexity of $O(n^{2.376})$, but not practical for typical matrix sizes.

9.2 Fast Fourier Transform

- Complexity: $O(n \log n)$ for n-point DFT
- Applications:
  - Polynomial multiplication
  - Convolution
  - Signal processing

9.3 Sparse Matrix Operations

- Storage formats:
  - COO (Coordinate format)
  - CSR (Compressed Sparse Row)
  - CSC (Compressed Sparse Column)
- Matrix-vector multiplication: $O(\text{nnz})$ where nnz is number of nonzeros
- Sparse matrix multiplication: $O(\text{flops})$ where flops is number of floating-point operations

9.4 Parallel Computing

9.4.1 Data Distribution

- Block distribution
- Cyclic distribution
- Block-cyclic distribution

9.4.2 Communication Patterns

- Point-to-point communication
- Collective operations (broadcast, reduce, scatter, gather)
- All-to-all communication

9.5 Cache Optimization

- Blocking/tiling for matrix multiplication
- Cache-oblivious algorithms
- Memory hierarchy considerations

9.6 GPU Computing

- CUDA programming model
- Memory coalescing
- Warp-level parallelism
- Shared memory optimization

9.7 Complexity Classes

- P: Problems solvable in polynomial time
- NP: Problems verifiable in polynomial time
- BPP: Problems solvable by probabilistic algorithms
- NC: Problems solvable in polylogarithmic time with polynomial processors

10 Practical Examples in Machine Learning

In this section, we build upon the linear algebra operations and reordering properties to highlight more sophisticated applications in modern Machine Learning. The focus is on using matrix and tensor operations efficiently and in novel ways to solve complex predictive tasks.

10.1 Low-Rank Factorization for Parameter Compression

For a large weight matrix $W \in \mathbb{R}^{m\times n}$ in a neural network, one can approximate it via SVD truncation:

$$W \approx U\Sigma V^\top,$$

where Σ is kept only for the top r singular values, leading to a rank-r factorization with fewer parameters.

10.2 Kronecker-Factored Approximate Curvature (K-FAC)

In second-order optimization for deep learning, one approximates the Fisher information matrix by a Kronecker product of smaller matrices:

$$F \approx A \otimes B.$$

Then inverting F is simpler because $(A \otimes B)^{-1} = A^{-1} \otimes B^{-1}$. K-FAC can significantly speed up training.

10.3 Tensor Decompositions in Recommender Systems

For a 3rd-order tensor $\mathcal{X} \in \mathbb{R}^{U\times I\times T}$ (user $\times$ item $\times$ time), a CP (CANDECOMP/PARAFAC) decomposition might approximate:

$$\mathcal{X} \approx \sum_{r=1}^{R} \mathbf{u}_r \circ \mathbf{i}_r \circ \mathbf{t}_r,$$

where each $\mathbf{u}_r \in \mathbb{R}^U$, $\mathbf{i}_r \in \mathbb{R}^I$, $\mathbf{t}_r \in \mathbb{R}^T$. Learning $\mathbf{u}_r$, $\mathbf{i}_r$, $\mathbf{t}_r$ is then akin to factorizing multi-way data for predictions.

10.4 Matrix Reordering and Efficient Computation

Large-scale Machine Learning models often rely on high-dimensional matrix multiplications. Strategic reordering (e.g., grouping operations via associativity) can improve computational efficiency and memory usage:

- **Batch Multiplication:** If we have a mini-batch of input vectors $\{\mathbf{x}_i\}$ and a weight matrix W, we can write the batched operation as

$$(W\mathbf{X}) = W \begin{pmatrix} | & | & & | \\ \mathbf{x}_1 & \mathbf{x}_2 & \cdots & \mathbf{x}_b \\ | & | & & | \end{pmatrix},$$

  where $\mathbf{X}$ is an $n \times b$ matrix (each column is an input). Reordering via associativity (grouping multiplications for matrix–matrix multiplication) can yield faster GPU implementations than multiple separate vector–matrix multiplications.

- **Transformer Blocks:** In self-attention mechanisms, we often compute $QK^\top$, where Q and K are query/key matrices of dimension (batch $\times$ sequence length $\times$ d). Certain frameworks rearrange these tensors for GPU-friendliness, taking advantage of the fact that

$$(QK^\top)^\top = KQ^\top$$

  so if we need $(QK^\top)^\top$ for some computation, we can store $KQ^\top$ directly, saving one transpose operation.

10.5 Automating Gradient Computation via Matrix Calculus

Automatic differentiation (AD) tools rely heavily on the chain rule in matrix form. The reordering rules are implicitly used to ensure that each gradient is computed in the correct order:

- **Parameter Updates in Networks:** If $\mathbf{h} = \sigma(W\mathbf{x} + \mathbf{b})$ is a hidden layer (with activation σ), then

$$\frac{\partial \mathbf{h}}{\partial W} = \frac{\partial \sigma(\cdot)}{\partial(W\mathbf{x} + \mathbf{b})} \otimes \mathbf{x}^\top,$$

  which arises from applying the chain rule and vectorizing $\mathbf{x}$. The final partial derivative is then rearranged by the AD system to match the shape of W in memory.

- **Second-Order Methods:** When using approximate second-order updates (e.g., K-FAC), we keep

$$(AB)^{-1} = B^{-1}A^{-1},$$

  in mind to more efficiently invert large Kronecker-factored matrices. This property drastically reduces computation compared to inverting the full Fisher information matrix directly.

10.6 Advanced Tensor Operations in Deep Learning

Beyond matrix multiplication, higher-order tensors appear in various deep learning contexts:

- **Convolution as a Matrix Multiplication:** A convolutional layer can be "unfolded" (or im2col-transformed) into a standard matrix multiplication $W \cdot \tilde{\mathbf{X}}$, where $\tilde{\mathbf{X}}$ is a matrix containing patches extracted from images. Modern libraries may reorder dimensions internally to speed up these operations and facilitate vectorization on GPUs.

- **Attention Weights in Transformers:** The 3D tensor for multi-head attention can be reordered via $(\text{batch}, \text{heads}, \text{seq}, d)$ or $(\text{batch}, \text{seq}, \text{heads}, d)$, etc. Reordering ensures that attention weight calculations (e.g. $QK^\top$ and subsequent softmax) are computed efficiently and remain memory-coalesced.

- **Tensor Factorizations in Multi-Modal Learning:** Extending beyond matrices, if a dataset has multiple modalities (e.g. image, text, and audio), each sample can be viewed as a higher-order tensor. Operations like *Khatri–Rao* or *Kronecker* products may be used to integrate or "merge" these modalities. The reordering properties become crucial to avoid repeated overhead in factorized multi-modal architectures.

- **Benchmark Reorderings:** Even though $(AB)C = A(BC)$, different GPU libraries might run faster with a particular parenthesization. When matrices are large, profiling the order of multiplication is often worthwhile.

- **Keep Track of Batch Dimensions:** In many deep learning frameworks, the first dimension is the batch size. Be mindful of whether the library is optimized for $(\text{batch} \times \dots)$ or $(\cdots \times \text{batch})$ ordering. Correct dimension reordering can yield surprising speed-ups in training and inference.

- **Careful with Transposes:** Repeatedly transposing large matrices or tensors can degrade performance. Try to restructure your graph or code so that transposes happen minimally, possibly by applying known reorder rules $(AB)^\top = B^\top A^\top$ to reduce the total number of transpose operations.

- **Matrix Inversions vs. Factorizations:** Inverting a matrix A directly can be costly. Factorizations such as QR, LU, or Cholesky are usually more stable and faster for solving linear systems $A\mathbf{x} = \mathbf{b}$. This is particularly important in Bayesian ML (e.g. Gaussian Processes), where large covariance matrices must be dealt with frequently.

Cheatsheet

- Vectors are typically denoted by lowercase bold letters, e.g. $\mathbf{x} \in \mathbb{R}^n$.
- Matrices are denoted by uppercase letters, e.g. $A \in \mathbb{R}^{m \times n}$.
- x_i denotes the i-th component of $\mathbf{x}$.
- A_{ij} denotes the (i, j)-th entry of A.
- $\|\mathbf{x}\|$ denotes a norm (often the Euclidean norm unless otherwise specified).
- $\mathbf{x}^\top$ denotes the transpose of vector $\mathbf{x}$.

10.7 Basic Vector/Matrix Products

$$\mathbf{x}^\top \mathbf{y} = \sum_i x_i y_i \qquad \text{(Dot product of two vectors in } \mathbb{R}^n.\text{)}$$

$$(AB)_{ij} = \sum_k A_{ik} B_{kj} \qquad \text{(Matrix multiplication.)}$$

$$(A\mathbf{x})_i = \sum_j A_{ij} x_j \qquad \text{(Matrix-vector multiplication.)}$$

$$(\mathbf{x}^\top A)_j = \sum_i x_i A_{ij} \qquad \text{(Vector-matrix multiplication, row-view.)}$$

$$(A\mathbf{x})^\top = \mathbf{x}^\top A^\top \qquad \text{(Transpose rule.)}$$

10.8 Quadratic Forms

A quadratic form in $\mathbf{x}$ w.r.t. a matrix M is $\mathbf{x}^\top M \mathbf{x}$. This expands to

$$\mathbf{x}^\top M \mathbf{x} = \sum_i \sum_j x_i \, M_{ij} \, x_j,$$

and is a scalar if M is $n \times n$.

Examples:

- If $M = I$ (the identity), then $\mathbf{x}^\top I \mathbf{x} = \mathbf{x}^\top \mathbf{x} = \|\mathbf{x}\|_2^2$.
- For any matrix A, $\|A\mathbf{x}\|_2^2 = (A\mathbf{x})^\top (A\mathbf{x}) = \mathbf{x}^\top A^\top A \mathbf{x}$.

10.9 Affine Functions and Expansions

Consider the function

$$f(\mathbf{x}) = \tfrac{1}{2} \mathbf{x}^\top A \mathbf{x} \; + \; \mathbf{b}^\top \mathbf{x} \; + \; c,$$

where A is an $n \times n$ matrix (often assumed symmetric in many contexts), $\mathbf{b} \in \mathbb{R}^n$, and c is a scalar. We can expand it component-wise:

$$f(\mathbf{x}) = \tfrac{1}{2} \sum_i \sum_j x_i A_{ij} x_j \; + \; \sum_i b_i x_i \; + \; c.$$

10.10 Matrix/Vector Identities and Tricks

Transpose Properties

$$
\begin{aligned}
(A^\top)^\top &= A, \\
(A+B)^\top &= A^\top + B^\top, \\
(AB)^\top &= B^\top A^\top, \\
\mathbf{x}^\top(A\mathbf{y}) &= (A^\top \mathbf{x})^\top \mathbf{y}.
\end{aligned}
$$

Invertibility

- A matrix A is invertible (nonsingular) if $\det(A) \neq 0$.
- $(A^{-1})^\top = (A^\top)^{-1}$.
- If A is orthogonal (i.e. $A^\top A = I$), then $A^{-1} = A^\top$.

Determinant Properties

$$
\begin{aligned}
\det(AB) &= \det(A)\,\det(B), \\
\det(A^\top) &= \det(A), \\
\det(cA) &= c^n \det(A) \quad \text{(for an } n \times n \text{ matrix)}.
\end{aligned}
$$

Rank and Projections

- The rank of A is the dimension of the column space of A.
- Orthogonal projection matrix onto a subspace U spanned by columns of A:

  $$P = A(A^\top A)^{-1} A^\top,$$

  provided A has full column rank.

10.11 Norms and Inner Products

- $\|\mathbf{x}\|_2 = \sqrt{\mathbf{x}^\top \mathbf{x}}$.
- $\|A\|_F = \sqrt{\sum_{i,j} A_{ij}^2} = \sqrt{\operatorname{trace}(A^\top A)}$ (Frobenius norm).
- $\|\mathbf{x}\|_1 = \sum_i |x_i|$.
- Cauchy-Schwarz: $|\mathbf{x}^\top \mathbf{y}| \leq \|\mathbf{x}\|_2 \|\mathbf{y}\|_2$.

10.12 Eigenvalue Decomposition and SVD (Selected Points)

- **Eigenvalue Decomposition**: If $A \in \mathbb{R}^{n \times n}$ is diagonalizable, then

  $$A = Q \Lambda Q^{-1},$$

  where Λ is diagonal of eigenvalues and columns of Q are eigenvectors.
- **Spectral Theorem**: If A is real symmetric,

  $$A = U \Lambda U^\top,$$

  where U is orthogonal ($U^\top U = I$) and Λ is diagonal of real eigenvalues.

- **SVD**: For any $A \in \mathbb{R}^{m\times n}$,
$$A = U\,\Sigma\,V^\top,$$
where $U \in \mathbb{R}^{m\times m}$ and $V \in \mathbb{R}^{n\times n}$ are orthogonal, and Σ is diagonal (padded with zeros if needed) of singular values $\sigma_1 \geq \sigma_2 \geq \dots$.

10.13 Matrix Calculus (Selected Formulas)

$$\begin{aligned}
\frac{\partial}{\partial \mathbf{x}}\bigl(\mathbf{b}^\top \mathbf{x}\bigr) &= \mathbf{b},\\
\frac{\partial}{\partial \mathbf{x}}\bigl(\mathbf{x}^\top A\mathbf{x}\bigr) &= (A + A^\top)\mathbf{x} \quad \text{(if A is symmetric, this simplifies to $2A\mathbf{x}$)},\\
\frac{\partial}{\partial \mathbf{x}}\|\mathbf{x}\|_2^2 &= 2\mathbf{x},\\
\frac{\partial}{\partial \mathbf{x}}(\mathbf{x}^\top \mathbf{x}) &= 2\mathbf{x}.
\end{aligned}$$