1. Numerical error & complexity

Floating point: real numbers stored with finite precision; roundoff errors inevitable.

Machine epsilon: smallest $\varepsilon > 0$ such that $1 + \varepsilon \neq 1$ in floating point.

Floating point and error.

- Double precision: $\varepsilon_{\text{mach}} \approx 2.2 \times 10^{-16}$.
- Relative error: $e_{\text{rel}} = (\hat{x} - x)/x$; absolute error: $e_{\text{abs}} = \hat{x} - x$.
- Avoid subtracting nearly equal numbers (catastrophic cancellation).

Truncation error: error from approximating an infinite process (e.g. Taylor series, step size).

Big-O: $f(n) = O(g(n))$ if $|f(n)| \leq C|g(n)|$ for large n.

Big-O complexity.

- Dot product: $O(n)$.
- Matrix–vector: $O(mn)$.
- Matrix–matrix: $O(mnp)$.
- Dense linear solve: $O(n^3)$.

Example (catastrophic cancellation):

$$\text{Compute } 1 - \cos(x) \text{ for small } x.$$

Naive: $1 - \cos(x)$ loses precision. Better:

$$1 - \cos(x) = 2\sin^2(x/2),$$

which is numerically more stable for small x.

2. Linear algebra essentials

Norm: function $\|\cdot\|$ measuring vector or matrix "size", satisfying positivity, homogeneity, and the triangle inequality.

Condition number: $\kappa(A) = \|A\|\|A^{-1}\|$; measures sensitivity of the solution of $Ax = b$ to perturbations.

Norms & conditioning.

- Vector p-norm: $\|\mathbf{x}\|_p = \left(\sum_i |x_i|^p\right)^{1/p}$.
- Matrix induced norms: $\|A\|_1 = \max_j \sum_i |a_{ij}|$, $\|A\|_\infty = \max_i \sum_j |a_{ij}|$.

Some properties

- Outer product of two non-zero vectors uv^T is always rank 1.
- Product of the eigenvalues is the determinant.
- If columns (or rows) are nearly linearly dependent or have wildly different scales, the matrix is likely ill-conditioned.
- If the Hessian at a critical point has all eigenvalues positive (symmetric case), the Hessian is positive definite and the point is a strict local minimum.

Singular vs nonsingular matrices.

Nonsingular (invertible) $A \in \mathbb{R}^{n\times n}$:

- $\det(A) \neq 0$.
- Inverse exists: $\exists\, A^{-1}$ with $AA^{-1} = A^{-1}A = I$.
- Columns (and rows) are linearly independent.
- Rank $= n$; null space $\mathcal{N}(A) = \{\mathbf{0}\}$.
- For every $\mathbf{b}$, the system $A\mathbf{x} = \mathbf{b}$ has a unique solution $\mathbf{x} = A^{-1}\mathbf{b}$.
- 0 is not an eigenvalue of A.

Singular $A \in \mathbb{R}^{n\times n}$:

- $\det(A) = 0$.
- No inverse exists.
- Columns (or rows) linearly dependent; rank $< n$.
- Nontrivial null space: $\exists\, \mathbf{x} \neq \mathbf{0}$ with $A\mathbf{x} = \mathbf{0}$.
- For some $\mathbf{b}$, $A\mathbf{x} = \mathbf{b}$ has no solution; for others, infinitely many.
- 0 is an eigenvalue: $\exists\, \mathbf{v} \neq \mathbf{0}$ with $A\mathbf{v} = \mathbf{0}$.

Example (check singularity and implications):

$$A = \begin{bmatrix} 1 & 2 \\ 2 & 4 \end{bmatrix}, \quad \det(A) = 1\cdot 4 - 2\cdot 2 = 0 \Rightarrow A \text{ is singular.}$$

So A^{-1} does not exist, columns are multiples, and there are nonzero solutions to $A\mathbf{x} = \mathbf{0}$.

Special matrices (examples and properties).

Identity matrix:

$$I_3 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}, \quad AI_3 = I_3A = A.$$

Eigenvalues are all 1; $\det(I_3) = 1$.

Diagonal matrix:

$$D = \begin{bmatrix} d_1 & 0 & 0 \\ 0 & d_2 & 0 \\ 0 & 0 & d_3 \end{bmatrix}.$$

- $\det(D) = d_1d_2d_3$.
- Eigenvalues are $\lambda_1 = d_1,\ \lambda_2 = d_2,\ \lambda_3 = d_3$.
- $D^{-1} = \text{diag}(1/d_1, 1/d_2, 1/d_3)$ if all $d_i \neq 0$.

Upper triangular matrix:

$$U = \begin{bmatrix} u_{11} & u_{12} & u_{13} \\ 0 & u_{22} & u_{23} \\ 0 & 0 & u_{33} \end{bmatrix}.$$

- $\det(U) = u_{11}u_{22}u_{33}$.
- Eigenvalues are the diagonal entries: $\lambda_i = u_{ii}$.
- Solve $U\mathbf{x} = \mathbf{b}$ by back-substitution in $O(n^2)$.

Lower triangular matrix:

$$L = \begin{bmatrix} \ell_{11} & 0 & 0 \\ \ell_{21} & \ell_{22} & 0 \\ \ell_{31} & \ell_{32} & \ell_{33} \end{bmatrix}.$$

- $\det(L) = \ell_{11}\ell_{22}\ell_{33}$.
- Eigenvalues are ℓ_{ii}.
- Solve $L\mathbf{x} = \mathbf{b}$ by forward substitution.

Symmetric matrix: $A = A^T$; eigenvalues are real and eigenvectors can be chosen orthonormal.

Four fundamental subspaces and solvability.

- Column space $\mathcal{R}(A)$, null space $\mathcal{N}(A)$, row space $\mathcal{R}(A^T)$, left null space $\mathcal{N}(A^T)$.
- Existence: $A\mathbf{x} = \mathbf{b}$ solvable iff $\mathbf{b} \in \mathcal{R}(A)$.
- Uniqueness: solution unique iff $\mathcal{N}(A) = \{\mathbf{0}\}$.

Eigenvalues, eigenvectors, and algorithm sketch.
Eigenpair: $(\lambda, \mathbf{v})$ with $\mathbf{v} \neq \mathbf{0}$ such that $A\mathbf{v} = \lambda\mathbf{v}$.
Basic properties:

- $\det(A - \lambda I) = 0$ gives the characteristic polynomial; its roots are eigenvalues.
- For each eigenvalue λ, eigenvectors solve $(A - \lambda I)\mathbf{v} = \mathbf{0}$.
- If A has a basis of eigenvectors, then $A = V\Lambda V^{-1}$ with $V = [\mathbf{v}_1 \ \dots \ \mathbf{v}_n]$, $\Lambda = \mathrm{diag}(\lambda_1, \dots, \lambda_n)$.

Power method (sketch) for dominant eigenvalue:

1. Choose nonzero $\mathbf{x}_0$.
2. For $k = 0, 1, \dots$: compute $\mathbf{y}_{k+1} = A\mathbf{x}_k$, then normalize $\mathbf{x}_{k+1} = \mathbf{y}_{k+1}/\|\mathbf{y}_{k+1}\|$.
3. Rayleigh quotient $\lambda_k = \mathbf{x}_k^T A\mathbf{x}_k$ approximates largest $|\lambda|$.

Practical eigenvalue algorithms: QR iteration with shifts (used by `eig`); repeated orthogonal similarity transforms drive A toward upper triangular (Schur form) whose diagonal entries approximate eigenvalues.
Example (eigenvalues and eigenvectors of a 2×2 matrix).
Let

$$A = \begin{bmatrix} 2 & 1 \\ 1 & 2 \end{bmatrix}.$$

1) Eigenvalues: solve $\det(A - \lambda I) = 0$.

$$A - \lambda I = \begin{bmatrix} 2-\lambda & 1 \\ 1 & 2-\lambda \end{bmatrix}, \quad \det(A - \lambda I) = (2-\lambda)^2 - 1.$$

Set this to zero:

$$(2-\lambda)^2 - 1 = 0 \ \Rightarrow \ 2 - \lambda = \pm 1 \ \Rightarrow \ \lambda_1 = 3, \ \lambda_2 = 1.$$

2) Eigenvector for $\lambda_1 = 3$: solve $(A - 3I)\mathbf{v}_1 = \mathbf{0}$.
We have

$$A - 3I = \begin{bmatrix} -1 & 1 \\ 1 & -1 \end{bmatrix},$$

so the equation $(A - 3I)\mathbf{v}_1 = \mathbf{0}$ is

$$-v_{1,1} + v_{1,2} = 0 \ \Rightarrow \ v_{1,1} = v_{1,2}.$$

Choose $v_{1,1} = 1$, so

$$\mathbf{v}_1 = \begin{bmatrix} 1 \\ 1 \end{bmatrix},$$

and the normalized eigenvector is

$$\hat{\mathbf{v}}_1 = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 \\ 1 \end{bmatrix}.$$

3) Eigenvector for $\lambda_2 = 1$: solve $(A - I)\mathbf{v}_2 = \mathbf{0}$.
We have

$$A - I = \begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix},$$

so

$$v_{2,1} + v_{2,2} = 0 \ \Rightarrow \ v_{2,2} = -v_{2,1}.$$

Choose $v_{2,1} = 1$, so

$$\mathbf{v}_2 = \begin{bmatrix} 1 \\ -1 \end{bmatrix},$$

and the normalized eigenvector is

$$\hat{\mathbf{v}}_2 = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 \\ -1 \end{bmatrix}.$$

Check notation:

$$A\hat{\mathbf{v}}_1 = \lambda_1\hat{\mathbf{v}}_1, \quad A\hat{\mathbf{v}}_2 = \lambda_2\hat{\mathbf{v}}_2.$$

With $V = [\hat{\mathbf{v}}_1 \ \hat{\mathbf{v}}_2]$ and $\Lambda = \mathrm{diag}(\lambda_1, \lambda_2)$, we have

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

and since A is symmetric, V can be chosen orthogonal so that $V^{-1} = V^T$.

Orthogonal matrices and projectors

Orthogonal matrix. $Q \in \mathbb{R}^{n\times n}$ is orthogonal if

$$Q^TQ = QQ^T = I.$$

Properties:

$$Q^{-1} = Q^T, \qquad \|Qx\|_2 = \|x\|_2, \qquad (Qx)\cdot(Qy) = x\cdot y,$$
$$\det(Q) = \pm 1, \qquad |\lambda_i(Q)| = 1 \text{ for all eigenvalues } \lambda_i.$$

Orthogonal projector. If $Q \in \mathbb{R}^{n\times k}$ has orthonormal columns, then

$$P = QQ^T$$

is the projector onto $\mathcal{R}(Q)$, with

$$P^T = P, \quad P^2 = P, \quad \lambda_i(P) \in \{0, 1\}, \quad \mathrm{rank}(P) = k.$$

For least squares, with design matrix X,

$$\hat{y} = X\hat{\theta} = X(X^TX)^{-1}X^Ty = P_Xy.$$

Positive (semi)definite symmetric matrices

$A \in \mathbb{R}^{n\times n}$ symmetric.
Positive definite (PD):

$$x^TAx > 0 \ \ \forall x \neq 0.$$

Positive semidefinite (PSD):

$$x^TAx \geq 0 \ \ \forall x.$$

Equivalent facts for symmetric A:

$$A \succ 0 \iff \lambda_i(A) > 0 \, \forall i, \qquad A \succeq 0 \iff \lambda_i(A) \geq 0 \, \forall i.$$

Cholesky: $A \succ 0 \iff \exists R$ upper triangular with $A = R^TR$.
Optimization link. If $H(x)$ (Hessian) is PD at a critical point $x^\star$:

$\nabla f(x^\star) = 0$, $H(x^\star) \succ 0 \ \Rightarrow \ x^\star$ is a strict local minimum (and loc

Symmetric matrices and eigendecomposition

For symmetric $A \in \mathbb{R}^{n\times n}$, there exists an orthogonal W and real diagonal Λ such that

$$A = W\Lambda W^T, \quad W^TW = I, \quad \Lambda = \mathrm{diag}(\lambda_1, \dots, \lambda_n), \ \lambda_i \in \mathbb{R}.$$

Useful identities:

$$\det(A) = \prod_{i=1}^{n} \lambda_i, \qquad \mathrm{tr}(A) = \sum_{i=1}^{n} \lambda_i.$$

If an eigenvalue λ has algebraic multiplicity m and A is symmetric, then its geometric multiplicity is also m, so

$$\dim \mathcal{N}(A - \lambda I) = m, \qquad \mathrm{rank}(A - \lambda I) = n - m.$$

Rank–nullity and condition number

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

$$\dim \mathcal{R}(A) + \dim \mathcal{N}(A) = n.$$

If A is invertible, the (matrix) condition number in the 2-norm is

$$\kappa_2(A) = \|A\|_2 \|A^{-1}\|_2 = \frac{\max_i |\lambda_i(A)|}{\min_i |\lambda_i(A)|}.$$

3. SNE and optimization

Jacobian: matrix of first partial derivatives: $J_{ij} = \partial f_i/\partial x_j$.

Newton's method (SNE): iteratively solve linearized system to update guess.

Solve $\mathbf{f}(\mathbf{x}) = \mathbf{0}$:

$$J(\mathbf{x}_k)\Delta\mathbf{x}_k = -\mathbf{f}(\mathbf{x}_k), \quad \mathbf{x}_{k+1} = \mathbf{x}_k + \Delta\mathbf{x}_k.$$

Example (scalar Newton): Solve $x^2 - 2 = 0$.

$$f(x) = x^2 - 2, \quad f'(x) = 2x,$$

$$x_{k+1} = x_k - \frac{f(x_k)}{f'(x_k)} = x_k - \frac{x_k^2 - 2}{2x_k}.$$

Gradient: vector of first partial derivatives; direction of steepest ascent.

Hessian: matrix of second partial derivatives; local curvature.

Gradient/Hessian examples.

Quadratic:

$$f(\mathbf{x}) = \tfrac{1}{2}\mathbf{x}^T Q\mathbf{x} + \mathbf{c}^T\mathbf{x} + d, \quad Q = Q^T,$$

$$\nabla f(\mathbf{x}) = Q\mathbf{x} + \mathbf{c}, \quad H(\mathbf{x}) = Q.$$

Least squares $f(\mathbf{x}) = \frac{1}{2}\|A\mathbf{x} - \mathbf{b}\|_2^2$:

$$\nabla f(\mathbf{x}) = A^T(A\mathbf{x} - \mathbf{b}), \quad H(\mathbf{x}) = A^T A.$$

Example (minimize quadratic): Solve $\min f(x) = \frac{1}{2}x^T Qx + c^T x$. Set gradient $Qx + c = 0 \Rightarrow x^* = -Q^{-1}c$, assuming $Q \succ 0$.

Lagrangian: $\mathcal{L}(x, \lambda) = f(x) + \lambda^T c(x)$ for equality constraints $c(x) = 0$.

Equality-constrained optimum:

$$\nabla_x \mathcal{L} = \nabla f + J_c^T \lambda = 0, \quad c(x) = 0.$$

Convex set: $C \subset \mathbb{R}^n$ is convex if

$$x_1, x_2 \in C,\ t \in [0,1] \ \Rightarrow\ tx_1 + (1-t)x_2 \in C.$$

Convex function: $f : \mathbb{R}^n \to \mathbb{R}$ is convex if

$$f(tx_1+(1-t)x_2) \le tf(x_1)+(1-t)f(x_2),\ \forall x_1, x_2,\ t \in [0,1].$$

If f is twice differentiable and $\nabla^2 f(x) \succeq 0$ for all x, then f is convex. If $\nabla^2 f(x) \succ 0$ for all x, f is *strictly convex* and has at most one minimizer.

Unconstrained first-/second-order conditions: For $\min_x f(x)$ with f differentiable:

$$\nabla f(x^\star) = 0 \quad \text{(first-order necessary)}.$$

If additionally f is twice differentiable,

$$\nabla f(x^\star) = 0, \quad \nabla^2 f(x^\star) \succ 0$$

is a *sufficient* condition for $x^\star$ to be a strict local minimizer (unique locally).

For convex f, any $x^\star$ with $\nabla f(x^\star) = 0$ is a *global* minimizer.

Equality-constrained KKT (first order):

$$\min_x f(x) \ \text{ s.t. } \ c(x) = 0,\ c : \mathbb{R}^n \to \mathbb{R}^m.$$

Lagrangian

$$\mathcal{L}(x, \lambda) = f(x) + \lambda c(x)^T.$$

KKT system (necessary):

$$\nabla_x \mathcal{L}(x^\star, \lambda^\star) = \nabla f(x^\star) + J_c(x^\star)^T \lambda^\star = 0, \quad c(x^\star) = 0.$$

KKT Newton system (matrix form): At iterate (x_k, λ_k), one Newton step for the KKT conditions solves

$$\begin{bmatrix} \nabla^2_{xx}\mathcal{L}(x_k, \lambda_k) & J_c(x_k)^T \\ J_c(x_k) & 0 \end{bmatrix} \begin{bmatrix} \Delta x_k \\ \Delta\lambda_k \end{bmatrix} = - \begin{bmatrix} \nabla f(x_k) + J_c(x_k)^T\lambda_k \\ c(x_k) \end{bmatrix},$$

then updates $x_{k+1} = x_k + \Delta x_k,\ \lambda_{k+1} = \lambda_k + \Delta\lambda_k$.

Steepest descent (gradient descent): For unconstrained $\min f(x)$,

$$d_k = -\nabla f(x_k), \quad x_{k+1} = x_k + \alpha_k d_k.$$

- Fixed small $\alpha_k = \alpha \Rightarrow$ guaranteed descent for sufficiently small α, but slow convergence near optimum. - Exact/line-search α_k (1D minimization along d_k) $\Rightarrow$ better convergence but more work per step.

Newton's method (optimization):

$$d_k = -\nabla^2 f(x_k)^{-1}\nabla f(x_k), \quad x_{k+1} = x_k + d_k.$$

- Quadratic convergence near a solution if $\nabla^2 f(x^\star)$ is nonsingular (typically PD). - May diverge or go uphill if $\nabla^2 f$ is indefinite (nonconvex problems).

Quasi-Newton / BFGS: Maintain an approximation $H_k \approx \nabla^2 f(x_k)^{-1}$ updated from gradients only.

$$s_k = x_{k+1} - x_k, \quad y_k = \nabla f(x_{k+1}) - \nabla f(x_k),$$

$$H_{k+1} = \left(I - \frac{s_k y_k^T}{y_k^T s_k}\right) H_k \left(I - \frac{y_k s_k^T}{y_k^T s_k}\right) + \frac{s_k s_k^T}{y_k^T s_k}.$$

Search direction:

$$d_k = -H_k \nabla f(x_k).$$

- Start with $H_0 = I$ (first step = steepest descent). - If $y_k^T s_k > 0$, BFGS preserves positive definiteness of H_k and typically gives superlinear convergence.

Interior-point (barrier) method for inequality constraints: For

$$\min_x f(x) \ \text{ s.t. } \ h_i(x) \le 0,\ i = 1, \ldots, m,$$

introduce barrier parameter $\mu > 0$ and barrier function

$$\phi_\mu(x) = f(x) - \mu \sum_{i=1}^m \log(h_i(x)),$$

defined only for strictly feasible x with $h_i(x) < 0$. Algorithm idea:

- For large μ, approximately minimize $\phi_\mu(x)$ (e.g., with Newton).
- Decrease $\mu \to 0$ and warm-start each new solve from the previous minimizer.

As $\mu \to 0$, solutions of the barrier problems approach a KKT point of the original constrained problem.

4. Probability basics

Random variable (RV): numerical outcome of a random experiment.

Discrete vs continuous RVs:

- Discrete: X takes values in a countable set $\{x_k\}$, $P(X = x_k) = p_X(x_k)$, $\sum_k p_X(x_k) = 1$.
- Continuous: X has density $f_X(x) \geq 0$, $P(a \leq X \leq b) = \int_a^b f_X(x)\,dx$, $\int_{-\infty}^{\infty} f_X(x)\,dx = 1$.

PMF, PDF, CDF (definitions + intuition):

- **PMF (probability mass function)** for discrete X:

$$p_X(k) = P(X = k).$$

  Properties: $p_X(k) \geq 0$ for all k; $\sum_k p_X(k) = 1$. Intuition: a *bar chart of exact probabilities* at each possible value.

- **PDF (probability density function)** for continuous X:

$$f_X(x) \geq 0, \quad \int_{-\infty}^{\infty} f_X(x)\,dx = 1.$$

  Probability over an interval: $P(a \leq X \leq b) = \int_a^b f_X(x)\,dx$. Properties: nonnegative; integrates to 1; the value $f_X(x)$ is a *density*, not a probability, so $P(X = x) = 0$. Intuition: a *smooth curve of probability intensity*; area under the curve over a region is probability.

- **CDF (cumulative distribution function)** for any X:

$$F_X(x) = P(X \leq x).$$

  Discrete: $F_X(x) = \sum_{x_k \leq x} p_X(x_k)$. Continuous: $F_X(x) = \int_{-\infty}^{x} f_X(t)\,dt$. Key properties: $0 \leq F_X(x) \leq 1$; nondecreasing in x; $\lim_{x\to-\infty} F_X(x) = 0$, $\lim_{x\to+\infty} F_X(x) = 1$; right-continuous. Intuition: an *accumulated probability curve*, showing how much probability mass has been accumulated up to threshold x.

Expectation: mean; $\mathbb{E}[X]$.

Key formulas.

- $\mathbb{E}[g(X)] = \sum_k g(k)p_X(k)$ or $\int g(x)f_X(x)\,dx$.
- $\mathrm{Var}(X) = \mathbb{E}[X^2] - (\mathbb{E}[X])^2$.
- $P(A|B) = P(A \cap B)/P(B)$.
- Bayes: $P(A|B) = \dfrac{P(B|A)P(A)}{P(B)}$.

Marginal and conditional: Joint pmf/pdf $p_{X,Y}(x, y)$:

$$p_X(x) = \sum_y p_{X,Y}(x, y) \quad \text{or} \quad p_X(x) = \int p_{X,Y}(x, y)\,dy,$$

$$p_{Y|X}(y|x) = \frac{p_{X,Y}(x, y)}{p_X(x)}.$$

Convolution (sum of independent RVs): If $Z = X + Y$ and X, Y independent continuous,

$$f_Z(z) = \int_{-\infty}^{\infty} f_X(x)f_Y(z - x)\,dx.$$

For discrete,

$$p_Z(k) = \sum_i p_X(i)\,p_Y(k - i).$$

Covariance and covariance matrix:

$$\mathrm{Cov}(X, Y) = \mathbb{E}[(X - \mathbb{E}X)(Y - \mathbb{E}Y)].$$

For random vector $x \in \mathbb{R}^n$,

$$C = \mathbb{E}\big[(x - \mu)(x - \mu)^T\big], \quad \mu = \mathbb{E}[x].$$

C is symmetric, PSD; diagonal entries are variances, off-diagonals are covariances.

Biased vs unbiased estimator: An estimator $\hat{\theta}$ of parameter θ is

$$\text{unbiased if } \mathbb{E}[\hat{\theta}] = \theta, \quad \text{biased otherwise.}$$

Examples:

- Sample mean: $\hat{\mu} = \frac{1}{N}\sum_{i=1}^{N} X_i$ is unbiased for μ.
- Sample variance: $\hat{\sigma}^2 = \frac{1}{N-1}\sum_i (X_i - \hat{\mu})^2$ is unbiased; with $\frac{1}{N}$ it is biased.

Central Limit Theorem (CLT): If X_i are IID with mean μ and variance $\sigma^2 < \infty$, then for large N,

$$\bar{X}_N = \frac{1}{N}\sum_{i=1}^{N} X_i \approx \mathcal{N}\Big(\mu, \frac{\sigma^2}{N}\Big).$$

Only the *sample mean* becomes approximately normal; the histogram of raw samples approaches the true underlying distribution, not necessarily Gaussian.

Example (law of total prob.): Events A_i partition and event B:

$$P(B) = \sum_i P(B|A_i)P(A_i).$$

Notable distributions

Bernoulli (discrete):

$$P(X = 1) = p, \quad P(X = 0) = 1 - p, \quad X \in \{0, 1\}.$$

Mean $\mathbb{E}X = p$, variance $\mathrm{Var}(X) = p(1 - p)$.

Binomial (discrete):

$$P(X = k) = \binom{n}{k} p^k (1 - p)^{n-k}, \quad k = 0, 1, \ldots, n.$$

Mean np, variance $np(1 - p)$.

Geometric (discrete, "# of failures before first success" version):

$$P(X = k) = p(1 - p)^k, \quad k = 0, 1, 2, \ldots$$

Mean $\frac{1-p}{p}$, variance $\frac{1-p}{p^2}$.

Poisson (discrete):

$$P(K = k) = \frac{\lambda^k e^{-\lambda}}{k!}, \quad k = 0, 1, 2, \ldots$$

Mean λ, variance λ.

Uniform (continuous on $[a, b]$):

$$f_X(x) = \begin{cases} \frac{1}{b-a}, & a \leq x \leq b, \\ 0, & \text{otherwise,} \end{cases}$$

Mean $\frac{a+b}{2}$, variance $\frac{(b-a)^2}{12}$.

Normal / Gaussian (continuous):

$$f_X(x) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(x - \mu)^2}{2\sigma^2}\right), \quad x \in \mathbb{R}.$$

Mean μ, variance σ^2.

Multivariate Normal (continuous): For $x \in \mathbb{R}^k$,

$$p_X(x) = \frac{1}{(2\pi)^{k/2}\sqrt{\det C}} \exp\Big(-\tfrac{1}{2}(x - \mu)^T C^{-1}(x - \mu)\Big),$$

mean vector μ, covariance C.

Exponential (continuous, rate $\lambda > 0$):

$$f_X(x) = \lambda e^{-\lambda x}, \quad x \geq 0.$$

Mean $1/\lambda$, variance $1/\lambda^2$.

Gamma (continuous, shape $\alpha > 0$, rate $\beta > 0$):

$$f_X(x) = \frac{\beta^\alpha}{\Gamma(\alpha)} x^{\alpha-1} e^{-\beta x}, \quad x \geq 0.$$

Mean α/β, variance α/β^2.

5. Models vs data & design

Linear models and estimation

Design matrix X: rows are input vectors $\mathbf{x}_i^T$, columns correspond to parameters.

Linear model (scalar output):

$$y_i = \mathbf{x}_i^T \boldsymbol{\theta} + \varepsilon_i, \quad \varepsilon_i \sim \mathcal{N}(0, \sigma^2),$$

$$\mathbf{y} = X\boldsymbol{\theta} + \boldsymbol{\varepsilon}.$$

Homoscedastic vs heteroscedastic errors:

- Homoscedastic: $\mathrm{Var}(\varepsilon_i) = \sigma^2$ for all i (same variance).
- Heteroscedastic: $\mathrm{Var}(\varepsilon_i)$ depends on i (or on $\mathbf{x}_i$).

OLS (ordinary least squares): homoscedastic, uncorrelated errors.

$$\hat{\boldsymbol{\theta}}_{\mathrm{OLS}} = \arg\min_{\theta} \sum_i (y_i - \mathbf{x}_i^T \theta)^2 = (X^T X)^{-1} X^T \mathbf{y}.$$

WLS (weighted least squares): known covariance V of ε.

$$\hat{\boldsymbol{\theta}}_{\mathrm{WLS}} = \arg\min_{\theta} (\mathbf{y} - \mathbf{y}^{\mathrm{model}}(\theta))^T V^{-1} (\mathbf{y} - \mathbf{y}^{\mathrm{model}}(\theta)).$$

If errors are Gaussian, OLS/WLS coincide with MLE:

Likelihood and MLE:

$$L(\theta) = p(D \mid \theta), \quad \hat{\theta}_{\mathrm{MLE}} = \arg\max_{\theta} L(\theta) = \arg\max_{\theta} \log L(\theta).$$

For independent Gaussian errors with covariance V,

$$\log L(\theta) = -\tfrac{1}{2}(\mathbf{y} - \mathbf{y}^{\mathrm{model}}(\theta))^T V^{-1} (\mathbf{y} - \mathbf{y}^{\mathrm{model}}(\theta)) + \text{const}.$$

$\Rightarrow$ maximizing likelihood $\equiv$ minimizing (weighted) SSE.

Example (OLS fit): Data (x_i, y_i), linear model $y = ax + b$.

$$X = \begin{bmatrix} 1 & x_1 \\ \vdots & \vdots \\ 1 & x_N \end{bmatrix}, \quad \hat{\boldsymbol{\theta}} = \begin{bmatrix} \hat{b} \\ \hat{a} \end{bmatrix} = (X^T X)^{-1} X^T \mathbf{y}.$$

Maximum Likelihood Estimation (MLE): Given data $x_1, \ldots, x_N$ assumed IID from a model with parameter θ, the *likelihood* is

$$L(\theta) = \prod_{i=1}^{N} p(x_i \mid \theta),$$

viewed as a function of θ. The MLE chooses the parameter value that makes the observed data most probable:

$$\hat{\theta}_{\mathrm{MLE}} = \arg\max_{\theta} L(\theta) = \arg\max_{\theta} \prod_{i=1}^{N} p(x_i \mid \theta).$$

Equivalently, because log is monotone,

$$\hat{\theta}_{\mathrm{MLE}} = \arg\max_{\theta} \sum_{i=1}^{N} \log p(x_i \mid \theta),$$

which is usually easier to optimize numerically. Intuition: pick the parameter that *best explains the data* under the assumed probabilistic model.

Bayesian vs frequentist, MAP, and Bayes' theorem

Frequentist view: θ is fixed but unknown; uncertainty is in data. Use point estimates (OLS/WLS/MLE) and confidence or χ^2 regions.

Bayesian view: θ is random with prior $p(\theta)$; data update beliefs via Bayes' theorem to posterior $p(\theta \mid D)$; use MAP and credible regions.

Bayes' theorem (parameters):

$$p(\theta \mid D) = \frac{p(D \mid \theta) p(\theta)}{p(D)}, \quad p(D) = \int p(D \mid \theta) p(\theta)\, d\theta.$$

Posterior $\propto$ likelihood $\times$ prior.

Prior/posterior/MAP:

- Prior $p(\theta)$: belief about θ before data.
- Posterior $p(\theta \mid D) \propto p(D \mid \theta) p(\theta)$.
- MAP estimator:

$$\hat{\theta}_{\mathrm{MAP}} = \arg\max_{\theta} p(\theta \mid D) = \arg\max_{\theta} [\log L(\theta) + \log p(\theta)].$$

Connections:

- OLS/WLS/MLE: minimize SSE or weighted SSE; frequentist.
- MAP: regularized SSE (log prior acts as penalty); Bayesian.

Conjugate prior: A prior family $p(\theta)$ is conjugate to a likelihood $p(D \mid \theta)$ if the posterior $p(\theta \mid D)$ is in the same family. Example: $\lambda \sim \mathrm{Gamma}(\alpha, \beta)$ and $k \mid \lambda \sim \mathrm{Poisson}(\lambda) \Rightarrow \lambda \mid k \sim \mathrm{Gamma}(\alpha + k, \beta + 1)$.

Fisher information, covariance, and design

Fisher information:

$$\mathcal{I}(\theta) = -\mathbb{E}\big[\nabla_\theta^2 \log L(\theta)\big].$$

For linear, homoscedastic model $\mathbf{y} = X\theta + \varepsilon$, $\varepsilon \sim \mathcal{N}(0, \sigma^2 I)$:

$$\widehat{\mathrm{Cov}}(\hat{\boldsymbol{\theta}}) \approx \sigma^2 (X^T X)^{-1}, \quad \mathcal{I}(\theta) = \frac{1}{\sigma^2} X^T X.$$

For general nonlinear models with sensitivity Jacobian $J(\theta)$ and error covariance V:

$$\mathcal{I}(\theta) \approx J(\theta)^T V^{-1} J(\theta).$$

Experimental design: Choose inputs (rows of X or evaluations of J) to reduce parameter uncertainty by optimizing a scalar function of the information matrix.

Let M denote an information matrix (e.g. $M = X^T X$ or $M = J^T J$ for homoscedastic errors). Common criteria:

- A-optimal: minimize $\mathrm{tr}(M^{-1})$ (average parameter variance).
- D-optimal: maximize $\det(M)$ (minimize volume of confidence ellipsoid).
- E-optimal: minimize largest eigenvalue of M^{-1} (worst-case variance).

χ^2, region of indifference, and credible regions

χ^2 misfit (Gaussian errors): If measurement covariance is V,

$$\chi^2(\theta) = (\mathbf{y} - \mathbf{y}^{\mathrm{model}}(\theta))^T V^{-1} (\mathbf{y} - \mathbf{y}^{\mathrm{model}}(\theta)).$$

Under correct Gaussian noise, $\chi^2(\theta_{\mathrm{true}})$ approximately follows a χ^2 distribution with

$$\nu = (\#\ \text{data}) - (\#\ \text{fitted parameters}).$$

Region of indifference (frequentist): For significance level α,

$$\mathcal{R}_\alpha = \{\theta : \ \chi^2(\theta) \le \chi^2_{\max}\},$$

where $\chi^2_{\max}$ is a chosen quantile of the χ^2_ν distribution (e.g. $1 - \alpha$). Parameters in $\mathcal{R}_\alpha$ give fits consistent with the data at level α.

Credible interval/region (Bayesian): A $(1-\alpha)$ credible set C satisfies

$$P(\theta \in C \mid D) = 1 - \alpha.$$

If $p(\theta \mid D) \approx \mathcal{N}(\hat{\theta}, V)$, an ellipsoidal credible region is

$$(\theta - \hat{\theta})^T V^{-1} (\theta - \hat{\theta}) \le c_\alpha,$$

with c_α chosen to contain posterior mass $1-\alpha$. This quadratic form is analogous to a χ^2 threshold (but interpreted Bayesianly).

6. Monte Carlo & MCMC

Monte Carlo integration

Monte Carlo integration (simple sampling): Approximate an integral by averaging random samples.

$$I = \int_\Omega f(\mathbf{x})\, d\mathbf{x} = \mathrm{Vol}(\Omega)\, \mathbb{E}[f(X)],\ X \sim \mathrm{Unif}(\Omega),$$

$$I \approx \frac{\mathrm{Vol}(\Omega)}{N} \sum_{i=1}^{N} f(\mathbf{x}_i), \quad \mathbf{x}_i \sim \mathrm{Unif}(\Omega).$$

Error scales as $O(N^{-1/2})$, independent of dimension.

Importance sampling: Rewrite

$$I = \int f(x)\, dx = \int f(x)\, \frac{p(x)}{p(x)}\, dx = \mathbb{E}_p\left[\frac{f(X)}{p(X)}\right],$$

where $p(x) > 0$ wherever $f(x) \neq 0$. Estimate:

$$I \approx \frac{1}{N} \sum_{i=1}^{N} \frac{f(x_i)}{p(x_i)}, \quad x_i \sim p.$$

Choosing p with similar shape to f reduces variance.

Normalized vs unnormalized densities

Normalized vs unnormalized:

- A normalized pdf $\pi(x)$ satisfies $\int \pi(x)\, dx = 1$.
- Often only known up to a constant: $\tilde{\pi}(x) \propto \pi(x)$.

In many computations (e.g. Metropolis–Hastings), the normalization constant cancels: ratios $\pi(x')/\pi(x)$ can be computed using $\tilde{\pi}$.

Examples:

- Posterior: $\pi(\theta \mid D) \propto p(D \mid \theta) p(\theta)$, evidence $p(D)$ unknown.
- Target for MCMC: only need $\tilde{\pi}$ for acceptance probability.

Markov chains and MCMC

Markov chain: Sequence $X_0, X_1, \ldots$ with

$$P(X_{k+1} \in A \mid X_k = x_k, \ldots, X_0 = x_0) = P(X_{k+1} \in A \mid X_k = x_k).$$

Target / stationary distribution: A distribution π is stationary for the chain if

$$X_0 \sim \pi \ \Rightarrow\ X_k \sim \pi\ \forall k.$$

A sufficient condition is *detailed balance*:

$$\pi(x) T(x \to x') = \pi(x') T(x' \to x) \quad \forall x, x',$$

where $T(x \to x')$ is the one-step transition density/mass.

MCMC: Construct a Markov chain with stationary distribution $\pi(x)$ (often known only up to a constant), then approximate expectations

$$\mathbb{E}_\pi[g(X)] = \int g(x)\pi(x)\, dx \approx \frac{1}{N} \sum_{i=B+1}^{B+N} g(x_i),$$

where $\{x_i\}$ is the chain, and B is burn-in.

Markov property (first-order chain): A discrete-time stochastic process $\{X_0, X_1, \ldots, X_n\}$ is a (first-order) Markov chain if for all k:

$$P(X_{k+1} = x_{k+1} \mid X_k = x_k, X_{k-1} = x_{k-1}, \ldots, X_0 = x_0) = \ldots$$

$$\ldots = P(X_{k+1} = x_{k+1} \mid X_k = x_k)$$

Intuition: the future depends on the present, not on the full past ("one-step memory").

Markov chain joint factorization: For a Markov chain $X_0, \ldots, X_n$,

$$P(X_0 = x_0, \ldots, X_n = x_n) = \ldots$$

$$\ldots = P(X_0 = x_0) \prod_{k=1}^{n} P(X_k = x_k \mid X_{k-1} = x_{k-1}).$$

If the chain is time-homogeneous with transition matrix P, then $P(X_k = j \mid X_{k-1} = i) = P_{ij}$ and the joint is fully determined by the initial distribution and P.

Bayesian network (directed Markov factorization): Let $X = (X_1, \ldots, X_n)$ be random variables with a directed acyclic graph (DAG) where $\mathrm{Pa}(i)$ are the parents of node i. If the distribution is Markov w.r.t. this DAG, its joint pdf/pmf factorizes as

$$p(x_1, \ldots, x_n) = \prod_{i=1}^{n} p\big(x_i \mid x_{\mathrm{Pa}(i)}\big).$$

Intuition: each node only "looks at" its parents; all other conditional independencies follow from the graph.

Metropolis–Hastings (MH) algorithm

Given current state x_k and target $\pi(x)$ (we can use unnormalized $\tilde{\pi}$):

1. Propose $x' \sim q(\cdot \mid x_k)$ (proposal kernel).
2. Compute acceptance probability

$$\alpha(x', x_k) = \min\left(1, \frac{\tilde{\pi}(x')\, q(x_k \mid x')}{\tilde{\pi}(x_k)\, q(x' \mid x_k)}\right).$$

3. Draw $u \sim \mathrm{Unif}(0,1)$; if $u \le \alpha$, set $x_{k+1} = x'$, else $x_{k+1} = x_k$.

If q is symmetric, $q(x' \mid x) = q(x \mid x')$, then

$$\alpha(x', x) = \min\left(1, \frac{\tilde{\pi}(x')}{\tilde{\pi}(x)}\right).$$

Using MCMC for expectations: If the chain is ergodic (can reach all relevant states and has unique stationary distribution),

$$\frac{1}{N} \sum_{i=B+1}^{B+N} g(x_i) \to \mathbb{E}_\pi[g(X)]$$

as $N \to \infty$.

Practical points

- **Burn-in:** discard initial samples before chain reaches stationarity.
- **Thinning:** (optional) keep every k-th sample to reduce autocorrelation; not strictly necessary if N is large.
- **Ergodicity/irreducibility:** proposal must allow reaching all regions where $\pi(x) > 0$; avoid one-sided proposals that can never go backward.
- **Unnormalized targets:** for posterior $p(\theta \mid D) \propto p(D \mid \theta)p(\theta)$, use $\tilde{\pi}(\theta) = p(D \mid \theta)p(\theta)$ directly in MH; no need to compute evidence $p(D)$.

```
function samples = metropolis_hastings(logpdf, x0, nsamples, proposal_std)
% METROPOLIS_HASTINGS Basic Metropolis-Hastings sampler
%
% Inputs:
% logpdf - function handle, logpdf(x) returns log p(x) up to a constant
% x0 - initial state (column vector or scalar)
% nsamples - number of samples to draw
% proposal_std - scalar std dev for Gaussian random-walk proposal
%
% Output:
% samples - matrix of samples, size = [length(x0), nsamples]
    % Preallocate storage for samples
    dim = numel(x0);
    samples = zeros(dim, nsamples);

    % Current state and its log-density
    x_curr = x0(:); % ensure column vector
    logp_curr = logpdf(x_curr);

    for k = 1:nsamples

        % --- PROPOSE A NEW STATE (Gaussian random walk) ---
        % For multivariate x, use independent N(0, proposal_std^2) in each dim
        x_prop = x_curr + proposal_std * randn(dim, 1);

        % --- COMPUTE ACCEPTANCE RATIO ---
        % For symmetric proposals q(x'|x) = q(x|x'), Hastings term cancels
        logp_prop = logpdf(x_prop);
        % log acceptance ratio: log alpha = log p(x_prop) - log p(x_curr)
        log_alpha = logp_prop - logp_curr;

        % Draw u ~ Uniform(0,1) and accept if log u < log alpha
        if log(rand) < log_alpha
            % Accept proposal
            x_curr = x_prop;
            logp_curr = logp_prop;
        end

        % Store current state (whether moved or not)
        samples(:, k) = x_curr;
    end
end
```

7. ODE IVPs

ODE IVP: find $\mathbf{x}(t)$ such that

$$\dot{\mathbf{x}} = f(t, \mathbf{x}), \quad \mathbf{x}(t_0) = \mathbf{x}_0.$$

Time-stepping methods

Explicit Euler (Forward Euler):

$$\mathbf{x}_{n+1} = \mathbf{x}_n + \Delta t\, f(t_n, \mathbf{x}_n).$$

Implicit Euler (Backward Euler):

$$\mathbf{x}_{n+1} = \mathbf{x}_n + \Delta t\, f(t_{n+1}, \mathbf{x}_{n+1}),$$

requires solving a nonlinear system each step (e.g. Newton). Good stability for stiff problems.

Midpoint / RK2 (explicit):

$$\mathbf{k}_1 = f(t_n, \mathbf{x}_n), \quad \mathbf{k}_2 = f\Big(t_n + \tfrac{\Delta t}{2}, \mathbf{x}_n + \tfrac{\Delta t}{2}\mathbf{k}_1\Big),$$

$$\mathbf{x}_{n+1} = \mathbf{x}_n + \Delta t\, \mathbf{k}_2.$$

Higher order than Euler for similar cost.

Classical RK4 (explicit):

$$\begin{aligned}
\mathbf{k}_1 &= f(t_n, \mathbf{x}_n),\\
\mathbf{k}_2 &= f\Big(t_n + \tfrac{\Delta t}{2}, \mathbf{x}_n + \tfrac{\Delta t}{2}\mathbf{k}_1\Big),\\
\mathbf{k}_3 &= f\Big(t_n + \tfrac{\Delta t}{2}, \mathbf{x}_n + \tfrac{\Delta t}{2}\mathbf{k}_2\Big),\\
\mathbf{k}_4 &= f\big(t_n + \Delta t, \mathbf{x}_n + \Delta t\, \mathbf{k}_3\big),\\
\mathbf{x}_{n+1} &= \mathbf{x}_n + \frac{\Delta t}{6}\big(\mathbf{k}_1 + 2\mathbf{k}_2 + 2\mathbf{k}_3 + \mathbf{k}_4\big).
\end{aligned}$$

Accuracy and local truncation error

Local truncation error (LTE): error made in one step assuming $\mathbf{x}_n$ is exact.

- Explicit Euler: LTE $= O(\Delta t^2)$, global error $= O(\Delta t)$ (1st order).
- Midpoint / RK2: LTE $= O(\Delta t^3)$, global error $= O(\Delta t^2)$ (2nd order).
- RK4: LTE $= O(\Delta t^5)$, global error $= O(\Delta t^4)$ (4th order).

Stability and stiffness

Test problem: $\dot{x} = \lambda x$, exact solution $x(t) = e^{\lambda t}x_0$.

Stability region: set of $z = \lambda \Delta t$ such that the numerical solution does not grow for $\mathrm{Re}(\lambda) < 0$. For a one-step method $x_{n+1} = R(z)x_n$, stability requires $|R(z)| \le 1$.

Examples:

- Explicit Euler: $R(z) = 1 + z$, stability if $|1 + z| \le 1$. For real $\lambda < 0$, this gives $-2 \le \lambda \Delta t \le 0$.
- Implicit Euler: $R(z) = 1/(1 - z)$; $|R(z)| < 1$ for all $\mathrm{Re}(z) < 0$ (A-stable). Good for stiff problems.

Stiffness: the system has widely separated time scales (eigenvalues with very different negative real parts). Explicit methods require very small Δt for stability; implicit methods allow much larger steps without instability (though accuracy still limits Δt).

Explicit Euler on decay:

$$\dot{x} = -kx, \quad x(0) = x_0.$$

Euler:

$$x_{n+1} = x_n - k\Delta t\, x_n = (1 - k\Delta t)x_n.$$

Stable if $|1 - k\Delta t| \le 1 \Rightarrow 0 \le k\Delta t \le 2$.

8. ODE BVPs

BVP (boundary value problem): differential equation with conditions at multiple points (typically interval endpoints). Example:

$$y''(x) = F(x, y, y'), \quad x \in [a, b],$$

$$\alpha_1(y(a), y'(a)) = 0, \ \alpha_2(y(b), y'(b)) = 0.$$

Shooting method (single shooting)

Idea: turn BVP into an IVP with unknown initial condition(s).

- Introduce unknown initial slope/parameters $\mathbf{c}$ (e.g. $y'(a) = c$).
- Solve IVP for given $\mathbf{c}$ to get $y(x; \mathbf{c})$.
- Evaluate boundary residual at $x = b$: $\mathbf{h}(\mathbf{c})$.
- Use a root finder (Newton or secant) to solve $\mathbf{h}(\mathbf{c}) = 0$.

Works well when IVP is well-behaved; can struggle for highly sensitive/unstable IVPs.

Example (linear BVP):

$$y'' = 0, \quad y(0) = 0, \ y(1) = 1.$$

Let $y'(0) = c$. Solve IVP: $y(x) = cx$. Enforce $y(1) = 1 \Rightarrow c = 1$, so $y(x) = x$.

Finite difference method (FDM) for BVPs

Example:

$$A\, y''(x) = f(x), \quad x \in [a, b],$$

with Dirichlet BCs $y(a) = \alpha$, $y(b) = \beta$.

Mesh: $x_i = a + i\Delta x$, $i = 0, \ldots, M + 1$ with $\Delta x = (b - a)/(M + 1)$. Use interior points $i = 1, \ldots, M$.

Centered second derivative:

$$y''(x_i) \approx \frac{y_{i+1} - 2y_i + y_{i-1}}{\Delta x^2}.$$

Leads to a tridiagonal linear system $A_h\mathbf{y} = \mathbf{b}$ for unknown interior values.

BC handling:

- Dirichlet: set $y_0 = \alpha$, $y_{M+1} = \beta$ directly.
- Neumann: approximate derivative with one-sided or ghost-point finite differences.

Remarks:

- Shooting leverages IVP solvers (adaptive time stepping) but can suffer from sensitivity.
- Finite difference BVP is more robust and leads to linear/nonlinear algebraic systems.

9. PDEs

Classification

General 2D second-order linear PDE:

$$Au_{xx} + Bu_{xy} + Cu_{yy} + Du_x + Eu_y + Gu = H.$$

Classification via $B^2 - 4AC$:

- Elliptic: $B^2 - 4AC < 0$ (e.g. Laplace, Poisson).
- Parabolic: $B^2 - 4AC = 0$ (e.g. heat equation).
- Hyperbolic: $B^2 - 4AC > 0$ (e.g. wave, advection).

Finite difference for elliptic PDEs

Example (Laplace):

$$u_{xx} + u_{yy} = 0 \quad \text{in a rectangle.}$$

Uniform grid with spacing h in both directions; interior node (i, j):

$$u_{xx}(x_i, y_j) \approx \frac{u_{i+1,j} - 2u_{i,j} + u_{i-1,j}}{h^2},$$

$$u_{yy}(x_i, y_j) \approx \frac{u_{i,j+1} - 2u_{i,j} + u_{i,j-1}}{h^2}.$$

5-point stencil:

$$u_{i+1,j} + u_{i-1,j} + u_{i,j+1} + u_{i,j-1} - 4u_{i,j} = 0.$$

Leads to a sparse linear system $A\mathbf{u} = \mathbf{b}$.

Method of lines (MoL) for time-dependent PDEs

Method of lines: discretize space (e.g. by finite differences) to obtain a large ODE system in time; then apply ODE IVP solvers.

Example: 1D diffusion

$$u_t = Du_{xx}, \quad x \in [0, L],$$

space discretization:

$$\frac{d}{dt}u_j(t) = D\frac{u_{j+1}(t) - 2u_j(t) + u_{j-1}(t)}{\Delta x^2},$$

which is a stiff ODE system that can be integrated with implicit methods or `ode15s`.

Advection, CFL, and upwinding

Example advection:

$$u_t + au_x = 0, \quad a > 0.$$

CFL number:

$$\text{CFL} = \frac{a\Delta t}{\Delta x}.$$

Roughly, Courant–Friedrichs–Lewy condition requires the numerical domain of dependence to contain the PDE's domain of dependence; gives step restrictions for explicit schemes.

Upwind scheme (first-order): For $a > 0$,

$$u_j^{n+1} = u_j^n - \text{CFL}\,(u_j^n - u_{j-1}^n).$$

This is stable for $0 \le \text{CFL} \le 1$.

Central difference in space + forward Euler in time:

$$u_j^{n+1} = u_j^n - \frac{\text{CFL}}{2}\,(u_{j+1}^n - u_{j-1}^n)$$

is typically unstable for pure advection, motivating upwinding and/or more advanced schemes.

Example (CFL check): $a = 2$, $\Delta x = 0.1$, want CFL $= 1$:

$$\text{CFL} = \frac{a\Delta t}{\Delta x} = 1 \Rightarrow \Delta t = 0.05.$$

10. Differential-Algebraic Equations (DAEs)

DAE: A system that mixes differential and algebraic equations.

Semi-explicit form:

$$\dot{\mathbf{x}} = f(t, \mathbf{x}, \mathbf{y}), \quad 0 = g(t, \mathbf{x}, \mathbf{y}),$$

where $\mathbf{x}$ are differential states (time derivatives appear) and $\mathbf{y}$ are algebraic states (no time derivatives).

Implicit form:

$$F\big(t, \mathbf{z}, \dot{\mathbf{z}}\big) = 0,$$

where $\mathbf{z}$ collects all states. If $\partial F/\partial \dot{\mathbf{z}}$ is singular, the system is a genuine DAE (not just an implicit ODE).

Mass-matrix form:

$$M(t, \mathbf{y})\,\dot{\mathbf{y}} = F(t, \mathbf{y}),$$

with (possibly singular) mass matrix M. If M is nonsingular, this is an ODE; if M is singular, it is a DAE suitable for mass-matrix solvers (e.g. `ode15s`).

Differential index: The smallest number of times the algebraic equations must be differentiated (with respect to time) to obtain an explicit ODE for all derivatives $\dot{\mathbf{x}}, \dot{\mathbf{y}}$.

Index-1 test (semi-explicit): Given

$$\dot{\mathbf{x}} = f(t, \mathbf{x}, \mathbf{y}), \quad 0 = g(t, \mathbf{x}, \mathbf{y}),$$

differentiate $g = 0$ once:

$$\frac{d}{dt} g(t, \mathbf{x}, \mathbf{y}) = g_t + g_{\mathbf{x}} \dot{\mathbf{x}} + g_{\mathbf{y}} \dot{\mathbf{y}} = 0.$$

If $g_{\mathbf{y}}$ is nonsingular, solve for $\dot{\mathbf{y}}$:

$$\dot{\mathbf{y}} = -g_{\mathbf{y}}^{-1} (g_t + g_{\mathbf{x}} \dot{\mathbf{x}}),$$

so the system is index-1. If further differentiation is needed to isolate derivatives, the index is > 1.

Implicit (hidden) constraints: Higher-index DAEs have additional constraints obtained by differentiating $g(t, \mathbf{x}, \mathbf{y}) = 0$ (e.g. $\dot{g} = 0$, $\ddot{g} = 0$). Solutions must satisfy both the explicit algebraic constraints $g = 0$ and these derivative constraints. Index-1 DAEs typically do not have nontrivial hidden constraints.

Consistent initialization: An initial triple $(\mathbf{x}_0, \dot{\mathbf{x}}_0, \mathbf{y}_0)$ that satisfies

- the DAE at t_0:

$$\dot{\mathbf{x}}_0 = f(t_0, \mathbf{x}_0, \mathbf{y}_0), \quad 0 = g(t_0, \mathbf{x}_0, \mathbf{y}_0),$$

- and, for higher-index systems, any implied derivative constraints obtained by differentiating $g = 0$ enough times (e.g. $\dot{g}(t_0, \mathbf{x}_0, \mathbf{y}_0, \dot{\mathbf{x}}_0, \dot{\mathbf{y}}_0) = 0$).

Not every choice of $(\mathbf{x}_0, \mathbf{y}_0)$ is allowed; algebraic and hidden constraints restrict admissible initial conditions.

Example (semi-explicit index-1):

$$\dot{x} = y, \quad 0 = x + y.$$

Algebraic constraint: $x + y = 0 \Rightarrow y = -x$. Substitute into the differential equation:

$$\dot{x} = -x,$$

which is an ODE. The system is index-1. Consistent initialization requires $y_0 = -x_0$.

Variables, equations, and DoF (semi-explicit index-1): Consider

$$\dot{\mathbf{x}} = f(t, \mathbf{x}, \mathbf{y}), \quad 0 = g(t, \mathbf{x}, \mathbf{y}),$$

with differential states $\mathbf{x} \in \mathbb{R}^{n_d}$, algebraic states $\mathbf{y} \in \mathbb{R}^{n_a}$, and $g_{\mathbf{y}}$ full rank so that g provides n_a independent algebraic equations. At t_0:

- Unknowns in the initialization: $(\mathbf{x}_0, \mathbf{y}_0) \in \mathbb{R}^{n_d + n_a}$.
- Algebraic constraints: $g(t_0, \mathbf{x}_0, \mathbf{y}_0) = 0$ impose n_a independent conditions.

Thus the number of degrees of freedom in specifying a consistent initialization is

$$\text{DoF} = (n_d + n_a) - n_a = n_d.$$

A convenient viewpoint for regular index-1 systems is:

- choose the n_d components of $\mathbf{x}_0$ freely,
- then solve $g(t_0, \mathbf{x}_0, \mathbf{y}_0) = 0$ for the n_a components of $\mathbf{y}_0$ to obtain a consistent $(\mathbf{x}_0, \mathbf{y}_0)$.

For higher-index DAEs, additional hidden constraints from differentiated g further reduce admissible initial values and may require specifying some derivatives at t_0 as well.

Index reduction (idea): Rewrite a higher-index DAE as an equivalent index-1 DAE by:

- differentiating algebraic equations to expose hidden constraints,
- eliminating redundant variables/constraints or introducing new variables for derivatives,
- reformulating the model in semi-explicit or mass-matrix index-1 form.

This facilitates numerical solution with standard DAE solvers.

Connection to ODE-IVPs:

- Index-0 DAE $\equiv$ ODE-IVP: $\dot{\mathbf{x}} = f(t, \mathbf{x})$.
- Index-1 DAEs can typically be solved directly with DAE-capable time-stepping methods once consistent initial conditions are available.
- Higher-index DAEs are usually reduced to index-1 before numerical solution.

Numerical solution in MATLAB (index-1 DAEs)

Mass-matrix solvers (`ode15s`, `ode23t`): Solve

$$M(t, \mathbf{y})\, \dot{\mathbf{y}} = F(t, \mathbf{y}),$$

with possibly singular M. You provide M, F, an initial $\mathbf{y}_0$ (satisfying algebraic constraints), and optionally $\dot{\mathbf{y}}_0$; the solver can also attempt to determine a consistent $\dot{\mathbf{y}}_0$.

Fully implicit solver (`ode15i`): Solve

$$F(t, \mathbf{y}, \dot{\mathbf{y}}) = 0$$

for index-1 DAEs in implicit form. Typical workflow:

- Define a function for $F(t, \mathbf{y}, \dot{\mathbf{y}})$.
- Use `decic` to compute consistent $(\mathbf{y}_0, \dot{\mathbf{y}}_0)$ from approximate guesses.
- Call `ode15i` with F, $(\mathbf{y}_0, \dot{\mathbf{y}}_0)$, and the time grid.

11. Quick pattern-matching table

Question type	Immediate move
Solve $Ax = b$	Check m, n, $\kappa(A)$; use backslash; think about uniqueness.
Nonlinear equations	Form $f(x)$, use Newton/-tangent iteration; need Jacobian and stopping criteria.
Unconstrained min	Write ∇f, H; use Newton/-gradient descent; check $H \succ 0$.
Regression	Build X, y; use OLS/WLS; compute θ, covariance, goodness-of-fit.
MLE / MAP	Write likelihood/log-likelihood; differentiate, set to 0; add prior for MAP.
MC integral	Express integral as expectation; choose proposal/target; define estimator.
ODE-IVP	Identify stiffness; choose explicit vs implicit; argue stability and accuracy.
BVP	BVP vs IVP; choose shooting or FD/FEM; formulate equations.
PDE (steady)	Classify; discretize in space; build sparse linear system.
PDE (time)	Use method of lines or explicit scheme; compute CFL; choose upwinding if needed.
DAE	Write semi-explicit/mass form; estimate index; enforce consistent init.

12. Useful MATLAB functions

Linear algebra:

- `A\b`: solve $Ax = b$.
- `eig(A)`: eigenvalues/vectors.
- `svd(A)`: SVD, rank, cond.
- `norm(A,p)`, `cond(A)`.

Optimization / SNE:

- `fsolve`: nonlinear systems.
- `fminunc`: unconstrained min.
- `fmincon`: constrained min.

Probability / stats:

- `rand`, `randn`.
- `binornd`, `poissrnd`.
- `normpdf`, `normcdf`.

ODEs / DAEs:

- `ode45`: nonstiff IVPs.
- `ode23s`, `ode15s`: stiff IVPs.
- `ode15i`: implicit ODE/index-1 DAE.

Sparse / linear systems:

- `sparse`: create sparse matrices.
- `gmres`, `bicgstab`: iterative solvers.

13. Exam patterns & tricks

Linear algebra & discretization

- Always mention conditioning: ill-conditioned matrices amplify data/round-off error; well-conditioned ones do not, regardless of solver. Tridiagonal/banded systems from 1D FDM/FVM solve in $\mathcal{O}(N)$ vs. $\mathcal{O}(N^3)$ for dense.
- For "does a solution exist / is it unique?", map to four subspaces: existence $\Leftrightarrow b \in \text{range}(A)$; uniqueness $\Leftrightarrow \text{null}(A) = \{0\}$ (full column rank).

ODE IVPs

- For stability questions, immediately apply the test problem $\dot{x} = \lambda x$ and write $x_{n+1} = R(z)x_n$, $z = \lambda\Delta t$. Stability region is $\{z : |R(z)| \leq 1\}$. Forward Euler: $R(z) = 1 + z$; Backward Euler: $R(z) = 1/(1 - z)$.
- For accuracy, say: explicit/implicit Euler are first-order in Δt; midpoint / trapezoidal are second-order; RK4 is fourth-order. Global error order is one less than the local truncation error order.
- Stiffness: if eigenvalues span many orders of magnitude in negative real part, explicit methods need $\Delta t \ll 1/|\lambda_{\max}|$ purely for stability; implicit methods permit much larger Δt for the same problem.

ODE BVPs

- Shooting method: define the scalar residual $h(c)$ (e.g. $h(c) = y(b; c) - y_b$), then view it as a 1D root-finding problem. Newton in c typically needs $h'(c)$ via sensitivities or finite differences.
- Finite differences: interior points use centered stencils giving $\mathcal{O}(\Delta x^2)$ truncation error; if your Taylor expansion yields $\mathcal{O}(\Delta x)$ you almost certainly misapplied the stencil or indices.
- Think of linear BVPs as sparse linear systems: interior nodes give a tridiagonal (1D) or banded (2D) matrix; mention using backslash on sparse matrices for $\mathcal{O}(N)$–$\mathcal{O}(N \text{ band}^2)$ cost.

PDEs

- Classification: look only at second-derivative coefficients A, B, C in $Au_{xx} + Bu_{xy} + Cu_{yy} + \cdots = 0$. If $B^2 - 4AC < 0$ elliptic, $= 0$ parabolic, > 0 hyperbolic. Lower-order terms do not affect type.
- FDM/FVM: "balance form" answers are valued. Always write: accumulation = in-flux − out-flux + reaction; then insert 2-point fluxes. This is often enough for most of the credit even if algebra gets messy.
- Method of lines: say "discretize space only, get large ODE-IVP system in time, then choose explicit/implicit ODE solver based on stiffness (e.g. ode45 vs. ode15s)." Mention this pipeline explicitly.
- Von Neumann / CFL: plug Fourier mode e^{ikx} into the scheme to get growth factor $G(k)$; require $|G(k)| \leq 1$ for all k. For upwind advection, recover $0 \leq \text{CFL} \leq 1$; central advection + FE is typically unstable.

DAEs

- Index trick: for semi-explicit DAEs $x' = f(t, x, y)$, $0 = g(t, x, y)$, differentiate algebraic equations until you can solve for all derivatives; the number of differentiations is the index (ODE-IVPs are index-0).
- Consistent initialization: check both explicit algebraic constraints and any hidden ones revealed by differentiation at t_0. If (x_0, y_0) does not satisfy all, the ICs are inconsistent.
- Mass-matrix form: if $M\dot{z} = F(t, z)$ with singular M, say "this is a DAE; use an implicit stiff solver (e.g. ode15s / ode15i) and supply a consistent initial condition, possibly computed with a 'decic'-style routine."

Probability, estimation, MCMC

- Likelihood vs. prior vs. posterior: exams love "write the likelihood" and "derive MLE / MAP" for simple models (normal, Poisson, exponential). Always write posterior $\propto$ likelihood $\times$ prior and normalize only if asked.
- For linear regression / experimental design, tie "better experiment" answers to the design matrix X: more spread in inputs and less collinearity $\Rightarrow$ more informative (smaller parameter covariance).
- MCMC "integral form": the standard answer is $\dfrac{\int g(x)f(x)\,dx}{\int f(x)\,dx}$. Recognize code patterns where samples are used to estimate expectations of this form; mention unbiasedness as $M \to \infty$.

14. Taylor Expansion

A Taylor expansion of a sufficiently smooth scalar function f about a point x_0 is

$$f(x) \approx \sum_{k=0}^{n} \frac{f^{(k)}(x_0)}{k!}(x - x_0)^k.$$

For specific orders:

$$\text{Order 1: } f(x) \approx f(x_0) + f'(x_0)(x - x_0),$$

$$\text{Order 2: } f(x) \approx f(x_0) + f'(x_0)(x - x_0) + \frac{1}{2}f''(x_0)(x - x_0)^2,$$

$$\text{Order 3: } f(x) \approx f(x_0) + f'(x_0)(x - x_0) + \frac{1}{2}f''(x_0)(x - x_0)^2 + \ldots$$
$$\ldots + \frac{1}{6}f^{(3)}(x_0)(x - x_0)^3.$$

Finite difference coefficients

Central finite differences (uniform h)

Deriv.	Acc.	−5	−4	−3	−2	−1	0	1	2	3	4	5
1	2					−1/2	0	1/2				
1	4				1/12	−2/3	0	2/3	−1/12			
1	6			−1/60	3/20	−3/4	0	3/4	−3/20	1/60		
1	8		1/280	−4/105	1/5	−4/5	0	4/5	−1/5	4/105	−1/280	
2	2					1	−2	1				
2	4				−1/12	4/3	−5/2	4/3	−1/12			
2	6			1/90	−3/20	3/2	−49/18	3/2	−3/20	1/90		
2	8		−1/560	8/315	−1/5	8/5	−205/72	8/5	−1/5	8/315	−1/560	
3	2				−1/2	1	0	−1	1/2			
3	4			1/8	−1	13/8	0	−13/8	1	−1/8		
3	6		−7/240	3/10	−169/120	61/30	0	−61/30	169/120	−3/10	7/240	
4	2				1	−4	6	−4	1			
4	4			−1/6	2	−13/2	28/3	−13/2	2	−1/6		
4	6		7/240	−2/5	169/60	−122/15	91/8	−122/15	169/60	−2/5	7/240	
5	2			−1/2	2	−5/2	0	5/2	−2	1/2		
5	4		1/6	−3/2	13/3	−29/6	0	29/6	−13/3	3/2	−1/6	
5	6	−13/288	19/36	−87/32	13/2	−323/48	0	323/48	−13/2	87/32	−19/36	13/288
6	2			1	−6	15	−20	15	−6	1		
6	4		−1/4	3	−13	29	−75/2	29	−13	3	−1/4	
6	6	13/240	−19/24	87/16	−39/2	323/8	−1023/20	323/8	−39/2	87/16	−19/24	13/240

Forward finite differences (uniform h)

Deriv.	Acc.	0	1	2	3	4	5	6	7	8
1	1	−1	1							
1	2	−3/2	2	−1/2						
1	3	−11/6	3	−3/2	1/3					
1	4	−25/12	4	−3	4/3	−1/4				
1	5	−137/60	5	−5	10/3	−5/4	1/5			
1	6	−49/20	6	−15/2	20/3	−15/4	6/5	−1/6		
2	1	1	−2	1						
2	2	2	−5	4	−1					
2	3	35/12	−26/3	19/2	−14/3	11/12				
2	4	15/4	−77/6	107/6	−13	61/12	−5/6			
2	5	203/45	−87/5	117/4	−254/9	33/2	−27/5	137/180		
2	6	469/90	−223/10	879/20	−949/18	41	−201/10	1019/180	−7/10	
3	1	−1	3	−3	1					
3	2	−5/2	9	−12	7	−3/2				
3	3	−17/4	71/4	−59/2	49/2	−41/4	7/4			
3	4	−49/8	29	−461/8	62	−307/8	13	−15/8		
3	5	−967/120	638/15	−3929/40	389/3	−2545/24	268/5	−1849/120	29/15	
3	6	−801/80	349/6	−18353/120	2391/10	−1457/6	4891/30	−561/8	527/30	−469/240
4	1	1	−4	6	−4	1				
4	2	3	−14	26	−24	11	−2			
4	3	35/6	−31	137/2	−242/3	107/2	−19	17/6		
4	4	28/3	−111/2	142	−1219/6	176	−185/2	82/3	−7/2	
4	5	1069/80	−1316/15	15289/60	−2144/5	10993/24	−4772/15	2803/20	−536/15	967/240

Backward finite differences (uniform h)

Deriv.	Acc.	−8	−7	−6	−5	−4	−3	−2	−1	0
1	1								−1	1
1	2							1/2	−2	3/2
1	3						−1/3	3/2	−3	11/6
2	1							1	−2	1
2	2						−1	4	−5	2
3	1						−1	3	−3	1
3	2					3/2	−7	12	−9	5/2
4	1					1	−4	6	−4	1
4	2				−2	11	−24	26	−14	3