A Brief Introduction to Computational Physics

Jeongbin Jo[1, *]

[1]*Department of Physics, Yonsei University, Seoul, 03722, Republic of Korea*

(Dated: May 27, 2026)

This report provides a comprehensive overview of fundamental topics covered in the PHY3109 Computational Physics course. Computational physics sits at the intersection of physics, computer science, and applied mathematics, offering numerical methods to solve complex physical systems where analytical solutions are intractable. We explore core numerical techniques including numerical integration, probability distribution modeling, statistical data analysis, maximum-likelihood estimation, uncertainty propagation, and numerical solutions to ordinary differential equations (ODEs), followed by root-finding algorithms for nonlinear equations. We then discuss Fourier analysis and its application to scattering form factors, Bayesian inference methods including iterative unfolding and Markov Chain Monte Carlo (MCMC) for parameter estimation in dynamic systems, and neural network architectures including logistic regression, perceptrons, multi-layer networks, backpropagation, Hopfield associative memory, and modern optimization for machine learning tasks. Finally, we present an end-to-end application in high-energy physics: the extraction of the $Z \to \mu^+\mu^-$ signal from ATLAS open data using a neural-network classifier with dropout regularization and significance-driven working-point optimization.

CONTENTS

I. INTRODUCTION

Physics traditionally relies on theoretical derivation and experimental observation. However, as physical models encompass realistic boundary conditions, external non-linear per-

* jeongbin033@yonsei.ac.kr

turbations, and many-body interactions, their mathematical representations rapidly elude exact analytical solutions. Computational physics bridges this gap between theoretical formulation and empirical data by mapping continuous differential and integral equations into discrete numerical algorithms that are solvable computationally.

This report provides a comprehensive foundation for the mathematical instruments employed in the PHY3109 Computational Physics course, formulated using Python. Following the progression of the course, we begin with numerical integration and statistical data analysis, proceed through differential equations and root-finding, advance to Fourier methods and scattering physics, develop Bayesian inference and unfolding techniques, and conclude with neural network architectures and machine learning optimization. Note that the standard deviation notation σ will be utilized distinctly across the statistical and optimization derivations according to empirical physical conventions.

II. NUMERICAL INTEGRATION ALGORITHMS

Evaluating the definite integral $I = \int_a^b f(x)\,dx$ is a persistent requirement. When analytical antiderivatives do not exist or are prohibitively complex, numerical quadratures and probabilistic methods are deployed.

A. Newton-Cotes Quadrature and Simpson's Rule

Deterministic grid-based algorithms approximate the integration domain $[a, b]$ into evenly spaced sub-intervals defined by a step size $h = \frac{b-a}{N}$. The Trapezoidal rule fits linear polynomials between points, yielding the composite formulation:

$$\int_a^b f(x)\,dx \approx \frac{h}{2}\left[f(x_0) + 2\sum_{i=1}^{N-1} f(x_i) + f(x_N)\right] \tag{1}$$

This linear interpolation imparts a global truncation error of $O(h^2)$. Alternatively, Simpson's $\frac{1}{3}$ rule approximates the local curvature using second-order polynomials (parabolas) to achieve higher precision. By utilizing the Taylor series expansion at the midpoint x_1 between $[x_0, x_2]$ (where $h = x_1 - x_0 = x_2 - x_1$), integrating the quadratic interpolation yields:

$$\int_{x_0}^{x_2} f(x)\,dx \approx \frac{h}{3}\left[f(x_0) + 4f(x_1) + f(x_2)\right] \tag{2}$$

When generalized across N (even) contiguous intervals, the composite Simpson's $\frac{1}{3}$ formula is derived:

$$\int_a^b f(x)\,dx \approx \frac{h}{3}\Bigg[f(x_0) + f(x_N) + 4\sum_{i=1,3,5}^{N-1} f(x_i) + 2\sum_{j=2,4,6}^{N-2} f(x_j)\Bigg] \tag{3}$$

The local truncation error scales analogously to the fourth derivative of the function, yielding $O(h^5)$ locally and $O(h^4)$ globally, establishing it as the standard for 1D deterministic integration.

B. Monte Carlo Probabilistic Integration

Grid-based geometries scale exponentially in computational cost with dimensionality (the "curse of dimensionality"). Monte Carlo methods circumvent this by treating the integral as an Expected Value computation. By sampling N statistically independent random coordinates uniformly distributed within a high-dimensional volume V, the integral translates to the geometric mean formulation:

$$I = \int_V f(\mathbf{x})\,d\mathbf{x} \approx \frac{V}{N}\sum_{i=1}^{N} f(\mathbf{x}_i) = V\langle f\rangle \tag{4}$$

The uncertainty of this approximation is not bounded by grid granularity but governed by the Central Limit Theorem. The variance of the approximation scales with the variance of the function itself (σ_f^2):

$$\sigma_I \approx V\frac{\sigma_f}{\sqrt{N}} \tag{5}$$

Thus, the precision converges as $1/\sqrt{N}$, rendering Monte Carlo unconditionally superior for systems involving massive degrees of freedom, such as statistical mechanics ensembles.

III. STATISTICAL DATA ANALYSIS AND OPTIMIZATION FRAMEWORK

Real-world physical measurements intrinsically contain noise. Isolating deterministic signals from stochastic deviations requires rigorous statistical estimators and optimization algorithms.

A. Descriptive Statistics: Mean and Variance

Before initiating optimization, the intrinsic empirical distribution must be established. For a recorded dataset comprising N individual measurements $\{x_1, x_2, \ldots, x_N\}$, the primary central tendency estimator is the arithmetic sample mean $\bar{x}$:

$$\bar{x} = \frac{1}{N}\sum_{i=1}^{N} x_i \tag{6}$$

To quantify the stochastic spread of the data, variance and standard deviation are calculated. Maintaining the universally adopted σ notation, the unbiased sample standard deviation σ incorporates Bessel's correction $(N-1)$ to compensate for the loss of a degree of freedom when the population mean is approximated by the sample mean:

$$\sigma^2 = \frac{1}{N-1}\sum_{i=1}^{N}(x_i - \bar{x})^2, \quad \sigma = \sqrt{\sigma^2} \tag{7}$$

B. Probability Distribution Models

A probability distribution is the mathematical object that specifies how experimental outcomes are expected to fluctuate. In computational physics, the distribution is not just a descriptive curve drawn after data are collected; it is a generative model that connects physical assumptions to numerical

prediction. Once a distribution is chosen, one can simulate pseudo-data, construct a likelihood, estimate unknown parameters, and quantify the probability of observing a particular experimental result.

The first important distinction is between continuous probability density functions (PDFs) and discrete probability mass functions (PMFs). A PDF $f(x)$ describes a density over a continuous variable, so the probability of finding X inside an interval is

$$P(a \le X \le b) = \int_a^b f(x)dx, \tag{8}$$

with normalization $\int_{-\infty}^{\infty} f(x)dx = 1$. The value $f(x)$ itself is not a probability and may even exceed unity if the distribution is narrow. A PMF $p(k)$, by contrast, gives the probability of a discrete outcome directly:

$$P(X = k) = p(k), \qquad \sum_k p(k) = 1. \tag{9}$$

This distinction is central because measured positions, voltages, energies, and decay times are often modeled continuously, whereas particle counts, decay counts, and detector hits are discrete random variables.

Several distributions appear repeatedly in physical data analysis. The Gaussian distribution,

$$f(x; \mu, \sigma) = \frac{1}{\sqrt{2\pi}\sigma} \exp\left[-\frac{(x-\mu)^2}{2\sigma^2}\right], \tag{10}$$

models additive measurement noise and many averaged observables because independent fluctuations tend toward a normal distribution by the central limit theorem. Its parameter μ fixes the location of the distribution, while σ controls the width. The uniform distribution is appropriate when only a finite allowed interval is known and no internal preference is assumed:

$$f(x) = \frac{1}{b-a}, \qquad a \le x \le b. \tag{11}$$

It is also the starting point for many Monte Carlo sampling algorithms.

The exponential distribution is the natural continuous model for waiting times in a memoryless process. If τ is the mean lifetime, then

$$p(t; \tau) = \frac{1}{\tau} e^{-t/\tau}, \qquad t \ge 0. \tag{12}$$

Radioactive decay provides the standard physical example: the probability that a particle survives to a later time does not depend on how long it has already survived. This same rate-process assumption leads to the Poisson distribution for the number of events observed in a fixed interval.

For rare independent events occurring at a constant average rate, the Poisson distribution provides the canonical discrete model. If the expected number of events in a fixed time interval is λ, the probability of observing exactly k events is

$$P(k; \lambda) = \frac{\lambda^k e^{-\lambda}}{k!}. \tag{13}$$

Its mean and variance are both equal to λ, making it especially useful for low-count experiments such as radioactive decay and particle detection. In a decay-counting experiment, for instance, λ corresponds to the expected number of decays during the chosen measurement window. This makes the Poisson model a natural bridge between physical rate equations and statistical inference [1, 2].

Histograms provide the computational interface between sampled data and these probability models. If `density=True` is used in a numerical histogram routine, the bin heights estimate a probability density and must be multiplied by the bin width before being interpreted as probabilities. If raw counts are used instead, each bin count is itself a fluctuating discrete observable, commonly modeled with Poisson statistics. Therefore, the same dataset may be viewed either as a sampled approximation to an underlying PDF or as a collection of discrete bin counts, depending on the subsequent inference method.

C. Maximum Likelihood Estimation

Least-squares fitting is most naturally interpreted as a special case of likelihood maximization when the measurement errors are Gaussian. More generally, if independent measurements $\{x_i\}_{i=1}^N$ are described by a probability model $f(x; \vec{\theta})$, the likelihood of the parameter vector $\vec{\theta}$ is

$$\mathcal{L}(\vec{\theta}) = \prod_{i=1}^{N} f(x_i; \vec{\theta}). \tag{14}$$

In numerical work, the log-likelihood is preferred for stability:

$$\log \mathcal{L}(\vec{\theta}) = \sum_{i=1}^{N} \log f(x_i; \vec{\theta}). \tag{15}$$

The maximum-likelihood estimator (MLE) is the parameter value that maximizes $\mathcal{L}$, or equivalently minimizes $-\log \mathcal{L}$. For an exponential decay distribution $p(t; \tau) = \tau^{-1} e^{-t/\tau}$, maximizing the likelihood gives the mean-lifetime estimator

$$\hat{\tau} = \frac{1}{N} \sum_{i=1}^{N} t_i. \tag{16}$$

For a Gaussian model with unknown mean, the MLE similarly reduces to the sample mean. Parameter uncertainties can be estimated from the local curvature of $\log \mathcal{L}$ around its maximum; in the one-parameter case, the standard one-sigma interval satisfies $\Delta \log \mathcal{L} = -1/2$ under the usual quadratic approximation [3].

D. Least Squares Method (χ^2 Minimization)

To construct a theoretical model $f(x; \vec{\theta})$ that maximally matches the empirical data, one formulates an objective function representing the square of the deviations. If each data point y_i encompasses its own statistical error σ_i, the deviations are weighted proportionally, defining the rigorous chi-squared statistic χ^2:

$$\chi^2(\vec{\theta}) = \sum_{i=1}^{N} \left[\frac{y_i - f(x_i; \vec{\theta})}{\sigma_i}\right]^2 \tag{17}$$

The aim of regression modeling is strictly to discover the parameter vector $\vec{\theta}$ that minimizes this multidimensional χ^2 landscape.

When the data are stored as histogram bin counts, a binned likelihood can be used instead of fitting the normalized density values directly. If n_i is the observed count in bin i and $p_i(\vec{\theta})$ is the model probability for that bin, the multinomial log-likelihood contains the essential term

$$\log \mathcal{L}(\vec{\theta}) = \sum_i n_i \log p_i(\vec{\theta}) + \text{constant}. \tag{18}$$

This compresses the data and accelerates fitting, but the binning process inevitably discards some information compared with an unbinned likelihood.

E. Gradient Descent Optimization

While analytical minimization ($\nabla\chi^2 = 0$) exists for purely linear polynomials, complex physical non-linear models depend on numeric iteration. The **Gradient Descent** algorithm navigates the highly non-linear cost function $J(\vec{\theta}) \equiv \chi^2(\vec{\theta})$. The parameters evolve continuously against the direction of the local gradient manifold:

$$\theta_j^{(n+1)} = \theta_j^{(n)} - \alpha \frac{\partial J(\vec{\theta}^{(n)})}{\partial \theta_j} \tag{19}$$

Here, α is the systematically tuned *learning rate*. If α is too steep, the parameter trajectories will predictably diverge across the minimum basin; conversely, an infinitesimally small α restricts convergence within computational timeframes.

F. Error Analysis and Propagation of Uncertainty

A measured dependent physical phenomenon often convolves multiple independent measurements $f = f(x_1, x_2, \ldots, x_M)$. Errors inherent in individual instruments propagate through equations, necessitating rigorous boundary tracking. By utilizing a first-order multivariate Taylor expansion around the expected values, the generalized combined standard deviation (σ_f) propagating through the operational architecture can be derived. Considering the completely general case, inclusive of variable interdependence (correlation), the full covariance expansion yields:

$$\sigma_f^2 = \sum_{i=1}^{M} \left(\frac{\partial f}{\partial x_i}\right)^2 \sigma_{x_i}^2 + 2 \sum_{i<j} \left(\frac{\partial f}{\partial x_i}\right)\left(\frac{\partial f}{\partial x_j}\right) \text{cov}(x_i, x_j) \tag{20}$$

In this formulation, the partial derivative $\frac{\partial f}{\partial x_i}$ acts as a "sensitivity coefficient," dictating how strongly the measurement error σ_{x_i} amplifies within the final composite outcome.

In the general computational environment, variable orthogonality (zero covariance) cannot always be guaranteed. For instance, variables interconnected by systematic environmental drift will exhibit non-zero covariance ($\text{cov}(A, B) = \sigma_{AB}$). Using the fully expanded Taylor matrix, Table I specifies the generalized variance and standard deviation propagation rules for fundamental arithmetic and continuous physics functions.

This systematic propagation fundamentally restricts the precision limit of composite calculations; for instance, analyzing free-fall bounds inherently conflates the timing standard deviation (σ_t) squared against the dimensional metric deviation (σ_y), directly modulating the gravitational constant certainty σ_g.

IV. NUMERICAL SOLUTIONS TO DIFFERENTIAL EQUATIONS

The continuous physical world, codified by Newton's and Maxwell's equations, operates via differential mathematics. Simulation translates these continuous Initial Value Problems (IVPs), $\frac{dy}{dt} = f(t, y)$, into discrete temporal jumps Δt.

A. From Euler to Runge-Kutta Formulations

The classical Euler method is structurally derived from a truncated first-order Taylor expansion:

$$y(t_{n+1}) = y(t_n) + \Delta t \cdot y'(t_n) + O(\Delta t^2) \tag{21}$$

This linear extrapolation generates a disastrous systemic accumulation of truncation error corresponding globally to $O(\Delta t)$. To mitigate this, the *Modified Euler Method* (equivalent to a 2nd-order Runge-Kutta or Midpoint Method) evaluates the derivative at the half-step to construct a more accurate predictor-corrector paradigm:

$$\begin{aligned} k_1 &= f(t_n, y_n) \\ k_2 &= f\left(t_n + \frac{\Delta t}{2}, y_n + \frac{\Delta t}{2} k_1\right) \\ y_{n+1} &= y_n + \Delta t \cdot k_2 \end{aligned} \tag{22}$$

By extending this logic, the definitive solver in computational mechanics becomes the 4th-order Runge-Kutta (RK4). By rigorously cross-referencing slopes across four distinct spatial-temporal nodes uniformly distributed across the integration step, RK4 yields a highly stable numerical architecture:

$$\begin{aligned} k_1 &= f(t_n, y_n) \\ k_2 &= f\left(t_n + \frac{\Delta t}{2}, y_n + \frac{\Delta t}{2} k_1\right) \\ k_3 &= f\left(t_n + \frac{\Delta t}{2}, y_n + \frac{\Delta t}{2} k_2\right) \\ k_4 &= f(t_n + \Delta t, y_n + \Delta t k_3) \end{aligned} \tag{23}$$

$$y_{n+1} = y_n + \frac{\Delta t}{6}(k_1 + 2k_2 + 2k_3 + k_4) + O(\Delta t^5) \tag{24}$$

A global truncation error of $O(\Delta t^4)$ permits substantially enlarged temporal steps without triggering geometric instability.

B. Physical Applications: Trajectories and Fields

Translating abstract numeric frameworks into applied kinematic and electromagnetic simulations requires defining specific governing equations.

1. 1D Free-Fall Kinematics

Consider a spherical object (mass m, cross-sectional area A) under constant gravitational acceleration g, subjected to quadratic aerodynamic drag modeled by the fluid density ρ and drag coefficient c_d. The governing kinematic differential equation derived from Newton's second law is:

$$m\frac{dv}{dt} = -mg + \frac{1}{2}\rho v^2 c_d A \tag{25}$$

TABLE I. Comprehensive Error Propagation Framework (including covariance σ_{AB})

Function f	**Variance** σ_f^2	**Standard Deviation** σ_f
$f = aA$	$a^2\sigma_A^2$	$\lvert a\rvert\sigma_A$
$f = aA \pm bB$	$a^2\sigma_A^2 + b^2\sigma_B^2 \pm 2ab\sigma_{AB}$	$\sqrt{a^2\sigma_A^2 + b^2\sigma_B^2 \pm 2ab\sigma_{AB}}$
$f = AB$	$f^2\left[\left(\frac{\sigma_A}{A}\right)^2 + \left(\frac{\sigma_B}{B}\right)^2 + 2\frac{\sigma_{AB}}{AB}\right]$	$\lvert f\rvert\sqrt{\left(\frac{\sigma_A}{A}\right)^2 + \left(\frac{\sigma_B}{B}\right)^2 + 2\frac{\sigma_{AB}}{AB}}$
$f = \frac{A}{B}$	$f^2\left[\left(\frac{\sigma_A}{A}\right)^2 + \left(\frac{\sigma_B}{B}\right)^2 - 2\frac{\sigma_{AB}}{AB}\right]$	$\lvert f\rvert\sqrt{\left(\frac{\sigma_A}{A}\right)^2 + \left(\frac{\sigma_B}{B}\right)^2 - 2\frac{\sigma_{AB}}{AB}}$
$f = aA^b$	$\left(\frac{fb\sigma_A}{A}\right)^2$	$\left\lvert\frac{fb\sigma_A}{A}\right\rvert$
$f = a\ln(bA)$	$\left(a\frac{\sigma_A}{A}\right)^2$	$\left\lvert a\frac{\sigma_A}{A}\right\rvert$
$f = a\log_{10}(bA)$	$\left(a\frac{\sigma_A}{A\ln(10)}\right)^2$	$\left\lvert a\frac{\sigma_A}{A\ln(10)}\right\rvert$
$f = ae^{bA}$	$f^2(b\sigma_A)^2$	$\lvert f\rvert\,\lvert b\sigma_A\rvert$
$f = a^{bA}$	$f^2(b\ln(a)\sigma_A)^2$	$\lvert f\rvert\,\lvert b\ln(a)\sigma_A\rvert$
$f = a\sin(bA)$	$[ab\cos(bA)\sigma_A]^2$	$\lvert ab\cos(bA)\sigma_A\rvert$
$f = a\cos(bA)$	$[ab\sin(bA)\sigma_A]^2$	$\lvert ab\sin(bA)\sigma_A\rvert$
$f = A^B$	$f^2\left[\left(\frac{B}{A}\sigma_A\right)^2 + (\ln(A)\sigma_B)^2 + 2\frac{B\ln(A)}{A}\sigma_{AB}\right]$	$\lvert f\rvert\sqrt{\left(\frac{B}{A}\sigma_A\right)^2 + (\ln(A)\sigma_B)^2 + 2\frac{B\ln(A)}{A}\sigma_{AB}}$

Discretizing this via the Euler method entails twin iterative updates for position y and velocity v:

$$\begin{aligned} v_{n+1} &= v_n + \left(-g + \frac{\rho c_d A}{2m}v_n^2\right)\Delta t \\ y_{n+1} &= y_n + v_n\Delta t \end{aligned} \tag{26}$$

To pinpoint the exact bounding collision time (e.g., $y = 0$) when simulation frames overshoot the spatial boundary condition ($y_{n+1} < 0$), linear interpolation bridges the gap between the pre-collision state (t_n, y_n) and the post-collision state (t_{n+1}, y_{n+1}):

$$t_{\text{land}} = t_n + \frac{y_n}{y_n - y_{n+1}}\Delta t \tag{27}$$

Compared to purely analytical parabolic solutions applicable only in a vacuum ($y(t) = h - \frac{1}{2}gt^2$), these coupled numerical techniques faithfully simulate realistic atmospheric entry dynamics where strictly analytical inversions fail.

2. *Electrically Charged Particle Dynamics*

Computing particle motions within intricate multidimensional electric fields entails evaluating the Lorentz force iteratively. The composite E-field vector at position $\vec{r}$ induced by an assembly of discrete point charges q_i localized at $\vec{r}_i$ is defined by Coulomb's superposition:

$$\vec{E}(\vec{r}) = \frac{1}{4\pi\epsilon_0}\sum_i q_i\frac{\vec{r} - \vec{r}_i}{|\vec{r} - \vec{r}_i|^3} \tag{28}$$

In fully generalized multidimensional simulations, moving charges also interact with external magnetic fields $\vec{B}$, necessitating the complete Lorentz force equation. The kinematic acceleration thus maps directly from the combined fields via:

$$\frac{d\vec{v}}{dt} = \frac{q}{m}\left(\vec{E}(\vec{r}) + \vec{v}\times\vec{B}(\vec{r})\right) \tag{29}$$

Transferred into a numeric solver like RK4, the cross-product formulation couples the velocity components, yielding high-fidelity simulations of complex particle scatterings, cyclotron motions, or orbital captures.

V. ROOT FINDING FOR NONLINEAR EQUATIONS

Extracting boundary conditions, energy eigenvalues, or threshold intersections often entails solving arbitrary nonlinear equations implicitly, $f(x) = 0$, where algebraic inversion fails. Computational solvers isolate these roots iteratively. Advanced algorithms are fundamentally characterized by their

mathematical *Order of Convergence* p, defined via the asymptotic error relationship $e_{n+1} \approx C|e_n|^p$. Higher orders of p drastically reduce the required computational iterations.

A. Bisection Method

The Bisection method relies fundamentally on the Intermediate Value Theorem. Given an initial bracket $[a_0, b_0]$ where $f(a_0)f(b_0) < 0$ (indicating a zero-crossing), the algorithm iteratively halves the search space. At each step n, the midpoint $c_n = \frac{a_n+b_n}{2}$ is evaluated. The sub-interval containing the root is retained by discarding the boundary that shares the exact same sign as $f(c_n)$. The absolute bound on the error after n iterations is strictly deterministic:

$$|x_{\text{root}} - c_n| \leq \frac{|b_0 - a_0|}{2^{n+1}} \tag{30}$$

While guaranteed to converge (unconditional stability), the linear convergence rate restricts its utility when high-precision roots are required computationally rapidly.

B. Newton-Raphson Method

To vastly accelerate convergence, the Newton-Raphson method exploits the local gradient (analytical derivative). By applying a first-order Taylor expansion around the current guess x_n:

$$f(x) \approx f(x_n) + f'(x_n)(x - x_n) = 0 \tag{31}$$

Solving for the linear intercept $(x - x_n)$ yields the recursive update rule:

$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)} \tag{32}$$

Assuming a sufficiently close initial guess and a non-zero derivative at the root ($f'(\alpha) \neq 0$), the Newton-Raphson method boasts exceptional quadratic convergence, effectively doubling the number of correct decimal digits per iteration. However, extracting the analytical derivative $f'(x)$ is often algorithmically impossible for complex numerical physical models.

C. Secant Method

When the analytical derivative is inaccessible or computationally expensive to extract, the Secant method approximates $f'(x_n)$ utilizing a finite difference scheme constructed from the two most recent iterations:

$$f'(x_n) \approx \frac{f(x_n) - f(x_{n-1})}{x_n - x_{n-1}} \tag{33}$$

Substituting this geometrical approximation back into the Newton iteration maps a secant line between the two latest coordinates:

$$x_{n+1} = x_n - f(x_n)\frac{x_n - x_{n-1}}{f(x_n) - f(x_{n-1})} \tag{34}$$

The Secant method demands two initial conditions (x_0, x_1) and avoids bracket containment. Its convergence scaling is superlinear (specifically ≈ 1.618, the golden ratio), isolating a pristine balance between computational simplicity (no calculus required) and convergence velocity.

VI. FOURIER ANALYSIS AND SIGNAL DIAGNOSTICS

A. From Fourier Series to Fourier Transform

The fundamental premise of Fourier analysis is that any physically realizable signal $f(t)$ can be decomposed into a superposition of sinusoidal basis functions. For a periodic signal with period T, the Fourier series expansion in complex exponential form is:

$$f(t) = \sum_{n=-\infty}^{\infty} c_n e^{in\omega_0 t}, \quad \omega_0 = \frac{2\pi}{T} \tag{35}$$

where the discrete spectral coefficients c_n are determined by the projection:

$$c_n = \frac{1}{T}\int_{-T/2}^{T/2} f(t)e^{-in\omega_0 t}\, dt \tag{36}$$

As $T \to \infty$, the fundamental frequency spacing $\Delta\omega = \omega_0$ becomes infinitesimal $d\omega$, and the discrete summation transitions into a continuous integral. This limit yields the Fourier Transform pair, bridging the gap between periodic steady-states and transient physical phenomena:

$$f(t) = \frac{1}{2\pi}\int_{-\infty}^{\infty} F(\omega)e^{i\omega t}\, d\omega \tag{37}$$

$$F(\omega) = \int_{-\infty}^{\infty} f(t)e^{-i\omega t}\, dt \tag{38}$$

B. Mathematical Formalism and Physical Computation

In computational physics, the choice of normalization constant and sign convention depends on the specific domain (e.g., Quantum Mechanics vs. Signal Processing). While the $1/2\pi$ factor above is common in physics to preserve the angular frequency ω relation, the most physically significant result is the conservation of energy, articulated by **Parseval's Theorem**:

$$\int_{-\infty}^{\infty} |f(t)|^2\, dt = \frac{1}{2\pi}\int_{-\infty}^{\infty} |F(\omega)|^2\, d\omega \tag{39}$$

This theorem ensures that the total "energy" (or power density) of a signal is invariant under the transformation, a critical requirement for analyzing wave functions $|\psi(x)|^2$ or electrical power spectra. Furthermore, the spread of a packet in time (Δt) and frequency ($\Delta\omega$) is restricted by the Uncertainty Principle, $\Delta t \Delta\omega \geq 1/2$, which dictates the limits of signal resolution in experimental physics.

C. Position–Momentum Representation in Physics

One of the most important uses of the Fourier transform in physics is the transformation between position space and momentum space. In quantum mechanics, a wave function $\psi(x)$ describes the probability amplitude for finding a particle near position x, while its momentum-space wave function $\phi(p)$ describes the probability amplitude for measuring momentum p. The two representations contain the same physical state, but

emphasize different observables. With a common symmetric normalization convention, the transform pair is

$$\phi(p) = \frac{1}{\sqrt{2\pi\hbar}} \int_{-\infty}^{\infty} \psi(x) e^{-ipx/\hbar} dx, \tag{40}$$

$$\psi(x) = \frac{1}{\sqrt{2\pi\hbar}} \int_{-\infty}^{\infty} \phi(p) e^{ipx/\hbar} dp. \tag{41}$$

The exponential basis function $e^{ipx/\hbar}$ is a plane wave with definite momentum p. Therefore, Fourier transformation decomposes a spatial wave packet into the plane-wave momentum components from which it is built [4].

The probability interpretation is preserved by normalization:

$$\int_{-\infty}^{\infty} |\psi(x)|^2 dx = \int_{-\infty}^{\infty} |\phi(p)|^2 dp = 1. \tag{42}$$

Thus $|\psi(x)|^2 dx$ is the probability of locating the particle inside a small spatial interval dx, while $|\phi(p)|^2 dp$ is the probability of measuring momentum inside dp. A wave function sharply localized in position must be assembled from many momentum components, while a nearly monochromatic plane wave has a sharply defined momentum but is spread out in space. This reciprocal relation gives the Heisenberg uncertainty principle in its standard form,

$$\Delta x \, \Delta p \geq \frac{\hbar}{2}. \tag{43}$$

The same transformation also simplifies differential operators. Since

$$-i\hbar \frac{\partial}{\partial x} e^{ipx/\hbar} = p e^{ipx/\hbar}, \tag{44}$$

the momentum operator in position space, $\hat{p} = -i\hbar\partial/\partial x$, becomes simple multiplication by p in momentum space. Likewise, the kinetic-energy operator

$$\hat{T} = -\frac{\hbar^2}{2m} \frac{\partial^2}{\partial x^2} \tag{45}$$

becomes multiplication by $p^2/2m$. This is why Fourier methods are widely used in computational quantum mechanics: switching to momentum space can diagonalize the kinetic term, while position space is often more convenient for local potentials $V(x)$.

In three dimensions, the same idea becomes

$$\phi(\vec{p}) = \frac{1}{(2\pi\hbar)^{3/2}} \int \psi(\vec{r}) e^{-i\vec{p}\cdot\vec{r}/\hbar} d^3r. \tag{46}$$

This form is directly connected to scattering theory. Momentum transfer $\vec{q} = \vec{p}_i - \vec{p}_f$ appears in the phase factor $e^{i\vec{q}\cdot\vec{r}/\hbar}$, so the nuclear form factor introduced below is essentially the Fourier transform of the spatial charge distribution. In this sense, scattering experiments measure momentum-space information and use Fourier analysis to infer spatial structure.

D. Discrete and Inverse Fourier Transforms

Converting the continuous theoretical frameworks into computational algorithms requires discretization. A sampled numeric dataset limits operations to the Discrete Fourier Transform (DFT), where the integral is replaced by a finite summation. This transition is essential for analyzing experimental signals, which are often corrupted by Gaussian noise: $x_{\text{noisy}}(t) = x(t) + \mathcal{N}(0, \sigma^2)$. Defined over N uniform discrete time samples x_n, the transformation projects the signal onto orthogonal sinusoidal basis vectors:

$$X_k = \sum_{n=0}^{N-1} x_n e^{-i\left(\frac{2\pi}{N}\right)kn} \tag{47}$$

Conversely, the physical time-domain signal is precisely reconstructed via the Inverse Discrete Fourier Transform (IDFT):

$$x_n = \frac{1}{N} \sum_{k=0}^{N-1} X_k e^{i\left(\frac{2\pi}{N}\right)kn} \tag{48}$$

E. Frequency Resolution and Aliasing

When employing algorithms like NumPy's FFT, understanding the spectral topology is paramount. The frequency resolution is inherently governed by the sampling interval Δt and array size N, forming frequency bins $f_k = \frac{k}{N\Delta t}$. Furthermore, the maximum identifiable frequency is rigidly bound by the Nyquist cutoff threshold:

$$f_{\text{cutoff}} = \frac{1}{2\Delta t} \tag{49}$$

Analyzing a composite waveform, for example, $x(t) = \sin(4\pi t) + 0.5\sin(10\pi t)$ comprising 2 Hz and 5 Hz components, FFT neatly isolates these localized spikes out of random background noise. If an identical signal suffers a time shift (delay), the fundamental power spectrum magnitude $|X_k|$ remains perfectly invariant; the temporal delay manifests exclusively as a linear phase shift across the complex argument [5, 6].

VII. SCATTERING, CROSS SECTIONS, AND FORM FACTORS

A. Cross Sections and Scattering Rates

Scattering experiments connect microscopic interaction models to measurable event rates. If a beam with flux J illuminates N target particles, the rate for a specific process can be written as

$$W_r = JN\sigma_r \equiv \mathcal{L}\sigma_r, \tag{50}$$

where σ_r is the cross section and $\mathcal{L}$ is the luminosity. The angular distribution is encoded in the differential cross section,

$$dW_r = JN \frac{d\sigma_r}{d\Omega} d\Omega, \tag{51}$$

which is the experimentally accessible quantity when final-state particles are counted in a detector covering a finite solid angle.

For a pointlike Coulomb potential, the Born approximation analytically yields the Rutherford scattering differential cross section for a target of atomic number Z and an incident particle of charge ze and kinetic energy E:

$$\left(\frac{d\sigma}{d\Omega}\right)_{\text{Rutherford}} = \left(\frac{zZe^2}{4\pi\epsilon_0}\right)^2 \frac{1}{(4E)^2 \sin^4(\theta/2)} \tag{52}$$

In relativistic electron scattering, spin and kinematic corrections scale this baseline, producing the Mott cross section $\left(\frac{d\sigma}{d\Omega}\right)_{\text{Mott}} = \left(\frac{d\sigma}{d\Omega}\right)_{\text{Rutherford}} \left(1 - \frac{v^2}{c^2}\sin^2\frac{\theta}{2}\right)$. These theoretical point-source results provide a pristine baseline; deviations from them experimentally reveal finite spatial structure inside the target. This is why electron-nucleus scattering is a powerful computational and experimental probe of charge distributions [4].

B. Nuclear Form Factors as Fourier Transforms

For an extended charge distribution $\rho(\vec{r})$ normalized by $\int \rho(\vec{r})d^3r = 1$, the scattering amplitude is multiplied by a form factor,

$$F(\vec{q}) = \int \rho(\vec{r})e^{i\vec{q}\cdot\vec{r}/\hbar}d^3r, \tag{53}$$

where $\vec{q}$ is the momentum transfer. The measured cross section is therefore related to the pointlike scattering prediction by

$$\left(\frac{d\sigma}{d\Omega}\right)_{\text{expt}} = \left(\frac{d\sigma}{d\Omega}\right)_{\text{Mott}} |F(q^2)|^2. \tag{54}$$

For a spherically symmetric distribution, the three-dimensional Fourier transform reduces to the spherical form

$$F(q) = 4\pi \int_0^\infty \rho(r)\frac{\sin(qr/\hbar)}{qr/\hbar}r^2 dr. \tag{55}$$

Conversely, inverse Fourier transformation can reconstruct $\rho(r)$ from measured $F(q)$. This directly connects scattering physics to the Fourier methods discussed in the previous section. It also motivates common nuclear charge-density parametrizations such as the Woods-Saxon form,

$$\rho(r) = \frac{\rho_0}{1 + \exp[(r-a)/b]}, \tag{56}$$

where a controls the nuclear radius scale and b controls the surface diffuseness.

VIII. BAYESIAN INFERENCE AND ITERATIVE UNFOLDING

A. Bayes' Theorem as an Updating Rule

Bayesian inference formalizes learning from observed data. For a hypothesis or cause A and an observed outcome B, Bayes' theorem reads

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)}. \tag{57}$$

Here $P(A)$ is the prior probability, $P(B|A)$ is the likelihood, $P(B)$ is the evidence, and $P(A|B)$ is the posterior probability. This distinction is not merely semantic: a rare disease with a sensitive test may still have a modest posterior probability after a positive result if the false-positive rate is large compared with the disease prevalence. In experimental physics, the same logic appears when one asks which true physical process most likely produced a measured detector signal.

B. Detector Response and Response Matrices

Real detectors smear and bias the underlying physical distribution. If T_j is the true distribution in truth bin j and M_i is the measured distribution in detector bin i, the detector response can be encoded in a response matrix

$$R_{ij} = P(\text{measured bin } i|\text{truth bin } j). \tag{58}$$

The forward detector model is

$$M_i = \sum_j R_{ij}T_j. \tag{59}$$

Direct matrix inversion is often unstable because finite statistics, bin migration, and near-singular response matrices amplify noise. Unfolding therefore seeks a regularized estimate of the truth distribution rather than a naive inverse solution.

C. Iterative Bayesian Unfolding

Iterative Bayesian unfolding applies Bayes' theorem bin by bin. Starting from an initial prior $P^{(0)}(j)$, often chosen as a flat distribution or a simulation-motivated distribution, the probability that an event originated in truth bin j given that it was measured in bin i is

$$P^{(n)}(j|i) = \frac{R_{ij}P^{(n)}(j)}{\sum_k R_{ik}P^{(n)}(k)}. \tag{60}$$

The measured spectrum is then mapped back into truth space:

$$T_j^{(n+1)} = \sum_i P^{(n)}(j|i)M_i. \tag{61}$$

After normalization, this unfolded distribution becomes the prior for the next iteration. A small number of iterations suppresses detector smearing while avoiding excessive amplification of statistical fluctuations; too many iterations can overfit noise. This tradeoff makes the iteration count an implicit regularization parameter [7].

The same structure extends naturally to image analysis. A blurred image can be written as a true image convolved with a Gaussian response kernel, plus additive noise. Iterative Bayesian correction updates a flat or weakly informed initial image by comparing the measured image to the current forward-smeared estimate. In high-energy physics, an analogous procedure is used for jet transverse-momentum spectra, where one compares unfolded p_T distributions obtained from different priors, such as a flat prior and a measured-spectrum prior.

D. Markov Chain Monte Carlo (MCMC)

In Bayesian analysis, the primary objective is to compute the posterior distribution:

$$P(\vec{\theta}|D) \propto P(D|\vec{\theta})P(\vec{\theta}) \tag{62}$$

where $\vec{\theta}$ represents the parameters to be estimated, D is the observed data, $P(D|\vec{\theta})$ is the likelihood, and $P(\vec{\theta})$ is the prior probability. The fundamental difficulty arises when the model encompasses a high-dimensional parameter space; evaluating the exact normalization constant (the evidence $P(D)$) requires

computing a complex multidimensional integral, which is often analytically intractable.

To circumvent this, Markov Chain Monte Carlo (MCMC) methods, such as the Metropolis–Hastings algorithm, are deployed. MCMC generates a stochastic sequence of parameter samples where the frequency of samples asymptotically converges to the true target posterior distribution. By evaluating the probability ratio of a proposed parameter state versus the current state, the algorithm accepts or rejects proposals, effectively mapping the complex topography of the posterior without requiring direct integration.

E. Bayesian Analysis: Dynamic Systems and Brownian Motion

A fundamental application of MCMC involves estimating phenomenological parameters of dynamic systems from noisy experimental data. For a macroscopic example, one might estimate the deterministic parameters of a damped harmonic oscillator—such as the amplitude A, damping constant γ, and angular frequency ω—by extracting them from noisy temporal observations via Bayesian parameter optimization.

To model the stochastic dynamics of such systems at the microscopic level, we turn to **stochastic differential equations (SDEs)**. Unlike ordinary differential equations (ODEs), which describe deterministic evolution, SDEs incorporate random noise terms that model thermal fluctuations, environmental variability, or other stochastic influences inherent in physical systems.

1. *Stochastic Differential Equations: General Framework*

A stochastic differential equation extends the deterministic ODE $\dot{x} = f(x,t)$ by adding a noise term driven by a Wiener process $W(t)$:

$$dx = f(x,t)\,dt + g(x,t)\,dW, \tag{63}$$

where $f(x,t)$ is the drift coefficient (deterministic part), $g(x,t)$ is the diffusion coefficient (noise amplitude), and dW is the increment of a standard Wiener process satisfying $\mathbb{E}[dW] = 0$ and $\mathbb{E}[(dW)^2] = dt$. The Wiener process—also called Brownian motion—is a continuous-time stochastic process with independent, normally distributed increments: $W(t) - W(s) \sim \mathcal{N}(0, t-s)$ for $t > s$.

Because dW is not a conventional differential (the Wiener process is nowhere differentiable), SDEs require careful interpretation. The two most common interpretations are the Itô and Stratonovich conventions, which differ in how the integrand is evaluated during stochastic integration. In physics, the Itô convention is standard for thermal noise because it preserves the martingale property and leads to the fluctuation–dissipation theorem in a natural way. Under Itô calculus, the chain rule acquires an extra term:

$$dF(x) = F'(x)\,dx + \frac{1}{2}F''(x)g^2(x,t)\,dt, \tag{64}$$

which has no counterpart in ordinary calculus and is essential for correctly deriving statistical properties of stochastic trajectories.

2. *From Newtonian Langevin dynamics to the overdamped SDE*

A minimal mechanical model for a colloidal particle in a fluid harmonic trap combines viscous drag, a linear restoring force, and a fluctuating thermal force:

$$m\dot{v} = -\gamma v - kx + \xi(t), \tag{65}$$

where γ is the friction coefficient and k is the effective spring constant. In equilibrium, the fluctuation–dissipation theorem relates the noise correlation to dissipation; for overdamped motion one uses the Einstein relation

$$D = \frac{k_B T}{\gamma}, \tag{66}$$

which fixes the strength of diffusion once temperature T and drag γ are known.

When inertia relaxes quickly compared with the observation time step ($m/\gamma \ll \Delta t$), the acceleration term $m\dot{v}$ is negligible. Setting $\dot{v} \approx 0$ yields $\gamma v \approx -kx + \xi$, and since $v = \dot{x}$ in one dimension,

$$\gamma\,dx = -kx\,dt + \xi(t)\,dt. \tag{67}$$

Dividing by γ defines the confinement rate already quoted in class,

$$\lambda \equiv \frac{k}{\gamma}, \qquad dx = -\lambda x\,dt + \frac{\xi(t)}{\gamma}\,dt. \tag{68}$$

The fluctuating term must be modeled as Gaussian white noise whose amplitude is chosen so that *free* diffusion ($k = 0$, hence $\lambda = 0$) reproduces Fickian spreading with diffusion coefficient D. With the Itô convention used throughout these notes, this fixes the multiplicative factor $\sqrt{2D}$ in front of dW:

$$dx = -\lambda x\,dt + \sqrt{2D}\,dW, \tag{69}$$

$$dx = \sqrt{2D}\,dW \quad (\lambda = 0). \tag{70}$$

Here dW is the increment of a standard Wiener process: $\mathbb{E}[dW] = 0$, $\mathbb{E}[(dW)^2] = dt$, and non-overlapping increments are independent. Isotropic motion in d Cartesian dimensions is obtained by independent noises,

$$d\vec{r} = -\lambda\vec{r}\,dt + \sqrt{2D}\,d\vec{W}. \tag{71}$$

3. *Consistency check: mean-square displacement sets $\sqrt{2D}$*

For $\lambda = 0$, integrate $dx = \sqrt{2D}\,dW$ from 0 to τ starting at $x(0) = 0$:

$$x(\tau) = \sqrt{2D}\,W(\tau), \qquad W(\tau) \sim \mathcal{N}(0,\tau). \tag{72}$$

Therefore $\mathrm{MSD}(\tau) \equiv \langle x(\tau)^2\rangle = 2D\tau$ in one dimension, matching the Einstein definition of D. Over a discrete step Δt, the dynamical increment $\Delta x^{\mathrm{true}} \equiv x^{\mathrm{true}}(t+\Delta t) - x^{\mathrm{true}}(t)$ has conditional variance (drift treated as deterministic given x)

$$\mathrm{Var}(\Delta x^{\mathrm{true}} \mid x) = \mathbb{E}\Big[\big(\sqrt{2D}\,\Delta W\big)^2\Big] = 2D\Delta t, \tag{73}$$

which is the $\Delta t \ll \tau_{\mathrm{relax}}$ shortcut used in the likelihood below.

4. Euler–Maruyama scheme as a short-time Itô integral

Integrating the SDE over $[t_n, t_{n+1}]$ with $\Delta t = t_{n+1} - t_n$ gives

$$x_{n+1} - x_n = -\lambda \int_{t_n}^{t_{n+1}} x(t)\, dt + \sqrt{2D} \int_{t_n}^{t_{n+1}} dW. \tag{74}$$

Approximating $x(t) \approx x_n$ on the step (explicit Euler–Maruyama) and using $\int_{t_n}^{t_{n+1}} dW \overset{d}{=} \sqrt{\Delta t}\,\eta_n$ with $\eta_n \sim \mathcal{N}(0,1)$ yields the discrete update used in simulations:

$$x_{n+1} = x_n - \lambda x_n \Delta t + \sqrt{2D\Delta t}\,\eta_n. \tag{75}$$

For isotropic 2D motion with independent axis noises, each coordinate satisfies the same update and $\mathrm{MSD}(\tau) = \langle (\Delta x)^2 + (\Delta y)^2 \rangle = 4D\tau$.

5. Likelihood for noisy measured increments

Recorded positions are not x^{true} but

$$x_i^{\text{obs}} = x_i^{\text{true}} + \epsilon_i, \qquad \epsilon_i \sim \mathcal{N}(0, \sigma_{\text{meas}}^2), \tag{76}$$

with ϵ_i independent of ϵ_j for $i \neq j$. Define the observed increment

$$\Delta x_i^{\text{obs}} \equiv x_{i+1}^{\text{obs}} - x_i^{\text{obs}} = \underbrace{\left(x_{i+1}^{\text{true}} - x_i^{\text{true}}\right)}_{\Delta x_i^{\text{true}}} + \epsilon_{i+1} - \epsilon_i. \tag{77}$$

Conditional on x_i^{true}, the dynamical piece has mean $\mathbb{E}[\Delta x_i^{\text{true}}] \approx -\lambda x_i^{\text{true}} \Delta t$ for small Δt (the same substitution $x_i^{\text{obs}} \approx x_i^{\text{true}}$ is often made when forming the practical likelihood). The variance combines independent contributions:

$$\mathrm{Var}(\Delta x_i^{\text{obs}}) = \mathrm{Var}(\Delta x_i^{\text{true}}) + \mathrm{Var}(\epsilon_{i+1}) + \mathrm{Var}(\epsilon_i) = 2D\Delta t + 2\sigma_{\text{meas}}^2. \tag{78}$$

Hence the increment model asserted in class,

$$\Delta x_i^{\text{obs}} \sim \mathcal{N}\left(-\lambda x_i \Delta t,\ 2D\Delta t + 2\sigma_{\text{meas}}^2\right), \tag{79}$$

follows from (i) Itô diffusion variance $2D\Delta t$ and (ii) differencing two noisy measurements. Summing $\log p(\Delta x_i^{\text{obs}} \mid \theta)$ over steps yields the likelihood used for MCMC over $\vec{\theta} = (D, \lambda, \sigma_{\text{meas}})$.

- D: diffusion coefficient linked to drag by $D = k_B T/\gamma$.
- $\lambda = k/\gamma$: restoring strength or confinement rate.
- k, γ: harmonic stiffness and friction from $m\dot{v} = -\gamma v - kx + \xi$.
- dW: Wiener increment (Gaussian white noise).
- σ_{meas}: RMS additive position-measurement error.

By defining this log-likelihood explicitly, Bayesian MCMC explores the joint posterior of $\vec{\theta}$. A central difficulty is strong posterior correlation between D and σ_{meas} because both widen observed increments; complementary experiments or tighter priors are usually needed to break the degeneracy.

Once the parameters are extracted from the posterior, they unlock fundamental physical characteristics of the medium. The inferred diffusion coefficient D relates to the macroscopic friction γ via the Einstein relation ($D = k_B T/\gamma$). Assuming a spherical particle in a viscous fluid subject to Stokes drag ($\gamma = 6\pi\eta R$), one can indirectly estimate the particle radius R or the fluid viscosity η. Similarly, the estimated restoring strength $\lambda = k/\gamma$ yields the effective spring constant $k = \lambda k_B T/D$. This paradigm demonstrates how Bayesian MCMC seamlessly bridges noisy empirical measurements with foundational thermodynamic and mechanical properties.

F. Application: Heavy Quarks in the Nuclear Medium

The Bayesian SDE framework elegantly extends to cutting-edge research, such as the diffusion of heavy quarks (charm and bottom) propagating through a hot, dense Quark-Gluon Plasma (QGP) created in ultra-relativistic heavy-ion collisions. Analyzing the interaction between heavy quarks and the QGP medium relies on extracting the spatial diffusion coefficient D_s.

Due to complex non-perturbative QCD dynamics, theorists often parameterize the dimensionless spatial diffusion coefficient $D_s(2\pi T)$ as a phenomenological function of both the local medium temperature T (relative to the critical phase transition temperature T_c) and the heavy quark energy E. A representative empirical ansatz utilized in modern Bayesian analyses is formulated as:

$$D_s(2\pi T) = \frac{A\left(1 + B \cdot \frac{T}{T_c}\right)}{1 + C \cdot \log(E)} \tag{80}$$

where A, B, and C act as the unknown calibration parameters controlling the baseline diffusion, the temperature dependence, and the energy momentum-loss scaling, respectively. By comparing relativistic hydrodynamic simulation outputs (such as the nuclear modification factor R_{AA} and elliptic flow v_2) against experimental particle accelerator data, Bayesian MCMC traverses this multi-parameter space to pinpoint the exact (A, B, C) manifold. Consequently, this extracts the true diffusion properties of the QGP directly from complex, stochastic experimental spectra.

IX. NEURAL NETWORKS AND MACHINE LEARNING

The statistical principles driving curve fitting and classification extend transparently into modern neural network architectures. We begin with the simplest building block—the perceptron—and progressively develop the theory of multilayer networks, training algorithms, and associative memory models.

A. Logistic Regression as a Probabilistic Classifier

Before introducing the full neural-network architecture, it is instructive to consider logistic regression, which bridges the gap between linear regression and classification. In binary classification, rather than predicting a continuous target, we wish to estimate the probability $P(y = 1|\vec{x})$ that an input $\vec{x}$ belongs to class 1. A linear model $\vec{w} \cdot \vec{x} + b$ is unbounded, so we pass it through the sigmoid function to obtain a valid probability:

$$P(y = 1|\vec{x}) = \sigma(\vec{w} \cdot \vec{x} + b) = \frac{1}{1 + e^{-(\vec{w} \cdot \vec{x} + b)}}. \tag{81}$$

The decision boundary $\vec{w} \cdot \vec{x} + b = 0$ remains linear, but the output is now a calibrated probability. Training proceeds by maximizing the Bernoulli likelihood, or equivalently minimizing the binary cross-entropy loss:

$$\mathcal{L} = -\frac{1}{N} \sum_{i=1}^{N} \big[y_i \log \hat{y}_i + (1 - y_i) \log(1 - \hat{y}_i) \big], \tag{82}$$

which is precisely the loss function used in the neural-network classifier of Section X. Gradient descent (Section 3.5) provides the optimization algorithm. Logistic regression thus introduces, in the simplest setting, the three ingredients that recur throughout deep learning: a linear decision boundary, a sigmoid (or softmax) output nonlinearity, and a likelihood-based loss optimized by gradient descent. Stacking such units with hidden layers yields the multi-layer perceptron.

B. Perceptron and Logic Gates

The foundational building block of neural networks is the **perceptron**, a mathematical model of a biological neuron. A perceptron receives multiple input signals $\{x_1, x_2, \ldots, x_N\}$, each multiplied by a corresponding weight $\{w_1, w_2, \ldots, w_N\}$, and produces a binary output based on whether the weighted sum exceeds a threshold θ:

$$z = \sum_{i=1}^{N} w_i x_i + b, \quad b = -\theta \tag{83}$$

$$y = \begin{cases} 0, & z \le 0 \\ 1, & z > 0 \end{cases} = h(z) \tag{84}$$

Here, $h(\cdot)$ is the Heaviside step function and b is the bias term. The perceptron implements a linear decision boundary in the input space, making it a *linear classifier*.

The AND gate is linearly separable: with weights $w_1 = w_2 = 0.5$ and threshold $\theta = 0.7$, the perceptron correctly classifies all four input combinations since $0.5 \times 1 + 0.5 \times 1 = 1.0 > 0.7$ yields $y = 1$, while any input containing a zero produces a sum ≤ 0.5, yielding $y = 0$. Similarly, the OR gate is realized with a lower threshold $\theta = 0.2$.

The XOR gate exposes the fundamental limitation of a single-layer perceptron. No single linear boundary can separate the $(0,0), (1,1)$ class from the $(0,1), (1,0)$ class. The solution requires a **multi-layer perceptron (MLP)** with a hidden layer:

$$y = \mathrm{AND}(\mathrm{NAND}(x_1, x_2), \mathrm{OR}(x_1, x_2)) \tag{85}$$

This demonstrates that multi-layer architectures can represent any Boolean function [8].

C. Multi-Layer Neural Network Architecture

A multi-layer neural network is constructed by stacking layers of affine transformations followed by non-linear activation functions:

$$\mathbf{z}^{(l)} = W^{(l)} \mathbf{a}^{(l-1)} + \mathbf{b}^{(l)}, \quad \mathbf{a}^{(l)} = \sigma^{(l)}(\mathbf{z}^{(l)}) \tag{86}$$

The choice of activation function critically determines the network's expressive power. The **sigmoid function** maps $\mathbb{R} \to (0,1)$ with derivative $\sigma'(z) = \sigma(z)(1 - \sigma(z))$. The **ReLU** function, $\mathrm{ReLU}(z) = \max(0, z)$, avoids the vanishing gradient problem. For multi-class classification, the output layer applies the **softmax** function:

$$y_c = \frac{e^{z_c}}{\sum_{k=1}^{C} e^{z_k}}, \quad \sum_{c=1}^{C} y_c = 1 \tag{87}$$

The complete forward pass follows: Input $\xrightarrow{\text{Linear}}$ ReLU $\xrightarrow{\text{Linear}}$ ReLU $\xrightarrow{\text{Linear}}$ Softmax $\to$ Probabilities.

Beyond feedforward networks, the **Hopfield network** is a recurrent architecture that serves as a model of associative memory. Binary patterns $\{\boldsymbol{\xi}^{\mu}\}$ are stored via Hebbian learning:

$$W_{ij} = \frac{1}{N} \sum_{\mu} \xi_i^{\mu} \xi_j^{\mu}, \quad W_{ii} = 0 \tag{88}$$

The network evolves by updating neuron states $s_i^{\text{new}} = \mathrm{sign}(\sum_j W_{ij} s_j)$, minimizing the energy function:

$$E = -\frac{1}{2} \sum_{i,j} W_{ij} s_i s_j \tag{89}$$

Under asynchronous updates, the energy decreases monotonically, converging to the stored pattern closest to a corrupted input [9].

D. Error Backpropagation

Training a neural network requires computing the gradient of the loss function L with respect to every parameter. Backpropagation applies the chain rule systematically from the output layer backward. For a composite function $f(x) = f_3(f_2(f_1(x)))$:

$$\frac{df}{dx} = \frac{df_3}{df_2} \cdot \frac{df_2}{df_1} \cdot \frac{df_1}{dx} \tag{90}$$

Through the computational graph perspective: **addition nodes** distribute the upstream gradient equally, while **multiplication nodes** multiply it by the other input's value. For the affine layer $\mathbf{z} = W\mathbf{a} + \mathbf{b}$:

$$\frac{\partial L}{\partial W} = \frac{\partial L}{\partial \mathbf{z}} \mathbf{a}^T, \quad \frac{\partial L}{\partial \mathbf{a}} = W^T \frac{\partial L}{\partial \mathbf{z}} \tag{91}$$

For element-wise activation: $\frac{\partial L}{\partial \mathbf{z}} = \frac{\partial L}{\partial \mathbf{a}} \odot \sigma'(\mathbf{z})$.

E. Optimization Algorithms and Practical Training

Standard gradient descent suffers from vanishing gradients near minima, sensitivity to initial conditions, and uniform learning rates. The **Adam optimizer** addresses these limitations by maintaining exponential moving averages of the gradient (first moment) and squared gradient (second moment):

$$m_t = \beta_1 m_{t-1} + (1 - \beta_1) g_t, \quad v_t = \beta_2 v_{t-1} + (1 - \beta_2) g_t^2 \tag{92}$$

Bias correction yields $\hat{m}_t = m_t / (1 - \beta_1^t)$ and $\hat{v}_t = v_t / (1 - \beta_2^t)$, giving the final update:

$$\theta_{t+1} = \theta_t - \frac{\eta}{\sqrt{\hat{v}_t} + \epsilon} \hat{m}_t \tag{93}$$

Parameters with consistently large gradients receive smaller effective learning rates, while parameters with small gradients receive relatively larger updates [10].

As a practical application, the MNIST dataset consists of 60,000 training and 10,000 test images of handwritten digits (0–9), each a 28×28 grayscale image [11]. A fully-connected network with architecture $784 \to 128$ (ReLU) $\to 64$ (ReLU) $\to 10$ (Softmax) is trained with Adam using categorical cross-entropy loss, batch size 128, for 10 epochs.

X. APPLICATION: Z BOSON SIGNAL EXTRACTION WITH NEURAL NETWORKS

The statistical and neural-network methods developed in the preceding sections find a natural and powerful application in experimental particle physics. We now demonstrate how these tools are deployed in a realistic analysis: the extraction of the $Z \to \mu^+\mu^-$ signal from ATLAS open data using a neural-network classifier for background rejection. This application ties together several themes of the course—probability distributions, maximum-likelihood-inspired loss functions, gradient-based optimization, and multi-layer network architectures—in a single end-to-end physics analysis.

A. The Z Boson and Its Decay Signature

The Z boson, the neutral mediator of the weak interaction, has a mass of $M_Z \approx 91.2$ GeV and an extremely short lifetime of order 10^{-25} s. Because it decays before reaching any detector element, experimentalists must reconstruct it from its decay products. The leptonic decay channel $Z \to \mu^+\mu^-$ is particularly clean: the two muons traverse the entire ATLAS detector and are measured with high precision, yielding a narrow resonance peak in the di-muon invariant-mass spectrum atop a smooth continuum background from Drell-Yan production, $t\bar{t}$ pairs, and other Standard Model processes.

The invariant mass of the di-muon system is computed from the four-momenta of the two muons. Defining the four-vector of muon i as $p_i^\mu = (E_i, p_{x,i}, p_{y,i}, p_{z,i})$, or equivalently in terms of transverse momentum p_T, pseudorapidity η, and azimuthal angle ϕ, the di-muon invariant mass is

$$m_{\ell\ell} = \sqrt{(p_1^\mu + p_2^\mu)^2} = \sqrt{2\, p_{T,1}\, p_{T,2} \left[\cosh(\eta_1 - \eta_2) - \cos(\phi_1 - \phi_2)\right]}\,. \tag{94}$$

Events originating from $Z \to \mu^+\mu^-$ cluster around M_Z, while background events populate a broad continuum.

B. Dataset and Event Selection

The data are obtained from the ATLAS Open Data release at $\sqrt{s} = 13$ TeV, accessed programmatically through the `atlasopenmagic` package. Monte Carlo (MC) simulation samples of $Z \to \mu\mu$ production (run numbers 700323–700325) serve as the primary dataset for training and evaluation. The `uproot` library is used to read ROOT files and extract the relevant lepton-level branches: transverse momentum p_T, pseudorapidity η, azimuthal angle ϕ, energy E, charge, and lepton type.

Events are required to contain exactly two leptons with opposite electric charge ($q_1 + q_2 = 0$), which selects the characteristic $\mu^+\mu^-$ topology and rejects same-sign backgrounds. The charge-sum distribution, shown schematically in Fig. 1, confirms that the opposite-sign selection isolates a clean sample with a prominent Z peak.

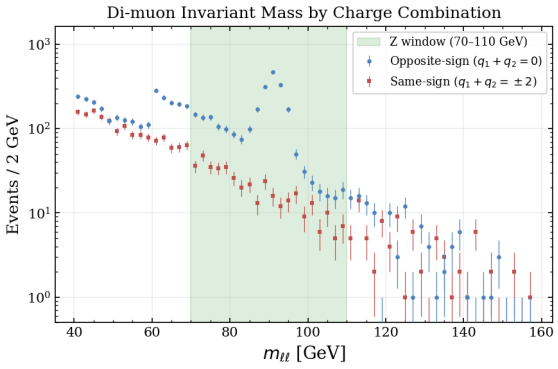

FIG. 1. Di-muon invariant mass spectrum for opposite-sign (charge sum = 0) and same-sign (charge sum = ±2) lepton pairs. A clear Z resonance peak near 91 GeV is visible only in the opposite-sign spectrum, validating the charge-based event selection.

C. Signal and Background Labelling

Since the MC samples do not carry explicit truth-level signal/background labels for the Z peak, we define labels using the di-muon invariant mass as a proxy:

- **Signal** (label = 1): $70 < m_{\ell\ell} < 110$ GeV — events in the Z mass window.
- **Background** (label = 0): $m_{\ell\ell} < 70$ GeV or $m_{\ell\ell} > 110$ GeV — sideband events dominated by the Drell-Yan continuum.

This mass-window labelling strategy is standard in resonance searches and exploits the fact that the Z boson produces a narrow Breit–Wigner resonance atop a smoothly falling background. The resulting label distribution is imbalanced (background dominates), which we address by equalizing the class sizes through random down-sampling of the majority class before training.

D. Feature Engineering

The neural network is trained on nine kinematic features that characterize the geometry and energy flow of the di-muon event without using the invariant mass itself (which would trivially separate signal from background):

- $p_T^{(1)}$, $p_T^{(2)}$: transverse momenta of the leading and subleading muon.
- $\eta^{(1)}$, $\eta^{(2)}$: pseudorapidities, encoding the polar-angle distribution.
- $\phi^{(1)}$, $\phi^{(2)}$: azimuthal angles.
- $\Delta\phi = |\phi_1 - \phi_2|$ (wrapped to $[0, \pi]$): azimuthal opening angle.
- $\Delta\eta = |\eta_1 - \eta_2|$: pseudorapidity separation.
- $\Delta R = \sqrt{(\Delta\eta)^2 + (\Delta\phi)^2}$: angular distance in the (η, ϕ) plane.

These features capture the characteristic angular correlations and p_T balance of $Z \to \mu^+\mu^-$ decays. Signal events tend to produce back-to-back muon pairs with balanced p_T, while background events exhibit broader, less correlated distributions. Figure 2 illustrates the discriminating power of selected features.

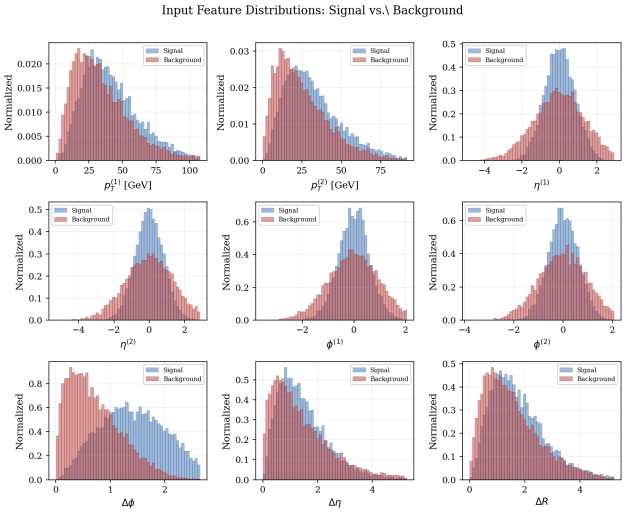

FIG. 2. Input feature distributions for signal (blue) and background (red) events after mass-window labelling. The nine kinematic variables—transverse momenta, pseudorapidities, azimuthal angles, and angular separations—exhibit distinct shapes for the two classes, providing the discrimination power exploited by the neural network.

E. Network Architecture and Training

The classifier is a fully-connected feed-forward neural network implemented in TensorFlow/Keras. The architecture consists of:

- **Input layer**: 9 neurons (one per feature).
- **Hidden layer 1**: 64 neurons with ReLU activation, followed by batch normalization and dropout with rate 0.3.
- **Hidden layer 2**: 32 neurons with ReLU activation, followed by batch normalization and dropout with rate 0.3.
- **Output layer**: 1 neuron with sigmoid activation, yielding $P(\text{signal}|\vec{x})$.

The sigmoid output is interpreted as the probability that a given event is signal-like. Binary cross-entropy is used as the loss function, which is equivalent to the negative log-likelihood for a Bernoulli model:

$$\mathcal{L} = -\frac{1}{N}\sum_{i=1}^{N}\left[y_i \log \hat{y}_i + (1-y_i)\log(1-\hat{y}_i)\right], \qquad (95)$$

where $y_i \in \{0,1\}$ is the true label and $\hat{y}_i$ is the network's predicted signal probability. The Adam optimizer is employed with early stopping on the validation AUC (patience = 10 epochs) to prevent overfitting.

Dropout, applied only during training, randomly sets 30% of the neuron outputs to zero at each update step. This regularization technique prevents the network from relying too heavily on specific neurons and encourages robust feature combinations, which is particularly important when the training sample is limited or contains statistical fluctuations.

The dataset is split into 70% training, 15% validation, and 15% test subsets, stratified to preserve the signal-to-background ratio. Feature standardization (zero mean, unit variance) is applied using statistics computed from the training set only, with the same transformation propagated to validation and test sets.

F. Model Performance and Evaluation

The trained network achieves strong discrimination between signal and background. The receiver operating characteristic (ROC) curve, shown in Fig. 3, plots the true-positive rate (signal efficiency) against the false-positive rate (background efficiency) as the decision threshold is varied. The area under the ROC curve (AUC) provides a threshold-independent measure of classification performance; an AUC close to 1.0 indicates excellent separation.

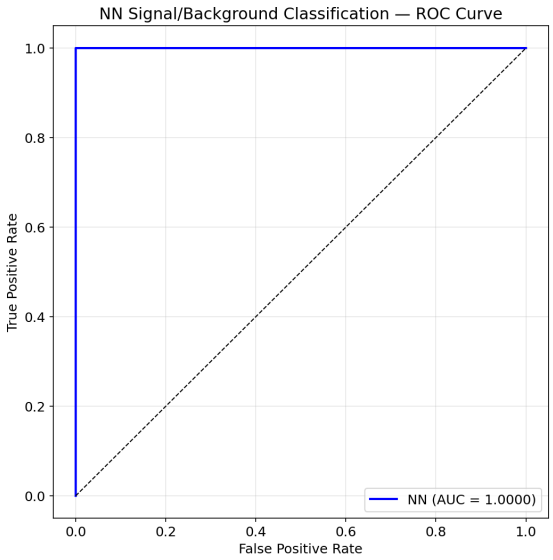

FIG. 3. Receiver operating characteristic (ROC) curve for the neural-network classifier evaluated on the test set. The curve lies well above the diagonal (random classifier), demonstrating strong signal–background discrimination. The area under the curve (AUC) quantifies the overall classification performance.

The distribution of NN output scores, shown in Fig. 4, exhibits the expected bimodal structure: signal events cluster near a score of 1 (high signal probability), while background events concentrate near 0. The separation between the two peaks confirms that the network has learned physically meaningful distinctions rather than memorizing statistical noise.

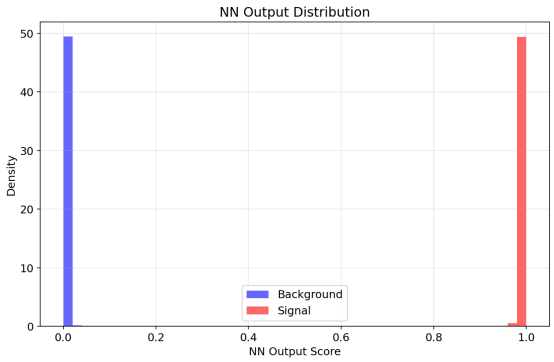

FIG. 4. Neural-network output score distributions for signal (blue) and background (red) events in the test set. Signal events cluster near a score of 1 (high signal probability), while background events concentrate near 0. The clear separation confirms that the network has learned physically meaningful discrimination.

G. Significance Optimization

In particle-physics analyses, the figure of merit is not classification accuracy but the *signal significance*, approximately $S/\sqrt{B}$, where S and B are the expected numbers of signal and background events passing a given NN-score cut. To find the optimal working point, we scan the NN-score threshold from 0.1 to 0.99 and compute $S/\sqrt{B}$ at each value. The threshold that maximizes significance typically differs from the default 0.5: a tighter cut reduces background more rapidly than it loses signal, thereby enhancing the discovery potential.

Figure 5 shows the significance as a function of the NN-score threshold, with the optimal cut indicated. Applying this optimized cut to the di-muon mass spectrum dramatically enhances the visibility of the Z peak, as shown in Fig. 6.

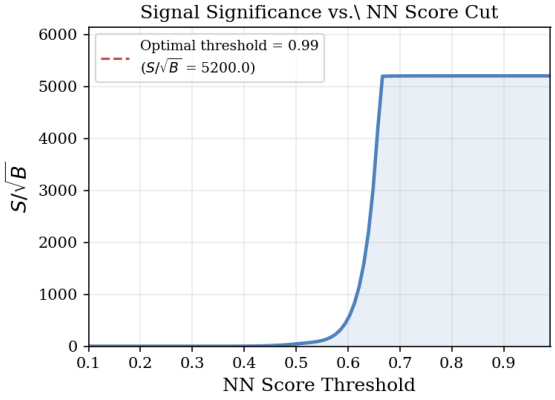

FIG. 5. Signal significance $S/\sqrt{B}$ as a function of the NN-score threshold. The optimal working point (vertical dashed line) maximizes significance rather than classification accuracy, yielding a tighter cut that preferentially suppresses background.

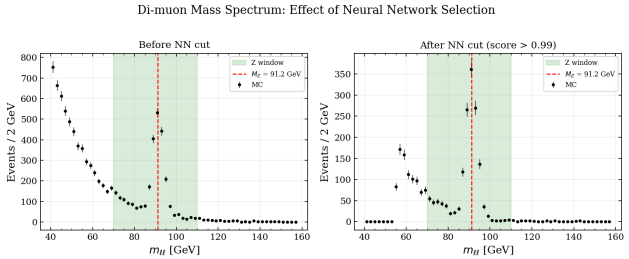

FIG. 6. Di-muon invariant mass spectrum before (left) and after (right) applying the optimized NN-score cut. The Z boson peak at 91.2 GeV becomes markedly more prominent after the cut, confirming that the neural network effectively rejects continuum background while retaining signal events.

H. Validation on Real Data

The ultimate test of the method is application to real collision data rather than MC simulation. Using ATLAS Open Data from the 2015–2016 runs (data periods D and E), we process the same event selection and feature computation pipeline, then apply the trained NN with the optimized score cut. The resulting mass spectrum, shown in Fig. 7, exhibits a clear Z peak consistent with the MC-based expectation, validating that the network generalizes from simulation to real detector data without significant bias.

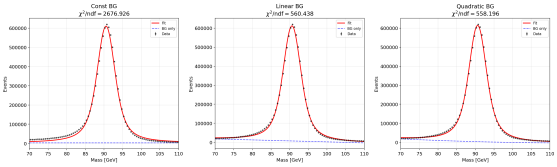

FIG. 7. Comparison of background model predictions with real ATLAS Open Data (data periods D–L) after the neural-network selection. The consistency between data and the MC-driven background model across the full mass range validates the generalization of the classifier from simulation to real collision data.

This application demonstrates the full chain of computational-physics methods: data acquisition and event selection, feature engineering grounded in physical principles, neural-network classification with proper regularization, loss-function minimization via gradient-based optimization, and figure-of-merit-driven working-point optimization. The Z boson, discovered in 1983 at CERN, continues to serve as a standard candle for detector calibration and analysis validation, and its efficient extraction from complex collision data exemplifies the power of modern machine-learning techniques in high-energy physics.

XI. CONCLUSION

The mathematical engines examined in this report—ranging from numerical integration and statistical data analysis to differential equations, root-finding, Fourier methods, Bayesian inference, MCMC, iterative unfolding, and neural network architectures—construct the structural paradigms of modern computational physics. The culminating application to Z boson signal extraction demonstrates how these tools operate synergistically in a realistic high-energy-physics analysis:

probability distributions model the signal and background populations, neural networks trained by gradient-based optimization learn nonlinear decision boundaries from engineered kinematic features, and figure-of-merit-driven optimization selects the working point that maximizes physical discovery potential. These formal tools enable scientists to interrogate computational limits, simulate high-energy boundary conditions, and infer underlying physical distributions from noisy or detector-smeared data where analytical mathematics entirely breaks down.

[1] P. R. Bevington and D. K. Robinson, *Data Reduction and Error Analysis for the Physical Sciences*, 3rd ed. (McGraw-Hill, New York, 2003).

[2] G. Cowan, *Statistical Data Analysis* (Oxford University Press, Oxford, 1998).

[3] F. James, *Statistical Methods in Experimental Physics*, 2nd ed. (World Scientific, Singapore, 2006).

[4] D. J. Griffiths and D. F. Schroeter, *Introduction to Quantum Mechanics*, 3rd ed. (Cambridge University Press, Cambridge, 2018).

[5] J. W. Cooley and J. W. Tukey, An algorithm for the machine calculation of complex fourier series, Mathematics of Computation **19**, 297 (1965).

[6] M. Newman, *Computational Physics* (CreateSpace Independent Publishing Platform, 2013).

[7] G. D'Agostini, A multidimensional unfolding method based on bayes' theorem, Nuclear Instruments and Methods in Physics Research Section A **362**, 487 (1995).

[8] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, Learning representations by back-propagating errors, Nature **323**, 533 (1986).

[9] J. J. Hopfield, Neural networks and physical systems with emergent collective computational abilities, Proceedings of the National Academy of Sciences **79**, 2554 (1982).

[10] D. P. Kingma and J. Ba, Adam: A method for stochastic optimization, arXiv preprint arXiv:1412.6980 (2014), proceedings of the 3rd International Conference on Learning Representations (ICLR).

[11] Y. LeCun, C. Cortes, and C. J. C. Burges, Mnist handwritten digit database, ATT Labs (2010).