Primer on Math
Probability, Statistics, Linear Algebra, and Calculus

Eslam Abdelaleem

Introduction

Mathematics is the language of the universe. We frequently encounter probability and statistics (to describe uncertainty and noise), linear algebra (to understand high-dimensional systems and transformations), and calculus (to analyze change, accumulation, and dynamics). This primer is designed as a friendly but rigorous guide to the essential tools you will need.

The goals are:

- Build intuition about concepts like probability distributions, random variables, and statistical laws.
- Understand the structure and manipulation of vectors and matrices, which form the backbone of modern data analysis and models of neural computation.
- Refresh the basics of calculus—derivatives, integrals, and series expansions—which are central to optimization, dynamics, and approximations.
- Collect a few "useful math facts" (factorials, combinations, Taylor expansions, Stirling's formula, etc.) that often appear in derivations.
- Provide small Python code snippets to illustrate computations, so you can experiment directly.

The primer assumes very little prior background. It is written in a way that emphasizes intuition alongside mathematical formality, with worked examples throughout. The intention is not to replace a full course in probability, linear algebra, or calculus, but to give you just enough foundation to feel comfortable as these ideas arise in the class.

1 What is Probability?

1.1 Experiments and Outcomes

A *random experiment* is a process whose outcome cannot be predicted with certainty in advance. Examples: flipping a coin, rolling a die, recording whether a neuron spikes in a time bin.

- The set of all possible outcomes is called the **sample space** S.
- Example: for a die, $S = \{1, 2, 3, 4, 5, 6\}$.
- An **event** is a subset of S.

1.2 Definition

The probability of an event A, written $P(A)$, is a number between 0 and 1 that satisfies:

(1) $P(A) \geq 0$ for any event A.

(2) $P(S) = 1$.

(3) If A and B are mutually exclusive, then $P(A \cup B) = P(A) + P(B)$.

2 Random Variables

A **random variable** X is a numerical outcome of a random experiment.

- Discrete random variable: takes countable values (e.g., number of heads in 10 coin flips).
- Continuous random variable: takes real values (e.g., time until next spike).

2.1 Expectation and Variance

The expectation (mean) of X:

$$\mathbb{E}[X] = \sum_x x\, P(X = x) \quad \text{(discrete)}, \qquad \mathbb{E}[X] = \int_{-\infty}^{\infty} x\, f(x)\, dx \quad \text{(continuous)}.$$

The variance measures spread:

$$\mathrm{Var}(X) = \mathbb{E}[(X - \mathbb{E}[X])^2] = \mathbb{E}[X^2] - (\mathbb{E}[X])^2.$$

Example. A fair die: $X \in \{1, \ldots, 6\}$, each with probability $1/6$.

$$\mathbb{E}[X] = \frac{1+2+3+4+5+6}{6} = 3.5, \quad \mathrm{Var}(X) = \frac{1}{6}\sum_{k=1}^{6}(k - 3.5)^2 = 35/12.$$

3 pmf, pdf, and cdf

Probability distributions can be described in different but related ways. These are the three most common functions you will encounter:

3.1 Probability Mass Function (pmf)

For a discrete random variable X (e.g., coin flips, number of spikes in a bin), the **pmf** gives the probability of each possible value:

$$p(x) = P(X = x).$$

- The pmf must satisfy $\sum_x p(x) = 1$.
- Example: if X is the result of a fair coin flip, $p(0) = 0.5$, $p(1) = 0.5$.

When useful: Whenever you deal with counts or categorical events (number of correct answers, number of spikes, success/failure).

```
from scipy.stats import binom
# Probability of exactly 3 spikes out of 10 trials, p=0.2
print(binom.pmf(3, n=10, p=0.2))
```

3.2 Probability Density Function (pdf)

For a continuous random variable X (e.g., reaction time, voltage, position), the probability of *any exact value* is essentially zero. Instead, we use a **pdf** $f(x)$ such that:

$$P(a \leq X \leq b) = \int_a^b f(x)\,dx.$$

- The pdf is not a probability itself; it can take values greater than 1, but the area under the curve over any interval gives a probability.
- The pdf integrates to 1: $\int_{-\infty}^{\infty} f(x)\,dx = 1$.

When useful: Continuous variables, where we care about ranges (e.g., "probability reaction time is between 250 and 300 ms").

```
from scipy.stats import norm
# Probability RT between 250 and 300 ms, mu=300, sigma=50
prob = norm.cdf(300, loc=300, scale=50) - norm.cdf(250, loc=300, scale=50)
print(prob)
```

3.3 Cumulative Distribution Function (cdf)

The **cdf** gives the probability that a random variable is less than or equal to x:

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

- For discrete variables, $F(x)$ is a step function.
- For continuous variables, $F(x)$ is smooth and non-decreasing.
- As $x \to -\infty$, $F(x) \to 0$; as $x \to +\infty$, $F(x) \to 1$.

Intuition: The cdf is like a running total of probability mass up to a point. It answers questions like:

- What fraction of trials had fewer than 3 spikes?
- What fraction of participants responded in under 400 ms?

Example (CDF of uniform). If $X \sim \text{Uniform}[0, 1]$, then

$$F(x) = \begin{cases} 0, & x < 0 \\ x, & 0 \leq x \leq 1 \\ 1, & x > 1. \end{cases}$$

```
import numpy as np
from scipy import stats

# Compute CDF values for Uniform[0,1]
x = np.linspace(-0.5, 1.5, 5)
cdf_vals = stats.uniform.cdf(x, loc=0, scale=1)
print(list(zip(x, cdf_vals)))
```

3.4 How They Work Together

- The pmf/pdf is like the "rate of change" of probability at a point.
- The cdf is the "accumulated probability" up to that point.
- For continuous variables: $f(x) = \frac{d}{dx}F(x)$.
- For discrete variables: $p(x) = F(x) - F(x-1)$.

Example intuition:

- pmf: "Probability of exactly 3 spikes."
- pdf: "Probability density around 300 ms."
- cdf: "Probability the neuron fired 3 or fewer spikes," or "Probability reaction time was less than 400 ms."

4 Common Distributions

4.1 Bernoulli Distribution

The Bernoulli distribution models a single *success/failure* event.

- Random variable $X \in \{0, 1\}$: 1 = success, 0 = failure.
- Probability of success: $P(X = 1) = p$, failure: $P(X = 0) = 1 - p$.
- Mean: $\mathbb{E}[X] = p$, Variance: $\text{Var}(X) = p(1 - p)$.

Example: A neuron either fires (1) or does not fire (0) in a time bin. If the probability of firing is $p = 0.2$, then $X \sim \text{Bernoulli}(0.2)$.

```
from scipy.stats import bernoulli

p = 0.2
sample = bernoulli.rvs(p, size=10)
print("Sample neuron spikes:", sample)
```

4.2 Binomial Distribution

The Binomial distribution counts the number of successes in n independent Bernoulli trials.

- $X \sim \text{Binomial}(n, p)$
- Mean: $\mathbb{E}[X] = np$, Variance: $\text{Var}(X) = np(1 - p)$

Example: Suppose we record a neuron across 10 consecutive time bins. Probability of spike per bin is $p = 0.2$. Let X be the number of spikes. Then $X \sim \text{Binomial}(10, 0.2)$.

```
from scipy.stats import binom

n, p = 10, 0.2
sample = binom.rvs(n, p, size=5)
print("Number of spikes in 10 bins:", sample)
```

Interpretation: You expect $np = 2$ spikes on average in 10 bins.

4.3 Poisson Distribution

The Poisson distribution models the number of events in a fixed interval, assuming events occur independently at a constant average rate λ.

- $X \sim \text{Poisson}(\lambda)$
- Mean = Variance = λ

Example: A neuron fires on average 5 times per second. Let X be the number of spikes in a second. Then $X \sim \text{Poisson}(5)$.

```
from scipy.stats import poisson

lambda_ = 5
sample = poisson.rvs(lambda_, size=5)
print("Spikes per second:", sample)
```

Interpretation: Most seconds you might see around 5 spikes, but sometimes more or less. This models naturally the random timing of spikes.

4.4 Normal (Gaussian) Distribution

The Normal distribution models continuous random variables that cluster around a mean μ with spread σ^2.

- PDF: $f(x) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-(x-\mu)^2/(2\sigma^2)}$
- Mean = μ, Variance = σ^2

Example: Reaction times in a simple task are often roughly Gaussian. Suppose the mean RT is 300 ms with SD 50 ms. Let X be the reaction time: $X \sim \mathcal{N}(300, 50^2)$.

```
from scipy.stats import norm

mu, sigma = 300, 50
sample = norm.rvs(loc=mu, scale=sigma, size=5)
print("Sample reaction times:", sample)
```

Interpretation: Most reaction times cluster around 300 ms, with decreasing probability for very fast or very slow responses.

4.5 Exponential Distribution

The Exponential distribution models waiting times between independent Poisson events.

- PDF: $f(x) = \lambda e^{-\lambda x}, \quad x \geq 0$
- Mean = $1/\lambda$, Variance = $1/\lambda^2$

Example: Suppose a neuron spikes on average once every 200 ms. The waiting time until the next spike $X \sim \text{Exponential}(\lambda = 5)$ (because $\lambda = 1/0.2$ s).

```
from scipy.stats import expon

lambda_ = 5
sample = expon.rvs(scale=1/lambda_, size=5)
print("Inter-spike intervals (s):", sample)
```

Interpretation: Short intervals are common, long intervals are less likely, but possible.

4.6 Summary Table

Distribution	Support	Mean	Interpretation
Bernoulli	0,1	p	Success/failure
Binomial	0..n	np	# of successes in n trials
Poisson	0,1,2...	λ	# events in fixed interval
Normal	$\mathbb{R}$	μ	Continuous variable around mean
Exponential	$x \geq 0$	$1/\lambda$	Waiting times

5 Moment Generating Function (MGF)

The **moment generating function (MGF)** of a random variable X is defined as

$$M_X(t) = \mathbb{E}[e^{tX}].$$

5.1 Why is it called "moment generating"?

- The derivatives of the MGF at $t = 0$ give the moments of X:

  $$M_X^{(n)}(0) = \mathbb{E}[X^n].$$

  For example:

  - $M_X'(0) = \mathbb{E}[X]$ (the mean),
  - $M_X''(0) = \mathbb{E}[X^2]$,
  - $\text{Var}(X) = M_X''(0) - (M_X'(0))^2$.

- The MGF (when it exists) uniquely determines the distribution.

- It often simplifies sums of independent random variables: if X and Y are independent, then

  $$M_{X+Y}(t) = M_X(t) \cdot M_Y(t).$$

5.2 Intuition and Use Cases

- Think of $M_X(t)$ as a "compressed encoding" of the distribution: by expanding it in a power series around $t = 0$, all the moments of X pop out.

- Helps prove theorems about sums of random variables (e.g., why the sum of independent Gaussians is Gaussian).

- Useful in probability theory, queueing models, and in neuroscience for spike-count models where moments (mean, variance) describe coding reliability.

5.3 Examples

5.3.1 Bernoulli

If $X \sim \text{Bernoulli}(p)$,

$$M_X(t) = (1-p) \cdot e^{0 \cdot t} + p \cdot e^{1 \cdot t} = 1 - p + pe^t.$$

From this:

- $M_X'(0) = p$ (mean),
- $M_X''(0) = p$ (since only the e^t term contributes),
- $\text{Var}(X) = p(1-p)$.

5.3.2 Poisson

If $X \sim \text{Poisson}(\lambda)$,

$$M_X(t) = \mathbb{E}[e^{tX}] = \sum_{k=0}^{\infty} e^{tk} \frac{e^{-\lambda}\lambda^k}{k!} = \exp\big(\lambda(e^t - 1)\big).$$

Moments:

- $M_X'(0) = \lambda$ (mean),
- $M_X''(0) = \lambda^2 + \lambda$,
- $\text{Var}(X) = \lambda$.

5.3.3 Normal

If $X \sim \mathcal{N}(\mu, \sigma^2)$,

$$M_X(t) = \exp\Big(\mu t + \tfrac{1}{2}\sigma^2 t^2\Big).$$

From this:

- $M_X'(0) = \mu$,
- $M_X''(0) = \mu^2 + \sigma^2$,
- $\text{Var}(X) = \sigma^2$.

6 Limit Theorems

6.1 Law of Large Numbers (LLN)

If $X_1, \ldots, X_n$ are i.i.d. with mean μ, then

$$\frac{1}{n}\sum_{i=1}^{n} X_i \to \mu \quad \text{as } n \to \infty.$$

Interpretation: sample averages converge to true expectation.

```
# LLN demo: sample averages of fair coin
trials = np.random.binomial(1, 0.5, size=10000)
running_avg = np.cumsum(trials) / np.arange(1, len(trials)+1)
print("Final average after 10000 trials:", running_avg[-1])
```

6.2 Central Limit Theorem (CLT)

Suppose $X_1, X_2, \ldots, X_n$ are independent and identically distributed (i.i.d.) random variables, each with mean μ and variance $\sigma^2 < \infty$. Let

$$\bar{X}_n = \frac{1}{n}\sum_{i=1}^{n} X_i$$

be the sample mean. Then the Central Limit Theorem states:

$$\frac{\sqrt{n}(\bar{X}_n - \mu)}{\sigma} \xrightarrow{d} \mathcal{N}(0, 1),$$

meaning that as $n \to \infty$, the standardized sample mean converges in distribution to a standard normal random variable.

What does this mean?

- Even if the original X_i are not Gaussian (they could be Bernoulli, Poisson, Uniform, etc.), the *average of many samples* will look Gaussian.
- The larger n is, the better the approximation.
- The theorem explains why Gaussian distributions appear so frequently in nature and cognition: many small, independent influences add up to produce approximately normal behavior.

Intuition

- Think of each X_i as one small "noise source" or factor.
- A single X_i might be skewed or discrete, but when we combine many of them, their irregularities "average out."
- The Gaussian distribution is the natural limit of this averaging process.

Examples

- **Coin flips.** Let $X_i \sim \text{Bernoulli}(0.5)$. Each X_i is just 0 or 1. But the average $\bar{X}_n$ (fraction of heads in n flips) becomes approximately Gaussian when n is large.
- **Reaction times.** Individual reaction times might follow skewed distributions (e.g., log-normal). But the average reaction time across many trials tends to look Gaussian.
- **Neural firing rates.** Spike counts in small bins may follow Poisson statistics. When averaged over many neurons or time windows, the result is close to Gaussian.

Why is it important?

- Justifies the widespread use of Gaussian models in statistics and neuroscience.
- Explains why standard errors and confidence intervals rely on the normal distribution, even if the data are not perfectly Gaussian.

Python Demo

```
import numpy as np
import matplotlib.pyplot as plt

# Example: CLT with Bernoulli(0.5)
n_trials = 10000   # number of experiments
n_samples = 50     # number of i.i.d. samples averaged

# Generate averages of Bernoulli trials
samples = np.random.binomial(1, 0.5, size=(n_trials, n_samples))
means = samples.mean(axis=1)

plt.hist(means, bins=30, density=True, alpha=0.6)
plt.title("Distribution of sample means (n=50 Bernoulli trials)")
plt.xlabel("Sample mean")
plt.ylabel("Density")
plt.show()
```

7 Linear Algebra Basics

Linear algebra provides the language for dealing with data in general. It's the basis of Machine Learning and computing in general. Here we recall the most useful concepts.

7.1 Vectors and Matrices

- A **vector** is an ordered list of numbers, often written as a column:

$$v = \begin{bmatrix} v_1 \\ v_2 \\ \vdots \\ v_n \end{bmatrix}.$$

- A **matrix** is a rectangular array of numbers, representing a linear transformation or dataset:

$$A = \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix}.$$

- Vectors can represent states (e.g., neural activity across n neurons). Matrices can represent transformations (e.g., weights, correlations, projections).

7.2 Dot Product and Norms

- The **dot product** of two vectors is

$$x \cdot y = \sum_{i=1}^{n} x_i y_i.$$

- It measures similarity: large if x, y point in the same direction, zero if orthogonal.
- The **norm** of a vector is its length:

$$\|x\| = \sqrt{x \cdot x}.$$

7.3 Linear Independence and Basis

- A set of vectors $\{v_1, \ldots, v_k\}$ is **linearly independent** if none of them can be written as a combination of the others.
- A **basis** is a minimal set of vectors that can be combined to express any vector in the space.
- Example: In $\mathbb{R}^2$, the standard basis is $\{(1, 0), (0, 1)\}$.
- Thinking in terms of basis is crucial for different tasks, if we figure out the correct basis for our problem, we can describe it with just a few basis vectors.

7.4 Matrix Operations

- **Matrix-vector product:** Av applies a linear transformation to v.
- **Matrix-matrix product:** AB represents applying B then A.
- **Transpose:** A^T flips rows and columns.
- **Inverse:** A^{-1} is the "undo" of A (if it exists).

7.5 Eigenvalues and Eigenvectors

- An **eigenvector** of A satisfies $Av = \lambda v$, where λ is the corresponding **eigenvalue**.
- Interpretation: eigenvectors point in directions preserved by A, eigenvalues tell how much they are stretched or shrunk.
- Example: In PCA (Principal Component Analysis), the directions of maximal variance are eigenvectors of the covariance matrix.

7.6 Orthogonality and Projections

- Vectors x, y are **orthogonal** if $x \cdot y = 0$.
- Orthogonal vectors represent independent directions (no overlap).
- The projection of x onto y is
$$\text{proj}_y(x) = \frac{x \cdot y}{\|y\|^2} y.$$

7.7 Why Useful?

- Describes different phenomena, such as neural population activity as vectors in high-dimensional space.
- Provides tools for compression and dimensionality reduction (PCA, SVD).
- Helps analyze transformations, such as how inputs are mapped to outputs in networks.

Python Demo

```
import numpy as np

# Define vectors
x = np.array([1, 2])
y = np.array([2, 1])

# Dot product and norms
print("dot(x,y) =", np.dot(x,y))
print("||x|| =", np.linalg.norm(x))

# Matrix and eigen-decomposition
A = np.array([[2, 1],
              [1, 2]])
vals, vecs = np.linalg.eig(A)
print("Eigenvalues:", vals)
print("Eigenvectors:\n", vecs)
```

8 Calculus Basics

Calculus provides the tools to describe change (derivatives) and accumulation (integrals). Both are everywhere in probability, statistics, and dynamical systems.

8.1 Derivatives

- The derivative of a function $f(x)$ measures its rate of change:

$$f'(x) = \lim_{h \to 0} \frac{f(x+h) - f(x)}{h}.$$

- Intuition: slope of the tangent line at x.
- Example: If $f(x) = x^2$, then $f'(x) = 2x$.
- Derivatives are useful for finding maxima/minima (optimization), sensitivity analysis, and describing dynamics.

8.2 Rules of Differentiation

- Linearity: $(af + bg)' = af' + bg'$.
- Product Rule: $(fg)' = f'g + fg'$.
- Chain Rule: If $h(x) = f(g(x))$, then $h'(x) = f'(g(x)) \cdot g'(x)$.
- Example (chain rule): If $f(x) = \sin(x^2)$, then $f'(x) = \cos(x^2) \cdot 2x$.

8.3 Partial Derivatives

- For multivariable functions $f(x, y)$, a partial derivative holds the other variable fixed:

$$\frac{\partial f}{\partial x}(x, y) = \lim_{h \to 0} \frac{f(x+h, y) - f(x, y)}{h}.$$

- Example: $f(x, y) = x^2 y \implies \frac{\partial f}{\partial x} = 2xy,\ \frac{\partial f}{\partial y} = x^2$.
- Useful in optimization (e.g., gradient descent in machine learning).

8.4 Integrals

- The integral accumulates area under a curve:

$$\int_a^b f(x)\, dx.$$

- Intuition: total accumulated value of f between a and b.
- Example: $\int_0^1 x^2 dx = \frac{1}{3}$.
- In probability: integrals compute expectations and probabilities for continuous random variables.

8.5 Fundamental Theorem of Calculus

- Links derivatives and integrals:

$$\frac{d}{dx} \int_a^x f(t)\, dt = f(x).$$

- Intuition: differentiation and integration are inverse operations.

8.6 Taylor Series

- Approximation of smooth functions near a point a:

$$f(x) \approx f(a) + f'(a)(x-a) + \frac{f''(a)}{2!}(x-a)^2 + \cdots$$

- Examples:

$$e^x = 1 + x + \tfrac{x^2}{2!} + \tfrac{x^3}{3!} + \ldots, \quad \ln(1+x) = x - \tfrac{x^2}{2} + \tfrac{x^3}{3} - \ldots$$

- Central in deriving approximations and intuition for functions.
- Useful for approximations and intuition in both probability (MGF expansions) and dynamical systems.
- Translates complicated functions into polynomials.

Python Demo

```
import sympy as sp

# Define a variable and function
x = sp.symbols('x')
f = sp.sin(x**2)

# Differentiate
dfdx = sp.diff(f, x)
print("f(x) =", f)
print("df/dx =", dfdx)

# Integrate
integral = sp.integrate(sp.exp(-x**2), (x, -sp.oo, sp.oo))
print(" exp(-x^2) dx from -inf to inf =", integral)

# Taylor series of exp(x) around 0 up to order 5
taylor_exp = sp.series(sp.exp(x), x, 0, 6)
print("Taylor expansion of exp(x):", taylor_exp)
```

9 Useful Math

In this section we collect some mathematical tools that often appear in probability, statistics, and physics in general. The goal is not just formulas, but also intuition about why they matter.

9.1 Factorials

For an integer $n \geq 1$,

$$n! = n \times (n-1) \times \cdots \times 1, \quad 0! := 1.$$

- Counts the number of ways to arrange n objects.
- Grows very quickly with n.

Example: $5! = 120$ different orders for 5 objects.

9.2 Combinations and Binomial Coefficients

The number of ways to choose k items out of n without order is

$$\binom{n}{k} = \frac{n!}{k!(n-k)!}.$$

- Appears in the Binomial distribution.
- Symmetry: $\binom{n}{k} = \binom{n}{n-k}$.

Example: $\binom{5}{2} = 10$ ways to pick 2 students from 5.

9.3 Stirling's Approximation

For large n, factorials are unwieldy. Stirling's formula provides:

$$n! \approx \sqrt{2\pi n}\left(\frac{n}{e}\right)^n.$$

A wider used form is after applying the logarithm on both sides and droping the $\mathcal{O}(\ln n)$ term:

$$\begin{aligned}\ln n! &= n\ln n - n + \mathcal{O}(\ln n)\\ &\approx n\ln n - n\end{aligned}$$

- Useful in approximating probabilities when n is large.

9.4 The Number e

Defined by the limit

$$e = \lim_{n\to\infty}\left(1+\tfrac{1}{n}\right)^n \approx 2.718.$$

- Arises naturally in growth, compounding interest, and Poisson processes.
- Basis of the natural logarithm.

9.5 Logarithms and Exponentials

- $\log_a(b)$ answers: "To what power must a be raised to get b?"
- In information theory, $\log_2(x)$ measures information in *bits*.
- Natural log $\ln(x)$ uses base e, which gives information in *nats*.

Key rules:

$$\ln(ab) = \ln a + \ln b, \quad \ln\left(\tfrac{a}{b}\right) = \ln a - \ln b.$$

9.6 Inequalities

Some inequalities are extremely useful in probability:

- **AM–GM inequality:** For $a, b \geq 0$, $\sqrt{ab} \leq (a+b)/2$.
- **Cauchy–Schwarz inequality:** $|\langle x, y\rangle| \leq \|x\|\|y\|$.
- These help in bounding expectations and variances.

9.7 Limits and Approximations

- For small x, $\ln(1+x) \approx x$.
- $(1+\frac{x}{n})^n \to e^x$ as $n \to \infty$.