QUANT INTERVIEW FOUNDATIONS

Ensure that you are interview ready!

1.1 Probability Fundamentals

Discrete Probability & Games

Most interviews start with coin, dice, or card games. The focus isn't just the answer, but the logic.

- **Combinatorics:** Permutations vs. Combinations (ordering matters vs. doesn't).
- **Expectation:** Calculating $E[X]$ for simple games.
- **Recursion:** Setting up equations for states (e.g., "expected flips to get HTH").

Conditional Probability

Understanding how new information updates probability.

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

Checklist:

- Bayes' Theorem applications (medical tests, spam filters).
- Law of Total Probability.

1.2 Distributions & Statistics

Key Distributions

Know the PDF, CDF, Mean, and Variance for these. You should know when to apply them.

- **Discrete:** Binomial, Poisson (arrival times), Geometric.
- **Continuous:** Uniform, Normal (Gaussian), Log-Normal (stock prices), Exponential.

Statistical Inference

Moving from data to models.

- **Linear Regression:** OLS assumptions, R^2, residual analysis.
- **CLT:** Central Limit Theorem and why it justifies using Normal distributions.
- **Estimators:** Difference between Bias and Variance.

2.1 Linear Algebra

Matrix Operations

Quant finance is built on high-dimensional data.

- **Eigenvalues/Vectors:** Understanding them as axes of variance (PCA).
- **Rank & Trace:** Properties and what they imply about solutions.
- **Positive Definite:** Why covariance matrices must be PSD.

Decompositions

Algorithms for solving linear systems efficiently.

- **Cholesky Decomposition:** Used for generating correlated random variables (Monte Carlo).
- **SVD/PCA:** Dimensionality reduction.
- **LU Decomposition:** Solving systems of linear equations.

2.2 Calculus

Multivariable Calculus

Required for optimization and risk sensitivities.

- **Partial Derivatives:** Calculating gradients.
- **Taylor Series:** Approximating functions (critical for understanding Convexity/Gamma).
- **Lagrange Multipliers:** Constrained optimization problems.

Integration & ODEs

- **Integration:** Used to find Expected Value over continuous distributions ($\int xf(x)dx$).
- **Differential Equations:** Basic separation of variables.
- **Concept:** Understanding that an option price satisfies a PDE (Black-Scholes).

3.1 Data Structures

Standard Containers

Know the Big O for Insert, Delete, and Lookup for each.

- **Arrays/Vectors:** Contiguous memory, $O(1)$ access.
- **Linked Lists:** $O(1)$ insert/delete if pointer is known.
- **Hash Maps:** $O(1)$ average lookup. Know how collisions are handled.

Specialized Structures

Often used in order book or streaming data problems.

- **Heaps (Priority Queues):** Efficiently accessing the min/max element ($O(1)$).
- **Binary Search Trees:** Balanced trees vs. unbalanced.
- **Stacks/Queues:** LIFO vs FIFO logic.

3.2 Algorithms

Sorting Searching

- **Sorting:** QuickSort vs. MergeSort. Why Quicksort is usually faster (cache locality) but worst-case $O(n^2)$.
- **Binary Search:** Implementing on sorted arrays ($O(\log n)$).

Problem Solving Patterns

- **Dynamic Programming:** Breaking problems into sub-problems (e.g., coin change problem).
- **Sliding Window:** Analyzing a specific subset of a stream.
- **Recursion/Backtracking:** Depth First Search (DFS).

4.1 C++ (Systems)

Memory Management

The defining feature of C++.

- **Stack vs. Heap:** Performance implications.
- **Pointers:** Raw pointers vs. Smart Pointers (‘std::unique$_p tr$‘, ‘$std :: shared_p tr$‘).
- **References:** When to pass by reference to avoid copying.

Modern C++ Features

- **Move Semantics:** ‘std::move‘ and r-values (optimizing large object transfers).
- **Templates:** Writing generic code.
- **STL:** Mastery of ‘std::vector‘ and ‘std::map‘.

4.2 Python (Data)

Vectorization

Why Python is slow but NumPy is fast.

- **Broadcasting:** Operations on arrays of different shapes.
- **Memory Views:** Avoiding copies during slicing.
- **Pandas:** Efficient data manipulation and cleaning.

Language Internals

- **GIL:** Global Interpreter Lock and its effect on multithreading.
- **Decorators/Generators:** Writing clean, pythonic code.
- **List Comprehensions:** Syntactic sugar vs loops.

5.1 Finance Basics

Arbitrage Parity

Identifying risk-free profit.

- **Put-Call Parity:** $C + Ke^{-rt} = P + S$. Be able to derive this via arbitrage arguments.
- **Forward Pricing:** Cost of Carry models.

Market Mechanics

How trading actually happens.

- **The Order Book:** Bids, Asks, Spreads, and Depth.
- **Types of Orders:** Limit vs. Market vs. Stop.
- **Liquidity:** What it means and how to measure it.

5.2 Options Theory

The Greeks (Sensitivities)

How the price changes when variables change.

- **Delta (Δ):** Directional risk.
- **Gamma (Γ):** Convexity (2nd derivative). Why it's good for long options.
- **Theta (Θ):** Time decay.

Black-Scholes Intuition

Don't just memorize the formula. Understand the assumptions.

- **Assumptions:** Geometric Brownian Motion, Log-normal prices, Constant Volatility.
- **Volatility:** The only unknown parameter.

Start with the basics.
Build from there.

Follow for more technical resources