**Example 1: Two-Stage Stochastic Optimization (Basic Example)**

import numpy as np

import cvxpy as cp

# Decision variable

x = cp.Variable()

# Uncertain demand scenarios

scenarios = np.array([10, 20, 30])

probabilities = np.array([0.3, 0.5, 0.2])

# Cost function (minimize expected cost)

cost = cp.sum(cp.multiply(probabilities, cp.pos(x - scenarios)))

objective = cp.Minimize(cost)

constraints = [x >= 0]

# Solve problem

problem = cp.Problem(objective, constraints)

problem.solve()

print(f"Optimal decision x: {x.value}")# Solve problem

problem = cp.Problem(objective, constraints)

problem.solve()

print(f"Optimal decision x: {x.value}")

**Example 2: Portfolio Optimization with Uncertainty**

# Set random seed for reproducibility

np.random.seed(42)

# Simulated returns for 4 assets over 100 scenarios

returns = np.random.normal(0.01, 0.02, (100, 4))

# Define decision variable for asset weights

weights = cp.Variable(4)

# Compute expected return

expected\_return = cp.sum(cp.multiply(cp.mean(returns, axis=0), weights))

# Compute risk using L2-norm as a variance proxy

risk = cp.norm(returns @ weights, 2)

# Define objective function (maximize return - risk penalty)

objective = cp.Maximize(expected\_return - 0.1 \* risk)

# Constraints: Weights sum to 1 and are non-negative (no short-selling)

constraints = [cp.sum(weights) == 1, weights >= 0]

# Solve optimization problem

problem = cp.Problem(objective, constraints)

problem.solve()

# Print optimal asset allocation

print(f"Optimal weights: {weights.value}")