Topic: Sampling and simulation methods

Concepts: Monte Carlo Simulations, Latin Hypercube Sampling, and Probability of Failure.

Note: *Start each problem with the starter code shared in the repository. Feel free to use scipy reference or any other source for syntax, but invest time in following the logic of the programs.* *https://docs.scipy.org/doc/scipy/reference/stats.html*

Important notes for submission:

Each time you generate a plot, save it with a filename like a8_prob1_fig1_yourInitials.png (assignment 8 → problem 1 → figure 1). Upload all your figures to the figs_results directory in your GitHub repo.

Generate a report using tables, figures, and your remarks. Upload this report as a PDF to your assignment-specific GitHub repository. This report will be graded along with your programs.

Before solving a problem, look at the provided starter python file named `assm8_student.py` and work on each task by uncommenting the corresponding function in the `__main__`. Understand the sequence of provided code and change it wherever instructed to do so;

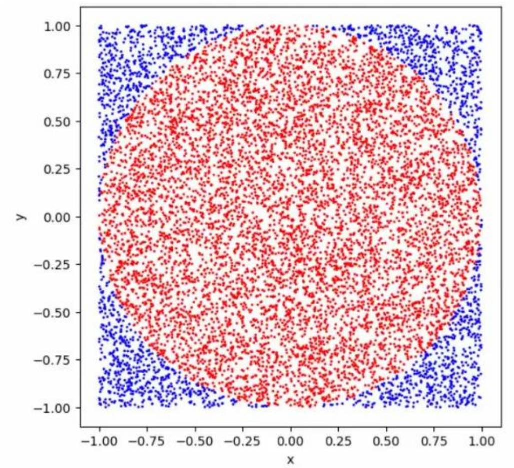
Background Concepts

Monte Carlo Simulation (MCS)

Estimator:

$\hat{\mu}_N = \frac{1}{N}\sum_{i=1}^{N} h(X_i)$, where X_i are iid samples and h is the model/quantity.

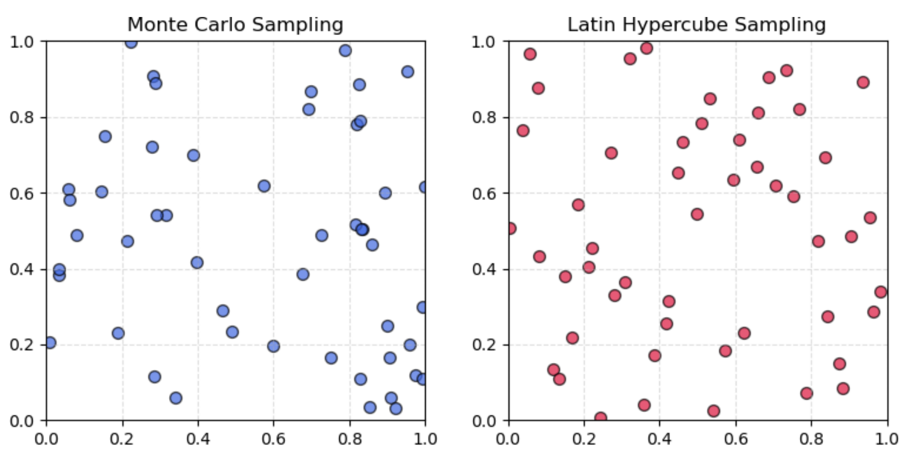
Description: Draw iid samples from input distributions; apply the model; average outputs to approximate expectations or probabilities. Convergence rate $\sim O(N^{-1/2})$.



Latin Hypercube Sampling (LHS)

Concept: stratify each marginal into N intervals; sample once per interval; combine via random permutations.

Description: LHS stratifies each input marginal to ensure uniform coverage, producing lower variance estimators than plain MCS for the same N in many problems.



Problem 1: Circle Area using Monte Carlo Simulations (MCS)

1. Setup:

Estimate the area of a circle numerically using Monte Carlo: sample uniform points in the unit square $[0,1]^2$ and count fraction inside the circle of radius 0.5.

2. Background Concepts:

See the previous section on MCS.

3. Objectives:

Implement simple MCS, observe estimator convergence, compute standard error and confidence interval for an estimated area, and visualize sample realizations to build intuition about stochastic approximation.

4. Step-by-step tasks:

1. Generate N uniform points in $[0,1]^2$.
2. Count points with $(x - 0.5)^2 + (y - 0.5)^2 \leq 0.5^2$. Estimate area = fraction × 1.
3. Compute Monte Carlo standard error $\sqrt{p(1-p)/N}$
4. Repeat for $N = 100$; $N = 1{,}000$; $N = 10{,}000$; plot area estimate vs N.
5. Plot scatter of points and color-code inside/outside circle.

<u>Submit (1) the updated programs, (2) generated figures and tables within the report.</u>

Problem 2: Beam Failure Probability using MCS

5. Setup:

For a simply supported beam under uniform load w(kN/m), midspan deflection is

$$\delta = \frac{5wL^4}{384EI}.$$

We will treat w, L, E, and I as uncertain inputs and model them as random variables (SI units). Use the limit-state

$$g(\mathbf{X}) = \delta_{\text{allow}} - \delta,$$

where the failure is said to occur, if $g < 0$ (i.e., $\delta > \delta_{\text{allow}}$). And $\mathbf{X} = [w, L, E, I]$ is the collection of all random variables.

Adopt the following distributions:

- $w \sim N(5.0, 0.5^2)$kN/m;
- $L \sim N(5.0, 0.05^2)$m;
- $E \sim N(200 \times 10^9, (10 \times 10^9)^2)$Pa;
- $I \sim N(8 \times 10^{-6}, (0.5 \times 10^{-6})^2)$m⁴.

Estimate $P_f = \Pr(\delta > \delta_{\text{allow}})$ using MCS.

6. Background Concepts:

See the previous section on MCS.

7. Objectives:

Use brute-force Monte Carlo to propagate input uncertainty to beam deflection, estimate probability of exceeding allowable deflection, compute estimator uncertainty, and *identify which inputs most influence failure risk*.

8. Step-by-step tasks:

1. Choose δ_{allow}(e.g., 0.02 m) and sample size N (use 100, 1000, and 10,000).
2. Draw independent samples for w, L, E, I; enforce positive draws (resample or use lognormal).
3. Compute δ for each sample; evaluate $g = \delta_{\text{allow}} - \delta$.
4. Estimate $P_f = N_f/N$ and standard error $\sqrt{P_f(1 - P_f)/N}$
5. Plot histogram of δ with δ_{allow} line; show convergence vs N.
6. Briefly interpret engineering significance and required sample budget.

Submit (1) the updated programs, (2) generated figures and tables within the report.

Problem 3: Beam Failure Probability using Latin Hypercube Sampling (LHS)

1. Setup:

Re-estimate the beam failure probability from Problem 2 using Latin Hypercube Sampling (LHS) to achieve better estimator precision for the same sample budget

2. Background Concepts:

See the previous section on LHS.

3. Objectives:

Demonstrate that LHS provides lower variance estimators than simple MCS by comparing p_f estimates and their uncertainty for equal N, using the same deflection limit-state.

4. Step-by-step tasks:

1. Implement LHS for four variables (stratify marginals into N strata, sample one per strata, permute).
2. Use same distributions and δ_{allow} as Problem 2; draw N samples with LHS.
3. Compute δ and estimate P_f and SE.
4. Compare LHS and MC results (means, SEs) for N = 200, 500, 1000; plot boxplots over multiple replicates.
5. Conclude on computational savings (e.g., N needed to reach target SE).

<u>Submit (1) the updated programs, (2) generated figures and tables within the report.</u>

Problem 4 (at home): Three-bar Truss FEA: probability of failure (MCS and LHS)

1. Setup:

A triangular 3-bar planar truss has nodes A(0,0), B(4,0), C(2,2). Elements: AC, BC (inclined), and AB (bottom chord). Node C carries a downward point load P. Inputs uncertain: P, E, A. Failure criterion: vertical displacement at C, δ_C, exceeds $\delta_{\text{allow}} = 0.02\text{m}$. A provided function analyze_truss(node_coords,elements,E_vec,A_vec,loads,supports) returns nodal displacements via linear-elastic stiffness assembly. Compute P_f using both Monte Carlo and LHS. Note: each FEA run is costlier than closed-form beam; sampling efficiency matters.

Adopt the following distributions:

- $P \sim N(50,\, 5^2)kN;$
- $E \sim N(200 \times 10^9, (10 \times 10^9)^2)Pa;$
- $A_i \sim N(A_{i0}, (0.1A_{i0})^2)\text{m}^2$ (for each member).

2. Background Concepts:

See the previous section on LHS/MCS.

3. Objectives:

Apply sampling strategies to a small FEA model to estimate system-level failure probability, compare MCS vs LHS efficiency, and observe how computational cost changes sampling strategy choice.

4. Step-by-step tasks:

1. Use provided analyze_truss (validate on baseline input).
2. Define uncertain inputs and supports; choose N (e.g., 500) and replicates for variance estimation.
3. Run MCS: sample inputs, call analyze_truss each iteration, record δ_C, count failures.
4. Run LHS: sample same dimensional inputs via LHS, compute δ_C, estimate P_f.
5. Compare estimates and their standard errors; plot histograms of δ_C.
6. Discuss runtime and scaling: estimate time per simulation and projected time to achieve given SE with MCS vs LHS.
7. (Optional) Report sensitivity of δ_C to P, E, A.

Submit (1) the updated programs, (2) generated figures and tables within the report.