# MTH208 End-Semester Examination

## Instructions

1. **Seating:** Sit according to the seating plan. Please arrive at the lab by **4:40 PM** to allow enough time to locate your assigned seat.
2. **Template & naming**
   - Download the template file **`es_template.R`** and the data files, if any, from the helloIITK exam page. These will be available five minutes before the exam begins.
   - Rename it to **`es_<ROLL>.R`**, where `<ROLL>` is your roll number (e.g., `es_230123.R`).
   - Work only in this renamed file.
3. **What to edit**
   - Follow the instructions for each question **exactly**.
   - **Only** edit the lines marked with dashes `-------` or comments `# your code here`.
   - **Do not** change function or object names required by the grader.
   - **Do not** add `rm(list = ls())`, `setwd()`, or absolute paths.
4. **Libraries**
   - You should need only `ggplot2` and Base R.
5. **I/O and paths**
   - Your script must run in a fresh R session with the working directory set to the folder containing your files.
6. **Submission**
   - Submit **only one file**: your renamed script **`es_<ROLL>.R`**.
   - Upload it on the **helloIITK** End Sem Exam page. Submissions by email or any other method will **not** be accepted.
7. **Grading run**
   - Your script **must run end-to-end** on the grader's machine with

     ```
     source("es_<ROLL>.R")
     ```

     without manual edits, assuming the provided data and template structure.
8. **Evaluation data:** Please avoid hard-coding values or assumptions beyond what's specified.

Consider the integral

$$I(f) = \int_0^1 f(x)dx$$

where, for simplicity, $f(x) \geq 0$ for all $x \in (0, 1)$. We study the estimation of $I(f)$ using three methods, while addressing numerical stability and evaluating performance.

**Method A:** Draw $N$ samples $x_1, \ldots, x_N \sim \text{Uniform}(0, 1)$ and estimate the integral as

$$I(f) \approx I_N^A(f) = \frac{1}{N} \sum_{i=1}^{N} f(x_i)$$

**Method B:** Draw $N$ samples $(x_1, y_1), \ldots, (x_N, y_N)$ where $x_i \sim \text{Uniform}(0, 1)$ and $y_i \sim \text{Uniform}(0, M)$ with

$$M \geq \max_{x \in [0,1]} f(x).$$

The integral is estimated as

$$I(f) \approx I_N^B(f) = \frac{|\{(x_i, y_i) : y_i \leq f(x_i)\}|}{N} M$$

where $|S|$ denotes the number of elements in the set $S$.

**Method C:** Estimate the integral as

$$I(f) \approx I_N^C(f) = \frac{1}{N-1} \left( \frac{1}{2} f(0) + \sum_{j=2}^{N-1} f\left( \frac{j-1}{N-1} \right) + \frac{1}{2} f(1) \right)$$

In this context, perform the following tasks by completing the code given in the template file.

## Tasks

1. <u>(1 point)</u> Compute the exact value of $I(f)$ when $f(x) = x \exp(-x^2)$.
2. Write an R code to estimate the integral using **Method A** when $f(x) = x \exp(-x^2)$.
   ***Note:*** <u>1 point</u> *for this task will be given based on the correct column output in task 6.*
3. Write an R code to estimate the integral using **Method B** when $f(x) = x \exp(-x^2)$.
   ***Note:*** <u>1 point</u> *for this task will be based on the correct column output in task 6.*
4. Write an R code to estimate the integral using **Method C** when $f(x) = x \exp(-x^2)$.
   ***Note:*** <u>1 point</u> *for this task will be based on the correct column output in task 6.*
5. For performance comparison, write an `R` function that computes **runtime** for a function in seconds (using `proc.time()`).
   ***Note:*** <u>1 point</u> *for this task will be based on the correct column output in task 6.*

6. (2+1+2+2+1+1+2+0+1=12 points) Run both methods for the following values of $N$:

$$N \in \{4^k : k = 4, 5, 6, 7, 8, 9\}$$

For each method and each value of $N$, using **50 independent repetitions** of the estimators,

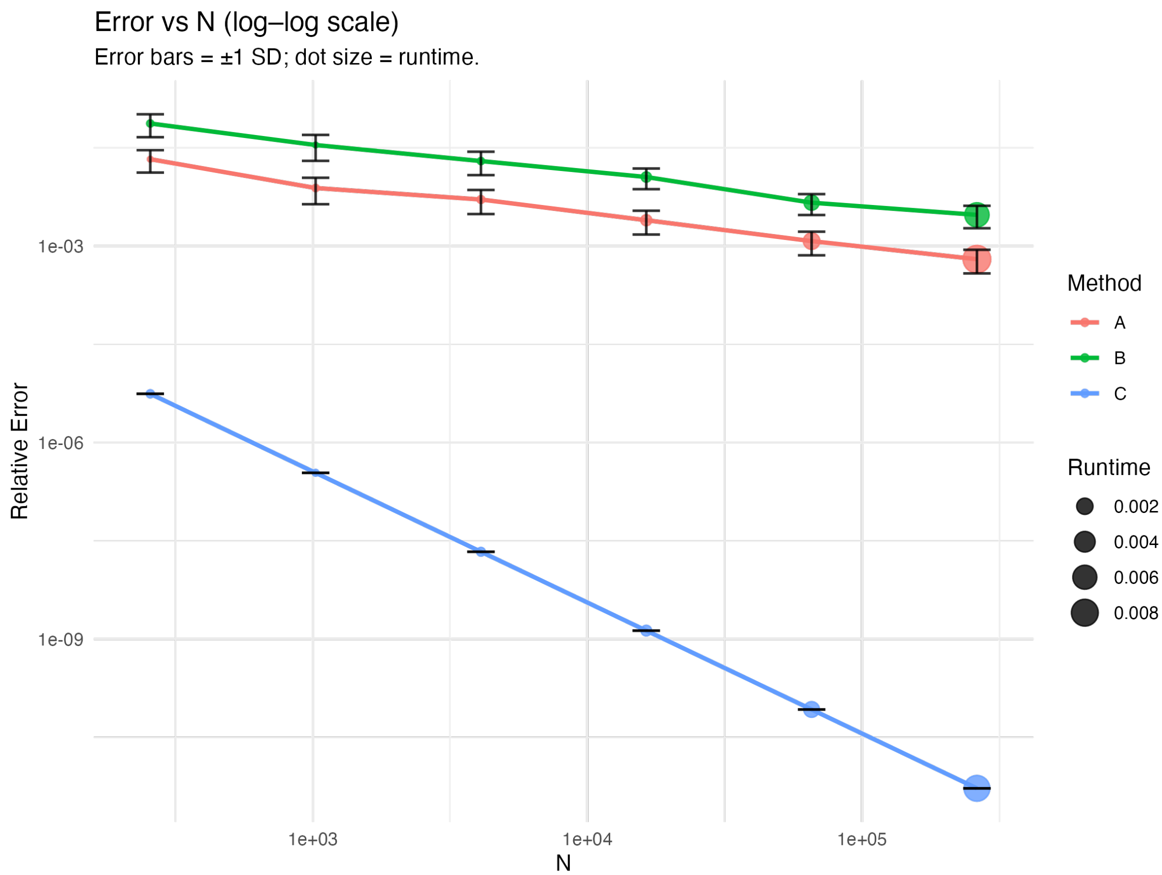
- Compute the mean error using methods A, B and C, that is, the mean of $|I_N^A(f) - I(f)|/|I(f)|$, $|I_N^B(f) - I(f)|/|I(f)|$ and $|I_N^C(f) - I(f)|/|I(f)|$, reported as ErrA, ErrB and ErrC respectively in the table below.
- Compute **variance** of the three estimators, that is, the variance of $I_N^A(f)$, $I_N^B(f)$ and $I_N^C(f)$, reported as VarA, VarB and VarC respectively in the table below.
- Compute the mean of the **runtime** obtained using the `runtime` function written in task 5 for the three estimators, reported as TimeA, TimeB, and TimeC respectively in the table below.

Create a table (an `R` `data.frame`) of the form

| N | ErrA | VarA | TimeA | ErrB | VarB | TimeB | ErrC | VarC | TimeC |
|---|---|---|---|---|---|---|---|---|---|
| 256 | ... | ... | ... | ... | ... | ... | ... | ... | ... |
| 1024 | ... | ... | ... | ... | ... | ... | ... | ... | ... |
| 4096 | ... | ... | ... | ... | ... | ... | ... | ... | ... |
| 16384 | ... | ... | ... | ... | ... | ... | ... | ... | ... |
| 65536 | ... | ... | ... | ... | ... | ... | ... | ... | ... |
| 262144 | ... | ... | ... | ... | ... | ... | ... | ... | ... |

and name it `comparison_table`.

7. (5 points) Using the data in `comparison_table`, create the ggplot called `performance_plot` that displays mean errors against N on the log-log scale along with the runtime (as size of point) and standard deviation (as errorbar – `?geom_errorbar`), closely resembling the example plot shown below:




8. <u>(1 point)</u> Compute the exact value of $I(f_\epsilon)$ when $f_\epsilon(x) = \log(\exp(50x) - \exp(50x - \epsilon))$.

9. <u>(1 point)</u> Write an R code to approximate the integral $I(f_\epsilon)$ using Method A when $f_\epsilon(x) = \log(\exp(50x) - \exp(50x - \epsilon))$.