Exercise – Understanding Monte Carlo integration using a logarithmic integral

1. Aim of the exercise

Monte Carlo (MC) integration is a technique for estimating integrals. We use this technique to estimate the logarithmic integral and prime counting function.

2. Theory

We do not provide a thorough treatment of MC theory here; this is covered in the dedicated exercise on MC integration theory. Instead, we illustrate the idea through an example.

One of the most important conjectures in mathematics is the Riemann Hypothesis. It has profound implications for the distribution of prime numbers, a phenomenon that, to this day, remains only partially understood. Nevertheless, the Prime Number Theorem provides a powerful asymptotic description of prime density, stating that

$$\pi(x) \sim \frac{x}{\log(x)} \quad \text{as} x \to \infty,$$

where $\pi(x)$ denotes the prime-counting function for $x \in \mathbb{R}$, defined as the number of primes less than or equal to x. For example,

$$\pi(3) = \#\{2,3\} = 2, \quad \text{and} \quad \pi(10) = 4,$$

where # denotes the cardinality of a set.

A more refined approximation is given by the offset logarithmic integral

$$Li(x) := \int_2^x \frac{1}{\log(y)} dy,$$

defined for all $x \in \mathbb{R}_{\geq 2}$. This function closely tracks the growth of $\pi(x)$ and reflects the heuristic that primes become increasingly sparse as x increases.

The Prime Number Theorem asserts that $\pi(x) \sim Li(x)$, meaning

$$\lim_{x \to \infty} \frac{\pi(x)}{Li(x)} = 1, \quad x \in \mathbb{R}_{\geq 2}.$$

In other words, $Li(x)$ and $\pi(x)$ are asymptotically equivalent. This reflects the heuristic that the probability of a large number x being prime is approximately $\frac{1}{\log(x)}$, making $Li(x)$ a remarkably accurate estimate for the number of primes up to and including x.

To estimate $Li(n)$, we consider a random variable $X \sim \text{Unif}(2, n)$, with probability density function

$$f_X(x) = \frac{1}{n-2}, \quad x \in [2, n].$$

This allows us to rewrite the logarithmic integral as an expectation. Observe that

$$\begin{aligned} Li(n) = \int_2^n \frac{1}{\log(y)}\, dy &= \int_2^n \frac{1}{\log(y)} \cdot \frac{n-2}{n-2}\, dy \\ &= \int_2^n \frac{n-2}{\log(y)} \cdot f_X(y)\, dy \\ &= \mathbb{E}\left[\frac{n-2}{\log(X)}\right]. \end{aligned}$$

Note that to compute the expected value of $\frac{n-2}{\log(X)}$, we use a rule from probability theory: if we have a function of a random variable, we can compute its expectation by integrating that function times the PDF of the original variable. This is called the law of the unconscious statistician. This law allows us to avoid computing the distribution of $\frac{n-2}{\log(X)}$ directly.

To estimate this expectation for a fixed input $n \geq 2$, we draw N independent samples $x_1, \ldots, x_N \sim \text{Unif}(2, n)$, and compute the MC estimator:

$$\frac{1}{N} \sum_{i=1}^{N} \frac{n-2}{\log(x_i)}.$$

Here, n determines the domain of integration and the scaling factor, while N controls the number of samples used in the approximation. This yields a numerical estimate of the logarithmic integral $Li(n)$ via standard MC integration.

3. Simulation setup

Create a vector of n values ranging from 100 to 10,000 in increments of 100. For each n, we estimate the logarithmic integral using MC integration and compute the exact number of primes less than or equal to n. This enables a direct comparison between the approximation and the true prime counts across a broad range of inputs.

```
%% 3. Simulation setup

% 3.1. Clear workspace and memory
clear;

% 3.2. Set number of random samples from Uniform(2, n)
N_samples = 1000;

% 3.3. Define evaluation points for prime counting and integration
N_values = 100:100:10000;
```

4. Monte Carlo integration: Estimating the logarithmic integral

We estimate $Li(n)$ using the following MATLAB code. First, we set the simulation parameters and define a sequence of evaluation points $n = 100, 200, \ldots, 10{,}000$. For each value of n, we sample `N_samples` points uniformly from the interval $[2, n]$. We then apply the transformation $\frac{1}{\log(x)}$ to each sample and compute the empirical mean. Multiplying this mean by the interval length $(n-2)$ yields a Monte Carlo estimate of the logarithmic integral $Li(n)$. In parallel, we compute the exact number of primes less than or equal to each n using MATLAB's built-in `primes` function. This allows us to compare the Monte Carlo approximation against the true prime counts across a wide range of inputs.

```
%% 4. Monte Carlo integration: Estimating the logarithmic integral

% 4.1. Preallocate vector for Monte Carlo estimates
MC_log_integral = NaN(size(N_values));

% 4.2. Preallocate vector for actual prime counts
```

```
prime_counts = NaN(size(N_values));

% 4.3. Loop over each n value and perform Monte Carlo estimation
for i = 1:length(N_values)
    % Current value of n
    N = N_values(i);
    % Draw random samples uniformly from [2,N]
    uniform_samples = random('Uniform',2,N,[N_samples 1]);
    % Apply transformation: f(x) = 1/log(x)
    log_transformation = 1./log(uniform_samples);
    % Estimate integral using sample mean
    mean_value = mean(log_transformation);
    MC_log_integral(i) = (N-2)*mean_value;
    % Compute actual number of primes up to N
    prime_counts(i) = length(primes(N));
end
```

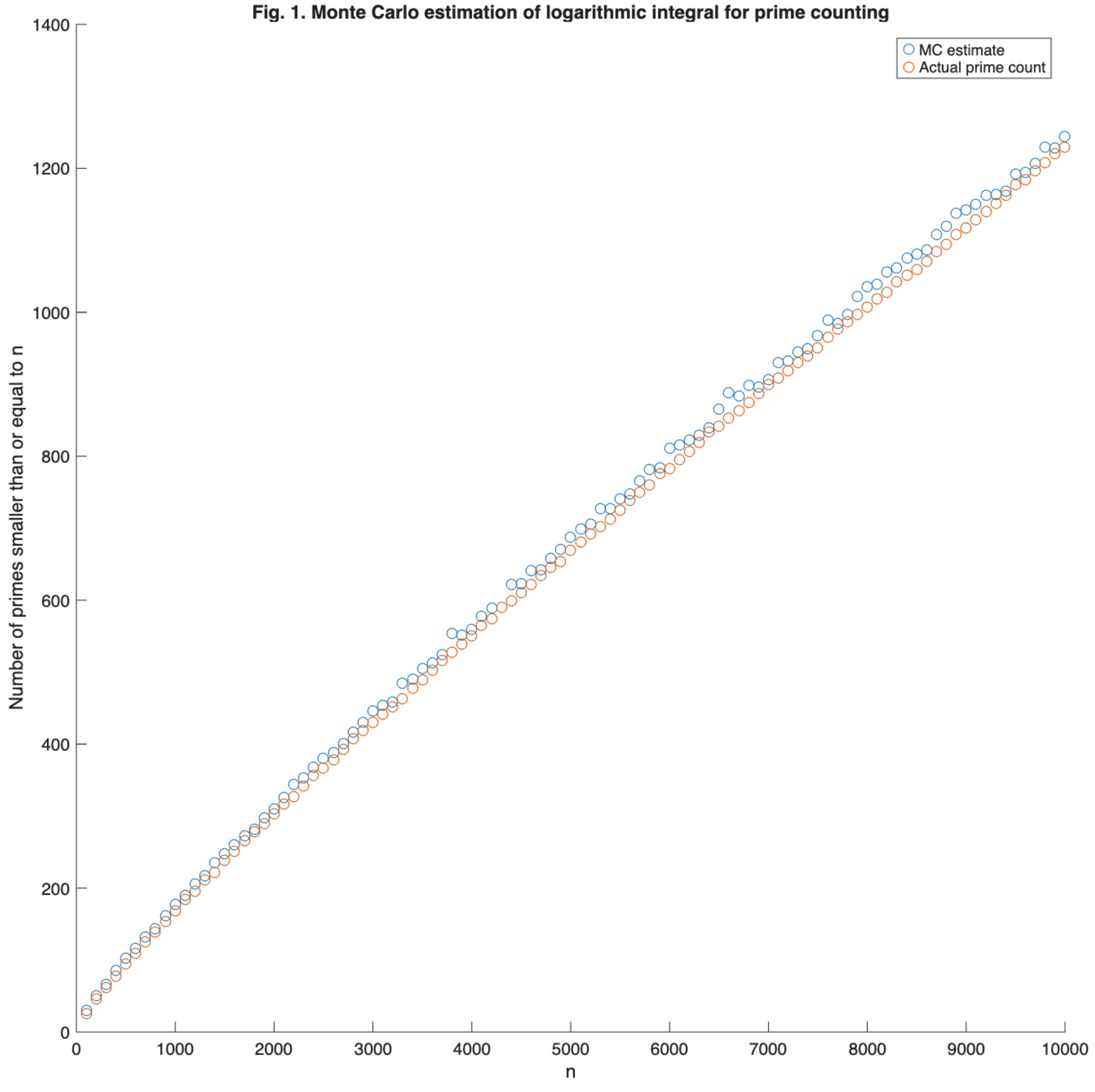
5. Visualization: Compare integral estimates with true prime counts

Here we compare the MC estimates of the logarithmic integral and the actual prime counts. As Figure 1 shows, the logarithmic integral provides a remarkably good approximation to the prime-counting function across a wide range of values.

```
%% 5. Visualization: Comparing integral estimates with true prime counts

% Plot
hold on
plot(N_values,MC_log_integral,'o','DisplayName','MC estimate');
plot(N_values,prime_counts,'o','DisplayName','Actual prime count');
title(['Fig. 1. Monte Carlo estimation of logarithmic integral' ...
    ' for prime counting']);
xlabel('n');
ylabel('Number of primes smaller than or equal to n');
legend('show')
hold off
```

Fig. 1. Monte Carlo estimation of logarithmic integral for prime counting



6. Final notes

This file is prepared and copyrighted by Jelmer Wieringa and Tunga Kantarcı. This file and the accompanying MATLAB file are available on GitHub and can be accessed using this link.