Chapter 3 - Monte Carlo Methods

Monte Carlo Applications to Statistical Inference.
Point Estimators.

Prof. Alex Alvarez, Ali Raisolsadat

School of Mathematical and Computational Sciences
University of Prince Edward Island

Statistical Inference Problems

The Monte Carlo methods studied in previous lectures may be used to solve some practical Statistical Inference problems.

Consider for example the classical inference problem of point estimation.

Suppose we have some observations $x = (x_1, x_2, ..., x_n)$ that are considered a random sample of a random variable $X = (X_1, X_2, ..., X_n) \sim P_\theta$, where $\theta \in \Theta$ is an unknown parameter, and the problem is to estimate θ from the observations $x = (x_1, x_2, ..., x_n)$.

An **estimator** of θ is a function $\hat{\theta} = \hat{\theta}(X) = \hat{\theta}(X_1, X_2, ..., X_n)$. Notice that $\hat{\theta}(X)$ is a random variable.

The value of the estimator for the observed data x, $\hat{\theta}(x)$ is an **estimate** of θ. Notice that $\hat{\theta}(x)$ is a point in Θ.

Quality of an Estimator $\hat{\theta}(X)$

Good estimators $\hat{\theta} = \hat{\theta}(X)$ of the parameter θ should have a distribution that concentrates around θ. Specifically, the bias and the variance of an estimator give a good measure of its quality:

$$MSE(\hat{\theta}) = bias^2(\hat{\theta}) + Var(\hat{\theta})$$

There are several cased in which we can find the bias and variance explicitly, for example

- If the random variables X_i are i.i.d. and $\theta = E(X_i)$, the empirical mean estimator is unbiased.
- If n is large,sometimes we can use some probability tools such as the Central Limit Theorem.

In other cases, finding the bias and the variance of an estimator explicitly is not possible. Is in situations like these, that Monte Carlo methods may be useful.

Estimating the Bias

We start with

$$bias_\theta(\hat{\theta}) = E(\hat{\theta}(X)) - \theta$$

For a given value of θ, if we are able to generate N samples: $\left\{x^{(j)}\right\}_{j=1,2,\dots,N}$ of the random variable X we could estimate the bias with

$$\widehat{bias}_\theta\left(\hat{\theta}\right) = \frac{1}{N}\sum_{j=1}^{N}\hat{\theta}\left(x^{(j)}\right) - \theta$$

We can do this for a range of values of θ to get an understanding of the bias of an estimator as a function of θ.

Remarks

- Notice that we will generate N samples of the random variable X.
- The j-th sample will be of the form $x^{(j)} = \left(x_1^{(j)}, x_2^{(j)}, ..., x_n^{(j)}\right)$
- In order to estimate the bias of $\hat{\theta}$ for a fixed value θ we will have to generate $N \times n$ random values.
- It is important not to confuse n (the size of the samples in the estimator $\hat{\theta}$) with N (the number of samples used to estimate the bias).

Example - Estimating the Bias

Consider that you are given the following sample of independent random numbers and you are told that they come from a uniform distribution $[0, \theta]$ where the parameter θ is unknown. The problem is to estimate θ.

0.90648057	0.60372528	1.06419351	3.38445752	0.05913391
2.23352471	2.55686224	0.20315035	1.11948303	2.38614136

Example - Estimating the Bias

Consider that you are given the following sample of independent random numbers and you are told that they come from a uniform distribution $[0, \theta]$ where the parameter θ is unknown. The problem is to estimate θ.

0.90648057	0.60372528	1.06419351	3.38445752	0.05913391
2.23352471	2.55686224	0.20315035	1.11948303	2.38614136

One approach that we can take in this example is to use the naive estimator $\hat{\theta}(X) = \max(X_1, X_2, ..., X_{10})$, so for this specific sample we will have $\hat{\theta}(x) = 3.38445752$.

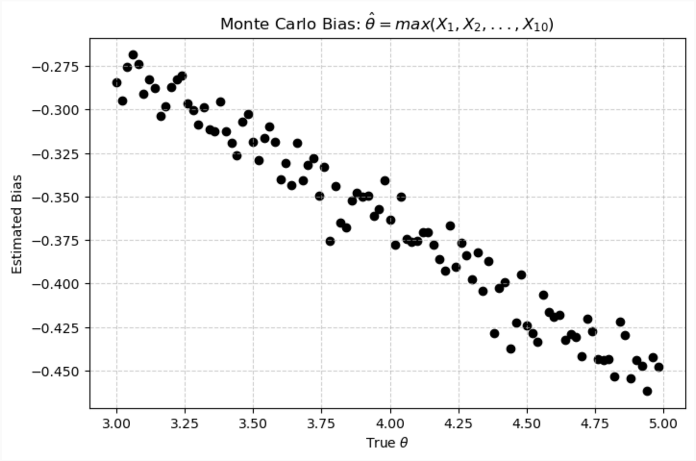
The naive estimator $\hat{\theta}(X)$ always underestimates the value of θ. In other words, we know that the estimator is (negatively) biased. We could try to estimate the bias for different values of θ.

Algorithm 1 Monte Carlo Bias Estimation for $\hat{\theta} = \max(U_1, \ldots, U_m)$

```
1:  Input: Number of replications $n = 1000$, number of samples $m = 10$, number of
    parameter values $J = 100$
2:  Initialize empty arrays: bias[1:J], theta[1:J]
3:  for $j = 1$ to $J$ do
4:      Set $\theta_j = 3 + 2 \times \frac{j}{J}$
5:      Initialize empty list samples
6:      for $i = 1$ to $n$ do
7:          Generate $U_{i1}, \ldots, U_{im} \sim \text{Uniform}(0, \theta_j)$
8:          Compute $\hat{\theta}_i = \max(U_{i1}, \ldots, U_{im})$
9:          Append $\hat{\theta}_i$ to samples
10:     end for
11:     Compute $\texttt{mean\_max}_j = \frac{1}{n} \sum_{i=1}^{n} \texttt{samples}_i$
12:     Compute $\texttt{bias}_j = \texttt{mean\_max}_j - \theta_j$
13: end for
14: Output: Arrays bias[1:J] and theta[1:J]
```

Algorithm 2 Semi-Vectorized Monte Carlo Bias Estimation for $\hat{\theta} = \max(U_1, \ldots, U_m)$

```
1: Input: Number of replications $n = 1000$, number of samples $m = 10$, number of parameter values $J = 100$
2: Initialize arrays: bias[1:J], theta[1:J]
3: for $j = 1$ to $J$ do
4:     Set $\theta_j = 3 + 2 \times \frac{j}{J}$
5:     Generate matrix $U \sim \text{Uniform}(0, \theta_j)$ of size $(n, m)$
6:     Compute vector $\hat{\boldsymbol{\theta}} = \max(U, \text{axis} = 1)$
7:     Compute $\texttt{mean\_max}_j = \frac{1}{n} \sum_{i=1}^{n} \hat{\theta}_i$
8:     Compute $\texttt{bias}_j = \texttt{mean\_max}_j - \theta_j$
9: end for
10: Output: Arrays bias[1:J] and theta[1:J]
```


Monte Carlo Bias: $\hat{\theta} = max(X_1, X_2, \ldots, X_{10})$
Estimated Bias
True θ
−0.275
−0.300
−0.325
−0.350
−0.375
−0.400
−0.425
−0.450
3.00
3.25
3.50
3.75
4.00
4.25
4.50
4.75
5.00

Example Remarks

From the previous graph we know that the bias seems to be linear on θ and that $\widehat{bias}_\theta(\hat{\theta}) \approx -0.09\theta$. In other words, **on average $\hat{\theta}(X)$ underestimates θ by a relative error of 9%** (at least if $3 \leq \theta \leq 5$).

(Actually, it can be proven that $E(\hat{\theta}) = \frac{n}{n+1}\theta = \frac{10}{11}\theta$ exactly but we are supposed not to know this)

You can use this information to improve the estimator, for example we could construct a new estimator $\hat{\theta_1} = \frac{100}{91}\hat{\theta}$ that will have a smaller bias.

Four our data we have $\hat{\theta}(x) = 3.38445752$ and $\hat{\theta_1}(x) = 3.719184$.

Can we guarantee that for our specific data the error in the estimate $\hat{\theta_1}(x)$ is going to be less than the error of $\hat{\theta}(x)$?

NO, we cannot guarantee that for specific samples x of X. Actually this data was generated with $\theta = 3.5$

Comparison of Monte Carlo Implementation Efficiency

Approach	Python Loops	FLOPs (approx.)	Relative Runtime
Fully Loop-Based	Double (over i, j)	2.0×10^6	$1\times$ (slowest)
Semi-Vectorized	Outer loop only	2.0×10^6	$\sim 10\times$ faster
Fully Vectorized	None	3.0×10^6	$\sim$50–100$\times$ faster

Observation: Even with similar FLOP counts, full vectorization reduces Python overhead dramatically, yielding large speedups.

Homework

Consider a random variable with exponential distribution with unknown mean $\mu \in [3,5]$. Out of a random sample of 9 independent random numbers from this distribution we are only given the minimum, the median and the maximum, and we should use that information only to estimate μ.

The following estimator of μ is proposed:

$$\hat{\mu}(X) = \frac{2min(X) + median(X) + max(X)}{4}$$

a) Use Monte Carlo Simulations to estimate the bias of this estimator, for values of μ on the interval $[3,5]$. Use $N = 1000$.

b) Based on your answer to a) propose a new estimator $\hat{\theta_1}$ of the form $\hat{\theta_1}(X) = \hat{\theta}(X) + C$, with $C \in \mathbb{R}$ such that that $\hat{\theta_1}$ has a smaller bias than $\hat{\theta}$ if $\mu \in [3,5]$.

Remark: Exponential distributions with mean μ have rate parameter $\lambda = 1/\mu$.