

Using Monte Carlo Integration and Control Variates to Estimate π

N. Cannady, P. Faciane, D. Miksa

LSU

July 9, 2009

We will demonstrate the utility of Monte Carlo integration by using this algorithm to calculate an estimate for π . In order to improve this estimate, we will also demonstrate how a family of covariate functions can be used to reduce the variance of this estimate. Finally, the optimal covariate function within this family is found numerically.

An Introduction to Monte Carlo Integration

Monte Carlo Integration is a method for approximating integrals related to a family of stochastic processes referred to as Monte Carlo Simulations. The method relies on the construction of a random sample of points so outputs are non-unique; however, these outputs probabilistically converge to the actual value of the integral as the number of sample points is increased.

An Introduction to Monte Carlo Integration

Since its development, Monte Carlo Integration has been used to evaluate many problems which otherwise become computationally inefficient or unsolvable by other methods. This process shows particularly useful in higher dimensions where the error found from other numerical methods grows too large.

The Monte Carlo Algorithm

To evaluate $I = \int_a^b f(x) dx$ by Monte Carlo Integration, first generate a sequence of N uniformly distributed random variables within the interval. That is, create $X_i \sim U[a, b]$ and let $Y_i = f(X_i)$ for $1 \leq i \leq N$.

$$\{X_1, X_2, \dots, X_N\} \rightarrow \{Y_1, Y_2, \dots, Y_N\}$$

and take

$$(b - a) \sum_{i=1}^N \frac{Y_i}{N}$$

as an approximation for I .

The Monte Carlo Algorithm

So given a particular function $f(x)$ and interval $[a, b]$, one can only control the size of the sequence generated, N , since the elements are randomly generated. Note that unlike deterministic methods (i.e. Simpson's Rule, Trapezoidal Rule), the estimate is liable to change with a particular N . One should also note, that similar to other methods, larger values of N yield better approximations.

The Monte Carlo Algorithm

We encounter similar methods throughout our daily lives. For example, voting is a simple discrete form of Monte Carlo integration where we attempt to measure a population's interest by collecting a sample of this population. The accuracy of a poll is often judged by the size and the distribution of the sample.

The Monte Carlo Algorithm

An example of this process is tossing rocks into a circular pond for an estimation of π . If we enclose a circular pond of radius $r = 1$ with a square having sides of length 2, we will see that $A_{square} * \frac{n}{N} \approx \pi$ where n is the number of rocks in the pond and N is the number of rocks within the square.

<http://www.eveandersson.com/pi/monte-carlo-demo>

Definition

For any continuous random variable $X \sim \rho(X)$ and $Y = f(X)$, the **expected value** of Y is defined as:

$$E[Y] = E[f(X)] = \int_{-\infty}^{\infty} f(x)\rho(x) dx.$$

If we take $\rho(x)$ to be the uniform probability density function on $[a, b]$ so that

$$\rho(x) = \begin{cases} \frac{1}{b-a} & \text{when } x \in [a, b], \\ 0 & \text{otherwise.} \end{cases}$$

then $E[Y]$ takes the form $\int_a^b f(x) \frac{1}{b-a} dx$. Hence,

$$I = \int_a^b f(x) \frac{b-a}{b-a} dx = (b-a) \int_a^b f(x) \frac{1}{b-a} dx = (b-a)E[Y].$$

Theorem

The Law of Large Numbers states that for any random variable X with $E[X] = \mu_X$, that $\bar{X}_N \xrightarrow{P} \mu_X$ as $N \rightarrow \infty$.

Because I can be expressed in terms of $E[Y]$, this means

$$(b - a)\bar{Y}_N \xrightarrow{P} (b - a)\mu_Y = (b - a)E[Y] = I.$$

Thus, we can say for large N , $(b - a)\bar{Y}_N = (b - a)\bar{f}(X_N) \approx I$.

Further Analysis of Monte Carlo Integration

Many of the details of MCI should seem strikingly similar to the process of Riemann integration. In both cases, we choose an arbitrary selection of points across the particular interval in mind, and use these values to construct a sum which proves more precise as the number of points is increased.

Theorem

Riemann Integrability

$\int_a^b f(x) dx$ exists and equals I if and only if

$$\forall \epsilon > 0, \exists \delta > 0 \text{ where } \left| \left(\sum_{i=1}^N f(\tilde{x}_i) \Delta x_i \right) - I \right| < \epsilon$$

$\forall X = \{x_1, x_2, \dots, x_{N+1}\}$ with $a \leq x_1 < x_2 < \dots < x_{N+1} \leq b$ and $\Delta x_i = x_{i+1} - x_i < \delta$ for $1 \leq i \leq N$ and $\forall \tilde{X} = \{\tilde{x}_i | \tilde{x}_i \in [x_i, x_{i+1}]\}$.

Definition

The Mesh of X

Denoted $\|X\|$, the mesh of a partition, X , is defined as the $\max\{\Delta x_i = x_{i+1} - x_i\}$ for $1 \leq i < N$ where N is the size of the partition.

Further Analysis..

Since for a set function and interval, the error of the Riemann Sum, ϵ , depends upon the size of the mesh, which is strictly less than δ , one should pay special attention to the behavior of the mesh during Monte Carlo simulations. Thus, the question becomes: if an interval is divided into N subintervals by $N - 1$ points chosen from the uniform distribution over that interval, what is the probability that no single subinterval is larger than δ ?

Further Analysis..

First, if the unit interval is split into N intervals, then $\|X\| \geq \frac{1}{N}$. Hence, the probability that the next point will refine the partition must be greater than or equal to $\frac{1}{N}$ which is positive for any $N \in 1, 2, \dots$

Furthermore, it can be shown that the probability that none of i specific subintervals will be less than δ is equal to $(1 - i\delta)^{N-1}$ [3]. Thus, the probability that the mesh of a partition is less than some given δ is given by

$$[3] \quad \Psi_{\delta}(N) = 1 - \sum_{i=1}^r (-1)^{i-1} \binom{N}{i} (1 - i\delta)^{N-1} \text{ where } r = \left\lfloor \frac{1}{\delta} \right\rfloor$$

which will show to increase in value as N is increased.

Further Analysis..

$$\Psi_{\delta}(N) \rightarrow 1 \text{ as } N \rightarrow \infty$$

Thus, sufficiently large values of N almost surely guarantees that the mesh of the partition from which the selection points are drawn is smaller than the required δ which in turn sharpens the approximation.

Further Analysis..

Yet, since the rate of convergence of Ψ is still in question and since the required δ may be impractically small, increasing N is not always the most efficient method to yield a more accurate result. We will later show yet another method for bettering the approximation.

Use of Monte Carlo Integration to Estimate π

Defining X and $f(X)$

We can now compute an estimate for the value of the definite integral $\int_{-1}^1 \frac{1}{1+x^2} dx$ using Monte Carlo Integration and use this to estimate the value of π . This is possible since it is known from calculus that

$$\int_{-1}^1 \frac{1}{1+x^2} dx = \frac{\pi}{2}$$

In order to use Monte Carlo Integration, first we define X to be a random variable uniformly distributed on the interval $[0,1]$, that is $X \sim U[0,1]$ for reasons that will become apparent in a moment. Next we let $f(X) = \frac{1}{1+x^2}$ which is a function of our random variable. By definition the expected value of $f(X)$ is

$$E[f(X)] = \int_{-\infty}^{\infty} f(x)g(x) dx$$

where $g(x)$ is the probability distribution of X .

Use of Monte Carlo Integration to Estimate π

Setting up the Simulation

Using the definitions from the introduction we see further that

$$E[f(X)] = \int_0^1 \frac{1}{1+x^2} dx$$

Since $f(X)$ is an even function, note that $2E[f(X)]$ is equal to the value of the desired definite integral.

Use of Monte Carlo Integration to Estimate π

Running the Simulation

Now we use a simulation to estimate $E[f(X)]$. This is done by instructing Mathematica to repeatedly pick a random number between 0 and 1 to use as X and then record the value of $f(X)$. Once this has been done one hundred thousand times, the mean is taken as an estimate of $E[f(X)]$. Recall that this is justified by the Law of Large Numbers explained previously. Finally, we multiply this estimate by 2 to get our estimate of the value of the desired definite integral. The precise Mathematica code utilizes the `Mean[]`, `Table[]`, and `Random[]` functions.

Use of Monte Carlo Integration to Estimate π

The results

Performing this simulation in Mathematica yields an estimate of 1.5713 which is fairly close to the known value of the definite integral $\frac{\pi}{2} \approx 1.570796\dots$. Furthermore, we can double our estimate and obtain 3.14261, a fairly close estimate of π .

Intro to Variance Reduction

Variance reduction refers to a variety of different methods which may be employed in conjunction with Monte Carlo simulations, including partial integration, systematic sampling, and control variates. In order to fully explain the following concepts, a few definitions must be established.

Definition

Variance

If X is a random variable with mean μ_X , the the variance of X , $Var(X)$, is defined by

$$Var(X) = E[(X - \mu_X)^2]$$

Definition

Covariance

Let X and Y be random variables. The covariance between X and Y , denoted $\text{Cov}(X, Y)$, is defined by

$$\text{Cov}(X, Y) = E[(X - E[X])(Y - E[Y])]$$

Definition

Correlation

The correlation of two random variables X and Y , denoted by $\rho(X, Y)$, is defined as

$$\rho(X, Y) = \frac{\text{Cov}(X, Y)}{\sqrt{\text{Var}(X)\text{Var}(Y)}}$$

as long as $\text{Var}(X)\text{Var}(Y) > 0$. It can be shown that $-1 \leq \rho(X, Y) \leq 1$

The joint goal of the aforementioned variance reduction methods is to minimize the variance on a simulation. The variance in a simulation represents the statistical uncertainty in the result. Thus, reduction of variance clearly leads to a more accurate result. We are interested in demonstrating a method using what are known as control variates and testing the efficacy of the control variates method.

Control Variates

The control variate method is useful when trying to simulate the expected value of a random variable, X . A second random variable, Y , for which the expected value is known, is introduced. The correlation between the two random variables must then be maximized such that the variance of the estimate of the X is reduced, leading to a more accurate simulation.

Suppose X is a random variable and that we wish to simulate $E[f(X)]$. Suppose also $\exists g(X)$ such that $E[g(X)] = \mu_g$. We then define a variable

$$W = f(X) + a[g(X) - \mu_g]$$

Note that

$$E[W] = E[f(X) + a[g(X) - \mu_g]] = E[f(X)]$$

Note that the variance of W is

$$\text{Var}(W) = \text{Var}[f(X)] + a^2 \text{Var}[g(X)] + 2a \text{Cov}[g(X), f(X)]$$

Derivation

The optimal value of a can be found using simple calculus by first differentiating with respect to a ,

$$\frac{d}{da}[\text{Var}(W)] = \frac{d}{da}[\text{Var}[f(X)] + a^2 \text{Var}[g(X)] + 2a \text{Cov}[g(X), f(X)]]$$

setting the derivative to 0,

$$0 = 2a \text{Var}[g(X)] + 2 \text{Cov}[g(X), f(X)]$$

and solving for a ,

$$a = -\frac{\text{Cov}[g(X), f(X)]}{\text{Var}[g(X)]}$$

We substitute this value of a into our formula for $Var(W)$ and get

$$Var(W) = Var[f(X)] - \frac{[Cov[g(X), f(X)]]^2}{Var[g(X)]}$$

We further define

$$R(\sigma) = \frac{[Cov[g(X), f(X)]]^2}{Var[g(X)]}$$

for notation convenience.

Family of $g_\sigma(X)$

In the variance reduction of our simulation, we used the family of functions

$$g_\sigma(X) = e^{-\frac{X^2}{\sigma}}$$

for

$$\sigma > 0$$

The parameter sigma must be optimized to determine which $g_\sigma(X)$ would most reduce the variance of our estimate.

We saw in the previous sections that

$$\text{Var}(W) = \text{Var}[f(X)] - \frac{[\text{Cov}[g(X), f(X)]]^2}{\text{Var}[g(X)]}$$

or

$$\text{Var}(W) = \text{Var}[f(X)] - R(\sigma)$$

We have no control over the value of $\text{Var}[f(X)]$ itself, due to its constancy. However, we if we can maximize the value of $R(\sigma)$, then we would minimize $\text{Var}(W)$. In order to analytically optimize $R(\sigma)$, we need to differentiate the term. We found $R(\sigma)$ to be analytically intractable and found alternative means for optimization.

Numerically Optimizing σ

To numerically calculate the optimal σ , rewrite the ratio in mind and use numerical methods to plot its value for a range of values.

$$R(\sigma) = \frac{\text{Cov}(f(x), g(x, \sigma))^2}{\text{Var}(g(x, \sigma))} = \frac{(E[f(x)g(x, \sigma)] - E[f(x)]E[g(x, \sigma)])^2}{E[g(x, \sigma)^2] - E[g(x, \sigma)]^2} =$$

$$\frac{(\frac{1}{2} \int_{-1}^1 f(x)g(x, \sigma) dx - (\frac{1}{2} \int_{-1}^1 f(x) dx)(\frac{1}{2} \int_{-1}^1 g(x, \sigma) dx))^2}{\frac{1}{2} \int_{-1}^1 g^2(x, \sigma) dx - (\frac{1}{2} \int_{-1}^1 g(x, \sigma) dx)^2} =$$

$$\frac{(\int_0^1 f(x)g(x, \sigma) dx - (\int_0^1 f(x) dx)(\int_0^1 g(x, \sigma) dx))^2}{\int_0^1 g^2(x, \sigma) dx - (\int_0^1 g(x, \sigma) dx)^2}$$

Numerically Optimizing σ

Intuitively, plotting $R(\sigma)$ should map out a peak near some region of σ and focusing in on this interval should justly yield an approximated σ . Since the integral form of $R(\sigma)$ can be evaluated both by Mathematica's built in functions or by the pre-described method of MCI, the optimal σ was evaluated using both methods for comparison.

Numerically Optimizing σ

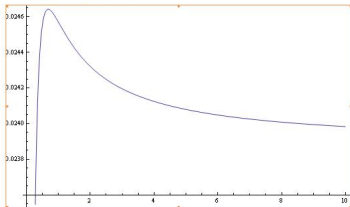
```
f[x_] = 1 / (1 + x^2);  
g[x_, s_] = E^(-x^2 / s);  
h[x_, s_] = f[x] g[x, s];
```

```
R = Table[RandomReal[], {1000}];
```

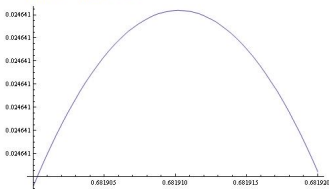
```
Covar[s_] = Mean[h[R, s]] - Mean[f[R]] Mean[g[R, s]];
```

```
VarG[s_] = Mean[(g[R, s] - Mean[g[R, s]])^2];
```

```
Plot[Covar[s]^2 / VarG[s], {s, .1, 10}]
```



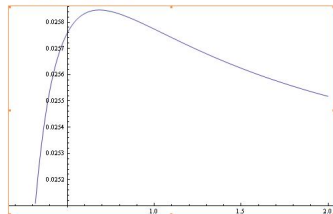
```
Plot[Covar[s]^2 / VarG[s], {s, .6819, .68192}]
```



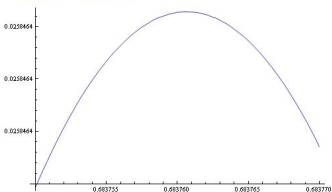
```
Covar[s_] = Integrate[h[x, s], {x, 0, 1}] - Integrate[f[x], {x, 0, 1}] Integrate[g[x, s], {x, 0, 1}];
```

```
VarG[s_] = Integrate[(g[x, s] - Integrate[g[x, s], {x, 0, 1}])^2, {x, 0, 1}];
```

```
Plot[Covar[s]^2 / VarG[s], {s, .2, 2}]
```



```
Plot[Covar[s]^2 / VarG[s], {s, .68375, .68377}]
```



Using MCI with a Control Variate to Estimate π

Defining X and W

Now that we have found the optimal value of σ , we define

$$W = f(X) + a[g_\sigma(X) - \mu_g]$$

where $\sigma = 0.68376$. Next, we define X to be a random variable uniformly distributed on the interval $[0,1]$, that is $X \sim U[0,1]$. As before, we then instruct Mathematica to repeatedly pick a random number between 0 and 1 to use as X and then record the value of W which is a function of X . Once this has been done several thousand times, the mean is taken as an estimate of $E[W] = E[f(X)]$. Finally, we multiply this estimate by 2 to get our estimate of the value of the desired definite integral.

Using MCI with a Control Variate to Estimate π




The final results

Performing this simulation in Mathematica yields an estimate of 1.57179 which is fairly close to the known value of the definite integral $\frac{\pi}{2} \approx 1.5708\dots$ Furthermore, we can double our estimate and obtain 3.14357, a fairly close estimate of π .

Acknowledgements

We would like to thank our professor, Dr. Walfredo Javier, and our graduate mentor, Ladorian Latin, as well as the VIGRE Summer program. We would also like to thank Dr. George Cochran, Dr. Robert Lax, Dr. Leonard Richardson, and Dr. Stephen Shipman for their insight into this study.

Bibliography

-  Ross, S., *Probability Models*, Elsevier, London, 2007.
-  Ross, S., *A First Course In Probability*, Pearson Prentice Hall, Upper Saddle River, New Jersey, 1976.
-  Parzen, E., *Modern Probability Theory*, John Wiley Sons, New York, 1960.