Monte Carlo method II

A. Godunov

1. Monte Carlo integration

1

Part 1:

Monte Carlo integration

2

Integration

- There are very many sophisticated methods for numerical integration
- Can Monte Carlo approach compete with traditional numerical methods?
- What can we gain, if anything, by applying "gambling" to integration?

3

There is clearly a problem with nD integration

Example: Integration for a system with 12 electrons.

- $3 * 12 = 36$ dimensional integral
- If 64 points for each integration then $=64^{36}$ points to evaluate
- For 1 Tera Flop computer = 10^{53} seconds
- That is … 3 times more then the age of the universe!

4

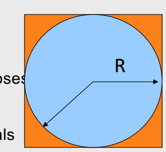
Two methods for MC integration

1. Monte Carlo Integration by "Stone Throwing" or "hit and miss" method.
2. Mean Value Integration (with many variations).

5

Calculating an area as hit and miss

Imagine that we need to evaluate the area of a circle (or any other shape)



Let the circle has a radius R

We draw a square box $2R \times 2R$ that completely encloses the circle (thus the area of the box if $4R^2$)

Then we set a counter n=0 and do a loop over N trials

1. We generate two random numbers x_i and y_i
2. If $x_i^2 + y_i^2 < R^2$ then the point is inside the circle and $n = n + 1$

Since $A_{circle}/A\_(square) \sim n/N$ for large enough N we have

$$A_{circle} \approx \frac{n}{N} A_{square} \approx \frac{n}{N} 4R^2$$

Note: we can even find π this way

$$\pi \approx 4n/N$$

6

Example (MatLab): Calculating π

```
rng('default')  % initilize RNG (default = "Mersenne Twister")
rng('shuffle')  % seed using current time
N = 10;
M = 10000000000;
while N < M
k = 0;
for j=1:N
    x = (2.0*rand-1.0);
    y = (2.0*rand-1.0);
    if sqrt(x*x+y*y) < 1.0
        k=k+1;
    end
end
MCpi = (k*4.0)/N;
fprintf(' N = %10i   MCpi = %8.6f \n',N,MCpi)
N=N*10;
end
```

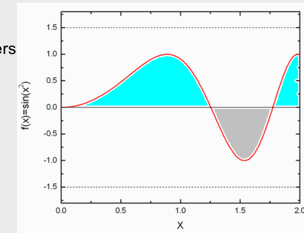
```
 N =         10   MCpi = 3.600000
 N =        100   MCpi = 3.240000
 N =       1000   MCpi = 3.164000
 N =      10000   MCpi = 3.136800
 N =     100000   MCpi = 3.141080
 N =    1000000   MCpi = 3.140732
 N =   10000000   MCpi = 3.140873
 N =  100000000   MCpi = 3.141517
 N = 1000000000   MCpi = 3.141654
```

7

1. Integration by rejection or hit and miss

Integral – area under a curve

Compute N pairs of random numbers x_i and y_i within the box, namely as $0.0 \le x_i \le 2.0, \ -1.5 \le y_i \le 1.5$ for the area $A = 2 * 3$.



If a point (x_i, y_i) is in the blue area then $n^+ = n^+ + 1$
In in the grey area then $n^- = n^- + 1$

Integral

$$I = A * \left(\frac{n^+ - n^-}{N}\right)$$

8

2. Mean value integration (cont.)

The standard Monte Carlo technique for integration is based on the mean value theorem

$$I = \int_a^b f(x)dx = (b-a) < f >$$

The Monte Carlo integration algorithm uses random points to evaluate the mean in the integral above.

$$I = \int_a^b f(x)dx \approx (b-a)\frac{1}{N}\sum_{i=1}^{N} f(x_i)$$

where x_i are uniform random numbers (random sampling) between a and b (unlike traditional numerical methods where x_i are chosen)

The laws of statistics ensure us that as $N \to \infty$, will approach the correct answer, at least if there were no round-off errors.

9

2. Mean value integration

We can estimate the accuracy of Monte Carlo integration as

$$I = \int_a^b f(x)dx \approx (b-a)\frac{1}{N}\sum_{i=1}^{N} f(x_i) \pm \Delta S$$

where

$$\Delta S = (b-a)\sqrt{\frac{\langle f^2 \rangle - \langle f \rangle^2}{N}}$$

$$\langle f \rangle = \frac{1}{N}\sum_{i=1}^{N} f(x_i), \qquad \langle f^2 \rangle = \frac{1}{N}\sum_{i=1}^{N} f^2(x_i),$$

Error in Monte Carlo integration $\sim \frac{1}{\sqrt{N}}$

10

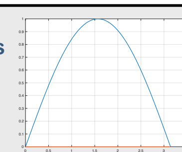
Example (MatLab) $\int_0^\pi \sin(x)\,dx = 2.0$

```
rng('default')      % reset RNG
rng('shuffle')      % use seed as current time
Nmax = 10000;
a = 0.0;
b = pi;
Sint = 0.0;
fav1 = 0.0;
fav2 = 0.0;
for it=1:Nmax
    x = a+(b-a)*rand;
    Sint = Sint + fint(x);

    fav1 = fav1 + fint(x);
    fav2 = fav2 + (fint(x))^2;
end
Sint = Sint*(b-a)/double(Nmax);
fprintf('Nmax = %9i \n', Nmax)

fav1 = fav1/double(Nmax);
fav2 = fav2/double(Nmax);
Serr = (b-a)*sqrt((fav2 - fav1^2)/Nmax);
fprintf('Sint = %9.6f ± %8.6f \n',Sint,Serr)

===
function F = fint(x)
    F = sin(x);
end
```

```
Nmax =     10000
Sint =  2.004742 ± 0.009631
```

11

Example: Comparing to other methods



n	Trapez.	Simpson	Monte Carlo
2	1.570796	2.094395	2.483686
4	1.896119	2.004560	2.570860
8	1.974232	2.000269	2.140117
16	1.993570	2.000017	1.994455
32	1.998393	2.000001	2.005999
64	1.999598	2.000000	2.089970
128	1.999900	2.000000	2.000751
256	1.999975	2.000000	2.065036
512	1.999994	2.000000	2.037365
1024	1.999998	2.000000	1.988752
2048	2.000000	2.000000	1.989458
4096	2.000000	2.000000	1.991806
8192	2.000000	2.000000	2.000583
16384	2.000000	2.000000	1.987582
32768	2.000000	2.000000	1.991398
65536	2.000000	2.000000	1.997360

$$\int_0^\pi \sin(x)\,dx = 2.0$$

Error in Monte Carlo integration $\sim \frac{1}{\sqrt{N}}$, while Simpson $\sim \frac{1}{N^4}$

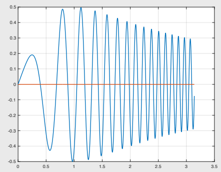
Is there advantage to use Monte Carlo method?

12

Example: Comparing to other methods

n	Trapez.	Simpson	Monte Carlo
64	0.004360	-0.013151	0.081207
128	0.001183	-0.001110	0.155946
256	0.000526	-0.000311	0.071404
512	0.000368	0.000006	0.002110
1024	0.000329	0.000161	-0.004525
2048	0.000319	0.000238	-0.010671
4096	0.000316	0.000277	0.000671
8192	0.000316	0.000296	-0.009300
16384	0.000316	0.000306	-0.009500
32768	0.000316	0.000311	-0.005308
65536	0.000316	0.000313	-0.000414
131072	0.000316	0.000314	0.001100
262144	0.000316	0.000315	0.001933
524288	0.000316	0.000315	0.000606
1048576	0.000316	0.000315	-0.000369
2097152	0.000316	0.000316	0.000866
4194304	0.000316	0.000316	0.000330

$$\int_0^{\pi} \frac{x}{x^2+1}\cos(10x^2)\,dx = 0.0003156$$



Very slow convergence for the MC method

13

Methods to increase accuracy of MC integration

Accuracy of the method can be improved either by increasing the number of samples (more points) OR by reducing the variance

Four most common methods for reducing the variance

1. Variance reduction by subtraction
2. Antithetic variates
3. Importance sampling (most efficient method!)
4. Stratified sampling

14

1. Variance reduction by subtraction

If the function being integrated never differs much from its average value, then the standard Monte Carlo mean value method should work well with a manageable number of points.

For a function with a large variance (i.e., one that is not "flat"), many of the evaluations of the function may occur for x values at which the function is very small - basically, a waste of time.

A variance reduction or subtraction technique - we devise a flatter function on which to apply the Monte Carlo technique.

Let construct a function $g(x)$ with the following properties on $[a, b]$:

1. The function can be evaluated analytically $\int_a^b g(x)dx = J$
2. And $g(x)$ is close to $f(x)$. $|f(x) - g(x)| < \delta$

Then $\int_a^b f(x)dx = \int_a^b \big(f(x) - g(x)\big)dx + J$

If the variance of $f(x) - g(x)$ less than that of $f\ (x)$, then we can obtain even more accurate answers in less time.

15

2. Antithetic variates

The antithetic variates is based on the concept that u_i and $\{1 - u_i\}$ are negatively correlated. (Note that u_i belongs to a uniform random number distribution between 0 and 1.)

Thus for $x_i = a + (b-a)u_i,\ x_{ia} = a + (b-a)(1-u_i)$ and the integral

$$I = \int_a^b f(x)dx = \frac{1}{2N}\sum_{i=1}^{N}\big(f(x_i) + f(x_{ia})\big)$$

The advantage of this technique is twofold:

1. It reduces the number of normal samples to be taken
2. It reduces the variance of the sample paths, improving the precision

16

3. Importance sampling

The objective of the importance sampling is to sample the integrand in the most important regions. It based on the identity

$$I = \int_a^b f(x)dx = \int_a^b \frac{f(x)}{p(x)}p(x)dx\,.$$

The integral can be approximated as

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

where $p(x)$ is a **normalized** probability distribution of x_i in $[a, b]$ interval

$$\int_a^b p(x)dx = 1$$

Note that in the uniform case $p(x) = 1/(b-a)$.

17

Importance sampling (cont.)

For a given integrand $f(x)$, we should choose $p(x)$, such that the modified integrand $f(x)/p(x)$ becomes as smooth as possible. The importance sampling can considerably improve the accuracy.

Example:

$$\int_0^{\infty} x\,e^{-x}dx = 1$$

Most contributions comes from the origin area (defined by e^{-x}). Thus,

$$\int_0^{\infty} x\,e^{-x}dx = \int_0^{\infty}\frac{xe^{-x}}{e^{-x}}e^{-x}dx$$

$$I = \frac{1}{N}\sum_{i=1}^{N}x_i$$

with x_i from a non-uniform distribution $p(x) = e^{-x}$ (that is already normalized)

18

On the practical side (steps)

1. Choose a function $p(x)$ so that $f(x)/p(x)$ is as smooth as possible
2. Normalize $p(x)$ so that $\int_a^b p(x) = 1$
3. Generate a non-uniform distribution of random numbers $\{x_1, x_2, \ldots\}$ based on $p(x)$ distribution/function
   You can use libraries, or one of methods: transformation, rejection, or Metropolis method*
4. ATTENTION!
   The generated non-uniform distribution of $\{x_1, x_2, \ldots\}$ must be within the $[a, b]$ interval (this is the tricky part!)
5. And now you can compute

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

19

Example: $\int_0^1 e^{-x^2}\,dx = 0.746824$

1. Sure, it's natural to sample the integral with $p(x) = e^{-x^2}$, however we try to sample with $p(x) = e^{-x}$.
2. Normalization: $\int_0^1 e^{-x}\,dx = 0.63212056$, or $p(x) = 1.582e^{-x}$
3. We can generate the distribution using the transform method as
   $$x = -\log(\text{rand})$$
   however in this case we will get x between 0 and ∞!
4. Scaling the distribution from 0 and ∞ to $[a, b]$ where $a = 0, b = 1$ using $a = -\log(e^{-a})$
   $$x = -\log\left[e^{-a} + \left(e^{-b} - e^{-a}\right) * rand\right]$$
5. And finally

$$I = \frac{1}{N}\sum_{i=1}^{N}\frac{e^{-x_i^2}}{e^{-x_i}}$$

20

Calculations

$$\int_0^1 e^{-x^2}\,dx = 0.746824$$

For two $p(x) = 1$ and $p(x) = 1.582e^{-x}$ with N=100,000 points

$p(x) = 1$: `Integral=0.746699, Error=0.000636`

$p(x) = 1.582e^{-x}$ `Integral=0.746758, Error=0.000174 (using transform)`

$p(x) = 1.582e^{-x}$ `Integral=0.747806, Error=0.000176 (using Metropolis)`

21

Importance sampling and Metropolis algorithm

While the transform method for generating a non-uniform distribution is superior to Metropolis method, we often use the later when we don't have the inverse function

However, in estimating integrals the estimated error using the Metropolis method is much smaller than the actual error!

The reason is that the $\{x_i\}$ are not statistically independent. The Metropolis algorithm produces a random walk whose points are correlated with each other over short times (measured-by the number of steps of the random walker).

The correlation of the points decays exponentially with time. If τ is the characteristic time for this decay, then only points separated by approximately 2 to 3τ can be considered statistically independent.

Calculate autocorrelation function $C(j)$ to see the period

$$C(j) = \frac{\langle x_{i+j}x_i\rangle - \langle x_i\rangle^2}{\langle x_i^2\rangle - \langle x_i\rangle^2}$$

see more in Gould et al (2006), page 437.

22

4. Stratified sampling

Divide the domain of integration into smaller parts.

23

Multidimensional integration

The mean value integration

$$\int_a^b\int_c^d f(x,y)\,dy\,dx \cong (b-a)(d-c)\frac{1}{N}\sum_{i=1}^{N} f(x_i, y_i)$$

Errors in integration

Monte Carlo 1D integration $\sim \frac{1}{\sqrt{N}}$

Monte Carlo nD integration $\sim \frac{1}{\sqrt{N}}$ (the same as 1D case!)

Simpson 1D $\sim \frac{1}{N^4}$

Simpson nD $\sim \left(\frac{1}{N^4}\right)^{1/n}$

Thus at at $n\sim 8$, the error in Monte Carlo integration is similar to that of conventional scheme!

Monte Carlo integration is efficient for multidimensional integration!

24

Example:

$$\int_0^1 dx_1 \int_0^1 dx_2 \int_0^1 dx_3 \int_0^1 dx_4 \int_0^1 dx_5 \int_0^1 dx_6 \int_0^1 (x_1 + x_2 + \ldots + x_7)^2 dx_7 = 12.83333333$$

N	7D Integral
8	11.478669
16	12.632578
32	13.520213
64	13.542921
128	13.263171
256	13.178140
512	12.850561
1024	12.747383
2048	12.745207
4096	12.836080
8192	12.819113
16384	12.790508
32768	12.765735
65536	12.812653
131072	12.809303
262144	12.831216
524288	12.832844

25