Computational Physics Lab

Multi-Dimensional & Monte Carlo Integration

04/21/2009

Outline

Multi-Dimensional Integration

$$\int_{x_a}^{x_b} \int_{y_a}^{y_b} f(x,y)\,dx\,dy$$

$$\int_{x_a}^{x_b} f(x)\,dx, \quad \textit{where} \quad f(x) = \int_{y_a}^{y_b} f(x,y)\,dy$$

Solve by Series Integration

- Newton-Cotes
- Trapezoidal
- 3-Point Simpson

➜ For N points in each integral, there are N^2 calculations.

Multi-Dimensional Integration

Example: Atomic Physics

$$I = \int_0^1 dx_1 \int_0^1 dx_2 \ldots \int_0^1 dx_{12} f(x_{1,} x_{2,} \ldots x_{12})$$

3 Dimensions/electron * 4 electrons = 12 Dimensions

For 100 points in each integration,
there are $100^{12} = 10^{24}$ calculations

Assuming 1 Giga evaluations/sec,
it would take over 10^7 years!!!!

Outline

Monte Carlo Integration

Using Random Numbers to Solve Integrals

Monte Carlo methods provide an alternative method of calculating an integral.

Pseudo-Random Numbers

drand48()

SYNOPSIS

```
#include <cstdlib>

double drand48(void);
void srand48(long int seedval);
```

The function *drand(48)* generates uniformly distributed pseudo-random number between [0.0,1.0). This method uses a linear congruential algorithm and 48-bit integer arithmetic.

Seed Generator

```
time_t t;    included in <ctime>
srand48(time(&t));
```

Get Random Number

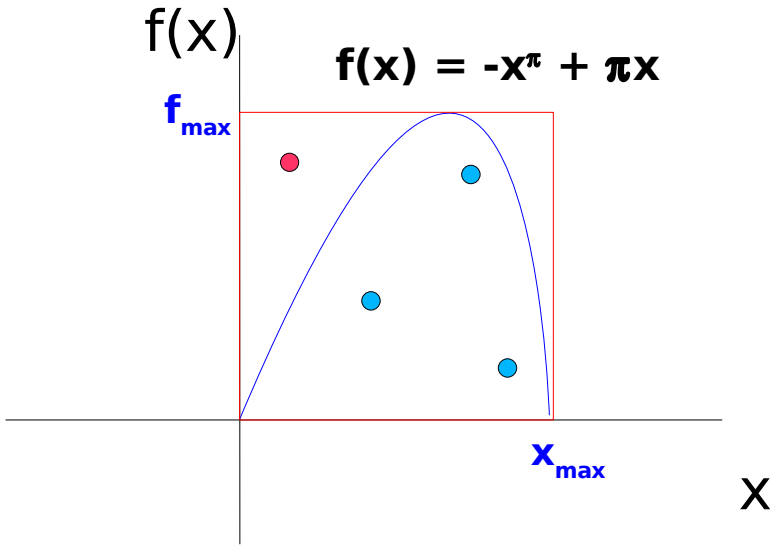
```
double randomX =
  xMin + (xMax - xMin) · drand48();
```

Monte Carlo Techniques

Two of the most common ways to employ Monte Carlo techniques are:

- Monte Carlo Sampling Method
- Monte Carlo Mean-Value Method

Monte Carlo Sampling Method



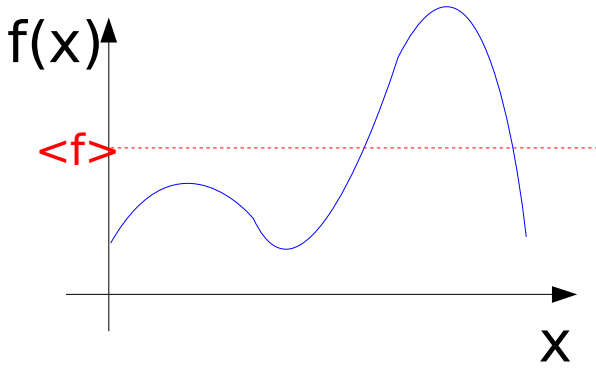
Box off the region of integration and calculate the area of the box.

- Randomly place points in the box.
- Count # of points in the box versus # of points under the function.
- $\text{Area}_{f(x)} = \text{Area}_{\text{box}} * \frac{N_{f(x)}}{N_{box}}$

Monte Carlo Mean-Value Method

$$I = \int_a^b dx\, f(x) = (b-a)\langle f \rangle$$

$$\langle f \rangle \simeq \frac{1}{N} \sum_{i=0}^{N} f(x_i)$$



Statistical Error

$$\delta I \sim \sigma_{\bar{f}}$$

standard deviation of mean

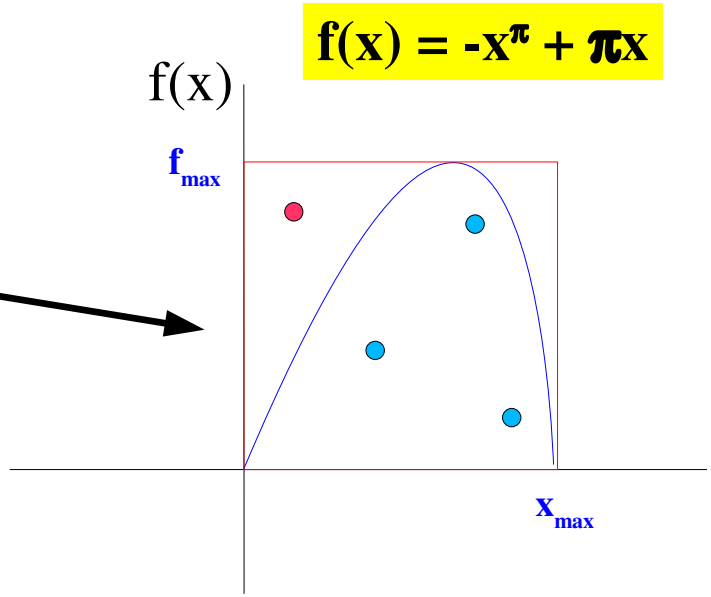
$$\sigma_{\bar{f}} = \sigma_f / \sqrt{N}$$

Integration using the Monte Carlo method is done by averaging the value of the function at randomly selected points within the integration interval.

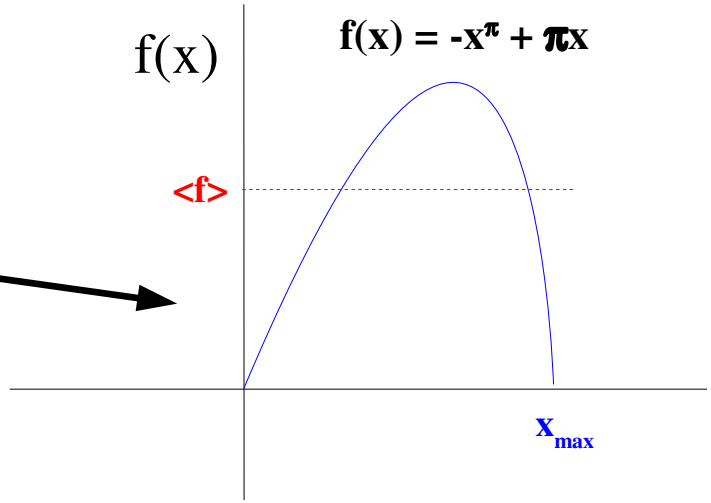
Outline

Monte Carlo Example

```
for(int i = 0; i < nPoints; i++ ){
   x = xMax * drand48();
   fRand = fMax * drand48();
   if( fRand < f( x ) )      nUnder++;
}
 cout << "Integration by Samples = "
          << ( fMax * xMax )
               * ( nUnder / nPoints ) << endl;
```




```
for(int i = 0, sum = 0; i < nPoints; i++ ){
   x = xMax * drand48();
   sum += f( x );
 }
cout << "Integration by Mean-Value = "
       << xmax * ( sum / nPoints ) << endl;
```



Multi-Dimensional Monte Carlo

- Easy to generalize mean-value integration to many dimensions
- Monte Carlo error is statistical
  - decreases as 1/sqrt(N)

2-Dimensions

$$I=\int_a^b dx_1 \int_c^d dx_2 f(x,y) \simeq (b-a)(d-c) * \frac{1}{N} \sum_i^N f(x_i, y_i)$$

Multi-Dimensional Integration

Example: Atomic Physics

$$I = \int_0^1 dx_1 \int_0^1 dx_2 \ldots \int_0^1 dx_{12} f(x_1, x_2, \ldots x_{12})$$

3 Dimensions/electron * 4 electrons = 12 Dimensions

$$\simeq (1-0)^{12} * \frac{1}{N} \sum_i^N f(x_1^i \;\; x_2^i \;\; \ldots \;\; x_{12}^i)$$

For N=10^6 random points in the MC integration
there are ~10^6 calculations

Assuming 1 Giga evaluations/sec,
It would take ~ 10^{-3} sec

compare to 10^7 yrs