# Adaptive Monte Carlo Integration with General Division Approach

**Houssam Abdul Rahman**

**Department of Mathematics**
**University of Alabama at Birmingham**

Feb 2012

# Monte Carlo Methods

- A *Monte Carlo* (MC) method is a procedure that involves the use of statistical sampling procedures to estimate the solution of mathematical or physical problems.
- A MC method solves non-probabilistic problems using probabilistic methods.

# The Problem

We consider the $d$-dimensional integral

$$I = \int_{\Gamma} f(\mathbf{x}) d\mathbf{x} \tag{1}$$

where

- $\mathbf{x} \in \Re^d$, $d \geq 1$.
- $\Gamma$ denotes the $d$-dimensional hyper-rectangular $[a_1, b_1] \times [a_2, b_2] \times \ldots \times [a_d, b_d] \subset \Re^d$.
- $f : \Gamma \longrightarrow \Re$ is square integrable function on $\Gamma$.

# The Hit or Miss MC

Consider the following one dimensional integral

$$I = \int_a^b f(x)dx$$

where $a, b \in \Re$, $0 \leq f(x) \leq M$ , $\forall\ x \in [a, b]$, $M \in \Re^+$.

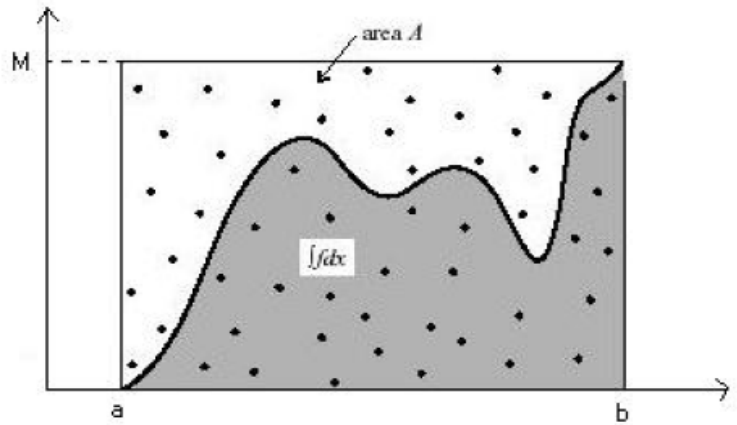


**Figure:** The Hit or Miss MC method

# The Sample Mean MC Integration

- We can rewrite the integral $I$ given in equation (1) as

$$I = \int_{\Gamma} h(\mathbf{x})\phi(\mathbf{x})d\mathbf{x}$$

where

$$\phi(\mathbf{x}) > 0, \quad \int_{\Gamma} \phi(\mathbf{x})d\mathbf{x} = 1$$

- The Sample-Mean MC integration is based on the following theorem

## Theorem

*If $N$ independent samples $\mathbf{x}_i$ are selected from $\Gamma$, according to the multidimensional p.d.f. $\phi(\mathbf{x})$ then*

$$\hat{I} = \frac{1}{N}\sum_{i=1}^{N} h(\mathbf{x}_i)$$

*is a consistent estimator of the integral $I$.*

## Corollary

*Consider the multidimensional integral (1) then, if we pick $N$ points $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N$ randomly from the multidimensional region $\Gamma$. then*

$$I \simeq V < f > \pm V\frac{\hat{\sigma}_f}{\sqrt{N}}$$

$$< f >= \frac{1}{N}\sum_{i=1}^{N} f(\mathbf{x}_i), \quad \hat{\sigma}_f^2 =< f^2 > - < f >^2$$

# $f$ is square integrable: Comment

Consider the following one dimensional integral

$$I = \int_0^1 f(x)dx$$

where

$$f(x) = \begin{cases} \frac{1}{\sqrt{x}} & \text{if } 0 < x \leq 1 \\ 0 & \text{if } x = 0 \end{cases}$$

The exact value of $I$ is 2.

| N | Basic MC | Absolute error |
|---|---|---|
| 1,000 | 2.056815 | 0.056815 |
| 10,000 | 2.050371 | 0.050371 |
| 100,000 | 1.990308 | 0.009692 |
| 1000,000 | 1.996605 | 0.003395 |

**Table:** The basic MC integration output

# Probabilistic Error Bounds

$$p\left(\frac{-V\sigma_f}{\sqrt{N}} \leq \frac{V}{N}\sum_{n=1}^{N} f(\mathbf{x}_n) - I \leq \frac{V\sigma_f}{\sqrt{N}}\right) \simeq 0.683$$

There is less than $68.3\%$ chance that the absolute error lies inside the estimated error bounds.

# Example

$$I = \int_0^1 \int_0^1 \int_0^1 \int_0^1 \int_0^1 \int_0^1 \int_0^1 \int_0^1 \int_0^1 \int_0^1 2^{10} \prod_{i=1}^{10} \{\sin^3(\pi x_i) dx_i\}$$

| N | Basic MC | Estimated error | Absolute error |
|---|---|---|---|
| 10,000 | 0.174743 | 0.022086 | 0.019457 |
| 100,000 | 0.182407 | 0.008792 | 0.011743 |
| 1000,000 | 0.195159 | 0.003042 | 0.000959 |

**Table:** Basic MC integration output.

# Integration over complicated regions

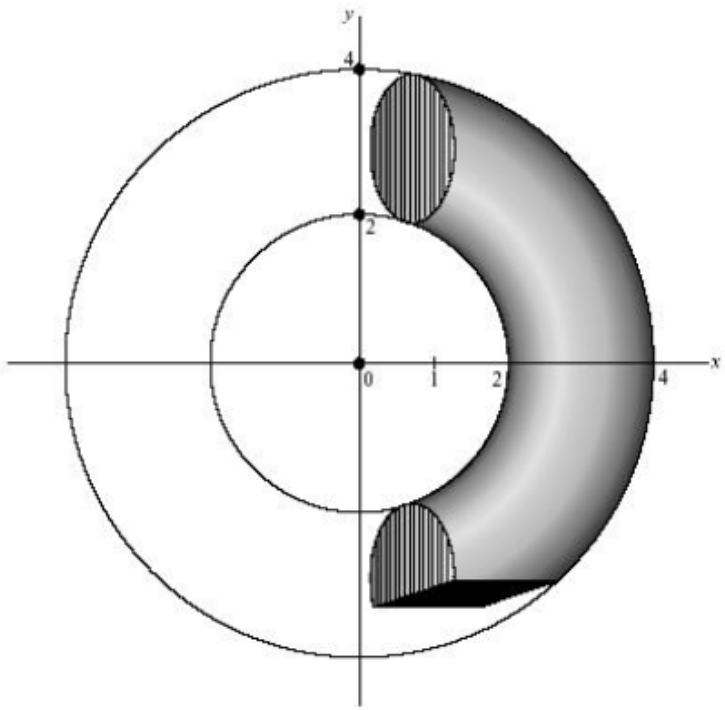


**Figure:** A portion of a torus

# High Dimensional Integration

| | conv. rate in one-dim. | conv. rate in $d$-dim. |
|---|---|---|
| Basic MC | $N^{-1/2}$ | $N^{-1/2}$ |
| Trapezoidal rule | $N^{-2}$ | $N^{-2/d}$ |
| Simpson's rule | $N^{-4}$ | $N^{-4/d}$ |

# Stratified Sampling

- The Stratified Sampling is an attempt to ensure that important regions of the domain get sampled.
- The idea is based on dividing the whole integration region $\Gamma$ into $k$ subregions $\Gamma_i$, $i = 1, 2, \ldots, k$.
- The basic MC integration is performed in each subregion.
- The estimated integral is the sum of the MC integrations in all subregions; so,

$$\hat{I} = \sum_{i=1}^{k} \frac{V_i}{N_i} \sum_{j=1}^{N_i} f(\mathbf{x}_{ij})$$

where $V_i$ is the volume of $\Gamma_i$, and $N_i$ is the samplings in it.

# Adaptive Monte Carlo Integration (AMC)

- Adaptive Algorithms aim to reduce the estimated (absolute) error(s) of the integration estimation (approximation)
- Adaptive Algorithms (in general) learn about the function as they proceed.
- We consider the adaptive algorithm that uses iteratively the idea of Stratified Sampling using a *Global Subdivision Approach*

# The General Form of the AMC Algorithm

**Algorithm 1:**

- Put the whole domain of integration into region collection.
- **While** Stopping criterion is not satisfied **Do**
  Choose subregion with largest estimated error.
  Split this region.
  Apply the basic MC method to newly created regions.
  Store new regions in region collection.
- **Endwhile**.

# Notations

- $\Omega_i$ : the region collection in iteration $i$.
- $\Gamma_i(j)$ : the subregion that belongs to $\Omega_i$ with index $j$.
- $\hat{I}_{ij}$ : the basic MC integration estimate over the region $\Gamma_i(j)$.
- $\hat{I}_i = \sum_j \hat{I}_{ij}$.
- $\hat{E}_{ij}$ : the standard estimated error of integration over the subregion $\Gamma_i(j)$.
- $\hat{E}_i = \sqrt{\sum_j \hat{E}_{ij}^2}$.
- $\alpha_i$ : the promising index; $\alpha_i = \arg\max_j \hat{E}_{ij}$.
- $\Gamma_i^*$ : the promising region; $\Gamma_i^* = \Gamma_i(\alpha_i)$

# The Division Of a Hyper-rectangle

- **The $s$-division:** choose $s$ distinct coordinates and divide each one into two intervals.
- The $s$-division produces $2^s$ subregions.
- So, if the $s$-division is performed then $|\Omega_i| = i(2^s - 1) + 1 = v_i$.

# The Storage Approach

- Suppose $\Omega_i$ contains

$$\Gamma_i(j) \equiv \{\mathbf{x}, \mathbf{x} \in [a_{j1}, b_{j1}] \times \ldots \times [a_{jd}, b_{jd}]\} \quad , j = 1, 2 \ldots, v_i$$

then all these subregions are represented by the following two arrays.

$$A^i = \begin{pmatrix} a_{11} & \ldots & a_{1d} \\ a_{21} & \ldots & a_{2d} \\ \vdots & & \\ a_{v_i 1} & \ldots & a_{v_i d} \end{pmatrix}, B^i = \begin{pmatrix} b_{11} & \ldots & b_{1d} \\ b_{21} & \ldots & b_{2d} \\ \vdots & & \\ b_{v_i 1} & \ldots & b_{v_i d} \end{pmatrix}$$

- Also, $TM^i = [\hat{I}_{i1}, \ldots, \hat{I}_{iv_i}]$ and $TE^i = [\hat{E}_{i1}, \ldots, \hat{E}_{iv_i}]$.

# The Transition From Iteration $i$ Into Iteration $i+1$

- $\Gamma_i^* = \Gamma_i(\alpha_i)$ will be divided into $2^s$ subregions $\Gamma_{i+1}(\alpha_i), \Gamma_{i+1}(\alpha_i + 1), \ldots, \Gamma_{i+1}(\alpha_i + 2^s - 1)$. So, $\Omega_{i+1} = \{\Gamma_i(1), \ldots, \Gamma_i(\alpha_i - 1), \Gamma_{i+1}(\alpha_i), \Gamma_{i+1}(\alpha_i + 1), \ldots, \Gamma_{i+1}(\alpha_i + 2^s - 1), \Gamma_i(\alpha_i + 1), \ldots, \Gamma_i(v_{i+1})\}$.
- $TM^i$ and $TE^i$ are updated in a same manner as $\Omega_i$.
- The $\alpha_i$ row of $A^i$ and $B^i$ will be replaced by news $2^s$ row.
- The inserted row are a copy rows of the $\alpha_i$ row, except for some values in their intersection with the $s$ columns that correspond with the $s$ selected divided coordinates.
- This intersection will become the following two arrays:

$$\begin{pmatrix} * & * & \dots & * & * \\ * & * & \dots & * & c_s \\ * & * & \dots & c_{s-1} & * \\ * & * & \dots & c_{s-1} & c_s \\ & & \vdots & & \\ * & c_2 & \dots & * & * \\ * & c_2 & \dots & * & c_s \\ * & c_2 & \dots & c_{s-1} & * \\ * & c_2 & \dots & c_{s-1} & c_s \\ & & \vdots & & \\ c_1 & c_2 & \dots & c_{s-1} & c_s \end{pmatrix}, \begin{pmatrix} c_1 & c_2 & \dots & c_{s-1} & c_s \\ c_1 & c_2 & \dots & c_{s-1} & * \\ c_1 & c_2 & \dots & * & c_s \\ c_1 & c_2 & \dots & * & * \\ & & \vdots & & \\ c_1 & * & \dots & c_{s-1} & c_s \\ c_1 & * & \dots & c_{s-1} & * \\ c_1 & * & \dots & * & c_s \\ c_1 & * & \dots & * & * \\ & & \vdots & & \\ * & * & \dots & * & * \end{pmatrix}$$

where $c_j$, $j = 1, \dots, s$ divides the axis number $j$.

# Numerical Example

$$J = \int_{[0,1]^{30}} \frac{4x_1 x_3^2 \exp(2x_1 x_3)}{(1 + x_2 + x_4)^2} \exp(x_5 + \ldots + x_{20}) x_{21} x_{22} \ldots x_{30} dx_1 \ldots dx_{30}$$

| Iteration $i$ | $\Gamma_i(j)$ | Estimated integral | Estimated error |
|---|---|---|---|
| iteration 0 | $\Gamma_0(1)$ | 3.351107 | 0.221240 |
| | $\Gamma$ | **3.351107** | **0.221240** |
| iteration 1 | $\Gamma_1(1)$ | 1.210097 | 0.062587 |
| | $\Gamma_1(2)$ | 2.289098 | 0.239714 |
| | $\Gamma$ | **3.499195** | **0.247750** |
| iteration 2 | $\Gamma_2(1)$ | 1.210097 | 0.062587 |
| | $\Gamma_2(2)$ | 1.253465 | 0.067066 |
| | $\Gamma_2(3)$ | 0.690014 | 0.040054 |
| | $\Gamma$ | **3.153576** | **0.100097** |
| iteration 3 | $\Gamma_3(1)$ | 1.210097 | 0.062587 |
| | $\Gamma_3(2)$ | 0.313236 | 0.020022 |
| | $\Gamma_3(3)$ | 0.935178 | 0.046537 |
| | $\Gamma_3(4)$ | 0.690014 | 0.040054 |
| | $\Gamma$ | **3.148525** | **0.08993** |

**Table:** A detailed 4 iterations output of 1-AMC algorithm using $N = 5 \times 10^4$ for estimating $J$.

| $i$ | $\hat{I}_i$ | $\hat{E}_i$ | Absolute error |
|---|---|---|---|
| 0 | 3.351107 | 0.221240 | 0.107107 |
| 1 | 3.499195 | 0.247750 | 0.255195 |
| 2 | 3.153576 | 0.100097 | 0.090424 |
| 3 | 3.148525 | 0.089933 | 0.095475 |
| 4 | 3.256002 | 0.109971 | 0.012002 |
| 5 | 3.234978 | 0.082208 | 0.009022 |
| 6 | 3.272827 | 0.081168 | 0.028827 |
| 7 | 3.238134 | 0.074790 | 0.005866 |
| 8 | 3.348110 | 0.088251 | 0.104110 |
| 9 | 3.235390 | 0.066780 | 0.007610 |

**Table:** A 10 iterations output of Algorithm 2 using $N = 5 \times 10^4$ for estimating $J$.

| Iteration $i$ | $\Gamma_i(j)$ | Estimated integral | Estimated error |
|---|---|---|---|
| iteration 0 | $\Gamma_0(1)$ | 3.349646 | 0.354838 |
| | $\Gamma$ | **3.349646** | **0.354838** |
| iteration 1 | $\Gamma_1(1)$ | 0.060929 | 0.006080 |
| | $\Gamma_1(2)$ | 0.784820 | 0.059846 |
| | $\Gamma_1(3)$ | 0.176693 | 0.020387 |
| | $\Gamma_1(4)$ | 2.456285 | 0.227911 |
| | $\Gamma$ | **3.478727** | **0.236596** |
| iteration 2 | $\Gamma_2(1)$ | 0.060929 | 0.006080 |
| | $\Gamma_2(2)$ | 0.784820 | 0.059846 |
| | $\Gamma_2(3)$ | 0.176693 | 0.020387 |
| | $\Gamma_2(4)$ | 0.330183 | 0.027868 |
| | $\Gamma_2(5)$ | 0.456953 | 0.029239 |
| | $\Gamma_2(6)$ | 0.578463 | 0.051105 |
| | $\Gamma_2(7)$ | 0.961674 | 0.129045 |
| | $\Gamma$ | **3.349716** | **0.157892** |

**Table:** A detailed 3 iterations output of 2-AMC algorithm using $N = 15,000$ for

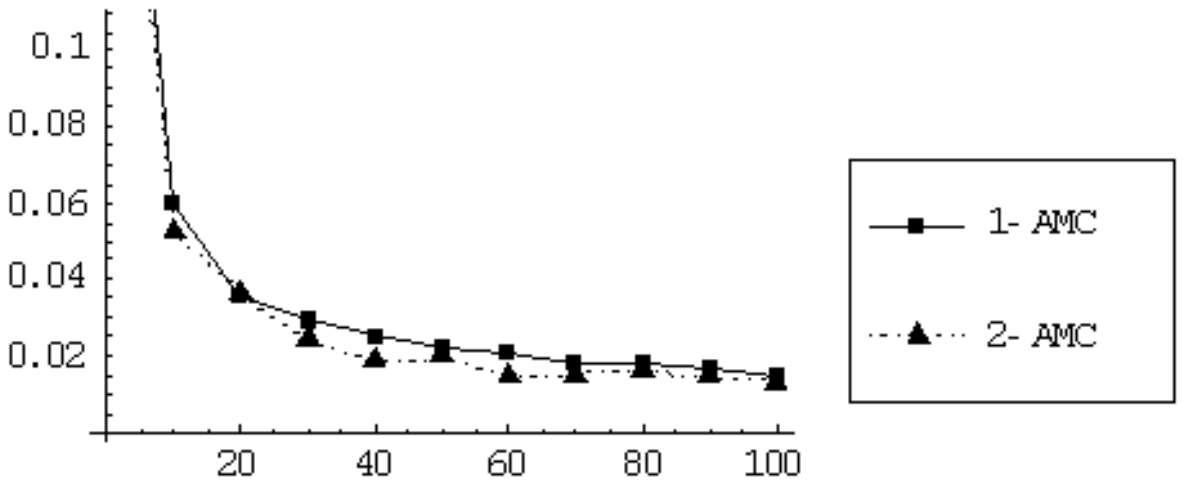


**Figure:** The estimated error in 1-AMC and 2-AMC algorithms during 100 iterations with almost the same number of sample points

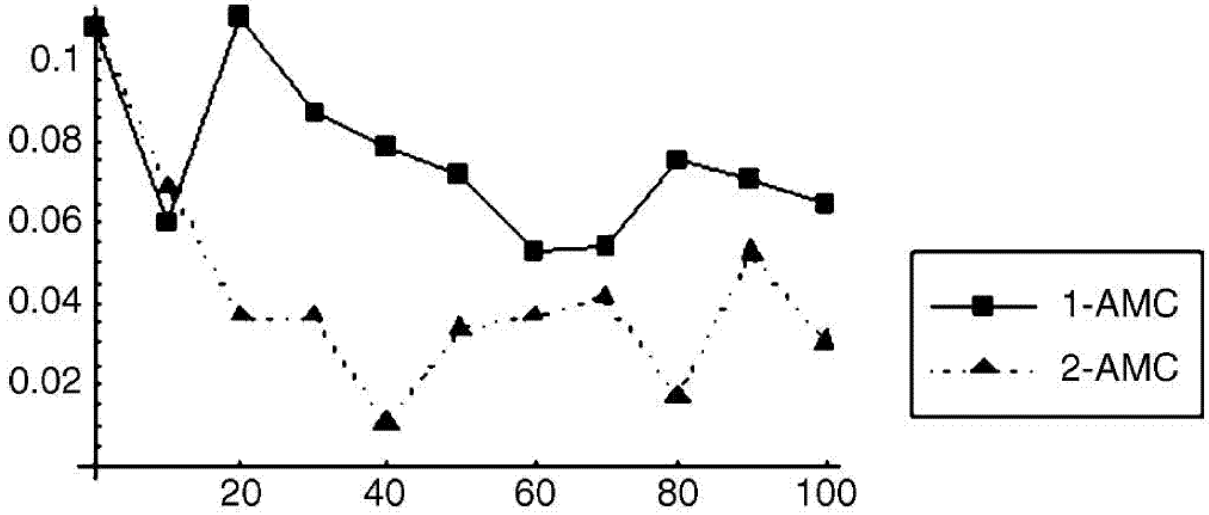


**Figure:** The absolute error in 1-AMC and 2-AMC algorithms during 100 iterations with almost the same number of sample points