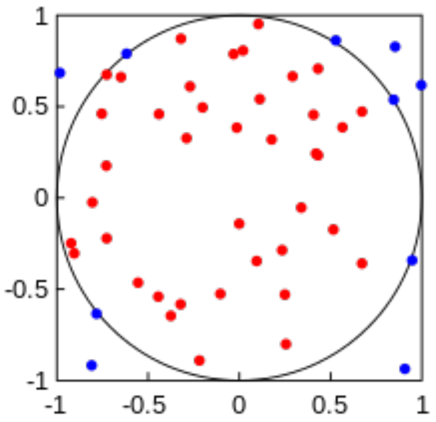
# Monte Carlo integration

From Wikipedia, the free encyclopedia



An illustration of Monte Carlo integration. In this example, the domain *D* is the inner circle and the domain E is the square. Because the square's area (4) can be easily calculated, the area of the circle ($\pi*1^2$) can be estimated by the ratio (0.8) of the points inside the circle (40) to the total number of points (50), yielding an approximation for the circle's area of 4*0.8 = 3.2 ≈ $\pi*1^2$.

In mathematics, Monte Carlo integration is a technique for numerical integration using random numbers. It is a particular Monte Carlo method that numerically computes a definite integral. While other algorithms usually evaluate the integrand at a regular grid,[1] Monte Carlo randomly choose points at which the integrand is evaluated.[2] This method is particularly useful for higher-dimensional integrals.[3]

There are different methods to perform a Monte Carlo integration, such as uniform sampling, stratified sampling, importance sampling, sequential Monte Carlo (a.k.a. particle filter), and mean field particle methods.

## Contents

- 1 Overview
  - 1.1 Example
  - 1.2 C example
  - 1.3 Wolfram Mathematica example
- 2 Recursive stratified sampling
  - 2.1 MISER Monte Carlo
- 3 Importance sampling
  - 3.1 VEGAS Monte Carlo
  - 3.2 Importance sampling algorithm
  - 3.3 Multiple and adaptive importance sampling
- 4 See also
- 5 Notes
- 6 References
- 7 External links

## Overview

In numerical integration, methods such as the trapezoidal rule use a deterministic approach. Monte Carlo integration, on the other hand, employs a non-deterministic approach: each realization provides a different outcome. In Monte Carlo, the final outcome is an approximation of the correct value with respective error bars, and the correct value is within those error bars.

The problem Monte Carlo integration addresses is the computation of a multidimensional definite integral

$$I = \int_{\Omega} f(\overline{\mathbf{x}})\, d\overline{\mathbf{x}}$$

where Ω, a subset of $R^m$, has volume

$$V = \int_{\Omega} d\overline{\mathbf{x}}$$

The naive Monte Carlo approach is to sample points uniformly on Ω:[4] given *N* uniform samples,

$$\overline{\mathbf{x}}_1, \cdots, \overline{\mathbf{x}}_N \in \Omega,$$

*I* can be approximated by

$$I \approx Q_N \equiv V \frac{1}{N} \sum_{i=1}^{N} f(\overline{\mathbf{x}}_i) = V \langle f \rangle.$$

This is because the law of large numbers ensures that

$$\lim_{N \to \infty} Q_N = I.$$

Given the estimation of *I* from $Q_N$, the error bars of $Q_N$ can be estimated by the sample variance using the unbiased estimate of the variance.

$$\mathrm{Var}(f) \equiv \sigma_N^2 = \frac{1}{N-1} \sum_{i=1}^{N} \left( f(\overline{\mathbf{x}}_i) - \langle f \rangle \right)^2.$$

which leads to

$$\mathrm{Var}(Q_N) = \frac{V^2}{N^2}\sum_{i=1}^{N}\mathrm{Var}(f) = V^2\frac{\mathrm{Var}(f)}{N} = V^2\frac{\sigma_N^2}{N}.$$

As long as the sequence

$$\left\{\sigma_1^2, \sigma_2^2, \sigma_3^2, \ldots\right\}$$

is bounded, this variance decreases asymptotically to zero as 1/$N$. The estimation of the error of $Q_N$ is thus

$$\delta Q_N \approx \sqrt{\mathrm{Var}(Q_N)} = V\frac{\sigma_N}{\sqrt{N}},$$

which decreases as $\frac{1}{\sqrt{N}}$. This is standard error of the mean multiplied with $V$. This result does not depend on the number of dimensions of the integral, which is the promised advantage of Monte Carlo integration against most deterministic methods that depend exponentially on the dimension.[5] It is important to notice that, like in deterministic methods, the estimate of the error is not a strict error bound; random sampling may not uncover all the important features of the integrand that can result in an underestimate of the error.

While the naive Monte Carlo works for simple examples, this is not the case in most problems. A large part of the Monte Carlo literature is dedicated in developing strategies to improve the error estimates. In particular, stratified sampling—dividing the region in sub-domains—, and importance sampling—sampling from non-uniform distributions—are two of such techniques.

## Example

A paradigmatic example of a Monte Carlo integration is the estimation of π.
Consider the function

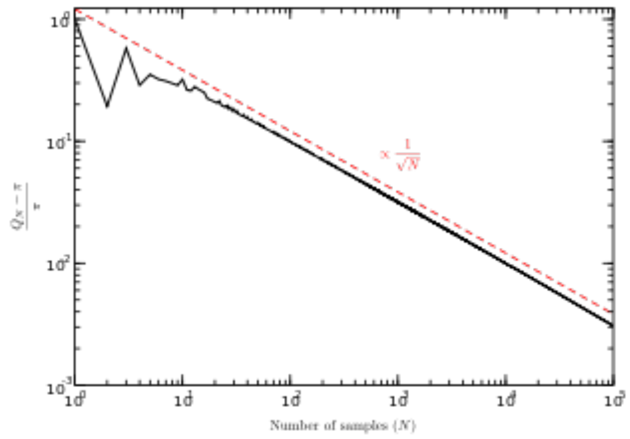
$$H\left(x,y\right) = \begin{cases} 1 & \text{if } x^2+y^2 \leq 1 \\ 0 & \text{else}\end{cases}$$

and the set Ω = [−1,1] × [−1,1] with $V$ = 4. Notice that

$$I_\pi = \int_\Omega H(x,y)dxdy = \pi.$$

Thus, a crude way of calculating the value of π with Monte Carlo integration is to pick $N$ random numbers on Ω and compute

$$Q_N = 4\frac{1}{N}\sum_{i=1}^{N} H(x_i, y_i)$$



Relative error as a function of the number of samples, showing the scaling $\frac{1}{\sqrt{N}}$

In the figure on the right, the relative error $\frac{Q_N-\pi}{\pi}$ is measured as a function of $N$, confirming the $\frac{1}{\sqrt{N}}$.

## C example

Keep in mind that a true random number generator should be used.

```
int i;
long double throws = 99999, circleDarts = 0, randX, randY, pi;

srand(time(NULL));
for (i = 0; i < throws; ++i) {
  randX = rand() / (double)RAND_MAX;
  randY = rand() / (double)RAND_MAX;
  if (1 > ((randX*randX) + (randY*randY))) ++circleDarts;
}
pi = 4 * (circleDarts/throws);
```

## Wolfram Mathematica example

The code below describes a process of integrating the function

$$f(x) = \frac{1}{1+\sinh(2x)\log(x)^2}$$

from $0.8 < x < 3$ using the Monte-Carlo method in Mathematica:

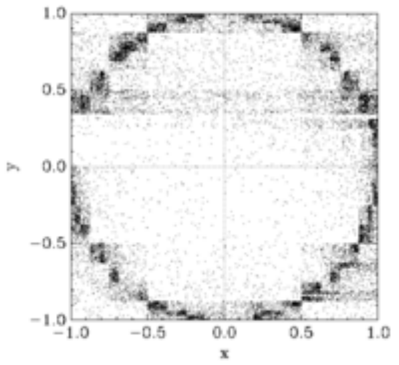
```
func[x_] := 1/(1 + Sinh[2*x]*(Log[x])^2);

(*Sample from truncated normal distribution to speed up convergence*)
Distrib[x_, average_, var_] :=   PDF[NormalDistribution[average, var], 1.1*x - 0.1];
n = 10;
RV = RandomVariate[TruncatedDistribution[{0.8, 3}, NormalDistribution[1, 0.399]], n];
Int = 1/n Total[func[RV]/Distrib[RV, 1, 0.399]]*Integrate[Distrib[x, 1, 0.399], {x, 0.8, 3}]

NIntegrate[func[x], {x, 0.8, 3}] (*Compare with real answer*)
```

## Recursive stratified sampling

Recursive stratified sampling is a generalization of one-dimensional adaptive quadratures to multi-dimensional integrals. On each recursion step the integral and the error are estimated using a plain Monte Carlo algorithm. If the error estimate is larger than the required accuracy the integration volume is divided into sub-volumes and the procedure is recursively applied to sub-volumes.

The ordinary 'dividing by two' strategy does not work for multi-dimensions as the number of sub-volumes grows far too quickly to keep track. Instead one estimates along which dimension a subdivision should bring the most dividends and only subdivides the volume along this dimension.

The stratified sampling algorithm concentrates the sampling points in the regions where the variance of the function is largest thus reducing the grand variance and making the sampling more effective, as shown on the illustration.



An illustration of Recursive Stratified Sampling. In this example, the function:

$$f(x,y) = \begin{cases} 1 & x^2+y^2<1 \\ 0 & x^2+y^2 \geq 1 \end{cases}$$

from the above illustration was integrated within a unit square using the suggested algorithm. The sampled points were recorded and plotted. Clearly stratified sampling algorithm concentrates the points in the regions where the variation of the function is largest.

The popular MISER routine implements a similar algorithm.

### MISER Monte Carlo

The MISER algorithm is based on recursive stratified sampling. This technique aims to reduce the overall integration error by concentrating integration points in the regions of highest variance.[6]

The idea of stratified sampling begins with the observation that for two disjoint regions *a* and *b* with Monte Carlo estimates of the integral $E_a(f)$ and $E_b(f)$ and variances $\sigma_a^2(f)$ and $\sigma_b^2(f)$, the variance Var(*f*) of the combined estimate

$$E(f) = \tfrac{1}{2}\left(E_a(f) + E_b(f)\right)$$

is given by,

$$\mathrm{Var}(f) = \frac{\sigma_a^2(f)}{4N_a} + \frac{\sigma_b^2(f)}{4N_b}$$

It can be shown that this variance is minimized by distributing the points such that,

$$\frac{N_a}{N_a+N_b} = \frac{\sigma_a}{\sigma_a+\sigma_b}$$

Hence the smallest error estimate is obtained by allocating sample points in proportion to the standard deviation of the function in each sub-region.

The MISER algorithm proceeds by bisecting the integration region along one coordinate axis to give two sub-regions at each step. The direction is chosen by examining all *d* possible bisections and selecting the one which will minimize the combined variance of the two sub-regions. The variance in the sub-regions is estimated by sampling with a fraction of the total number of points available to the current step. The same procedure is then repeated recursively for each of the two half-spaces from the best bisection. The remaining sample points are allocated to the sub-regions using the formula for $N_a$ and $N_b$. This recursive allocation of integration points continues down to a user-specified depth where each sub-region is integrated using a plain Monte Carlo estimate. These individual values and their error estimates are then combined upwards to give an overall result and an estimate of its error.

## Importance sampling

### VEGAS Monte Carlo

The VEGAS algorithm takes advantage of the information stored during the sampling, and uses it and importance sampling to efficiently estimate the integral $I$. It samples points from the probability distribution described by the function $|f|$ so that the points are concentrated in the regions that make the largest contribution to the integral.

In general, if the Monte Carlo integral of *f* is sampled with points distributed according to a probability distribution described by the function *g*, we obtain an estimate:

$$E_g(f;N) = E\left(\frac{f}{g};N\right)$$

with a corresponding variance,

$$\mathrm{Var}_g(f;N) = \mathrm{Var}\left(\frac{f}{g};N\right)$$

If the probability distribution is chosen as

$$g = \frac{|f|}{I(|f|)}$$

then it can be shown that the variance $V_g(f;N)$ vanishes, and the error in the estimate will be zero. In practice it is not possible to sample from the exact distribution *g* for an arbitrary function, so importance sampling algorithms aim to produce efficient approximations to the desired distribution.

The VEGAS algorithm approximates the exact distribution by making a number of passes over the integration region which creates the histogram of the function *f*. Each histogram is used to define a sampling distribution for the next pass. Asymptotically this procedure converges to the desired distribution.[7] In order to avoid the number of histogram bins growing like $K^d$, the probability distribution is approximated by a separable function:

$$g(x_1, x_2, \ldots) = g_1(x_1)g_2(x_2)\ldots$$

so that the number of bins required is only *Kd*. This is equivalent to locating the peaks of the function from the projections of the integrand onto the coordinate axes. The efficiency of VEGAS depends on the validity of this assumption. It is most efficient when the peaks of the integrand are well-localized. If an integrand can be rewritten in a form which is approximately separable this will increase the efficiency of integration with VEGAS.

VEGAS incorporates a number of additional features, and combines both stratified sampling and importance sampling.[7] The integration region is divided into a number of "boxes", with each box getting a fixed number of points (the goal is 2). Each box can then have a fractional number of bins, but if bins/box is less than two, Vegas switches to a kind variance reduction (rather than importance sampling).

This routines uses the VEGAS Monte Carlo algorithm to integrate the function *f* over the *dim*-dimensional hypercubic region defined by the lower and upper limits in the arrays *xl* and *xu*, each of size *dim*. The integration uses a fixed number of function calls. The result and its error estimate are based on a weighted average of independent samples.

The VEGAS algorithm computes a number of independent estimates of the integral internally, according to the iterations parameter described below, and returns their weighted average. Random sampling of the integrand can occasionally produce an estimate where the error is zero, particularly if the function is constant in some regions. An estimate with zero error causes the weighted average to break down and must be handled separately.

### Importance sampling algorithm

Importance sampling provides a very important tool to perform Monte-Carlo integration.[3][8] The main result of importance sampling to this method is that the uniform sampling of $\overline{\mathbf{x}}$ is a particular case of a more generic choice, on which the samples are drawn from any distribution $p(\overline{\mathbf{x}})$. The idea is that $p(\overline{\mathbf{x}})$ can be chosen to decrease the variance of the measurement $Q_N$.

Consider the following example where one would like to numerically integrate a gaussian function, centered at 0, with σ = 1, from −1000 to 1000. Naturally, if the samples are drawn uniformly on the interval [−1000, 1000], only a very small part of them would be significant to the integral. This can be improved by choosing a different distribution from where the samples are chosen, for instance by sampling according to a gaussian distribution centered at 0, with σ = 1. Of course the "right" choice strongly depends on the integrand.

Formally, given a set of samples chosen from a distribution

$$p(\overline{\mathbf{x}}): \qquad \overline{\mathbf{x}}_1, \cdots, \overline{\mathbf{x}}_N \in V,$$

the estimator for $I$ is given by[3]

$$Q_N \equiv \frac{1}{N}\sum_{i=1}^{N}\frac{f(\overline{\mathbf{x}}_i)}{p(\overline{\mathbf{x}}_i)}$$

Intuitively, this says that if we pick a particular sample twice as much as other samples, we weight it half as much as the other samples. This estimator is naturally valid for uniform sampling, the case where $p(\overline{\mathbf{x}})$ is constant.

The Metropolis-Hastings algorithm is one of the most used algorithms to generate $\overline{\mathbf{x}}$ from $p(\overline{\mathbf{x}})$,[3] thus providing an efficient way of computing integrals.

### Multiple and adaptive importance sampling

When different proposal distributions, $p_n(\overline{\mathbf{x}})$ , $n = 1, \ldots, N,$ are jointly used for drawing the samples $\overline{\mathbf{x}}_1, \cdots, \overline{\mathbf{x}}_N \in V,$ different proper weighting functions can be employed (e.g., see [9][10][11]). In an adaptive setting, the proposal distributions, $p_{n,t}(\overline{\mathbf{x}})$ , $n = 1, \ldots, N,$ and $t = 1, \ldots, T,$ are updated each iteration $t$ of the adaptive importance sampling algorithm. Hence, since a population of proposal densities is used, several suitable combinations of sampling and weighting schemes can be employed.[12][13][14][15][16]

## See also

- Auxiliary field Monte Carlo
- Monte Carlo method in statistical physics
- Monte Carlo method
- Variance reduction

## Notes

1. Press et al, 2007, Chap. 4.
2. Press et al, 2007, Chap. 7.
3. Newman, 1999, Chap. 2.
4. Newman, 1999, Chap. 1.
5. Press et al, 2007
6. Press, 1990, pp 190-195.
7. Lepage, 1978
8. Kroese, D. P.; Taimre, T.; Botev, Z. I. (2011). *Handbook of Monte Carlo Methods*. John Wiley & Sons.
9. Veach, Eric; Guibas, Leonidas J. (1995-01-01). "Optimally Combining Sampling Techniques for Monte Carlo Rendering" (http://doi.acm.org/10.1145/218380.218498). *Proceedings of the 22Nd Annual Conference on Computer Graphics and Interactive Techniques*. SIGGRAPH '95. New York, NY, USA: ACM: 419–428. ISBN 0-89791-701-4. doi:10.1145/218380.218498 (https://doi.org/10.1145%2F218380.218498).
10. Owen, Art; Associate, Yi Zhou (2000-03-01). "Safe and Effective Importance Sampling" (http://www.tandfonline.com/doi/abs/10.1080/01621459.2000.10473909). *Journal of the American Statistical Association*. 95 (449): 135–143. ISSN 0162-1459 (https://www.worldcat.org/issn/0162-1459). doi:10.1080/01621459.2000.10473909 (https://doi.org/10.1080%2F01621459.2000.10473909).
11. Elvira, V.; Martino, L.; Luengo, D.; Bugallo, M.F. (2015-10-01). "Efficient Multiple Importance Sampling Estimators" (http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=7105865). *IEEE Signal Processing Letters*. 22 (10): 1757–1761. ISSN 1070-9908 (https://www.worldcat.org/issn/1070-9908). doi:10.1109/LSP.2015.2432078 (https://doi.org/10.1109%2FLSP.2015.2432078).
12. Cappé, O.; Guillin, A.; Marin, J. M.; Robert, C. P. (2004-12-01). "Population Monte Carlo" (http://dx.doi.org/10.1198/106186004X12803). *Journal of Computational and Graphical Statistics*. 13 (4): 907–929. ISSN 1061-8600 (https://www.worldcat.org/issn/1061-8600). doi:10.1198/106186004X12803 (https://doi.org/10.1198%2F106186004X12803).
13. Cappé, Olivier; Douc, Randal; Guillin, Arnaud; Marin, Jean-Michel; Robert, Christian P. (2008-04-25). "Adaptive importance sampling in general mixture classes" (http://link.springer.com/article/10.1007/s11222-008-9059-x). *Statistics and Computing*. 18 (4): 447–459. ISSN 0960-3174 (https://www.worldcat.org/issn/0960-3174). doi:10.1007/s11222-008-9059-x (https://doi.org/10.1007%2Fs11222-008-9059-x).
14. Cornuet, Jean-Marie; Marin, Jean-Michel; Mira, Antonietta; Robert, Christian P. (2012-12-01). "Adaptive Multiple Importance Sampling" (http://onlinelibrary.wiley.com/doi/10.1111/j.1467-9469.2011.00756.x/abstract). *Scandinavian Journal of Statistics*. 39 (4): 798–812. ISSN 1467-9469 (https://www.worldcat.org/issn/1467-9469). doi:10.1111/j.1467-9469.2011.00756.x (https://doi.org/10.1111%2Fj.1467-9469.2011.00756.x).
15. Martino, L.; Elvira, V.; Luengo, D.; Corander, J. (2015-08-01). "An Adaptive Population Importance Sampler: Learning From Uncertainty" (http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=7117437). *IEEE Transactions on Signal Processing*. 63 (16): 4422–4437. ISSN 1053-587X (https://www.worldcat.org/issn/1053-587X). doi:10.1109/TSP.2015.2440215 (https://doi.org/10.1109%2FTSP.2015.2440215).
16. Bugallo, Mónica F.; Martino, Luca; Corander, Jukka (2015-12-01). "Adaptive importance sampling in signal processing" (http://www.sciencedirect.com/science/article/pii/S1051200415001864). *Digital Signal Processing*. Special Issue in Honour of William J. (Bill) Fitzgerald. 47: 36–49. doi:10.1016/j.dsp.2015.05.014 (https://doi.org/10.1016%2Fj.dsp.2015.05.014).

## References

- Caflisch, R. E. (1998). "Monte Carlo and quasi-Monte Carlo methods". *Acta Numerica*. Cambridge University Press. 7: 1–49. doi:10.1017/S0962492900002804 (https://doi.org/10.1017%2FS0962492900002804).
- Weinzierl, S. (2000). "Introduction to Monte Carlo methods". *ArXiv preprint*. arXiv:hep-ph/0006269 (https://arxiv.org/abs/hep-ph/0006269).
- Press, W. H.; Farrar, G. R. (1990). "Recursive Stratified Sampling for Multidimensional Monte Carlo Integration". *Computers in Physics*. 4. doi:10.1063/1.4822899 (https://doi.org/10.1063%2F1.4822899).
- Lepage, G. P. (1978). "A New Algorithm for Adaptive Multidimensional Integration". *Journal of Computational Physics*. 27 (2): 192–203. doi:10.1016/0021-9991(78)90004-9 (https://doi.org/10.1016%2F0021-9991%2878%2990004-9).
- Lepage, G. P. (1980). "VEGAS: An Adaptive Multi-dimensional Integration Program". *Cornell preprint CLNS 80-447*.
- Hammersley, J. M.; Handscomb, D. C. (1964). *Monte Carlo Methods*. Methuen. ISBN 0-416-52340-4.
- Press, WH; Teukolsky, SA; Vetterling, WT; Flannery, BP (2007). *Numerical Recipes: The Art of Scientific Computing* (3rd ed.). New York: Cambridge University Press. ISBN 978-0-521-88068-8.
- Newman, MEJ; Barkema, GT (1999). *Monte Carlo Methods in Statistical Physics*. Clarendon Press.
- Robert, CP; Casella, G (2004). *Monte Carlo Statistical Methods* (2nd ed.). Springer. ISBN 978-1-4419-1939-7.

## External links

- Café math : Monte Carlo Integration (http://www.cafemath.fr/mathblog/article.php?page=MonteCarlo.php) : A blog article describing Monte Carlo integration (principle, hypothesis, confidence interval)

Retrieved from "https://en.wikipedia.org/w/index.php?title=Monte_Carlo_integration&oldid=795264461"

- This page was last edited on 13 August 2017, at 03:08.
-