Monte Carlo Methods for Uncertainty Quantification

Mike Giles

Mathematical Institute, University of Oxford

KU Leuven Summer School on Uncertainty Quantification

May 30–31, 2013

Lecture outline

Lecture 1: Monte Carlo basics

- random number generation
- Monte Carlo estimation
- Law of Large Numbers and confidence interval
- basic mean/variance manipulations
- antithetic sampling
- control variate

Lecture 2: Variance reduction

- importance sampling
- stratified sampling
- Latin Hypercube
- randomised quasi-Monte Carlo

Lecture outline

Lecture 3: financial applications

- financial models
- approximating SDEs
- weak and strong convergence
- mean square error decomposition
- multilevel Monte Carlo

Lecture 4: PDE applications

- PDEs with uncertainty
- examples
- multilevel Monte Carlo

Random Number Generation

Monte Carlo simulation starts with random number generation, usually split into 2 stages:

- generation of independent uniform $(0, 1)$ random variables
- conversion into random variables with a particular distribution (e.g. Normal)

Very important: never write your own generator, always use a well validated generator from a reputable source

- Matlab
- NAG
- Intel MKL
- AMD ACML

Uniform Random Variables

Pseudo-random number generators use a deterministic (i.e. repeatable) algorithm to generate a sequence of (apparently) random numbers on $(0,1)$ interval.

What defines a good generator?

- a long period – how long it takes before the sequence repeats itself
  2^{32} is not enough – need at least 2^{40}
- various statistical tests to measure "randomness"
  well validated software will have gone through these checks
- trivially-parallel Monte Carlo simulation on a compute cluster requires the ability to "skip-ahead" to an arbitrary starting point in the sequence

  first computer gets first 10^6 numbers
  second computer gets second 10^6 numbers, etc

Uniform Random Variables

For information see

- Intel MKL information
  `www.intel.com/cd/software/products/asmo-na/eng/266864.htm`
- NAG library information
  `www.nag.co.uk/numeric/CL/nagdoc_cl08/pdf/G05/g05_conts.pdf`
- Matlab information
  `www.mathworks.com/moler/random.pdf`
- Wikipedia information
  `en.wikipedia.org/wiki/Random_number_generation`
  `en.wikipedia.org/wiki/List_of_random_number_generators`
  `en.wikipedia.org/wiki/Mersenne_Twister`

Normal Random Variables

$N(0,1)$ Normal random variables (mean 0, variance 1) have the probability distribution

$$p(x) \;=\; \phi(x) \;\equiv\; \frac{1}{\sqrt{2\pi}}\,\exp(-\tfrac{1}{2}x^2)$$

The Box-Muller method takes two independent uniform $(0,1)$ random numbers y_1, y_2, and defines

$$\begin{aligned} x_1 &= \sqrt{-2\log(y_1)}\,\cos(2\pi y_2) \\ x_2 &= \sqrt{-2\log(y_1)}\,\sin(2\pi y_2) \end{aligned}$$

It can be proved that x_1 and x_2 are $N(0,1)$ random variables, and independent:

$$p_{\text{joint}}(x_1,x_2) = p(x_1)\;p(x_2)$$

Inverse CDF

A more flexible alternative uses the cumulative distribution function $CDF(x)$ for a random variable X, defined as

$$CDF(x) \;=\; \mathbb{P}(X < x)$$

If Y is a uniform $(0,1)$ random variable, then can define X by

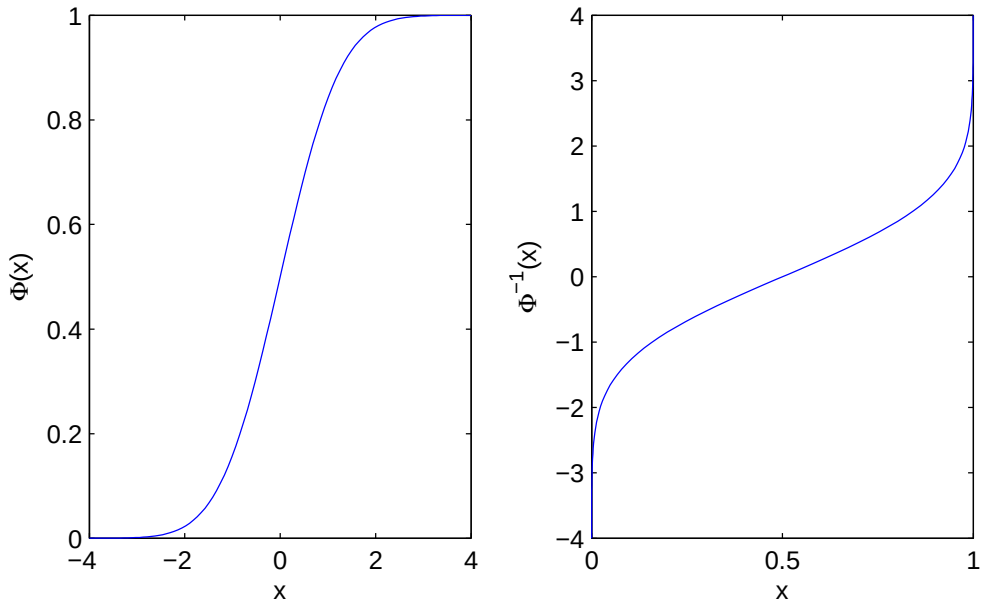
$$X = CDF^{-1}(Y).$$

For $N(0,1)$ Normal random variables,

$$CDF(x) \;=\; \Phi(x) \;\equiv\; \int_{-\infty}^{x} \phi(s)\,\mathrm{d}s \;=\; \frac{1}{\sqrt{2\pi}}\int_{-\infty}^{x} \exp\left(-\tfrac{1}{2}s^2\right)\,\mathrm{d}s$$

$\Phi^{-1}(y)$ is approximated in software in a very similar way to the implementation of $\cos, \sin, \log$.

Normal Random Variables



Normal Random Variables

Some useful weblinks:

- `home.online.no/~pjacklam/notes/invnorm/`
  code for Φ^{-1} function in many different languages
- `lib.stat.cmu.edu/apstat/241/`
  single and double precision code in FORTRAN
- `en.wikipedia.org/wiki/Normal_distribution`
  Wikipedia definition of Φ matches mine
- `mathworld.wolfram.com/NormalDistribution.html`
  `mathworld.wolfram.com/DistributionFunction.html`
  Good Mathworld items, but their definition of Φ is slightly different; they call the cumulative distribution function $D(x)$.

Normal Random Variables

The Normal CDF $\Phi(x)$ is related to the error function $\mathrm{erf}(x)$ through

$$\Phi(x) = \tfrac{1}{2} + \tfrac{1}{2}\,\mathrm{erf}(x/\sqrt{2}) \quad \Longrightarrow \quad \Phi^{-1}(y) = \sqrt{2}\,\mathrm{erf}^{-1}(2y-1)$$

This is the function I use in Matlab:

```
% x = ncfinv(y)
%
% inverse Normal CDF

function x = ncfinv(y)

x = sqrt(2)*erfinv(2*y-1);
```


Correlated Normal Random Variables

We often need a vector y of Normally distributed variables with a prescribed covariance matrix, so that $\mathbb{E}[y\,y^T] = \Sigma$.

Suppose x is a vector of independent $N(0,1)$ variables, and define $y = L\,x$.

Each element of y is Normally distributed, $\mathbb{E}[y] = L\,\mathbb{E}[x] = 0$, and

$$\mathbb{E}[y\,y^T] = \mathbb{E}[L\,x\,x^T L^T] = L\;\mathbb{E}[x\,x^T]\;L^T = L\,L^T$$

since $\mathbb{E}[x\,x^T] = I$ because

- elements of x are independent $\Longrightarrow \mathbb{E}[x_i\,x_j] = 0$ for $i \neq j$
- elements of x have unit variance $\Longrightarrow \mathbb{E}[x_i^2] = 1$

Hence choose L so that $LL^T = \Sigma$

Correlated Normal Random Variables

One choice is a Cholesky factorisation in which L is lower-triangular.

Alternatively, if Σ has eigenvalues $\lambda_i \geq 0$, and orthonormal eigenvectors u_i, so that

$$\Sigma\, u_i = \lambda_i\, u_i, \quad \Longrightarrow \quad \Sigma\, U = U\, \Lambda$$

then

$$\Sigma = U\, \Lambda\, U^T = L\, L^T$$

where

$$L = U\, \Lambda^{1/2}.$$

This is the PCA decomposition; it is no better than the Cholesky decomposition for standard Monte Carlo simulation, but is often better for stratified sampling and quasi-Monte Carlo methods.

Expectation and Integration

If X is a random variable uniformly distributed on $[0,1]$ then the expectation of a function $f(X)$ is equal to its integral:

$$\overline{f} = \mathbb{E}[f(X)] = I[f] = \int_0^1 f(x)\,\mathrm{d}x.$$

The generalisation to a d-dimensional "cube" $I^d = [0,1]^d$, is

$$\overline{f} = \mathbb{E}[f(X)] = I[f] = \int_{I^d} f(x)\,\mathrm{d}x.$$

Thus the problem of finding expectations is directly connected to the problem of numerical quadrature (integration), often in very large dimensions.

Expectation and Integration

Suppose we have a sequence X_n of independent samples from the uniform distribution.

An approximation to the expectation/integral is given by

$$I_N[f] = N^{-1} \sum_{n=1}^{N} f(x_n).$$

Two key features:

- Unbiased: $\mathbb{E}\left[I_N[f]\right] = I[f]$
- Convergent: $\lim_{N\to\infty} I_N[f] = I[f]$

Expectation and Integration

In general, define

- error $\quad \varepsilon_N(f) = I[f] - I_N[f]$
- bias $= \mathbb{E}[\varepsilon_N(f)]$
- RMSE, "root-mean-square-error" $= \sqrt{\mathbb{E}[(\varepsilon_N(f))^2]}$

The Central Limit Theorem proves (roughly speaking) that for large N

$$\varepsilon_N(f) \sim \sigma\, N^{-1/2}\, Z$$

with Z a $N(0,1)$ random variable and σ^2 the variance of f:

$$\sigma^2 = \mathbb{E}[(f - \overline{f})^2] = \int_{I^d} \left(f(x) - \overline{f}\right)^2 \,\mathrm{d}x.$$

Expectation and Integration

More precisely, provided σ is finite, then as $N \longrightarrow \infty$,

$$\text{CDF}(N^{1/2}\sigma^{-1}\varepsilon_N) \longrightarrow \text{CDF}(Z)$$

so that

$$\mathbb{P}\left[N^{1/2}\sigma^{-1}\varepsilon_N < s\right] \longrightarrow \mathbb{P}[Z < s] = \Phi(s)$$

and

$$\mathbb{P}\left[\left|N^{1/2}\sigma^{-1}\varepsilon_N\right| > s\right] \longrightarrow \mathbb{P}[|Z| > s] = 2\,\Phi(-s)$$

$$\mathbb{P}\left[\left|N^{1/2}\sigma^{-1}\varepsilon_N\right| < s\right] \longrightarrow \mathbb{P}[|Z| < s] = 1 - 2\,\Phi(-s)$$

Expectation and Integration

Given N samples, the empirical variance is

$$\widetilde{\sigma}^2 = N^{-1}\sum_{n=1}^{N}(f(x_n) - I_N)^2 = I_N^{(2)} - (I_N)^2$$

where

$$I_N = N^{-1}\sum_{n=1}^{N} f(x_n), \qquad I_N^{(2)} = N^{-1}\sum_{n=1}^{N}(f(x_n))^2$$

$\widetilde{\sigma}^2$ is a slightly biased estimator for σ^2; an unbiased estimator is

$$\widehat{\sigma}^2 = (N-1)^{-1}\sum_{n=1}^{N}(f(x_n) - I_N)^2 = \frac{N}{N-1}\left(I_N^{(2)} - (I_N)^2\right)$$

Expectation and Integration

How many samples do we need for an accuracy of $\overline{\varepsilon}$ with probability c?

Since

$$\mathbb{P}\left[N^{1/2}\sigma^{-1}|\varepsilon| < s\right] \approx 1 - 2\,\Phi(-s),$$

define s so

$$1 - 2\,\Phi(-s) = c \iff s = -\Phi^{-1}((1-c)/2)$$

c	0.683	0.9545	0.9973	0.99994
s	1.0	2.0	3.0	4.0

Then $|\varepsilon| < N^{-1/2}\,\sigma\,s$ with probability c, so to get $|\varepsilon| < \overline{\varepsilon}$ we can put

$$N^{-1/2}\,\widehat{\sigma}\,s(c) = \overline{\varepsilon} \implies N = \left(\frac{\widehat{\sigma}\,s(c)}{\overline{\varepsilon}}\right)^2.$$

Note: twice as much accuracy requires 4 times as many samples.

Expectation and Integration

How does Monte Carlo integration compare to grid based methods for d-dimensional integration?

MC error is proportional to $N^{-1/2}$ independent of the dimension.

If the integrand is sufficiently smooth, trapezoidal integration with $M = N^{1/d}$ points in each direction has

$$\text{Error} \propto M^{-2} = N^{-2/d}$$

This scales better than MC for $d < 4$, but worse for $d > 4$.
i.e. MC is better at handling high dimensional problems.

Finance Applications

Geometric Brownian motion for a single asset:

$$S_T = S_0 \ \exp\left((r - \tfrac{1}{2}\sigma^2)T + \sigma W_T\right)$$

W_T is $N(0,T)$ random variable, so can put

$$W_T = \sqrt{T}\ Y = \sqrt{T}\ \Phi^{-1}(U)$$

where Y is a $N(0,1)$ r.v. and U is a uniform $(0,1)$ r.v.

We are then interested in the price of financial options which can be expressed as

$$V = \mathbb{E}\left[f(S(T))\right] = \int_0^1 f(S(T))\ \mathrm{d}U,$$

for some "payoff" function $f(S)$.

Finance Applications

For the European call option,

$$f(S) = \exp(-rT)\ (S-K)^+$$

while for the European put option

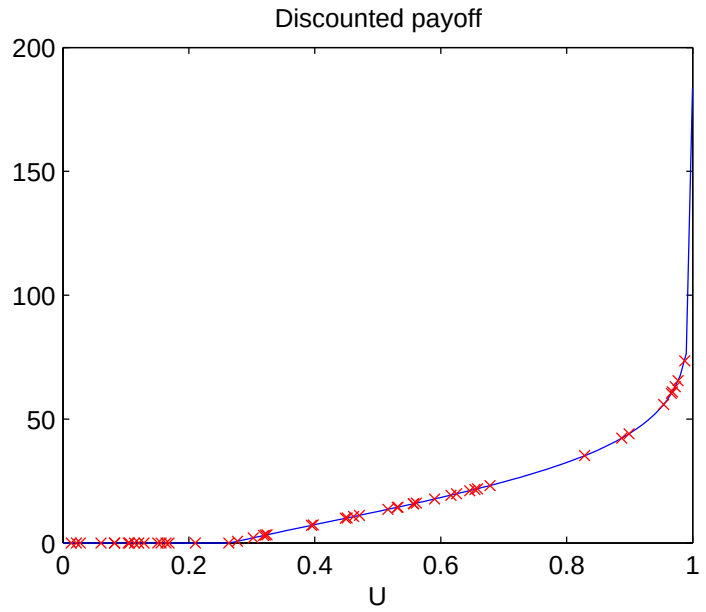
$$f(S) = \exp(-rT)\ (K-S)^+$$

where K is the strike price, and $(y)^+ \equiv \max(0,y)$.

For numerical experiments we will consider a European call with $r=0.05, \quad \sigma=0.2, \quad T=1, \quad S_0=110, \quad K=100.$

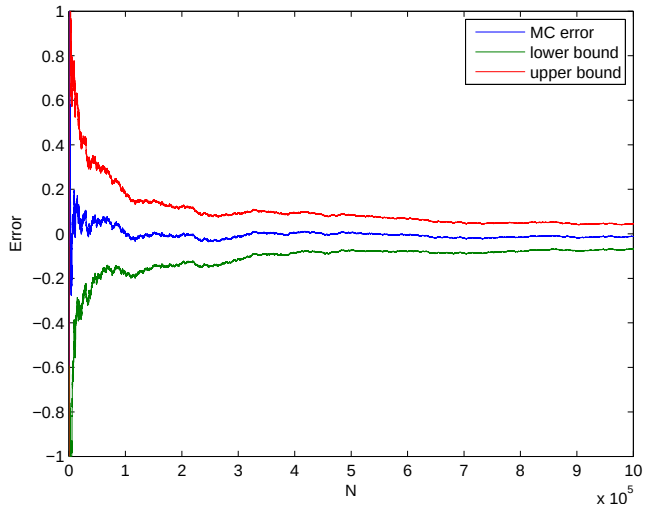
The analytic value is known for comparison.

Finance Applications



Finance Applications

MC calculation with up to 10^6 paths; true value = 17.663



Finance Applications

The upper and lower bounds are given by

$$\text{Mean} \pm \frac{3\,\widetilde{\sigma}}{\sqrt{N}},$$

so more than a 99.7% probability that the true value lies within these bounds.

Finance Applications

MATLAB code:

```
r=0.05;  sig=0.2;  T=1;  S0=110;  K=100;
N = 1:1000000;
U = rand(1,max(N));       % uniform random variable
Y = ncfinv(U);            % inverts Normal cum. fn.
S = S0*exp((r-sig^2/2)*T + sig*sqrt(T)*Y);
F = exp(-r*T)*max(0,S-K);

sum1 = cumsum(F);         % cumulative summation of
sum2 = cumsum(F.^2);      % payoff and its square
val  = sum1./N;
rms  = sqrt(sum2./N - val.^2);
```


Finance Applications

```
err = european_call(r,sig,T,S0,K,'value') - val;

plot(N,err,                    ...
     N,err-3*rms./sqrt(N),     ...
     N,err+3*rms./sqrt(N))
axis([0 length(N) -1 1])
xlabel('N'); ylabel('Error')
legend('MC error','lower bound','upper bound')
```


Finance Applications

New application: a European call based on average of M stocks which are correlated.

$$S_i(T) = S_i(0)\ \exp\left((r - \tfrac{1}{2}\sigma_i^2)T + \sigma_i W_i(T)\right)$$

If $\sigma_i W_i(T)$ has covariance matrix Σ, then use Cholesky factorisation $LL^T = \Sigma$ to get

$$S_i(T) = S_i(0)\ \exp\left((r - \tfrac{1}{2}\sigma_i^2)T + \sum_j L_{ij} Y_j\right)$$

where Y_j are independent $N(0,1)$ random variables.

Each Y_i can in turn be expressed as $\Phi^{-1}(U_i)$ where the U_i are uniformly, and independently, distributed on $[0,1]$.

Finance Applications

The payoff is

$$f = \exp(-rT)\left(\frac{1}{M}\sum_i S_i - K\right)^+$$

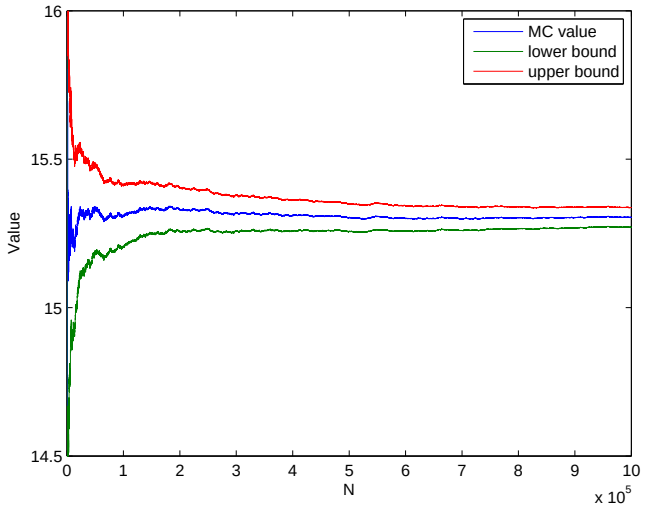
and so the expectation can be written as the M-dimensional integral

$$\int_{I^M} f(U)\,\mathrm{d}U.$$

This is a good example for Monte Carlo simulation – cost scales linearly with the number of stocks, whereas it would be exponential for grid-based numerical integration.

Finance Applications

MC calculation with up to 10^6 paths



Finance Applications

MATLAB code:

```
r=0.05;  sig=0.2;  T=1;  S0=110;  K=100;

Sigma = sig^2*T*( eye(5) + 0.1*(ones(5)-eye(5)));
L     = chol(Sigma,'lower');

N = 1:1000000;
U = rand(5,max(N));       % uniform random variable
Y = ncfinv(U);            % inverts Normal cum. fn.
S = S0*exp((r-sig^2/2)*T + L*Y);
F = exp(-r*T)*max(0,sum(S,1)/5-K);

sum1 = cumsum(F);         % cumulative summation of
sum2 = cumsum(F.^2);      % payoff and its square
val  = sum1./N;
rms  = sqrt(sum2./N - val.^2);
```


Summary so far

- Monte Carlo quadrature is straightforward and robust
- confidence bounds can be obtained as part of the calculation
- can calculate the number of samples N needed for chosen accuracy
- much more efficient than grid-based methods for high dimensions
- accuracy $= O(N^{-1/2})$, CPU time $= O(N)$

  $\Longrightarrow$ accuracy $= O(\text{CPU time}^{-1/2})$

  $\Longrightarrow$ CPU time $= O(\text{accuracy}^{-2})$

- the key now is to reduce number of samples required by reducing the variance – antithetic variables and control variates in this lecture

Elementary Manipulations

If X_1 and X_2 are independent continuous random variables, then

$$p_{\text{joint}}(x_1, x_2) = p_1(x_1)\; p_2(x_2)$$

and hence

$$\begin{aligned}
\mathbb{E}[f_1(X_1)\; f_2(X_2)] &= \int\!\!\int f_1(x_1)\; f_2(x_2)\; p_{\text{joint}}(x_1, x_2)\; \mathrm{d}x_1\; \mathrm{d}x_2 \\
&= \int\!\!\int f_1(x_1)\; f_2(x_2)\; p_1(x_1)\; p_2(x_2)\; \mathrm{d}x_1\; \mathrm{d}x_2 \\
&= \left(\int f_1(x_1)\; p_1(x_1)\; \mathrm{d}x_1\right)\left(\int f_2(x_2)\; p_2(x_2)\; \mathrm{d}x_2\right) \\
&= \mathbb{E}[f_1(X_1)]\;\; \mathbb{E}[f_2(X_2)]
\end{aligned}$$

and in particular

$$\begin{aligned}
\mathrm{Cov}[X_1, X_2] &\equiv \mathbb{E}\Big[\,(X_1 - \mathbb{E}[X_1])\,(X_2 - \mathbb{E}[X_2])\Big] \\
&= \mathbb{E}[X_1 - \mathbb{E}[X_1]\,]\;\; \mathbb{E}[X_2 - \mathbb{E}[X_2]\,] \;=\; 0
\end{aligned}$$

Elementary Manipulations

If a, b are random variables, and λ, μ are constants, then

$$\begin{aligned}
\mathbb{E}[a + \mu] &= \mathbb{E}[a] + \mu \\
\mathbb{V}[a + \mu] &= \mathbb{V}[a] \\
\mathbb{E}[\lambda\, a] &= \lambda\; \mathbb{E}[a] \\
\mathbb{V}[\lambda\, a] &= \lambda^2\, \mathbb{V}[a] \\
\mathbb{E}[a + b] &= \mathbb{E}[a] + \mathbb{E}[b]
\end{aligned}$$

where $$\mathbb{V}[a] \;\equiv\; \mathbb{E}\Big[\,(a - \mathbb{E}[a])^2\Big] \;=\; \mathbb{E}\left[a^2\right] - (\mathbb{E}[a])^2$$

Elementary Manipulations

In addition,

$$\mathbb{V}[a+b] = \mathbb{V}[a] + 2\,\mathrm{Cov}[a, b] + \mathbb{V}[b]$$

where $$\mathrm{Cov}[a, b] \equiv \mathbb{E}\Big[\,(a - \mathbb{E}[a])\,(b - \mathbb{E}[b])\Big]$$

Since

$$|\mathrm{Cov}[a, b]| \;\le\; \sqrt{\mathbb{V}[a]\;\mathbb{V}[b]}$$

it follows that

$$\begin{aligned}
\mathbb{V}[a+b] &\le \left(\sqrt{\mathbb{V}[a]} + \sqrt{\mathbb{V}[b]}\right)^2 \\
\Longrightarrow\quad \sqrt{\mathbb{V}[a+b]} &\le \sqrt{\mathbb{V}[a]} + \sqrt{\mathbb{V}[b]}
\end{aligned}$$

If a, b are independent then $\mathbb{V}[a+b] = \mathbb{V}[a] + \mathbb{V}[b]$, and more generally the variance of a sum of independents is equal to the sum of their variances.

Antithetic variables

The simple estimator for $\mathbb{E}[f(X)]$ from the last lecture has the form

$$N^{-1} \sum_i f(X^{(i)})$$

where $X^{(i)}$ is the i^{th} independent sample of the random variable X.

If X has a symmetric probability distribution, $-X$ is just as likely. Antithetic estimator replaces $f(X^{(i)})$ by

$$\overline{f}^{(i)} = \tfrac{1}{2}\left(f(X^{(i)}) + f(-X^{(i)})\right)$$

Clearly still unbiased since

$$\mathbb{E}\left[\,\overline{f}\,\right] \;=\; \tfrac{1}{2}\Big(\mathbb{E}[f(X)] + \mathbb{E}[f(-X)]\Big) \;=\; \mathbb{E}[f(X)]$$

Antithetic variables

The variance is given by

$$\begin{aligned}\mathbb{V}[\overline{f}] &= \tfrac{1}{4}\Big(\mathbb{V}[f(X)] + 2\,\mathrm{Cov}[f(X), f(-X)] + \mathbb{V}[f(-X)]\Big) \\ &= \tfrac{1}{2}\Big(\mathbb{V}[f(X)] + \mathrm{Cov}[f(X), f(-X)]\Big)\end{aligned}$$

The variance is always reduced, but the cost is almost doubled, so net benefit only if $\mathrm{Cov}[f(X), f(-X)] < 0$.

Two extremes:

- A linear payoff, $f = a + bX$, is integrated exactly since $\overline{f} = a$ and $\mathrm{Cov}[f(X), f(-X)] = -\mathbb{V}[f]$
- A symmetric payoff $f(X) = f(-X)$ is the worst case since $\mathrm{Cov}[f(X), f(-X)] = \mathbb{V}[f]$

General assessment – usually not very helpful, but can be good in particular cases where the payoff is nearly linear

Control Variates

Suppose we want to estimate $\mathbb{E}[f(X)]$, and there is another function $g(X)$ for which we know $\mathbb{E}[g(X)]$.

We can use this by defining a new estimator

$$\widehat{f} = \overline{f} - \lambda\,(\overline{g} - \mathbb{E}[g])$$

Again unbiased since $\mathbb{E}[\widehat{f}] = \mathbb{E}[\overline{f}] = \mathbb{E}[f]$

Control Variates

For a single sample,

$$\begin{aligned}\mathbb{V}[f - \lambda\,(g - \mathbb{E}[g])] &= \mathbb{V}[f - \lambda\, g] \\ &= \mathbb{V}[f] - 2\,\lambda\,\mathrm{Cov}[f, g] + \lambda^2\,\mathbb{V}[g]\end{aligned}$$

For an average of N samples,

$$\mathbb{V}[\overline{f} - \lambda\,(\overline{g} - \mathbb{E}[g])] = N^{-1}\Big(\mathbb{V}[f] - 2\,\lambda\,\mathrm{Cov}[f, g] + \lambda^2\,\mathbb{V}[g]\Big)$$

To minimise this, the optimum value for λ is

$$\lambda = \frac{\mathrm{Cov}[f, g]}{\mathbb{V}[g]}$$

Control Variates

The resulting variance is

$$N^{-1}\,\mathbb{V}[f]\left(1 - \frac{(\mathrm{Cov}[f, g])^2}{\mathbb{V}[f]\,\mathbb{V}[g]}\right) = N^{-1}\,\mathbb{V}[f]\,(1 - \rho^2)$$

where $-1 < \rho < 1$ is the correlation between f and g.

The challenge is to choose a good g which is well correlated with f.
The covariance, and hence the optimal λ, can be estimated numerically.