

# Stochastic Simulation: Lecture 2

Christoph Reisinger

Oxford University Mathematical Institute

Modified from earlier slides by Prof. Mike Giles.

# Variance Reduction

Monte Carlo starts as a very simple method – the complexity comes from trying to reduce the variance, to reduce the number of samples that have to be simulated to achieve a given accuracy.

- ▶ antithetic variables
- ▶ control variates
- ▶ importance sampling
- ▶ stratified sampling
- ▶ Latin hypercube
- ▶ quasi-Monte Carlo (lecture 5)

## Review of elementary results

If  $a, b$  are random variables, and  $\lambda, \mu$  are constants, then

$$\mathbb{E}[a + \mu] = \mathbb{E}[a] + \mu$$

$$\mathbb{V}[a + \mu] = \mathbb{V}[a]$$

$$\mathbb{E}[\lambda a] = \lambda \mathbb{E}[a]$$

$$\mathbb{V}[\lambda a] = \lambda^2 \mathbb{V}[a]$$

$$\mathbb{E}[a + b] = \mathbb{E}[a] + \mathbb{E}[b]$$

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

where

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

$$\operatorname{Cov}[a, b] \equiv \mathbb{E} \left[ (a - \mathbb{E}[a]) (b - \mathbb{E}[b]) \right]$$

# Review of elementary results

If  $a, b$  are independent random variables then

$$\mathbb{E}[f(a) g(b)] = \mathbb{E}[f(a)] \mathbb{E}[g(b)]$$

Hence,  $\text{Cov}[a, b] = 0$  and therefore  $\mathbb{V}[a + b] = \mathbb{V}[a] + \mathbb{V}[b]$

Extending this to a set of  $N$  iid (independent identically distributed) r.v.'s  $x_n$ , we have

$$\mathbb{V} \left[ \sum_{n=1}^N x_n \right] = \sum_{n=1}^N \mathbb{V}[x_n] = N \mathbb{V}[x]$$

and so

$$\mathbb{V} \left[ N^{-1} \sum_{n=1}^N x_n \right] = N^{-1} \mathbb{V}[x]$$

# Antithetic variables

The simple estimator from the last lecture has the form

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

where  $W^{(i)}$  is the value of the Brownian path  $W_T$  at time  $T$ .

$W_T$  is Normally distributed so  $-W_T$  is just as likely.

# Antithetic variables

Antithetic estimator replaces  $f(W^{(i)})$  by

$$\bar{f}^{(i)} = \frac{1}{2} \left( f(W^{(i)}) + f(-W^{(i)}) \right)$$

Clearly still unbiased since

$$\mathbb{E}[\bar{f}] = \frac{1}{2} \left( \mathbb{E}[f(W)] + \mathbb{E}[f(-W)] \right) = \mathbb{E}[f(W)]$$

The variance is given by

$$\begin{aligned} \mathbb{V}[\bar{f}] &= \frac{1}{4} \left( \mathbb{V}[f(W)] + 2 \operatorname{Cov}[f(W), f(-W)] + \mathbb{V}[f(-W)] \right) \\ &= \frac{1}{2} \left( \mathbb{V}[f(W)] + \operatorname{Cov}[f(W), f(-W)] \right) \end{aligned}$$

# Antithetic variables

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

Two extremes:

- ▶ A linear payoff,  $f = a + b W$ , is integrated exactly since  $\bar{f} = a$  and  $\text{Cov}[f(W), f(-W)] = -\mathbb{V}[f]$
- ▶ A symmetric payoff  $f(W) = f(-W)$  is the worst case since  $\text{Cov}[f(W), f(-W)] = \mathbb{V}[f]$

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 approximate  $\mathbb{E}[f]$  using a simple Monte Carlo average  $\bar{f}$ .

If there is another payoff  $g$  for which we know  $\mathbb{E}[g]$ , can use  $\bar{g} - \mathbb{E}[g]$  to reduce error in  $\bar{f} - \mathbb{E}[f]$ .

How? By defining a new estimator

$$\hat{f} = \bar{f} - \lambda (\bar{g} - \mathbb{E}[g])$$

Again unbiased since  $\mathbb{E}[\hat{f}] = \mathbb{E}[\bar{f}] = \mathbb{E}[f]$



# Control Variates

For a single sample,

$$\mathbb{V}[f - \lambda (g - \mathbb{E}[g])] = \mathbb{V}[f] - 2\lambda \text{Cov}[f, g] + \lambda^2 \mathbb{V}[g]$$

For an average of  $N$  samples,

$$\mathbb{V}[\bar{f} - \lambda (\bar{g} - \mathbb{E}[g])] = N^{-1} \left( \mathbb{V}[f] - 2\lambda \text{Cov}[f, g] + \lambda^2 \mathbb{V}[g] \right)$$

To minimise this, the optimum value for  $\lambda$  is

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

# Control Variates

The resulting variance is

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

where  $\rho$  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  $\lambda$ , can be estimated from the data.

# Importance Sampling

Importance sampling involves a change of probability measure. Instead of taking  $X$  from a distribution with p.d.f.  $p_1(X)$ , we instead take it from a different distribution with p.d.f.  $p_2(X)$ .

$$\begin{aligned}\mathbb{E}_1[f(X)] &= \int f(X) p_1(X) \, dX \\ &= \int f(X) \frac{p_1(X)}{p_2(X)} p_2(X) \, dX \\ &= \mathbb{E}_2[f(X) R(X)]\end{aligned}$$

where  $R(X) = p_1(X)/p_2(X)$  is the Radon-Nikodym derivative.

# Importance Sampling

We want the new variance  $\mathbb{V}_2[f(X) R(X)]$  to be smaller than the old variance  $\mathbb{V}_1[f(X)]$ .

How do we achieve this? Ideal is to make  $f(X)R(X)$  constant, so its variance is zero.

More practically, make  $R(X)$  small where  $f(X)$  is large, and make  $R(X)$  large where  $f(X)$  is small.

Small  $R(X) \iff$  large  $p_2(X)$  relative to  $p_1(X)$ , so more random samples in region where  $f(X)$  is large.

Particularly important for rare event simulation where  $f(X)$  is zero almost everywhere.

# Stratified Sampling

The key idea is to achieve a more regular sampling of the most “important” dimension in the uncertainty.

Start by considering a one-dimensional problem:

$$I = \int_0^1 f(U) \mathrm{d}U.$$

Instead of taking  $N$  samples, drawn from uniform distribution on  $[0, 1]$ , instead break the interval into  $M$  strata of equal width and take  $L$  samples from each.

# Stratified Sampling

Define  $U_{ij}$  to be the value of  $i^{th}$  sample from strata  $j$ ,

$$\bar{F}_j = L^{-1} \sum_i f(U_{ij}) = \text{average from strata } j,$$

$$\bar{F} = M^{-1} \sum_j \bar{F}_j = \text{overall average}$$

and similarly let

$$\mu_j = \mathbb{E}[f(U) \mid U \in \text{strata } j],$$

$$\sigma_j^2 = \mathbb{V}[f(U) \mid U \in \text{strata } j],$$

$$\mu = \mathbb{E}[f],$$

$$\sigma^2 = \mathbb{V}[f].$$

# Stratified Sampling

With stratified sampling,

$$\mathbb{E}[\bar{F}] = M^{-1} \sum_j \mathbb{E}[\bar{F}_j] = M^{-1} \sum_j \mu_j = \mu$$

so it is unbiased.

The variance is

$$\begin{aligned} \mathbb{V}[\bar{F}] &= M^{-2} \sum_j \mathbb{V}[\bar{F}_j] = M^{-2} L^{-1} \sum_j \sigma_j^2 \\ &= N^{-1} M^{-1} \sum_j \sigma_j^2 \end{aligned}$$

where  $N = LM$  is the total number of samples.

# Stratified Sampling

Without stratified sampling,  $\mathbb{V}[\bar{F}] = N^{-1}\sigma^2$  with

$$\begin{aligned}\sigma^2 &= \mathbb{E}[f^2] - \mu^2 \\ &= M^{-1} \sum_j \mathbb{E}[f(U)^2 \mid U \in \text{strata } j] - \mu^2 \\ &= M^{-1} \sum_j (\mu_j^2 + \sigma_j^2) - \mu^2 \\ &= M^{-1} \sum_j ((\mu_j - \mu)^2 + \sigma_j^2) \\ &\geq M^{-1} \sum_j \sigma_j^2\end{aligned}$$

Thus stratified sampling reduces the variance.



# Stratified Sampling

How do we use this for MC simulations?

For a one-dimensional application:

- ▶ Break  $[0, 1]$  into  $M$  strata
- ▶ For each stratum, take  $L$  samples  $U$  with uniform probability distribution
- ▶ Define  $X = \Phi^{-1}(U)$  and use this for  $W_T$
- ▶ Compute average within each stratum, and overall average.

# Stratified Sampling

For a multivariate Normal application, one approach is to:

- ▶ Break  $[0, 1]$  into  $M$  strata
- ▶ For each stratum, take  $L$  samples  $U$  with uniform probability distribution
- ▶ Define  $X_1 = \Phi^{-1}(U)$
- ▶ Simulate other elements of  $X$  using standard Normal random number generation
- ▶ Multiply  $X$  by matrix  $C$  to get  $Y = CX$  with desired covariance
- ▶ Compute average within each stratum, and overall average

# Stratified Sampling

Alternatively, for a  $d$ -dimensional application, can split each dimension of the  $[0, 1]^d$  hypercube into  $M$  strata producing  $M^d$  sub-cubes.

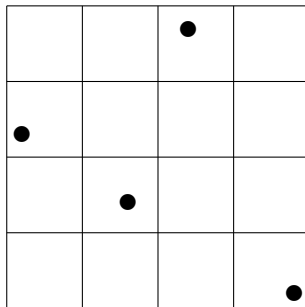
One generalisation of stratified sampling is to generate  $L$  points in each of these hypercubes

However, the total number of points is  $L M^d$  which for large  $d$  would force  $M$  to be very small in practice.

Instead, use a method called Latin Hypercube sampling

# Latin Hypercube

Generate  $M$  points, dimension-by-dimension, using 1D stratified sampling with 1 value per stratum, assigning them randomly to the  $M$  points to give precisely one point in each stratum



# Latin Hypercube

This gives one set of  $M$  points, with average

$$\bar{f} = M^{-1} \sum_{m=1}^M f(U_m)$$

Since each of the points  $U_m$  is uniformly distributed over the hypercube,

$$\mathbb{E}[\bar{f}] = \mathbb{E}[f]$$

The fact that the points are not independently generated does not affect the expectation, only the (reduced) variance

# Latin Hypercube

We now take  $L$  independently-generated set of points, each giving an average  $\bar{f}_l$ .

Averaging these

$$L^{-1} \sum_{l=1}^L \bar{f}_l$$

gives an unbiased estimate for  $\mathbb{E}[f]$ , and the empirical variance for  $\bar{f}_l$  gives a confidence interval in the usual way.

# Latin Hypercube

Note: in the special case in which the function  $f(U)$  is a sum of one-dimensional functions:

$$f(U) = \sum_i f_i(U_i)$$

where  $U_i$  is the  $i^{th}$  component of  $U$ , then Latin Hypercube sampling reduces to 1D stratified sampling in each dimension.

In this case, potential for very large variance reduction by using large sample size  $M$ .

Much harder to analyse in general case.

# Final comments

- ▶ Antithetic variables are usually of little benefit
- ▶ Control variates can be very effective
- ▶ Importance sampling can be very good in certain situations
- ▶ Stratified sampling is very effective in 1D, but not so clear how to use it in multiple dimensions
- ▶ Latin Hypercube is one generalisation – very effective when function can be decomposed into a sum of 1D functions
- ▶ Hard to predict which variance reduction approach will be most effective
- ▶ Advice: when facing a new class of applications, try each one, and don't forget you can sometimes combine different techniques (e.g. stratified sampling with antithetic variables, or Latin Hypercube with importance sampling)