Statistical Inference and Methods

David A. Stephens

Department of Mathematics
Imperial College London

d.stephens@imperial.ac.uk
http://stats.ma.ic.ac.uk/∼das01/

24th January 2006

Part V

Session 5: Simulation Methods

Session 5: Simulation Methods

- Random Number Generation
- Monte Carlo
- Importance Sampling
- Markov chain Monte Carlo
- Stochastic Optimization

Session 5: Simulation Methods

Modern statistics has been revolutionized by the advent of increased computing power. This has facilitated the analysis of data and models that could not have been considered previously.

- random number generation and its uses
- simulation-based inference
- computer-generated modelling and analysis

Session 5: Simulation Methods

Simulation in Statistical Inference: Simulation plays an important role in many areas of statistical inference

- Bootstrap estimation of standard errors
  - Compute parameter estimates for observed data
  - Re-sample “new” data set from observed data
  - Re-compute estimate
  - Repeat to produce large “sample” of estimates
  - Estimate variance of estimator (standard error) from this sample

Session 5: Simulation Methods

- Randomization/Permutation Testing
  - Compute test statistic for observed data
  - Re-sample “new” data set under Null Hypothesis
  - Re-compute estimate for permuted/randomized samples
  - Repeat to produce large “sample” of test statistics
  - Compute *p*-values on the basis of the estimated null distribution given by this sample.

Session 5: Simulation Methods

- Bayesian Inference
  - Bayesian Inference/Decision Making requires integration
  - Rather than computing integrals analytically, produce sample from posterior distributions, and estimate the required integrals
  - Summarize posterior via posterior samples
  - Sample from posterior predictive distribution for forecasting

Session 5: Simulation Methods

Many of the important tasks in computational statistics centre on the generation of random numbers from different probability distributions. Thus we need to be able to

- produce streams of uniform random variates
- devise methods of converting them to non-uniform variates from standard distributions
- seek extensions to non-standard, possibly complicated multivariate ones distributions.
- utilize the variates to perform numerical approximation tasks.

Uniform Random Numbers

The first task is to produce a stream of *uniform* random numbers. Without using actual mechanical experimentation (coin tossing, selecting balls from bags etc.) this is not possible, so instead we produce *pseudorandom* numbers, that is, numbers that appear to be randomly generated, but are in fact produced by some *deterministic* system

There are many different methods of pseudorandom generation

- linear congruential generation
- recursive generators
- linear shift generators

Session 5: Simulation Methods

Linear congruential generator: consider the sequence of real values $\{u_n\}$ defined recursively by

$$u_{n+1} = au_n + c \qquad \mod m$$

where u_0 is user-specified.

By judicious choice of the constants (a, c, m) the sequence of numbers produced by the recursion appears as if it is a stream of random numbers uniformly distributed on the set $\{0, 1, 2, ..., m - 1\}$.

Session 5: Simulation Methods

For example, can choose

- $a = 1366$, $c = 150889$, and $m = 714025$
- $a = 16807$, $c = 0$ and $m = 2^{31} - 1 = 2147483647$.

Can also combine different streams additively (for example, the *Wichman-Hill* algorithm)

For pseudorandom variates from interval $(0, 1)$, use $\{u_n/m\}$.

Transformation Methods

The simplest method of generation from standard distributions is the *method of transformation.* This exploits fundamental results relating the standard distributions. For example

$$U \sim \mathit{Uniform}(0,1) \qquad X = -\log U \qquad \Longrightarrow X \sim \mathit{Exponential}\,(1)$$

or

$$Z \sim N(0,1) \qquad X = Z^2 \qquad \Longrightarrow X \sim \mathit{Chisquared}\,(1)\,.$$

By using summation methods, other random variables can be generated

$$Z_1, ..., Z_K \sim N(0,1) \qquad X = \sum_{i=1}^{K} Z_i^2 \qquad \Longrightarrow X \sim \mathit{Chisquared}\,(K)\,.$$

Session 5: Simulation Methods

An important general simulation technique is *cdf inversion*. If a random variable has cumulative distribution function (cdf) F, so that

$$P\left[X \leq x\right] = F\left(x\right)$$

(that is, $F(x)$ is a known non-decreasing function of x, taking values between 0 and 1), and $U \sim \mathit{Uniform}(0,1)$, then it can be shown that

$$X = F^{-1}\left(U\right)$$

is a random variable with cdf F.

Session 5: Simulation Methods

For example, if $X \sim \textit{Exponential}\,(\lambda)$

$$F(x) = 1 - e^{-\lambda x} \qquad x > 0$$

and thus

$$F^{-1}\,(x) = -\frac{1}{\lambda}\log\,(1 - x)$$

so we can generate an exponential random variate by taking a uniform random variate u and returning $-\log\,(1-u)\,/\lambda$.

NOTES:

- works completely generally, for any F_X
- F^{-1} (the "inverse cdf") function may not be available analytically. For example, if $X \sim N\left(\mu, \sigma^2\right)$

$$F(x) = \int_{-\infty}^{x} \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left\{-\frac{1}{2\sigma^2}(t-\mu)^2\right\} = \Phi\left(\frac{t-\mu}{\sigma}\right)$$

does not yield an inverse function.

Session 5: Simulation Methods

- F^{-1} may not be a 1-1 function, that is, for a given u such that $0 < u < 1$
$$x = F^{-1}(u)$$
may not have a unique solution. This is the case if X is a discrete variable (*Binomial*, *Poisson* etc.)

Session 5: Simulation Methods

Transformation techniques also used for multivariate generation. For example, suppose

$$U_1 \sim \mathit{Uniform}\,(0,1) \qquad U_2 \sim \mathit{Uniform}\,(0,1)$$

and consider the transformation to

$$Z_1 = \sqrt{-2\log U_1}\cos(2\pi U_2) \qquad Z_2 = \sqrt{-2\log U_1}\sin(2\pi U_2)\,.$$

Then it follows that

$$Z_1 \sim N(0,1) \qquad Z_2 \sim N(0,1)\,.$$

This is the *Box-Muller* method of random number generation. (Z_1, Z_2) are independent normal variables, but dependent, non-standard variables can be subsequently generated using a location-scale linear transformation

Rejection

Rejection sampling is a method of producing random variates from a non-standard distribution.

Suppose we wish to generate a variate from a pdf f, and suppose that another probability density g from which random variates can be readily generated, such that $f(x)/g(x)$ is *bounded* on $\mathbb{X}$

$$\frac{f(x)}{g(x)} \leq M < \infty$$

say, for s. Then it is possible to produce a sample from f by generating a variate from g, and then only accepting it as a variate from f if a certain test is passed.

Rejection Algorithm:

1. Generate x from g
2. Generate u from $Uniform(0, 1)$ independently of x
3. Compute

$$t = \frac{uMg(x)}{f(x)}$$

- If $t \leq 1$,
  - **accept** x as a variate from f
  - **STOP**
- If $t > 1$,
  - **reject** x,
  - **return to 1**

Session 5: Simulation Methods

NOTES:

- the probability of acceptance is

$$\frac{1}{M}$$

  and hence the expected number of generations before acceptance is M.
- the smaller M is, the higher the acceptance rate is, and the algorithm is more efficient
- if g resembles f, then the algorithm is more efficient
- can be improved by adaptation of g in light of the rejected points (*adaptive rejection*) or by *pre-testing*/*squeezing*
- works for multivariate distributions

Markov Chain Sampling

Markov chain sampling from probability distributions uses ideas from applied probability to generate random variables. They are especially useful for sampling from complicated or high-dimensional probability distributions.

The key idea is that a Markov chain is a recursive scheme that generates a random sequence of states using elementary simulation methods, but where the next state is generated conditional on the current state.

Session 5: Simulation Methods

Depending on how the next state is generated, it can be proved that, eventually, the proportion of time spent in each the states settles down to some equilibrium.

This *equilibrium*, *invariant*, or *stationary* probability distribution depends only on the mechanism of *transition* from state-to-state.

Basic Markov Chain Theory

Consider a set of states $(s_1, ..., s_K)$ and the sequence of random variables $\{X_n : n \geq 1\}$ where, for each n, $X_n = s_k$ for some k. Suppose that

$$P\left[X_{n+1} = s_j | X_n = s_i\right] = p_{ij}$$

for each n; this defines the conditional probability of moving from state i to state j at step n.

Session 5: Simulation Methods

Combining all of the conditional probabilities into a $K \times K$ matrix, we have

$$P = \begin{bmatrix} p_{11} & p_{12} & \cdots & p_{1K} \\ p_{21} & p_{22} & \cdots & p_{2K} \\ \vdots & \vdots & \ddots & \vdots \\ p_{K1} & p_{K2} & \cdots & p_{KK} \end{bmatrix}$$

where the rows or matrix P sum to 1.

Session 5: Simulation Methods

The matrix P is sufficient to specify the Markov chain. Given an initial state $X_0 = x_0$, we successively generate from the conditional distributions defined by the rows of P, and obtain the sequence $\{x_n : n \geq 1\}$. For example, if $X_n = s_i$, then

$$X_{n+1} = \begin{cases} s_1 & \text{with probability } p_{i1} \\ s_2 & \text{with probability } p_{i2} \\ \quad\vdots & \qquad\vdots \\ s_K & \text{with probability } p_{iK} \end{cases}$$

Session 5: Simulation Methods

The important question now is what happens to the sequence $\{x_n : n \geq 1\}$ in the long run, that is how does the relative frequency

$$\frac{T_i\left(N_1+1, N_2\right)}{N_2 - N_1} = \frac{\text{Number of times } X_n \text{ takes the value } s_i \text{ for } N_1 + 1 \leq n \leq N_2}{N_2 - N_1}$$

behave for $N_1 < N_2$ large ?

Session 5: Simulation Methods

Formally, we can find the *equilibrium* or *steady-state* distribution of the Markov chain, denoted $\pi = (\pi_1, ..., \pi_K)^\mathsf{T}$, by solving the system of equations

$$\pi = P\pi$$

for P. This can be done either using the usual methods for solving linear systems, or using *eigendecompositions*, or simply by looking at the n step ahead transition probabilities given by

$$P \times P \times ... \times P = P^n$$

as $n \to \infty$. We then have that, for fixed large N_1

$$\frac{T_i\left(N_1 + 1, N_1 + n\right)}{n} \to \pi_i$$

that is, the sample relative frequency of being in state i converges to π_i.

Session 5: Simulation Methods

EXAMPLE : Suppose $K = 3$ and

$$P = \begin{bmatrix} \frac{1}{3} & \frac{1}{6} & \frac{1}{2} \\ \frac{1}{8} & \frac{1}{8} & \frac{3}{4} \\ \frac{1}{2} & 0 & \frac{1}{2} \end{bmatrix}$$

Then

$$P^{100} = \begin{bmatrix} 0.4038 & 0.0769 & 0.5192 \\ 0.4038 & 0.0769 & 0.5192 \\ 0.4038 & 0.0769 & 0.5192 \end{bmatrix}$$

so that $\pi = (0.4038, 0.0769, 0.5192)^{\mathsf{T}}$.

Thus, in the long run, the chain spends 40.38% of time in state 1, 7.69% of time in state 2 and 51.92% of time in state 3.

Session 5: Simulation Methods

The method described above can be used to produce approximately independent samples from the equilibrium distribution; to do this the Markov chain is run forward in time, recursively, and after an initial *burn-in* period of N_1 steps, every r^{th} value is collected. That is, the values

$$\{x_{N_1+rn} : n \geq 1\}$$

are deemed to be an approximately independent sample from the discrete distribution π.

Session 5: Simulation Methods

Notes:

- burn-in is necessary so that the chain “forgets” where it started from, and moves into the stationary phase
- “$r-$thinning” is needed to remove dependence between the samples inherent in the Markovian scheme
- often, $N_1 \leq 1000$ and $r = 10 - 50$ is quite sufficient.

Session 5: Simulation Methods

For the example above

N_1	$\left[P^{N_1}\right]_{11}$	N_1	
1	0.3333	6	0.4038
2	0.3819	7	0.4039
3	0.4077	8	0.4038
4	0.4046	9	0.4038
5	0.4037	10	0.4038

where $\left[P^{N_1}\right]_{11}$ is the $(1,1)$ element of $P^{N_{11}}$. This illustrates that convergence to the stationary distribution can be quite rapid.

Continuous State-Space Markov Chains

The theory above extends (reasonably straightforwardly) to continuous state spaces, that is, the countable state set $\{s_1, ..., s_K\}$ is replaced by a continuum of possible values, denoted as above $\mathbb{X}$.

In this case, instead of having a transition matrix, we have a *transition kernel*

$$P(x, B)$$

Session 5: Simulation Methods

$P(x, B)$ determines the probability of making the transition from current value x into the set $B \subset \mathbb{X}$ in any given step.

We retain the discrete time nature of the Markov chain, and again consider outcome sequences $\{x_1, x_2, ..., x_n, ...\}$).

Usually, the transitions are implemented using a *transition density*

$$p(x, y) \equiv p(x \to y)$$

which specifies a conditional probability density in y, given the current value x, for $x, y \in \mathbb{X}$.

Session 5: Simulation Methods

The equilibrium distribution π of the continuous state space model satisfies

$$\pi(x)p(x,y) = \pi(y)\,p(y,x)$$

and given p, we can, in theory solve for π.

However, in the context of sampling from probability distributions as detailed in earlier sections, this is **not** the problem, we are required to solve. We wish to **specify** π, and then find a p such that its equilibrium distribution is π.

This looks like a hard problem to solve. Fortunately, there is a prescribed method that works for general π.

The Metropolis-Hastings Algorithm

Let Q be any transition kernel suitable for moving (exhaustively) around $\mathbb{X}$, with associated transition density q such that

$$q\left(x,y\right)>0$$

for all x,y (in fact, this can be relaxed to the condition that requires $Q^{n}\left(x,y\right)>0$ for all $x,y\in\mathbb{X}$, separated by n steps in the chain).

Then, for $y \neq x$, define

$$p(x, y) = q(x, y)\,\alpha(x, y)$$

where

$$\alpha(x, y) = \min\left\{1, \frac{\pi(y)}{\pi(x)}\frac{q(y, x)}{q(x, y)}\right\}$$

defines an *acceptance probability* for the move from x to y.

Under this transition kernel P with transition density p, if the current state of the chain at step n is $x_n = x$, then the next value of the chain is either some new value $x_{n+1} = y$, generated from the conditional density $q\left(x, y\right)$, or the current value $x_{n+1} = x$. We call y the *candidate* state.

Thus, starting from the n^{th} step when $x_n = x$, we have the following algorithm for implementing the continuous state space Markov chain:

Session 5: Simulation Methods

1. Generate candidate y from the conditional density $q\left(.,.\right)$ given x
2. Compute $\alpha\left(x,y\right)$
3. Generate u from *Uniform* $\left(0,1\right)$
    - if $u \leq \alpha\left(x,y\right)$, **accept** the move to y and set $x_{n+1} = y$
    - if $u > \alpha\left(x,y\right)$, **reject** the move to y and set $x_{n+1} = x$
4. Return to 1 to generate x_{n+2}..

This is the *Metropolis-Hastings algorithm*

Metropolis Algorithm

The general algorithm above has some special cases of interest. If q is chosen such that

$$q(x, y) = q(y, x)$$

so that q is symmetric in its arguments, then

$$\alpha(x, y) = \min\left\{1, \frac{\pi(y)}{\pi(x)}\right\}$$

and the move to y is accepted with certainty if the target probability density at y is higher than at x.

Session 5: Simulation Methods

A simple symmetric transition density has

$$Y|X_n = x \sim N\left(x, \sigma_q^2\right)$$

Choosing σ_q^2 small encourages many small moves; this is the original Markov chain algorithm, known as the *Metropolis Algorithm*.

Many such "local" moves can be proposed. Note that it is important to respect any parameter constraints in the proposal.

Independence Metropolis-Hastings

Another common choice is the *independence Metropolis-Hastings algorithm*, where

$$q(x,y) = q(y)$$

that is, independent of the current value of the chain. This still defines a Markov chain as

$$p(x,y) = q(y)\,\alpha(x,y)$$

still depends on x through $\alpha(x,y)$, and there is a probability that this chain does not move at each step.

Session 5: Simulation Methods

An good independence Markov chain (that traverses $\mathbb{X}$ quickly) is more difficult to construct without knowledge of π.

However, if π can be well-approximated by a density q (as in rejection sampling), then this method can work well.

Gibbs Sampler

The Metropolis-Hastings algorithm above is valid for both univariate and multivariate probability distributions, but is more complicated in high dimensions.

The objective is to choose a transition density q that moves around the space $\mathbb{X}$ quickly, which means that we wish to have the acceptance probability reasonably large.

In high dimensions, this is often difficult to achieve.

Session 5: Simulation Methods

The *Gibbs Sampler* algorithm attempts to solve this problem by breaking down a high-dimensional problem into several lower dimensional problems that are solved iteratively and simultaneously.

Suppose that π is a probability density in K dimensions, and let the variables be denoted $\left(X^1, ..., X^K\right)$.

Session 5: Simulation Methods

Define the conditional density of X^k given $\left(X^1, ..., X^{k-1}, X^{k+1}, ...X^K\right)$

$$\pi_k\left(x^k; x^{(k)}\right) = \frac{\pi\left(x^1, ..., x^K\right)}{\pi\left(x^1, ..., x^{k-1}, x^{k+1}, ...x^K\right)} \propto \pi\left(x^1, ..., x^K\right)$$

where the denominator is the marginal distribution of $X^{(k)}$, the $K-1$ variables excluding X^k,

The Gibbs Sampler utilizes this set of K conditional distributions to construct a Markov chain on the support of the joint distribution.

Session 5: Simulation Methods

It is implemented using the following algorithm:

1. Set a vector of starting values for the K variables $\left(x_0^1, ..., x_0^K\right)$.
2. Sample in turn from the conditional distributions $\pi_k\left(x^k; x^{(k)}\right)$ as follows
   - (a) sample x_1^1 from $\pi_1\left(x^1; x_0^2, x_0^3, ..., x_0^K\right)$
   - (b) sample x_1^2 from $\pi_2\left(x^2; x_1^1, x_0^3, ..., x_0^K\right)$
   - (c) sample x_1^3 from $\pi_3\left(x^3; x_1^1, x_1^1, ..., x_0^K\right)$

     $\vdots$
   - (K) sample x_1^K from $\pi_K\left(x^2; x_1^1, x_1^1, ..., x_1^{K-1}\right)$
3. Return to 2 (a), and repeat to obtain, at step n, the sampled variates $\left(x_n^1, x_n^2, ..., x_n^K\right)$

Session 5: Simulation Methods

This Markov chain defines, in steps 2(a)-2(K), a means of updating the vector x_n to vector x_{n+1}. Each of the steps can be achieved using direct sampling from the conditional distribution if that is possible, but can also involve individual Metropolis-Hastings steps, with acceptance probabilities

$$\alpha_k (x, y) = \min \left\{ 1, \frac{\pi_k \left(y; x^{(k)} \right)}{\pi_k \left(x; x^{(k)} \right)} \frac{q_k (y, x)}{q_k (x, y)} \right\}$$

for $k = 1, ..., K$.

Finally, these steps can be achieved with the scalar variables X^1,...,X^K or with these components as vector variables; deciding on which *blocks* of variables to update simultaneously is often a key issue.

Key Issues And Extensions

- Convergence Assessment
- Designing Algorithms
- Adaptive Schemes & Delayed Rejection
- Utilizing Auxiliary Variables
- Variable Dimension Algorithms
- Perfect Sampling

Example: Stochastic Volatility Model

A simple stochastic volatility model for time-varying variance is based on an $AR(1)$ *state-space* model.

For observable time series data $\{y_t, t = 1, \ldots, T\}$ (e.g. log-returns), we have

$$\begin{aligned} Y_t &= \exp\{h_t/2\}u_t \\ h_t &= \mu + \phi(h_{t-1} - \mu) + v_t \end{aligned}$$

where $\{u_t\}$ and $\{v_t\}$ are (independent) Gaussian processes, with variances 1 and σ^2 respectively. Require $|\phi| < 1$ for stationarity of the second (state) equation.

Session 5: Simulation Methods

Parameters are

- State Equation Parameters (μ, ϕ, σ^2)
- States $\{h_1, \ldots, h_T\}$

Bayesian solution requires computation of the posterior distribution for the unknown parameters

$$\pi(\mu, \phi, \sigma^2, h_1, \ldots, h_T | \mathbf{y})$$

which is exceptionally high-dimensional.

Other solutions equally difficult to implement

Session 5: Simulation Methods

MCMC Solution Consider *Gibbs sampler* Strategy.

Must sample from following *full conditionals*:

- $\pi(\mu|\phi, \sigma^2, \mathbf{h}, \mathbf{y})$
- $\pi(\phi|\mu, \sigma^2, \mathbf{h}, \mathbf{y})$
- $\pi(\sigma^2|\mu, \phi, \mathbf{h}, \mathbf{y})$
- $\pi(h_t|\phi, \sigma^2, \mathbf{h}_{(t)}, \mathbf{y})$, $t = 1, \ldots, T$.

Session 5: Simulation Methods

Likelihood:

$$L(\mathbf{h}) = \left(\frac{1}{2\pi}\right)^{T/2} \prod_{t=1}^{T} \exp\left\{-\frac{1}{2}\left[h_t + e^{-h_t} y_t^2\right]\right\}$$

Priors: Let $\boldsymbol{\theta} = (\mu, \phi, \sigma^2)$

- $p(\boldsymbol{\theta})$
- $p(\mathbf{h}|\boldsymbol{\theta}) = p(h_1|\boldsymbol{\theta}) \prod_{t=2}^{T} p(h_t|h_{t-1}, \boldsymbol{\theta})$, where

$$p(h_t|h_{t-1}, \boldsymbol{\theta}) = \left(\frac{1}{2\pi\sigma}\right)^{1/2} \exp\left\{-\frac{1}{2\sigma^2}(h_t - \mu - \phi(h_{t-1} - \mu)^2\right\}$$

Session 5: Simulation Methods

For example, full conditional for h_t given by

$$\begin{aligned}\pi(h_t|\phi,\sigma^2,\mathbf{h}_{(t)},\mathbf{y}) &\propto L(\mathbf{h})\pi(h_t|\phi,\sigma^2,h_{t-1},h_{t+1},\mathbf{y})\\ &\propto L_t(h_t)p(h_t|h_{t-1},\boldsymbol{\theta})p(h_{t+1}|h_t,\boldsymbol{\theta})\end{aligned}$$

where

$$L_t(h_t) = \exp\left\{-\frac{1}{2}\left[h_t + e^{-h_t}y_t^2\right]\right\}$$

This is a non-standard distribution. Can be sampled using rejection sampling, or a proposal using a Metropolis-Hastings type proposal can be used.

Session 5: Simulation Methods

Key Issue: Convergence

- convergence can be slow due to slow *mixing*
- high autocorrelation in sampled values
- can use special strategies
  - reparameterization
  - sampling $\boldsymbol{\theta}$ as a vector
  - *blocking* the elements of $\mathbf{h}$, that is, sampling the vector of states $(h_{s+1}, \ldots, h_{s+b})$ conditional on the other parameters, for some pair (s, b)

Monte Carlo Integration

Suppose we want to compute the integral

$$I = \int g(x)dx$$

which is known to be finite, but is not analytically tractable. Numerical methods (trapezium rule, quadrature) are often available, but if the integral high-dimensional, these methods may be inaccurate.

Session 5: Simulation Methods

An alternative method is to regard this integral as an expectation with respect to some probability distribution with density f_X

$$I = \int g(x)\frac{f(x)}{f(x)}dx = \int \frac{g(x)}{f(x)}f(x)\,dx = E_f\left[\frac{g(X)}{f(X)}\right]$$

Session 5: Simulation Methods

This suggests a suitable method of approximation. If $x_1, ..., x_N$, is a large sample of observed values from the distribution f_X then a key statistical result (The Law of Large Numbers) relates the observed sample mean to the expectation with respect to f_X, that is,

$$\overline{x} \rightarrow E_{f_X}[X]$$

as $N \rightarrow \infty$.

Session 5: Simulation Methods

By a simple extension

$$\frac{1}{N}\sum_{i=1}^{N}\frac{g(x_i)}{f(x_i)} \rightarrow E_f\left[\frac{g(X)}{f(X)}\right] = I$$

suggesting an estimate

$$\widehat{I} = \frac{1}{N}\sum_{i=1}^{N}\frac{g(x_i)}{f(x_i)}.$$

This is the basic *Monte Carlo method of integration*.

Session 5: Simulation Methods

The Monte Carlo Method is used in many fields of application. Its principal use is in the calculation on expected values:

- in probability and statistics, for computing moments of distributions
- in statistics, for computing properties of estimates (the bootstrap, confidence intervals etc.)
- in finance, for computing Option Prices
  - forward simulation of underlying asset to time T
  - empirical calculation of the value of the option at time 0
- in engineering, for carrying out risk assessment

Importance Sampling

Importance sampling is a modified form of Monte Carlo approximation. Suppose we wish to compute the expected value of function g with respect to f

$$I_f(g) = \int g(x) f(x) dx$$

Session 5: Simulation Methods

Then we may re-write this as an expectation with respect to another density h

$$\begin{aligned} I_f(g) &= \int g(x) f(x) \frac{h(x)}{h(x)} dx = \int \frac{g(x) f(x)}{h(x)} h(x)\, dx \\ &= E_h\left[\frac{g(X) f(X)}{h(X)}\right] \end{aligned}$$

which may then be approximated using standard Monte Carlo as

$$\widehat{I}_f^{(h)}(g) = \frac{1}{N} \sum_{i=1}^{N} \frac{g(x_i) f(x_i)}{h(x_i)}.$$

where $x_1, \ldots, x_N$ are random samples from h.

Session 5: Simulation Methods

This method may produce a more efficient method of estimation of $I_f(g)$ than that given by the usual Monte Carlo estimate

$$\widehat{I}_f(g) = \frac{1}{N}\sum_{i=1}^{N} g(x_i)$$

where $x_1, ..., x_N$ are random samples from g.

Choice of the function h needs some care

Sequential Monte Carlo

Recall basic Monte Carlo approach for Bayesian problems:

- Consider the problem of calculating

$$I = \int_E h(x)\pi(x)dx \tag{1}$$

- We assume that

$$\pi(x) = f(x)/Z$$

where $Z = \int_E f(x)dx < \infty$ is unknown and f may be evaluated pointwise.

- Typically I is of high dimension, cannot be accurately evaluated.

Monte Carlo:

- The basic idea is to draw iid $x^1, \ldots, x^N$ samples from π and use the estimate:

$$\frac{1}{N}\sum_{i=1}^{N} h(x^i) \longrightarrow \int_E h(x)\pi(x)dx.$$

- Monte Carlo integration provides a ‘dimension free’ way to estimate the integral, at the cost of simulating from a high dimensional distribution.
- A major drawback, especially in Bayesian statistics, is that we are unable to draw independent samples from π.

Importance Sampling:

- Suppose that we have a probability density $q(x) > 0 \ \forall x \in E$, that we are able to simulate from.
- The idea of IS is to use the samples from q and reweight them, so we can estimate (1). That is:

$$I = \int_E h(x)w(x)q(x)dx$$

where $w(x) = \pi(x)/q(x)$ is the importance weight.

Session 5: Simulation Methods

- We then use the estimate

$$\frac{1}{N}\sum_{i=1}^{N} h(x^i)w(x^i) \quad \longrightarrow \quad \int_E h(x)\pi(x)dx$$

with $x^1, \ldots, x^N$ drawn from q.

Problems with IS

- Since π is only known up to proportionality we cannot evaluate w. This is easily dealt with, by using the estimate:

$$\widehat{I} = \frac{\sum_{i=1}^{N} h(x^i) w(x^i)}{\sum_{i=1}^{N} w(x^i)} \quad \longrightarrow \quad \int_E h(x)\pi(x)dx.$$

- However, the variance of the estimate is

$$Var\left(\widehat{I}\right) \quad \propto \quad 1 + Var_q\big(w(X)\big)$$

which can be infinite if we are unable to find a suitable q

Session 5: Simulation Methods

So why use IS?

- In *rare event* simulation, it can often be difficult to s imulate a rare event from the original density of interest, but may be achieved using a suitable q and correcting via IS.
- Secondly, an important area of MCMC methodology is *adaptive* methods, that is to adapt the transition kernel to improve the performance of the algorithm.
- Finally, we can easily use the method for parallel simulation.

Sequential Importance Sampling

- Suppose we have a *sequence* of densities $\{\pi_t\}$ with $\pi_t \equiv \pi$, with increasing dimension. Consider also a sequence of corresponding q's $q_0, \ldots, q_t$ of the form:

  $$q_n(x_{0:n}) \quad = \quad q_n(x_n|x_{0:n-1})q_{n-1}(x_{0:n-1}) \; n = 0, \ldots, t$$

  where $x_{0:n} = (x_0, \ldots, x_n)$, $q_{-1} = 1$, $q_n(x_{0:n}) > 0 \; \forall x \in T_n$ and $E = \prod_{n=0}^{t} T_n$.
- The idea is to draw $x_0^1, \ldots, x_0^N$ from q_0 and weight according to the first density:

  $$w_0(x_0^i) \quad = \quad \frac{\pi_0(x_0^i)}{q_0(x_0^i)} \; i = 1, \ldots, N.$$

Session 5: Simulation Methods

- To use the samples for the next density sample x_1^i from $q_1(\cdot|x_0^i)$ and calculate the incremental weight:

$$\omega_1(x_{0:1}^i) = \frac{\pi_1(x_{0:1}^i)}{\pi_0(x_0^i)q_1(x_1^i|x_0^i)}$$

  taking the new weights as $w_1(x_{0:1}^i) = \omega_1(x_{0:1}^i)w_0(x_0^i)$. We continue in this way until 'time' t is reached.
- The idea is to build up a sample, by attempting to solve smaller dimensional problems. That is, to sidestep the difficulty of selecting a single q.

Session 5: Simulation Methods

Drawback of SIS:

- The fundamental difficulty of SIS is that as time proceeds, the weights will have a tendency to degenerate to 0, except for a single sample (or *particle*) which dominates Monte Carlo estimates; this is called *weight degeneracy*.
- Indeed, the particle with the largest weight may not even be relevant for estimation as it is only the ‘best’ particle wrt the others that have been sampled (the *population*).

Session 5: Simulation Methods

Selection:

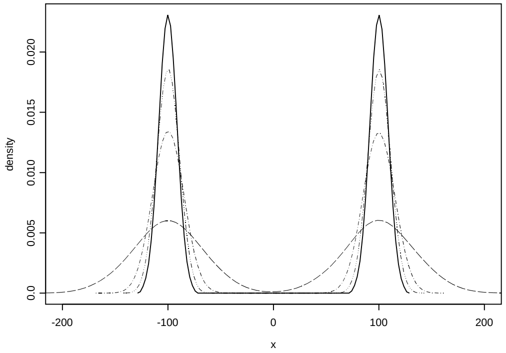
- One way to deal with the weight degeneracy problem is to use selection or resampling.
- The most basic way in which this is achieved is the multinomial approach; this is performed by sampling, with replacement, N particles with probabilities proportional to the weights, we then reset all of the weights to 1 (selection is taken as proxy for a draw from the current target density as will be explained later).
- The idea is to remove the lowest weighted particles and replicate the highest weighted particles with the hope that samples in the future will lie in regions of high density wrt target π_n.

Sequential Monte Carlo

- We now have some idea about SIS and selection, which are key ingredients of SMC samplers.
- The approach of SMC samplers is to use SIS, selection and Markov chain Monte Carlo methods to simulate from our target distribution π.
- Consider the below picture. We want to draw from the density with the full line, but may be difficult; it is far easier to draw, sequentially, from the flattest density and reweight for the next etc.

Session 5: Simulation Methods

Sequence of mixture densities



Session 5: Simulation Methods

SMC Samplers: Basic Idea

- Suppose now that $\{\pi_t\}$ is a sequence of densities so that $\pi_0, \ldots, \pi_{t-1}$ approach, in some sense, π.
- For example, on the previous slide we had $\pi_j(x) \propto \pi(x)^{\zeta_j}$ $0 \leq \zeta_j \leq 1$.
- Basic idea: Generate N particles from initial distribution η_0 and weight according to π_0:

$$w_0(x_0^i) \quad = \quad \frac{\pi_0(x_0^i)}{\eta_0(x_0^i)}.$$

Session 5: Simulation Methods

- Now consider sampling x_1^i from some Markov kernel $K_1(x_0^i, \cdot)$ and reweighting as in SIS:

$$\omega_1(x_{0:1}^i) = \frac{\pi_1(x_1^i)\eta(x_0^i)}{\pi_0(x_0^i)\int_E \eta(x)K_1(x, x_1^i)dx}. \qquad (2)$$

- Drawback: Cannot typically evaluate the integral in (2).
- Solution: Extend state space and create an auxiliary distribution $\widetilde{\pi}_1$ which admits marginal π_1.
- New incremental weight (assuming it is well defined):

$$\omega_1(x_{0:1}^i) = \frac{\widetilde{\pi}_1(x_{0:1}^i)}{\pi_0(x_0^i)K_1(x_0^i, x_1^i)}.$$

Session 5: Simulation Methods

- Continue in this way, i.e. define $\widetilde{\pi}_2$ with marginal π_2 and sample particles from K_2 and then define a $\widetilde{\pi}_4$ etc, so at time t we target $\widetilde{\pi}_t$ which has π as a marginal.
- Could select $\widetilde{\pi}_n$ is to use:

$$\widetilde{\pi}_n(x_{0:j}) \quad = \quad \pi_n(x_n) \prod_{k=0}^{n-1} L_k(x_{k+1}, x_k)$$

where $\{L_{t-1}\}$ are a sequence of *backwards* Markov kernels.

Reasons to use SMC Samplers:

- For many difficult problems (e.g. mixtures) MCMC samplers fail to sample the state space efficiently with extremely poor mixing properties.
- SMC samplers can improve upon such approaches by allowing samples to interact via resampling and allowing a large number of strategies that are prohibited in MCMC; may use *time adaptive* and *non-reversible* kernels in SMC, or *stratification* approaches.
- Our experience with SMC samplers suggest that it is preferable to MCMC in many cases.

Quasirandom Monte Carlo

The Monte Carlo methods described above concentrate on pseudorandom samples from some density f (or h). By definition, these samples are (pseudo) **randomly** distributed around the support of the probability density.

However, it can be shown that, for certain types of integral, it is better to have a **deterministically even distribution** of points around the support of the density, in that the Monte Carlo approximation error is **smaller** in that case.

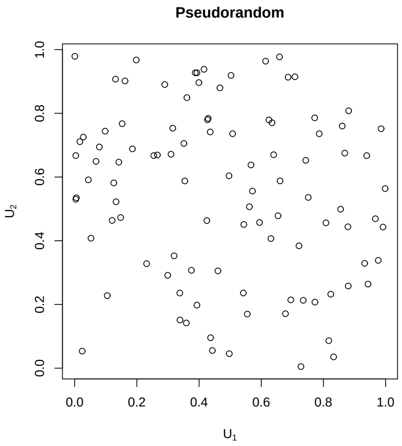
Session 5: Simulation Methods

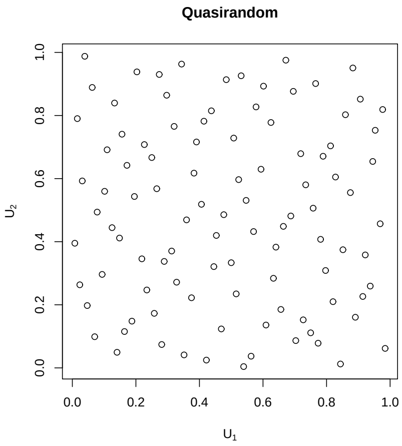
Thus it may be preferable to use *quasirandom* (*low discrepancy*) samples rather than *pseudorandom* samples. There are several well-known methods of placing points optimally in high-dimensions

- Faure sequences
- Halton sequences
- Sobol sequences

Session 5: Simulation Methods

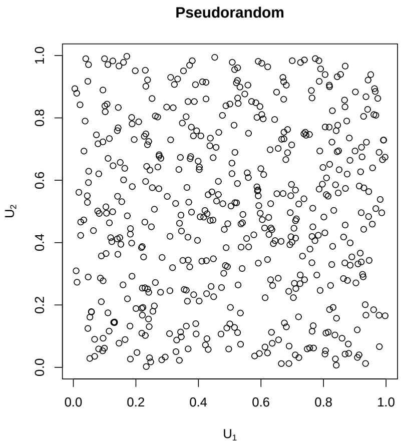
100 Points

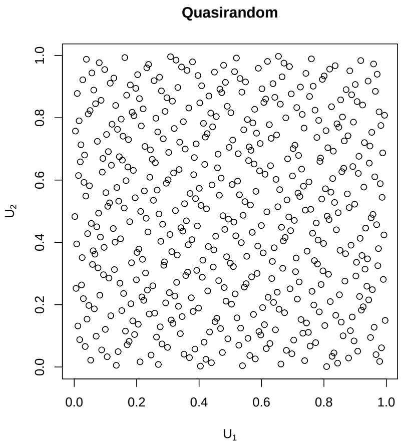




Session 5: Simulation Methods

500 Points





Stochastic Optimization

Optimization is a key topic in many areas of statistics. A generic problem is to minimize (or maximize) some lower (upper) bounded objective function of a vector of variables; that is we seek

$$\mathbf{x}^* = \arg\min_{\mathbf{x}} \phi(\mathbf{x})$$

for objective function ϕ of K variables $\mathbf{x}$.

Session 5: Simulation Methods

The most common approach to optimization in mathematics is to use calculus; we would seek solutions to the K equations

$$\frac{\partial \phi(\mathbf{x})}{\partial x_k} = 0 \qquad k = 1, ..., K$$

to find a turning point of the function. If K is large, or the function ϕ is complicated, these equations may be difficult to solve.

Session 5: Simulation Methods

Simulation-based methods offer a potential solution. Suppose we wish to minimize the function ϕ. First, we define a probability distribution via ϕ as follows. Let

$$\pi_T(\mathbf{x}) \propto \exp\left\{-\frac{1}{T}\phi(\mathbf{x})\right\} = \{\exp\{-\phi(\mathbf{x})\}\}^{1/T} = \{\pi_1(x)\}^{1/T}$$

for some fixed constant T, so that minimizing ϕ equates to maximizing π_T.

Session 5: Simulation Methods

The Metropolis-Hastings approach described above can be used to find high probability density regions that correspond to regions within which ϕ is minimized.

Furthermore, however, the Metropolis-Hastings approach can be used to find a global maximum/minimum if the algorithm is implemented over a slowly changing value of T, that is, if we slowly let $T \to 0$, then the density function would become more concentrated around its maximum.

Session 5: Simulation Methods

The Metropolis algorithm is modified as follows.

- Set $T = T_0 \gg 1$, and $X_1 = x_0 = x$
- Propose move to candidate point $X_1 = x_1 = y$ via q.
- Accept the move to y with probability

$$\alpha_0(x, y) = \min\left\{1, \frac{\pi_{T_0}(y)}{\pi_{T_0}(x)} \frac{q(y, x)}{q(x, y)}\right\}$$

otherwise set $X_1 = x_1 = x$.

- Set $T = T_1 < T_0$
- Return to 2. and 3 with acceptance probability $\alpha_n(x, y)$ with $\pi_{T_0}(.)$ replaced by $\pi_{T_n}(.)$ in the above formula at step n, where

$$T_0 > T_1 > T_2 > ... > T_n$$

such that $T_n \to 0$ as $n \to \infty$.

Session 5: Simulation Methods

If this scheme is used, then the values eventually sampled from the chain are not samples from any of the families of probability distributions

$$\pi_{T_n}(.)$$

but is rather a sample from the vicinity of the mode of the distribution.

The parameter T is known as the *temperature*, and this scheme is referred to as *simulated annealing*.

A logarithmic "cooling" scheme is suitable, so that

$$\frac{1}{T} = a \log(1 + bn)$$