Part A Simulation and Statistical Programming HT14

Lecturer: Geoff Nicholls

University of Oxford

Lecture 2: simulation

Notes and Problem sheets are available at

www.stats.ox.ac.uk\ ∼nicholls\PartASSP

Transformation Methods (continued)

Say $Y \sim Q$, $Y \in \Omega_Q$ we **can** simulate (for example $Y \sim U(0,1)$)

$X \sim P$, $X \in \Omega_P$ we **want to** simulate (eg $X \sim Exp(1)$).

If we can find a function $f : \Omega_Q \to \Omega_P$ with the property that

$$f(Y) \sim P$$

then we can simulate X by simulating

$$Y \sim Q \qquad \text{and setting} \qquad X = f(Y)$$

(for example, set $f(y) = -\log(y)$ and we know from above that if $X = f(Y)$ then $X \sim Exp(1)$).

Example: Suppose we want to simulate $X \sim Exp(1)$ and we can simulate $Y \sim U(0,1)$. Try setting $X = -\log(Y)$. Does that work?

Ans: recall that if Y has density $q(y)$ and $X = f(Y)$ then the density of X is $p(x) = q(y(x))|dy/dx|$. Now $q(y) = 1$ (the density of $U(0,1)$) and $y(x) = \exp(-x)$, so $p(x) = \exp(-x)$ and so X has the density of an $Exp(1)$ rv.

Example: Inversion is a transformation method: Q is $U(0,1)$; Y is U; and $X = f(Y)$ with $f(y) = F^{-1}(y)$ and F the CDF of the target distribution P.

We can generalize the idea. We can take functions of collections of variables.

Example: Suppose we want to simulate $X \sim \Gamma(a, \beta)$ with $a \in 1, 2, 3, ...$ and we can simulate $Y \sim Exp(1)$ (the $\Gamma(a, \beta)$ density is $p(x) = \frac{\beta^\alpha}{\Gamma(\alpha)} x^{\alpha-1} e^{-\beta x}$ for $x > 0$).

Simulate $Y_i \sim Exp(1), i = 1, 2, ..., a$ and set $X = \sum_{i=1}^{a} Y_i / \beta$. Then $X \sim \Gamma(a, \beta)$.

Proof: Use moment generating functions. The MGF of the $Exp(1)$ rv Y is

$$E\left(e^{tY}\right) = (1 - t)^{-1}$$

so the MGF of X is

$$E\left(e^{tX}\right) = \prod_{i=1}^{a} E\left(e^{tY_i/\beta}\right) = (1 - t/\beta)^{-a}$$

which is the MGF of a $\Gamma(a, \beta)$ variate.

Transformation: Box-Muller algorithm for a scalar normal

We often need to simulate iid $X \sim N(0,1)$ rv. The cdf $\Phi(x)$ is not available in closed form so inversion is not straightforward. Here is a simple algorithm.

Box-Muller algorithm (simulates $X, Y \sim N(0,1)$ iid)
Simulate $U_1, U_2 \sim U[0,1]$ iid and set

$$\begin{aligned} R^2 &= -2\log(U_1) \\ \Theta &= 2\pi U_2 \sim U[0, 2\pi] \end{aligned}$$

and

$$\begin{aligned} X &= R\cos\Theta \\ Y &= R\sin\Theta. \end{aligned}$$

Proposition: The X, Y-values simulated by the the Box-Muller algorithm are iid standard normal.

Proof: First, R^2 and Θ are clearly independent with $\Theta \sim U[0, 2\pi]$. Also, $R^2 \sim Exp(\frac{1}{2})$ since $-2\log(U_1)$ is just the inversion rule for $Exp(1/2)$. The joint density of R^2, Θ is

$$f_{R^2,\Theta}(r^2, \theta) = \frac{1}{2}\exp\left(-r^2/2\right)\frac{1}{2\pi}, \qquad 0 \leq R^2, 0 \leq \theta \leq 2\pi$$

To compute the joint density of X and Y we use the change of variables rule for densities. We have

$$f_{X,Y}(x, y) = f_{R^2,\Theta}(r^2, \theta)\left|\det\frac{\partial(r^2, \theta)}{\partial(x, y)}\right|, \qquad r^2 = r^2(x, y), \theta = \theta(x, y)$$

$$\left|\frac{\partial(r^2,\theta)}{\partial(x,y)}\right| = \left|\begin{pmatrix} \frac{\partial x}{\partial r^2} & \frac{\partial x}{\partial \theta} \\ \frac{\partial y}{\partial r^2} & \frac{\partial y}{\partial \theta} \end{pmatrix}\right|^{-1} = \left|\begin{pmatrix} \frac{\cos\theta}{2r} & -r\sin\theta \\ \frac{\sin\theta}{2r} & r\cos\theta \end{pmatrix}\right|^{-1} = 2,$$

and so

$$\begin{aligned} f_{X,Y}(x,y) &= \frac{1}{2}\exp\left(-(x^2+y^2)/2\right)\frac{1}{2\pi}\times 2 \\ &= \exp(-x^2/2)/\sqrt{2\pi} \times \exp(-y^2/2)/\sqrt{2\pi}. \end{aligned}$$

Since $f_{X,Y}(x,y) = f_X(x)f_Y(y)$ factorizes into standard normal densities, we have $X, Y \sim N(0,1)$ iid as supposed.

Comment: This still requires evaluating $\log, \cos$ and $\sin$. The algorithm can be modified to avoid this (problem sheet).

Simulating Multivariate Normal

Let consider $X \in \mathbb{R}^d$, $X \sim N(\mu, \Sigma)$ where μ is the mean and Σ is the (positive definite) covariance matrix.

$$f_X(x) = (2\pi)^{-d/2}|\Sigma|^{-1/2} \exp\left(-\frac{1}{2}(x-\mu)^T \Sigma^{-1} (x-\mu)\right).$$

Proposition: Let $Z = (Z_1, ..., Z_d)$ be a collection of d independent standard normal random variables. Let L be a real $d \times d$ matrix satisfying

$$LL^T = \Sigma.$$

If

$$X = LZ + \mu.$$

then $X \sim N(\mu, \Sigma)$.

Proof: We have $f_Z(z) = (2\pi)^{d/2} \exp\left(-\frac{1}{2} z^T z\right)$. The joint density of the new variables is

$$f_X(x) = f_Z(z) \left| \frac{\partial z}{\partial x} \right|, \qquad z = z(x).$$

In terms of x,

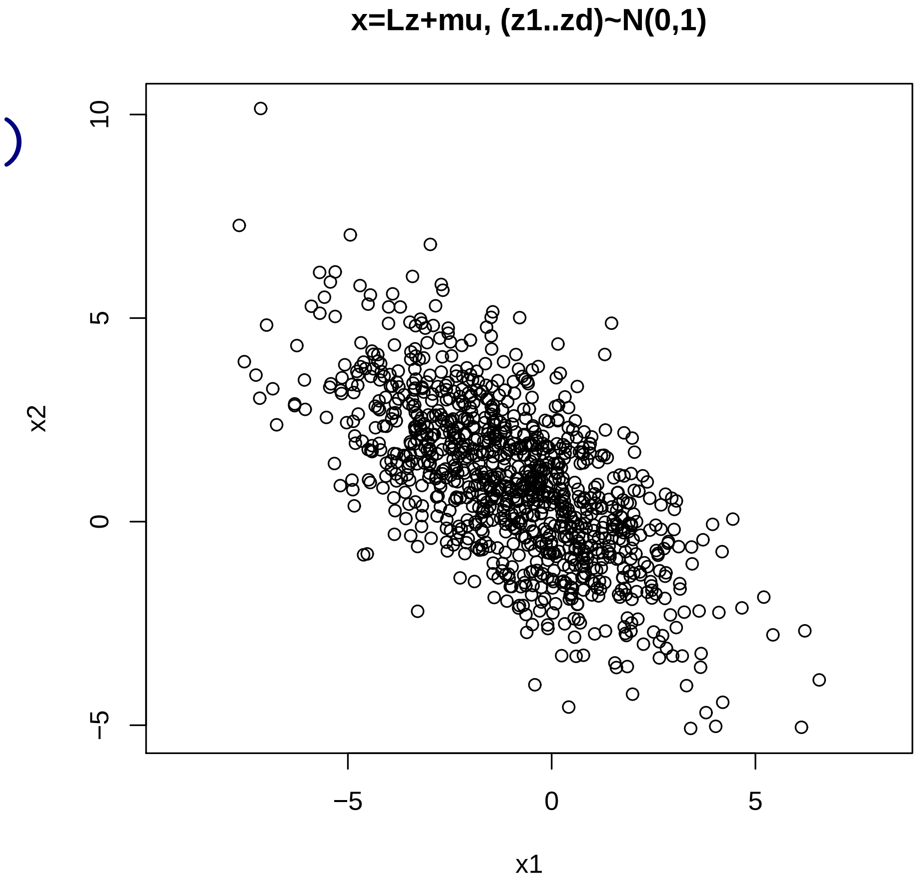
$$\begin{aligned} z^T z &= (x-\mu)^T \left(L^{-1}\right)^T L^{-1} (x-\mu) \\ &= (x-\mu)^T \Sigma^{-1} (x-\mu) . \end{aligned}$$

and $\frac{\partial z}{\partial x} = L^{-1}$ which does not depend on x. It follows that $f_X(x) \propto N(x; \mu, \Sigma)$. If two densities are proportional on the same sample space then they are equal, so $X \sim N(\mu, \Sigma)$.

There are many matrices L satisfying $LL^T = \Sigma$. If $\Sigma = VDV^T$ is the eigenvector decomposition of Σ, we can pick $L = VD^{1/2}$.

The Cholesky factorization with $\Sigma = LL^T$ with L lower triangular is favored, as faster to compute than eigenvectors.

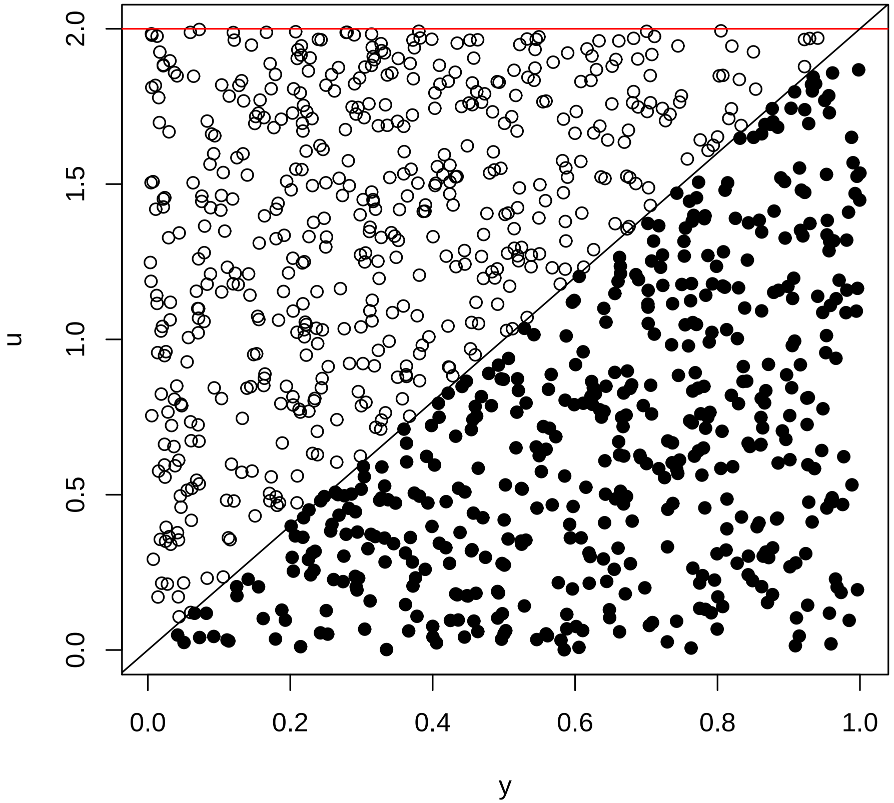
```
#Example dimension d=2, mu=(-1,1), covariance matrix s
#s=| 5 -3|
#  |-3  4|
> mu<-c(-1,1)
> s<-matrix(c(5,-3,-3,4),2,2)
> u<-chol(s)
> (X<-t(u)%*%rnorm(2)+mu)
          [,1]
[1,]  1.769832
[2,] -2.094927
```



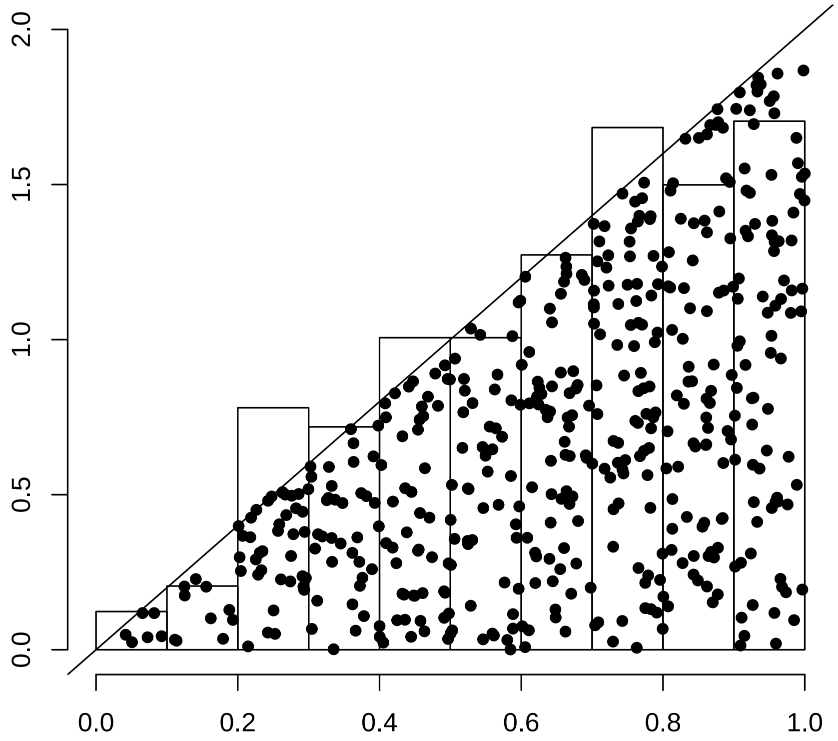
Rejection sampling

Suppose I want to sample $X \sim p(x)$. If I throw darts uniformly at random 'under the curve' of $p(x)$ then the $x - values$ are distributed like $p(x)$. We do this by throwing darts UAR under a function $q(x)$ that sits over $p(x)$, and keeping the darts that fall under $p(x)$.

Example: suppose I want to sample $X \sim p$ with $p_X(x) = 2x,\ 0 \leq x \leq 1$. If I sample $Y \sim U(0,1)$ and $U \sim U(0,1)$ then $(Y, 2U)$ is uniform in the box $[0,1] \times [0,2]$. The points under the curve are the ones we want since $Y|2U < 2Y \sim p_X$.



u
y
0.0
0.5
1.0
1.5
2.0
0.0
0.2
0.4
0.6
0.8
1.0



0.0
0.5
1.0
1.5
2.0
0.0
0.2
0.4
0.6
0.8
1.0

The Rejection algorithm

Say $Y \sim q(y)$, $y \in \Omega$ is a rv we **can** simulate.

$X \sim p(x)$, $x \in \Omega$ is a rv we **want to** simulate.

Suppose there exists M such that $Mq(x) > p(x)$ for all $x \in \Omega$.

Rejection Algorithm simulate $X \sim p(x)$:
[1] Simulate $y \sim q(y)$ and $u \sim U(0,1)$.
[2] If $u < p(y)/Mq(y)$ return $X = y$ ('accept y') and stop, and otherwise goto [1] (reject y).

This algorithm works because $(y, uMq(y))$ is simulated UAR 'under' $q(y)$. We keep trying until we get a point with $V = uMq(y)$ 'under' $p(y)$. This point is UAR under $p(y)$ so its y-component is distributed according to p.

Proposition: The rejection algorithm simulates $X \sim p(x)$.

Proof (for a univariate continuous rv): Let $F(x) = \int_{-\infty}^{x} p(y)dy$ be the cdf of X. We want to show that the algorithm returns values $X = x$ with cdf F. What is the cdf of X? The joint density of a generic pair u and y is $p_{U,Y}(u, y) = q(y)$ because U and Y are independent, $Y \sim q$ and $U \sim U[0, 1]$ so...

$$\begin{aligned}
\Pr(X < x) &= \Pr(Y < x \mid Y \text{ is accepted}) \\
&= \Pr(Y < x \mid U < p(Y)/Mq(Y)) \\
&= \frac{\Pr(Y < x,\ U < p(Y)/Mq(Y))}{\Pr(U < p(Y)/Mq(Y))}.
\end{aligned}$$

Integrate the joint distribution of (u, y) to get these probabilities:

$$
\begin{aligned}
\Pr(Y < x,\ U < p(Y)/Mq(Y)) &= \int_{-\infty}^{x} \int_{0}^{p(y)/Mq(y)} p_{U,Y}(u, y) du dy \\
&= \int_{-\infty}^{x} \int_{0}^{p(y)/Mq(y)} q(y) du dy \\
&= \int_{-\infty}^{x} p(y)/M dy;
\end{aligned}
$$

similarly

$$\Pr(U < p(Y)/Mq(Y)) = \int_{-\infty}^{\infty} p(y)/M dy.$$

The M's will cancel. Now $\int_{-\infty}^{\infty} p(y)dy = 1$ so

$$\begin{aligned} \Pr(X < x) &= \int_{-\infty}^{x} p(y)dy \\ &= F(x) \end{aligned}$$

and we are done.

See you on Friday

Our next meeting is the Evenlode room in the OUCS facility at 13 Banbury road.

The first problem sheet is online at

`http://www.stats.ox.ac.uk/~nicholls/PartASSP/`

and due Monday 9am of Week 3 at 1 South Parks Road (see tray by mailboxes).

Homework: (same as last week) install R and run the code from this lecture.