Generating Random Numbers and Random Variables

Ozgur KARAYALCIN

Technische Universität Kaiserslautern, Germany

Monte Carlo Methods in Finance Seminar, 2007

Outline

Which methods do we investigate?

- Methods for generating uniformly distributed random variables.
- Methods for transforming those variables to other distributions.

Which methods do we investigate?

- Methods for generating uniformly distributed random variables.
- Methods for transforming those variables to other distributions.

Which methods do we investigate?

- Methods for generating uniformly distributed random variables.
- Methods for transforming those variables to other distributions.

Objectives of the section

We are mimicking genuine randomness.

- to discuss primary considerations of random number generators.
- to present a few simple generators.
- to discuss their implementation.

Objectives of the section

We are mimicking genuine randomness.

- to discuss primary considerations of random number generators.
- to present a few simple generators.
- to discuss their implementation.

Objectives of the section

We are mimicking genuine randomness.

- to discuss primary considerations of random number generators.
- to present a few simple generators.
- to discuss their implementation.

Objectives of the section

We are mimicking genuine randomness.

- to discuss primary considerations of random number generators.
- to present a few simple generators.
- to discuss their implementation.

Objectives of the section

A generator of genuinely random numbers is a mechanism for producing a sequence of random variables $U_1, U_2, ...$ with the properties

- each U_i uniformly distributed between 0 and 1.
- the U_i are mutually independent.

Objectives of the section

A generator of genuinely random numbers is a mechanism for producing a sequence of random variables $U_1, U_2, \ldots$ with the properties

- each U_i uniformly distributed between 0 and 1.
- the U_i are mutually independent.

Objectives of the section

A generator of genuinely random numbers is a mechanism for producing a sequence of random variables $U_1, U_2, ...$ with the properties

- each U_i uniformly distributed between 0 and 1.
- the U_i are mutually independent.

Modular arithmetic

The operation $y \bmod m$ is the remainder of y after division by m.

$$y \bmod m = y - \lfloor y/m \rfloor m$$

where $\lfloor x \rfloor$ denotes the greatest integer less than or equal to x.

For instance $7 \bmod 5 = 2,\ \ 10 \bmod 5 = 0,\ \ 43 \bmod 5 = 3$¸

An example: Linear Congruential Generator

Linear Congruential Generator

$$\begin{aligned} x_{i+1} &= a x_i \mod m \\ u_{i+1} &= x_{i+1}/m \end{aligned}$$

Consider $a = 6$, $m = 11$ and seed $x_0 = 1$. We have the sequence

$$1, 6, 3, 7, 9, 10, 5, 8, 4, 2, 1, 6, 3, ...$$

This sequence is full period.

Some terms to construct a generator

- period length
- reproducibility
- speed
- portability
- randomness

Some terms to construct a generator

- period length
- reproducibility
- speed
- portability
- randomness

Some terms to construct a generator

- period length
- reproducibility
- speed
- portability
- randomness

Some terms to construct a generator

- period length
- reproducibility
- speed
- portability
- randomness

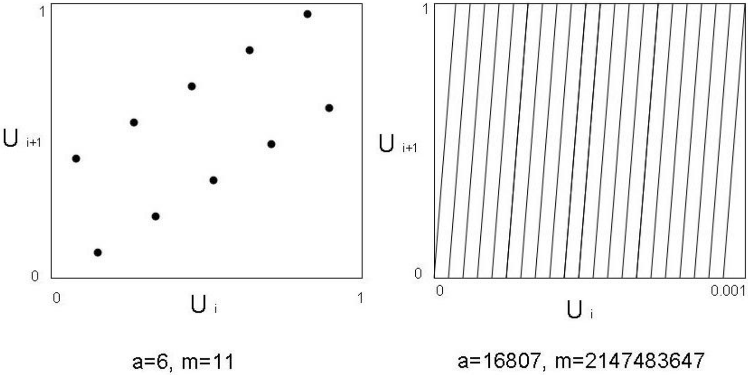
Some terms to construct a generator

- period length
- reproducibility
- speed
- portability
- randomness

Some parameters

Modulus m	Multiplier a	Reference
$2^{31}-1$	16807	Lewis, Goodman, and Miller [234],
$(= 2147483647)$		Park and Miller [294]
	39373	L'Ecuyer [222]
	742938285	Fishman and Moore [123]
	950706376	Fishman and Moore [123]
	1226874159	Fishman and Moore [123]
2147483399	40692	L'Ecuyer [222]
2147483563	40014	L'Ecuyer [222]

Lattice structure



Lattice structure of linear congruential generators

Lattice structure

Spectral test $\Longrightarrow$
measures the degree of equidistribution of points on a lattice.

HOW?

- For each dimension d and each set of parallel hyperplanes containing all points in the lattice, it takes the maximum of the distances between adjacent hyperplanes.
- Computing the spectral test becomes increasingly difficult for higher dimensions.

Lattice structure

Spectral test $\Longrightarrow$
measures the degree of equidistribution of points on a lattice.

HOW?

- For each dimension d and each set of parallel hyperplanes containing all points in the lattice, it takes the maximum of the distances between adjacent hyperplanes.
- Computing the spectral test becomes increasingly difficult for higher dimensions.

Lattice structure

Spectral test $\Longrightarrow$
measures the degree of equidistribution of points on a lattice.

HOW?

- For each dimension d and each set of parallel hyperplanes containing all points in the lattice, it takes the maximum of the distances between adjacent hyperplanes.
- Computing the spectral test becomes increasingly difficult for higher dimensions.

Lattice structure

Spectral test $\Longrightarrow$
measures the degree of equidistribution of points on a lattice.

HOW?

- For each dimension d and each set of parallel hyperplanes containing all points in the lattice, it takes the maximum of the distances between adjacent hyperplanes.
- Computing the spectral test becomes increasingly difficult for higher dimensions.

Combining generators

Why do we combine generators?

- They preserve attractive computational features of original generators
- They extend the period
- They decrease the lattice structure (Some cases)

TECHNISCHE UNIVERSITÄT KAISERSLAUTERN

Combining generators

Why do we combine generators?

- They preserve attractive computational features of original generators
- They extend the period
- They decrease the lattice structure (Some cases)

Combining generators

Why do we combine generators?

- They preserve attractive computational features of original generators
- They extend the period
- They decrease the lattice structure (Some cases)

Combining generators

Why do we combine generators?

- They preserve attractive computational features of original generators
- They extend the period
- They decrease the lattice structure (Some cases)

Wichmann-Hill and L'Ecuyer

Consider J generators, the jth having parameters $a_j,\ m_j$:

$$x_{j,i+1} = a_j x_{j,i} \mod m_j, \quad u_{j,i+1} = x_{j,i+1}/m_j \quad j = 1, \ldots, J$$

The Wichmann-Hill combination sets u_{i+1} equal to the fractional part of

$$u_{1,i+1} + u_{2,i+1} + \cdots + u_{J,i+1}.$$

L'Ecuyer's combination has the form

$$x_{i+1} = \sum_{j=1}^{J} (-1)^{(j-1)} x_{j,i+1} \mod (m_1 - 1)$$

and

$$u_{i+1} = \begin{cases} x_{i+1}/m_1 & x_{i+1} > 0 \\ (m_1 - 1)/m_1 & x_{i+1} = 0 \end{cases}.$$

It assumes that m_1 is the largest of the m_j.

General Sampling Methods

We assume that we have a sequence $U_1, U_2, \ldots$ of independent random variables which which are uniformly distributed. We will investigate two methods

- Inverse Transform Method
- Acceptance -Rejection Method

General Sampling Methods

We assume that we have a sequence $U_1, U_2, \ldots$ of independent random variables which which are uniformly distributed. We will investigate two methods

- Inverse Transform Method
- Acceptance -Rejection Method

TECHNISCHE UNIVERSITÄT KAISERSLAUTERN

General Sampling Methods

We assume that we have a sequence $U_1, U_2, \ldots$ of independent random variables which which are uniformly distributed. We will investigate two methods

- Inverse Transform Method
- Acceptance -Rejection Method

Inverse transform method

Suppose we want to sample from a cumulative distribution F, which means we want to generate a r.v. X s.t. $P(X \leq x) = F(x) \quad \forall x$.

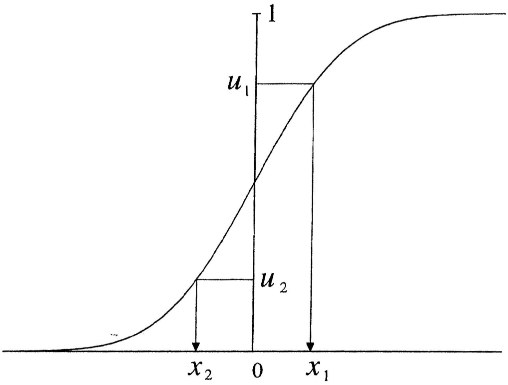
$$X = F^{-1}(U), \quad U \sim \text{Unif}[0,1],$$

where F^{-1} is the inverse of F and Unif$[0,1]$ denotes the uniform distribution on $[0,1]$ Let's verify this method generates sample from F:

$$\begin{aligned} P(X \leq x) &= P(F^{-1}(U) \leq x) \\ &= P(U \leq F(x)) \\ &= F(x)) \end{aligned}$$

F(x) is the CDF of the pdf f(x). The x drawn from this method is distributed as f(x). The method is called Inverse transform because x = F^{-1}(u)

Inverse transform method



Inverse transform method.

Examples (Exponential distribution)

$$
\begin{aligned}
F(x) &= 1 - e^{-x/\theta}, \quad x \geq 0 \\
U &= 1 - e^{-X/\theta} \\
1 - U &= e^{-X/\theta} \\
\log(1-U) &= -X/\theta \\
\log(1-U) * (-\theta) &= X \\
-\theta \log(U) &= X
\end{aligned}
$$

$f(x) = dF(x)/dx = 1/\theta * e^{-x/\theta}$

x is distributed ad f(X)

Inverse transform method

Which important features does it have ?

- It is easy to sample from conditional distribution
- It is useful for implementation of variance reduction techniques.
- It uses just ine uniform r.v. This is important in using quasi-Monte Carlo methods

Inverse transform method

Which important features does it have ?

- It is easy to sample from conditional distribution
- It is useful for implementation of variance reduction techniques.
- It uses just ine uniform r.v. This is important in using quasi-Monte Carlo methods

Inverse transform method

Which important features does it have ?

- It is easy to sample from conditional distribution
- It is useful for implementation of variance reduction techniques.
- It uses just ine uniform r.v. This is important in using quasi-Monte Carlo methods

Inverse transform method

Which important features does it have ?

- It is easy to sample from conditional distribution
- It is useful for implementation of variance reduction techniques.
- It uses just ine uniform r.v. This is important in using quasi-Monte Carlo methods

Acceptance-Rejection Method

Suppose we want to generate samples from a density f defined on some set X.

Let g be a density on X from which we know how to generate samples and with the property that

$$f(x) \leq cg(x), \quad \forall x \in X$$

for some constant c.

Acceptance-Rejection Method

1. generate X from distribution g
2. generate U from Unif[0,1]
3. if $U \leq f(X)/cg(X)$
   return X
   otherwise
   go to Step 1.

The acceptance-rejection method for sampling from density f using candidates from density g.

Acceptance-Rejection Method

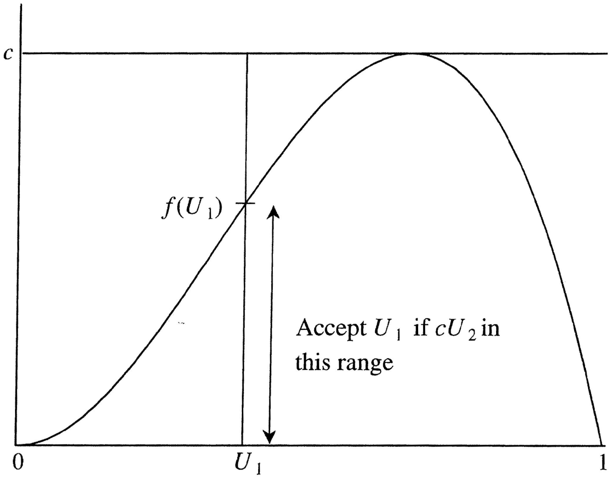


Illustration of the acceptance-rejection method using uniformly distributed

Acceptance-Rejection Method

Let us verify the validity of this method. Let Y be a sample we got from the algorithm and Y has the distribution of X conditional on $U \leq \frac{f(X)}{cg(X)}$.

$$P(Y \in A) = P(X \in A \setminus U \leq \frac{f(X)}{cg(X)}) = \frac{P(X \in A, U \leq \frac{f(X)}{cg(X)})}{P(U \leq \frac{f(X)}{cg(X)})}$$

$$\begin{aligned} P(U \leq \frac{f(X)}{cg(X)}) &= \int_X P(U \leq \frac{f(X)}{cg(X)} \mid X = x) P(X = x) dx \\ &= \int_X \frac{f(x)}{cg(x)} g(x) dx = 1/c \end{aligned}$$

$$P(Y \in A) = cP(X \in A, U \leq \frac{f(X)}{cg(X)}) = c\int_A \frac{f(x)}{cg(x)} g(x) dx = \int_A f(x) dx$$

TECHNISCHE UNIVERSITÄT KAISERSLAUTERN

Normal from double exponential

Pdf of double exponential on $(-\infty, \infty)$:

$$g(x) = \exp\left(-\frac{|x|}{2}\right)$$

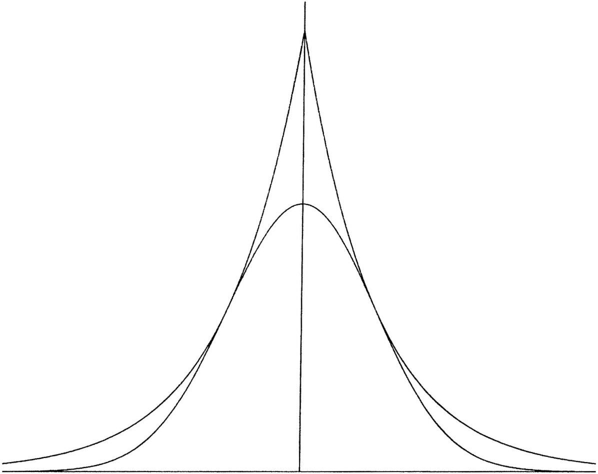
Pdf of normal distribution when $\mu = 0$ and $\sigma^2 = 1$:

$$f(x) = \frac{1}{\sqrt{2\pi}} \exp\left(-\frac{x^2}{2}\right)$$

The rejection test $u > \frac{f(x)}{cg(x)}$ can be implemented as

$$u > \exp\left(-\frac{1}{2}x^2 + |x| - \frac{1}{2}\right) = \exp\left(-\frac{1}{2}(|x| - 1)^2\right)$$

Normal from double exponential



Normal density and scaled double exponential.

Acceptance-Rejection Method

- It can be accelerated through squeeze method, in which simpler tests are applied before the exact one.
- It can be also applied to discrete distributions.
- It is generally used combining with other techniques.
- It uses one uniform r.v. per nonuniform r.v. generated. When simulation problems are formulated as numerical integration problems, the dimension of the integrand is equal to maximum number of uniform variables.
- It is generall inapplicable with quasi-Monte Carlo methods.

Acceptance-Rejection Method

- It can be accelerated through squeeze method, in which simpler tests are applied before the exact one.
- It can be also applied to discrete distributions.
- It is generally used combining with other techniques.
- It uses one uniform r.v. per nonuniform r.v. generated. When simulation problems are formulated as numerical integration problems, the dimension of the integrand is equal to maximum number of uniform variables.
- It is generall inapplicable with quasi-Monte Carlo methods.

Acceptance-Rejection Method

- It can be accelerated through squeeze method, in which simpler tests are applied before the exact one.
- It can be also applied to discrete distributions.
- It is generally used combining with other techniques.
- It uses one uniform r.v. per nonuniform r.v. generated. When simulation problems are formulated as numerical integration problems, the dimension of the integrand is equal to maximum number of uniform variables.
- It is generall inapplicable with quasi-Monte Carlo methods.

Acceptance-Rejection Method

- It can be accelerated through squeeze method, in which simpler tests are applied before the exact one.
- It can be also applied to discrete distributions.
- It is generally used combining with other techniques.
- It uses one uniform r.v. per nonuniform r.v. generated. When simulation problems are formulated as numerical integration problems, the dimension of the integrand is equal to maximum number of uniform variables.
- It is generall inapplicable with quasi-Monte Carlo methods.

Acceptance-Rejection Method

- It can be accelerated through squeeze method, in which simpler tests are applied before the exact one.
- It can be also applied to discrete distributions.
- It is generally used combining with other techniques.
- It uses one uniform r.v. per nonuniform r.v. generated. When simulation problems are formulated as numerical integration problems, the dimension of the integrand is equal to maximum number of uniform variables.
- It is generall inapplicable with quasi-Monte Carlo methods.

Acceptance-Rejection Method

- It can be accelerated through squeeze method, in which simpler tests are applied before the exact one.
- It can be also applied to discrete distributions.
- It is generally used combining with other techniques.
- It uses one uniform r.v. per nonuniform r.v. generated. When simulation problems are formulated as numerical integration problems, the dimension of the integrand is equal to maximum number of uniform variables.
- It is generall inapplicable with quasi-Monte Carlo methods.

Basic properties

$$\begin{aligned}
\phi(x) &= \frac{1}{\sqrt{2\pi}} e^{-x^2/2}, \quad -\infty < x < \infty \\
\Phi(x) &= \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{x} e^{-u^2/2} du \\
\phi_{\mu,\sigma}(x) &= \frac{1}{\sqrt{2\pi}\sigma} e^{-(x-\mu)^2/2\sigma^2} \\
\Phi_{\mu,\sigma}(x) &= \Phi(\frac{x-\mu}{\sigma})
\end{aligned}$$

Box-Muler method

This method generates a sample from bivariate standard normal.

If $Z \sim N(0, I_2)$ then

- $R = Z_1^2 + Z_2^2$ is exponentially distributed with mean 2, i.e.

$$P(R \leq x) = 1 - e^{-x/2}$$

- given R, the point (Z_1, Z_2) is uniformly distributed on the circle of radius $\sqrt{R}$ centered at the origin.

Box-Muler method

This method generates a sample from bivariate standard normal.

If $Z \sim N(0, I_2)$ then

- $R = Z_1^2 + Z_2^2$ is exponentially distributed with mean 2, i.e.

$$P(R \leq x) = 1 - e^{-x/2}$$

- given R, the point (Z_1, Z_2) is uniformly distributed on the circle of radius $\sqrt{R}$ centered at the origin.

Box-Muler method

This method generates a sample from bivariate standard normal.

If $Z \sim N(0, I_2)$ then

- $R = Z_1^2 + Z_2^2$ is exponentially distributed with mean 2, i.e.

$$P(R \leq x) = 1 - e^{-x/2}$$

- given R, the point (Z_1, Z_2) is uniformly distributed on the circle of radius $\sqrt{R}$ centered at the origin.

Box-Muler method

- Generate R and choose a point uniformly from the circle of radius $\sqrt{R}$. $(R = -2\log(U_1))$
- To generate a random point on a circle , generate a random angle uniformly between 0 and 2π and map the angle to a point on the circle.
- The random angle can be generated as $V = 2\pi U_2$ and the corresponding point has coordinates $(\sqrt{R}\cos(V)), (\sqrt{R}\sin(V))$.

Box-Muler method

- Generate R and choose a point uniformly from the circle of radius $\sqrt{R}$. $(R = -2\log(U_1))$
- To generate a random point on a circle , generate a random angle uniformly between 0 and 2π and map the angle to a point on the circle.
- The random angle can be generated as $V = 2\pi U_2$ and the corresponding point has coordinates $(\sqrt{R}\cos(V)), (\sqrt{R}\sin(V))$.

Box-Muler method

- Generate R and choose a point uniformly from the circle of radius $\sqrt{R}$. $(R = -2\log(U_1))$
- To generate a random point on a circle , generate a random angle uniformly between 0 and 2π and map the angle to a point on the circle.
- The random angle can be generated as $V = 2\pi U_2$ and the corresponding point has coordinates $(\sqrt{R}\cos(V)), (\sqrt{R}\sin(V))$.

Box-Muler method

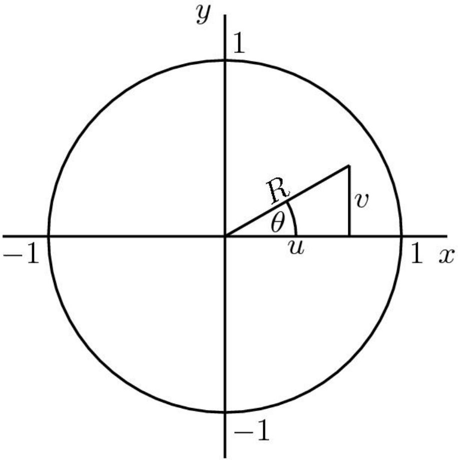
- Generate R and choose a point uniformly from the circle of radius $\sqrt{R}$. $(R = -2\log(U_1))$
- To generate a random point on a circle , generate a random angle uniformly between 0 and 2π and map the angle to a point on the circle.
- The random angle can be generated as $V = 2\pi U_2$ and the corresponding point has coordinates $(\sqrt{R}\cos(V)), (\sqrt{R}\sin(V))$.

Box-Muler method

generate U_1, U_2 independent Unif[0,1]
$R \leftarrow -2\log(U_1)$
$V \leftarrow 2\pi U_2$
$Z_1 \leftarrow \sqrt{R}\cos(V)$, $Z_2 \leftarrow \sqrt{R}\sin(V)$
return Z_1, Z_2.

Box-Muller algorithm for generating normal random variables.

Box-Muler method



$$R^2 = u^2 + v^2$$
$$\cos\theta = \frac{u}{R}$$
$$\sin\theta = \frac{v}{R}$$

Modification of Box-Muler method

while $(X > 1)$

  generate $U_1, U_2 \sim \text{Unif}[0,1]$

  $U_1 \leftarrow 2 * U_1 - 1, \quad U_2 \leftarrow 2 * U_2 - 1$

  $X \leftarrow U_1^2 + U_2^2$

end

$Y \leftarrow \sqrt{-2 \log X / X}$

$Z_1 \leftarrow U_1 Y, \quad Z_2 \leftarrow U_2 Y$

return Z_1, Z_2.

Marsaglia-Bray algorithm for generating normal random variables.

Approximating the inverse normal

Because of the symetry of normal distribution ;

$$\Phi^{-1}(1-u) = -\Phi^{-1}(u), \quad 0 < u < 1;$$

Beasley and Springers approximation

$$\Phi^{-1}(u) \approx \frac{\sum_{n=0}^{3} a_n (u-\frac{1}{2})^{2n+1}}{1+\sum_{n=0}^{3} b_n (u-\frac{1}{2})^{2n}}, \text{ for } 0.5 \leq u \leq 0.92$$

$$\Phi^{-1}(u) \approx g(u) = \sum_{n=0}^{8} c_n [\log(-\log(1-u))]^n, \quad 0.92 \leq u < 1$$

Approximating the inverse normal

Input: u between 0 and 1
Output: x, approximation to $\Phi^{-1}(u)$.
$y \leftarrow u - 0.5$
if $|y| < 0.42$
  $r \leftarrow y * y$
  $x \leftarrow y * (((a_3 * r + a_2) * r + a_1) * r + a_0) /$
    $((((b_3 * r + b_2) * r + b_1) * r + b_0) * r + 1)$
else
  $r \leftarrow u;$
  if $(y > 0)$ $r \leftarrow 1 - u$
  $r \leftarrow \log(-\log(r))$
  $x \leftarrow c_0 + r * (c_1 + r * (c_2 + r * (c_3 + r * (c_4 +$
    $r * (c_5 + r * (c_6 + r * (c_7 + r * c_8)))))))$
  if $(y < 0)$ $x \leftarrow -x$
return x

Beasley-Springer-Moro algorithm for approximating the inverse normal.

Approximating the inverse normal

$$
\begin{aligned}
a_0 &= 2.50662823884 & b_0 &= -8.47351093090\\
a_1 &= -18.61500062529 & b_1 &= 23.08336743743\\
a_2 &= 41.39119773534 & b_2 &= -21.06224101826\\
a_3 &= -25.44106049637 & b_3 &= 3.13082909833
\end{aligned}
$$

$$
\begin{aligned}
c_0 &= 0.3374754822726147 & c_5 &= 0.0003951896511919\\
c_1 &= 0.9761690190917186 & c_6 &= 0.0000321767881768\\
c_2 &= 0.1607979714918209 & c_7 &= 0.0000002888167364\\
c_3 &= 0.0276438810333863 & c_8 &= 0.0000003960315187\\
c_4 &= 0.0038405729373609 & &
\end{aligned}
$$

Constants for approximations to inverse normal.

Approximating the inverse normal

We can simply try to find the root x of the equation $\Phi(x) = u$. For instance Newtons method produces the iteration

Marsaglia and Zaman

$$x_{n+1} = x_n - \frac{\Phi(x_n) - u}{\phi(x_n)},$$

which can be written also

$$x_{n+1} = x_n + (u - \Phi(x_n)) \exp(-0.5 x_n \,.\, x_n + c), \;\; c \equiv \log(\sqrt{\frac{2}{\pi}})$$

Marsaglia and Zaman recommend the starting point

$$x_0 = \pm\sqrt{\mid -1.6 \log(1.0004 - (1-2u)^2) \mid}$$

Approximating the cumulative normal

$b_1 = 0.319381530$ $\quad p = 0.2316419$
$b_2 = -0.356563782$ $\quad c = \log(\sqrt{2\pi}) = 0.918938533204672$
$b_3 = 1.781477937$
$b_4 = -1.821255978$
$b_5 = 1.330274429$

Input: x
Output: y, approximation to $\Phi(x)$
$a \leftarrow |x|$
$t \leftarrow 1/(1 + a * p)$
$s \leftarrow ((((b_5 * t + b_4) * t + b_3) * t + b_2) * t + b_1) * t$
$y \leftarrow s * \exp(-0.5 * x * x - c)$
if $(x > 0)$ $y \leftarrow 1 - y$
return y;

Hastings' [171] approximation to the cumulative normal distribution as modified in Abramowitz and Stegun [3].

Approximating the cumulative normal

$$
\begin{aligned}
v_1 &= 1.2533141373155000 & v_9 &= 0.12313196325793229\\
v_2 &= 0.6556795424187985 & v_{10} &= 0.1097872825783083\\
v_3 &= 0.4213692292880545 & v_{11} &= 0.09902859647173193\\
v_4 &= 0.3045902987101033 & v_{12} &= 0.09017567550106468\\
v_5 &= 0.2366523829135607 & v_{13} &= 0.08276628650136917\\
v_6 &= 0.1928081047153158 & v_{14} &= 0.0764757610162485\\
v_7 &= 0.1623776608968675 & v_{15} &= 0.07106958053885211\\
v_8 &= 0.1401041834530502 & &\\
c &= \log(\sqrt{2\pi}) = 0.918938533204672 & &
\end{aligned}
$$

Input: x between -15 and 15
Output: y, approximation to $\Phi(x)$.
$j \leftarrow \lfloor \min(|x| + 0.5, 14) \rfloor$
$z \leftarrow j, \quad h \leftarrow |x| - z, \quad a \leftarrow v_{j+1}$
$b \leftarrow z * a - 1, \quad q \leftarrow 1, \quad s \leftarrow a + h * b$
for $i = 2, 4, 6, \ldots, 24 - j$
  $a \leftarrow (a + z * b)/i$
  $b \leftarrow (b + z * a)/(i + 1)$
  $q \leftarrow q * h * h$
  $s \leftarrow s + q * (a + h * b)$
end
$y = s * \exp(-0.5 * x * x - c)$
if $(x > 0)$ $y \leftarrow 1 - y$
return y

Algorithm of Marsaglia et al. [251] to approximate the cumulative normal distribution.

Thank you for your attention!