Session 10. Simulation Models

* Simulation

- Real-world phenomenon is too *complex*, and no analytical solution can be obtained from the model. In such a case, simulate the real-world phenomenon!
- Monte Carlo methods (or Monte Carlo experiments) are a class of computational algorithms that rely on repeated random sampling to compute their results.

* Simulation Models

- A *static* model is a representation of a system at a particular point in time.
- A *dynamic* simulation is a representation of a system as it evolves over time.

* Random Number Generators

- *Uniform* random numbers
- *Inverse transformation method*
- *Acceptance-rejection method*

* Simple Monte Carlo Methods

- Expected value μ
- Population proportion π

* Advanced Monte Carlo Methods

- Integration of a function: *Importance sampling*
- Markov Chain Monte Carlo (*MCMC*) method
- *Gibbs sampling* for *multivariate* distributions

A. Uniform Random Number Generators

* Modulo Operation

- It returns the remainder after a number is divided by a divisor.
- For example, 125/13= 9 + 8/13. Thus, *mod*(125, 13) =
- If m is a power of 2 (e.g., $m = 2^{32}$ or $m = 2^{64}$), then this allows the modulo operation to be computed by merely truncating all but the *rightmost* 32 or 64 bits.
- For example, 29 *modulo* 8 (=2^3) is *mod*(29, 8) = or *mod*(11101, 1000) = 101

* Random Number Generator

- Linear congruential generator (*LCG*):

$$x_{i+1} = (a\,x_i + c) \text{ modulo } m$$
$$= x_i - m * int(x_i / m), \qquad \text{for } 0 \le x_i \le m\text{-}1$$

where
$x_0 =$ initial seed, $0 \le x_0 \le m$-1
$a =$ constant multiplier, $0 < a < m$
$c =$ increment, $0 \le c < m$
$m =$ divisor

mod(number, divisor) in Excel

Generate a standard uniform random number (0, 1):

$$U_i = x_i / m \qquad \text{for } 0 \le U_i < 1$$

- If $c = 0$, the generator is often called a *multiplicative* congruential generator (*MCG*).
- LCGs are fast and require minimal memory (typically 32 or 64 bits) to retain each state.

Ex 1] Suppose that a = 13, c = 65, and m = 100. With the initial value x_0 = 35, generate *uniform* random numbers.

i	0	1	2	3	4	…
X_i	35	20	25			…
U_i		0.20	0.25			…

- X_1 = (13*35 + 65) *modulo* 100 = 520 modulo 100 = 20
- X_2 = (13*20 + 65) *modulo* 100 = 325 modulo 100 = 25
- X_3 = (13*25 + 65) *modulo* 100 = 390 modulo 100 =
- X_4 = (13*90 + 65) *modulo* 100 = 1235 modulo 100 =

Ex 2] Suppose that a = 13, c = 3, and $m = 16 = 2^4$. With the initial value x_0 = 5, generate *uniform* random numbers in the following table.

i	x_i	$a\,x_i + c$	x_{i+1}	Binary ($a\,x_i + c$)	Binary (x_{i+1})
0	5	68	4	01000100	0100
1	4	55		00110111	
2	7	94		01011110	
3	14	185	9	10111001	1001
4	9	120	8	01111000	1000
5	8	107	11	01101011	1011

- x_i are uniform random numbers between 0 and 15.
- $m=2^4$. Thus, simply truncate all but the rightmost 4 bits!

* Random Numbers and Monte Carlo Simulation

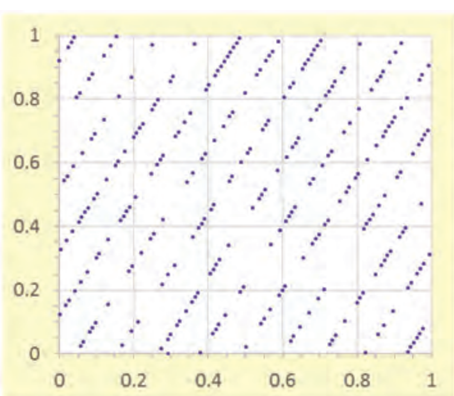
- Use a generated *random number*, instead of spinning a roulette wheel or rolling a die.
- A *random number* is an *independent* random sample drawn from a probability distribution $f(x)$.

* Choice of Parameters

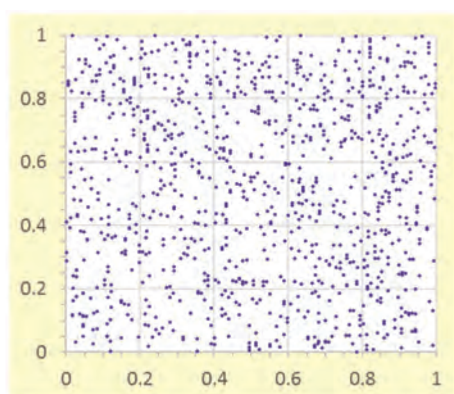
- Eventually, the sequence of random numbers will *repeat* itself, yielding a period for the random number generator.
- m is usually a power of 2, most often $m = 2^{32}$ or $m = 2^{64}$, because this allows the modulus operation to be computed by merely truncating all but the *rightmost* 32 or 64 bits.

* Testing Randomness I: *Lag Plot*

- If x is truly random, then there should be no correlation between successive values of x. Thus, a good way of testing our random number generator is to plot x_i versus x_{i+1} for many different values of i.
- The lag plot checks whether a data set or time series is random or not.
- For a good random number generator, the plotted points should densely fill the unit square. Moreover, there should be no discernible pattern in the distribution of points.



Poor choice of a, c, and m



Looks more random!

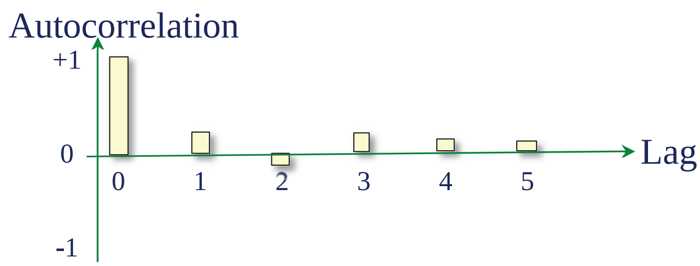
- Lag plots can be generated for any arbitrary lag, although the most commonly used lag is k=1.

* Testing Randomness II: *Autocorrelation Plot*

- Autocorrelation, also known as *serial correlation* or cross-autocorrelation, is the cross-correlation of a signal with itself at different points in time:

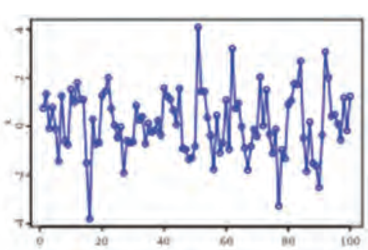
  Autocorrelation coefficient with lag $k = Corr(X_i, X_{i+k})$

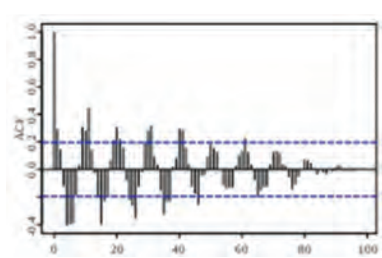
- Autocorrelation plots (or correlograms) are commonly used for checking *randomness* in a data set. The randomness is ascertained by computing *autocorrelations* for data values at varying time lags, k.



- If random, such autocorrelations should be near zero for any and all time-lag separations.
- If non-random, then one or more of the autocorrelations will be significantly non-zero.
- Do you remember Durbin-Watson test for autocorrelation (for k=1) in regression analysis?

Ex] A plot shows 100 random numbers with a "hidden" sine function, and a correlogram of the series.



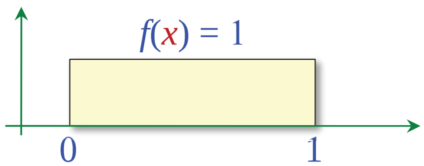


* Random Number Generators

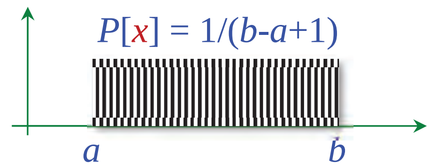
- **Microsoft Excel**

  =rand( ) for the *continuous* uniform random numbers between 0 and 1.

  =randbetween(a, b) for the *discrete* random numbers between a and b.



Continuous uniform



Discrete uniform

- **SAS**: *Functions* and *subroutines*

  - RAND(dist, parm….): It has a very long period (2^{19937} - 1) and very good statistical properties.

    dist = Bernoulli, beta, binomial, Cauchy, chisquare, Erlang, exponential, F, gamma, geometric, hypergeometric, lognormal, negbinomial, normal/Gaussian, Poisson, t, table, triangle, uniform, and Weibull.

  - Other SAS function:
    RANBIN, *RANCAU*, *RANEXP*, *RANGAM*, *RANNOR*, *RANPOI*, *RANTBL*, *RANTRI*, and *RANUNI*.

* Non-Uniform Random Numbers

- There are two basic methods of constructing *non-uniformly* distributed random variables:

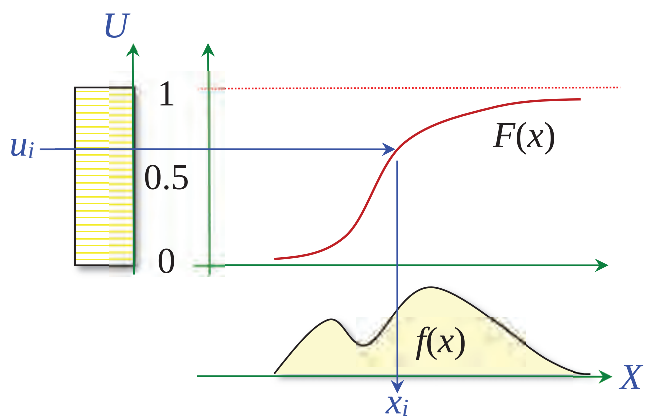
  (1) Inverse transformation method

  (2) Acceptance-Rejection method.

B. Inverse Transformation Method

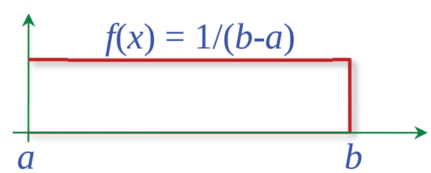
* Inverse Transformation Method

- The standard uniform distribution between 0 and 1 can be used to generate other type of random numbers.
- Let U be a uniform (0, 1) random variable, and $F(x)$ is a cumulative distribution function (*cdf*) for which F^{-1} exists.
- Then, $x=F^{-1}(U)$ is a random draw from the probability density function (*pdf*), $f(x)$.



Ex 1] Uniform distribution (a, b)

$$f(x)=\begin{cases} 1/(b-a) & \text{if } a \le x \le b \\ 0 & \text{otherwise} \end{cases}$$

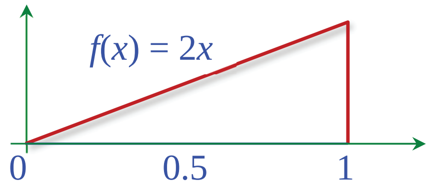


- Its cumulative distribution function (*cdf*) is $F(x)=(x-a)/(b-a)$ and the inverse function is $F^{-1}(U) = a + U*(b-a)$.
- Thus, $X = a + U*(b-a)$ has the uniform distribution between a and b.

Uniform, U	0.231	0.573	0.168	0.962	…
Uniform (-1, +1)	-0.538	0.146			

Ex 2] Triangular distribution

$$f(x) = \begin{cases} 2x & \text{if } 0 \le x \le 1 \\ 0 & \text{otherwise} \end{cases}$$

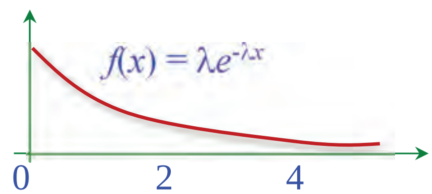


- Its cdf is $F(x)$ =
  and the inverse function is $F^{-1}(U)$ =
- Thus, $X = U^{0.5}$ has the triangular distribution for $0 \le x \le 1$.

Uniform, U	0.231	0.573	0.168	0.962	…
Triangular X	0.481	0.757			

Ex 3] Exponential distribution with λ

$$f(x) = \begin{cases} \lambda e^{-\lambda x} & \text{if } 0 \le x < \infty \\ 0 & \text{otherwise} \end{cases}$$



- Its cdf is $F(x)$ =
  and the inverse function is $F^{-1}(U)$ =
- Thus, $X = -(1/\lambda)\, ln(1-U)$ has the exponential distribution with λ.

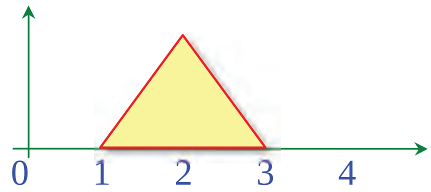
Uniform, U	0.231	0.573	0.168	0.962	…
Exp X with λ=2	0.131	0.425			

Inverse probability functions available in Microsoft Excel

=norm.inv	*=norm.s.inv*	*=binom.inv*
=gamma.inv	*=beta.inv*	*=chisq.inv*
=lognorm.inv	*=F.inv*	*=t.inv*

Ex 4] Triangular distribution

$$f(x) = \begin{cases} x - 1 & \text{if } 1 \le x \le 2 \\ 3 - x & \text{if } 2 \le x \le 3 \end{cases}$$



(a) Find the cumulative distribution function (*cdf*) of the random variable *x*.

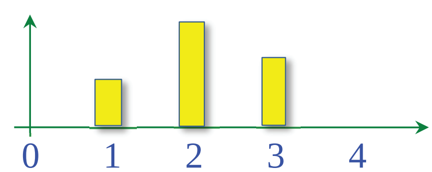
$F(x) =$

(b) Use the inverse tra*nsformation* method to generate random numbers from the probability distribution $f(x)$.

Uniform, U	0.233	0.573	0.170	0.960	…
Triangular distribution	1.683	2.076			

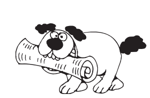
Ex 5] Discrete distribution

x	1	2	3
P[x]	0.2	0.5	0.3



- Generate a *discrete* uniform random number U between 0 and 9.

  If $0 \le U \le 1$, then $x = 1$
  If $2 \le U \le 6$, then $x = 2$
  If $7 \le U \le 9$, then $x = 3$

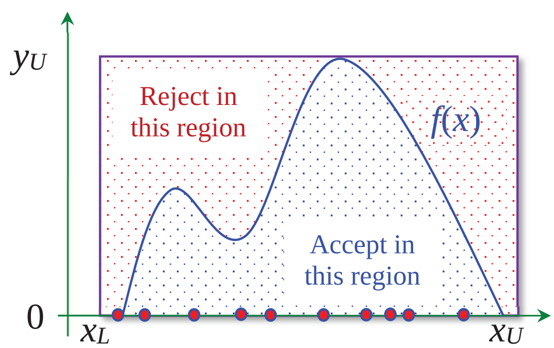


C. Simple Acceptance-Rejection Method

- It is a basic technique used to generate sample observations from a probability distribution $f(x)$. It is also commonly called the “reject sampling” or "accept-reject algorithm" and is a type of Monte Carlo method.
- Suppose that we want to sample from a density $f(x)$ and we cannot use $F^{-1}(U)$ where U is uniform in [0; 1] because F^{-1} is too complicated or not available.
- In such a case, the desired random variates can be obtained by generating *candidate samples* which are either *accepted* or *rejected* to obtain the desired distribution $f(x)$.

* Basic Rejection Method

1. Generate two *uniform* random samples x and y for which $x_L \le x \le x_U$ and $0 \le y \le y_U$.
2. If $y \le f(x)$, then accept x as a realization of $f(x)$; Otherwise reject it and return to step 1.



- Rejection sampling can lead to a lot of unwanted samples being taken if the function being sampled is highly concentrated in a certain region, for example, a function that has a spike at some location.
- In addition, as the dimensions of the problem get larger, a lot of rejections can take place before a useful sample is generated, thus making the algorithm *inefficient* and *impractical*.

Ex 1] Generate a random variable x from the U-shaped density $f(x)=12(x-0.5)^2$ for $0<x<1$.

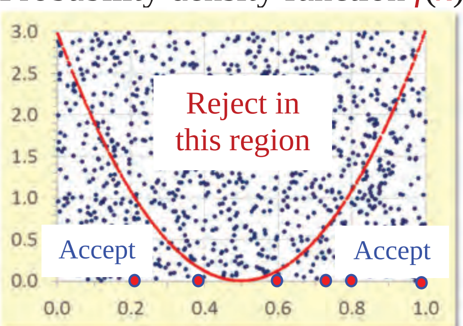
(a) Inverse transformation method?

- The *cdf* is $F(x) = x(4x^2 - 6x+3)$, but can you find the inverse function $F^{-1}(U)$?

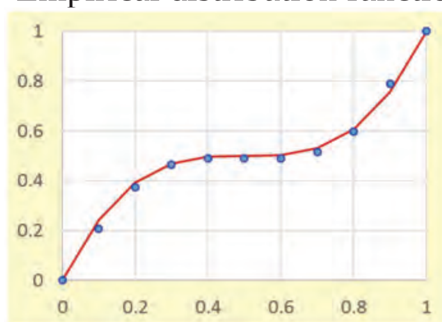
(b) Acceptance-rejection method

1. Generate two uniform random numbers U_1 and U_2.
2. Let $x = U_1$ and $y=3U_2$.
3. If $y < f(x) = 12(x-0.5)^2$, accept x. Otherwise, reject x.

- Probability density function $f(x)$



- Empirical distribution function



i	*x*	*y*	*f*(*x*)	*x*?
1	0.6751	2.3354	0.3679	-
2	0.8214	0.8284		
3	0.1270	0.7371	1.6693	0.1270
4	0.1827	2.9311		
5	0.0726	0.6038	2.1921	0.0726
.	.	.	.	.
499	0.2881	1.7153	0.5390	-
500	0.2120	0.3174	0.9951	0.2120
Average	0.5065	1.4541	0.9869	$E[X]$=**0.5062**

The acceptance rate is 33.33%. The average of the random samples x is the expected value $E[X]$ of the random variable x.

Ex 2] Generate a random variable x from the density $f(x) = 0.25x^2 - x/3 + 1/12$ for $1<x<3$.

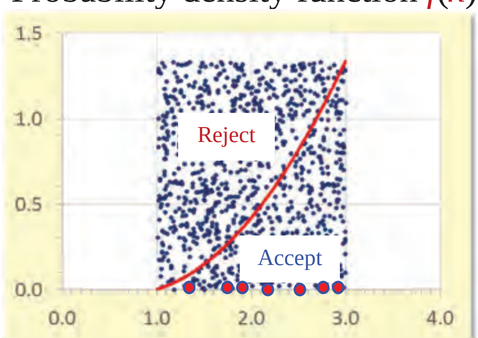
(a) Inverse transformation method?

- The cdf is $F(x) = x^3/12 - x^2/6 + x/12$, but can you find the inverse function $F^{-1}(U)$?

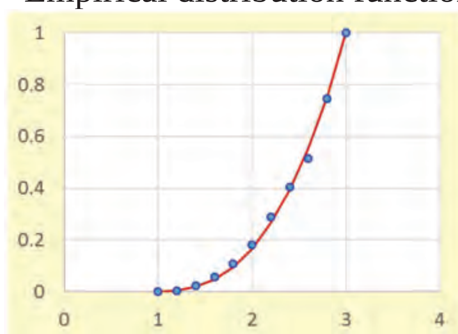
(b) Acceptance-rejection method

1. Generate two uniform random numbers U_1 and U_2.
2. Let $x = 2U_1+1$ and $y=4U_2/3$
3. If $y < f(x) = 0.25x^2 - x/3 + 1/12$, accept x.
4. Otherwise, reject x.

- Probability density function $f(x)$



- Empirical distribution function



i	x	y	$f(x)$	x?
1	1.1014	1.0730	0.0195	-
2	2.5432	0.3784		
3	1.1603	1.2690	0.0331	-
4	2.9801	0.2033	1.3102	2.9801
5	2.3798	0.6647	0.7060	2.3798
.	.	.	.	.
499	1.1424	0.5155	0.0288	-
500	2.9744	1.2352	1.3036	2.9744
Average	1.9858	0.6572	0.4882	$E[X]$=2.4169

The acceptance rate is 3/8 = 37.50%.

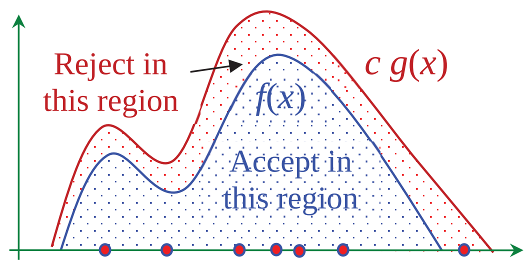
D. General Acceptance-Rejection Method

* Proposal Distribution

- Suppose that it is hard to sample from $f(x)$, and that there is another density $g(x)$ and a constant $c > 1$ such that $f(x)/g(x) \le c$ for all x. The constant c is a bound on $f(x)/g(x)$.
- A more efficient approach is to sample in the area under the graph of *arbitrary* function $g(x)$, with $f(x) \le c\, g(x)$ for all x.

* Procedure

1. Generate a random sample x having probability density, $g(x)$.
2. Generate a standard uniform random variable $U \in [0, 1]$.



3. Check whether or not $U \le f(x)/[\, c\, g(x)\,]$.
4. If this holds, accept x as a realization of $f(x)$. Otherwise reject the value of x and repeat the sampling step.

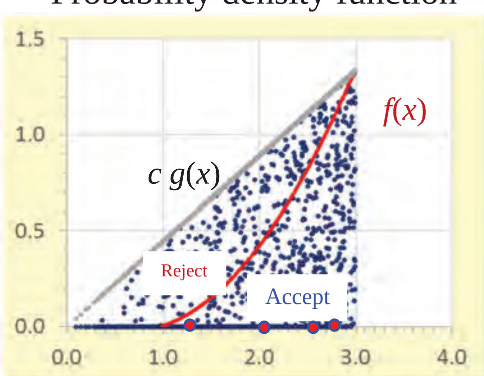
* Properties

- This is a slightly less efficient approximation than when we generate the sample *directly* from $f(x)$, because if we count the steps needed in the acceptance-rejection cycle, then more steps are needed to generate random samples x.
- The acceptance rate is shown to be $1/c$.
- The basic rejection method is a special case in which $g(x)$ is a *uniform* distribution.
- The discrete case is analogous to the continuous case.

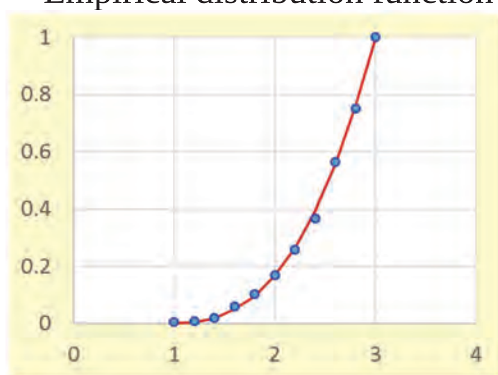
Ex 1] Generate a random sample x from the density $f(x)$ = $0.25x^2 - x/3 + 1/12$ for $1<x<3$. As the *proposal* distribution, use a *triangular* distribution $g(x) = 2x/9$ for $0<x<3$

1. Generate a uniform random number U_1.
2. Transform U_1 to a *triangular* random number x from $g(x)$ [i.e., $x = 3*sqrt(U_1)$]
3. Generate another uniform random number U_2.
4. If $U_2 \leq f(x)/[\ c\ g(x)\ ]$ where c=2, accept x. Otherwise, reject x.

- Probability density function



- Empirical distribution function



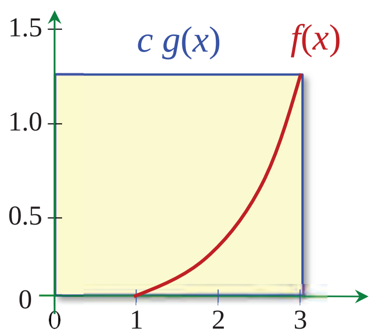
i	x	$g(x)$	$c\ g(x)$	$f(x)$	U_2	x?
1	0.7788	0.1731	0.3461	-0.0246	0.9937	-
2	1.0812				0.0232	
3	2.9162	0.6481	1.2961	1.2374	0.5105	2.9162
4	1.7961	0.3991	0.7983	0.2911	0.5780	-
.	.	.	.	.	.	.
499	2.4029	0.5340	1.0680	0.7259	0.2546	2.4029
500	2.1662	0.4814	0.9627	0.5343	0.6962	
Average	2.0221	0.4494	0.8987	0.5544	0.4972	2.4348

The acceptance rate of the rejection algorithm is $1/c$ = 50%!

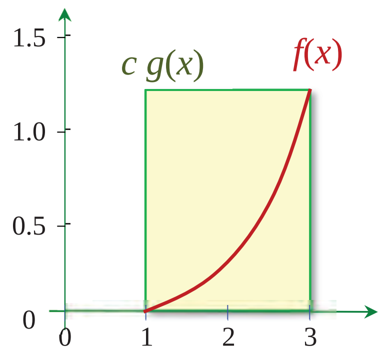
* Proposal Distribution *g*(*x*)

- The acceptance-rejection method is more *effective* if the distributions $f(x)$ and $g(x)$ are somewhat similar.
- In high dimensions, it is suggested to use a Markov chain Monte Carlo (MCMC) method such as Metropolis-Hasting sampling or Gibbs sampling.

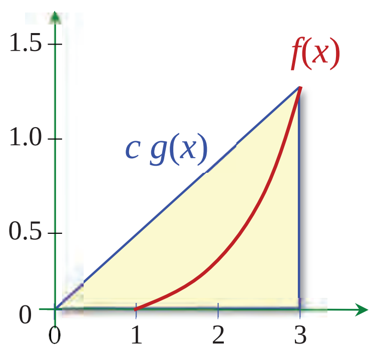
Ex 2] Generate a random sample x from the density $f(x) = 0.25x^2 - x/3 + 1/12$ for $1<x<3$. Find the acceptance rate.



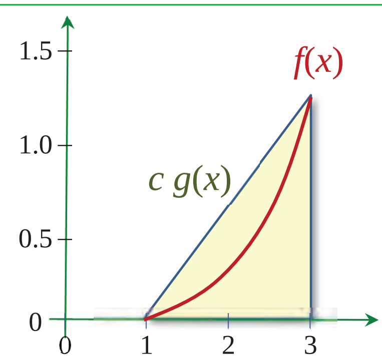
(a) $g(x) = 1/3$ for $0<x<3$
Then, c =
Acceptance rate =



(b) $g(x) = 0.5$ for $1<x<3$
Then, c =
Acceptance rate =



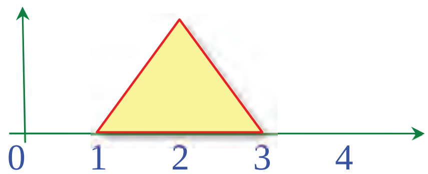
(c) $g(x) = 2x/9$ for $0<x<3$
Then, c =
Acceptance rate =



(d) $g(x) = -0.5 + 0.5x$ for $1<x<3$
Then, c =
Acceptance rate =

Ex 3] Using the *acceptance-rejection method* with the proposal distribution $g(x) = x/8$ for $0 \le x \le 4.0$, we want to generate random numbers from the triangular distribution $f(x)$.

$$f(x) = \begin{cases} x - 1 & 1 \le x \le 2 \\ 3 - x & 2 \le x \le 3 \end{cases}$$



(a) Find the acceptance rate of the acceptance-rejection algorithm.

(b) Use the inverse transformation method and generate random numbers from the proposal distribution $g(x)$.

$x =$

(c) If $U_1 = 0.418$ and $U_2 = 0.152$, what is the random number x?

i	1	2	3	4	5	…
Uniform U_1	0.031	0.213	0.368	0.418	0.962	…
$x = 4U_1^{0.5}$	0.704	1.846	2.427		3.923	
$f(x)$	0.000	0.846	0.573		0.000	
$g(x)$	0.088	0.231	0.303		0.490	
$f(x) / [4\, g(x)]$	0.000	0.917	0.473		0.000	
Uniform U_2	0.478	0.807	0.752	0.152	0.138	…
Random number x	-	1.846	-		-	…

E. Estimation with Monde Carlo Simulation

* Simple Monte Carlo Methods

- *Simple* Monte Carlo is a *direct* simulation of the problem of interest. Simple Monte Carlo is often called crude Monte Carlo to distinguish it from more *sophisticated* methods.

- Our goal is to estimate (1) a population expectation $\mu = E[h(Y)]$ by the corresponding sample expectation, or (2) a population proportion π by the average proportion.

I. Estimating the Expected Value

- We express the quantity we want to know as the expected value of a random variable Y, such as μ = E[Y]. Then, we generate values $Y_1, Y_2, \ldots, Y_N$ independently and randomly from the distribution of Y. Their average,

$$\hat{\mu}_N = \frac{1}{N} \sum_{i=1}^{N} Y_i,$$

is taken as our estimate of μ.

- It can be shown that the sample average $\hat{\mu}_N$ has

$$E[\hat{\mu}_N] = \frac{1}{N} \sum_{i=1}^{N} E[Y_i] = \mu \quad \text{and} \quad Var[\hat{\mu}_N] = \frac{\sigma^2}{N}.$$

- Commonly, $Y = h(X)$ where the random variable X has a probability density function $f(x)$, and h is a real-valued function. Then, the population expectation is

$$\mu = E[h(x)] = \int h(x) f(x) dx,$$

which can be estimated by $\hat{\mu}_N = \frac{1}{N} \sum_{i=1}^{N} h(X_i)$.

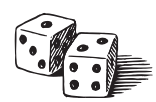
Ex] Chuck-a-Luck

Three fair dice are rolled in a wire cage. You place a bet on any number from 1 to 6. If any one of the three dice comes up with your number, you win the amount of your bet. (You also get your original stake back.) If more than one die comes up with your number, you win the amount of your bet for each match. If you bet $1, what is your expected payoff?

(a) Analytical solution

- Number of matches: Binomial distribution (n=3 and p=1/6)

$$P[X] = \binom{3}{x}\left(\frac{1}{6}\right)^{x}\left(\frac{5}{6}\right)^{3-x} = \binom{3}{x}\frac{5^{3-x}}{216}$$



- Payoff: $Y = h(X) = \begin{cases} 2X & \text{if } X \geq 1 \\ 0 & \text{if } X = 0 \end{cases}$

Matches, X	0	1	2	3	Expected Payoff μ
P[X]	125/216	75/216	15/216	1/216	
Payoff, Y	0	\$2	\$3	\$4	**\$199/216**

(b) Monte Carlo simulation method

i	1	2	3	…	499	500	Average
Z_1	5	2	3	…	1	4	3.52
Z_2	4	6	5	…	3	6	3.48
Z_3	3	2	1	…	5	6	3.53
Matches, X_i	0	1	0	…	0	2	0.501
Payoff, Y_i	0	2	0		0	3	**\$0.924**

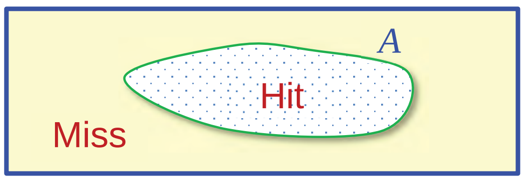
- The average payoff $\mu = E[h(X)]$ is estimated by

$$\hat{\mu}_N = \frac{1}{N}\sum_{i=1}^{N} h(X_i).$$

II. Estimating the Population Proportion

- An important special case arises when the function $h(x)$ to be averaged only has two possible values, conventionally taken to be 1 or 0, or "hit or miss", respectively:

$$h(x) = \begin{cases} 1 & \text{if } \ x \in A \\ 0 & \text{if } \ x \notin A \end{cases}$$



- Suppose that X has probability density function $f(x)$. Then, the average proportion $\pi = E[h(X)]$ is simply the probability $P[X \in A]$ where X is a random variable from $f(x)$.

- In the binary case, we prefer to write the sample average $\hat{\mu}_N$ as the sample proportion $\hat{\pi}_N = N_s/N$, where N_s is the number of successes out of N.

- It can be shown that

$$E[\hat{\pi}_N] = \pi \quad \text{and} \quad Var[\hat{\pi}_N] = \frac{\pi(1-\pi)}{N}.$$

- From the central limit theorem, the confidence interval of π is expressed as

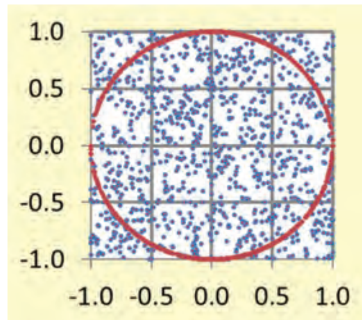
$$\hat{\pi}_N \pm z\sqrt{\frac{\hat{\pi}_N(1-\hat{\pi}_N)}{N}}.$$

- The binary case has one special challenge. Sometimes we get no "hits" in the data. In such a case, the confidence interval is from 0 to 0, which does not make sense.

Ex 1] Monte Carlo Estimation of π

- "Hit or miss"?

1. Draw a circle inscribed in a square.
2. The area of the square is 4.0, and the area of the circle is $\pi r^2 = \pi$.
3. Throw your darts N times.
4. Count the number of hits N_s inside the circle.



- Monte Carlo simulation

1. Generate a pair of *uniform* random numbers X_i between -1 and +1:

$$X_1 = 2\,U_1 - 1 \text{ and } \qquad X_2 = 2\,U_2 - 1.$$

2. If $x_1^2 + x_2^2 < 1$, we hit the target inside the circle:

$$Y = h(X) = \begin{cases} 1 & \text{if } x_1^2 + x_2^2 \leq 1 \\ 0 & \text{otherwise} \end{cases}$$

j	X_1	X_2	$X_1^2+X_2^2$	Inside?
1	-0.94	0.52	1.154	0
2	-0.79	0.65	1.047	0
3	-0.86	-0.24	0.797	1
.	.	.		.
5	=2*rand()-1	=2*rand()-1	= b5^2+c5^2	=if(d5<1,1,0)
.	.	.		.
499	0.97	-0.02		
500	0.66	-0.82		
			Average=	**0.784**

3. The sample proportion is $\hat{\pi}_N = N_s/N = 392/500 = 0.784$, and π is estimated as $4\hat{\pi}_N =$ **3.136**.

Ex 2] Random Walk Process

Suppose that a daily stock price x_t at time t follows a random walk process as follows: $x_t = x_{t-1} + \varepsilon_t$, where the random error terms ε_t are *i.i.d.* random variables from a *normal* distribution with μ=\$2.5 and σ=\$10. If the price at time t=0 is \$120, what is the probability that the price will drop below \$100 at time t=10?

(a) *Analytical solution*

- $x_{10} = x_9 + \varepsilon_{10} = (x_8+\varepsilon_9) + \varepsilon_{10} = x_0 + \sum_{i=1}^{10} \varepsilon_i$.
- Let $S = \sum_{i=1}^{10} \varepsilon_i$. Then, S is normally distributed with $E[S] = 10\mu = \$25$ and $Var[S] = 10\,\sigma^2 = 1000$.
- Thus, $P[x_{10} < 100]$

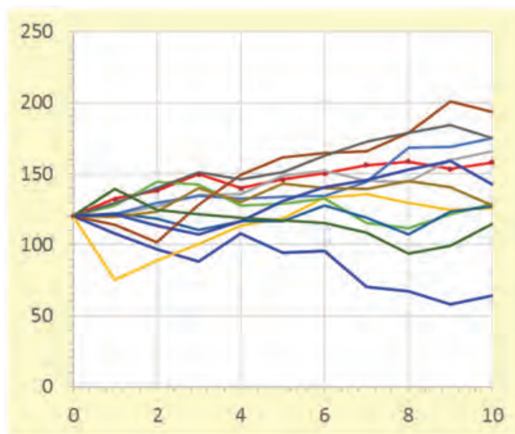
$$= P\left[120 + \sum_{i=1}^{10} \varepsilon_i < 100\right]$$

$$= P[S < -20]$$

$$= P\left[Z < \frac{-20-E[S]}{\sqrt{Var[S]}}\right]$$

$$= P[Z < -1.423\ ]$$

$$= \mathbf{7.74\%}$$



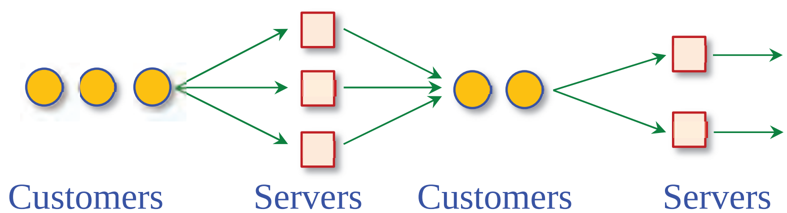
(b) *Monte Carlo method* with N=500 simulation runs

$i \setminus t$	0	1	2	3	4	5	6	7	8	9	10	$I(x_{10}<100)$
1	120	106	108	111	129	132	135	145	154	167	177	0
2	120	110	113	118	122	131	112	113	96	95	98	1
3	120	123	120	142	126	138	136	121	103	119	134	0
-	-	-	-	-	-	-	-	-	-	-	-	-
499	120	128	129	131	137	132	123	116	123	127	132	0
500	120	127	109	111	108	107	116	95	119	113	110	0
											Average =	**0.078**

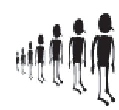
F. Case Study: Waiting Line Model*

* Waiting Line Models

- **The service system**
  - Number of waiting lines: Single or multiple lines
  - Number of servers: Single or multiple servers
  - Number of activities: Single or multiple phases



- **Single-server, single-line, single-phase system**



1. The customers are patient (no balking, reneging, or jockeying) and come from a population that can be considered infinite.
2. Customer arrivals are described by a *Poisson* distribution with a mean arrival rate of λ. (This means that the time between successive customer arrivals follows an *exponential* distribution with an average of $1/\lambda$.)
3. The customer service rate is described by a *Poisson* distribution with a mean service rate of μ. (This means that the service time for one customer follows an *exponential* distribution with an average of $1/\mu$.)
4. The waiting line priority rule used is first-come, first-served.

1. Analytical Solutions: Performance measures for the single-server, single-line, single-phase system

Service system

- *Poisson* arrival rate λ customers / time unit
- *Poisson* service rate μ customers / time unit
- Average utilization rate of the system (i.e., percentage of time the server is busy)

  $$\rho = \frac{\lambda}{\mu}$$

- Average number of customers in the service system

  $$L_s = \frac{\lambda}{\mu - \lambda}$$

- Average number of customers waiting in line

  $$L_w = \rho\, L_s$$

- Average time spent waiting in the system, including service

  $$W_s = \frac{1}{\mu - \lambda}$$

- Average time spent waiting in line

  $$W_w = \rho\, W_s$$

- Probability that *n* customers are in the system at a given time

  $$P_n = (1 - \rho)\, \rho^n$$

Ex] Office Hour:

During Dr. Chun's office hour, students patiently form a single line in front of his office to wait for help. Students are served based on a first-come, first-served priority rule. On average, 15 students per hour arrive at the office. Student arrivals are best described using a Poisson distribution. Dr. Chun can help an average of 20 students per hour, with the service rate being described by an exponential distribution.

(a) Analytical solution

- Arrival rate $\lambda =$ students / hour
- Service rate $\mu =$ students / hour
- Average utilization rate

$$\rho = \frac{\lambda}{\mu} =$$

- Average number of students *in the system*

$$L_s = \frac{\lambda}{\mu - \lambda} =$$

- Average number of students *waiting in line*

$$L_w = \rho\, L_s =$$

- Average time spent waiting *in the system*, including service

$$W_s = \frac{1}{\mu - \lambda} =$$

- Average time spent *waiting in line*

$$W_w = \rho\, W_s =$$

- Probability that 3 students are *in the system* at a given time

$$P_n = (1 - \rho)\, \rho^n =$$

(b) Monte Carlo method

- Arrival rate, λ = 15 students per hour
- Service rate, μ = 20 students per hour

ID i	U_1	Exp X_i	Arrival time	Start time	U_2	Exp Y_i	Finish time	Waiting Time	Idle Time
0			0.000				0.000		
1	0.158	0.123	0.123	0.123	0.529	0.032	0.155	0.032	0.123
2	0.646	0.029	0.152	0.155	0.309	0.059	0.213	0.061	0.000
3	0.229	0.098	0.250	0.250	0.052	0.148	0.398	0.148	0.037
.	.	.	.	.	.	.	.	.	.
498	0.401	0.061	0.662	0.662	0.771	0.013	0.675	0.013	0.007
499	0.404		0.722		0.781	0.012			
500	0.881	0.008			0.435	0.042	0.776		
							Average=	0.199	

- ID for i = ID for i-1 + 1
- U_1 = $rand()$
- X_i = $-ln(U_1)/\lambda$
- Arrival time for i
  = Arrival time for i-1 + X_i
- Start time for i
  = max{Arrival time for i, Finish time for i-1}
- U_2 = $rand()$
- Y_i = $-ln(U_2)/\mu$
- Finish time for i
  = Start time for i + Y_i

- Waiting time for i
  = Finish time for i – Arrival time for i
- Idle time for i
  = Start time for i - Finish time for i-1