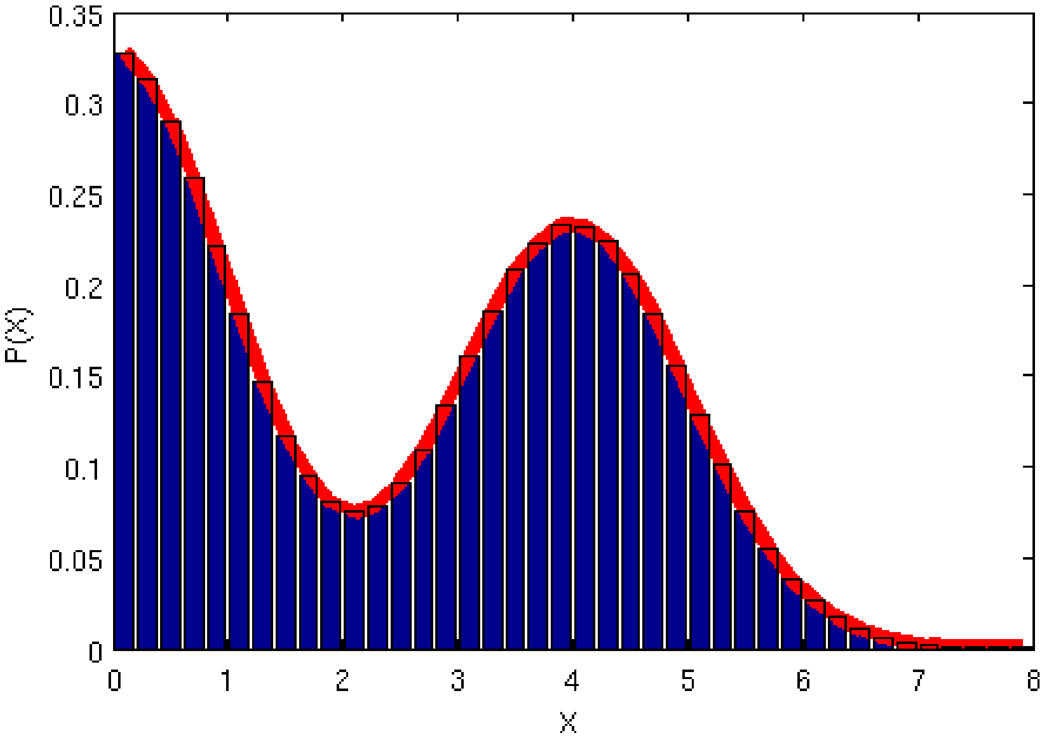
Математическое моделирование РТУ и С

Лекция 12. Формирование реализаций случайных величин с заданным законом распределения



Преподаватель:

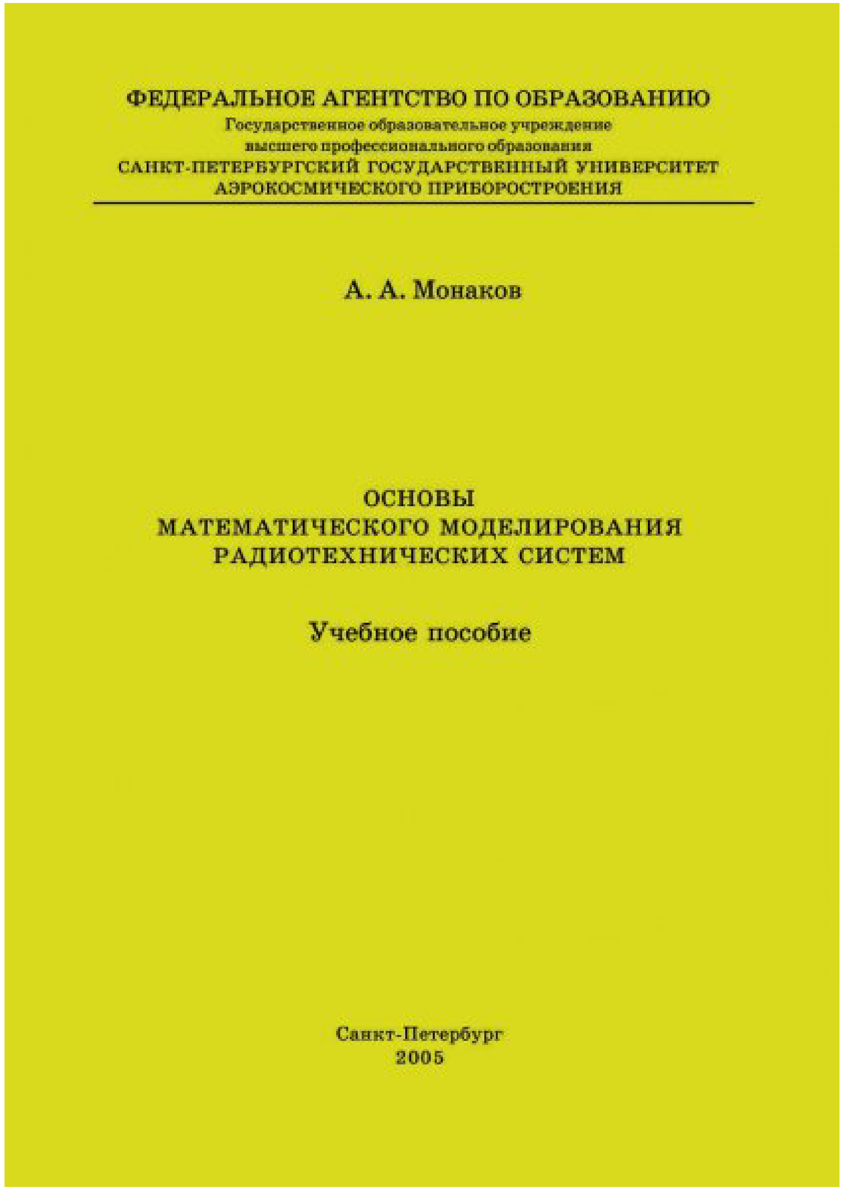
Корогодин Илья

korogodin@srns.ru

Литература

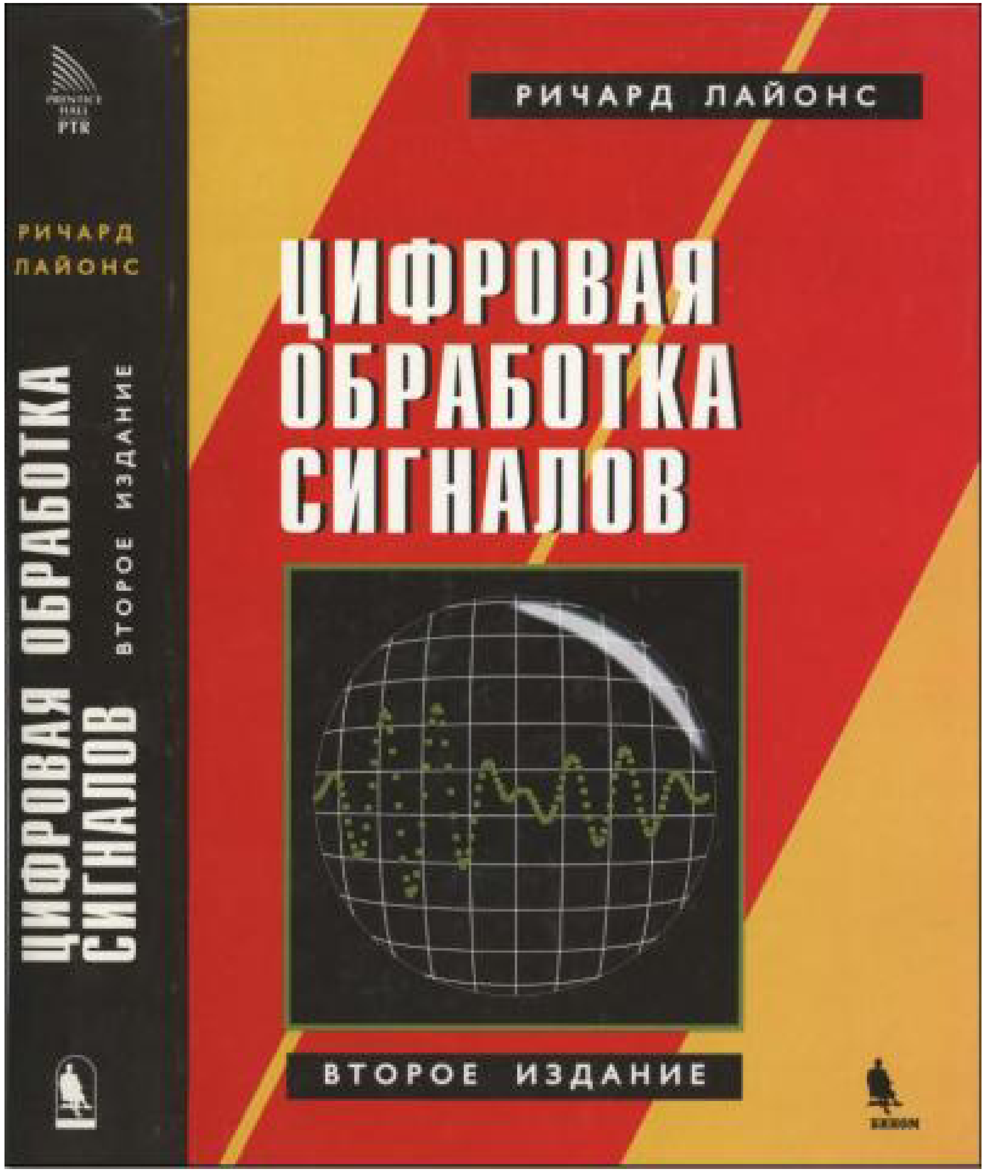
Монаков А.А. Основы математического моделирования радиотехнических систем. Учебное пособие. – СПб.: ГУАП, 2005. – 100с.

Раздел 1.2 Моделирование радиосигналов со случайными параметрами



Литература

Ричард Лайонс - Цифровая обработка сигналов / Understanding Digital Signal Processing, 2006



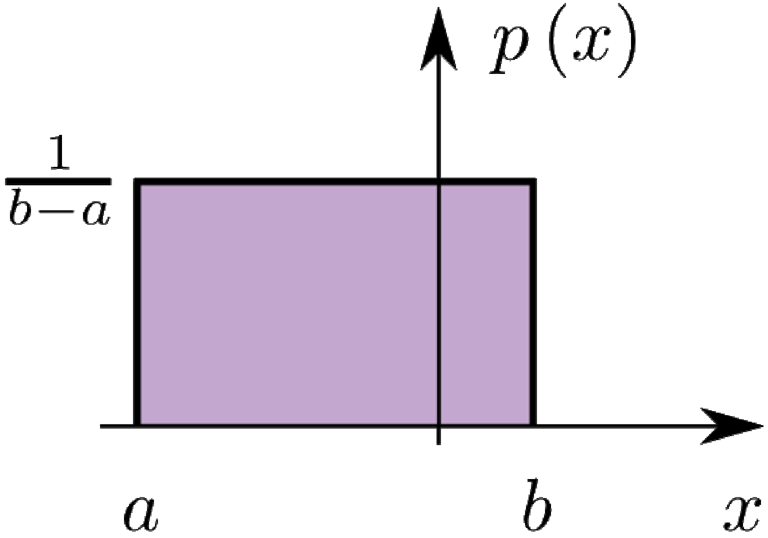
Раздел 13.11. Генерация нормально распределенных случайных сигналов

Равномерное распределение

$x \sim U(a,b)$

Базовое распределение, используется в алгоритмах формирования СВ

$$p(x) = \begin{cases} 0, & x < a, x > b \\ \dfrac{1}{b-a}, & x \ge a,\ x \le b \end{cases}$$



Обычно формируется $U(0,1)$ **на основе рекуррентного алгоритма:**

$$y_n = \mathrm{mod}\left(M \cdot y_{k-1},\ 2^m\right) \qquad x_n = {y_n}/{2^m}$$

m, M **- константы**

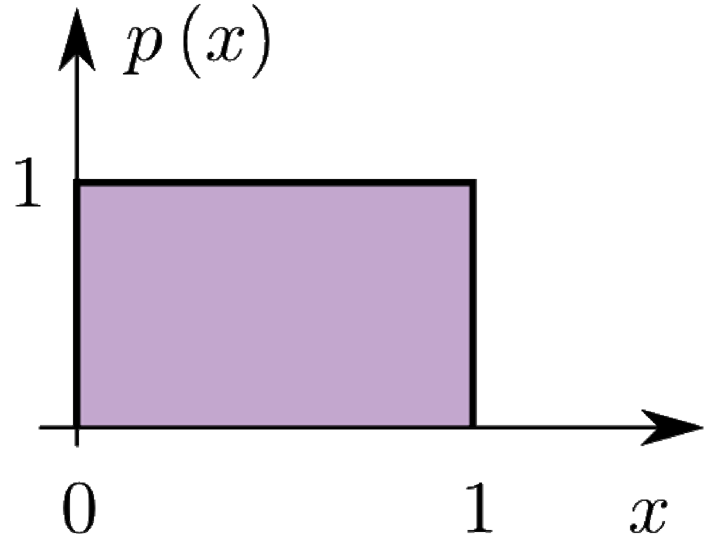
y_1 **- «зерно» (*seed*), обычно инициализируется по счетчику тактов процессора**

Получить $U(a,b)$ **из** $U(0,1)$ **легко:**

$x \sim U(0,1)$

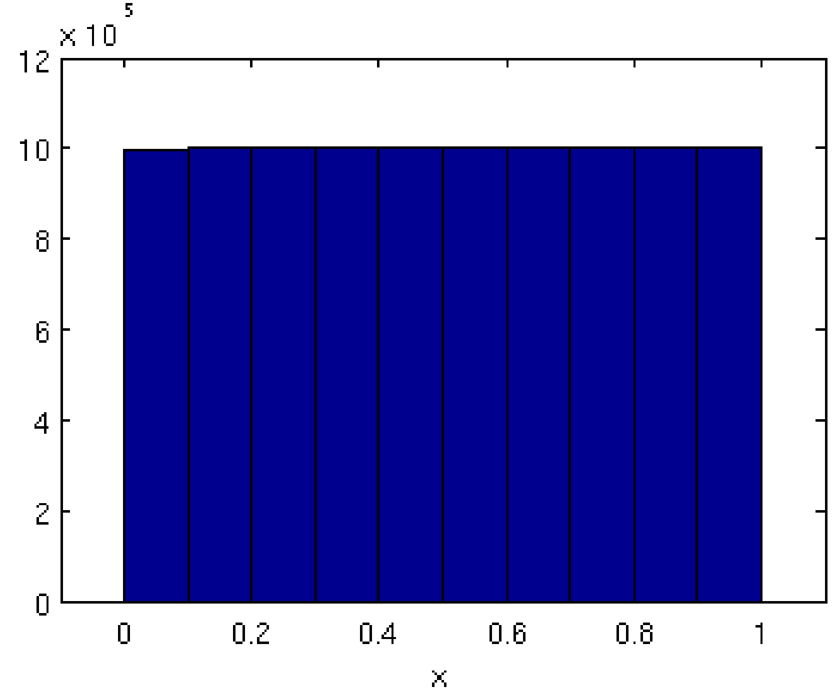
$$\Rightarrow$$

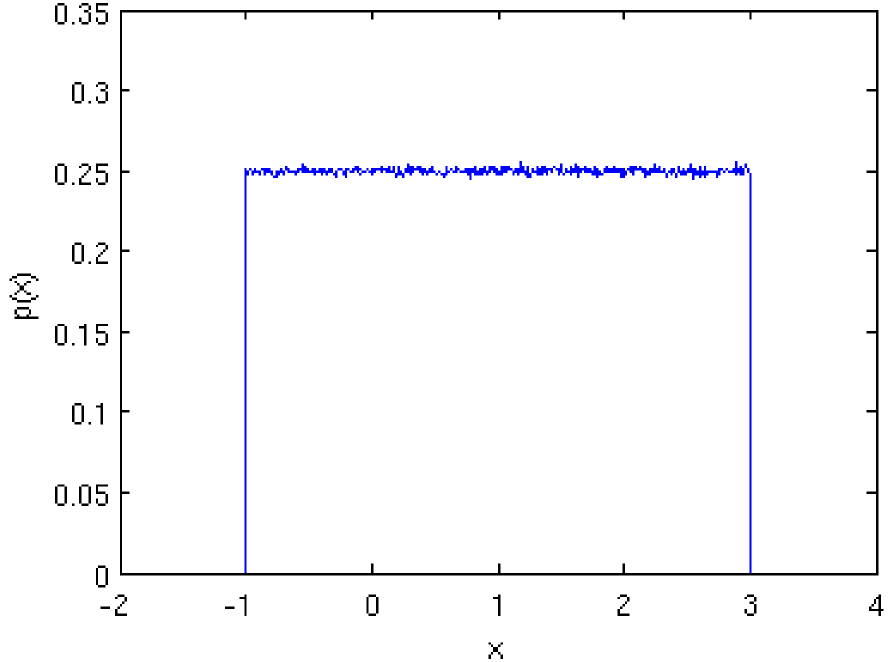
$$x \cdot (b-a) + a \sim U(a,b)$$



Равномерное распределение

```
clear all; close all; clc;
t = 1023; M = 8*t + 3;
m = 32;
N = 1e7;

y = nan(1, N); y(1) = 13;
for i = 2:N
    y(i) = mod(M*y(i-1), 2^m);
end
x = y / 2^m;

figure(1)
hist(x);
xlabel('x');
xlim([-0.1 1.1]);

a = -1; b = 3; x = x*(b-a) + a;
dx = 0.01;
[h, xb] = hist(x, a-dx/2:dx:b+dx/2);
figure(2)
p = h / (sum(h) * dx);
plot(xb, p);
ylim([0 1.1]);
xlabel('x'); ylabel('p(x)');
```





Равномерное распределение

В MATLAB реализуется функцией rand():

```
clear all; close all; clc;
N = 1e7;

rand(1, N);

figure(1)
hist(x);
xlabel('x'); xlim([-0.1 1.1]);

a = -1; b = 3; x = x*(b-a) + a;
dx = 0.01;
[h, xb] = hist(x, a-dx/2:dx:b+dx/2);
figure(2)
p = h / (sum(h) * dx);
plot(xb, p);
ylim([0 1.1]);
xlabel('x'); ylabel('p(x)');
```

rand

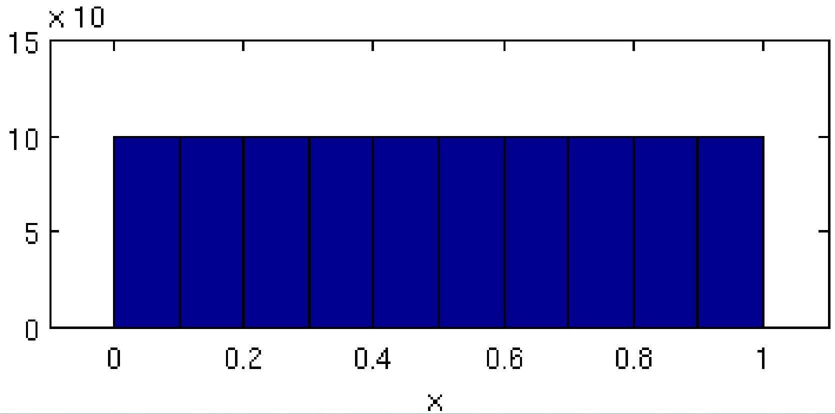
Uniformly distributed pseudorandom numbers

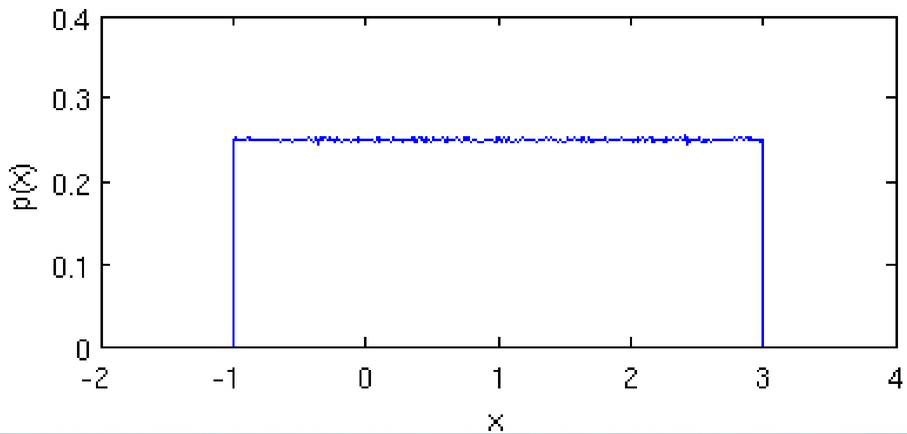
Syntax

```
r = rand(n)
r = rand(m,n)
r = rand([m,n])
r = rand(m,n,p,...)
r = rand([m,n,p,...])
r = rand
r = rand(size(A))
r = rand(..., 'double')
r = rand(..., 'single')
```

Description

`r = rand(n)` returns an n-by-n matrix containing pseudorandom values drawn from the standard uniform distribution on the open interval (0,1). `r = rand(m,n)` or `r = rand([m,n])` returns an m-by-n matrix. `r = rand(m,n,p,...)` or `r = rand([m,n,p,...])` returns an m-by-n-by-p-by-... array. `r = rand` returns a scalar. `r = rand(size(A))` returns an array the same size as **A**.





Метод обратных функций

Задача: сформировать реализацию СВ y с ПВ $p(y)$

1 Находим обратную функцию F^{-1}
от требуемой функции распределения $F(y) = \int\limits_{-\infty}^{x} p(y)dx$

2 Генерируем реализацию
равномерно распределенной СВ x_n^u

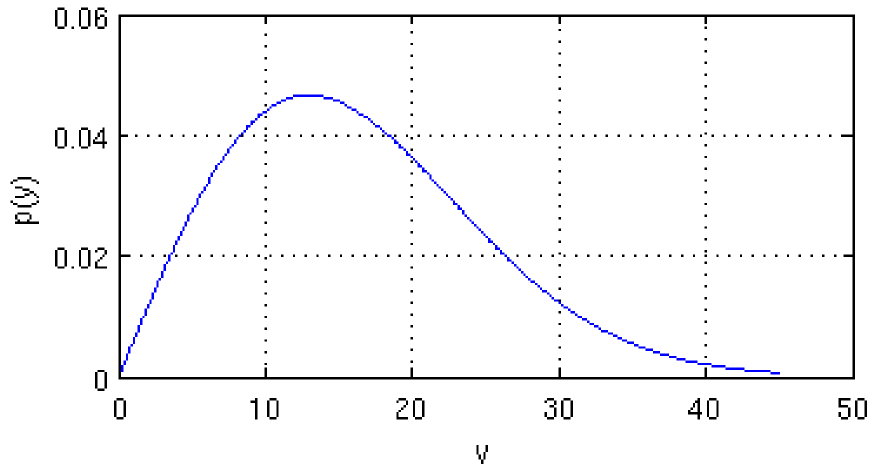
3 Вычисляем $y_n = F^{-1}\left(x_n^u\right)$

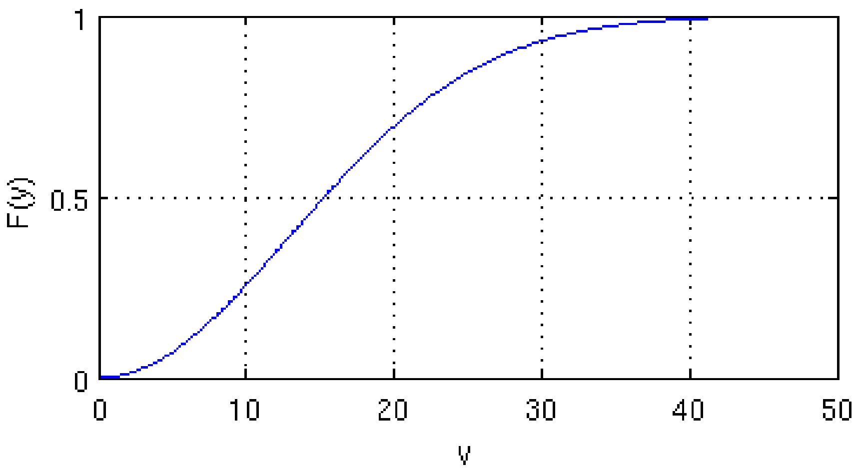
Пример: распределение Релея

$$p(y) = \frac{y}{\sigma^2}\exp\left(-\frac{y^2}{2\sigma^2}\right),\ y > 0$$

$$F(y) = 1 - \exp\left(-\frac{y^2}{2\sigma^2}\right),\ y > 0$$

Возьмем $\sigma = 13$





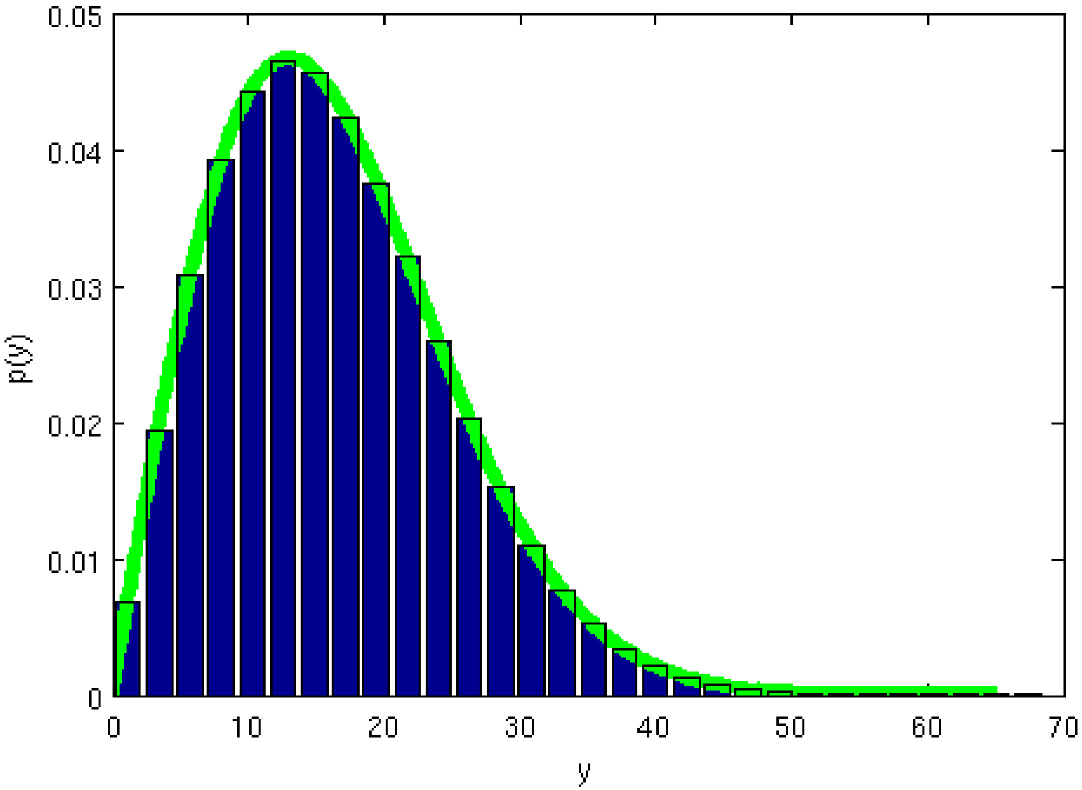
Метод обратных функций

Найдем обратную функцию от функции распределения:

$$F(y)=1-\exp\left(-\frac{y^2}{2\sigma^2}\right)\to 1-F(y)=\exp\left(-\frac{y^2}{2\sigma^2}\right)\to \ln\left(1-F(y)\right)=-\frac{y^2}{2\sigma^2}\to$$

$$\to \sigma\sqrt{-2\ln\left(1-F(y)\right)}=y\to F^{-1}=\sigma\sqrt{-2\ln\left(1-x_n\right)}\to \sigma\sqrt{-2\ln\left(x_n\right)}$$

```
sigma = 13;
N = 1e6;

y_t = 0:0.1:65;
p_t = y_t/sigma^2 .* ...
exp(- y_t.^2 / 2 / sigma^2 );

x_n = rand(1, N);
y_n = sigma*sqrt(-2*log(x_n));

[h, y] = hist(y_n, 30);
inte = sum(h) * (y(2) - y(1));
p = h / inte;
figure(3);
plot(y_t, p_t, 'g', 'LineWidth', 5);
hold on; bar(y, p); hold off
xlabel('y'); ylabel('p(y)');
```



Метод отказов

Он же метод отбора Неймана:

1 Генерируем пару равномерно распределенных СВ x_1 **и** x_2

2 Проверяем, попала ли точка

$$\left[p_{\max} x_2;\, a + x_1(b-a)\right]$$

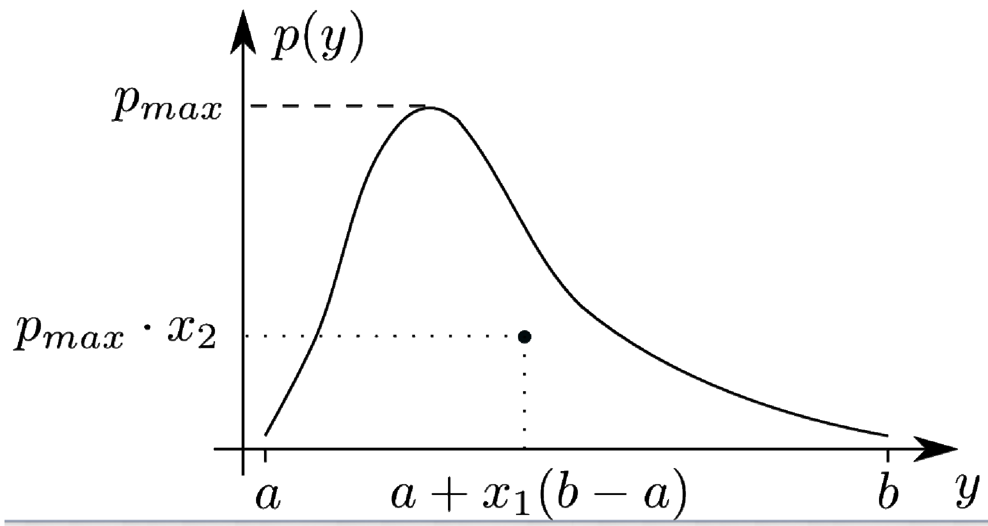
под график ПВ:

$$p_{\max} x_2 < p\Big(a + x_1(b-a)\Big)$$

3 Если попала, то оставляем

$$y = a + x_1(b-a)$$

иначе возвращаемся к шагу 1.



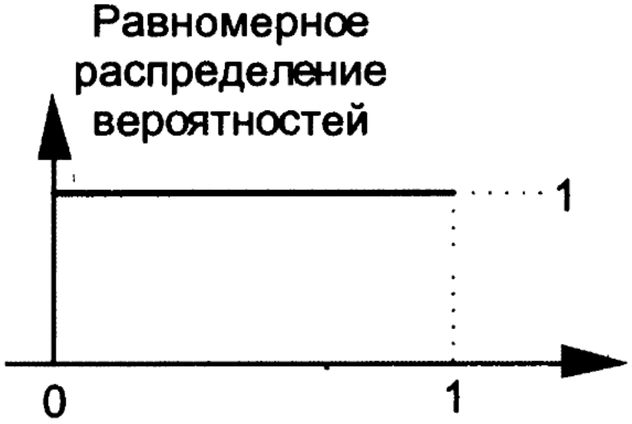

```
clear all; close all; clc;

sigma = 13;
N = 1e6;

y_t = 0:0.1:65;
p_t = y_t/sigma^2 .* ...
exp(- y_t.^2 / 2 / sigma^2 );

a = min(y_t); b = max(y_t);
maxp = max(p_t);
x1 = rand(1, N); x2 = rand(1, N);
x1r = x1*(b-a) + a; x2r = x2*maxp;
ind = find(x2r < x1r/sigma^2 .* ...
exp(- x1r.^2 / 2 / sigma^2));
y_n = x1r(ind);

[h, y] = hist(y_n, 30);
inte = sum(h) * (y(2) - y(1)); p = h / inte;
figure(1); plot(y_t, p_t, 'g', 'LineWidth', 5);
hold on; bar(y, p); hold off
xlabel('y'); ylabel('p(y)');
```

График аналогичен предыдущему

Нормальное распределение

$$x \sim N\left(m,\sigma^2\right) \qquad p(x)=\frac{1}{\sqrt{2\pi\sigma^2}}\exp\left(-\frac{(x-m)^2}{2\sigma^2}\right)$$

Благодаря ЦПТ самое востребованное распределение



?

Вариант «в лоб» - воспользоваться ЦПТ:

$$\sum_{n=1}^{M} x_n^{uniform} \to N\left(\frac{M}{2},\frac{M}{12}\right)$$

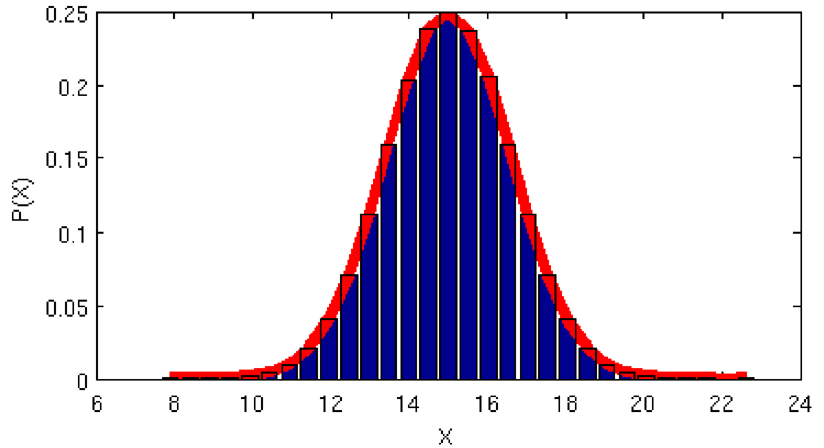
т.к. при суммировании мат. ожидания и дисперсии складываются

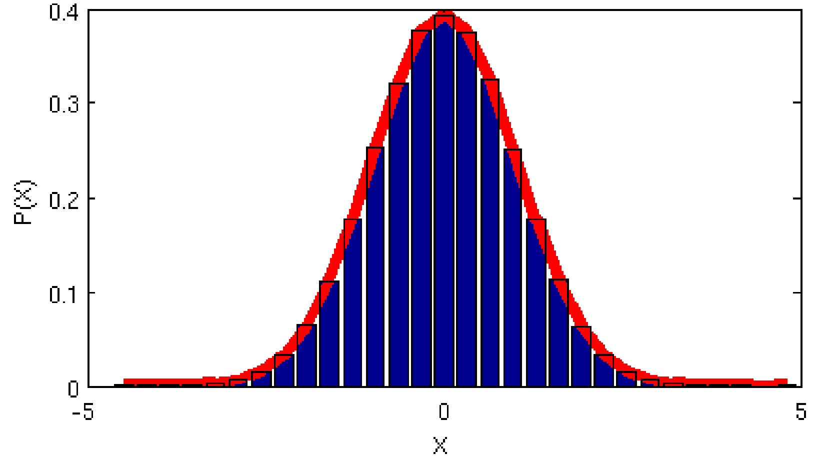
$$x^{norm,1}=\frac{2\sqrt{3}}{\sqrt{M}}\left(\sum_{n=1}^{M} x_n^{uniform}-\frac{M}{2}\right)\sim N(0,1),\ M\sim 30$$

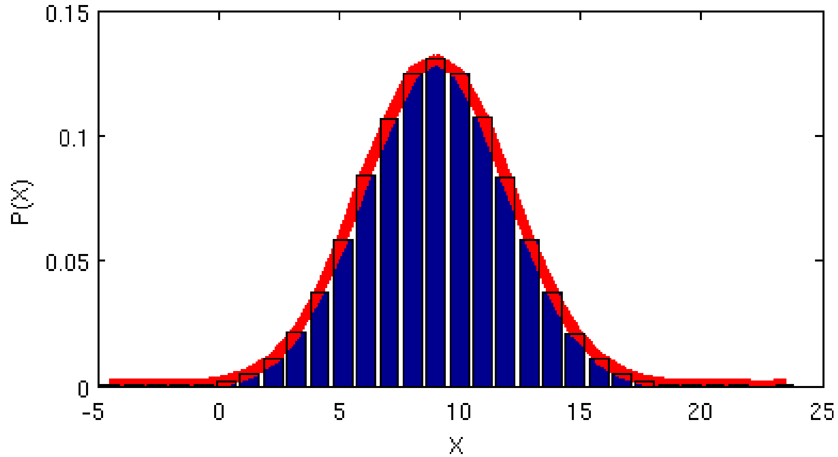
$$x^{norm}=\sigma x^{norm,1}+m=\sigma\frac{2\sqrt{3}}{\sqrt{M}}\left(\sum_{n=1}^{M} x_n^{uniform}-\frac{M}{2}\right)+m\sim N\left(m,\sigma^2\right)$$

Нормальное распределение

```
M = 30; N = 1e6;

x_u = rand(M, N);
x_n = sum(x_u); % суммируем столбцы

[h, x] = hist(x_n, 30);
inte = sum(h) * (x(2) - x(1)); p = h / inte;
figure(1); plot(x, p, 'r', 'LineWidth', 5);
hold on; bar(x, p); hold off
xlabel('X'); ylabel('P(X)');

x_n = (x_n - M/2) / sqrt(M/12);
[h, x] = hist(x_n, 30);
inte = sum(h) * (x(2) - x(1)); p = h / inte;
figure(2); plot(x, p, 'r', 'LineWidth', 5);
hold on; bar(x, p); hold off
xlabel('X'); ylabel('P(X)');

x_n = 3*x_n + 9;
[h, x] = hist(x_n, 30);
inte = sum(h) * (x(2) - x(1)); p = h / inte;
figure(3); plot(x, p, 'r', 'LineWidth', 5);
hold on; bar(x, p); hold off
xlabel('X'); ylabel('P(X)');
```







Нормальное распределение

Преобразование Бокса-Мюллера:

1 Сформировать равномерно распределенные независимые СВ x_1 **и** x_2 $\sim U(0,1)$

2 Вычислить

$$y_1 = \cos(2\pi x_1)\sqrt{-2\ln x_2}$$

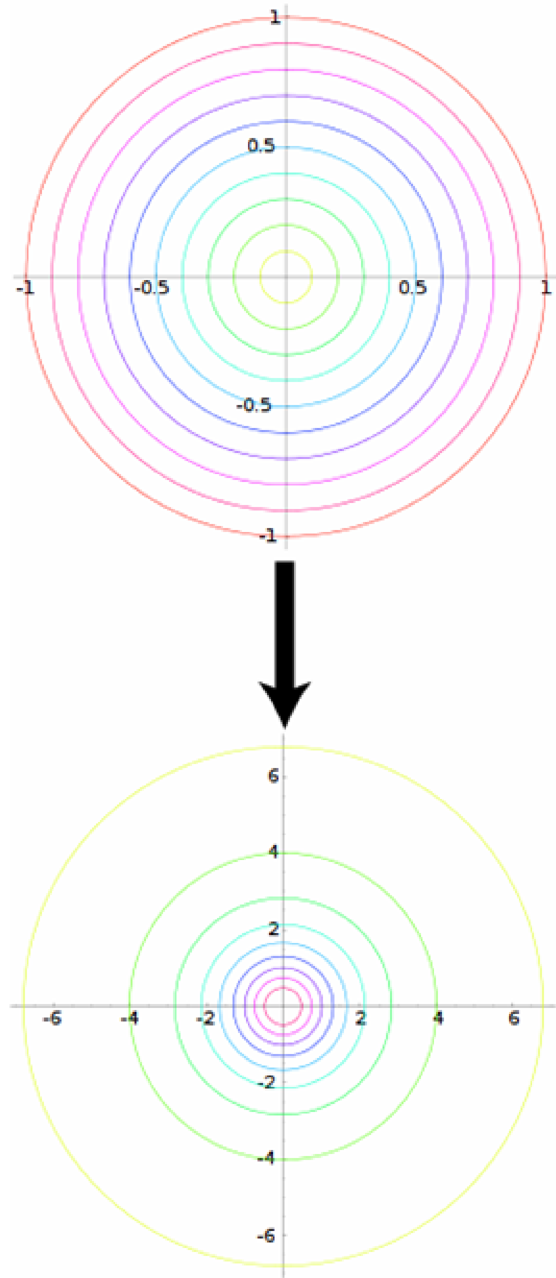
$$y_2 = \sin(2\pi x_1)\sqrt{-2\ln x_2}$$

Получаем независимые СВ $y_1, y_2 \sim N(0,1)$

Интерпретация – Хи-квадрат наизнанку.

Есть второй, не требующий sin, cos, вариант.

```
N = 1e6;  x1 = rand(1, N); x2 = rand(1, N);
y1 = cos(2*pi*x1) .* sqrt(-2 * log(x2));
y2 = sin(2*pi*x1) .* sqrt(-2 * log(x2));

fprintf('Cov:\t%f\t%f\n\t%f\t%f\n', cov(y1, y2));

    >> Cov:        1.002058        0.001024
                   0.001024        1.000542
```



Нормальное распределение

```
M = 30; N = 1e6;

x_n = randn(1, N);

[h, x] = hist(x_n, 30);
inte = sum(h) * (x(2) - x(1)); p = h / inte;
figure(1); plot(x, p, 'r', 'LineWidth', 5);
hold on; bar(x, p); hold off
xlabel('X'); ylabel('P(X)');

x_n = 3*x_n + 9;
[h, x] = hist(x_n, 30);
inte = sum(h) * (x(2) - x(1)); p = h / inte;
figure(2); plot(x, p, 'r', 'LineWidth', 5);
hold on; bar(x, p); hold off
xlabel('X'); ylabel('P(X)');
```

randn

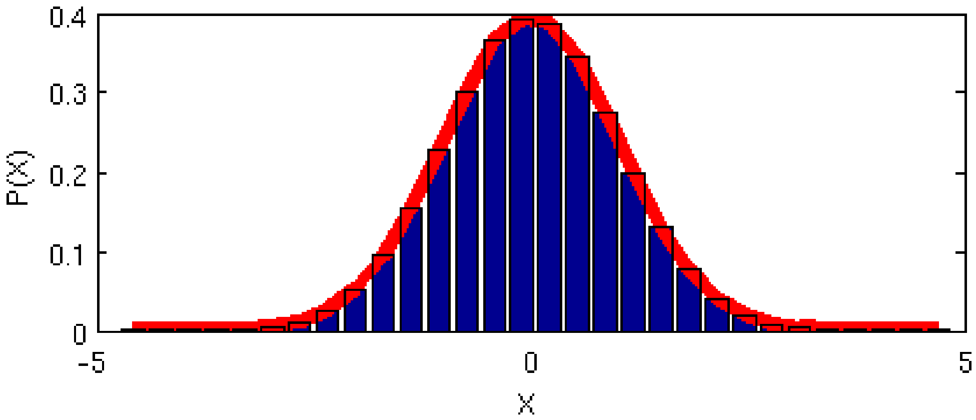
Normally distributed pseudorandom numbers

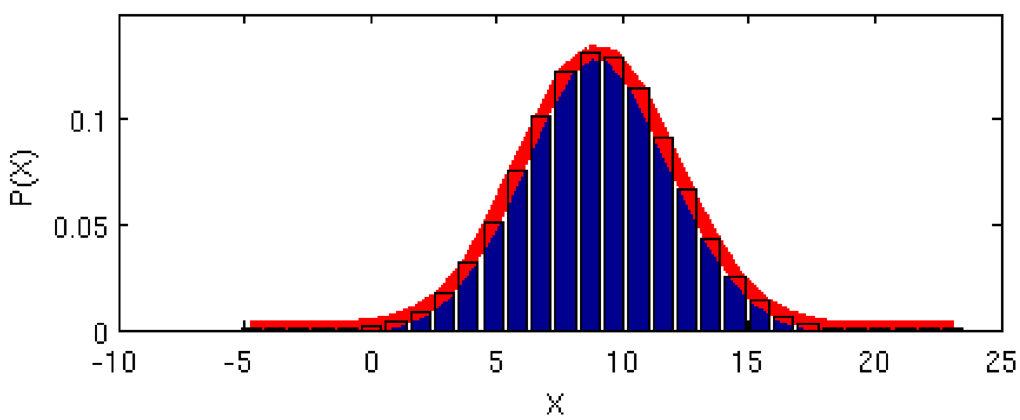
Syntax

```
r = randn(n)
r = randn(m,n)
r = randn([m,n])
r = randn(m,n,p,...)
r = randn([m,n,p,...])
r = randn
r = randn(size(A))
r = randn(..., 'double')
r = randn(..., 'single')
```

Description

`r = randn(n)` returns an n-by-n matrix containing pseudorandom values drawn from the standard normal distribution. `r = randn(m,n)` or `r = randn([m,n])` returns an m-by-n matrix. `r = randn(m,n,p,...)` or `r = randn([m,n,p,...])` returns an m-by-n-by-p-by-... array. `r = randn` returns a scalar. `r = randn(size(A))` returns an array the same size as A.





О ГСЧ в MATLAB

Statistics Toolbox:

betarnd	mvnrnd
binornd	mvtrnd
chi2rnd	nbinrnd
evrnd	ncfrnd
exprnd	nctrnd
datasample	ncx2rnd
frnd	normrnd
gamrnd	pearsrnd
geornd	
gevrnd	poissrnd
gprnd	random
hygernd	randsample
iwishrnd	raylrnd
johnsrnd	slicesample
lhsdesign	trnd
lhsnorm	unidrnd
lognrnd	unifrnd
mhsample	wblrnd
	wishrnd

Инициализируйте ГСЧ!

rng

Control random number generation

Syntax

```
rng(sd)
rng('shuffle')
rng(sd, generator)
rng('shuffle', generator)
rng('default')
s = rng
rng(s)
s = rng(...)
```

Description

> **Note** To use the rng function instead of rand or randn with the 'seed', 'state', or 'twister' inputs, see the documentation on Updating Your Random Number Generator Syntax

rng(sd) seeds the random number generator using the nonnegative integer sd so that rand, randi, and randn produce a predictable sequence of numbers.

rng('shuffle') seeds the random number generator based on the current time so that rand, randi, and randn produce a different sequence of numbers after each time you call rng.