

Gaussian Distributed Random Variables Generation

Di Wu

WINLAB, Rutgers University

{diwu}@winlab.rutgers.edu

1 Introduction

This project explores techniques for creating a random variable X according to a normal distribution. The purpose is to experiment with the properties of probability distributions and convergence of random variables.

The report is organized as follows. In section 2, a well known performance metrics, chi-square test is introduced. In the chi-square test, we quantize the sampling data of the random variable X into bins and compared the observation frequency with the theoretical probability. Thus, the chi-square test could measure the convergence of the random variables. In section 3, we explore four approaches to generate normal distributed random variates and use MATLAB to obtain the chi-square measurements. In section 4, we compare these approaches and conclude.

2 Chi-square test

The accepted test for difference between binned distributions is the *chi-square test*. For continuous data as a function of a single variable, the most generally accepted test is the *kolmogorov-smirnov test*. One can always turn continuous data into binned data, by grouping the events into specified ranges of the continuous variables. Also, there is often considerable arbitrariness as to how the bins should be chosen.

In this project, we choose *chi-square test* as the performance metric to evaluate the four techniques in the next section. Figure 1, we present the perfect quantized probability

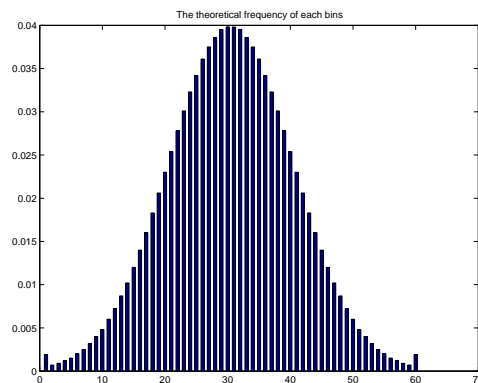


Figure 1: the theoretical frequency in each bins

x	$\Phi(x)$	x	$\Phi(x)$	x	$\Phi(x)$
0.00	0.5000	1.00	0.8413	2.00	0.9773
0.10	0.5398	1.10	0.8643	2.10	0.9821
0.20	0.5793	1.20	0.8849	2.20	0.9861
0.30	0.6179	1.30	0.9032	2.30	0.9893
0.40	0.6554	1.40	0.9192	2.40	0.9918
0.50	0.6915	1.50	0.9332	2.50	0.9938
0.60	0.7257	1.60	0.9452	2.60	0.9953
0.70	0.7580	1.70	0.9554	2.70	0.9965
0.80	0.7881	1.80	0.9641	2.80	0.9974
0.90	0.8159	1.90	0.9713	2.90	0.9981

Table 1: The standard normal CDF $\Phi(x)$.

density function of gaussian distributed random variable. This is the reference that we take when we apply *chi-square test*.

We postulate a "null hypothesis" and try to determine if the observations are consistent with that hypothesis to some level of statistical significance. In this project, we have a set of 1000000 random samples and want to know if their values are consistent with the hypothesis that they are drawn from a normal distribution with zero mean and unit variance.

Suppose that N_i is the number of events observed in the i^{th} bin, and that n_i is the number expected according to some known distribution. Note that the N_i 's are integers, while the n_i 's may not be. Then the chi-square statistic is

$$\chi^2 = \sum_i \frac{(N_i - n_i)^2}{n_i}$$

where the sum is over all bins. A large value of χ^2 indicates that the null hypothesis (that the N_i 's are drawn from the population represented by the n_i 's) is rather unlikely.

Any term j with $n_j = N_j = 0$ in the above equations should be omitted from the sum. A term with $n_j = 0, N_j \neq 0$ gives an infinite χ^2 , as it should be, since in this case the N_i 's cannot possibly be drawn from the n_i 's.

The m-file "chstest.m" in the Appendix illustrates the details: According to the Table 1, we quantize the sampling data of the random variables into 60 bins. Then we can generate the observed histogram and the histogram predicted by our hypothesis. From chetest.m, it returns the "chi-square measurement" value being used as a diagnostic (small means good fit).

3 Normal distributed random variates generation

In this section, normal distributed random variate is generated by employing four different approaches.

3.1 Sums of Uniform Random Variables

We have learned from the Central Limit Theorems that the average of a set of iid random variables tends toward the normal distribution. Generate n uniform 0,1 random variables and add/scale them according to the Central Limit Theorem to generate a $N(0, 1)$ random variate.

N	chi-square value	df
2	0.0322	59
3	0.0088	59
4	0.0043	59
5	0.0025	59
6	0.0018	59
7	0.0012	59
8	8.9530e-004	59
9	7.5058e-004	59
10	6.9693e-004	59
100	3.8444e-004	59
1000	1.0828e-004	59
10000	7.1212e-005	59

Table 2: Sums of Uniform Random Variables with different N .

The random variable X is generated from the "add/scale" of a large number of uniform random variables:

$$X_n = \frac{\sum_{i=1}^N U_i - E\{\sum_{i=1}^N U_i\}}{\sqrt{Var\{\sum_{i=1}^N U_i\}}}$$

Since $E\{U_i\} = 0.5$ and $Var\{U_i\} = 1/12$, we get

$$X_n = \frac{\sum_{i=1}^N U_i - N/2}{\sqrt{N/12}}$$

where, N is the number of uniform distributed random variate which we sum up.

In the experiment, we select $N = 2, 3, 4, 5, 6, 7, 8, 9, 10, 10^2, 10^3, 10^4$ respectively.

Note that, when we increase the N , the generated random variate become more and more converge to the normal distributed random variable. The tradeoff here is, if we increase the N , the generation time goes up. When $N = 7$, it is enough to have almost the same histogram, even at the tail, with the theoretical one. So, **we prefer chose $N = 7$!**

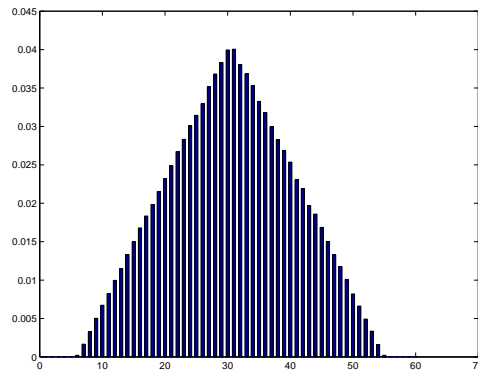


Figure 2: The observed frequency in each bins with $N=2$. Both the shape and the tail do not matched with theoretical one.

3.2 Box-Muller

The following technique is known as the *Box-Muller method*. Let U_1 and U_2 be iid uniform random variables with values from 0, 1. Set $X_1 = \sqrt{-2\ln U_1} \cos(2\pi U_2)$ and

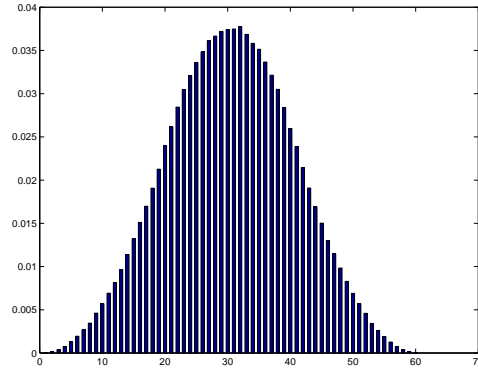


Figure 3: The observed frequency in each bins with $N=3$. The shape almost matches with the theoretical one, but the tail does not.

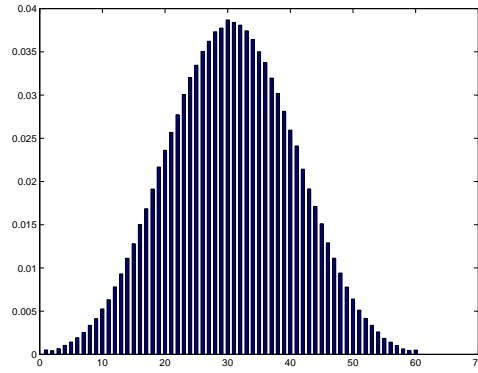


Figure 4: The observed frequency in each bins with $N=4$. The shape looks good, but the tail is still bad.

$X_2 = \sqrt{-2\ln U_1} \sin(2\pi U_2)$. It can be shown (you don't have to do this) that X_1 and X_2 are iid $N(0, 1)$ random variates.

3.3 Polar Technique

Let U_1 and U_2 be iid uniform Random variables with values from 0, 1. The procedure for the polar technique is (a) Let $V_i = 2U_i - 1$, and define $W = V_1^2 + V_2^2$. (b) If $W > 1$ then go back to step(a). Else, let $Y = \sqrt{(-2\ln U_1)/W}$, $X_1 = V_1 Y$ and $X_2 = V_2 Y$. Then X_1 and X_2 are iid $N(0, 1)$.

Type	chi-square measurement	df
<i>Cos()</i>	7.7237e-005	59
<i>Sin()</i>	6.4280e-005	59

Table 3: Box-Muller using *Cos()* and *Sin()*

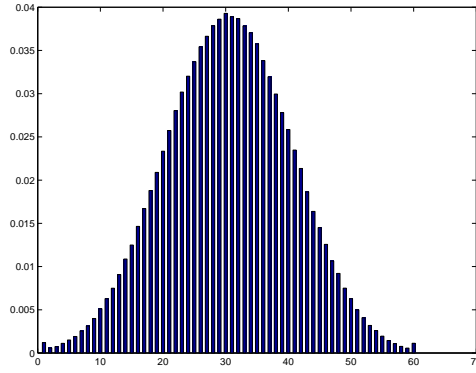


Figure 5: the observed frequency in each bins with $N=7$. We prefer $N = 7$, since even the tail looks good.

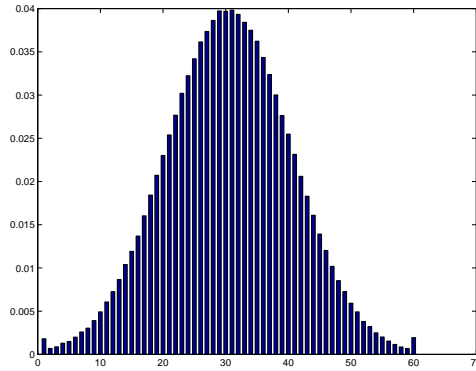


Figure 6: The observed frequency in each bins with $\text{Cos}()$.

3.4 Inverse Distribution

A random variate X can be generated by using the inverse CDF function. Using the following approximation for the $Q(x)$ function:

$$Q(x) \simeq \left[\frac{1}{\left(1 - \frac{1}{\pi}\right)x + \frac{1}{\pi}\sqrt{x^2 + 4\pi^2}} \right] \frac{1}{\sqrt{2\pi}} e^{-x^2/2}, x > 0$$

construct a numerical representation for the (approximate) inverse CDF F_X^{-1} . Using the inverse CDF you can create a $N(0, 1)$ random variate.

Wrong! In the above approximation equation, we set $x \approx 0$, then the result should approximately be 0.5. But we failed to get the desired value, which means, the approximation function is not correct!

Alternate Way I will use the quantized $\Phi(\cdot)$ function instead of the approximate one.

The steps to generate the Gaussian Random Variable is:

Type	chi-square measurement	df
<i>PolarTest1</i>	9.9293e-005	59
<i>PolarTest2</i>	6.6741e-005	59
<i>PolarTest2</i>	10.590e-005	59

Table 4: Polar Techniques to generate $N(0, 1)$

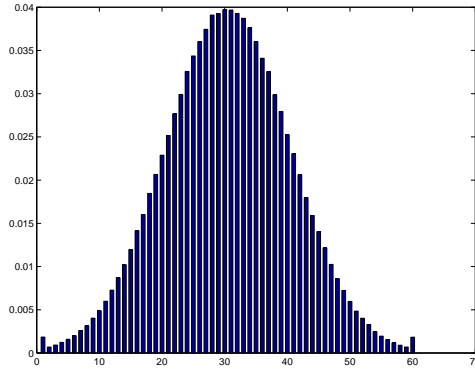


Figure 7: The observed frequency in each bins with $Sin()$.

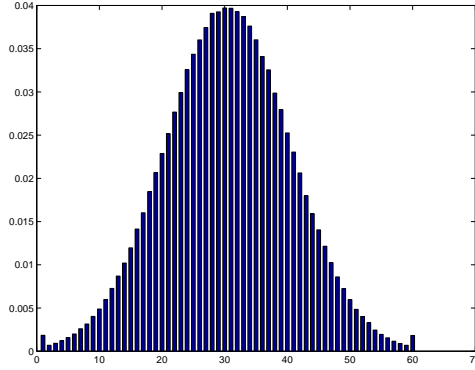


Figure 8: The observed frequency in each bins with polar technique.

1. Generate $U = U(0, 1)$ a uniform r.v. with values from $[0, 1]$.
2. Return $X = F^{-1}(U)$

Figure 9 shows the distribution of the Gaussian random variable and Table 6 depicts the *chi-square test* of this approach.

4 Conclusion

In this project, by employing four approaches, we create a random variable X according to a normal distribution. Then, *chi-square test* was used as the performance metric

x	$\Phi(x)$	x	$\Phi(x)$	x	$\Phi(x)$
0.00	0.5000	1.00	0.8413	2.00	0.9773
0.10	0.5398	1.10	0.8643	2.10	0.9821
0.20	0.5793	1.20	0.8849	2.20	0.9861
0.30	0.6179	1.30	0.9032	2.30	0.9893
0.40	0.6554	1.40	0.9192	2.40	0.9918
0.50	0.6915	1.50	0.9332	2.50	0.9938
0.60	0.7257	1.60	0.9452	2.60	0.9953
0.70	0.7580	1.70	0.9554	2.70	0.9965
0.80	0.7881	1.80	0.9641	2.80	0.9974
0.90	0.8159	1.90	0.9713	2.90	0.9981

Table 5: The numerical result of $\Phi(x)$.

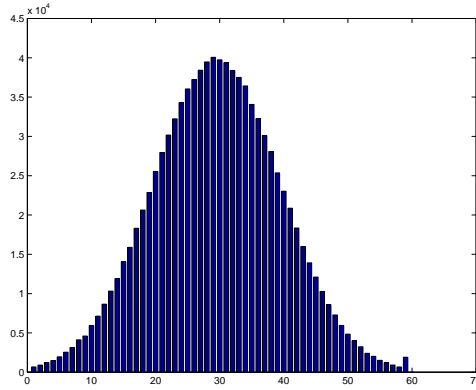


Figure 9: The observed frequency in each bins with Inverse F technique.

<i>Type</i>	chi-square measurement	df
Inverse F	0.0142	59

Table 6: Inverse F approach

to evaluate the above techniques.

From the table 7, the approach with the least *chi-square test measurement* is the desired one. The "Box-Muller", "Polar Technique", and the "sums of uniform RV" with higher summation number are good in chi-square test.

The reasons why inverse distribution method has highest chi-square test measurement are complex. It is due to the which approximation function you chosen, due to the precision of the numerical solution of the inverse function, due to the step of size of your quantization. But anyway, it is not a so good method, when the inverse function is hard to present.

It is clear that when increase the summation number, we could achieve better performance RV in "Sums of Uniform RV" approach. But with the higher summation number, you need more time to do the summation. It trades the higher performance with generating time. In this project experiments condition, we prefer choose the summation number to be 7.

Overall, under the experiments conditions of this project, we **prefer "Box-Muller" approach**. It has higher performance with less generating time.

5 Appendix

The simulation source codes:

<i>Type</i>	chi-square measurement	df
<i>InverseDistribution</i>	0.0142	59
<i>PolarTest1</i>	9.9293e-005	59
<i>PolarTest2</i>	6.6741e-005	59
<i>PolarTest2</i>	10.590e-005	59
<i>Box – MullerCos()</i>	7.7237e-005	59
<i>Box – MullerSin()</i>	6.4280e-005	59
Sum 2 Uniform RV	0.0322	59
Sum 3 Uniform RV	0.0088	59
Sum 4 Uniform RV	0.0043	59
Sum 5 Uniform RV	0.0025	59
Sum 6 Uniform RV	0.0018	59
Sum 7 Uniform RV	0.0012	59
Sum 8 Uniform RV	8.9530e-004	59
Sum 9 Uniform RV	7.5058e-004	59
Sum 10 Uniform RV	6.9693e-004	59
Sum 100 Uniform RV	3.8444e-004	59
Sum 1000 Uniform RV	1.0828e-004	59
Sum 10000 Uniform RV	7.1212e-005	59

Table 7: Inverse F approach

```

%-----
% From Box-Muller Technique to generate a N(0,1)
% random variate.
%-----
%                               Di Wu
%   WINLAB, Rutgers University    10/20/02
%-----
Total_variate=1000000;
%-----Set bins
Phi=[...
0.5000 0.5398 0.5793 0.6179 0.6554...
0.6915 0.7257 0.7580 0.7881 0.8159...
0.8413 0.8643 0.8849 0.9032 0.9192...
0.9332 0.9452 0.9554 0.9641 0.9713...
0.9773 0.9821 0.9861 0.9893 0.9918...
0.9938 0.9953 0.9965 0.9974 0.9981];

K=2*length(Phi);

Phi2=fliplr(1-Phi);
Phi3=[Phi2 Phi(2:length(Phi))];

n(1)=Phi3(1);
for ii=2:K-1,
n(ii)=Phi3(ii)-Phi3(ii-1);
end
n(K)=1-Phi3(K-1);
N=zeros(size(n));
%-----Generate RV

for qq=1:Total_variate,

Uni=rand(1,2);
x=sqrt(-2*log(Uni(1)))*cos(2*pi*Uni(2));
%x=sqrt(-2*log(Uni(1)))*sin(2*pi*Uni(2));

%-----Count the number of RV
x=10*x;
S=sign(x);

if S==-1,
if ceil(x)<=-29
Index=1;
else
Index=30+ceil(x);
end
else
if ceil(x)>=30
Index=60;
else
Index=30+ceil(x);
end
end

N(Index)=N(Index)+1;
end

```

```

function [chsq,prob,df]=chstest(bins,ebins,knstrn)
% CHSTEST Chi-Square Significance Test
% [CHSQ,PROB,DF]=chstest(BINS,EBINS,KNSTRN) performs a Chi-Square
% Test on the data histogram in BINS to the distribution in
% EBINS. KNSTRN is the number of constraints used in the model
% after the data was collected, usually zero. CHSQ is Chi-Squared
% value, and PROB is the "significance". A small value of PROB
% indicates a significant difference between the distributions
% BINS and EBINS. DF is the number of degrees of freedom.

if nargin<3,
    knstrn=0;
end
if any(ebins<0),
    error('Non-positive expected number in ebins')
end

I=find((bins~=0) | (ebins~=0)); % omit 0=nj=Nj from sum
nbins=length(I);
df=nbins-1-knstrn
chsq=sum(((bins(I)-ebins(I)).^2)./ebins(I))
prob=(1-gammainc(chsq/2,df/2)*gamma(df/2))/gamma(df/2)

%-----
% From the Inversed F(x) to generate a N(0,1)
% random variate.
%-----
% Di Wu
% WINLAB, Rutgers University 10/20/02
%-----
Total_variate=1000000;

%-----Set bins
Phil=[...
0.5000 0.5398 0.5793 0.6179 0.6554...
0.6915 0.7257 0.7580 0.7881 0.8159...
0.8413 0.8643 0.8849 0.9032 0.9192...
0.9332 0.9452 0.9554 0.9641 0.9713...
0.9773 0.9821 0.9861 0.9893 0.9918...
0.9938 0.9953 0.9965 0.9974 0.9981];

K=2*length(Phil);

Phi2=fliplr(1-Phil);
Phi3=[Phi2 Phil(2:length(Phil))];

n(1)=Phi3(1);
for ii=2:K-1,
n(ii)=Phi3(ii)-Phi3(ii-1);
end
n(K)=1-Phi3(K-1);
N=zeros(size(n));
%-----Generate RV
for qq=1:Total_variate,
    Uni=rand(1,1);

```

```

    if Uni<Phi3(1)
        z_n=(1-30)/10;
    else if Uni>Phi3(59)
        z_n=(59-30)/10;
    else
        for kk=1:58
            if Uni>Phi3(kk) & Uni<Phi3(kk+1),
                z_n=(kk-30)/10;
            end
        end
    end
    x=z_n;
%-----Count the number of RV
x=10*x;
S=sign(x);
if S==-1,
    if ceil(x)<=-29
        Index=1;
    else
        Index=30+ceil(x);
    end
else
    if ceil(x)>=30
        Index=60;
    else
        Index=30+ceil(x);
    end
end
N(Index)=N(Index)+1;
end
end

%-----
% From the Polar Technique to generate a N(0,1)
% random variate.
%-----
%                               Di Wu
%      WINLAB, Rutgers University      10/20/02
%-----
Total_variate=1000000;
%-----Set bins
Phil=[...
0.5000 0.5398  0.5793  0.6179  0.6554...
0.6915 0.7257  0.7580  0.7881 0.8159...
0.8413 0.8643  0.8849  0.9032  0.9192...
0.9332  0.9452 0.9554  0.9641  0.9713...
0.9773  0.9821 0.9861  0.9893  0.9918...
0.9938 0.9953  0.9965  0.9974  0.9981];
K=2*length(Phil);
Phi2=fliplr(1-Phil);
Phi3=[Phi2 Phil(2:length(Phil))];
n(1)=Phi3(1);
for ii=2:K-1,
n(ii)=Phi3(ii)-Phi3(ii-1);
end
n(K)=1-Phi3(K-1);

```

```

N=zeros(size(n));
%-----Generate RV
Count=0;
while Count < Total_variate,
    Uni=rand(1,2);
    V=2*Uni-1;
    W=V(1).^2+V(2).^2;
    if W<1,
        Count=Count+1;
        Y=sqrt(-2*log(W)/W);
        x=V(1)*Y;
%-----Count the number of RV
        x=10*x;
        S=sign(x);
        if S==-1,
            if ceil(x)<=-29
                Index=1;
            else
                Index=30+ceil(x);
            end
        else
            if ceil(x)>=30
                Index=60;
            else
                Index=30+ceil(x);
            end
        end
        N(Index)=N(Index)+1;
    end
end

%-----
% From the Central Limit Theorems that the average
% of a set of iid random variables tends toward the
% normal distribution. Generate n uniform u(0,1)
% random variables and add/scale them according to
% the Central Limit Theorem to generate a N(0,1)
% random variate.
%-----
%           Di Wu
%   WINLAB, Rutgers University      10/20/02
%-----
clear;
Total_variate=1000000;
%-----Set bins
Phil=[...
0.5000 0.5398    0.5793    0.6179    0.6554...
0.6915 0.7257    0.7580    0.7881 0.8159...
0.8413 0.8643    0.8849    0.9032    0.9192...
0.9332    0.9452 0.9554    0.9641    0.9713...
0.9773    0.9821 0.9861    0.9893    0.9918...
0.9938 0.9953    0.9965    0.9974    0.9981];
K=2*length(Phil);
Phi2=fliplr(1-Phil);
Phi3=[Phi2 Phil(2:length(Phil))];

```

```

n(1)=Phi3(1);
for ii=2:K-1,
n(ii)=Phi3(ii)-Phi3(ii-1);
end
n(K)=1-Phi3(K-1);
N=zeros(size(n));
%-----Generate RV
Sum_all=0;
Var_all=0;
for qq=1:Total_variate,
Sample_N=7;
Uni=rand(1,Sample_N);
S_n=sum(Uni);
Z_n=(S_n-Sample_N*0.5)/sqrt(Sample_N/12);
Sum_all=Z_n+Sum_all;
Var_all=Z_n^2+Var_all;
x=Z_n;
%-----Count the number of RV
x=10*x;
S=sign(x);
if S==-1,
    if ceil(x)<=-29
        Index=1;
    else
        Index=30+ceil(x);
    end
else
    if ceil(x)>=30
        Index=60;
    else
        Index=30+ceil(x);
    end
end
N(Index)=N(Index)+1;
end

mean=Sum_all/1000000
var= Var_all/1000000

```