Assignment 4

Name: Jian Wang FSUID: Jw09r

1.Write a computer code for the Halton sequence. The input should be the dimension s, and n, and the output should be the nth Halton vector where the bases are the first s prime numbers. Then, generate 1000 numbers from the standard normal distribution using the Moro's method and the Box-Muller method. What is the constructive dimension for each method? What is the dimension of the sequence you use for each method? Draw two histograms for each data set. **Do** they "look" right?

Answer:

The van der Corput sequence and its generalization, the Halton se-quence, is a well-known low-discrepancy sequence. The van der Corput sequence $\phi_b(n)$; n = 1,2,…,n is defined as follows:

If

$n = (a_{k,\dots,}a_1, a_0)_b = a_0 + a_1 b + \cdots + a_k b^k$

Then $\phi_b(n) = (.a_0 a_1 \dots a_k)_b = \frac{a_0}{b} + \frac{a_1}{b^2} + \frac{a_2}{b^3} + \cdots + \frac{a_k}{b^{k+1}}$

The Halton sequence is a generalization of the van der Corput sequence to higher dimensions. The s-dimensional Halton sequence in the bases b1,…, bs is defined as $(\phi_{b1}(n), \dots, \phi_{bs}(n))_{n=1}^{\infty}$ The Halton sequence is u.d. mod 1 if its bases b1,…, bs are relatively prime. Usually, these numbers are chosen simply as prime numbers, i.e., bn is taken as the nth prime number.

I used the **one dimension halton method** to generate the random data for the Moro method.

The data of the 1000 numbers in each interval is as follows:

x	-3.25	-3	-2.75	-2.5	-2.25	-2	-1.75	-1.5	-1.25	-1	-0.75	-0.5	-0.25	0
No	1	2	3	6	11	18	26	40	53	68	82	92	99	99
x	0.25	0.5	0.75	1	1.25	1.5	1.75	2	2.25	2.5	2.75	3		
No	93	81	69	52	40	26	17	11	6	3	2	0		

The histogram is as follows:

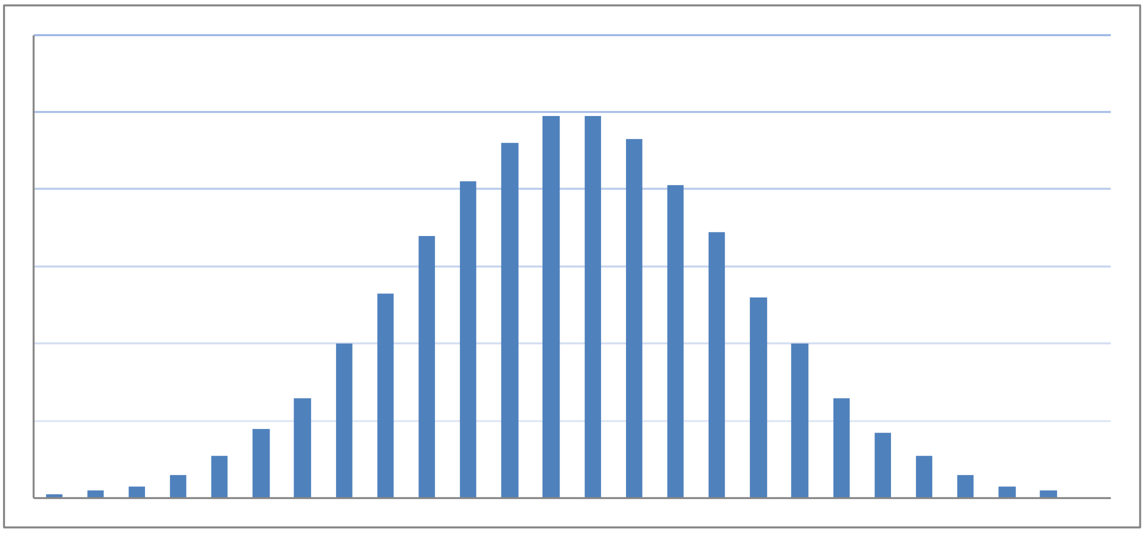

Figure 1 The Moro data

I used the **two dimensions Hilton method** to generate the numbers of the Box-Muller method.

The data of the 1000 numbers in each interval is as follows:

x	-3.25	-2.75	-2.25	-1.75	-1.25	-0.75	-0.25	0.25	0.75	1.25	1.75
N	3	8	24	61	116	172	235	172	112	62	24

The histogram is as follows:

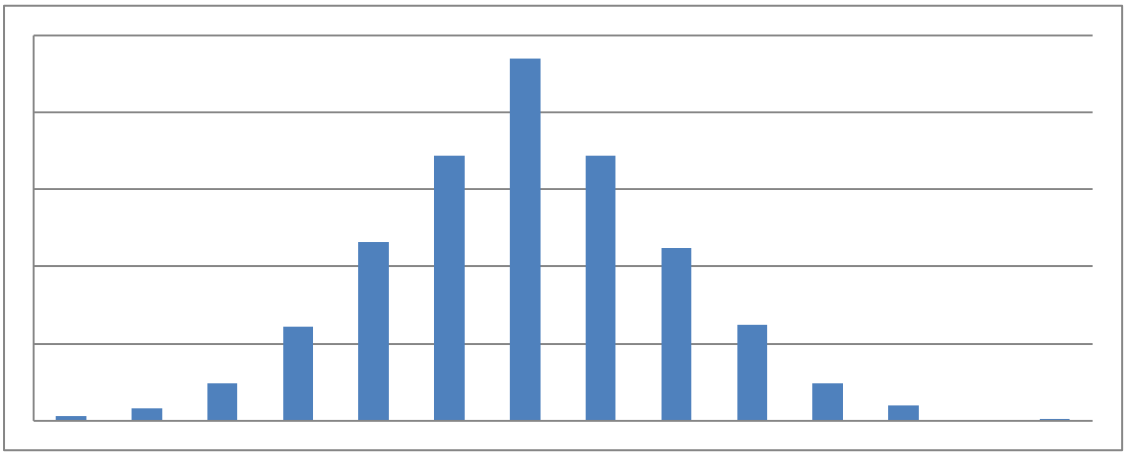

Figure 2 The BM data

Conclusion:

- **The dimension of the Halton method for the BM method is two and is one for the Moro method**
- **From the data and the histogram, the method of Halton for generating the normal distribution via the BM and Moro method is good.**

2. Write a computer code for the Faure sequence, and the scrambled Faure sequence (as explained in the Mathematica example)

(a) Use scrambled Faure sequences to obtain 40 estimates for the price of a European call option with parameters: T = expiry = 1, K =exercise price = 50; r = 0:1; σ = 0:3; S0 = 50: For each estimate, use N = 10 000 price paths. To price the option, you will simulate the GBM with u= r, and using 10 time steps t0 = 0; t1 = Δt; t2 =2Δt,…, t10 = 10Δt = T: Use the Box-Muller method in generating the stock price paths. (Note that the European call option can actually be estimated by directly generating the price at expiry, so there is no need to generate the complete price path. Nevertheless, this problem prepares you for the pricing of path dependent options.

(b) Do part (a) using Moro.s method.

(c) Compute the Black-Scholes-Merton price of the option, i.e., the exact option price.

(d) Compare the accuracy of the estimates you obtained in parts (a) and (b) as follows: Each set of 40 estimates should be distributed according to the normal distribution whose mean is the true option price you found in part (c), and an unknown variance. Apply the Anderson-Darling statistic to test the data you obtained in parts (a) and (b), for this hypothesis. Consult Stephens.paper on Anderson-Darling statistic for critical points. Which one of the cases discussed in the paper applies to this problem? What are your conclusions for each data set?

(e) Read the paper by Joy, Boyle, and Tan (on Blackboard), in particular, the footnote on page 930. Do you agree with the authors that the Box-Muller method is worse than Moro's method?

Answer:

(a) With the help of the Matlab, I generate the scrambled Faure sequence(the code is as attachment) Since there are ten time sequences, so I choose the dimension as 10 to generate the different stock price in each path.

Firstly, I generate 10000 scrambled Faure sequences o f 10 dimensions, then used the Box-Muller method to generate the normal distribution data $Z_{i,j}$, finally

Used the formula:

$$S^j(t_i) = S^j(t_{i-1})\exp\left(\left(u - \frac{\sigma^2}{2}\right)(t_i - t_{i-1}) + \sigma\sqrt{t_i - t_{i-1}}Z_{i,j}\right)$$

to calculate the stock price at different path.

Then calculate the expectation of the value Max(s-x,0) at the last paths to obtain the value of the option.

Do this process for 40 times and we can get 40 different option prices which I listed as follows:

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
7.982131	7.517244	8.053087	8.256022	8.22197	7.43725	7.320049	7.797912	7.883472	8.089847	7.706286	7.023483	8.264545	8.027424	7.274469	7.473033	7.722769	7.293058	7.303889	7.243711
21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40
7.585463	7.893393	7.864641	7.047832	8.152597	8.509181	7.745314	7.497974	8.326068	7.438205	7.22116	7.653616	7.571922	7.583336	7.377116	8.118611	7.499019	7.384693	7.910838	7.975223

The final result is 7.706196

Here S0=50, K=50, r=0.1,$\sigma = 0.3$,T=1, N=10000

b) Similar as in a), I just substituted the Box method by the Moro method. The results are as follows:

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
8.693043	8.380919	8.967208	8.910357	8.208119	8.906124	9.023796	8.649222	8.172312	8.860685	8.431099	8.566169	8.896196	8.278425	8.574155	8.636912	8.153253	8.729992	8.901107	8.32045
21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40
7.982647	8.226783	8.802185	8.539698	8.555638	8.013138	8.541352	8.729634	8.617182	9.158047	8.149045	8.328526	8.720994	8.317276	8.647453	8.746807	8.235726	8.619387	8.109553	8.400063

The final result is 8.542517

C) With the help of the program I wrote in the Computational finance course and searching in the website, I got the true option price is 8.367

d) The Anderson-Darling statistic is a statistical test of whether there is evidence that a given sample of data did not arise from a given probability distribution

The formula of $A^2 is$ A^2=−{ 2i−1 [$lnzini$=1+ln(1−Zn+1−i)]}n−n

The option price should be distributed to a normal distribution.

In this case we use the true option price which is the 8.367 as the mean and the unknown variance.
So it is the case two in the paper besides using the $\sum_i (x_i - u)^2/n$ as the s_i^2

The result is as follows:

For the Box-Muller method, the A^2 is 20.253643

For the Box-Muller method, the A^2 5.6090873.

My conclusion is that the result of the Moro method is better than the method of the Box-Muller method, since the value of AD test is smaller. However, they will both be rejected at the 1% level of the case 2 which is the 3.69.

e) From reading the paper, I agreed that the Box-Muller method is worse than the Moro method in the Quasi-Monte Carlo simulation method. The main reason is that the Box-Muller transformation does not preserve the matrix on $\tilde{I}^s$ and thus fails to preserve the low discrepancy of the original Faure sequence. That is the even spacing of the Faure sequence will be scrambled, resulting in the loss of low error bound.

Attachment:
Code listing

1) Sub Halton_sequence()

```
Dim n, s, j As Integer
Dim x(1000), y(1000, 1000) As Integer
Dim z(1000) As Double
Worksheets("sheet1").Range(Cells(1, "D"), Cells(2000, "d")).Clear

n = Worksheets("sheet1").Range("b3").Value
s = Worksheets("sheet1").Range("b4").Value
i = 0
j = 2
Do While i <> s
If prime(j) = "T" Then
i = i + 1
x(i) = j
End If
j = j + 1
Loop
For j = 1 To s
b = n
k = 1
z(j) = 0
Do
        y(k, j) = b Mod x(j)
        b = (b - y(k, j)) / x(j)
        z(j) = z(j) + y(k, j) / x(j) ^ (k)
        k = k + 1
    Loop While b <> 0
    Next j
For j = 1 To s
Worksheets("sheet1").Cells(j, "d").Value = z(j)
Next j
End Sub

Function prime(n As Integer) As String
Dim i As Integer
If n = 2 Or n = 3 Or n = 5 Or n = 7 Or n = 11 Or n = 13 Then
prime = "T"
Else

j = 0
For i = 2 To Sqr(n) + 1
```

```
        If n Mod i = 0 Then

        j = j + 1

        End If
Next i
        If j = 0 Then
        prime = "T"
                Else
                prime = "F"
                End If

End If
End Function

Sub generate_number()

Dim i As Integer

For i = 1 To Worksheets("sheet1").Range("b6").Value


Worksheets("sheet1").Range("b3").Value = i

Call halton_sequence

j = 0

Worksheets("sheet1").Range("d1").Select

Do While ActiveCell.Value <> ""

Cells(i, 7 + j).Value = ActiveCell.Value

j = j + 1

ActiveCell.Offset(1, 0).Select

Loop

Next i

End Sub
```

2) Faure.M

```
dim = 10; p = 11; max = 1000; m=1000;

uppersize = floor(log(max)/log(p))+1;

sfaure=[];
clear global sfaure;

  for n= 1: m

for h= 1 : dim

u=pascal(h,uppersize,p);
v=matrix(h,uppersize,p);

genmatrix=mod(v*u,p);

digits=fliplr(digit(n,p));

size= length(digits);

w=p.^(-range1(size));

sfaure(n,h)=mod((submatrix(genmatrix,size)*digits.').',p)*w.';

end

end

xlswrite('c:\2.xlsx',sfaure);
```

3) Matrix.m

```
function y=matrix(h,uppersize,p)

clear global v
for j = 1 : uppersize
    for i =1: uppersize
        if i<j
            v(i,j)= 0;
          elseif i==j
          v(i,j)=randint(1,1,[1,p-1]);
          elseif i>j
          v(i,j)=randint(1,1,[0,p-1]);
        end
```

```
        end
        end
        y=v;

    end
```

4) Pascal.M

```
        function y=pascal(h,uppersize,p)

        if h==1
        clear global u

        u=eye(uppersize);

        else
        for j = 1 : uppersize
            for i =1: uppersize
                if i>j
                    u(i,j)= 0;
                else
                   u(i,j)=mod(nchoosek(j-1,i-1)*(h-1)^(j-1),p);
                end
            end
        end
        end

        y=u;

    end
```