|  |
| --- |
| Lab 1 |
| Simulation And Modeling |
| Random number generation and the checking of the uniformity of generated sequence using Chi-Squared and Komolgorov Smirnov Tests. |

|  |
| --- |
| Manil Singha Malla  3/5/2018 |

* **Theory**
* **Random Numbers**

A sequence of numbers is said to be random if it fulfills the following properties

* The probability of each element of the population should be equal.
* The sequence should be uniformly distributed.
* The current output should be independent of any previous outputs.
* The sequence should have an infinite cycle period.
* **Pseudo Random Numbers**

The sequences of numbers which don’t possess an infinite cycle period are called Pseudo Random Numbers.

These numbers are generated through some algorithm and hence their randomness is questioned and they are dubbed Pseudo (or False) Random Numbers.

* **Kolmogorov Smirnov Test**

The Kolmogorov Smirnov Test is a test used to test the uniformity of a given sequence and is especially useful if the no of elements in the series is <20.

It is named after [Andrey Kolmogorov](https://en.wikipedia.org/wiki/Andrey_Kolmogorov) and [Nikolai Smirnov](https://en.wikipedia.org/wiki/Nikolai_Smirnov_(mathematician)).

* **Chi-Squared Test**

The Chi-Squared Test is one of the most popular Non-Parametric Tests for testing uniformity of a given sequence.

In this program we use it for sequences which have elements <20.

* **Code**

#include<stdio.h>

#include<conio.h>

void sort(float [], int);

int main()

{

FILE \*fp,\*fks,\*fcs;

float r[200],dp[200],dm[200],Xo,Do,Dt=5.25,Xt=16.9,sum=0.0,cl=0.0,cu=0.1;

int x[200],i,j,n,Oi[10]={0},Ei[10],m=101;

fp=fopen("rand.txt","w");

printf("Enter no. of random values:");

scanf("%d",&n);

printf("Enter seed values r[0] and r[1]");

scanf("%d%d",&x[0],&x[1]);

r[0]=(float) x[0]/100;

r[1]=(float) x[1]/100;

fprintf(fp,"%.2f %.2f ",r[0],r[1]);

for(i=2;i<n;i++)

{

x[i]=(x[i-1]+x[i-2])%m;

r[i]=(float) x[i]/100;

fprintf(fp,"%.2f ",r[i]);

}

fclose(fp);

if(n<20)

{

fks=fopen("ks.txt","w");

sort(r,n);

for(i=0;i<n;i++)

{

dp[i]= ((i+1)/n)-r[i];

if(dp[i]<0)

dp[i]=0;

dm[i]=r[i]-(i+1-1)/n;

if(dm[i]<0)

dm[i]=0;

sort(dp,n);

sort(dm,n);

if(dp[n-1]>dm[n-1])

Do=dp[n-1];

else

Do=dm[n-1];

}

printf("Kolmogorov-Smirnov Test\n");

fprintf(fks,"Kolmogorov-Smirnov Test\n");

printf("\ti\tr[i]\tdp[i]\tdm[i]\n");

fprintf(fks,"\ti\tr[i]\tdp[i]\tdm[i]\n");

for(i=0;i<n;i++)

{

printf("\t%d\t%.2f\t%.2f\t%.2f\n",i+1,r[i],dp[i],dm[i]);

fprintf(fks,"\t%d\t%.2f\t%.2f\t%.2f\n",i+1,r[i],dp[i],dm[i]);

}

printf("\n\nDo= %.2f",Do);

fprintf(fks,"\n\nDo= %.2f",Do);

printf("\nDt= %.2f",Dt);

fprintf(fks,"\nDt= %.2f",Dt);

if(Do<=Dt)

{

printf("\n\nSince %.2f <= %.2f,the generated random values are uniformly distributed.", Do, Dt);

fprintf(fks,"\n\nSince %.2f <= %.2f,the generated random values are uniformly distributed.", Do, Dt);

}

else

{

printf("\n\nSince %.2f > %.2f,the generated random values are not uniformly distributed.", Do, Dt);

fprintf(fks,"\n\nSince %.2f > %.2f,the generated random values are not uniformly distributed.", Do, Dt);

}

fclose(fks);

}

else

{

fcs=fopen("cs.txt","w");

for(i=0;i<10;i++)

Ei[i]=n/10;

for(i=0;i<n;i++)

{

if(r[i]>=0.0&&r[i]<=0.1)

Oi[0]++;

else if(r[i]>0.1&&r[i]<=0.2)

Oi[1]++;

else if(r[i]>0.2&&r[i]<=0.3)

Oi[2]++;

else if(r[i]>0.3&&r[i]<=0.4)

Oi[3]++;

else if(r[i]>0.4&&r[i]<=0.5)

Oi[4]++;

else if(r[i]>0.5&&r[i]<=0.6)

Oi[5]++;

else if(r[i]>0.6&&r[i]<=0.7)

Oi[6]++;

else if(r[i]>0.7&&r[i]<=0.8)

Oi[7]++;

else if(r[i]>0.8&&r[i]<=0.9)

Oi[8]++;

else

Oi[9]++;

}

for(i=0;i<10;i++)

{

sum=sum+(Oi[i]-Ei[i])\*(Oi[i]-Ei[i]);

}

Xo=(float) sum/Ei[0];

printf("Chi Square Test\n");

fprintf(fcs,"Chi Square Test\n");

printf("i class Oi[i] Ei[i] (Oi[i]-Ei[i])^2 (Oi[i]-Ei[i])^2/Ei\n");

fprintf(fcs,"i class Oi[i] Ei[i] (Oi[i]-Ei[i])^2 (Oi[i]-Ei[i])^2/Ei\n");

for(i=0;i<10;i++)

{

printf("%d %.1f-%.1f\t%d\t%d\t%d\t\t%.2f\n",i+1,cl,cu,Oi[i],Ei[i],(Oi[i]-Ei[i])\*(Oi[i]-Ei[i]),(float)((Oi[i]-Ei[i])\*(Oi[i]-Ei[i]))/Ei[0]);

fprintf(fcs,"%d %.1f-%.1f\t%d\t%d\t%d\t\t%.2f\n",i+1,cl,cu,Oi[i],Ei[i],(Oi[i]-Ei[i])\*(Oi[i]-Ei[i]),(float)((Oi[i]-Ei[i])\*(Oi[i]-Ei[i]))/Ei[0]);

cl=cl+0.1;

cu=cu+0.1;

}

printf("\n\nXo= %.2f",Xo);

fprintf(fcs,"\n\nXo= %.2f",Xo);

printf("\nXt= %.2f",Xt);

fprintf(fcs,"\nXt= %.2f",Xt);

if(Xo<=Xt)

{

printf("\n\nSince %.2f <= %.2f,the generated random values are uniformly distributed.", Xo, Xt);

fprintf(fcs,"\n\nSince %.2f <= %.2f,the generated random values are uniformly distributed.", Xo, Xt);

}

else

{

printf("\n\nSince %.2f > %.2f,the generated random values are not uniformly distributed.", Xo, Xt);

fprintf(fcs,"\n\nSince %.2f > %.2f,the generated random values are not uniformly distributed.", Xo, Xt);

}

fclose(fcs);

}

getch();

}

void sort(float a[], int n)

{

int i,j;

float t;

for(i=0;i<n;i++)

{

for(j=i+1;j<n;j++)

{

if(a[i]>a[j])

{

t=a[i];

a[i]=a[j];

a[j]=t;

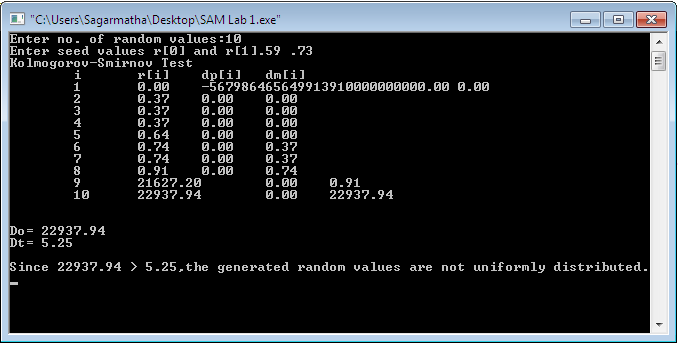
}

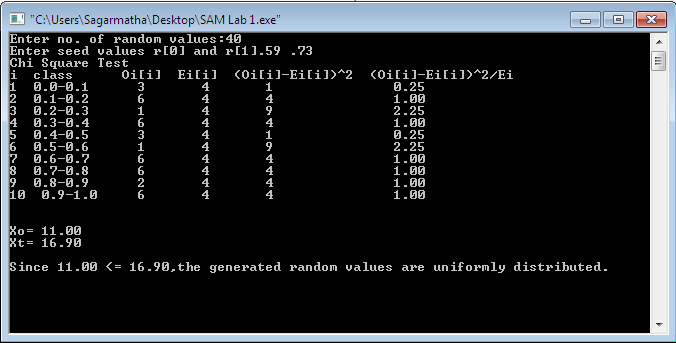
}

}

}

* **Output**

****

****

* **Conclusion**

Thus the program to create a sequence of random numbers and check its uniformity was generated.