Journal of Natural Sciences Research www.iiste.org
ISSN 2224-3186 (Paper) ISSN 2225-0921 (Online)
Vol.7, No.9, 2017

Monte Carlo Simulation for Evaluating Definite Intgrals

*Kassaye Bewketu Zellelew Ashenafi Abebe Asfaw Samuel Dea Sedisso
College of Natural and Computational Sciences, Wolaita Sodo University, Wolaita Sodo, Ethiopia

Abstract
Numerical integration techniques, such as Trapezoidal and Simpson's rules are commonly used for approximating definite integrals where the integrand has continuous derivatives to a certain order and when there is a certain amount of knowledge about the function. If a function fails to satisfy these requirements, then we must often apply a Monte Carlo method to obtain the approximation **(**Ján Ščigulinský**,** 2012**)**. Furthermore, Monte Carlo simulation offers an analyst a greater degree of flexibility to dictate his simulation conditions and sampling plans than does an experimenter in a real world environment (E. J. McGrath and D. C. Irving, 1975). The main objective of the study was to test the overall accuracy of Crude Monte Carlo method for evaluating the definite integrals of common functions having different dimensions. To realize this goal, softwares such as MATLAB version 2011, FORTRAN 90/95 and EXCEL 2010 were used. Functions that can be integrated both numerically and analytically were selected so as to compare the results and test the accuracy of the methods. For evaluating the integrals, random numbers were generated between 0 and 1 by typing the 'rand' function on the command window of MATLAB version 2011 and also by running the computer program written with FORTRAN 90/95. The integration of too complicated functions was performed numerically using the adaptive Simpson's Quadrature and results were considered as exact values. To measure the magnitude of the error in the method, the variance of the mesh points were calculated and then compared with the true absolute errors.The results revealed that the overall accuracy of the method is good for one dimensional integration and the accuracy declines as the dimension increases in contrast to some of the literature. Moreover, the accuracy did not necessarily decline as the number of mesh points increased. The variances of the mesh points were found to overestimate the true absolute errors. Theoretical errors were calculated to be greater than the true absolute errors and less than the variances. It can be concluded that one can use Crude Monte Carlo method to estimate the integral of one dimensional complicated functions with known confidence intervals for the true values.
Keywords: Accuracy, Crude Monte Carlo, Relative Error, Simulation, Definite Integrals, random number.

1. INTRODUCTION

Monte Carlo methods are widely used in Science, from the integration of multi-dimensional integrals to solving different problems in chemistry, physics, medicine, biology, or even Dow- Jones forecasting. They are any procedures which use pseudo random numbers to compute or estimate an integral (Badis Ydri, 2016). More specifically, Monte Carlo integrations are sampling methods, based on probability theory. They rely on trials to reveal information and are capable of handling quite complicated and large problems (Tudelft, 2009) and are of the most popular numerical methods for multidimensional integration. Another essential feature of Monte Carlo method is that its rate of convergence does not depend on integral dimension where the integration error is of the order O($n^{0.5}$) for 'n' defined to be the number of mesh points taken within the integration interval at which integrand is evaluated (VLADIMIR M. ET AL. , 2004). Moreover, accuracy attained depends on the number of trials and hence, the key point is to get random numbers (Tudelft, 2009).

This is very different from traditional cubature rules, whose rate of convergence exponentially decreases with the increase of dimension. As we increase the number of dimensions the error becomes worse. In other words, classical numerical integration methods become impractical at sufficiently higher dimensions. This is the fundamental appeal of Monte Carlo methods in quantum mechanics and statistical mechanics where we usually and so often encounter integrals of infinite dimensionality (Badis Ydri, 2016).

Monte Carlo method is distinguished from other techniques in numerical analysis by the use of random sampling to construct the solution of a physical or mathematical problem (L. L. Carter and E. D. Cash well, 1975) where integration accuracy can be substantially increased by use of various variance reduction methods such as importance sampling, correlated and control variate sampling, antithetic variates and stratified sampling (VLADIMIR M. et al., 2004).

In Monte Carlo Methods, we think of every sample as an experiment and make a loop over N sample points. These samples are often called Monte Carlo cycles or just samples (Morten Hjorth-Jensen , 2007).

Furthermore, Monte Carlo methods are a class of computational algorithms that rely on repeated random sampling to compute their results and are often used in simulating physical and mathematical systems. The methods are most suited to calculation by a computer and tend to be used when it is infeasible to compute an exact result with a deterministic algorithm (www.wikipedia.org).

Numerical integration techniques, such as the Trapezoidal rule and Simpson's rule, are commonly utilized for the purpose of approximating the value of a definite integral. However, these methods of approximation will guarantee a particular rate of convergence only when the integrand has continuous derivatives to a certain order. Before it can be applied, a numerical integration technique also requires a certain amount of knowledge about the function, such as the location of its domain, and where any discontinuities may exist. If a function fails to satisfy the requirements associated with a given numerical integration technique, or if it behaves in such a manner that one cannot in advance discern a sufficient amount of information about the function in order to circumvent any problems that may arise with such an approach, then we can often apply a Monte Carlo method to obtain an approximation of the integral (http://www.mathcs.emory.edu/ccs/ccs215/monte/node13.htm**)**. Monte Carlo methods can deal with integrands with no or little regularity, usual quadrature rules like Gauss product rules are built for functions having a given regularity or belonging to particular bases while adaptive integration methods rely on a quadrature rule designed for the non-adaptive case (Christophe D.L et al., 2016).

A useful feature of Monte Carlo simulation is that the analyst has the flexibility to dictate his simulation conditions and sampling plans to a much greater extent than does an experimenter in a real world environment. This extra scope for freedom of action provides an excellent opportunity for optimal design of simulations to obtain estimates with minimal sampling size. This will effectively reduce the time and effort involved in computation as the number of trials necessary to achieve a given accuracy is thereby reduced (E. J. McGrath and D. C. Irving, 1975)

2. MATERIAALS AND METHODS

2.1 MATERIALS

For the sake of convenience and effectiveness of the study, softwares such as MATLAB version 2011, FORTRAN 90/95 and EXCEL 2010 were used to generate and manipulate input random numbers, evaluate the integrals at these random numbers, plot the graphs corresponding to the resulting values and analyze of the results.

2.2 METHODS

Most of the time, definite integrals $I = \int_a^b f(x)dx$ of practical importance in many fields of science and engineering are too difficult or impossible to evaluate analytically. Monte-Carlo strategy for evaluating such an integral of a function f(x) is depicted by:

$$I = \int_a^b f(x)dx \approx h\sum_{i=1}^{N} wif(xi)$$

where w_i are the weights determined by the specific integration method (like Simpson's or Taylor's methods) $h = \frac{b-a}{N}$ is the step size, N is the number of mesh points used and xi represent mesh points taken between a and b. If we employ the crudest possible approach, then $w_i = 1$ for all integration methods. This crude approach corresponds to the rectangle method of approximating definite integrals. This method is very easy for application even though the error corresponding to it may be large as compared to other variance reduction techniques. Up on replacing $w_i = 1$ in the formula given above, we will get the Crude Monte Carlo method given by (A. Kong et al., 2003; Tudelft, 2009)

$$I = \int_a^b f(x)dx \approx \frac{b-a}{N}\sum_{i=1}^{N} f(xi)$$

The researchers selected functions that can be integrated numerically and analytically so as to compare the results obtained by analytic and numerical means and test the accuracy of the methods. After the selection of the functions, they were integrated both analytically and by Crude Monte Carlo Method using F90/95 codes and MATLAB version 2011. To calculate the error and examine the accuracy, the variance of the mesh points were calculated and then compared with the true absolute errors. This was done to check if the variance of the mesh points could over-estimate the true absolute error. If the variances over-estimate the true errors, then one can evaluate difficult integrals by using Crude Monte Carlo Method and estimate the error in the method by calculating the variance of the randomly generated numbers.

For evaluating the integrals, random numbers were generated between 0 and 1 by typing the 'rand' function on the command window of MATLAB version 2011 and also by running the computer program written with FORTRAN 90/95. The integration of too difficult integrals was performed numerically using the adaptive Simpson's Quadrature. EXCEL 2010 was used to plot the graph of function values at mesh points against the mesh points to see the nature of output function or result of integrations within the range of the independent variable where it was allowed to vary between 0 and 1.

For complicated functions, the values of the integrals obtained by Adaptive Simpson's Quadrature method were considered as exact values. These errors were compared and check if the variance over estimates the absolute errors.

In theory, integration error is in the order of $n^{-0.5}$ where n is the number of mesh points used to evaluate the integral. Comparison of such errors with true absolute errors and variance of the mesh points was also presented for further analysis.

3. RESULTS AND DISCUSSIONS

3.1 Graphical analysis of the results obtained by Crude Monte Carlo and analytical approaches

It is clear from figure 1, 2, 3, 4, 5 and 6 that Monte Carlo method offers a good approximation to one dimensional definite integrals. As can be observed from the graphs, values obtained by analytic and Crude Monte Carlo methods are nearly equal to each other. The values indicated in tables 1 and 4 also verify this situation.

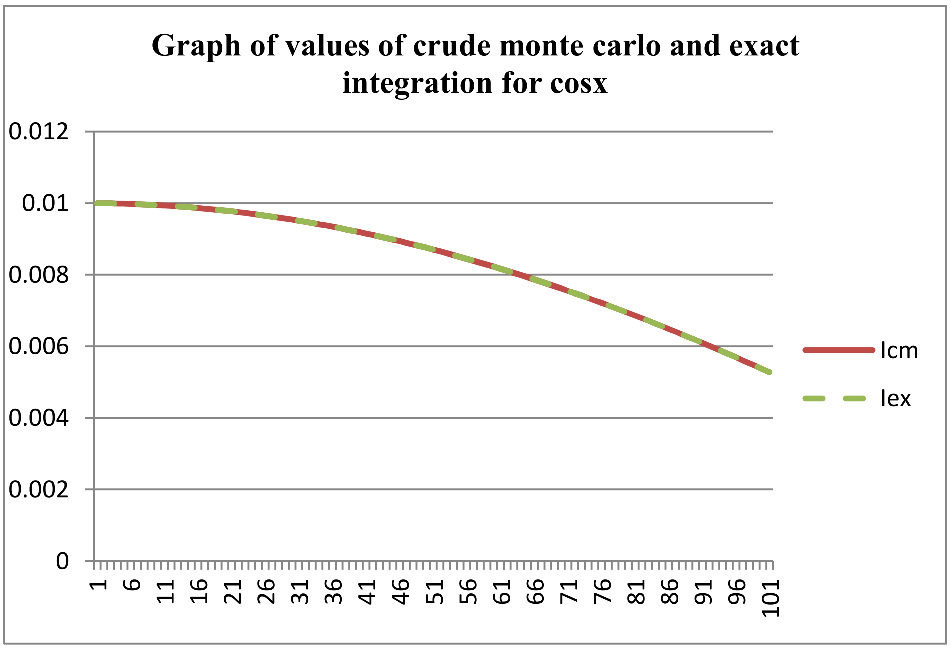

Figure 1

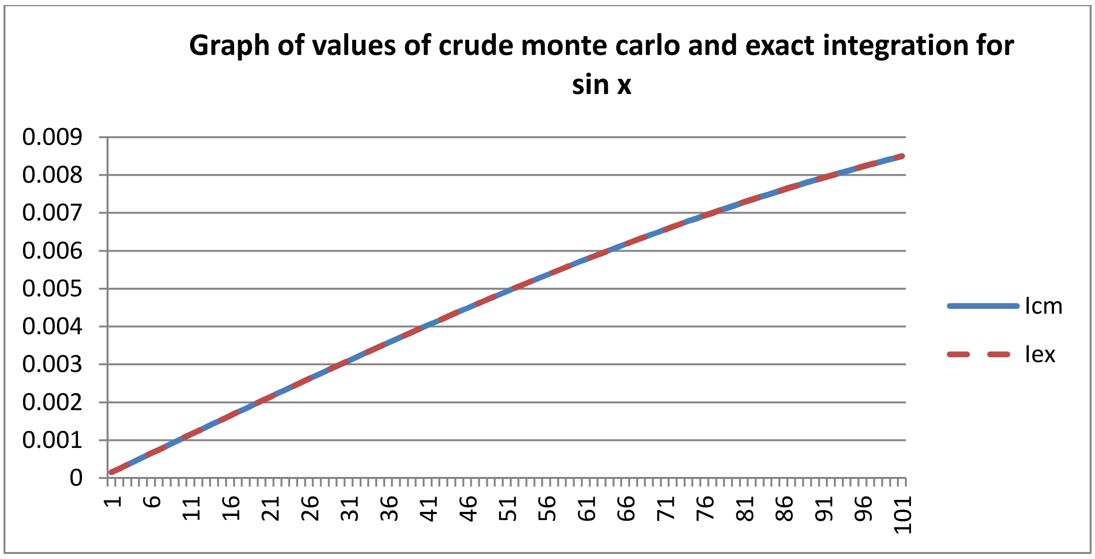

Figure 2

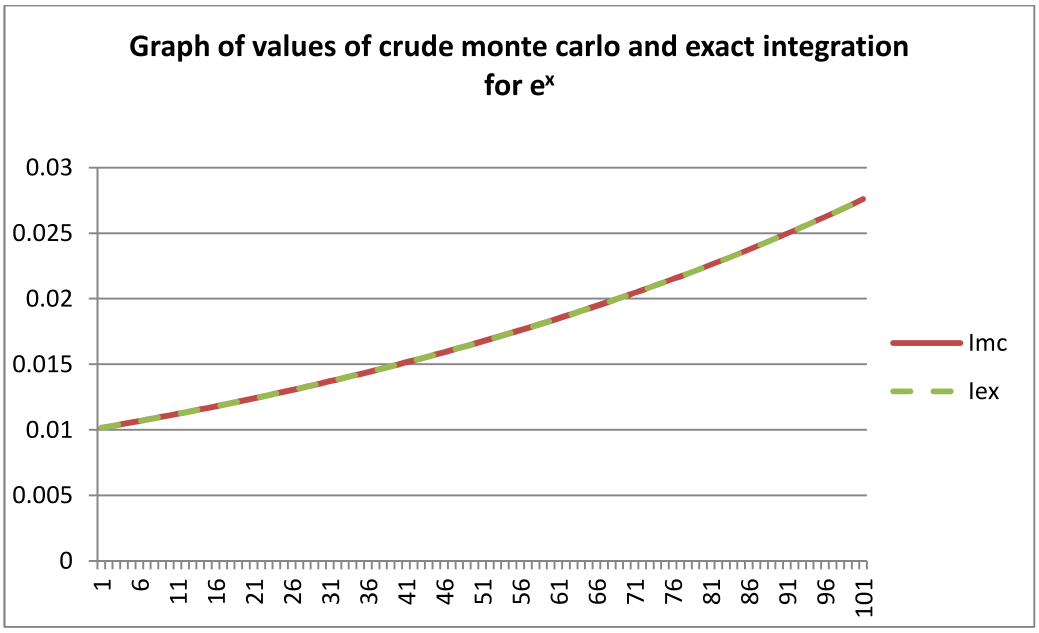

Figure 3

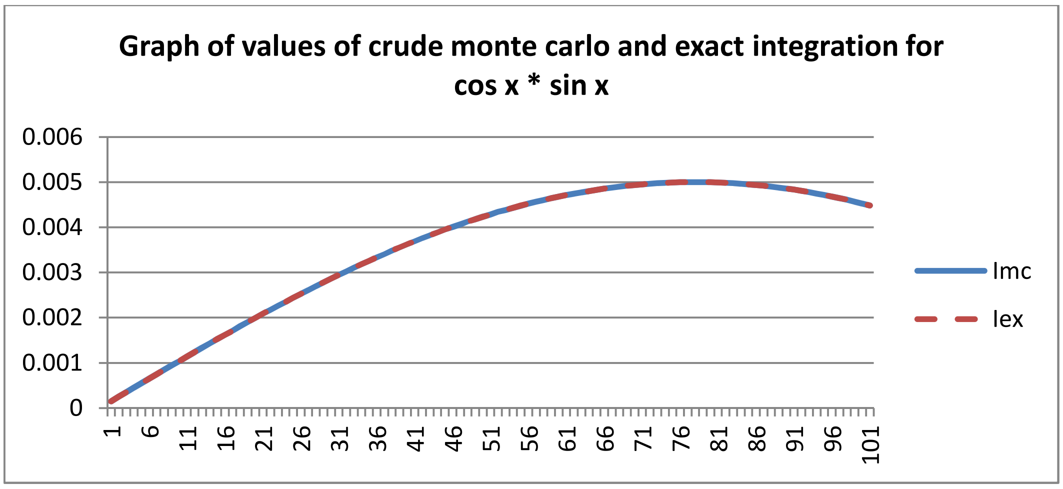

Figure 4

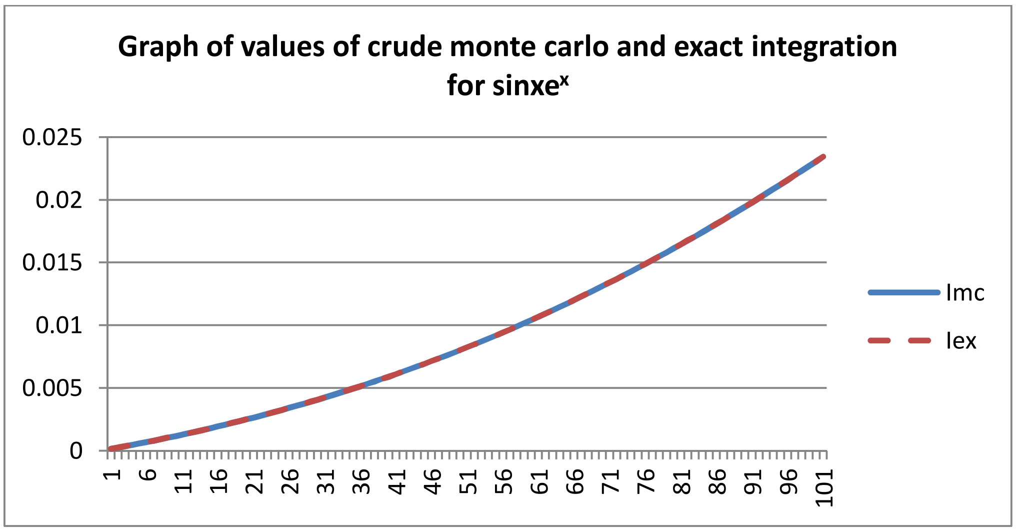

Figure 5

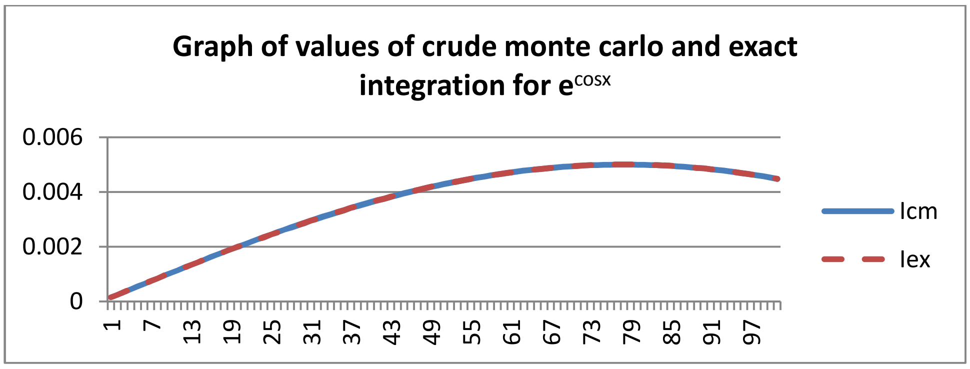

Figure 6

3.2 Representation of convergence of the Crude Monte Carlo Method with dimension of the integrand.
Statistical errors are inherent to the Monte Carlo method due to the stochastic nature which utilizes random numbers. In order to say something about the quality of an obtained, it is crucial to have the proper tools for analyzing the errors.

Tables 1, 2 and 3 clearly show that the accuracy of crude Monte Carlo method decreases with dimension of the integrand. This is in contrast with Vladimir M. et al. (2004) and in agreement with Badis Ydri (2016). As the number of independent variables increase the error in the method also increases because of the cumulative effect resulting from the error corresponding to each variable. Moreover, as the number of mesh points increase, the error in the method does not necessarily decrease. The reason for this is due to over accumulation of round off errors as the mesh points get finer and finer.

It is evident from table 2 and 3 that for 100 mesh points; the absolute errors are greater than the theoretical errors for both double and triple integrals. This is another verification representing poor accuracy of Crude Monte Carlo method for approximating double and triple integrals.

Table 1:- *Convergence of Crude Monte Carlo Method for evaluating $f(x) = e^x$ between a=0 and b=1 and also with exact value = 1.718281828459046, MC = Monte Carlo Approximation*

n	MC	Absolute Error	Relative error(%)
10	1.900577	0.182295	10.60915
100	1.763159	0.044877	2.611747
1000	1.719994	0.001712	0.099644
10000	1.718738	0.000456	0.026548
100000	1.719103	0.000821	0.04779
1000000	1.725776	0.007494	0.436143

Table 2:- *Convergence of Crude Monte Carlo Method for evaluating $f(x,y) = e^{xy}$ between a=0 and b=1 and also with exact value = 1.317902151454477 (Obtained using Adaptive Simpson quadrature), MC = Monte Carlo Approximation*

n	MC	Absolute Error	Relative error (%)
10	1.489994	0.172092	13.05801
100	1.452598	0.134696	10.22047
1000	1.461640403	0.143738	10.9066
10000	1.463542	0.14564	11.05088
100000	1.462748	0.144846	10.99064
1000000	1.462662	0.14476	10.98411

Table 3:- *Convergence of Crude Monte Carlo Method for evaluating $f(x,y,z) = e^{xyz}$ between a=0 and b=1 and also with exact value = 1.146499072732518 (Obtained using Adaptive Simpson quadrature) MC = Monte Carlo Approximation*

n	MC	Absolute Error	Relative error (%)
10	1.320378	0.173879	15.16608
100	1.34982	0.203321	17.73407
1000	1.340465176	0.193966	16.91812
10000	1.341994	0.195495	17.05147
100000	1.342218	0.195719	17.07101
1000000	1.343413	0.196914	17.17524

3.3 Representation of the results obtained by Crude Monte Carlo method and analytical approaches

The results in table 5 reveal that true absolute errors are much far less than the variances of the mesh points. This clearly shows that the function values converge faster than the mesh points used for the function evaluations indicating good accuracy of the method used. The theoretical errors are by far greater than the true errors particularly, about 9 to 161 times greater. As we can observe from table 4 all the relative errors are less than 1% except for the natural logarithmic function which has small fractional values between 0 and 1 and hence subject to more round of errors than the other functions.

Table 4:- *Comparison of the results obtained by Crude Monte Carlo and analytical approaches. Imc = Integral Approximation by Monte Carlo Method, Iex = Values from Exact Integration, E_a = absolute error, E_r = relative error*

Function	Imc	Iex	Ea =|Imc-Iex|	Er(%)
cos(x)	0.83784	0.841471	0.003631	0.432
sin(x)	0.46284	0.459698	0.003142	0.683
e^x	1.729553	1.718282	0.011271	0.656
ln x	-0.9775	-1	0.0225	2.25
e^{cosx}	2.334556	2.3416	0.007044	0.301
e^{sinx}	1.620228	1.6319	0.011672	0.715
cosx*sinx	0.35079	0.3540	0.00321	0.907
x^3+x^2-x+2	2.073168965	2.0833	0.01013104	0.486
$\frac{\mathbf{x3+x2-x+2}}{x+2}$	0.832680249	0.8333	0.000620	0.0620

Table 5:- *Comparison of true absolute error, theoretical error and variances of the mesh points.*

Function	Ea =|Imc-Iex|	var	Theoretical error= $n^{-0.5}$	var/Ea
cos x	0.003631	0.087617	0.1000	24.130
sin x	0.003142	0.087617	0.1000	27.886
e^x	0.011271	0.087617	0.1000	7.774
ln x	0.0225	0.083441	0.1000	3.708
e^{cosx}	0.007044	0.087617	0.1000	12.439
e^{sinx}	0.011672	0.083441	0.1000	7.149
cosx*sinx	0.00321	0.083441	0.1000	25.994
x^3+x^2-x+2	0.01013104	0.083441	0.1000	8.236
$\frac{\mathbf{x3+x2-x+2}}{x+2}$	0.000620	0.083441	0.1000	134.582

3.4 Representing the accuracy of Crude Monte Carlo method for multidimensional integrals

Table 6: *Efficiency Crude Monte Carlo Simulation for double integrals for a=0, b=1*

Function	I_{mc} (using Crude Monte Carlo) x in (0,1) with	I_{ex} (exact integral)	Ea = $\|I_{mc}$-$I_{ex}\|$	E_r (relative error	Var	var/Ea
cosxy	0.9450	0.9461	0.0011	0.116267	0.0846	76.90909
sinxy	0.2566	0.2398	0.0168	7.005838	0.0846	5.035714
cosxy*sinxy	0.2194	0.2118	0.0076	3.588291	0.0846	11.13158
e^{xy}	1.2864	1.3179	0.0315	2.390166	0.0846	2.685714
lnxy	-1.9691	-2.0000	0.0309	1.545	0.0846	2.737864
e^{cosxy}	2.5846	2.5833	0.0013	0.050323	0.0846	65.07692
e^{sinxy}	1.2718	1.2989	0.0271	2.086381	0.0846	3.121771
x^3y+xy^2-y+2	1.7681	1.7917	0.0236	1.317185	0.0846	3.584746
$\frac{x^3+y^2-y+2}{y+2}$	0.8428	0.8451	0.0023	0.272157	0.0846	36.78261

Table 7: *Efficiency Crude Monte Carlo Simulation for triple integrals for a=0, b=1*

Function	I_{mc} (using Crude Monte Carlo) x in (0,1) with 100 points	I_{ex} (exact integral) Quad)	Ea = $\lvert I_{mc}-I_{ex}\rvert$	E_r (%)	Var	var/Ea
cosxyz	0.98204	0.9818	0.00024	0.024445	0.0846	352.5
sinxyz	0.11411	0.1224	0.00829	6.772876	0.0846	10.20507
cosxyz*sinxyz	0.11392	0.1152	0.00128	1.111111	0.0846	66.09375
e^{xyz}	1.3352	1.1465	0.2037	17.76712	0.0846	0.415317
lnxyz	-2.98445	-3.0000	0.01555	0.51833	0.0846	5.440514
e^{cosxyz}	2.56824	2.6713	0.10306	3.858047	0.0846	0.820881
e^{sinxyz}	1.32819	1.1422	0.18599	16.28349	0.0846	0.454863
$\boldsymbol{x^3yz + xzy^2 - z + 2}$	1.66204	1.6458	0.01624	0.986754	0.0846	5.20936
$\boldsymbol{\frac{x^3 + y^2 - xz + 2}{y + 2}}$	0.94127	0.9328	0.00847	0.908019	0.0846	9.988194

Table 8: *Efficiency Crude Monte Carlo Simulation for triple integrals for a=0, b=2*

Function	I_{mc} (using Crude Monte Carlo)	I_{ex} (exact integral)	Ea = $\lvert I_{mc}-I_{ex}\rvert$	E_r (relative error in %)	Var.	var/Ea
cosxyz	3.57111	4.1589	0.58779	14.1333	0.3446	0.586264
sinxyz	2.14499	3.1205	0.97551	31.26134	0.3446	0.353251
cosxyz*sinxyz	0.92319	1.2989	0.37571	28.92524	0.3446	0.917197
e^{xyz}	653.9231	83.4654	570.4577	683.4661	0.3446	0.000604
lnxyz	-6.26056	-7.3646	1.10404	14.9912	0.3446	0.312126
e^{cosxyz}	14.4662	15.6404	1.1742	7.507481	0.3446	0.293476
e^{sinxyz}	11.60159	12.8527	1.25111	9.734219	0.3446	0.275435
$\boldsymbol{x^3yz + xzy^2 - z + 2}$	79.42224	34.6667	44.75554	129.1024	0.3446	0.0077
$\boldsymbol{\frac{x^3 + y^2 - xz + 2}{y + 2}}$	11.51362	11.4081	0.10552	0.924957	0.3446	3.265732

Table 8 clearly shows that Crude Monte Carlo method seems to be useless for approximating triple integrals particularly in case of large integration width. This is really worse especially for triple integral of exponential functions. The reason for this is that the function increases or decreases rapidly and thus, a small change in the variables give rise to large change in the function.

As we can be seen from table 7 above, Crude Monte Carlo method seems to be relatively good for approximating triple integrals where integration width is 1. Furthermore, the variances of the mesh points underestimate the true absolute errors and hence one cannot use the method for approximating triple integrals and cannot indicate a confidence interval for the exact value.

To reduce the absolute error in Crude Monte Carlo method of evaluating an integral, the most important step is to use the MATLAB function '*rng shuffle*' every time random numbers are generated, take as many trials as possible and then take averages. The other thing of minor importance is to use as many random numbers as possible depending on the integration width. The variance of the mesh points is mainly affected by the integration width and not so much affected by the number of randomly taken mesh points used for fixed integration width. For example, the variance of 100,1000,10000 randomly selected numbers b/n 0 and 1 is respectively, 0.0846, 0.0836 and 0.0832 whereas the variance of 100 points taken between 0 and 2 is 0.3446 which is about 4.1 times greater than the variance of points taken between 0 and 1(as depicted in table 7 and 8) .

4. CONCLUSIONS

The results of the study (as depicted from figures 1, 2, 3, 4, 5, 6 and particularly table 1 and 4) revealed that Crude Monte Carlo method is good for approximating one dimensional integrals having smaller width except for those outlined with red colors. For large integration width, it is advisable to sub-divide the interval into unit-width intervals and apply the method for each sub interval. Moreover, it is realized that convergence does not necessarily increase with the number of mesh points. This is largely due to over accumulation of round off errors as mesh points get finer and finer.

It is also revealed from the results of the study that for one dimensional function, the true absolute errors are far less than the variance of the mesh points used to evaluate the integral indicating good accuracy of the method.

To use the Crude Monte Carlo method for approximating multidimensional integrals, the integration width needs to be relatively smaller than 1 or large number of mesh points must be used. The poor accuracy of the method is also indicated by the large values of true absolute errors as compared to the theoretical errors.

AKNOWLEDGEMENTS

We are grateful to the almighty God, the merciful and compassionate that helped us in all endeavors of our lives. We would like to forward the most heartfelt thanks and appreciations to our family members for their motivations and love.

REFERENCES

A. Kong, P. McCullagh, X. -L. Meng, D. Nicolae, Z. Tan (2003). A Theory of Statistical Models for Monte Carlo Integration. Journal of the Royal Statistical Society, Blackwell Publishing for the Royal Statistical Society.

Badis Ydri(2016). Computational Physics: An Introduction to Monte Carlo Simulations of Matrix Field Theory. Department of Physics, Faculty of Sciences, BM Annaba University, Annaba, Algeria.

Christophe De Luigi, Jerome Lelong, Sylvain Maire(2016). Adaptive numerical integration and control variates for pricing Basket Options. Applied Numerical Mathematics, Elsevier.

E. J. McGrath and D. C. Irving (1975). TECHNIQUES for EFFICIENT MONTE CARLO SIMULATION Vol. III, VARIANCE REDUCTION. Office of Naval Research Department of the Navy Arlington, Virginia 22217 by Science Applications, Incorporated.

http://www.mathcs.emory.edu/ccs/ccs215/monte/node13.htm. Monte Carlo Method for Definite Integration. Accessed on 10 May, 2017 17:05 http://www.wikipedia.org. Monte Carlo Simulation. Accessed on 10 May, 2017 9:30

Ján Ščigulinský (2012). Educational materials on Monte Carlo Methods. Masaryk University Faculty of informatics, Bachelor Thesis (published)

L. L. Carter and E. D. Cash well (1975). Particle Transport Simulation with the Monte Carlo Method. Technical Information Center, Office of Public Affairs, U. S. Energy Research and Development Administration, Los Alamos Scientific Laboratory, U.S.A

Morten Hjorth-Jensen (2007). Introduction to Monte Carlo Methods, Integration and Probability Distribution. Department of Physics and Center of Mathematics for Applications University of Oslo, N-0316 Oslo, Norway. C

Tudelft (2009). Summer school Computational Finance, Hitotsubashi University**.**

VLADIMIR M. IVANOV, MAXIM L. KORENEVSKY (2004). Sequential Monte Carlo and adaptive numerical integration. Department of Computer Science Saint-Petersburg State Poly technical University, Politehnicheskaya, Saint-Petersburg, Russia.

APPENDICES

Appendix A:- The computer algorithm that calculates the exact and Crude Monte Carlo values for the integral of cos(x) between 0 and 1 using 100 mesh points.

```
Program mont_carlo
Implicit none
Integer :: n, j
Real:: a,b,xj,delx,s,xav,var,s1,s2,Imc,Imc2,Iex,Iex2,Ea,Eap,Ef
open(20, file='cosine.txt')
Print*, 'enter value of a,b, delx'
Read*,a,b, delx
write(20,13) 'xj','Imc2', 'Iex2','Ea','Eap'
13 format (a10, a10, a10, a10, a10)
n=(b-a)/ delx
Xj=a, S=0.0, s1=0.0, s2=0.0
Do j=0,n
S1=s1+xj
End do
Xav=s1/n
Do j=0,n
S2=s2+(xj-xav)*(xj-xav)
end do ! end of looping
Var= sqrt(s2/(n-1))  !unbiased estimator of the variance
```

```
Eap=sqrt(var)
print*, 'iterative values using crude monte carlo and analytic means'
 Do j=0,n
Xj=xj+ delx
Imc2=(xj+delx-xj)/(2.0)*(cos(xj)+cos(xj+delx))     !iterative values using crude monte carlo method
Iex2=sin(xj+delx)-sin(xj)                          !iterative values using analytic  method
Ea=abs(Iex2-Imc2)
s=s+cos(xj)
  var=sqrt(s2/(n-1))
  Eap=sqrt(var)
write(20,12) Iex2
!write(20,12) xj,Imc2,Iex2,Ea,Eap
12 format (F13.10,F13.10,F13.10,F13.10,F13.10)
End do
print*,    'final values using crude monte carlo and analytic means'
Imc= (b-a)* (s/n )  ! final value of the integral using crude Monte Carlo Method
Iex=sin(b)-sin(a)   ! final value of the integral using analytical means.
Ef=abs(Iex-Imc)
print*, Imc,Iex,Ef
! calculate the average value of the mesh points, xav
!calculate the variance of the mesh points
End program mont_carlo
```

Appendix B:- The computer algorithm that calculates the exact and Crude Monte Carlo values for the integral of ln(x) between 0 and 1 using 100 mesh points.

```
Program mont_carlo
Implicit none
Integer :: n, j
Real:: a,b,xj,dx,s,xav,var,s1,s2,Imc,Imc2,Iex,Iex2,Ea,Eap,Ef
open(20, file='lnx.txt')
!Setting  values of the lower and upper limits of integration and interval or change in x
Print*, 'enter value of a,b, dx'
Read*,a,b, dx
!Determine the no of mesh points!
write(20,13) 'xj','Imc2', 'Iex2','Ea','Eap'
13 format (a10,a10,a10,a10,a10)
n=(b-a)/ dx
!determine the actual mesh points and sum of the function values at the mesh points
Xj=a, S=0.0, s1=0.0, s2=0.0
Do j=0,n
S1=s1+xj
End do
Xav=s1/n
Do j=0,n
S2=s2+(xj-xav)*(xj-xav)
end do ! end of looping
Var= sqrt(s2/(n-1))  !unbiased estimator of the variance
Eap=sqrt(var)  !the square root of the variance equals the standard deviation which approximates the absolute
               error
print*, 'iterative values using crude monte carlo and analytic means'
 Do j=0,n
Xj=xj+ dx
Imc2=(xj+delx-xj)/(2.0)*(log(xj)+log(xj+delx))     !iterative values using crude monte carlo method
Iex2=(xj+delx)*log(xj+delx)-(xj+delx)-xj*log(xj)+xj          !iterative values using analytic  method
Ea=abs(Iex2-Imc2)
s=s+log(xj)
  var=sqrt(s2/(n-1))
  Eap=sqrt(var)
write(20,12) xj,Imc2,Iex2,Ea,Eap
```

```
12 format (F13.10,F13.10,F13.10,F13.10,F13.10)
End do
print*,    'final values using crude monte carlo and analytic means'
Imc= (b-a)* (s/n )  ! final value of the integral using crude Monte Carlo Method
Iex=b*log(b)-a*log(a)+a-b   ! final value of the integral using analytical means.
Ef=abs(Iex-Imc)
print*, Imc,Iex,Ef
End program mont_carlo
```

Appendix C:- The computer algorithm that calculates the exact and Crude Monte Carlo values for the integral of e^x between 0 and 1 using 100 mesh points.

```
Program mont_carlo
Implicit none
Integer :: n, j
Real:: a,b,xj,delx,s,xav,var,s1,s2,Imc,Imc2,Iex,Iex2,Ea,Eap,Ef
open(20, file='exp2.xls')
!Setting  values of the lower and upper limits of integration and interval or change in x
Print*, 'enter value of a,b, delx'
Read*,a,b, delx
!Determine the no of mesh points!
write(20,13) 'xj','Imc2', 'Iex2','Ea','Eap'
13 format (a10,a10,a10,a10,a10)
n=(b-a)/ delx
!determine the actual mesh points and sum of the function values at the mesh points
Xj=a, S=0.0, s1=0.0, s2=0.0
Do j=0,n
S1=s1+xj
End do
Xav=s1/n
Do j=0,n
S2=s2+(xj-xav)*(xj-xav)
end do ! end of looping
Var= sqrt(s2/(n-1))  !unbiased estimator of the variance
Eap=sqrt(var)  !the square root of the variance equals the standard deviation which approximates the absolute
               error
print*, 'iterative values using crude monte carlo and analytic means'
 Do j=0,n
Xj=xj+ delx
Imc2=(0.5)*(xj+delx-xj)*(exp(xj)+exp(xj+delx))     !iterative values using crude monte carlo method
Iex2=exp(xj+delx)-exp(xj)                          !iterative values using analytic  method
Ea=abs(Iex2-Imc2)
s=s+exp(xj)
  var=sqrt(s2/(n-1))
  Eap=sqrt(var)
write(20,12) xj,Imc2,Iex2,Ea,Eap
12 format (F20.10,F20.10,F20.10,F20.10,F20.10)
End do
print*,    'final values using crude monte carlo and analytic means'
Imc= (b-a)* (s/n )  ! final value of the integral using crude Monte Carlo Method
Iex=exp(b)-exp(a)   ! final value of the integral using analytical means.
Ef=abs(Iex-Imc)
15 format (F20.15,F20.15,F20.15,F20.15,F20.15)
print*, Imc, Iex, Ef
End program mont_carlo
```