Student ID : 58328870
Student Name : Smitha Gurunathakrishnan Balagurunatan

CS 206P Final Project Report

In this project I have compared **Monte Carlo vs. Deterministic Volume Integration** methods for computing the volume of the hypersphere. I will be estimating the volume of a d-dimensional (hyper-)sphere of radius r=1 centered on the origin.

Part-1
Monte Carlo Integration:

In this method, I have surrounded the hypersphere with the smallest hypercube that can enclose it. This hypercube is centered at the origin with sides of length 2 and has a volume of 2^d. I have then selected N points at random inside the hypercube. I used Rejection algorithm to determine the count of points that are inside the hypersphere. I do this by generating 2D matrix N*d size with random numbers in range [-1,1] and checking if sum of the squares row wise is <=1.

The volume of the hypersphere is then

$$(\text{N-points inside} / \text{N-points in total}) * 2^d$$

I have assured 4 digits of accuracy value by taking difference between analytical model formula and determined volume (mean) and making sure it's equal to zero. I have taken different values of points N and runs n. For each dimension d, I have used a fixed number of N and n to calculate mean and standard deviation. I computed error bar or standard deviation as $\sigma/\sqrt{n}$.

I have used 95% confidence interval. For precision, I have calculated it as follows –

- Standard Deviation (std) = $\sigma / \sqrt{n}$, where σ is the standard deviation, n is the number of runs.
- Low = mean – 2(std)
- High = mean + 2(std)
- **Precision = (High – Low)/mean < = 10^-4**

Precision calculated will be used as the stopping criteria for 'd' value in this method. Mean is the determined volume from Monto Carlo Integration. In my results for d, n is 100 and I have chosen N so that the estimated value lies between ±0.0001.

The 'd' value has been pushed by –

- Ensuring 95% confidence interval
    - (High – Low)/mean < = 10^-4

The following are my results I have my **'n' is 100 fixed**, I have **varied N** as follows -

Dimension	Mean	No of Points - N	Precision	Standard Deviation	Estimated volume at 95% confidence level
1	2.0	100000	0	0	2±0
2	3.1416	4000000	0.0001	0.0001	3.1416±0.0001
3	4.1888	8000000	0.0001	0.0001	4.1888±0.0001
4	4.9347	12100000	0.0001	0.0001	4.9347±0.0001
5	5.2635	16300000	0.0001	0.0001	5.2635±0.0001
6	5.1673	20600000	0.0001	0.0001	5.1673±0.0001
7	4.7246	25000000	0.0001	0.0001	4.7246±0.0001
8	4.0587	29500000	0.0001	0.0001	4.0587±0.0001
9	3.2975	34100000	0.0001	0.0001	3.2975±0.0001
10	2.5510	38800000	0.0001	0.0001	2.5510±0.0001

Student ID : 58328870
Student Name : Smitha Gurunathakrishnan Balagurunatan

I was able to push till d=10, for dimensions higher than d=10, probability that a point in a cube falls inside a hypersphere is very small and I observed very large deviations in precision upto 0.002 instead of 0.0001 and also it lacks accuracy and efficiently. N random numbers were selected from a uniform distribution and the volume was estimated to 4-digit precision.

Cube Based Integration:

In cube based method, I have divided each of the d dimensions into K segments, so that the side of the smaller is of length 2/k. The volume of smaller hypercube is $(2/k)^d$.

Then I find the distance from small hypercube's center to the hypersphere's center to determine whether it is completely

- inside the hypersphere if distance <= (radius-half_diagonal_distance) (Val 1 for box type)
- outside the hypersphere if distance >= (radius+half_diagonal_distance) (Val 2 for box type)
- intersecting the hypersphere otherwise (Val 3 for box type)

The lower and upper bound are found as –

- lower_bound = inside_count* $(2/k)^d$
- upper_bound = inside_count+(intersect_count/2) *$(2/k)^d$

upper_bound is considered as **volume** here in part 1=

- **inside_count+(intersect_count/2) *$(2/k)^d$**

Precision is calculated and will be used as the stopping criteria for 'd' value in this method

- **(upper_bound - lower_bound)/average (upper_bound, lower_bound) <= 0.0001**

I have determined how large K should be by using an iterative method to increase the valueof k and calculating the volume, till 4-digit precision is reached which was used as a stopping criteria. I have also ensured that K-1 does not satisfy this precision.

I have been able to push 'd' till 3. The results are as follows –

Dimension	Volume (Upper Bound)	Actual Volume	Precision	K value
1	1.9998	2.0	0.0001	8889
2	3.1413	3.1416	0.0001	15502
3	4.1889	4.1888	0.0001	35607

As you can see, I'm able to find the volumes of the hypersphere in Monte Carlo till d=10 and Cube based integration till d=3. Given a same amount of time I can find volumes more accurately in Monte Carlo even for high dimensions. Cube based method takes a lot of time to run as dimensions increases and accuracy is less. In Cube based integration for d=4, the run time is longer than Monto Carlo's d=15, hence I had to stop with D=3 for cube based. Therefore, **Monte Carlo** method **is more efficient** since it allows **dimension to become larger** given the **4 digits of precision** as insisted in the answer.

If **8 digits of precision** is demanded, the observations will be similar. **Monte Carlo** method **will be more efficient** since it will allow dimension to be pushed to a larger value with lesser run time compared to cube based integration. Monto Carlo gives more accurate precisions for lesser runtime and larger 'd' value with an accuracy maintained.

Student ID : 58328870
Student Name : Smitha Gurunathakrishnan Balagurunatan

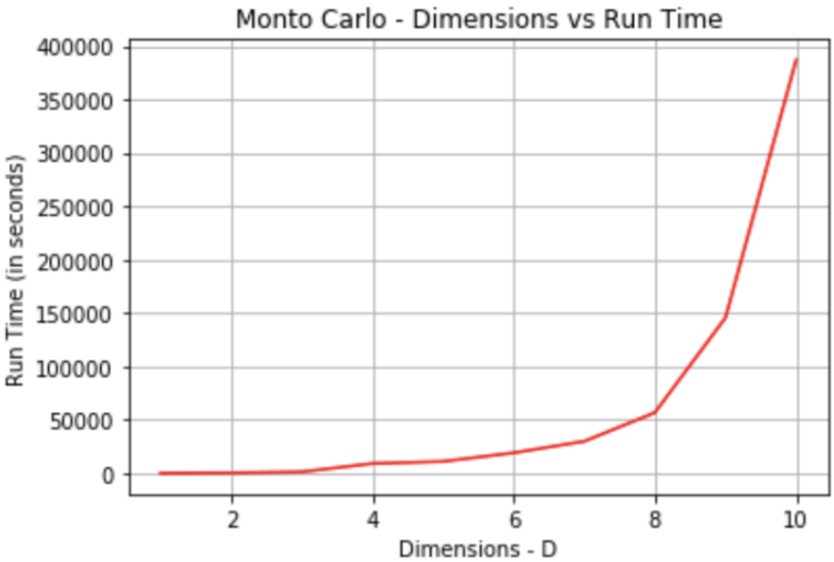
In Cube based, the value needed for **K is high** to get that precision and the **run time of Cube based** is **$O(K^d)$.** Hence as dimension increases, the run time increases exponentially for it. Hence Monte Carlo method is more efficient.

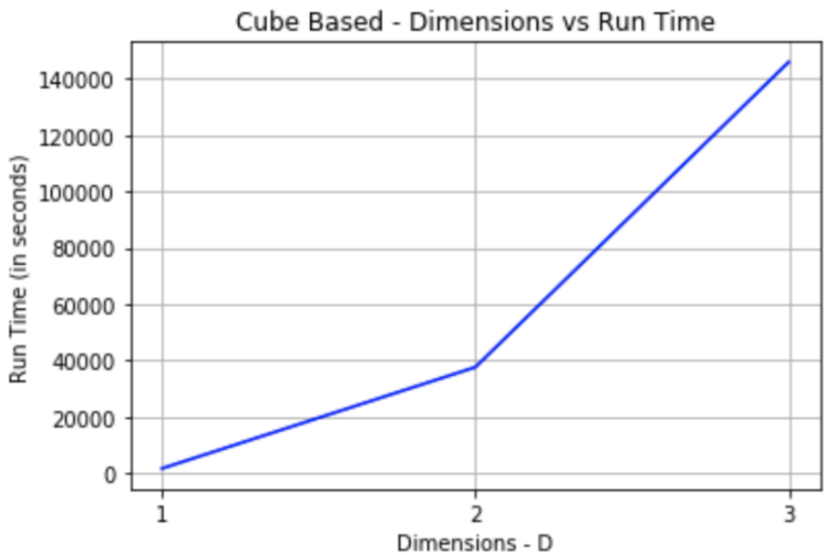
Graphs for Run time and Accuracy as a function of dimension in Monto Carlo and Cube Based Integration:

For a given 4 digits of precision, I have calculated Dimension vs Run Time and Dimension vs Accuracy in **Monto Carlo** for **D = 1 to 10** and **N in range 100000 to 38800000** and **n=100**.
For **Cube Based Integrations** I have plotted same graphs for **D= 1 to 3, K= 8889, 15502, 35607**.

Dimension vs Run Time for both methods:

It is observed that run time increases with dimension in both the cases. The run time for d=10 is Monto Carlo is closer to run time for d=4 in Cube Based. The value of K should be large in Cube based for accuracy and as dimension increases, the run time increases exponentially for it - $O(K^d)$. Hence **Monte Carlo method is more efficient** as it is observed here that the amount of time it takes to compute volume for d=9 in Monto Carlo is same as the time it takes to run d=3 in cube based integration. Python's time.time() function was used here which computes the clock time of run.

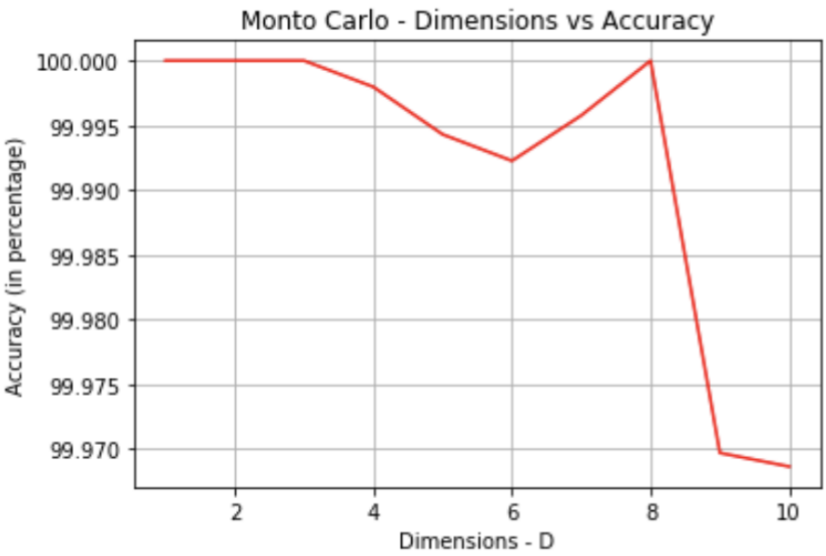


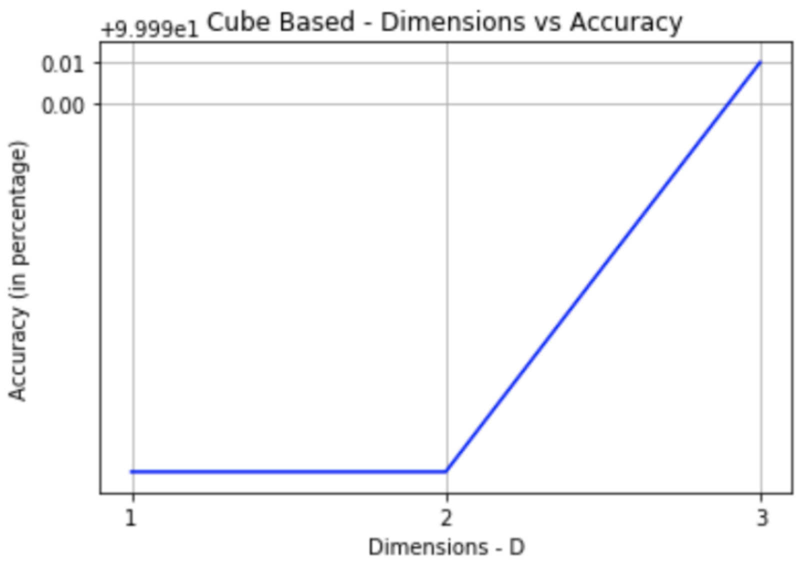


Student ID : 58328870
Student Name : Smitha Gurunathakrishnan Balagurunatan

Dimension vs Accuracy for both methods:

It is seen that for both the methods, the accuracy is in range of 99-100%, but there is a large difference in the number of dimensions it gives accuracy for. In Monto Carlo there is accuracy till d=10 whereas in Cube based there is only accuracy till d=3, beyond which it seems to run forever. Cube based method grows exponentially with the dimensions making it impractical to generate accurate results in a day's time of a computer even for d=4 as compared to Monto Carlo. Hence **Monto Carlo is more efficient** in this case.





<u>Part 2 –</u>

In this part, N is assumed to be 10^6 and K = round($N^{1/d}$). The volume has been computed using Monto Carlo and Cube Based till d=10. The run time is computed here using python's time.time() function which computes the wall clock run time and accuracy is computed as percentage using the following formula –

- **Accuracy = 100 - (100*(Absolute(Mean-Actual_vol)/Actual_vol)) %**
  - Mean is the volume determined by Monto Carlo or Cube Based and
  - Actual_vol is the volume determined by analytical formula (Wikipedia page)

Student ID : 58328870
Student Name : Smitha Gurunathakrishnan Balagurunatan

In Cube Based, the volume of the hypersphere is calculated to be –

- **Volume in Cube Based = Average(lower_bound , upper_bound)**
    - lower_bound = inside_count* $(2/k)^d$
    - upper_bound = inside_count+(intersect_count/2) *$(2/k)^d$

From experiments in part 1, it's evident that results are more efficient in Monto Carlo method. Here we can both the algorithms for $N=10^6$ and K = round($N^{1/d}$). Accuracy in **Monto Carlo is always** maintained between **99-100%** and the run time is also reasonable for the accuracy it gives. In **Cube Based, K reduces exponentially as 'd' increases** and the accuracy of 99-100% range reduces after d=2. For d=3 & 4 the accuracy is maintained decently with above 90% range but for dimensions **d>=5 Cube Based method completely falls apart and fails terribly to give accurate results for fixed N = 10^6**. The accuracy results are highlighted in red for cube based. This is because for d=10, K becomes 4 which means there are only few cubes surrounding the hypercube and finding lower and upper bound in that will lead to erroneous estimate of volume.

d	N points (10^6)	K = ($N^{1/d}$)	Monto Carlo - volume 95% confidence	Run time (seconds)	Monto Carlo - Accuracy (in percent)	Cube Based – volume	Run time (seconds)	Cube Based – Accuracy (in percent)
1	1000000	10^6	2.0±0	261.32	100	2.0	11.14	100
2	1000000	1000	3.1417±0.0001	283.027	99.996	3.1369	11.71	99.850
3	1000000	100	4.1886±0.0004	330.92	99.995	4.1026	12.37	97.942
4	1000000	32	4.9359±0.0006	364.84	99.977	4.5698	15.049	92.603
5	1000000	16	5.2639±0.009	413.28	99.998	4.4246	15.08	84.057
6	1000000	10	5.1677±0.047	432.38	100.0	4.395	13.92	85.0475
7	1000000	7	4.7241±0.060	457.97	99.985	3.75	12.21	89.466
8	1000000	6	4.06±0.073	499.076	99.967	4.6627	18.34	85.118
9	1000000	5	3.2991±0.079	584.647	99.981	8.6181	35.51	61.273
10	1000000	4	2.5436±0.080	599.69	99.741	14.0	19.93	8.976

Part 3 –

Having said that N is unknown and there will be a fixed value given for it, I have assumed N to be from 100 till 10^6 and determined pi value for d=2, n=100. I have run using Monto Carlo and Cube Based Integration. It is observed that for all ranges of a fixed N value, Monto Carlo has very high accuracy in range 99-100% whereas Cube based gives lesser accuracy for values before N=100000 i.e **K= round($N^{1/d}$).**

The accuracy as highlighted in red shows cube based failed to give accuracy in range 99-100% for N<100000. The accuracy percentage here is calculated as follows –

- **Accuracy = 100 - (100*(Absolute(Mean-Actual_vol)/Actual_vol)) %**
    - Mean is the volume determined by Monto Carlo or Cube Based and
    - Actual_vol is the volume determined by analytical formula (Wikipedia page)

The following table shows the results, it is evident that **Monto Carlo is preferred** to find pi value for **dimension d=2** because of its **more accuracy results** than cube based integration, comparable run time, for any given fixed value of N points.

Student ID : 58328870
Student Name : Smitha Gurunathakrishnan Balagurunatan

N value	**K value**	**d**	**Monto Carlo - pi value**	**Monto Carlo - Accuracy (in percent)**	**Run time (seconds)**	**Cube based - pi value**	**Cube Based – Accuracy (in percent)**	**Run time (seconds)**
100	10	2	3.1596	99.427	0.029	2.52	80.213	0.002
1000	32	2	3.1448	99.898	0.263	2.9805	94.872	0.013
10000	100	2	3.1399	99.945	3.122	3.0908	98.382	0.110
100000	316	2	3.1423	99.977	28.112	3.1275	99.551	1.155
1000000	1000	2	3.1416	100.0	293.10	3.1369	99.850	11.986