

# Monte Carlo-Simulation of DBSCAN Parameter Effects on LiDAR Point Cloud

Mohamad Nweder

Department of Computer Engineering  
University West

Trollhättan, Sweden

m.neweder@gmail.com

**Abstract**—This article will present a Carlo simulation-based study on how DBSCAN parameters *epsilon* and *min\_samples* could affect LiDAR point cloud clustering. Simulation will be done by randomly varying these parameters (Number of clusters, size of cluster and noise ratio were measured). The results will prove that DBSCAN is sensitive to parameter changes and that simulation could use as effective method to evaluate algorithm behaviors without real experiments.

**Keywords**—component, (DBSCAN, LiDAR, clustering)

## I. INTRODUCTION

Simulation method are among the most important tools used today as they save both money and time. Project simulation are often applied to study algorithm behaviour, control conditions and observe how algorithm will responds to different parameter as input without requiring real-world experiments.

Methods used in simulation are applied in research to study and observe algorithm behaviour under complex or uncertain conditions conditions [1].

In this experiments, a Monto Carlo Simulation was used to study behaviour of the DBSCAN clustering algorithm when its parameters change, using LiDAR point-cloud data obtained from PhD research work conducted at Luleå University of Technology (LTU) under the supervision of Professor Ramin Karim [1]. The simulation executed DBSCAN repeatedly while randomly varying the parameters (*epsilon* and *min\_samples*) within predefined intervals. Each simulation run generated different values representing numbers of cluster, size of cluster the biggest cluster and ratio of noise points.

The main goal of this simulation is to show how Monto Carlo simulation method can be used effectively to examine the sensitivity of changing parameters in DBSCAN algorithm without doing any physical or real world experiments.

## II. METHOD

Simulation structured to show and confirm how DBSCAN – algorithm reacts when we chang parameters repeatedly while running the code. Experiment was performed with python and used different libraies.

### A. Simulation Steps

Simulation followed Monte Carlo principle, which mean that the same processes get repeated lot of times with different randomized data.

Every iteration was executed by DBSCAN algorithm with random parameters value:

- Epsilon ( $\epsilon$ ): Radius define how near points must be to be calculate as neighbours.
- *min\_samples*: Lowest number of head points requires to build cluster.

Both parameters randomized between  $\epsilon$  (0.05 and 0.25) and *min\_samples* (3 and 7).

Each running saved three results :

- 1- Number of founded cluster.
- 2- Biggest cluster size.
- 3- Point parts that classed as cluster

### B. Matrial of data

Used data was consisted of LiDAR point-cloud data used in this simulation originates from PhD research conducted at Luleå University of Technology (LTU) under the supervision of Professor Ramin Karim [1].

Each datapoint had three coordinator (X,Y,Z). Ground level calculated with histogram that shows over Z-values to filter underground points. After filtering, the experiment will used only the above ground points.

### C. Implementation

Experiment runs totally 2000 iteration were run using DBSCAN with different random parameters.

To ensure that reproducibility used with solid *seed-value* with random number generate.

The results from each used iteration saved automatically to files in csv format (all iteration) and JSON format (summery).

Figures were generated showing histogram, cumulative distribution, convergent and sensitive measurement for all runed simulations.

### III. SIMULATION RESULTS

Simulation runs with totally 2000 iterations with help of DBSCAN algorithm. Parameters *epsilon* and *min samples* was randomized between given intervals. Each iteration saved the result locally in two different format, JSON and CSV. It gives the experiments opportunity to verify all runs and tests correctly and to present them as figures.

Two main figure will be purge down will shows exemplify results of simulation.

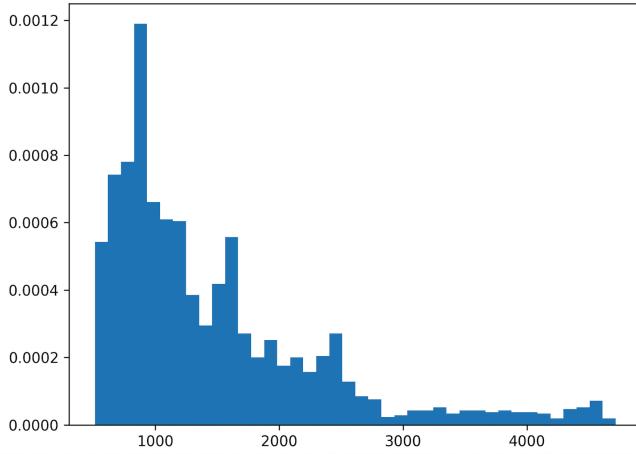


Fig. 1 Histogram of cluster counts.

Histogram in figure 1 shows how many cluster that founded between different iterations.

The X-axis present the numbers of clusters founded with the help of DBSCAN in each run. Every bar corresponds to an interval of cluster counts. Those values come from recorded results of each simulation run, where the number of clusters found by DBSCAN was saved automatically for every iteration.

The Y-axis shows how many iterations produced a specific range of cluster counts. If the bar was zero, then we don't have any cluster founded. The higher bar we have, the more iterations resulted in that range of cluster numbers.

Distribution stretches from the lowest value (fewer cluster) to higher values (many small cluster), which ensures that the random variations values of parameters worked correctly and that simulation performed as planned. Histogram also shows that most of runs resulted in values within the middle range of the interval, which means that our simulation generated stable and realistic spreads of the results.

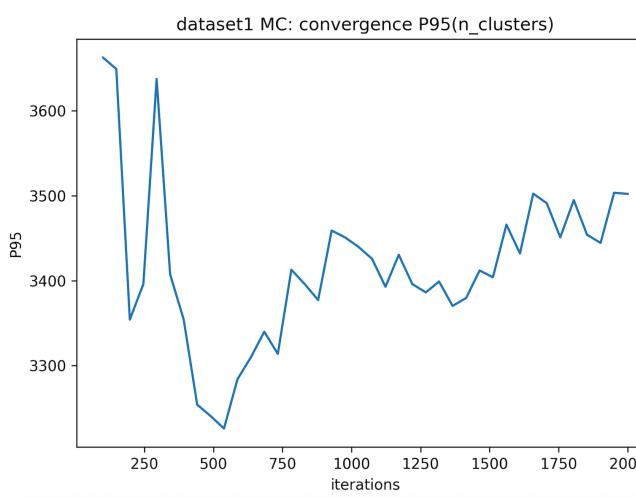


Fig 2 Convergence of cluster count (95th percentile).

Fig 2 explain how the number of clusters (calculated as the 95th percentile) changes and stabilizes as the number of Monte Carlo iterations increases.

X-axis shows 95th percentile of the number of cluster found at each step, calculated progressively as the simulation proceeds. It also shows that the result become stable after many iterations, which proves that the Carlo simulation has reached convergence. This confirms that the model works as expected and that's around on 2000 iterations to see stable result.

#### IV. DISSCIOSN

The simulation proved how useful the Monte Carlo method can be for investigating the sensitivity of parameter changes in DBSCAN without performing any real physical experiments.

The results also proved that our algorithm reacts strongly when we used and changed different values in  $\epsilon$  (epsilon) and even `min_samples`. That means that our parameters have a big impact on how many clusters could be formed and how large the clusters are. We also saw that when we made 2000 different runs, it proved to be stable and enough to produce reproducible outcomes.

That proves that our developed simulation could be used for similar experiments but with different data or different algorithms to test how sensitive it could be when changing different parameters.

In the future, this simulation could be developed further, maybe by using another algorithm to compare this experiment with the new one and see what the differences would be, as well as to gain a better understanding of how sensitive it could be under different scenarios.

#### V. CONCLUSION

The simulation was done using the Monte Carlo method and showed that the simulation worked to make repeating runs and to investigate changes that happen when we change our parameters in a controlled and structured way.

By making 2000 different runs, it showed stability and reproducibility, proving that the simulation worked as expected and planned.

The experiment also showed that we can use this type of simulation in the future to reach results faster without physical testing, and it could be used for other studies with different datasets or algorithms.

#### REFERENCES

- [1] Luleå University of Technology, *LiDAR point-cloud data from PhD research supervised by Prof. Ramin Karim*, Luleå, Sweden, 2025. Clerk Maxwell, A Treatise on Electricity and Magnetism, 3rd ed., vol.

GITHUB

<https://github.com/Nweder/DTA400>