Point Process Modelling of Point Patterns & Sampling Additional Realizations

Sunia Tanweer[1]

[1] Dept. of Mechanical Engineering, Michigan State University, East Lansing, USA

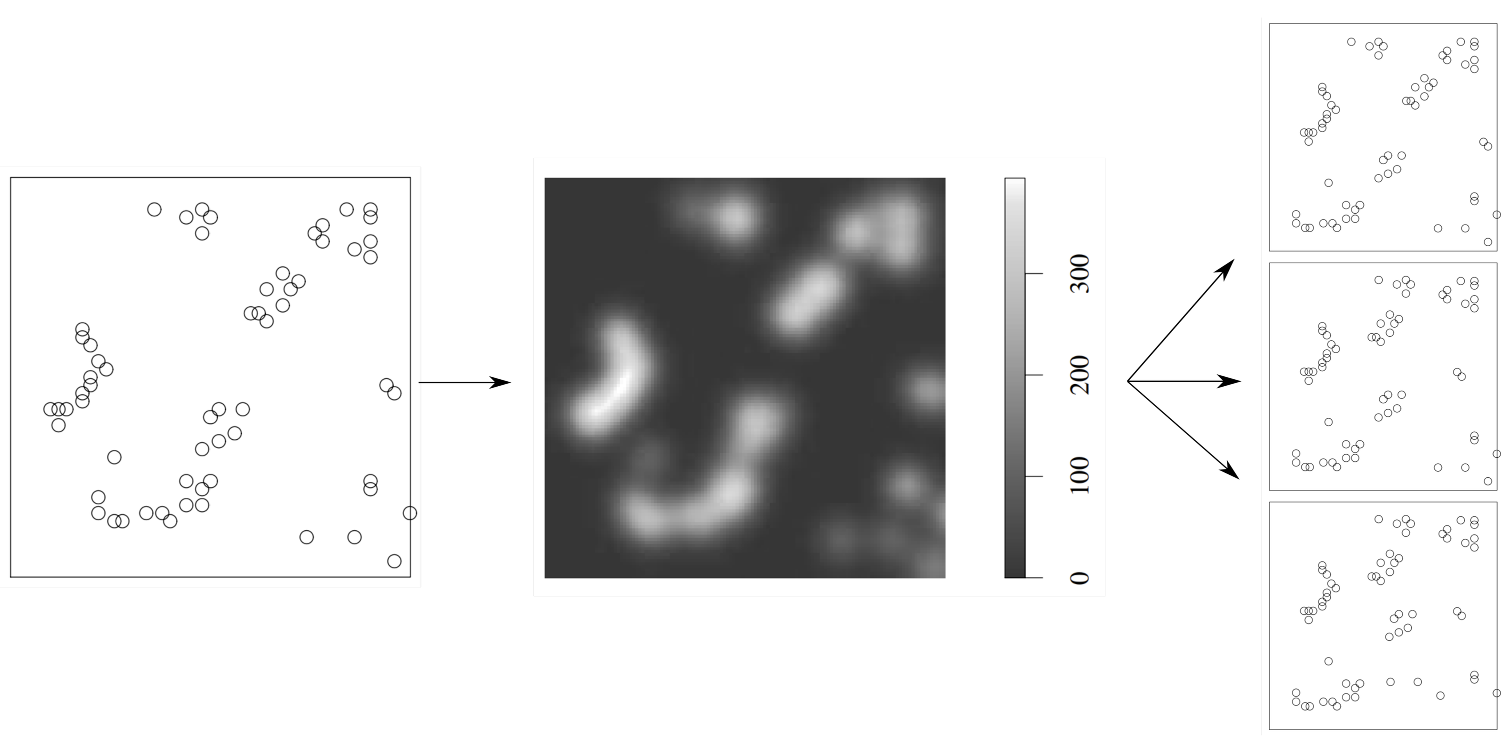
Abstract

Spatial point patterns serve as fundamental tools for understanding spatial distributions. Modeling of these patterns is a powerful technique which allows establishing probabilistic models to capture the generation of observed point patterns—and generate additional samples or realizations of spatial point patterns. This ability is invaluable in scenarios where data may be limited, unreliable, or insufficient, e.g. Cosmological Background Noise (CMB) and Big Data. Data deficiencies can be compensated for and statistical reliability of any analyses can be enhanced. This project focuses on modeling spatial point patterns in two-dimensional space using the Strauss model and generating additional samples via Monte Carlo methods. We parallelize the simulation using OpenMP and MPI for improved efficiency. The parallelized implementation accelerates the analysis of spatial point patterns, showcasing both accuracy and scalability over a range of problem sizes. A strong, weak and thread-to-thread study are conducted for performance analysis which show near-to-ideal performance.

Point Process Modelling

Basic idea is outlined as follows [1]:

1. Assume a stochastic process χ with expectation E exists.
2. Original point pattern P is realization of χ.
3. Use P to estimate E.
4. Sample new patterns P_i given E and P using monte carlo methods.



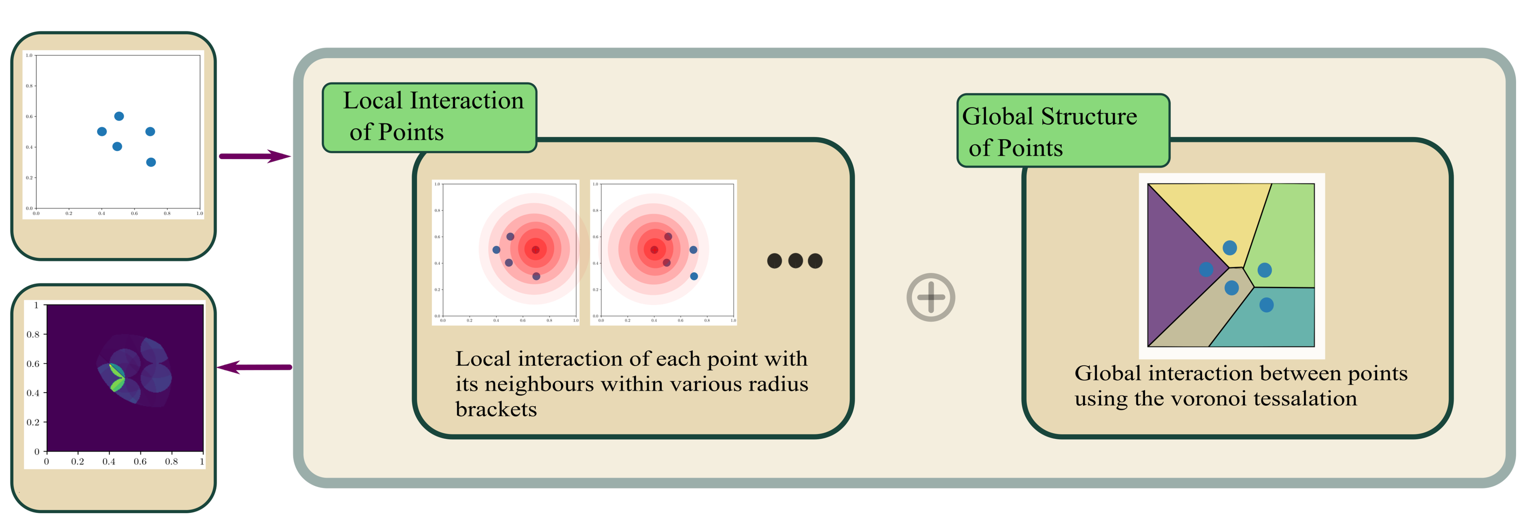
Computing Expectation

Define a rule for interaction between points: typically global and local

Local: distances, nearest neighbours, fixed radius, etc.

Global: kernel density, tesselations, fixed number per area, etc.

Example Pairwise Interaction Point Process Model [2]:



Strauss Model

A Strauss process is defined by the following parameters:

λ: an intensity of points per unit area

r: radius such that other points are less likely to exist inside this

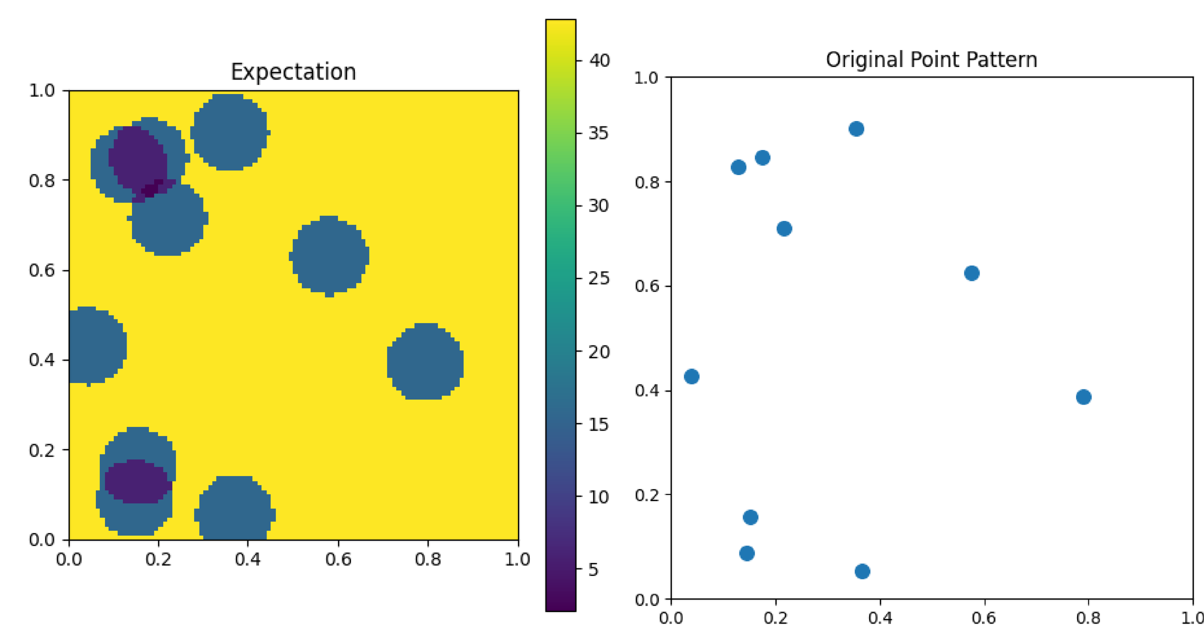
$$p(\mathbf{x}) = \lambda \exp(-n)$$

Monte-Carlo Markov Chain Method

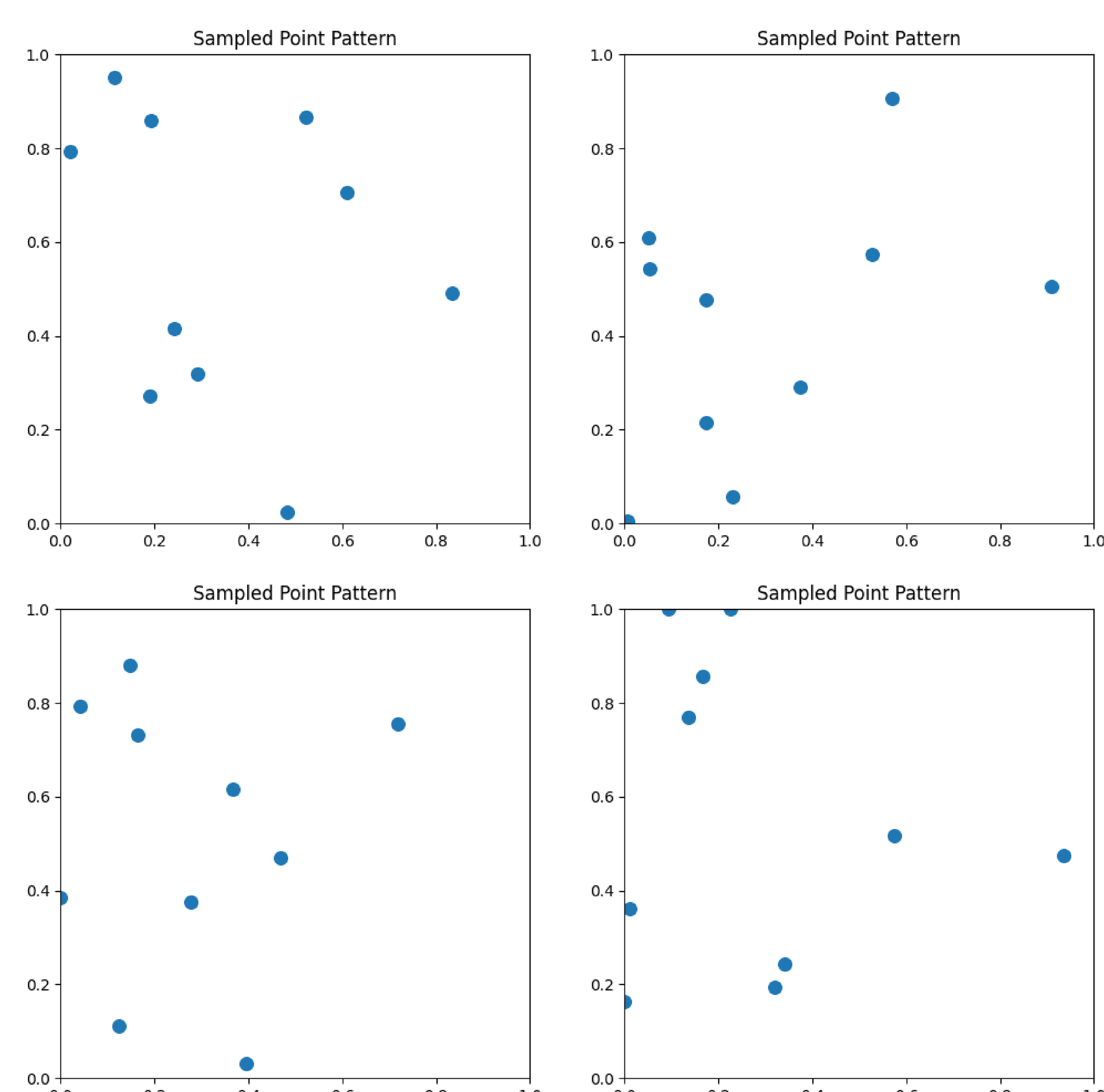
1. Start with an initial state.
2. Propose a new state from a proposal distribution.
3. Calculate the acceptance probability for the proposed state.
4. Accept the proposed state if the acceptance probability is greater than $D \sim \mathcal{U}(0, 1)$.
5. Repeat steps 2-4 for a burn-in period.

Example

Strauss Modelling

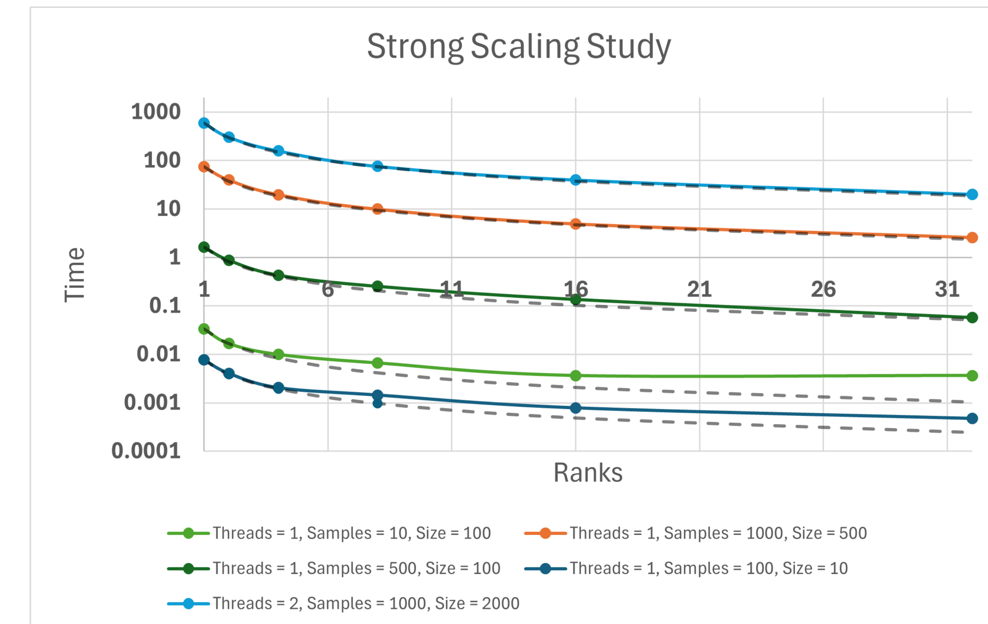


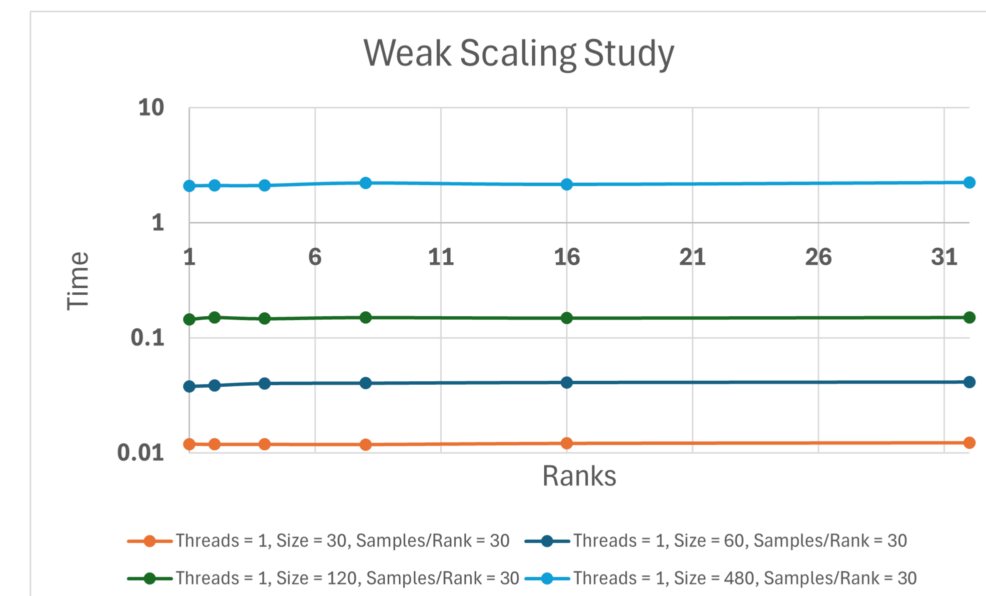
Sampled Realizations

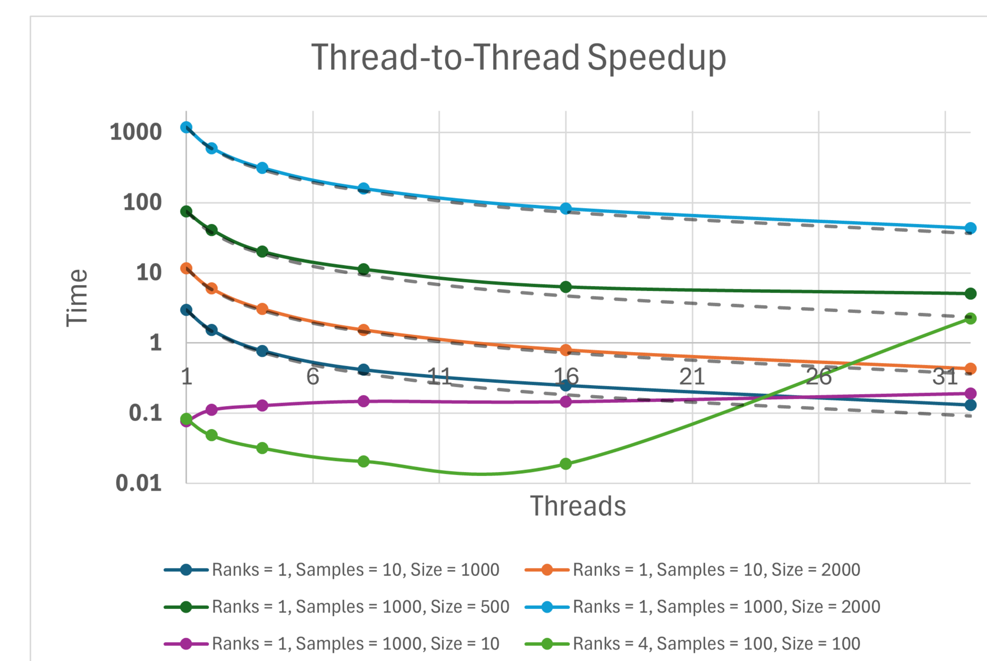


Parallelization

N patterns generated over MPI ranks. New point proposed for each of the M points in each pattern over OpenMP threads.







Conclusion

1. Efficient parallelization through the combined use of OpenMP and MPI.
2. Significant performance improvements and close to ideal behavior.
3. **Limitations** due to computer specifications and communication overhead at small problem sizes.
4. **Future Work** of exploring other parallelization strategies e.g. adding another parallelization for the burn-in loop.

References

[1] Adrian Baddeley, Ege Rubak, and Rolf Turner.
Spatial Point Patterns.
Chapman & Hall/CRC Interdisciplinary Statistics. Apple Academic Press, Oakville, MO, November 2015.

[2] Sunia Tanweer and Firas A. Khasawneh.
Topological detection of phenomenological bifurcations with unreliable kernel densities, 2024.