Walk on Subdomains
A Sharp Restart Point Evaluation using Machine Learning

By Christoph Hagenauer – April 29th 2025

In loving memory of Oma

Topics

- Introduction
- Background
- Methodology
- Machine Learning
- Conclusion
- Lessons learned

Introduction

Algorithm estimates electrostic energy of a biomolecule in a solution

It uses two Monte Carlo Algorithms

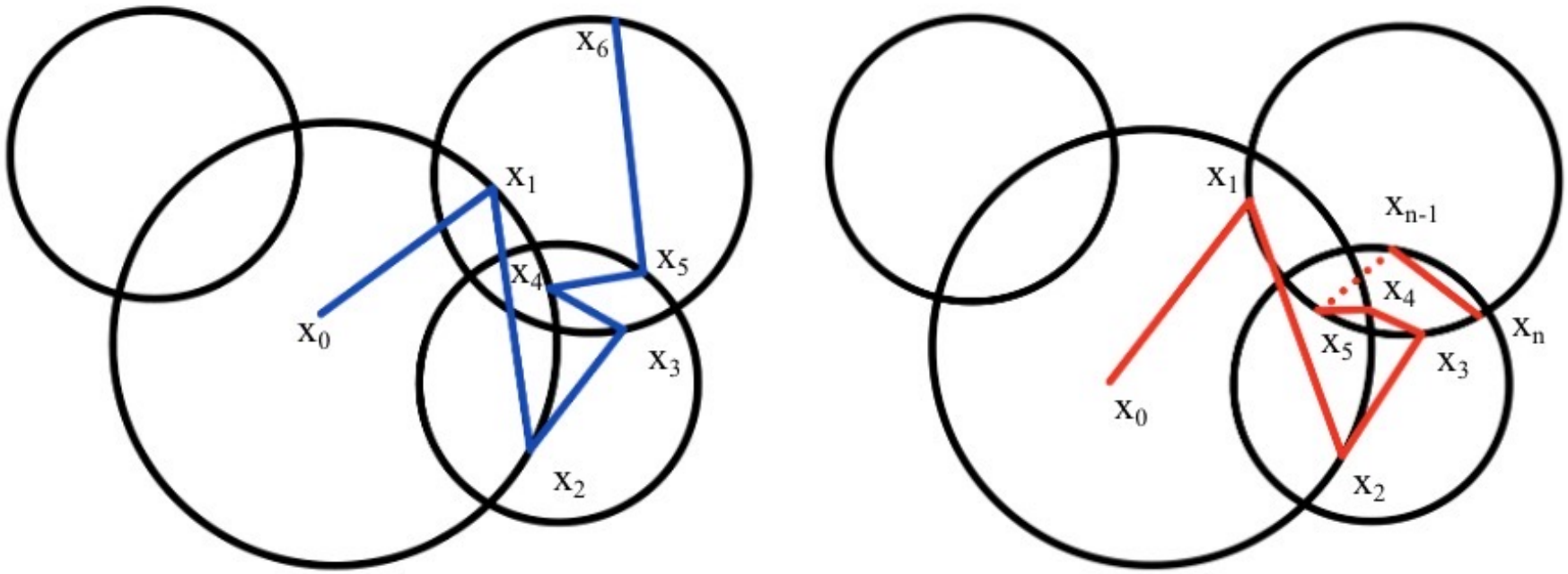
Walk on Spheres – WoS

- Estimates outside portion of linear Poisson-Boltzmann equation

Walk on Subdomains – WoSD

- Estimates internal portion of linear Poisson-Boltzmann equation

Introduction – Description of Challenge



Introduction Focus of this work

Focus on WoSD Sharp Restart Point

Gather lots of results for machine learning

Finding a good restart point, defined by:

- It gives the same underlying estimate
- It provides some speedup compared to the non-restart version
- It stabilizes overall runtime

This is very important for the parallel version of the code

Background Walk on Spheres

- Used to solve Dirichlet problems for variety of elliptic and parabolic partial differential equations
- WoS first introduced by Müller
- Solved the N-dimensional Dirichlet problem for the Laplace equation
- Introduced Monte Carlo methods using stochastic models which are Markov processes
- These techniques are proven to converge with probability 1
- Yield statistical estimate for N-dimensional Dirichlet problem

Background WoS Müller

- D is bounded finitely connect N-dimensonal Euclidean space
- $\Gamma[D]$ boundary points of domain D
- Point x with x having coordinates $(x_1, x_2, x_3, \dots, x_N)$
- There exists a continues function $f(x)$
- The task is to define a function $u(x)$ which is continues on $\mathrm{D} + \Gamma[D]$

$$\Delta^2 u(x) = \sum_{i=1}^{N} \frac{\partial^2 u(x)}{\partial^2 x_i^2} = 0,$$

$$x \in D, u(x) = f(x), \qquad x \in \Gamma[D]$$

Background WoS Müller

- Proved that first-passage probabilities of Brownian motion can be used to estimate the N-dimensional Dirichlet problem
- Proves Spherical process can be used to simulate Brownian motion – which we know as WoS
- Brownian motion is defined as follows:
  - Probability space (Ω, ϵ, Pr), with $\Omega = \{\omega\}$ is a set of elements ω, $\epsilon = \{E\}$, which is a Borel field of subsets E of Ω. $Pr(E)$ is a probability measure defined on ϵ that is countably additive and satisfies the normalization condition $Pr(\Omega) = 1$.
- Defines $X(t, \omega)$ as the well-known N-dimensional Brownian motion process starting from x, with $X(t, \omega) = \{(x^{(1)}(t, \omega), x^{(2)}(t, \omega), x^{(3)}(t, \omega), \ldots, x^{(N)}(t, \omega)) \mid 0 \leq t < \infty, \omega \in \Omega\}$

Background WoS Müller

- Prove the shperical process with:
  - Given any point x belonging to a domain D with boundary $\Gamma[D]$, then with probability 1, the Spherical process originating from x converges to a point of the boundary $\Gamma[\mathrm{D}]$.
- Provides termination clause to terminate process – δ-truncation
- Once walk within δ of $\Gamma[D]$ walk is terminated
- δ-truncation truncation is later referred to as ϵ-shell

Background Alternative to WoS – Green´s function

- For a given geometry, the boundary Green´s function is equal to the first passage-probability distribution of the Brownian motion
- By precalculating Green´s function the exact first-passage probability can be used to terminate walks
- ϵ-shell can be omitted and exact solution can be calculated
- The catch:
  - Green´s function is only known for a few shapes
  - Therefore we use WoS

Background Other Applications of WoS

- Haji-Sheikh and Sparrow expand on WoS to provide solution of heat conduction problems
- Booth used weighted WoS to solve homogeneous elliptic partial differential equations with constant coefficients
- Hwang, Mascagni, and Won use WoS to compute the capacitance of the unit cube

Background WoS connection WoSD

- Both equivalent to simulating first-passage location of Brownian motion from a domain
- WoSD essentially builds on WoS and has the advantage of being able to simulate more complex shapes
- WoSD is faster than WoS

Background Walk on Subdomains

- A given point is within a subdomain then it is possible to find an exit point of said subdomain
- If in another subdomain, repeat process until at exterior
- Fastest and most precise method would be Green´s function with the limitations as explained earlier

The Algorithm - Biomolecule

- Molecule can be considered a domain G with a boundary $\Gamma[G]$
- Can consider G as the union of intersecting spheres B, where each sphere represents an atom

$$G = \cup_{m=1}^{M} B(x_m, r_m)$$

- where x_m is the center and r_m is the radius of the sphere

The Algorithm - WoS

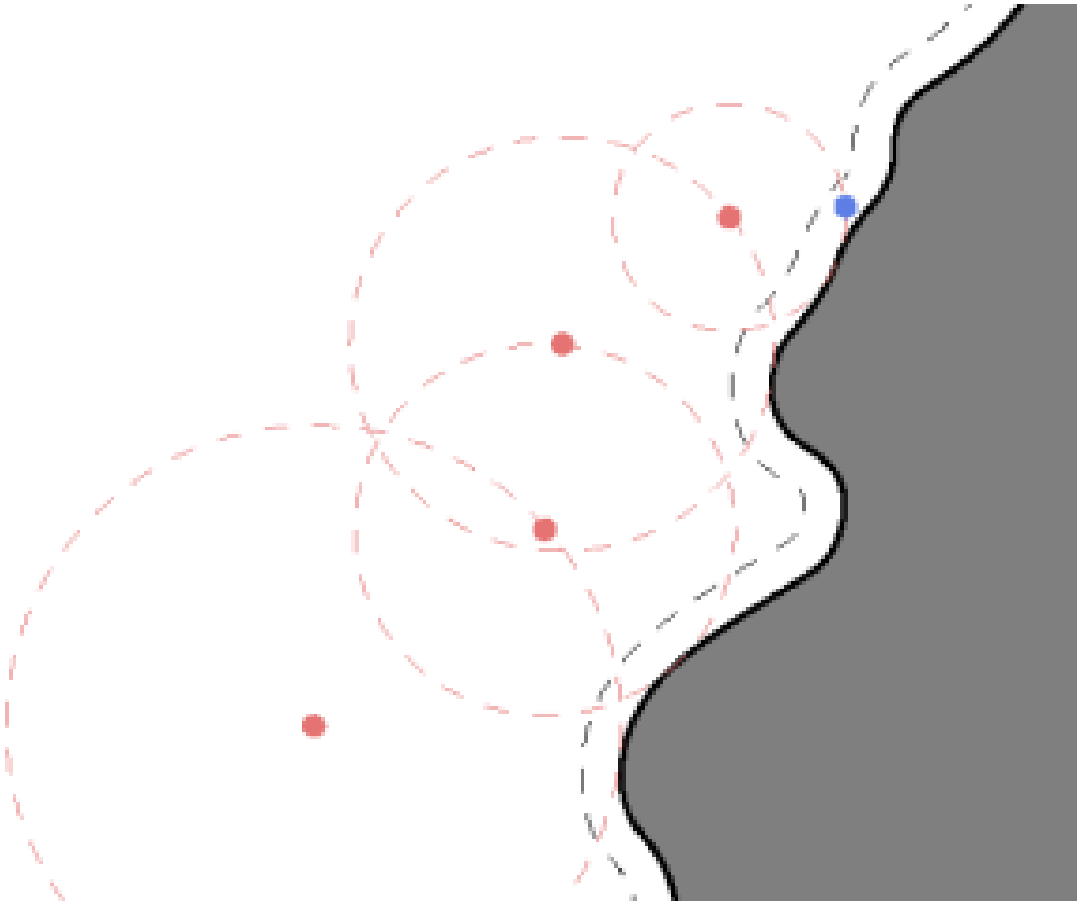
- Charge is described by lineraized Poisson-Boltzmann equation

$$\Delta\, u(r) = \kappa^2 * u(x)$$

$$\kappa^2 = \frac{8\,\pi\, c_b e^2}{\epsilon_e k_b \tau}$$

- where c_b is the bulk salt concentration, e is the fundamental charge, ϵ_e is the exterior dielectric constant, k_b is the Boltzmann constant, and τ is the absolute temperature
- WoS uses ϵ-shell to terminate walks

The Algorithm – WoS Steps



The Algorithm - WoSD

- Charge distribution modled by

$$\rho(x) = \sum_{m=1}^{M} q_m \delta(x - x_m)$$

- where q_m is the electrical charge, δ is the Dirac delta function, and the electrostatic potential $u(x)$ is the solution to a boundary value problem of Poisson's equation:

$$\nabla u(x) = -\frac{1}{\epsilon_i} * \rho(x), \qquad x \in G, \qquad with\ G \subset \mathbb{R}^3$$

- where ϵ_i is the interior dielectric permittivity and the domain G consisting of a union M overlapping spheres
- With this we can represent the potential as the sum of two functions $u(x) = u^{(0)}(x) + g(x)$, with

$$g(x) = \sum_{m=1}^{M} \frac{q_m}{4\,\pi\epsilon_i} \frac{1}{|x - x_m|}$$

- For grounded molecule the boundary values of u(x) are represented by

$$u(x) = 0\ or\ u^{(0)} = -g(x), \qquad x \in \Gamma[G]$$

WoSD - Observations

- An unbiased estimator for each subdomain is unbiased on the entire domain
- Using WoSD to sample to the exit point of G has the same properties as an estimate based on direct simulation of the exit point
- For a sphere S_x, centered at x_c with radius of r, we have Poisson's formula for a function u_L, which satisfies the Laplace equation at every point $x \in S(x_c, r)$:

$$u_L(\mathrm{x}) = \int_{S(x_c,r)} p_p(x \rightarrow y)\, u(y)\, d\sigma(y)$$

- where the Poisson kernel is

$$p_p(x \rightarrow y) = \frac{1}{4\,\pi\, r} \frac{r^2 - |x - x_2|^2}{|x - y|^4}$$

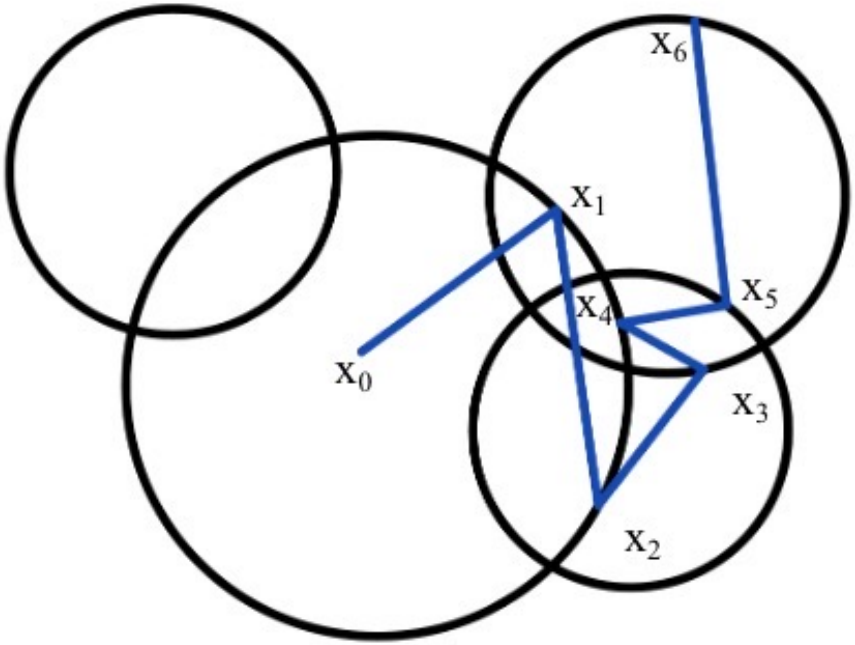
The Algorithm – Sharp Restart in WoSD

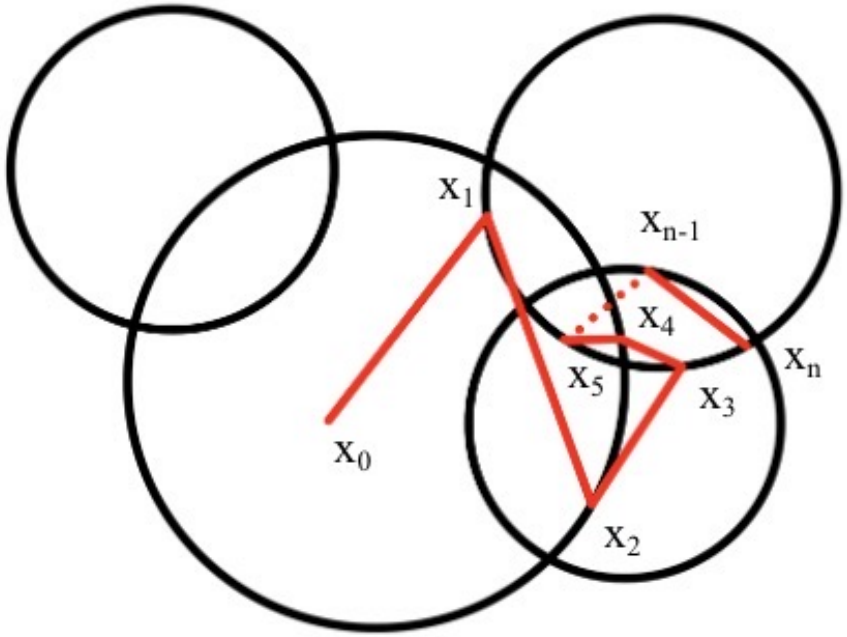
- Some walks become entrapped in the geometry which leads the walker to oscillate between two atoms
- Entrapment happens due to low connectivity at any given point in the molecule and the bias of the Poisson kernel to sample towards points near the current walker´s position
- Sharp restart is essential to make parallel version of the code run more efficiently
- Specific implementation is based on research regarding first passage under restart (FPUR)
  - Best runtime is achieved by implementing a sharp restart point after a non-random number of steps

The Algorithm – Sharp Restart in WoSD

- WoSD sharp restart refers to restarting a portion of the algorithm after a non-random number of steps R
- Restarting the algorithm did not significantly change the result of the calculation, which is true for larger values of R
- Smaller values for R more likely to bias the estimate, but reduce the overall runtime of the algorithm
- Two different approaches to choose R:
  - One Larger Restart Point of 10,000,000
  - Calculated Restart Point (based on linear model)

The Algorithm – WoSD steps



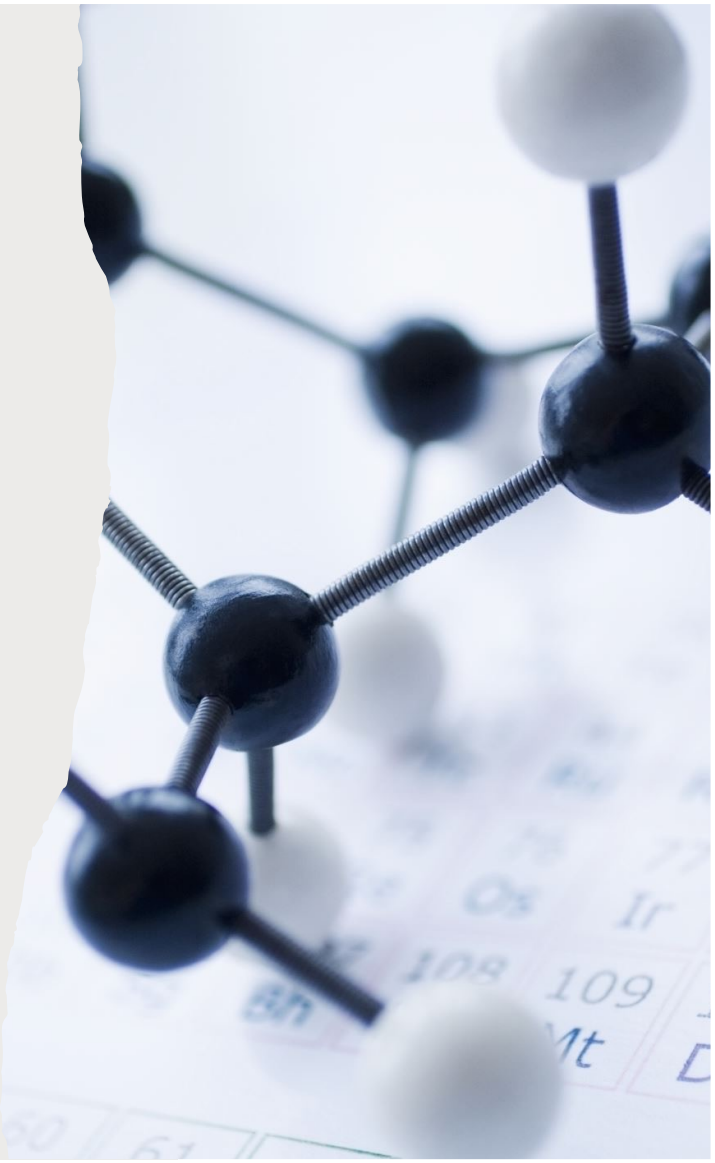


Goal for this work

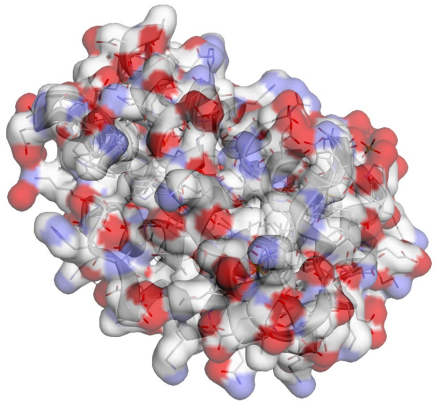
- Predict a good restart point which is defined as follows:
  - It gives the same underlying estimate
  - It provides some speedup compared to the non-restart version
  - It stabilizes overall runtime
- Proposing two-step process:
  - Predict if the result is valid
  - Predict the runtime

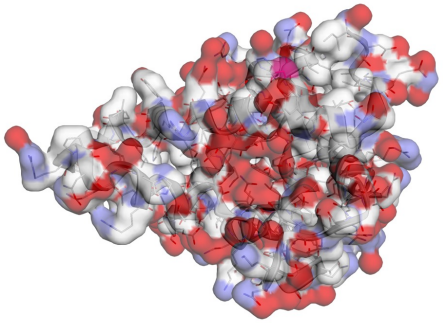
Methodology – Selecting Molecules

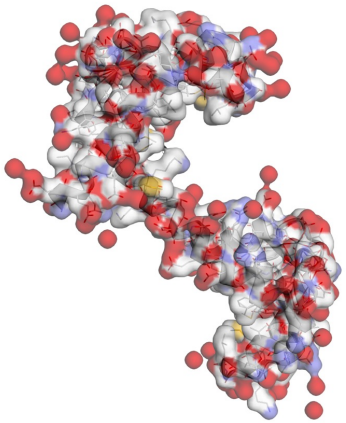
- Started with 20 handpicked molecules
  - Paid attention to shape, size, and connectivity
- Collected 578 molecules with rcsapi – Protein databank
  - Structure similarity search with threshold > 0.7
- Linearized Poisson Boltzmann solver requires coordinates, charge, and radii for each atom of the molecule
- Converted molecules using pdb2pqr – parameters were kept consistent with previous work done on this algorithm
- Had 562 molecules to use after conversion

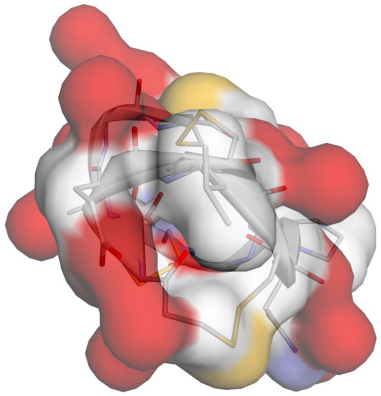


Methodology – some Sample Molecules









Methodology – Algorithm Results

- Used USCB´s partition in USC´s high performance cluster
  - Each node equipt with Intel Xeon Platinum 8260 CPU @2.40 GHz and 192 GB of system memory
- Uses Slurm to schedule jobs
- Each job for this work was run using 1 Node and 1 Core
- Algorithm was run with and without restart, as well as geometry calculation
  - Used 20 seeds and 20 restart points for each molecule
- Algorithmic parameters were kept consistent with previous work

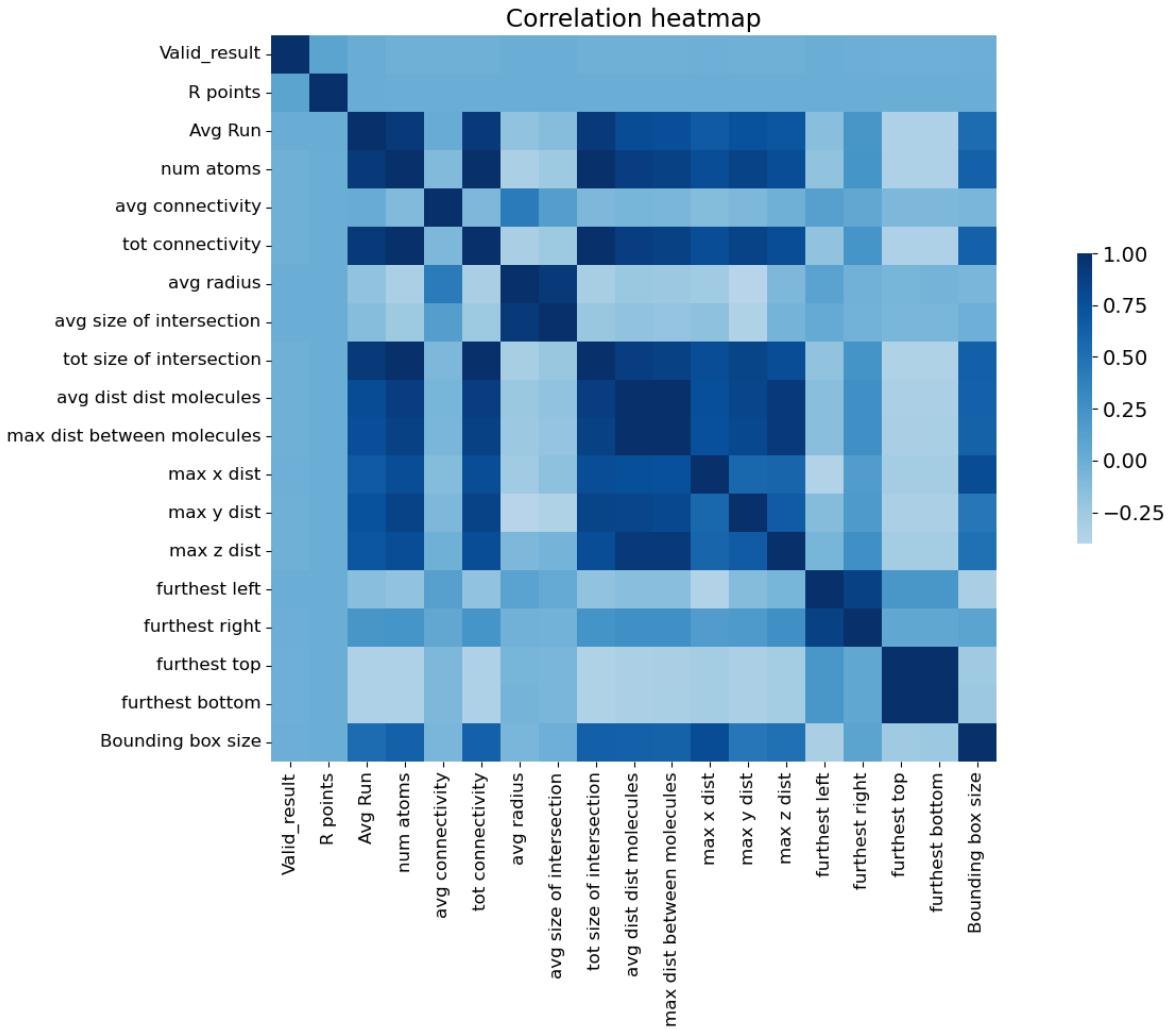
Methodology – Parallel Version

- Originally planned to use parallel version
- Discovered a bug during initial testing which caused inconsistencies within the estimates
- Therefore, the serial version was used

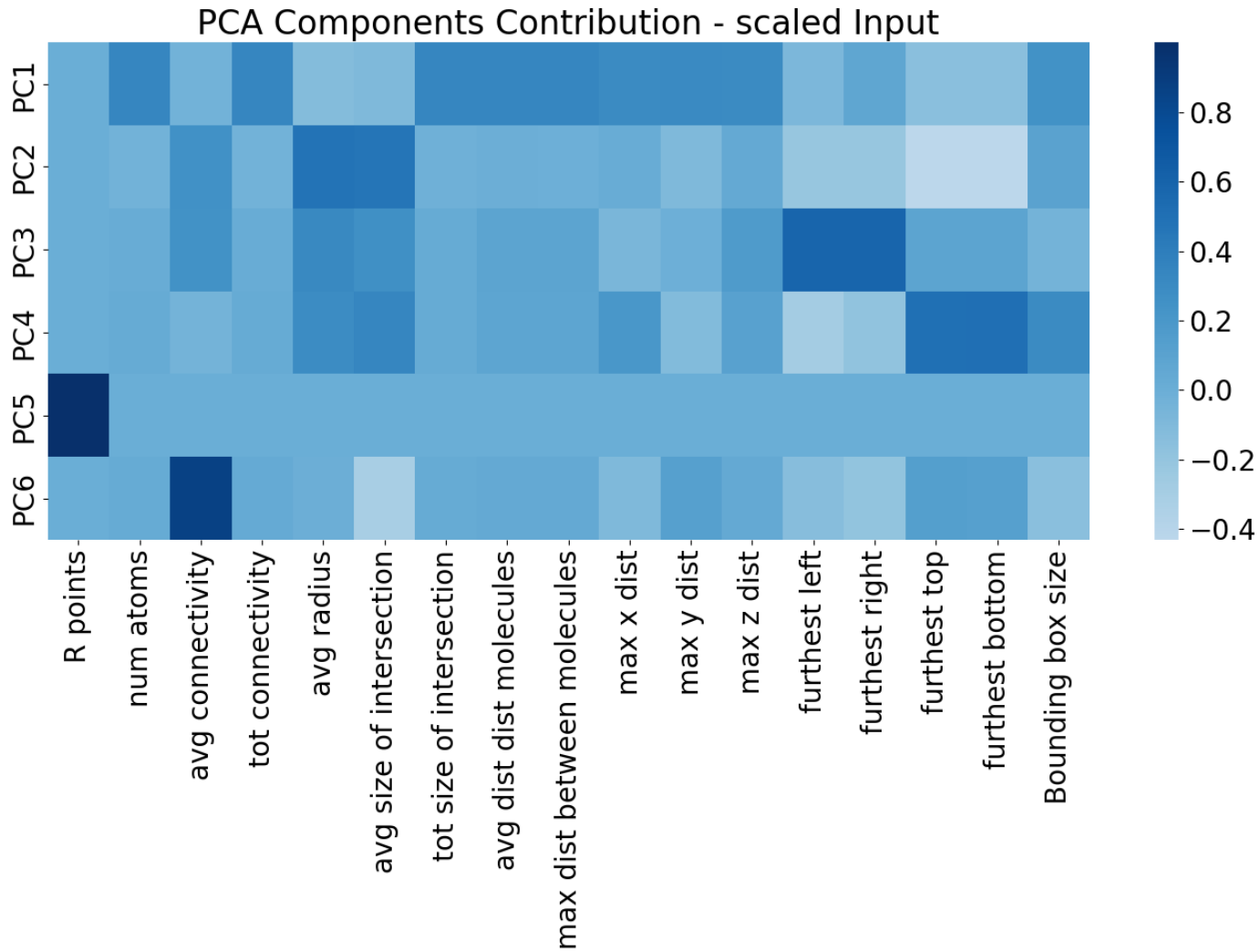
Machine Learning – Preparing the Data

- Used python to collect all the individual results
- Results include
  - Geometry, Runtime, Estimate, Error, Walk length
- Calculated Statistic metrics
- Validated restart estimates using ANOVA test with significance of $p < 0.05$

Machin


Correlation heatmap
Valid_result
R points
Avg Run
num atoms
avg connectivity
tot connectivity
avg radius
avg size of intersection
tot size of intersection
avg dist dist molecules
max dist between molecules
max x dist
max y dist
max z dist
furthest left
furthest right
furthest top
furthest bottom
Bounding box size
1.00
0.75
0.50
0.25
0.00
−0.25

Macl


PCA Components Contribution - scaled Input
PC1
PC2
PC3
PC4
PC5
PC6
R points
num atoms
avg connectivity
tot connectivity
avg radius
avg size of intersection
tot size of intersection
avg dist dist molecules
max dist between molecules
max x dist
max y dist
max z dist
furthest left
furthest right
furthest top
furthest bottom
Bounding box size
0.8
0.6
0.4
0.2
0.0
−0.2
−0.4

Machine Learning – Scaling

- Sclaed all continues data with Standard Scaler

$$z = (x - u)/s$$

- u is sample mean and s is sample standard deviation
- Scaling is required as columns have vastly differing scales
- After scaling columns have a mean of 0 and standard deviation of 1

Machine Learning – Balance in the target variable

R point	Total	Positive	Negative	Pos [%]	Neg [%]
20	562	0	562	0	100
25	560	12	548	2.14	97.86
50	562	522	40	92.88	7.12
75	562	533	29	94.84	5.16
100	562	536	26	95.37	4.63
125	562	546	16	97.15	2.85
150	562	551	11	98.04	1.96
175	562	545	17	96.98	3.02
200	562	555	7	98.75	1.25
225	562	554	8	98.58	1.42
250	562	552	10	98.22	1.78
500	562	556	6	98.93	1.07
1000	562	556	6	98.93	1.07
2500	562	554	8	98.58	1.42
5000	562	555	7	98.75	1.25
10000	562	556	6	98.93	1.07
100000	562	556	6	98.93	1.07
1000000	562	558	4	99.29	0.71
10000000	562	557	5	99.11	0.89
100000000	562	559	3	99.47	0.53
Total	11238	9913	1325	88.21	11.79

Machine Learning – Predicting if the result is going to be valid

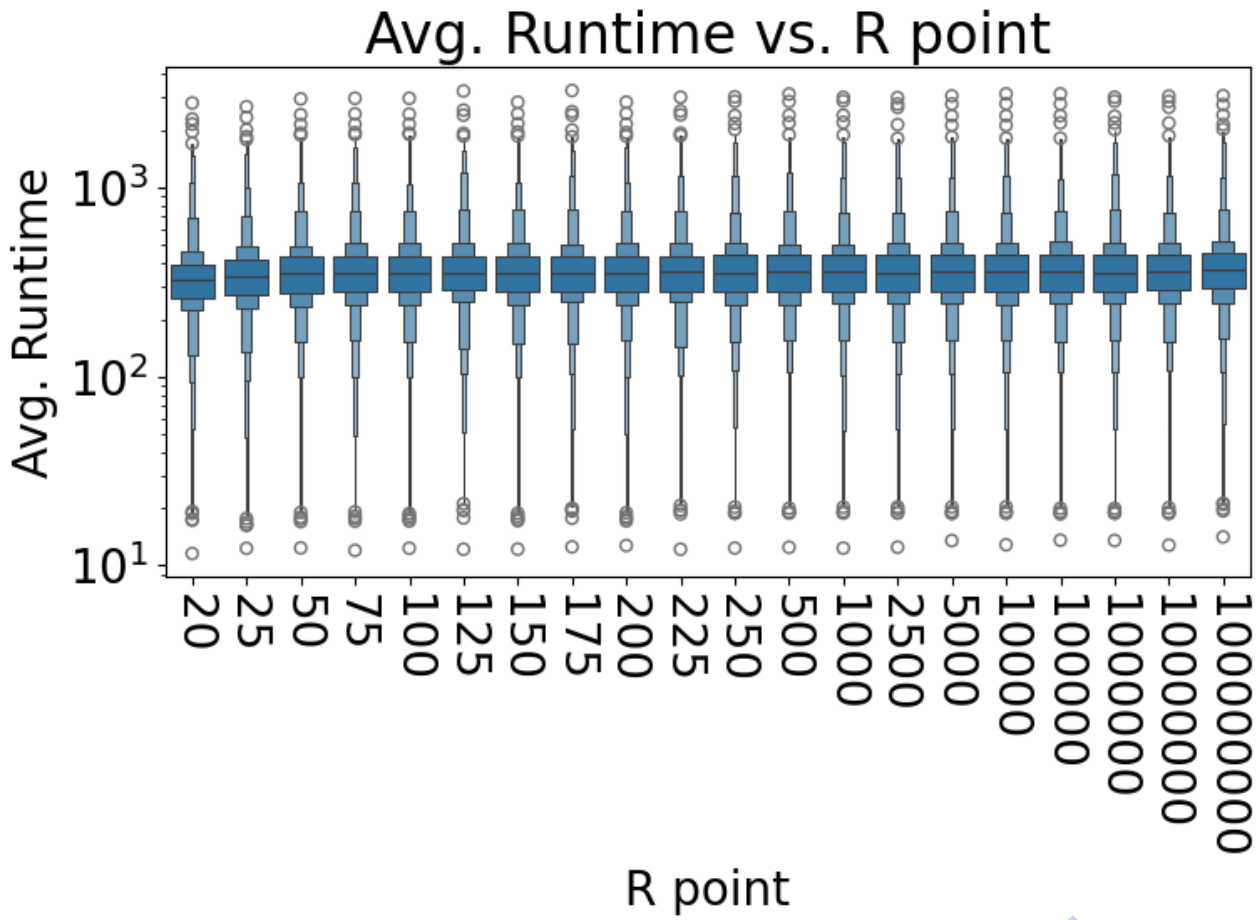
- Data was split 60/40
- Scaled the data
- Focused on ensemble classifier
  - Gradient Boositing Classifier
  - Random Forest Classifier
  - Extra Trees Classifier
  - Bagging Classifier
  - Ada Boosting Classifier

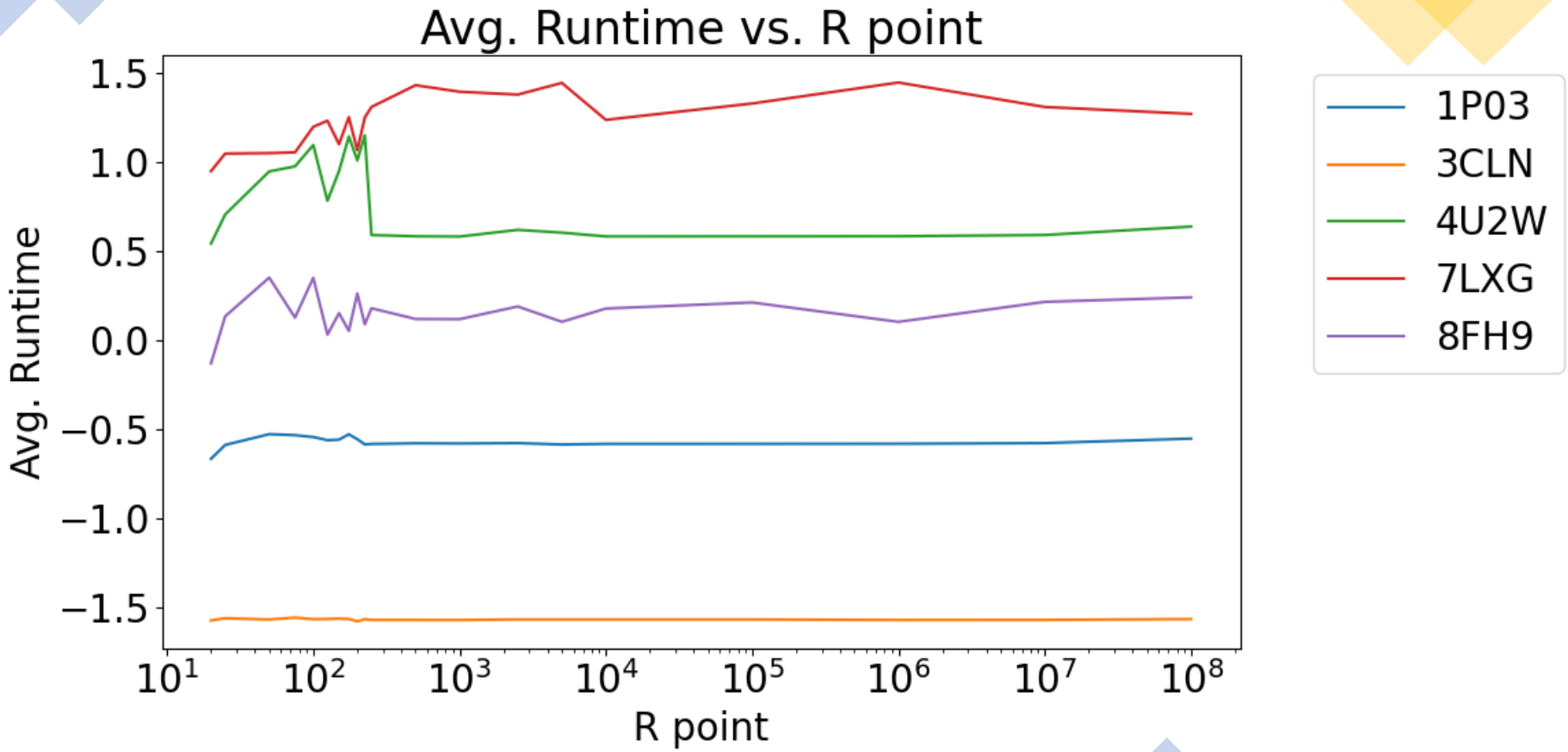
Results for Classification

Model	Accuracy	Precision	Recall	F-1 Score	AUC
Gradient Boosting Classifier	98%	98%	81%	99%	90%
Random Forest Classifier	97%	98%	82%	98%	91%
Extra Trees Classifier	91%	94%	52%	95%	74%
Bagging Classifier	97%	98%	82%	98%	98%
Ada Boosting Classifier	98%	98%	81%	99%	91%

Predicting Runtime

- Focus on relationship between geometry and restart point with runtime
- Look at
  - overall effect
  - molecule specific effect


Avg. Runtime vs. R point
Avg. Runtime
10^3
10^2
10^1
20
25
50
75
100
125
150
175
200
225
250
500
1000
2500
5000
10000
100000
1000000
10000000
100000000
R point


Avg. Runtime vs. R point
Avg. Runtime
R point
1.5
1.0
0.5
0.0
−0.5
−1.0
−1.5
10^1
10^2
10^3
10^4
10^5
10^6
10^7
10^8
1P03
3CLN
4U2W
7LXG
8FH9

Predicting Runtime – Models

Regressor	MSE	RMSE	R2-score
Gradient Boosting Regressor	0.01	0.11	0.99
Random Forest Regressor	0.01	0.08	0.99
Extra Trees Regressor	0.01	0.09	0.99

Conclusion

- Showed feasibility of implementing machine learning to predict good restart point
- Two-step process – based on geometry
  - Predicting the validity of the result
  - Predicting the runtime
- Sharp restart stabilizes runtime
- Have to look at individual results to see differences
- True speed up comes from parallelization
- Do not implement for serial version as the speedup is marginal

Lessons learned

- Start early
- Meet with your advisor regularly
- Do NOT neglect your background and spend enough time on it
- Plan enough time for writing as it is very different to your regular papers
- Do NOT get discouraged by struggles, when you struggle you learn the most
- Expect yourself to look at this work at some point and reconsider every step

References

- [1] Hervé Abdi and Lynne J. Williams. “Principal component analysis”. In: WIREsComputational Statistics 2.4 (2010), pp. 433–459. doi: https://doi.org/10.1002/wics.101. eprint: https://wires.onlinelibrary.wiley.com/doi/pdf/10.1002/wics.101. url: https://wires.onlinelibrary.wiley.com/doi/abs/10.1002/wics.101.
- [2] MdManjurulAhsanetal.“EffectofDataScalingMethodsonMachineLearning Algorithms and Model Performance”. In: Technologies 9.3 (2021). issn: 2227-7080. doi: 10.3390/technologies9030052. url: https://www.mdpi.com/2227-7080/9/3/52.
- [3] Thomas E Booth. “Exact Monte Carlo solution of elliptic partial differential equations”. In: Journal of Computational Physics 39.2 (1981), pp. 396–404. issn: 0021-9991. doi: https : / / doi . org / 10 . 1016 / 0021 - 9991(81)90159- 5. url: https://www.sciencedirect.com/science/article/pii/0021999181901595.
- [4] Lars Buitinck et al. “API design for machine learning software: experiences from the scikit-learn project”. In: ECML PKDD Workshop: Languages for Data Mining and Machine Learning. 2013, pp. 108–122.
- [5] Rudolf N Cardinal and Michael RF Aitken. ANOVA for the behavioral sciences researcher. Psychology Press, 2013.
- [6] Malcolm E Davis and J Andrew McCammon. “Electrostatics in biomolecular structure and dynamics”. In: Chemical Reviews 90.3 (1990), pp. 509–521.
- [7] BS Elepov et al. “Solution of boundary value problems by the Monte Carlo method”. In: Science: Novosibirsk, Russia (1980).
- [8] Marcia O. Fenley et al. “Using correlated Monte Carlo sampling for efficiently solving the linearized Poisson-Boltzmann equation over a broad range of salt concentration”. In: Journal of Chemical Theory and Computation 6 (1 2010). issn: 15499618. doi: 10.1021/ct9003806.
- [9] Charles Fleming, Michael Mascagni, and Nikolai Simonov. “An Efficient Monte Carlo Approach for Solving Linear Problems in Biomolecular Electrostatics”.In: Computational Science – ICCS 2005. Ed. by Vaidy S. Sunderam et al. Berlin, Heidelberg: Springer Berlin Heidelberg, 2005, pp. 760–765. isbn: 978-3-540-32118-7.

References

- [10] F. Fogolari, A. Brigo, and H. Molinari. “The Poisson–Boltzmann equation for biomolecular electrostatics: a tool for structural biology”. In: Journal of Molecular Recognition 15.6 (2002), pp. 377–392. doi: https://doi.org/10.1002/jmr.577. eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/jmr.577. url: https://onlinelibrary.wiley.com/doi/abs/10.1002/jmr.577.
- [11] Federico Fogolari et al. “Biomolecular electrostatics with the linearized Poisson-Boltzmann equation”. In: Biophysical Journal 76 (1 I 1999). issn: 00063495.doi: 10.1016/S0006-3495(99)77173-0.
- [12] James A. Given, Michael Mascagni, and Chi Ok Hwang. “Continuous pathbrownian trajectories for diffusion monte carlo via first- and last-passage distributions”. In: Lecture Notes in Computer Science (including subseries LectureNotes in Artificial Intelligence and Lecture Notes in Bioinformatics). Vol. 2179.2001. doi: 10.1007/3-540-45346-6_4.
- [13] A. Haji-Sheikh and E. M. Sparrow. “The Solution of Heat Conduction Problems by Probability Methods”. In: Journal of Heat Transfer 89.2 (May 1967), pp. 121–130. issn: 0022-1481. doi: 10 . 1115 / 1 . 3614330. eprint: https : // asmedigitalcollection.asme.org/heattransfer/article-pdf/89/2/121/5718589/121_1.pdf. url: https://doi.org/10.1115/1.3614330.
- [14] Preston Hamlin et al. “Geometry entrapment in Walk-on-Subdomains”. In: Monte Carlo Methods and Applications 25 (4 2019). issn: 15693961. doi: 10.1515/mcma-2019-2052.
- [15] Chi-Ok Hwang, Michael Mascagni, and James A. Given. In: Monte Carlo Methods and Applications 7.3-4 (2001), pp. 213–222. doi: doi:10.1515/mcma.2001. 7.3-4.213. url: https://doi.org/10.1515/mcma.2001.7.3-4.213.
- [16] Chi-Ok Hwang, Michael Mascagni, and Taeyoung Won. “Monte Carlo methods for computing the capacitance of the unit cube”. In: Mathematics and Computers in Simulation 80.6 (2010), pp. 1089–1095.
- [17] B. Z. Lu et al. Recent progress in numerical methods for the Poisson-Boltzmann equation in biophysical applications. 2008.
- [18] Travis MacKoy et al. “Numerical optimization of a walk-on-spheres solver for the linear Poisson-Boltzmann equation”. In: Communications in Computational Physics. Vol. 13. 2013. doi: 10.4208/cicp.220711.041011s.

References

- [19] Martín Abadi et al. TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems. Software available from tensorflow.org. 2015. url: https://www.tensorflow.org/.
- [20] Michael Mascagni and Chi Ok Hwang. “ϵ-shell error analysis for "Walk On Spheres" algorithms”. In: Mathematics and Computers in Simulation 63 (22003). issn: 03784754. doi: 10.1016/S0378-4754(03)00038-7.
- [21] Michael Mascagni and Nikolai A. Simonov. “Monte Carlo method for calculating the electrostatic energy of a molecule”. In: Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics) 2657 (2003). issn: 16113349. doi: 10.1007/3-540-44860-8_7.
- [22] Michael Mascagni and Nikolai A. Simonov. “Monte Carlo methods for calculating some physical properties of large molecules”. In: SIAM Journal on Scientific Computing 26 (1 2005). issn: 10648275. doi: 10.1137/S1064827503422221.
- [23] Wes McKinney. “Data Structures for Statistical Computing in Python”. In: Proceedings of the 9th Python in Science Conference.Ed.byStéfanvanderWalt and Jarrod Millman. 2010, pp. 56–61. doi: 10.25080/Majora-92bf1922-00a.
- [24] Mervin E. Müller. “Some Continuous Monte Carlo Methods for the Dirichlet Problem”. In: The Annals of Mathematical Statistics 27 (3 1956). issn: 0003-4851. doi: 10.1214/aoms/1177728169.
- [25] Gábor Náray-Szabó and Arieh Warshel. Computational approaches to biochemical reactivity. Vol. 19. Springer Science & Business Media, 2002.
- [26] Arnab Pal and Shlomi Reuveni. “First Passage under Restart”. In: Phys. Rev. Lett. 118 (3 Jan. 2017), p. 030603. doi: 10.1103/PhysRevLett.118.030603. url: https://link.aps.org/doi/10.1103/PhysRevLett.118.030603.
- [27] Dennis W. Piehl et al. “rcsb-api: Python Toolkit for Streamlining Access to RCSB Protein Data Bank APIs”. In: Journal of Molecular Biology (2025), p. 168970. issn: 0022-2836. doi: https://doi.org/10.1016/j.jmb.2025.168970. url: https : / / www . sciencedirect . com / science / article / pii /S0022283625000361.

References

- [28] AkhlaqurRahmanandSumairaTasnim.“EnsembleClassifiersandTheirApplications: A Review”. In: International Journal of Computer Trends and Technology 10.1 (Apr. 2014), pp. 31–35. issn: 2231-2803. doi: 10.14445/22312803/ijctt - v10p107. url: http : / / dx . doi . org / 10 . 14445 / 22312803 / IJCTT - V10P107.
- [29] Yana Rose et al. “RCSB Protein Data Bank: Architectural Advances Towards Integrated Searching and Efficient Access to Macromolecular Structure Data from the PDB Archive”. In: Journal of Molecular Biology 433.11 (2021). Computation Resources for Molecular Biology, p. 166704. issn: 0022-2836. doi: https : / / doi . org / 10 . 1016 / j . jmb . 2020 . 11 . 003. url: https : / / www .sciencedirect.com/science/article/pii/S0022283620306227.
- [30] Karl K Sabelfeld and Nikolai A Simonov. Stochastic methods for boundary value problems: numerics for high-dimensional PDEs and applications. Walter de Gruyter GmbH & Co KG, 2016.
- [31] Karl Karlovich Sabelfeld. “Monte Carlo methods in boundary value problems”. In: (No Title) (1991).
- [32] Nikolai A Simonov. “A random walk algorithm for the solution of boundary value problems with partition into subdomains”. In: Methods and algorithms for statistical modelling. Akad. Nauk SSSR Sibirsk. Otdel., Vychisl. Tsentr, Novosibirsk (1983), pp. 48–58.
- [33] The pandas development team.pandas-dev/pandas: Pandas.Versionlatest.Feb.2020. doi: 10 . 5281 / zenodo .3509134. url: https : / / doi . org / 10 . 5281 /zenodo.3509134.
- [34] W. John Thrasher and Michael Mascagni. “Examining sharp restart in a Monte Carlo method for the linearized Poisson–Boltzmann equation”. In: Monte Carlo Methods and Applications 26 (3 2020). issn: 15693961. doi: 10.1515/mcma2020-2069.