Monte Carlo Methods: Early History and The Basics

Prof. Michael Mascagni

Department of Computer Science
Department of Mathematics
Department of Scientific Computing
Graduate Program in Molecular Biophysics
Florida State University, Tallahassee, FL 32306 **USA**
AND
Applied and Computational Mathematics Division, ITL
National Institute of Standards and Technology, Gaithersburg, MD 20899-8910 **USA**
E-mail: mascagni@fsu.edu or mascagni@math.ethz.ch
or mascagni@nist.gov
URL: http://www.cs.fsu.edu/~mascagni

Research supported by ARO, DOE, NASA, NATO, NIST, and NSF
with equipment donated by Intel and Nvidia

HPCS Tutorial: July 21, 2014

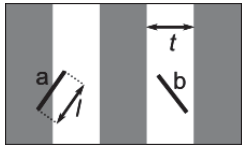
Outline of the Talk

Early History of Probability Theory

- Probability was first used to understand games of chance
  1. Antoine Gombaud, chevalier de Méré, a French nobleman called on Blaise Pascal and Pierre de Fermat were called on to resolve a dispute
  2. Correspondence between Pascal and Fermat led to Huygens writing a text on "Probability"
  3. Jacob Bernoulli, Abraham de Moivre, and Pierre-Simon, marquis de Laplace, led development of modern "Probability"
  4. 1812: Laplace, *Théorie Analytique des Probabilités*

Early History of Monte Carlo: Before Los Alamos

- Buffon Needle Problem: Early Monte Carlo (experimental mathematics)



  1. Problem was first stated in 1777 by Georges-Louis Leclerc, comte de Buffon
  2. Involves dropping a needle on a lined surface and can be used to estimate π
  3. Note: Union Capt. Fox did this while in a CSA prison camp, and produced good results that later turned out to be "fudged"
- In the 1930's, Fermi used sampling methods to estimate quantities involved in controlled fission

The Stars Align at Los Alamos

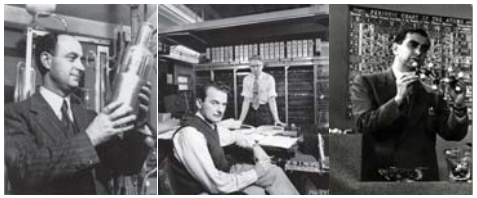
- Los Alamos brought together many interesting factors to give birth to modern Monte Carlo algorithms
  1. The Problems: Simulation of neutron histories (neutronics), hydrodynamics, thermonuclear detonation
  2. The People: Enrico Fermi, Stan Ulam, John von Neumann, Nick Metropolis, Edward Teller, ...
  3. The Technology: Massive human computers using hand calculators, the Fermiac, access to early digital computers
- The Name: Ulam's uncle would borrow money from the family by saying that "I just have to go to Monte Carlo"

The Problems

- Simulation of neutron histories (neutronics)
  1. Given neutron positions/momenta, geometry
  2. Compute flux, criticality, fission yield
- Hydrodynamics due to nuclear implosion
- Simulation of thermonuclear reactions: ignition, overall yield
  1. All these problems were more easily solved using Monte Carlo/Lagrangian methods
  2. Geometry is problematic for deterministic methods but not for MC

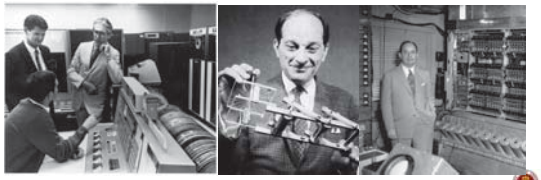
The People

- Los Alamos brought together many interesting people to work on the fission problem:
- The Physicists
  1. Enrico Fermi: experimental Nuclear Physics and computational approaches
  2. Nick Metropolis: one of the first “computer programmers” for these problems
  3. Edward Teller: more interested in the “super”



The People

- The Mathematicians
  1. Robert Richtmyer: ran the numerical analysis activities at Los Alamos
  2. Stanislaw (Stan) Ulam: became interested in using “statistical sampling” for many problems
  3. John von Neumann: devised Monte Carlo algorithms and helped develop digital computers



The Technology

- Simulation via computation was necessary to make progress at Los Alamos
- Many different computational techniques were in used
  1. Traditional digital computation: hand calculators used by efficient technicians
  2. Analog computers including the Fermiac (picture to follow)
  3. Shortly after the war, access to digital computers: ENIAC at Penn/Army Ballistics Research Laboratory (BRL)
  4. Continued development and acquisition of digital computers by Metropolis including the MANIAC

An Analog Monte Carlo Computer: The Fermiac

- Neutronics required simulating exponentially distributed flights based on material cross-sections
- Many neutron histories are required to get statistics
- Fermiac allows simulation of exponential flights inputting the cross-section manually
- Fermiac is used on a large piece of paper with the geometry drawn for neutronics simulations
- Fermiac allows an efficient graphical simulation of neutronics
- Parallelism is achievable with the Fermiac

An Analog Monte Carlo Computer: The Fermiac

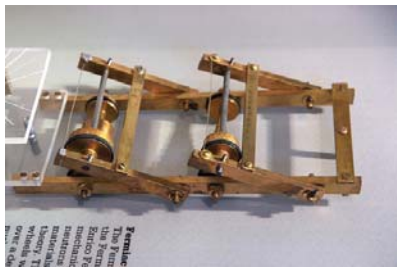


Figure: Enrico Fermi’s Fermiac at the Boston Computer Museum

An Analog Monte Carlo Computer: The Fermiac

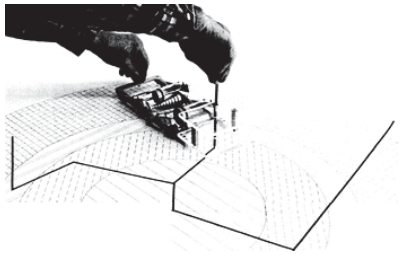


Figure: The Fermiac in Action

An Early Digital Computer: The ENIAC

- ENIAC: Electronic Numerical Integrator And Computer
- Funded by US Army with contract signed on June 5, 1943
- Built in secret by the University of Pennsylvania's Moore School of Electrical Engineering
- Completed February 14, 1946 in Philadelphia and used until November 9, 1946
- Moved (with upgrade) to Aberdeen Proving Grounds and began operations July 29, 1947
- Remained in continuous operation at the Army BRL until 1955

An Early Digital Computer: The ENIAC

- ENIAC is a completely programmable computer using first a plug panel
- ENIAC first contained (military rejects!)
  1. 17,468 vacuum tubes
  2. 7,200 crystal diodes
  3. 1,500 relays, 70,000 resistors
  4. 10,000 capacitors
  5. about 5 million hand-soldered joints
- Clock was 5KHz
- Ended up with a 100-word core memory
- Metropolis would go to BRL to work on the "Los Alamos" problem on the ENIAC

An Early Digital Computer: The ENIAC

Figure: The ENIAC at the University of Pennsylvania

An Early Digital Computer: The ENIAC

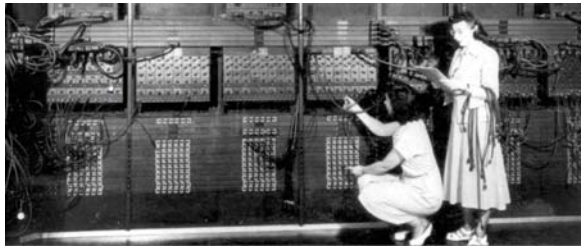


Figure: Programming the ENIAC

An Early Digital Computer: The ENIAC

Figure: Tubes from the ENIAC

The Birth of Monte Carlo Methods

- After the was digital computer was perfect for "statistical sampling"
  1. Individual samples were often very simple to program
  2. Small memory was not a big constraint for these methods
  3. A much better use for digital vs. human computers
- Early Monte Carlo Meetings
  1. 1952, Los Angeles: RAND Corp., National Bureau of Standards (NBS now NIST), Oak Ridge
  2. 1954, Gainesville, FL: University of Florida Statistical Lab

Integration: The Classic Monte Carlo Application

1. Consider computing $I = \int_0^1 f(x)\,dx$
2. Conventional quadrature methods:
$$I \approx \sum_{i=1}^{N} w_i f(x_i)$$
   - *Rectangle*: $w_i = \frac{1}{N}, x_i = \frac{i}{N}$
   - *Trapezoidal*: $w_i = \frac{2}{N}, w_1 = w_N = \frac{1}{N}, x_i = \frac{i}{N}$
3. Monte Carlo method has two parts to estimate a numerical quantity of interest, I
   - The random process/variable: $x_i \sim U[0,1]$ i.i.d.
   - The score: $f(x_i)$
   - One averages and uses a confidence interval for an error bound

$$\bar{I} = \frac{1}{N}\sum_{i=1}^{N} f(x_i), \quad var(I) = \frac{1}{N-1}\sum_{i=1}^{N}(f(x_i)-\bar{I})^2 = \frac{1}{N-1}\left[\sum_{i=1}^{N} f(x_i)^2 - N\bar{I}^2\right],$$

$$var(\bar{I}) = \frac{var(I)}{N}, \quad I \in \bar{I} \pm k \times \sqrt{var(\bar{I})}$$

Other Early Monte Carlo Applications

- Numerical linear algebra based on sums: $S = \sum_{i=1}^{N} a_i$
  1. Define $p_i \geq 0$ as the probability of choosing index i, with $\sum_{i=1}^{M} p_i = 1$, and $p_i > 0$ whenever $a_i \neq 0$
  2. Then a_i/p_i with index i chosen with $\{p_i\}$ is an unbiased estimate of S, as $E[a_i/p_i] = \sum_{i=1}^{M}\left(\frac{a_i}{p_i}\right)p_i = S$
- Can be used to solve linear systems of the form $x = Hx + b$
- Consider the linear system: $x = Hx + b$, if $||H|| = \mathbb{H} < 1$, then the following iterative method converges:
$$x^{n+1} := Hx^n + b, \quad x^0 = 0,$$
and in particular we have $x^k = \sum_{i=0}^{k-1} H^i b$, and similarly the Neumann series converges:
$$N = \sum_{i=0}^{\infty} H^i = (I-H)^{-1}, \quad ||N|| = \sum_{i=0}^{\infty}||H^i|| \leq \sum_{i=0}^{\infty}\mathbb{H}^i = \frac{1}{1-\mathbb{H}}$$
- Formally, the solution is $x = (I-H)^{-1}b$

Other Early Monte Carlo Applications

- Methods for partial differential and integral equations
  1. Integral equation methods are similar in construction to the linear system methods
  2. PDEs can be solved by using the Feynman-Kac formula
  3. Note Kac and Ulam both were trained in Lwów

Monte Carlo Methods: Numerical Experimental that Use Random Numbers

- A Monte Carlo method is any process that consumes random numbers

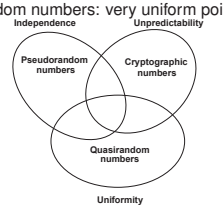
1. Each calculation is a numerical experiment
   - Subject to known and unknown sources of error
   - Should be reproducible by peers
   - Should be easy to run anew with results that can be combined to reduce the variance
2. Sources of errors must be controllable/isolatable
   - Programming/science errors under your control
   - Make possible RNG errors approachable
3. Reproducibility
   - Must be able to rerun a calculation with the same numbers
   - Across different machines (modulo arithmetic issues)
   - Parallel and distributed computers?

Early Random Number Generators on Digital Computers

- Middle-Square method: von Neumann
  1. 10 digit numbers: $x_{n+1} = \lfloor \frac{x_n^2}{10^5} \rfloor \pmod{10^{10}}$
  2. Multiplication leads to good mixing
  3. Zeros in lead to short periods and cycle collapse
- Linear congruential method: D. H. Lehmer
- $x_{n+1} = ax_n + c \pmod{m}$
- Good properties with good parameters
- Has become very popular

What are Random Numbers Used For?

- There are many types of random numbers
  1. "Real" random numbers: a mathematical idealization
  2. Random numbers based on a "physical source" of randomness
  3. Computational Random numbers
  1. Pseudorandom numbers: deterministic sequence that passes tests of randomness
  2. Cryptographic numbers: totally unpredictable
  3. Quasirandom numbers: very uniform points



Future Work on Random Numbers

Please come to my Friday Plenary on these topics!!

1. Support for new architectures
   - Multicore processors: OpenMP-SPRNG
   - GPGPU support: LCGs and FLGs
2. Testing Random Numbers
   - Hardware random numbers
   - Cryptographic test suites
   - Support for new RNG suites
3. Applications
   - Monte Carlo on new architectures
   - Reproducibility and system integrity
   - Computational resilience and fault tolerance
4. Commercialization of SPRNG

Please come to my Friday Plenary on these topics!!

References

[M. Mascagni, T. Anderson, H. Yu and Y. Qiu (2014)]
Papers on SPRNG generators for Multicore and GPGPU
One submitted and three in preparation

[M. Mascagni and H. Chi (2004)]
Parallel Linear Congruential Generators with Sophie-Germain Moduli,
Parallel Computing, **30**: 1217–1231.

[M. Mascagni and A. Srinivasan (2004)]
Parameterizing Parallel Multiplicative Lagged-Fibonacci Generators,
Parallel Computing, **30**: 899–916.

[M. Mascagni and A. Srinivasan (2000)]
Algorithm 806: SPRNG: A Scalable Library for Pseudorandom Number Generation,
ACM Transactions on Mathematical Software, **26**: 436–461.

Questions?

Monte Carlo Methods and Partial Differential Equations: Algorithms and Implications for High-Performance Computing

Prof. Michael Mascagni

Department of Computer Science
Department of Mathematics
Department of Scientific Computing
Graduate Program in Molecular Biophysics
Florida State University, Tallahassee, FL 32306 **USA**
AND
Applied and Computational Mathematics Division, ITL
National Institute of Standards and Technology, Gaithersburg, MD 20899-8910 **USA**
E-mail: mascagni@fsu.edu or mascagni@math.ethz.ch
or mascagni@nist.gov
URL: http://www.cs.fsu.edu/~mascagni

Research supported by ARO, DOE, NASA, NATO, NIST, and NSF
with equipment donated by Intel and Nvidia

HPCS Tutorial: July 21, 2014

Outline of the Talk

Early History of MCMs for PDEs

1. Courant, Friedrichs, and Lewy: Their pivotal 1928 paper has probabilistic interpretations and MC algorithms for linear elliptic and parabolic problems
2. Fermi/Ulam/von Neumann: Atomic bomb calculations were done using Monte Carlo methods for neutron transport, their success inspired much post-War work especially in nuclear reactor design
3. Kac and Donsker: Used large deviation calculations to estimate eigenvalues of a linear Schrödinger equation
4. Forsythe and Leibler: Derived a MCM for solving special linear systems related to discrete elliptic PDE problems

Integration: The Classic Monte Carlo Application

1. Consider computing $I = \int_0^1 f(x)\,dx$
2. Conventional quadrature methods:
$$I \approx \sum_{i=1}^{N} w_i f(x_i)$$
   - Standard quadrature is of this form with deterministic error bounds
   - If we hold work, $f(x_i)$, constant as dimension increases we see the MC advantage vs. the curse of dimensionality
3. Monte Carlo method has two parts to estimate a numerical quantity of interest, I
   - The random process/variable: $x_i \sim U[0,1]$ i.i.d.
   - The estimator or score: $f(x_i)$
   - One averages and uses a confidence interval for an error bound

$$\bar{I} = \frac{1}{N}\sum_{i=1}^{N} f(x_i), \quad var(I) = \frac{1}{N-1}\sum_{i=1}^{N}(f(x_i)-\bar{I})^2 = \frac{1}{N-1}\left[\sum_{i=1}^{N} f(x_i)^2 - N\bar{I}^2\right],$$

$$var(\bar{I}) = \frac{var(I)}{N}, \quad I \in \bar{I} \pm k \times \sqrt{var(\bar{I})}$$

Other Early Monte Carlo Applications

- Numerical linear algebra based on sums: $S = \sum_{i=1}^{M} a_i$
  1. Define $p_i \geq 0$ as the probability of choosing index i, with $\sum_{i=1}^{M} p_i = 1$, and $p_i > 0$ whenever $a_i \neq 0$
  2. Then a_i/p_i with index i chosen with $\{p_i\}$ is an unbiased estimate of S, as $E[a_i/p_i] = \sum_{i=1}^{M}\left(\frac{a_i}{p_i}\right) p_i = S$
- Can be used to solve linear systems of the form $x = Hx + b$
- Consider the linear system: $x = Hx + b$, if $||H|| = \mathbb{H} < 1$, then the following iterative method converges:
$$x^{n+1} := Hx^n + b, \quad x^0 = 0,$$
  and in particular we have $x^k = \sum_{i=0}^{k-1} H^i b$, and similarly the Neumann series converges:
$$N = \sum_{i=0}^{\infty} H^i = (I-H)^{-1}, \quad ||N|| = \sum_{i=0}^{\infty} ||H^i|| \leq \sum_{i=0}^{\infty} \mathbb{H}^i = \frac{1}{1-\mathbb{H}}$$
- Formally, the solution is $x = (I-H)^{-1}b$

More Modern Monte Carlo Applications

- Methods for partial differential and integral equations based on random walks/Markov chains (no need to find a discrete approximation to the PDE/IE)
  1. Integral equation methods are similar in construction to the linear system methods
  2. PDEs can be solved by using the Feynman-Kac formula
  3. Some Monte Carlo methods can now beat deterministic solvers (electrostatics)
- Efficient methods that exploit fast probabilistic application of a linear operator
- Modern sampling methods linear algebra (SVD) based loosely on the Johnson-Lindestrauss projection method
- Generation of random fields
- Stochastic DEs and PDEs
- Financial computing
- Uncertainty quantification (UQ)

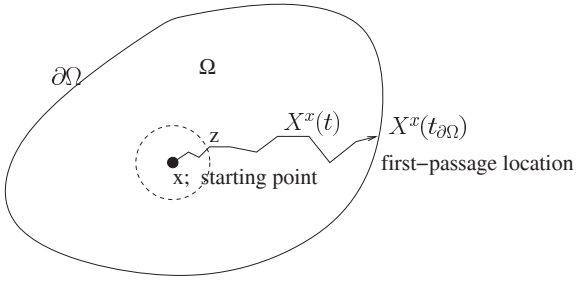
The First Passage (FP) Probability is the Green's Function

Back to our canonical elliptic boundary value problem:

$$\frac{1}{2}\Delta u(x) = 0, \quad x \in \Omega$$
$$u(x) = f(x), \quad x \in \partial\Omega$$

- Distribution of z is uniform on the sphere
- Mean of the values of $u(z)$ over the sphere is $u(x)$
- $u(x)$ has mean-value property and harmonic
- Also, $u(x)$ satisfies the boundary condition
$$u(x) = \mathbb{E}_x[f(X^{\mathbf{x}}(t_{\partial\Omega}))] \qquad (1)$$

The First Passage (FP) Probability is the Green's Function



The First Passage (FP) Probability is the Green's Function

Reinterpreting as an average of the boundary values

$$u(x) = \int_{\partial\Omega} p(x,y)\, f(y)\, dy \tag{2}$$

Another representation in terms of an integral over the boundary

$$u(x) = \int_{\partial\Omega} \frac{\partial g(x,y)}{\partial \mathbf{n}}\, f(y)\, dy \tag{3}$$

$g(x,y)$ – Green's function of the Dirichlet problem in Ω

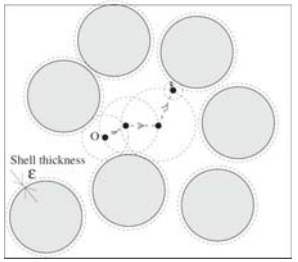
$$\Longrightarrow p(x,y) = \frac{\partial g(x,y)}{\partial \mathbf{n}} \tag{4}$$

'Walk on Spheres' (WOS) and 'Green's Function First Passage' (GFFP) Algorithms

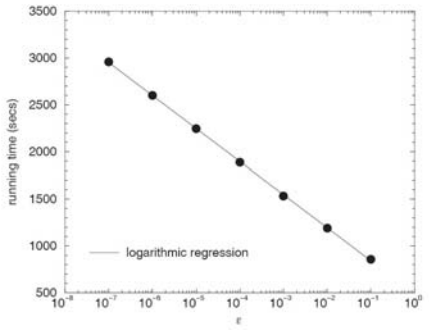
- Green's function is known
  $\Longrightarrow$ direct simulation of exit points and computation of the solution through averaging boundary values
- Green's function is unknown
  $\Longrightarrow$ simulation of exit points from standard subdomains of Ω, e.g. spheres
  $\Longrightarrow$ Markov chain of 'Walk on Spheres' (or GFFP algorithm)
  $x_0 = x, x_1, \ldots, x_N$
  $x_i \to \partial\Omega$ and hits ε-shell is $N = O(|\ln(\varepsilon)|)$ steps
  x_N simulates exit point from Ω with $O(\varepsilon)$ accuracy

'Walk on Spheres' (WOS) and 'Green's Function First Passage' (GFFP) Algorithms

WOS:



Timing with WOS



Parallelization of the Monte Carlo Method

- These Monte Carlo methods are naturally parallel, and have many possible sources of independent parallel work due to their sampling nature
- Parallelization based on processing different samples that can almost always be executed without decomposition and hence communication
- In integration can parallelize based on
  1. Sample numbers (with different RNG streams)
  2. Domain decomposition
  3. Can have adaptivity with only the cost of some initial variance estimation
- Only the final sample (1 integer, 2 reals) needs to be asynchronously communicated to compute the overall mean and variance, very cheap application-level checkpointing

Memory and Communication

- Often the Monte Carlo method deals with the geometry without discretization, much less memory is needed to represent the entire problem
- Mean and variance are computed by calculating a running (1) sum, (2) sum of squares, and (3) samples
- Independent sampling means that one can do AS MUCH computation per core as you wish before even these three values need be communicated (tuning the level of compute-boundedness)
- It's even OK with adaptivity
  1. Initial variance estimate to guess at N given tolerance, ϵ
  2. The N samples can be computed with a static or dynamic parallel work allocation

Architectural Considerations

- Some trends in HPC architectures
  1. Memory per processor/core has inflected and is now decreasing
  2. Long-term trend is that memory bandwidth is the limiting factor for performance and cost
  3. High clock rates and high bandwidth communication lead to high energy consumption and hot boxes that need cooling
- These Monte Carlo algorithms avoid all three of issues due to their innate performance
  1. Minimal memory usage has always been a benefit of Monte Carlo methods
  2. Independent sampling means that the communication to computation ratio is extremely small and tunable
- Monte Carlo is a very simple computational paradigm to explore fundamental aspects of parallelism, algorithmic resilience, fault-tolerance

All This Depends on High-Quality Pseudorandom Number Generators

- The ability of a Monte Carlo method to work depends on the quality random numbers used
- In a serial application, this is essentially the ability of a pseudorandom number generator to pass an extensive suite of test of randomness (mostly statistical)
- For good parallel performance, the streams used in each independent realization must lead to qualitatively independent sampling
  1. Must be free if intra- and inter-stream correlations
  2. Must be able to supply potentially very long computations
- There are very few packages available that even attempt to provide this functionality
  1. Scalable Parallel Random Number Generators (`SPRNG`) library
  2. TINA Is No Acronym (`TINA`)
  3. `RNGStream`
  4. `Random123`
- Must give up absolute reproducibility and embrace "forensic reproducibility"

Porous Media: Complicated Interfaces

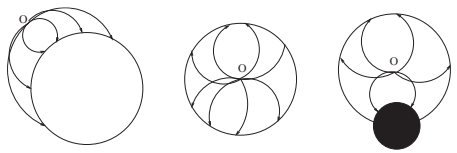
Computing Capacitance Probabilistically

- Hubbard-Douglas: can compute permeability of nonskew object via capacitance
- Recall that $C = \frac{Q}{u}$, if we hold conductor (Ω)at unit potential $u = 1$, then $C =$ total charge on conductor (surface)
- The PDE system for the potential is

$$\Delta u = 0, \quad x \notin \Omega; \quad u = 1, \quad x \in \partial\Omega; \quad u \to 0 \text{ as } x \to \infty \tag{5}$$

- Recall $u(\mathbf{x}) = \mathbb{E}_x[f(X^{\mathbf{x}}(t_{\partial\Omega}))] =$ probability of walker starting at $\mathbf{x}$ hitting Ω before escaping to infinity
- Charge density is first passage probability
- Capacitance (relative to a sphere) is probability of walker starting at $\mathbf{x}$ (random chosen on sphere) hitting Ω before escaping to infinity

Various Laplacian Green's Functions for Green's Function First Passage (GFFP)



(a) Putting back (b) Void space (c) Intersecting

Escape to ∞ in A Single Step

- Probability that a diffusing particle at $r_0 > b$ will escape to infinity

$$P_{esc} = 1 - \frac{b}{r_0} = 1 - \alpha \tag{6}$$

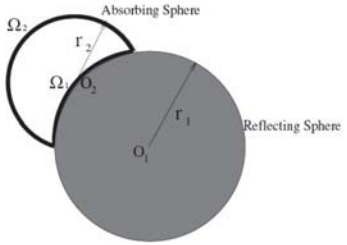
- Putting-back distribution density function

$$\omega(\theta, \phi) = \frac{1 - \alpha^2}{4\pi[1 - 2\alpha\cos\theta + \alpha^2]^{3/2}} \tag{7}$$

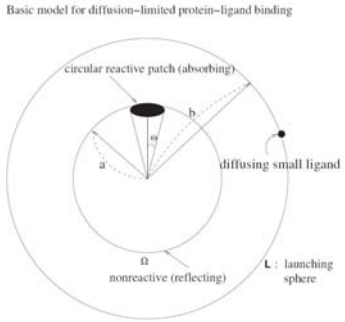
  - (b, θ, ϕ) ; spherical coordinates of the new position when the old position is put on the polar axis

The Simulation-Tabulation (S-T) Method for Generalization

- Green's function for the non-intersected surface of a sphere located on the surface of a reflecting sphere

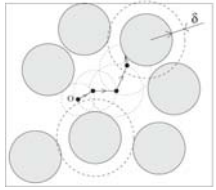


Solc-Stockmayer Model without Potential

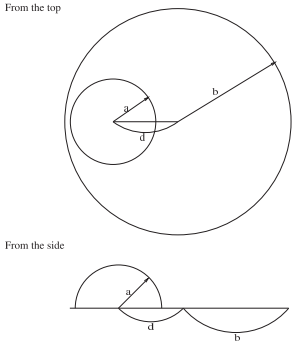


Another S-T Application: Mean Trapping Rate

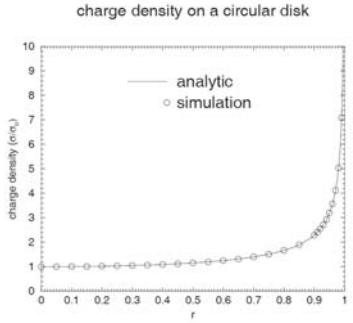
In a domain of nonoverlapping spherical traps :



Charge Density on a Circular Disk via Last-Passage



Charge Density on the Circular Disk



Unit Cube Edge Distribution

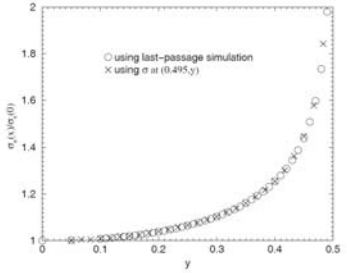


Figure: First- and last-passage edge computations

Walk on the Boundary Algorithm

- $\mu(y) = -\frac{1}{4\pi}\frac{\partial \phi}{\partial n}(y)$; surface charge density
- $\phi(x) = \int_{\partial\Omega} \frac{1}{|x-y|}\mu(y)d\sigma(y)$; electrostatic potential

Limit properties of the normal derivative ($x \to y$ outside of Ω):

$$\mu(y) = \int_{\partial\Omega} \frac{n(y)\cdot(y-y')}{2\pi|y-y'|^3}\mu(y')d\sigma(y')$$

By the ergodic theorem (convex Ω)

$$\int_{\partial\Omega} v(y)\pi_\infty(y)d\sigma(y) = \lim_{N\to\infty}\frac{1}{N}\sum_{n=1}^{N} v(y_n)$$

Walk on the Boundary Algorithm

- π_∞ - stationary distribution of Markov chain $\{y_n\}$ with transition density $p(y_n \to y_{n+1}) = \frac{n(y_{n+1})\cdot(y_{n+1}-y_n)}{2\pi|y_{n+1}-y_n|^3}$
- $\mu = C\pi_\infty$
- C - capacitance if $\phi|_{\partial\Omega} = 1$
- $\phi(x) = 1$ for $x \in \Omega$

$$C = (\lim_{N\to\infty}\frac{1}{N}\sum_{n=1}^{N} v(y_n))^{-1} \quad \text{for} \quad v(y) = \frac{1}{x-y}$$

Capacitance of the Unit Cube

Reitan-Higgins (1951)	0.6555
Greenspan-Silverman (1965)	0.661
Cochran (1967)	0.6596
Goto-Shi-Yoshida (1992)	$0.6615897 \pm 5 \times 10^{-7}$
Conjectured Hubbard-Douglas (1993)	0.65946...
Douglas-Zhou-Hubbard (1994)	0.6632 ± 0.0003
Given-Hubbard-Douglas (1997)	0.660675 ± 0.00001
Read (1997)	0.6606785 ± 0.000003
First passage method (2001)	0.660683 ± 0.000005
Walk on boundary algorithm (2002)	0.6606780 ± 0.0000004

Continuum Biochemical Electrostatics

Motivation

- Experimental Data: Folding, stability & binding behavior of biomolecules can be modulated by changes in salt concentration
- Physical Model: Implicit solvent-based Poisson-Boltzmann model can provide accurate predictions of salt dependent behavior of biomolecules
- Mathematical Model: Elliptic boundary-value problems

Specific Problems

- Electrostatic free energy for linear case: only finite number of electrostatic potential point values
- Dependence of energy on geometry: needs accurate treatment
- Singularities in solution: have to be taken into account analytically
- Behavior at infinity: must be exactly enforced
- Functional dependence on salt concentration: needs accurate estimate

Mathematical Model: Molecular Geometry

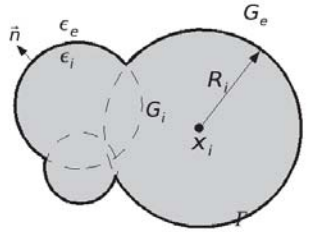


Figure: Biomolecule with dielectric ϵ_i and region region G_i is in solution with dielectric ϵ_e and region G_e. On the boundary of the biomolecule, electrostatic potential and normal component of dielectric displacement continue

Mathematical Model: Partial Differential Equations

- Poisson equation for the electrostatic potential, Φ_i, and point charges, Q_m, inside a molecule (in CGS units):

$$\epsilon_i\Delta\Phi_i(x) + 4\pi\sum_{m=1}^{M} Q_m\delta(x-x^{(m)}) = 0\ ,\ x \in G_i$$

- For 1-1 salt (such as *NaCl*) Poisson-Boltzmann equation (PBE):

$$\Delta\Phi_e(x) - \kappa^2\sinh(\Phi_e(x)) = 0\ ,\ x \in G_e\ ,$$

but we only consider the linearized PBE:

$$\Delta\Phi_e(x) - \kappa^2\Phi_e(x) = 0\ ,\ x \in G_e$$

- For one-surface model: continuity condition on the dielectric boundary

$$\Phi_i = \Phi_e\ ,\ \epsilon_i\frac{\partial\Phi_i}{\partial n(y)} = \epsilon_e\frac{\partial\Phi_e}{\partial n(y)}\ ,\ y \in \Gamma$$

Electrostatic Potential and Energy

- Point values of the potential: $\Phi(x) = \Phi_{rf}(x) + \Phi^c(x)$
  Here, singular part of Φ:

$$\Phi^c(x) = \sum_{m=1}^{M} \frac{Q_m}{|x - x^{(m)}|}$$

- Reaction field electrostatic free energy of a molecule is linear combination of point values of the regular part of the electrostatic potential:

$$W_{rf} = \frac{1}{2} \sum_{m=1}^{M} \Phi_{rf}(x^{(m)}) Q_m \,,$$

- Electrostatic solvation free energy = difference between the energy for a molecule in solvent with a given salt concentration and the energy for the same molecule in vacuum:

$$\Delta G^{elec}_{solv} = W_{rf}(\epsilon_i, \epsilon_e, \kappa) - W_{rf}(\epsilon_i, 1, 0)$$

The Feynman-Kac Formula

- If we set $f(x) = 0$ and have $g(x) \neq 0$, the solution is

$$u(y) = \mathbb{E}\left[\int_0^{\tau_{\partial\Omega}} g(\beta_y(s))\, ds \right]$$

- By linear superposition, the solution to Poisson equation is given probabilistically as

$$u(y) = \mathbb{E}\left[\int_0^{\tau_{\partial\Omega}} g(\beta_y(s))\, ds + f(\beta_y(\tau_{\partial\Omega})) \right]$$

- The linearized Poisson-Boltzmann equation is given by

$$\Delta u(x) - \kappa^2 u(x) = 0, \;\; x \in \Omega, \;\; u(x) = f(x), \;\; x \in \partial\Omega, \;\; u \to 0 \text{ as } |x| \to \infty$$

and has Wiener integral representation:

$$u(y) = \mathbb{E}\left[f(\beta_y(\tau_{\partial\Omega})) e^{-\int_0^{\tau_{\partial\Omega}} \kappa^2\, ds} \right]$$

'Walk-on-Spheres' Algorithm

- Walk-on-spheres (WOS) algorithm for general domains with a regular boundary
- Define a Markov chain $\{x_i, \; i = 1, 2, \ldots\}$
- Set $x_0 = x^{(m)}$ for some m, $x_i = x_{i-1} + d_i \omega_i \,, \quad i = 1, 2, \ldots$, where
  1. $d_i = d(x_{i-1})$ is distance from x_{i-1} to Γ
  2. $\{\omega_i\}$ is sequence of independent unit isotropic vectors
  3. x_i is the exit point from the ball, $B(x_{i-1}, d(x_{i-1}))$, for a Brownian motion starting at x_{i-1}
- Outside the molecule, on every step, walk-on-spheres terminates with probability $1 - q(\kappa, d_i)$, where $q(\kappa, d_i) = \dfrac{\kappa d_i}{\sinh(\kappa d_i)}$ to deal with LPBE

'Walk-on-Spheres' and 'Walk-in-Subdomains'

- For general domains, an efficient way to simulate exit points is a combination of
  1. Inside the molecule: 'walk-in-subdomains'
  2. Outside the molecule 'walk-on-spheres'
- The whole domain, G_i, is represented as a union of intersecting subdomains:

$$G_i = \bigcup_{m=1}^{M} G^m$$

- 'Walk-in-Subdomains': Simulate exit point separately in every G^m
  1. $x_0 = x, x_1, \ldots, x_N$ – Markov chain, every x_{i+1} is an exit point from the corresponding subdomain for Brownian motion starting at x_i
  2. For spherical subdomains, $B(x_i^m, R_i^m)$, exit points are distributed in accordance with the Poisson kernel:

$$\frac{1}{4\pi R_i^m} \frac{|x_i - x_i^m|^2 - (R_i^m)^2}{|x_i - x_{i+1}|^3}$$

Monte Carlo Estimates

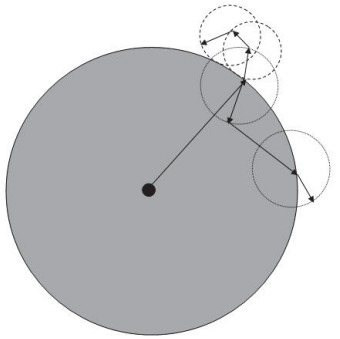
- The estimate for the reaction-field potential point value:

$$\begin{aligned} \xi[\Phi_{rf}](x^{(m)}) &= -\Phi^c(x_1^*) \\ &+ \sum_{j=2}^{N_{ins}} F_j(\kappa) \left(\Phi^c(x_j^{ins}) - \Phi^c(x_{j,ins}^*) \right) \end{aligned} \tag{8}$$

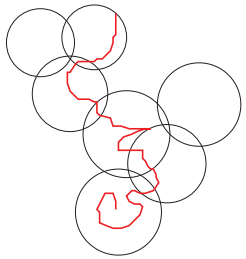
- Here $\{x_{j,ins}^*\}$ is a sequence of boundary points, after which the random walker moves inside the domain, G_i, to x_j^{ins}
- The estimate for the reaction-field energy:

$$\xi[W_{rf}] = \frac{1}{2} \sum_{m=1}^{M} Q_m \, \xi[\Phi_{rf}](x^{(m)}) \tag{9}$$

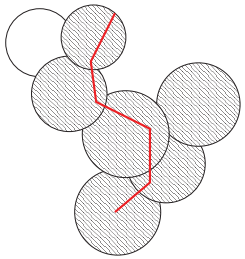
A Picture: The Algorithm for a Single Spherical Atom



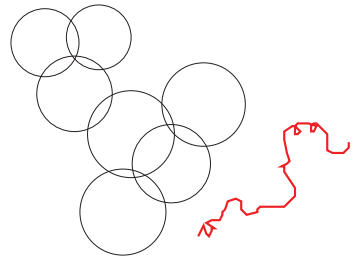
The Algorithm in Pictures: Walk Inside



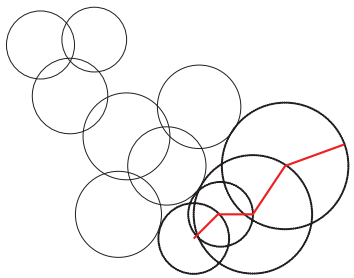
The Algorithm in Pictures: Walk Inside



The Algorithm in Pictures: Walk Outside



The Algorithm in Pictures: Walk Outside



Monte Carlo Algorithm's Computational Complexity

Cost of a single trajectory

- Number of steps is random walk is not dependent on M, the number of atoms
- The cost of finding the nearest sphere is $M \log_2(M)$ due to optimizations

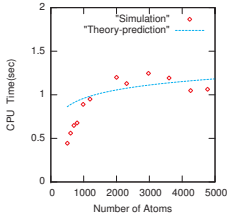


Figure: The CPU time per atom per trajectory is plotted as function of number of atoms. For small number of atoms the CPU time scales linearly and for large number of atoms it asymptotically scales logarithmically

Geometry: Problem Descriptions

There are many geometric problems that arise in this algorithm:

- Efficiently determining if a point is on the surface of the molecule or inside of it (for interior walks)
- Efficiently determining the closest sphere to a given exterior point (for walks outside molecule)
- Efficiently determining if a query point is inside of the convex hull of the molecule
- Efficiently finding the largest possible sphere enclosing a query point for external walks

Correlated and Uncorrelated Sampling

- Correlated sampling in Monte Carlo is essential for two important reasons
  1. To obtain smooth curves with a minimum of sampling (function-wise vs. point-wise sampling)
  2. To obtain accurate results from quantities defined as the differences of Monte Carlo estimates
- With this correlated sampling sampling you can get a “smooth curve" with three orders of magnitude less sampling, note: you still have $O(N^{-1/2})$ errors, just in “curve space," not point by point

Correlated Sampling: Salt Concentration

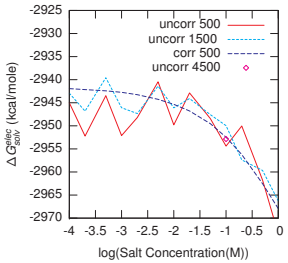


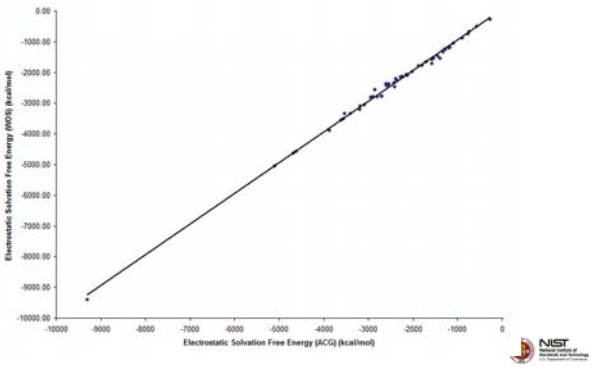
Figure: Electrostatic Solvation free Energy of `3icb` calculated with three four conditions: uncorrelated sampling with 500 number of trajectories per concentration, uncorrelated sampling with 1500 number of trajectories per concentration, uncorrelated sampling with 4500 number of iterations, and correlated sampling with 500 number of trajectories

Dependence on Salt Concentration

- Values of scalar energies as a function of external salt concentration are important
  1. Smooth curves of internal energy vs. salt concentration (see above)
  2. Numerical estimate of the derivative as salt concentration vanishes
- For κ used in simulations, $F_j(\kappa) = 1$
- For an arbitrary $\kappa' > \kappa$:

  $F_j(\kappa')$ is multiplied by the ratio $\frac{q(\kappa', d)}{q(\kappa, d)}$ on every step of the WOS in the exterior
- The results obtained with the estimates (8) and (9) for different values of κ are **highly correlated**

Accuracy: Monte Carlo vs. Deterministic



Sampling Error and Bias

- In Monte Carlo there are biases (errors) and sampling error
  1. Sampling error is based on standard error $O(N^{-1/2})$
  2. Difference between expected value and PDE solution is bias
     - Capture thickness (ϵ): bias is $O(\epsilon)$
     - Auxiliary sphere radius (a): bias is $O(a^3)$
     - Effective Van der Waals sphere radius, R
     - Overall bias: $\left(\frac{a}{2R}\right)^3 + \left(\frac{\epsilon}{2R}\right)$
  3. $Var[\sum_i q_i \Phi(x_i)] = \sum_i q_i^2 Var[\Phi(x_i)]$
  4. Given a desired variance, divide it evenly over this sum
  5. Running time $\propto \frac{|\ln(\epsilon)|}{a}$
  6. Can reduce running time by 2 orders of magnitude by bias/variance balancing and using larger ϵ, a and ANN
  7. Large ANN means errors in drawing the largest sphere outside the molecule for WOS

Timing: Better Than Expected

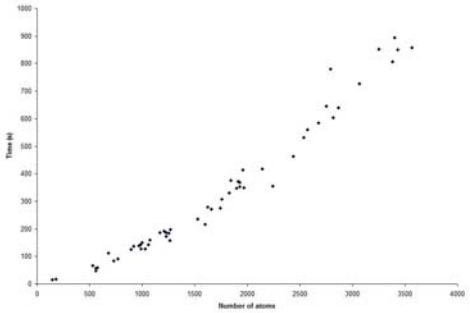


Figure: $O(M \log M)$?

Conclusions

- Over the years we have developed many MC tools for PDEs and more recently:
- We have developed a novel stochastic linear PBE solver that can provide highly accurate salt-dependent electrostatic properties of biomolecules in a single PBE calculation
- Advantages of the stochastic linear PBE solver over the more mature deterministic methods include: the subtle geometric features of the biomolecule can be treated with higher precision, the continuity and outer boundary conditions are accounted for exactly, a singularity free scheme is employed and straightforward implementation on parallel computer platform is possible
- Codes provide higher accuracy (on demand) and do not suffer losses in accuracy near the boundary
- Only way to handle large ($M >> 10000$) molecules

Future Work

- Binding computations: using correlated sampling by directly reprocessing walks
- Simple code interface for distribution with
  1. Desired accuracy as input that allows a precalculation of the number of needed trajectories
  2. Importance sampling for optimal estimation of scalar energy values
  3. Built-in `CONDOR` support for distribution of concurrent tasks
  4. Multicore distributed computing support for the code: OpenMP/OpenMPI
  5. Precompiled code module distribution to protect IP
  6. Webpage to describe the method and the mathematical background and application
- Exploit the implicit inverse computation this methods provides
  1. Can do computation without knowing charges until the end (an inverse)
  2. Simple to examine many charge distributions in a perfectly correlated setting

Future Work

- Further algorithmic development
  1. Computation of gradients using existing Markov chains
  2. Global computation of field variables and their visualization
  3. Nonlinear BVPs perhaps via branching processes
  4. Using “Walk-on-the-Boundary" (WOB) techniques
- Geometric Issues
  1. Computation of the three region model problem
  2. More complicated surfaces (solvent-excluded and ion-excluded)
  3. Accuracy issues related to the Van der Waals surface
- Optimize the performance
  1. Error/bias/variance balancing
  2. Importance sampling and the outer walks
  3. WOB to eliminate walks outside
  4. QMC methods

Bibliography

- [T. Mackoy, R. C. Harris, J. Johnson, M. Mascagni and M. O. Fenley (2011)] Numerical Optimization of a Walk-on-Spheres Solver for the Linear Poisson-Boltzmann Equation *Communications in Computational Physics*, in the press.
- [M. Fenley, M. Mascagni, J. McClain, A. Silalahi and N. Simonov (2010)] Using Correlated Monte Carlo Sampling for Efficiently Solving the Linearized Poisson-Boltzmann Equation Over a Broad Range of Salt Concentrations *Journal of Chemical Theory and Computation*, **6(1)**: 300–314.
- [N. Simonov and M. Mascagni and M. O. Fenley (2007)] Monte Carlo Based Linear Poisson-Boltzmann Approach Makes Accurate Salt-Dependent Solvation Energy Predictions Possible *Journal of Chemical Physics*, **187(18)**, article #185105, 6 pages.

Bibliography

- [M. Mascagni and N. A. Simonov (2004)] Monte Carlo Methods for Calculating Some Physical Properties of Large Molecules *SIAM Journal on Scientific Computing*, **26(1)**: 339–357.
- [N. A. Simonov and M. Mascagni (2004)] Random Walk Algorithms for Estimating Effective Properties of Digitized Porous Media *Monte Carlo Methods and Applications*, **10**: 599–608.
- [M. Mascagni and N. A. Simonov (2004)] The Random Walk on the Boundary Method for Calculating Capacitance *Journal of Computational Physics*, **195**: 465–473.
- [A. Karaivanova, N. A. Simonov and M. Mascagni(2004)] Parallel Quasirandom Walks on the Boundary *Monte Carlo Methods and Applications*, **11**: 311–320.

Bibliography

- [C.-O. Hwang and M. Mascagni (2003)] Analysis and Comparison of Green’s Function First-Passage Algorithms with “Walk on Spheres” Algorithms *Mathematics and Computers in Simulation*, **63**: 605–613.
- [C.-O. Hwang and M. Mascagni (2001)] Efficient modified “walk on spheres" algortihm for the linearized Poisson-Boltzmann equation *Applied Physics Letters*, **78**: 787–789.
- [C.-O. Hwang, J. A. Given and M. Mascagni (2001)] The Simulation-Tabulation Method for Classical Diffusion Monte Carlo *Journal of Computational Physics*, **174**: 925–946.
- [C.-O. Hwang, J. A. Given and M. Mascagni (2000)] On the Rapid Calculation of Permeability for Porous Media Using Brownian Motion Paths *Physics of Fluids*, **12**: 1699–1709.