Monte Carlo Methods for Partial Differential Equations

Prof. Michael Mascagni

Department of Computer Science
Department of Mathematics
Department of Scientific Computing
Florida State University, Tallahassee, FL 32306 **USA**

E-mail: mascagni@fsu.edu or mascagni@math.ethz.ch
URL: http://www.cs.fsu.edu/∼mascagni

In collaboration with Drs. Marcia O. Fenley, and Nikolai Simonov and Messrs. Alexander Silalahi, and James McClain

Research supported by ARO, DOE/ASCI, NATO, and NSF

Introduction

Introduction

Introduction

Introduction

Introduction

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

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

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

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

Early History of MCMs for PDEs

1. Curtiss: Compared Monte Carlo, direct and iterative solution methods for $\mathbf{Ax} = \mathbf{b}$
   - General conclusions of all this work (as other methods were explored) is that random walk methods do worse than conventional methods on serial computers except when modest precision and few solution values are required
   - Much of this "conventional wisdom" needs revision due to complexity differences with parallel implementations

Early History of MCMs for PDEs

1. Curtiss: Compared Monte Carlo, direct and iterative solution methods for $\mathbf{Ax} = \mathbf{b}$
   - General conclusions of all this work (as other methods were explored) is that random walk methods do worse than conventional methods on serial computers except when modest precision and few solution values are required
   - Much of this “conventional wisdom” needs revision due to complexity differences with parallel implementations

Early History of MCMs for PDEs

1. Curtiss: Compared Monte Carlo, direct and iterative solution methods for $\mathbf{Ax} = \mathbf{b}$
   - General conclusions of all this work (as other methods were explored) is that random walk methods do worse than conventional methods on serial computers except when modest precision and few solution values are required
   - Much of this "conventional wisdom" needs revision due to complexity differences with parallel implementations

Elliptic PDEs as Boundary Value Problems

1. Elliptic PDEs describe equilibrium, like the electrostatic field set up by a charge distribution, or the strain in a beam due to loading
2. No time dependence in elliptic problems so it is natural to have the interior configuration satisfy a PDE with boundary conditions to choose a particular global solution
3. Elliptic PDEs are thus part of boundary value problems (BVPs) such as the famous Dirichlet problem for Laplace's equation:

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

4. Here $\Omega \subset \mathbb{R}^s$ is a open set (domain) with a smooth boundary $\partial\Omega$ and $g(x)$ is the given boundary condition

Elliptic PDEs as Boundary Value Problems

1. Elliptic PDEs describe equilibrium, like the electrostatic field set up by a charge distribution, or the strain in a beam due to loading
2. No time dependence in elliptic problems so it is natural to have the interior configuration satisfy a PDE with boundary conditions to choose a particular global solution
3. Elliptic PDEs are thus part of boundary value problems (BVPs) such as the famous Dirichlet problem for Laplace's equation:

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

4. Here $\Omega \subset \mathbb{R}^s$ is a open set (domain) with a smooth boundary $\partial\Omega$ and $g(x)$ is the given boundary condition

Elliptic PDEs as Boundary Value Problems

1. Elliptic PDEs describe equilibrium, like the electrostatic field set up by a charge distribution, or the strain in a beam due to loading
2. No time dependence in elliptic problems so it is natural to have the interior configuration satisfy a PDE with boundary conditions to choose a particular global solution
3. Elliptic PDEs are thus part of boundary value problems (BVPs) such as the famous Dirichlet problem for Laplace's equation:

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

4. Here $\Omega \subset \mathbb{R}^s$ is a open set (domain) with a smooth boundary $\partial\Omega$ and $g(x)$ is the given boundary condition

Elliptic PDEs as Boundary Value Problems

1. Elliptic PDEs describe equilibrium, like the electrostatic field set up by a charge distribution, or the strain in a beam due to loading
2. No time dependence in elliptic problems so it is natural to have the interior configuration satisfy a PDE with boundary conditions to choose a particular global solution
3. Elliptic PDEs are thus part of boundary value problems (BVPs) such as the famous Dirichlet problem for Laplace's equation:

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

4. Here $\Omega \subset \mathbb{R}^s$ is a open set (domain) with a smooth boundary $\partial\Omega$ and $g(x)$ is the given boundary condition

Elliptic PDEs as Boundary Value Problems

- An important equivalence for the Laplace equation is the mean value property (MVP), i.e. if $u(x)$ is a solution to (1) then:

$$u(x) = \frac{1}{|\partial S^n(x,r)|} \int_{\partial S^n(x,r)} u(y)\, dy$$

  $\partial S^n(x,r)$ is the surface of an n-dimensional sphere centered at x with radius r
- Another way to express $u(x)$ is via the Green's function: $u(x) = \int_{\partial\Omega} G(x,y)u(y)\, dy$
- Showing a function has the MVP and the right boundary values establishes it as the unique solution to (1)

Elliptic PDEs as Boundary Value Problems

- An important equivalence for the Laplace equation is the mean value property (MVP), i.e. if $u(x)$ is a solution to (1) then:

$$u(x) = \frac{1}{|\partial S^n(x,r)|} \int_{\partial S^n(x,r)} u(y)\, dy$$

  $\partial S^n(x,r)$ is the surface of an n-dimensional sphere centered at x with radius r
- Another way to express $u(x)$ is via the Green's function: $u(x) = \int_{\partial\Omega} G(x,y)u(y)\, dy$
- Showing a function has the MVP and the right boundary values establishes it as the unique solution to (1)

Elliptic PDEs as Boundary Value Problems

- An important equivalence for the Laplace equation is the mean value property (MVP), i.e. if $u(x)$ is a solution to (1) then:

$$u(x) = \frac{1}{|\partial S^n(x,r)|} \int_{\partial S^n(x,r)} u(y)\, dy$$

  $\partial S^n(x,r)$ is the surface of an n-dimensional sphere centered at x with radius r
- Another way to express $u(x)$ is via the Green's function: $u(x) = \int_{\partial\Omega} G(x,y)u(y)\, dy$
- Showing a function has the MVP and the right boundary values establishes it as the unique solution to (1)

Probabilistic Approaches to Elliptic PDEs

- Early this century probabilists placed measures on different sets including sets of continuous functions
  1. *Called Wiener measure*
  2. *Gaussian based:* $\frac{1}{\sqrt{2\pi t}}e^{-\frac{x^2}{2t}}$
  3. *Sample paths are Brownian motion*
  4. *Related to linear PDEs*
- E.g. $u(x) = E_x[g(\beta(\tau_{\partial\Omega}))]$ is the Wiener integral representation of the solution to (1), to prove it we must check:
  1. *$u(x) = g(x)$ on $\partial\Omega$*
  2. *$u(x)$ has the MVP*
- Interpretation via Brownian motion and/or a probabilistic Green's function
- Important: $\tau_{\partial\Omega}$ = first passage (hitting) time of the path $\beta(\cdot)$ started at x to $\partial\Omega$, statistics based on this random variable are intimately related to elliptic problems

Probabilistic Approaches to Elliptic PDEs

- Early this century probabilists placed measures on different sets including sets of continuous functions
  1. *Called Wiener measure*
  2. *Gaussian based:* $\frac{1}{\sqrt{2\pi t}} e^{-\frac{x^2}{2t}}$
  3. *Sample paths are Brownian motion*
  4. *Related to linear PDEs*
- E.g. $u(x) = E_x[g(\beta(\tau_{\partial\Omega}))]$ is the Wiener integral representation of the solution to (1), to prove it we must check:
  1. $u(x) = g(x)$ *on* $\partial\Omega$
  2. $u(x)$ *has the MVP*
- Interpretation via Brownian motion and/or a probabilistic Green's function
- Important: $\tau_{\partial\Omega}$ = first passage (hitting) time of the path $\beta(\cdot)$ started at x to $\partial\Omega$, statistics based on this random variable are intimately related to elliptic problems

Probabilistic Approaches to Elliptic PDEs

- Early this century probabilists placed measures on different sets including sets of continuous functions
  1. *Called Wiener measure*
  2. *Gaussian based:* $\frac{1}{\sqrt{2\pi t}}e^{-\frac{x^2}{2t}}$
  3. *Sample paths are Brownian motion*
  4. *Related to linear PDEs*
- E.g. $u(x) = E_x[g(\beta(\tau_{\partial\Omega}))]$ is the Wiener integral representation of the solution to (1), to prove it we must check:
  1. *$u(x) = g(x)$ on $\partial\Omega$*
  2. *$u(x)$ has the MVP*
- Interpretation via Brownian motion and/or a probabilistic Green's function
- Important: $\tau_{\partial\Omega}$ = first passage (hitting) time of the path $\beta(\cdot)$ started at x to $\partial\Omega$, statistics based on this random variable are intimately related to elliptic problems

Probabilistic Approaches to Elliptic PDEs

- Early this century probabilists placed measures on different sets including sets of continuous functions
  1. *Called Wiener measure*
  2. *Gaussian based:* $\frac{1}{\sqrt{2\pi t}} e^{-\frac{x^2}{2t}}$
  3. *Sample paths are Brownian motion*
  4. *Related to linear PDEs*
- E.g. $u(x) = E_x[g(\beta(\tau_{\partial\Omega}))]$ is the Wiener integral representation of the solution to (1), to prove it we must check:
  1. $u(x) = g(x)$ *on* $\partial\Omega$
  2. $u(x)$ *has the MVP*
- Interpretation via Brownian motion and/or a probabilistic Green's function
- Important: $\tau_{\partial\Omega}$ = first passage (hitting) time of the path $\beta(\cdot)$ started at x to $\partial\Omega$, statistics based on this random variable are intimately related to elliptic problems

Probabilistic Approaches to Elliptic PDEs

- Early this century probabilists placed measures on different sets including sets of continuous functions
  1. *Called Wiener measure*
  2. *Gaussian based:* $\frac{1}{\sqrt{2\pi t}} e^{-\frac{x^2}{2t}}$
  3. *Sample paths are Brownian motion*
  4. *Related to linear PDEs*
- E.g. $u(x) = E_x[g(\beta(\tau_{\partial\Omega}))]$ is the Wiener integral representation of the solution to (1), to prove it we must check:
  1. $u(x) = g(x)$ *on* $\partial\Omega$
  2. $u(x)$ *has the MVP*
- Interpretation via Brownian motion and/or a probabilistic Green's function
- Important: $\tau_{\partial\Omega}$ = first passage (hitting) time of the path $\beta(\cdot)$ started at x to $\partial\Omega$, statistics based on this random variable are intimately related to elliptic problems

Probabilistic Approaches to Elliptic PDEs

- Early this century probabilists placed measures on different sets including sets of continuous functions
  1. *Called Wiener measure*
  2. *Gaussian based:* $\frac{1}{\sqrt{2\pi t}} e^{-\frac{x^2}{2t}}$
  3. *Sample paths are Brownian motion*
  4. *Related to linear PDEs*
- E.g. $u(x) = E_x[g(\beta(\tau_{\partial\Omega}))]$ is the Wiener integral representation of the solution to (1), to prove it we must check:
  1. $u(x) = g(x)$ *on* $\partial\Omega$
  2. $u(x)$ *has the MVP*
- Interpretation via Brownian motion and/or a probabilistic Green's function
- Important: $\tau_{\partial\Omega}$ = first passage (hitting) time of the path $\beta(\cdot)$ started at x to $\partial\Omega$, statistics based on this random variable are intimately related to elliptic problems

Probabilistic Approaches to Elliptic PDEs

- Early this century probabilists placed measures on different sets including sets of continuous functions
  1. *Called Wiener measure*
  2. *Gaussian based:* $\frac{1}{\sqrt{2\pi t}} e^{-\frac{x^2}{2t}}$
  3. *Sample paths are Brownian motion*
  4. *Related to linear PDEs*
- E.g. $u(x) = E_x[g(\beta(\tau_{\partial\Omega}))]$ is the Wiener integral representation of the solution to (1), to prove it we must check:
  1. $u(x) = g(x)$ *on* $\partial\Omega$
  2. $u(x)$ *has the MVP*
- Interpretation via Brownian motion and/or a probabilistic Green's function
- Important: $\tau_{\partial\Omega} =$ first passage (hitting) time of the path $\beta(\cdot)$ started at x to $\partial\Omega$, statistics based on this random variable are intimately related to elliptic problems

Probabilistic Approaches to Elliptic PDEs

- Early this century probabilists placed measures on different sets including sets of continuous functions
  1. *Called Wiener measure*
  2. *Gaussian based:* $\frac{1}{\sqrt{2\pi t}} e^{-\frac{x^2}{2t}}$
  3. *Sample paths are Brownian motion*
  4. *Related to linear PDEs*
- E.g. $u(x) = E_x[g(\beta(\tau_{\partial\Omega}))]$ is the Wiener integral representation of the solution to (1), to prove it we must check:
  1. $u(x) = g(x)$ *on* $\partial\Omega$
  2. $u(x)$ *has the MVP*
- Interpretation via Brownian motion and/or a probabilistic Green's function
- Important: $\tau_{\partial\Omega}$ = first passage (hitting) time of the path $\beta(\cdot)$ started at x to $\partial\Omega$, statistics based on this random variable are intimately related to elliptic problems

Probabilistic Approaches to Elliptic PDEs

- Early this century probabilists placed measures on different sets including sets of continuous functions
  1. *Called Wiener measure*
  2. *Gaussian based:* $\frac{1}{\sqrt{2\pi t}}e^{-\frac{x^2}{2t}}$
  3. *Sample paths are Brownian motion*
  4. *Related to linear PDEs*
- E.g. $u(x) = E_x[g(\beta(\tau_{\partial\Omega}))]$ is the Wiener integral representation of the solution to (1), to prove it we must check:
  1. $u(x) = g(x)$ *on* $\partial\Omega$
  2. $u(x)$ *has the MVP*
- Interpretation via Brownian motion and/or a probabilistic Green's function
- Important: $\tau_{\partial\Omega}$ = first passage (hitting) time of the path $\beta(\cdot)$ started at x to $\partial\Omega$, statistics based on this random variable are intimately related to elliptic problems

Probabilistic Approaches to Elliptic PDEs

- Early this century probabilists placed measures on different sets including sets of continuous functions
  1. *Called Wiener measure*
  2. *Gaussian based:* $\frac{1}{\sqrt{2\pi t}} e^{-\frac{x^2}{2t}}$
  3. *Sample paths are Brownian motion*
  4. *Related to linear PDEs*
- E.g. $u(x) = E_x[g(\beta(\tau_{\partial\Omega}))]$ is the Wiener integral representation of the solution to (1), to prove it we must check:
  1. $u(x) = g(x)$ *on* $\partial\Omega$
  2. $u(x)$ *has the MVP*
- Interpretation via Brownian motion and/or a probabilistic Green's function
- Important: $\tau_{\partial\Omega}$ = first passage (hitting) time of the path $\beta(\cdot)$ started at x to $\partial\Omega$, statistics based on this random variable are intimately related to elliptic problems

Probabilistic Approaches to Elliptic PDEs

- Can generalize Wiener integrals to different BVPs via the relationship between elliptic operators, stochastic differential equations (SDEs), and the Feynman-Kac formula
- E.g. consider the general elliptic PDE:

$$\begin{aligned} Lu(x) - c(x)u(x) &= f(x),\ x \in \Omega,\ c(x) \geq 0, \\ u(x) &= g(x),\ x \in \partial\Omega \end{aligned} \tag{2.1a}$$

where L is an elliptic partial differential operator of the form:

$$L = \frac{1}{2}\sum_{i,j=1}^{s} a_{ij}(x)\frac{\partial^2}{\partial x_i \partial x_j} + \sum_{i=1}^{s} b_i(x)\frac{\partial}{\partial x_i}, \tag{2.1b}$$

Probabilistic Approaches to Elliptic PDEs

- Can generalize Wiener integrals to different BVPs via the relationship between elliptic operators, stochastic differential equations (SDEs), and the Feynman-Kac formula
- E.g. consider the general elliptic PDE:

$$\begin{aligned} Lu(x) - c(x)u(x) &= f(x),\ x \in \Omega,\ c(x) \geq 0, \\ u(x) &= g(x),\ x \in \partial\Omega \end{aligned} \tag{2.1a}$$

where L is an elliptic partial differential operator of the form:

$$L = \frac{1}{2}\sum_{i,j=1}^{s} a_{ij}(x)\frac{\partial^2}{\partial x_i \partial x_j} + \sum_{i=1}^{s} b_i(x)\frac{\partial}{\partial x_i}, \tag{2.1b}$$

Probabilistic Approaches to Elliptic PDEs

- The Wiener integral representation is:

$$u(x) = E_x^L \left[\int_0^{\tau_{\partial\Omega}} \left\{ \frac{g(\beta(\tau_{\partial\Omega}))}{\tau_{\partial\Omega}} - f(\beta(t)) \right\} e^{-\int_0^t c(\beta(s))\,ds}\, dt \right] \tag{2.2a}$$

  the expectation is w.r.t. paths which are solutions to the following (vector) SDE:

$$d\beta(t) = \sigma(\beta(t))\, dW(t) + b(\beta(t))\, dt, \;\; \beta(0) = x \tag{2.2b}$$

- The matrix $\sigma(\cdot)$ is the Choleski factor (matrix-like square root) of $a_{ij}(\cdot)$ in (2.1b)
- To use these ideas to construct MCMs for elliptic BVPs one must:
  1. *Simulate sample paths via SDEs (2.2b)*
  2. *Evaluate (2.2a) on the sample paths*
  3. *Sample until variance is acceptable*

Probabilistic Approaches to Elliptic PDEs

- The Wiener integral representation is:

$$u(x) = E_x^L \left[\int_0^{\tau_{\partial\Omega}} \left\{ \frac{g(\beta(\tau_{\partial\Omega}))}{\tau_{\partial\Omega}} - f(\beta(t)) \right\} e^{-\int_0^t c(\beta(s))\,ds} \, dt \right] \tag{2.2a}$$

  the expectation is w.r.t. paths which are solutions to the following (vector) SDE:

$$d\beta(t) = \sigma(\beta(t))\,dW(t) + b(\beta(t))\,dt, \;\; \beta(0) = x \tag{2.2b}$$

- The matrix $\sigma(\cdot)$ is the Choleski factor (matrix-like square root) of $a_{ij}(\cdot)$ in (2.1b)
- To use these ideas to construct MCMs for elliptic BVPs one must:
  1. *Simulate sample paths via SDEs (2.2b)*
  2. *Evaluate (2.2a) on the sample paths*
  3. *Sample until variance is acceptable*

Probabilistic Approaches to Elliptic PDEs

- The Wiener integral representation is:

$$u(x) = E_x^L \left[\int_0^{\tau_{\partial\Omega}} \left\{ \frac{g(\beta(\tau_{\partial\Omega}))}{\tau_{\partial\Omega}} - f(\beta(t)) \right\} e^{-\int_0^t c(\beta(s))\,ds}\, dt \right] \tag{2.2a}$$

  the expectation is w.r.t. paths which are solutions to the following (vector) SDE:

$$d\beta(t) = \sigma(\beta(t))\, dW(t) + b(\beta(t))\, dt, \;\; \beta(0) = x \tag{2.2b}$$

- The matrix $\sigma(\cdot)$ is the Choleski factor (matrix-like square root) of $a_{ij}(\cdot)$ in (2.1b)
- To use these ideas to construct MCMs for elliptic BVPs one must:
  1. *Simulate sample paths via SDEs (2.2b)*
  2. *Evaluate (2.2a) on the sample paths*
  3. *Sample until variance is acceptable*

Probabilistic Approaches to Elliptic PDEs

- The Wiener integral representation is:

$$u(x) = E_x^L\left[\int_0^{\tau_{\partial\Omega}} \left\{\frac{g(\beta(\tau_{\partial\Omega}))}{\tau_{\partial\Omega}} - f(\beta(t))\right\} e^{-\int_0^t c(\beta(s))\,ds}\,dt\right] \tag{2.2a}$$

  the expectation is w.r.t. paths which are solutions to the following (vector) SDE:

$$d\beta(t) = \sigma(\beta(t))\,dW(t) + b(\beta(t))\,dt,\ \ \beta(0) = x \tag{2.2b}$$

- The matrix $\sigma(\cdot)$ is the Choleski factor (matrix-like square root) of $a_{ij}(\cdot)$ in (2.1b)
- To use these ideas to construct MCMs for elliptic BVPs one must:
  1. *Simulate sample paths via SDEs (2.2b)*
  2. *Evaluate (2.2a) on the sample paths*
  3. *Sample until variance is acceptable*

Probabilistic Approaches to Elliptic PDEs

- The Wiener integral representation is:

$$u(x) = E_x^L \left[\int_0^{\tau_{\partial\Omega}} \left\{ \frac{g(\beta(\tau_{\partial\Omega}))}{\tau_{\partial\Omega}} - f(\beta(t)) \right\} e^{-\int_0^t c(\beta(s))\,ds}\, dt \right] \tag{2.2a}$$

  the expectation is w.r.t. paths which are solutions to the following (vector) SDE:

$$d\beta(t) = \sigma(\beta(t))\,dW(t) + b(\beta(t))\,dt, \;\; \beta(0) = x \tag{2.2b}$$

- The matrix $\sigma(\cdot)$ is the Choleski factor (matrix-like square root) of $a_{ij}(\cdot)$ in (2.1b)
- To use these ideas to construct MCMs for elliptic BVPs one must:
  1. *Simulate sample paths via SDEs (2.2b)*
  2. *Evaluate (2.2a) on the sample paths*
  3. *Sample until variance is acceptable*

Probabilistic Approaches to Elliptic PDEs

- The Wiener integral representation is:

$$u(x) = E_x^L \left[\int_0^{\tau_{\partial\Omega}} \left\{ \frac{g(\beta(\tau_{\partial\Omega}))}{\tau_{\partial\Omega}} - f(\beta(t)) \right\} e^{-\int_0^t c(\beta(s))\,ds}\, dt \right] \tag{2.2a}$$

  the expectation is w.r.t. paths which are solutions to the following (vector) SDE:

$$d\beta(t) = \sigma(\beta(t))\,dW(t) + b(\beta(t))\,dt, \;\; \beta(0) = x \tag{2.2b}$$

- The matrix $\sigma(\cdot)$ is the Choleski factor (matrix-like square root) of $a_{ij}(\cdot)$ in (2.1b)
- To use these ideas to construct MCMs for elliptic BVPs one must:
  1. *Simulate sample paths via SDEs (2.2b)*
  2. *Evaluate (2.2a) on the sample paths*
  3. *Sample until variance is acceptable*

Probabilistic Approaches to Parabolic PDEs via Feynman-Kac

- Can generalize Wiener integrals to a wide class of IBVPs via the relationship between elliptic operators, stochastic differential equations (SDEs), and the Feynman-Kac formula
- Recall that $t \to \infty$ parabolic $\to$ elliptic
- E.g. consider the general elliptic PDE:

$$u_t = Lu(x) - c(x)u(x) - f(x),\ x \in \Omega,\ c(x) \geq 0,$$
$$u(x) = g(x),\ x \in \partial\Omega \tag{2.3a}$$

where L is an elliptic partial differential operator of the form:

$$L = \frac{1}{2}\sum_{i,j=1}^{s} a_{ij}(x)\frac{\partial^2}{\partial x_i \partial x_j} + \sum_{i=1}^{s} b_i(x)\frac{\partial}{\partial x_i}, \tag{2.3b}$$

Probabilistic Approaches to Parabolic PDEs via Feynman-Kac

- Can generalize Wiener integrals to a wide class of IBVPs via the relationship between elliptic operators, stochastic differential equations (SDEs), and the Feynman-Kac formula
- Recall that $t \to \infty$ parabolic $\to$ elliptic
- E.g. consider the general elliptic PDE:

$$u_t = Lu(x) - c(x)u(x) - f(x),\ x \in \Omega,\ c(x) \geq 0,$$
$$u(x) = g(x),\ x \in \partial\Omega \tag{2.3a}$$

where L is an elliptic partial differential operator of the form:

$$L = \frac{1}{2}\sum_{i,j=1}^{s} a_{ij}(x)\frac{\partial^2}{\partial x_i \partial x_j} + \sum_{i=1}^{s} b_i(x)\frac{\partial}{\partial x_i}, \tag{2.3b}$$

Probabilistic Approaches to Parabolic PDEs via Feynman-Kac

- Can generalize Wiener integrals to a wide class of IBVPs via the relationship between elliptic operators, stochastic differential equations (SDEs), and the Feynman-Kac formula
- Recall that $t \to \infty$ parabolic $\to$ elliptic
- E.g. consider the general elliptic PDE:

$$\begin{aligned} u_t &= Lu(x) - c(x)u(x) - f(x),\ x \in \Omega,\ c(x) \geq 0, \\ u(x) &= g(x),\ x \in \partial\Omega \end{aligned} \tag{2.3a}$$

where L is an elliptic partial differential operator of the form:

$$L = \frac{1}{2}\sum_{i,j=1}^{s} a_{ij}(x)\frac{\partial^2}{\partial x_i \partial x_j} + \sum_{i=1}^{s} b_i(x)\frac{\partial}{\partial x_i}, \tag{2.3b}$$

Probabilistic Approaches to Parabolic PDEs via Feynman-Kac

- The Wiener integral representation is:

$$u(x,t) = E_x^L\left[g(\beta(\tau_{\partial\Omega}) - \int_0^t f(\beta(t))e^{-\int_0^t c(\beta(s))\,ds}\,dt \right] \quad (2.4a)$$

  the expectation is w.r.t. paths which are solutions to the following (vector) SDE:

$$d\beta(t) = \sigma(\beta(t))\,dW(t) + b(\beta(t))\,dt, \;\; \beta(0) = x \quad (2.4b)$$

- The matrix $\sigma(\cdot)$ is the Choleski factor (matrix-like square root) of $a_{ij}(\cdot)$ in (2.3b)

Probabilistic Approaches to Parabolic PDEs via Feynman-Kac

- The Wiener integral representation is:

$$u(x,t) = E_x^L\left[g(\beta(\tau_{\partial\Omega}) - \int_0^t f(\beta(t))e^{-\int_0^t c(\beta(s))\,ds}\,dt\right] \quad (2.4a)$$

  the expectation is w.r.t. paths which are solutions to the following (vector) SDE:

$$d\beta(t) = \sigma(\beta(t))\,dW(t) + b(\beta(t))\,dt, \;\; \beta(0) = x \quad (2.4b)$$

- The matrix $\sigma(\cdot)$ is the Choleski factor (matrix-like square root) of $a_{ij}(\cdot)$ in (2.3b)

Different SDEs, Different Processes, Different Equations

- The SDE gives us a process, and the process defines L (note: a complete definition of L includes the boundary conditions)
- We have solved only the Dirichlet problem, what about other BCs?
- Neumann Boundary Conditions: $\frac{\partial u}{\partial n} = g(x)$ on $\partial\Omega$
- If one uses reflecting Brownian motion, can sample over these paths
- Mixed Boundary Conditions: $\alpha\frac{\partial u}{\partial n} + \beta u = g(x)$ on $\partial\Omega$
- Use reflecting Brownian motion and first passage probabilities, together

Different SDEs, Different Processes, Different Equations

- The SDE gives us a process, and the process defines L (note: a complete definition of L includes the boundary conditions)
- We have solved only the Dirichlet problem, what about other BCs?
- Neumann Boundary Conditions: $\frac{\partial u}{\partial n} = g(x)$ on $\partial\Omega$
- If one uses reflecting Brownian motion, can sample over these paths
- Mixed Boundary Conditions: $\alpha\frac{\partial u}{\partial n} + \beta u = g(x)$ on $\partial\Omega$
- Use reflecting Brownian motion and first passage probabilities, together

Different SDEs, Different Processes, Different Equations

- The SDE gives us a process, and the process defines L (note: a complete definition of L includes the boundary conditions)
- We have solved only the Dirichlet problem, what about other BCs?
- Neumann Boundary Conditions: $\frac{\partial u}{\partial n} = g(x)$ on $\partial\Omega$
- If one uses reflecting Brownian motion, can sample over these paths
- Mixed Boundary Conditions: $\alpha\frac{\partial u}{\partial n} + \beta u = g(x)$ on $\partial\Omega$
- Use reflecting Brownian motion and first passage probabilities, together

Different SDEs, Different Processes, Different Equations

- The SDE gives us a process, and the process defines L (note: a complete definition of L includes the boundary conditions)
- We have solved only the Dirichlet problem, what about other BCs?
- Neumann Boundary Conditions: $\frac{\partial u}{\partial n} = g(x)$ on $\partial\Omega$
- If one uses reflecting Brownian motion, can sample over these paths
- Mixed Boundary Conditions: $\alpha \frac{\partial u}{\partial n} + \beta u = g(x)$ on $\partial\Omega$
- Use reflecting Brownian motion and first passage probabilities, together

Different SDEs, Different Processes, Different Equations

- The SDE gives us a process, and the process defines L (note: a complete definition of L includes the boundary conditions)
- We have solved only the Dirichlet problem, what about other BCs?
- Neumann Boundary Conditions: $\frac{\partial u}{\partial n} = g(x)$ on $\partial\Omega$
- If one uses reflecting Brownian motion, can sample over these paths
- Mixed Boundary Conditions: $\alpha \frac{\partial u}{\partial n} + \beta u = g(x)$ on $\partial\Omega$
- Use reflecting Brownian motion and first passage probabilities, together

Different SDEs, Different Processes, Different Equations

- The SDE gives us a process, and the process defines L (note: a complete definition of L includes the boundary conditions)
- We have solved only the Dirichlet problem, what about other BCs?
- Neumann Boundary Conditions: $\frac{\partial u}{\partial n} = g(x)$ on $\partial\Omega$
- If one uses reflecting Brownian motion, can sample over these paths
- Mixed Boundary Conditions: $\alpha \frac{\partial u}{\partial n} + \beta u = g(x)$ on $\partial\Omega$
- Use reflecting Brownian motion and first passage probabilities, together

Parabolic PDEs and Initial Value Problems

- Parabolic PDEs are evolution equations: the heat equation specifies how an initial temperature profile evolves with time, a pure initial value problem (IVP):

$$\frac{\partial u}{\partial t} = \frac{1}{2}\Delta u, \quad u(x,0) = u_0(x) \tag{3.1a}$$

- As with elliptic PDEs, there are Feynman-Kac formulas for IVPs and initial-boundary value problems (IBVPs) for parabolic PDEs
- Instead of this approach try to use the fundamental solution, which has a real probabilistic flavor, $\frac{1}{\sqrt{2\pi t}}e^{-\frac{x^2}{2t}}$ is the fundamental solution of (3.1a), in the construction of a MCM

Parabolic PDEs and Initial Value Problems

- Parabolic PDEs are evolution equations: the heat equation specifies how an initial temperature profile evolves with time, a pure initial value problem (IVP):

$$\frac{\partial u}{\partial t} = \frac{1}{2}\Delta u, \quad u(x,0) = u_0(x) \tag{3.1a}$$

- As with elliptic PDEs, there are Feynman-Kac formulas for IVPs and initial-boundary value problems (IBVPs) for parabolic PDEs
- Instead of this approach try to use the fundamental solution, which has a real probabilistic flavor, $\frac{1}{\sqrt{2\pi t}} e^{-\frac{x^2}{2t}}$ is the fundamental solution of (3.1a), in the construction of a MCM

Parabolic PDEs and Initial Value Problems

- Parabolic PDEs are evolution equations: the heat equation specifies how an initial temperature profile evolves with time, a pure initial value problem (IVP):

$$\frac{\partial u}{\partial t} = \frac{1}{2}\Delta u, \quad u(x,0) = u_0(x) \tag{3.1a}$$

- As with elliptic PDEs, there are Feynman-Kac formulas for IVPs and initial-boundary value problems (IBVPs) for parabolic PDEs
- Instead of this approach try to use the fundamental solution, which has a real probabilistic flavor, $\frac{1}{\sqrt{2\pi t}}e^{-\frac{x^2}{2t}}$ is the fundamental solution of (3.1a), in the construction of a MCM

MCMs for Linear Parabolic IVPs

- Consider the IVP in (3.1a), if $u_0(x) = \delta(x - x_0)$ (spike at $x = x_0$), the exact solution is $u(x,t) = \frac{1}{\sqrt{2\pi t}} e^{-\frac{(x-x_0)^2}{2t}}$, can interpret this as $u(x,t)$ is $N(x_0, t)$ for MCM sampling of values of $u(x,t)$
- To solve (3.1a) with $u_0(x)$ general, must approximate $u_0(x)$ with spikes and "move" the spikes via their individual normal distributions
- The approximation of a smooth $u_0(x)$ by spikes is quite poor, and so the MCM above gives a solution with large statistical fluctuations (variance)
- Instead, can solve for the gradient of $u(x,t)$ and integrate back to give a better solution, i.e. if we call $v(x,t) = \frac{\partial u(x,t)}{\partial x}$, $v(x,t)$ solves:

$$\frac{\partial v}{\partial t} = \frac{1}{2}\Delta v, \quad v(x,0) = \frac{\partial u_0(x)}{\partial x} \tag{3.1b}$$

MCMs for Linear Parabolic IVPs

- Consider the IVP in (3.1a), if $u_0(x) = \delta(x - x_0)$ (spike at $x = x_0$), the exact solution is $u(x,t) = \frac{1}{\sqrt{2\pi t}} e^{-\frac{(x-x_0)^2}{2t}}$, can interpret this as $u(x,t)$ is $N(x_0, t)$ for MCM sampling of values of $u(x,t)$
- To solve (3.1a) with $u_0(x)$ general, must approximate $u_0(x)$ with spikes and "move" the spikes via their individual normal distributions
- The approximation of a smooth $u_0(x)$ by spikes is quite poor, and so the MCM above gives a solution with large statistical fluctuations (variance)
- Instead, can solve for the gradient of $u(x,t)$ and integrate back to give a better solution, i.e. if we call $v(x,t) = \frac{\partial u(x,t)}{\partial x}$, $v(x,t)$ solves:

$$\frac{\partial v}{\partial t} = \frac{1}{2}\Delta v, \quad v(x,0) = \frac{\partial u_0(x)}{\partial x} \tag{3.1b}$$

MCMs for Linear Parabolic IVPs

- Consider the IVP in (3.1a), if $u_0(x) = \delta(x - x_0)$ (spike at $x = x_0$), the exact solution is $u(x,t) = \frac{1}{\sqrt{2\pi t}} e^{-\frac{(x-x_0)^2}{2t}}$, can interpret this as $u(x,t)$ is $N(x_0, t)$ for MCM sampling of values of $u(x,t)$
- To solve (3.1a) with $u_0(x)$ general, must approximate $u_0(x)$ with spikes and “move” the spikes via their individual normal distributions
- The approximation of a smooth $u_0(x)$ by spikes is quite poor, and so the MCM above gives a solution with large statistical fluctuations (variance)
- Instead, can solve for the gradient of $u(x,t)$ and integrate back to give a better solution, i.e. if we call $v(x,t) = \frac{\partial u(x,t)}{\partial x}$, $v(x,t)$ solves:

$$\frac{\partial v}{\partial t} = \frac{1}{2}\Delta v, \quad v(x,0) = \frac{\partial u_0(x)}{\partial x} \tag{3.1b}$$

MCMs for Linear Parabolic IVPs

- Consider the IVP in (3.1a), if $u_0(x) = \delta(x - x_0)$ (spike at $x = x_0$), the exact solution is $u(x,t) = \frac{1}{\sqrt{2\pi t}} e^{-\frac{(x-x_0)^2}{2t}}$, can interpret this as $u(x,t)$ is $N(x_0, t)$ for MCM sampling of values of $u(x,t)$
- To solve (3.1a) with $u_0(x)$ general, must approximate $u_0(x)$ with spikes and "move" the spikes via their individual normal distributions
- The approximation of a smooth $u_0(x)$ by spikes is quite poor, and so the MCM above gives a solution with large statistical fluctuations (variance)
- Instead, can solve for the gradient of $u(x,t)$ and integrate back to give a better solution, i.e. if we call $v(x,t) = \frac{\partial u(x,t)}{\partial x}$, $v(x,t)$ solves:

$$\frac{\partial v}{\partial t} = \frac{1}{2}\Delta v, \quad v(x,0) = \frac{\partial u_0(x)}{\partial x} \tag{3.1b}$$

MCMs for Linear Parabolic IVPs

- This variance reduction idea is the basis of the random gradient method:
  1. *Set up the gradient problem*
  2. *Initial gradient is spiky*
  3. *Evolve the gradient via MCM*
  4. *Integrate to recover function*
- Since $v(x,t) = \frac{\partial u(x,t)}{\partial x}$, $u(x,t) = \int_{-\infty}^{x} v(y,t)\, dy$
- Note that if $v(x,t) = \frac{1}{N}\sum_{i=1}^{N} \delta(x - x_i)$ then $u(x,t) = \frac{1}{N}\sum_{\{i|x_i \le x\}} 1$, i.e. a step function
- More generally if $v(x,t) = \sum_{i=1}^{N} a_i \delta(x - x_i)$ then $u(x,t) = \sum_{\{i|x_i \le x\}} a_i$, i.e. a step function, here we can approximate more than monotone initial conditions with a_i's negative
- The random gradient method is very efficient and allows solving pure IVPs on infinite domains without difficulty

MCMs for Linear Parabolic IVPs

- This variance reduction idea is the basis of the random gradient method:
  1. *Set up the gradient problem*
  2. *Initial gradient is spiky*
  3. *Evolve the gradient via MCM*
  4. *Integrate to recover function*
- Since $v(x,t) = \frac{\partial u(x,t)}{\partial x}$, $u(x,t) = \int_{-\infty}^{x} v(y,t)\,dy$
- Note that if $v(x,t) = \frac{1}{N}\sum_{i=1}^{N} \delta(x - x_i)$ then $u(x,t) = \frac{1}{N}\sum_{\{i | x_i \le x\}} 1$, i.e. a step function
- More generally if $v(x,t) = \sum_{i=1}^{N} a_i \delta(x - x_i)$ then $u(x,t) = \sum_{\{i | x_i \le x\}} a_i$, i.e. a step function, here we can approximate more than monotone initial conditions with a_i's negative
- The random gradient method is very efficient and allows solving pure IVPs on infinite domains without difficulty

MCMs for Linear Parabolic IVPs

- This variance reduction idea is the basis of the random gradient method:
  1. *Set up the gradient problem*
  2. *Initial gradient is spiky*
  3. *Evolve the gradient via MCM*
  4. *Integrate to recover function*
- Since $v(x,t) = \frac{\partial u(x,t)}{\partial x}$, $u(x,t) = \int_{-\infty}^{x} v(y,t)\,dy$
- Note that if $v(x,t) = \frac{1}{N}\sum_{i=1}^{N} \delta(x - x_i)$ then $u(x,t) = \frac{1}{N}\sum_{\{i|x_i \le x\}} 1$, i.e. a step function
- More generally if $v(x,t) = \sum_{i=1}^{N} a_i \delta(x - x_i)$ then $u(x,t) = \sum_{\{i|x_i \le x\}} a_i$, i.e. a step function, here we can approximate more than monotone initial conditions with a_i's negative
- The random gradient method is very efficient and allows solving pure IVPs on infinite domains without difficulty

MCMs for Linear Parabolic IVPs

- This variance reduction idea is the basis of the random gradient method:
  1. *Set up the gradient problem*
  2. *Initial gradient is spiky*
  3. *Evolve the gradient via MCM*
  4. *Integrate to recover function*
- Since $v(x,t) = \frac{\partial u(x,t)}{\partial x}$, $u(x,t) = \int_{-\infty}^{x} v(y,t)\,dy$
- Note that if $v(x,t) = \frac{1}{N}\sum_{i=1}^{N} \delta(x - x_i)$ then $u(x,t) = \frac{1}{N}\sum_{\{i|x_i \le x\}} 1$, i.e. a step function
- More generally if $v(x,t) = \sum_{i=1}^{N} a_i \delta(x - x_i)$ then $u(x,t) = \sum_{\{i|x_i \le x\}} a_i$, i.e. a step function, here we can approximate more than monotone initial conditions with a_i's negative
- The random gradient method is very efficient and allows solving pure IVPs on infinite domains without difficulty

MCMs for Linear Parabolic IVPs

- This variance reduction idea is the basis of the random gradient method:
  1. *Set up the gradient problem*
  2. *Initial gradient is spiky*
  3. *Evolve the gradient via MCM*
  4. *Integrate to recover function*
- Since $v(x,t) = \frac{\partial u(x,t)}{\partial x}$, $u(x,t) = \int_{-\infty}^{x} v(y,t)\, dy$
- Note that if $v(x,t) = \frac{1}{N}\sum_{i=1}^{N} \delta(x - x_i)$ then $u(x,t) = \frac{1}{N}\sum_{\{i|x_i \le x\}} 1$, i.e. a step function
- More generally if $v(x,t) = \sum_{i=1}^{N} a_i \delta(x - x_i)$ then $u(x,t) = \sum_{\{i|x_i \le x\}} a_i$, i.e. a step function, here we can approximate more than monotone initial conditions with a_i's negative
- The random gradient method is very efficient and allows solving pure IVPs on infinite domains without difficulty

MCMs for Linear Parabolic IVPs

- This variance reduction idea is the basis of the random gradient method:
  1. *Set up the gradient problem*
  2. *Initial gradient is spiky*
  3. *Evolve the gradient via MCM*
  4. *Integrate to recover function*
- Since $v(x,t) = \frac{\partial u(x,t)}{\partial x}$, $u(x,t) = \int_{-\infty}^{x} v(y,t)\,dy$
- Note that if $v(x,t) = \frac{1}{N}\sum_{i=1}^{N} \delta(x - x_i)$ then $u(x,t) = \frac{1}{N}\sum_{\{i|x_i \le x\}} 1$, i.e. a step function
- More generally if $v(x,t) = \sum_{i=1}^{N} a_i \delta(x - x_i)$ then $u(x,t) = \sum_{\{i|x_i \le x\}} a_i$, i.e. a step function, here we can approximate more than monotone initial conditions with a_i's negative
- The random gradient method is very efficient and allows solving pure IVPs on infinite domains without difficulty

MCMs for Linear Parabolic IVPs

- This variance reduction idea is the basis of the random gradient method:
  1. *Set up the gradient problem*
  2. *Initial gradient is spiky*
  3. *Evolve the gradient via MCM*
  4. *Integrate to recover function*
- Since $v(x,t) = \frac{\partial u(x,t)}{\partial x}$, $u(x,t) = \int_{-\infty}^{x} v(y,t)\, dy$
- Note that if $v(x,t) = \frac{1}{N}\sum_{i=1}^{N} \delta(x - x_i)$ then $u(x,t) = \frac{1}{N}\sum_{\{i|x_i \le x\}} 1$, i.e. a step function
- More generally if $v(x,t) = \sum_{i=1}^{N} a_i \delta(x - x_i)$ then $u(x,t) = \sum_{\{i|x_i \le x\}} a_i$, i.e. a step function, here we can approximate more than monotone initial conditions with a_i's negative
- The random gradient method is very efficient and allows solving pure IVPs on infinite domains without difficulty

MCMs for Linear Parabolic IVPs

- This variance reduction idea is the basis of the random gradient method:
  1. *Set up the gradient problem*
  2. *Initial gradient is spiky*
  3. *Evolve the gradient via MCM*
  4. *Integrate to recover function*
- Since $v(x,t) = \frac{\partial u(x,t)}{\partial x}$, $u(x,t) = \int_{-\infty}^{x} v(y,t)\,dy$
- Note that if $v(x,t) = \frac{1}{N}\sum_{i=1}^{N} \delta(x - x_i)$ then $u(x,t) = \frac{1}{N}\sum_{\{i|x_i \le x\}} 1$, i.e. a step function
- More generally if $v(x,t) = \sum_{i=1}^{N} a_i \delta(x - x_i)$ then $u(x,t) = \sum_{\{i|x_i \le x\}} a_i$, i.e. a step function, here we can approximate more than monotone initial conditions with a_i's negative
- The random gradient method is very efficient and allows solving pure IVPs on infinite domains without difficulty

MCMs for Linear Parabolic IVPs

- This variance reduction idea is the basis of the random gradient method:
  1. *Set up the gradient problem*
  2. *Initial gradient is spiky*
  3. *Evolve the gradient via MCM*
  4. *Integrate to recover function*
- Since $v(x,t) = \frac{\partial u(x,t)}{\partial x}$, $u(x,t) = \int_{-\infty}^{x} v(y,t)\,dy$
- Note that if $v(x,t) = \frac{1}{N}\sum_{i=1}^{N} \delta(x - x_i)$ then $u(x,t) = \frac{1}{N}\sum_{\{i|x_i \le x\}} 1$, i.e. a step function
- More generally if $v(x,t) = \sum_{i=1}^{N} a_i \delta(x - x_i)$ then $u(x,t) = \sum_{\{i|x_i \le x\}} a_i$, i.e. a step function, here we can approximate more than monotone initial conditions with a_i's negative
- The random gradient method is very efficient and allows solving pure IVPs on infinite domains without difficulty

MCMs for Linear Parabolic IVPs

- Consider the related linear IVP (c is constant):

$$\frac{\partial u}{\partial t} = \frac{1}{2}\Delta u + cu, \quad u(x,0) = u_0(x) \tag{3.2a}$$

- The first term on the r.h.s. is diffusion and its effect may be sampled via a normally distributed random number
- The second term on the r.h.s. is an exponential growth/shrinkage term, can also sample its effect probabilistically:
- Think of dispatching random walkers to do the sampling
  1. *Choose Δt s.t. $\Delta t|c| < 1$*
  2. *Move all walkers via $N(0, \Delta t)$*
  3. *Create/destroy with prob. $= \Delta t|c|$*
  4. *$c > 0$: create by doubling*
  5. *$c < 0$: destroy by removal*

MCMs for Linear Parabolic IVPs

- Consider the related linear IVP (c is constant):

$$\frac{\partial u}{\partial t} = \frac{1}{2}\Delta u + cu, \quad u(x,0) = u_0(x) \tag{3.2a}$$

- The first term on the r.h.s. is diffusion and its effect may be sampled via a normally distributed random number
- The second term on the r.h.s. is an exponential growth/shrinkage term, can also sample its effect probabilistically:
- Think of dispatching random walkers to do the sampling
  1. *Choose Δt s.t. $\Delta t|c| < 1$*
  2. *Move all walkers via $N(0, \Delta t)$*
  3. *Create/destroy with prob. $= \Delta t|c|$*
  4. *$c > 0$: create by doubling*
  5. *$c < 0$: destroy by removal*

MCMs for Linear Parabolic IVPs

- Consider the related linear IVP (*c* is constant):

$$\frac{\partial u}{\partial t} = \frac{1}{2}\Delta u + cu, \quad u(x,0) = u_0(x) \tag{3.2a}$$

- The first term on the r.h.s. is diffusion and its effect may be sampled via a normally distributed random number
- The second term on the r.h.s. is an exponential growth/shrinkage term, can also sample its effect probabilistically:
- Think of dispatching random walkers to do the sampling
  1. *Choose Δt s.t. $\Delta t|c| < 1$*
  2. *Move all walkers via $N(0, \Delta t)$*
  3. *Create/destroy with prob. $= \Delta t|c|$*
  4. *$c > 0$: create by doubling*
  5. *$c < 0$: destroy by removal*

MCMs for Linear Parabolic IVPs

- Consider the related linear IVP (c is constant):

$$\frac{\partial u}{\partial t} = \frac{1}{2}\Delta u + cu, \quad u(x,0) = u_0(x) \tag{3.2a}$$

- The first term on the r.h.s. is diffusion and its effect may be sampled via a normally distributed random number
- The second term on the r.h.s. is an exponential growth/shrinkage term, can also sample its effect probabilistically:
- Think of dispatching random walkers to do the sampling

  1. *Choose Δt s.t. $\Delta t|c| < 1$*
  2. *Move all walkers via $N(0, \Delta t)$*
  3. *Create/destroy with prob. $= \Delta t|c|$*
  4. *$c > 0$: create by doubling*
  5. *$c < 0$: destroy by removal*

MCMs for Linear Parabolic IVPs

- Consider the related linear IVP (c is constant):

$$\frac{\partial u}{\partial t} = \frac{1}{2}\Delta u + cu, \quad u(x,0) = u_0(x) \tag{3.2a}$$

- The first term on the r.h.s. is diffusion and its effect may be sampled via a normally distributed random number
- The second term on the r.h.s. is an exponential growth/shrinkage term, can also sample its effect probabilistically:
- Think of dispatching random walkers to do the sampling
  1. *Choose Δt s.t. $\Delta t|c| < 1$*
  2. *Move all walkers via $N(0, \Delta t)$*
  3. *Create/destroy with prob. $= \Delta t|c|$*
  4. *$c > 0$: create by doubling*
  5. *$c < 0$: destroy by removal*

MCMs for Linear Parabolic IVPs

- Consider the related linear IVP (c is constant):

$$\frac{\partial u}{\partial t} = \frac{1}{2}\Delta u + cu, \quad u(x,0) = u_0(x) \tag{3.2a}$$

- The first term on the r.h.s. is diffusion and its effect may be sampled via a normally distributed random number
- The second term on the r.h.s. is an exponential growth/shrinkage term, can also sample its effect probabilistically:
- Think of dispatching random walkers to do the sampling
  1. *Choose Δt s.t. $\Delta t|c| < 1$*
  2. *Move all walkers via $N(0, \Delta t)$*
  3. *Create/destroy with prob. $= \Delta t|c|$*
  4. *$c > 0$: create by doubling*
  5. *$c < 0$: destroy by removal*

MCMs for Linear Parabolic IVPs

- Consider the related linear IVP (c is constant):

$$\frac{\partial u}{\partial t} = \frac{1}{2}\Delta u + cu, \quad u(x,0) = u_0(x) \tag{3.2a}$$

- The first term on the r.h.s. is diffusion and its effect may be sampled via a normally distributed random number
- The second term on the r.h.s. is an exponential growth/shrinkage term, can also sample its effect probabilistically:
- Think of dispatching random walkers to do the sampling
    1. *Choose Δt s.t. $\Delta t|c| < 1$*
    2. *Move all walkers via $N(0, \Delta t)$*
    3. *Create/destroy with prob. $= \Delta t|c|$*
    4. *$c > 0$: create by doubling*
    5. *$c < 0$: destroy by removal*

MCMs for Linear Parabolic IVPs

- Consider the related linear IVP (c is constant):

$$\frac{\partial u}{\partial t} = \frac{1}{2}\Delta u + cu, \quad u(x,0) = u_0(x) \tag{3.2a}$$

- The first term on the r.h.s. is diffusion and its effect may be sampled via a normally distributed random number
- The second term on the r.h.s. is an exponential growth/shrinkage term, can also sample its effect probabilistically:
- Think of dispatching random walkers to do the sampling
  1. *Choose Δt s.t. $\Delta t|c| < 1$*
  2. *Move all walkers via $N(0, \Delta t)$*
  3. *Create/destroy with prob. $= \Delta t|c|$*
  4. *$c > 0$: create by doubling*
  5. *$c < 0$: destroy by removal*

MCMs for Linear Parabolic IVPs

- Consider the related linear IVP (c is constant):

$$\frac{\partial u}{\partial t} = \frac{1}{2}\Delta u + cu, \quad u(x,0) = u_0(x) \tag{3.2a}$$

- The first term on the r.h.s. is diffusion and its effect may be sampled via a normally distributed random number
- The second term on the r.h.s. is an exponential growth/shrinkage term, can also sample its effect probabilistically:
- Think of dispatching random walkers to do the sampling
  1. *Choose Δt s.t. $\Delta t|c| < 1$*
  2. *Move all walkers via $N(0, \Delta t)$*
  3. *Create/destroy with prob. $= \Delta t|c|$*
  4. *$c > 0$: create by doubling*
  5. *$c < 0$: destroy by removal*

MCMs for Linear Parabolic IVPs

- Consider the related gradient IVP:

$$\frac{\partial v}{\partial t} = \frac{1}{2}\Delta v + cv, \quad v(x,0) = \frac{\partial u_0(x)}{\partial x} \tag{3.2b}$$

- Let us summarize the algorithm for the MCM to advance the solution of (3.2a) one time step using Δt:
  1. Represent $v(x,t) = \frac{1}{N}\sum_{i=1}^{N}\delta(x - x_i)$
  2. Choose Δt s.t. $\Delta t|c| < 1$
  3. Move x_i to $x_i + \eta_i$ where η_i is $N(0, \Delta t)$
  4. If $c > 0$ create new walkers at those x_i where $\xi_i < \Delta t c$ with ξ_i $U[0,1]$
  5. If $c < 0$ destroy walkers at those x_i where $\xi_i < -\Delta t c$ with ξ_i $U[0,1]$
  6. Over the remaining points, $u(x, t+\Delta t) = \frac{1}{N}\sum_{\{i|x_i \le x\}} 1$
- This is the linear PDE version of the random gradient method (RGM)

MCMs for Linear Parabolic IVPs

- Consider the related gradient IVP:

$$\frac{\partial v}{\partial t} = \frac{1}{2}\Delta v + cv, \quad v(x,0) = \frac{\partial u_0(x)}{\partial x} \tag{3.2b}$$

- Let us summarize the algorithm for the MCM to advance the solution of (3.2a) one time step using Δt:
  1. Represent $v(x,t) = \frac{1}{N}\sum_{i=1}^{N}\delta(x-x_i)$
  2. Choose Δt s.t. $\Delta t|c| < 1$
  3. Move x_i to $x_i + \eta_i$ where η_i is $N(0,\Delta t)$
  4. If $c > 0$ create new walkers at those x_i where $\xi_i < \Delta t c$ with ξ_i $U[0,1]$
  5. If $c < 0$ destroy walkers at those x_i where $\xi_i < -\Delta t c$ with ξ_i $U[0,1]$
  6. Over the remaining points, $u(x,t+\Delta t) = \frac{1}{N}\sum_{\{i|x_i\le x\}} 1$
- This is the linear PDE version of the random gradient method (RGM)

MCMs for Linear Parabolic IVPs

- Consider the related gradient IVP:

$$\frac{\partial v}{\partial t} = \frac{1}{2}\Delta v + cv, \quad v(x,0) = \frac{\partial u_0(x)}{\partial x} \tag{3.2b}$$

- Let us summarize the algorithm for the MCM to advance the solution of (3.2a) one time step using Δt:
  1. Represent $v(x,t) = \frac{1}{N}\sum_{i=1}^{N} \delta(x - x_i)$
  2. Choose Δt s.t. $\Delta t|c| < 1$
  3. Move x_i to $x_i + \eta_i$ where η_i is $N(0, \Delta t)$
  4. If $c > 0$ create new walkers at those x_i where $\xi_i < \Delta tc$ with ξ_i $U[0,1]$
  5. If $c < 0$ destroy walkers at those x_i where $\xi_i < -\Delta tc$ with ξ_i $U[0,1]$
  6. Over the remaining points, $u(x, t + \Delta t) = \frac{1}{N}\sum_{\{i|x_i \leq x\}} 1$
- This is the linear PDE version of the random gradient method (RGM)

MCMs for Linear Parabolic IVPs

- Consider the related gradient IVP:

$$\frac{\partial v}{\partial t} = \frac{1}{2}\Delta v + cv, \quad v(x,0) = \frac{\partial u_0(x)}{\partial x} \tag{3.2b}$$

- Let us summarize the algorithm for the MCM to advance the solution of (3.2a) one time step using Δt:
  1. Represent $v(x,t) = \frac{1}{N}\sum_{i=1}^{N} \delta(x - x_i)$
  2. Choose Δt s.t. $\Delta t|c| < 1$
  3. Move x_i to $x_i + \eta_i$ where η_i is $N(0,\Delta t)$
  4. If $c > 0$ create new walkers at those x_i where $\xi_i < \Delta tc$ with ξ_i $U[0,1]$
  5. If $c < 0$ destroy walkers at those x_i where $\xi_i < -\Delta tc$ with ξ_i $U[0,1]$
  6. Over the remaining points, $u(x, t + \Delta t) = \frac{1}{N}\sum_{\{i|x_i \le x\}} 1$
- This is the linear PDE version of the random gradient method (RGM)

MCMs for Linear Parabolic IVPs

- Consider the related gradient IVP:

$$\frac{\partial v}{\partial t} = \frac{1}{2}\Delta v + cv, \quad v(x,0) = \frac{\partial u_0(x)}{\partial x} \tag{3.2b}$$

- Let us summarize the algorithm for the MCM to advance the solution of (3.2a) one time step using Δt:
  1. Represent $v(x,t) = \frac{1}{N}\sum_{i=1}^{N} \delta(x - x_i)$
  2. Choose Δt s.t. $\Delta t|c| < 1$
  3. Move x_i to $x_i + \eta_i$ where η_i is $N(0, \Delta t)$
  4. If $c > 0$ create new walkers at those x_i where $\xi_i < \Delta tc$ with ξ_i $U[0,1]$
  5. If $c < 0$ destroy walkers at those x_i where $\xi_i < -\Delta tc$ with ξ_i $U[0,1]$
  6. Over the remaining points, $u(x, t + \Delta t) = \frac{1}{N}\sum_{\{i|x_i \le x\}} 1$
- This is the linear PDE version of the random gradient method (RGM)

MCMs for Linear Parabolic IVPs

- Consider the related gradient IVP:

$$\frac{\partial v}{\partial t} = \frac{1}{2}\Delta v + cv, \quad v(x,0) = \frac{\partial u_0(x)}{\partial x} \tag{3.2b}$$

- Let us summarize the algorithm for the MCM to advance the solution of (3.2a) one time step using Δt:
  1. Represent $v(x,t) = \frac{1}{N}\sum_{i=1}^{N} \delta(x - x_i)$
  2. Choose Δt s.t. $\Delta t|c| < 1$
  3. Move x_i to $x_i + \eta_i$ where η_i is $N(0, \Delta t)$
  4. If $c > 0$ create new walkers at those x_i where $\xi_i < \Delta tc$ with ξ_i $U[0,1]$
  5. If $c < 0$ destroy walkers at those x_i where $\xi_i < -\Delta tc$ with ξ_i $U[0,1]$
  6. Over the remaining points, $u(x, t + \Delta t) = \frac{1}{N}\sum_{\{i|x_i \le x\}} 1$
- This is the linear PDE version of the random gradient method (RGM)

MCMs for Linear Parabolic IVPs

- Consider the related gradient IVP:

$$\frac{\partial v}{\partial t} = \frac{1}{2}\Delta v + cv, \quad v(x,0) = \frac{\partial u_0(x)}{\partial x} \tag{3.2b}$$

- Let us summarize the algorithm for the MCM to advance the solution of (3.2a) one time step using Δt:
  1. Represent $v(x,t) = \frac{1}{N}\sum_{i=1}^{N} \delta(x - x_i)$
  2. Choose Δt s.t. $\Delta t|c| < 1$
  3. Move x_i to $x_i + \eta_i$ where η_i is $N(0, \Delta t)$
  4. If $c > 0$ create new walkers at those x_i where $\xi_i < \Delta tc$ with ξ_i $U[0,1]$
  5. If $c < 0$ destroy walkers at those x_i where $\xi_i < -\Delta tc$ with ξ_i $U[0,1]$
  6. Over the remaining points, $u(x, t + \Delta t) = \frac{1}{N}\sum_{\{i|x_i \leq x\}} 1$
- This is the linear PDE version of the random gradient method (RGM)

MCMs for Linear Parabolic IVPs

- Consider the related gradient IVP:

$$\frac{\partial v}{\partial t} = \frac{1}{2}\Delta v + cv, \quad v(x,0) = \frac{\partial u_0(x)}{\partial x} \tag{3.2b}$$

- Let us summarize the algorithm for the MCM to advance the solution of (3.2a) one time step using Δt:
  1. Represent $v(x,t) = \frac{1}{N}\sum_{i=1}^{N}\delta(x - x_i)$
  2. Choose Δt s.t. $\Delta t|c| < 1$
  3. Move x_i to $x_i + \eta_i$ where η_i is $N(0, \Delta t)$
  4. If $c > 0$ create new walkers at those x_i where $\xi_i < \Delta tc$ with ξ_i $U[0,1]$
  5. If $c < 0$ destroy walkers at those x_i where $\xi_i < -\Delta tc$ with ξ_i $U[0,1]$
  6. Over the remaining points, $u(x, t + \Delta t) = \frac{1}{N}\sum_{\{i|x_i \leq x\}} 1$
- This is the linear PDE version of the random gradient method (RGM)

MCMs for Linear Parabolic IVPs

- Consider the related gradient IVP:

$$\frac{\partial v}{\partial t} = \frac{1}{2}\Delta v + cv, \quad v(x,0) = \frac{\partial u_0(x)}{\partial x} \tag{3.2b}$$

- Let us summarize the algorithm for the MCM to advance the solution of (3.2a) one time step using Δt:
  1. Represent $v(x,t) = \frac{1}{N}\sum_{i=1}^{N} \delta(x - x_i)$
  2. Choose Δt s.t. $\Delta t|c| < 1$
  3. Move x_i to $x_i + \eta_i$ where η_i is $N(0, \Delta t)$
  4. If $c > 0$ create new walkers at those x_i where $\xi_i < \Delta tc$ with ξ_i $U[0,1]$
  5. If $c < 0$ destroy walkers at those x_i where $\xi_i < -\Delta tc$ with ξ_i $U[0,1]$
  6. Over the remaining points, $u(x, t + \Delta t) = \frac{1}{N}\sum_{\{i|x_i \le x\}} 1$
- This is the linear PDE version of the random gradient method (RGM)

The RGM for Nonlinear Parabolic IVPs

- Consider the IVP for a nonlinear scalar reaction diffusion equation:

$$\frac{\partial u}{\partial t} = \frac{1}{2}\Delta u + c(u), \quad u(x,0) = u_0(x) \tag{3.3a}$$

- The associated gradient equation is:

$$\frac{\partial v}{\partial t} = \frac{1}{2}\Delta v + c'(u)v, \quad v(x,0) = \frac{\partial u_0(x)}{\partial x} \tag{3.3b}$$

- The similarity of (3.3b) to (3.2b) make it clear how to extend the RGM method to these nonlinear scalar reaction diffusion equations

The RGM for Nonlinear Parabolic IVPs

- Consider the IVP for a nonlinear scalar reaction diffusion equation:

$$\frac{\partial u}{\partial t} = \frac{1}{2}\Delta u + c(u), \quad u(x,0) = u_0(x) \tag{3.3a}$$

- The associated gradient equation is:

$$\frac{\partial v}{\partial t} = \frac{1}{2}\Delta v + c'(u)v, \quad v(x,0) = \frac{\partial u_0(x)}{\partial x} \tag{3.3b}$$

- The similarity of (3.3b) to (3.2b) make it clear how to extend the RGM method to these nonlinear scalar reaction diffusion equations

The RGM for Nonlinear Parabolic IVPs

- Consider the IVP for a nonlinear scalar reaction diffusion equation:

$$\frac{\partial u}{\partial t} = \frac{1}{2}\Delta u + c(u), \quad u(x,0) = u_0(x) \tag{3.3a}$$

- The associated gradient equation is:

$$\frac{\partial v}{\partial t} = \frac{1}{2}\Delta v + c'(u)v, \quad v(x,0) = \frac{\partial u_0(x)}{\partial x} \tag{3.3b}$$

- The similarity of (3.3b) to (3.2b) make it clear how to extend the RGM method to these nonlinear scalar reaction diffusion equations

The RGM for Nonlinear Parabolic IVPs

- Summary of the algorithm for the RGM to advance the solution of (3.3a) one time step using Δt:
  1. Represent $v(x,t) = \frac{1}{N}\sum_{i=1}^{N} \delta(x - x_i)$
  2. Choose Δt s.t. $\Delta t\,(\sup_u |c'(u)|) < 1$
  3. Move x_i to $x_i + \eta_i$ where η_i is $N(0, \Delta t)$
  4. At those x_i where $c'(u) > 0$ create new walkers if $\xi_i < \Delta t c'(u)$ with ξ_i $U[0,1]$
  5. At those x_i where $c'(u) < 0$ destroy walkers if $\xi_i < -\Delta t c'(u)$ with ξ_i $U[0,1]$
  6. Over the remaining points, $u(x, t + \Delta t) = \frac{1}{N}\sum_{\{i|x_i \leq x\}} 1$
- This is the nonlinear scalar reaction diffusion version of the RGM

The RGM for Nonlinear Parabolic IVPs

- Summary of the algorithm for the RGM to advance the solution of (3.3a) one time step using Δt:
  1. Represent $v(x,t) = \frac{1}{N}\sum_{i=1}^{N} \delta(x - x_i)$
  2. Choose Δt s.t. $\Delta t\,(\sup_u |c'(u)|) < 1$
  3. Move x_i to $x_i + \eta_i$ where η_i is $N(0, \Delta t)$
  4. At those x_i where $c'(u) > 0$ create new walkers if $\xi_i < \Delta t c'(u)$ with ξ_i $U[0,1]$
  5. At those x_i where $c'(u) < 0$ destroy walkers if $\xi_i < -\Delta t c'(u)$ with ξ_i $U[0,1]$
  6. Over the remaining points, $u(x, t + \Delta t) = \frac{1}{N}\sum_{\{i|x_i \le x\}} 1$
- This is the nonlinear scalar reaction diffusion version of the RGM

The RGM for Nonlinear Parabolic IVPs

- Summary of the algorithm for the RGM to advance the solution of (3.3a) one time step using Δt:
  1. Represent $v(x,t) = \frac{1}{N}\sum_{i=1}^{N} \delta(x - x_i)$
  2. Choose Δt s.t. $\Delta t \left(\sup_u |c'(u)|\right) < 1$
  3. Move x_i to $x_i + \eta_i$ where η_i is $N(0, \Delta t)$
  4. At those x_i where $c'(u) > 0$ create new walkers if $\xi_i < \Delta t c'(u)$ with ξ_i $U[0,1]$
  5. At those x_i where $c'(u) < 0$ destroy walkers if $\xi_i < -\Delta t c'(u)$ with ξ_i $U[0,1]$
  6. Over the remaining points, $u(x, t + \Delta t) = \frac{1}{N}\sum_{\{i|x_i \leq x\}} 1$
- This is the nonlinear scalar reaction diffusion version of the RGM

The RGM for Nonlinear Parabolic IVPs

- Summary of the algorithm for the RGM to advance the solution of (3.3a) one time step using Δt:
  1. Represent $v(x,t) = \frac{1}{N}\sum_{i=1}^{N} \delta(x - x_i)$
  2. Choose Δt s.t. $\Delta t \left(\sup_u |c'(u)|\right) < 1$
  3. Move x_i to $x_i + \eta_i$ where η_i is $N(0, \Delta t)$
  4. At those x_i where $c'(u) > 0$ create new walkers if $\xi_i < \Delta t c'(u)$ with ξ_i $U[0,1]$
  5. At those x_i where $c'(u) < 0$ destroy walkers if $\xi_i < -\Delta t c'(u)$ with ξ_i $U[0,1]$
  6. Over the remaining points, $u(x, t + \Delta t) = \frac{1}{N}\sum_{\{i|x_i \leq x\}} 1$
- This is the nonlinear scalar reaction diffusion version of the RGM

The RGM for Nonlinear Parabolic IVPs

- Summary of the algorithm for the RGM to advance the solution of (3.3a) one time step using Δt:
  1. Represent $v(x,t) = \frac{1}{N}\sum_{i=1}^{N} \delta(x - x_i)$
  2. Choose Δt s.t. $\Delta t\,(\sup_u |c'(u)|) < 1$
  3. Move x_i to $x_i + \eta_i$ where η_i is $N(0, \Delta t)$
  4. At those x_i where $c'(u) > 0$ create new walkers if $\xi_i < \Delta t c'(u)$ with ξ_i $U[0,1]$
  5. At those x_i where $c'(u) < 0$ destroy walkers if $\xi_i < -\Delta t c'(u)$ with ξ_i $U[0,1]$
  6. Over the remaining points, $u(x, t + \Delta t) = \frac{1}{N}\sum_{\{i|x_i \le x\}} 1$
- This is the nonlinear scalar reaction diffusion version of the RGM

The RGM for Nonlinear Parabolic IVPs

- Summary of the algorithm for the RGM to advance the solution of (3.3a) one time step using Δt:
  1. Represent $v(x,t) = \frac{1}{N}\sum_{i=1}^{N}\delta(x - x_i)$
  2. Choose Δt s.t. $\Delta t\,(\sup_u |c'(u)|) < 1$
  3. Move x_i to $x_i + \eta_i$ where η_i is $N(0, \Delta t)$
  4. At those x_i where $c'(u) > 0$ create new walkers if $\xi_i < \Delta t c'(u)$ with ξ_i $U[0, 1]$
  5. At those x_i where $c'(u) < 0$ destroy walkers if $\xi_i < -\Delta t c'(u)$ with ξ_i $U[0, 1]$
  6. Over the remaining points, $u(x, t + \Delta t) = \frac{1}{N}\sum_{\{i|x_i \le x\}} 1$
- This is the nonlinear scalar reaction diffusion version of the RGM

The RGM for Nonlinear Parabolic IVPs

- Summary of the algorithm for the RGM to advance the solution of (3.3a) one time step using Δt:
  1. Represent $v(x,t) = \frac{1}{N}\sum_{i=1}^{N}\delta(x - x_i)$
  2. Choose Δt s.t. $\Delta t\,(\sup_u |c'(u)|) < 1$
  3. Move x_i to $x_i + \eta_i$ where η_i is $N(0, \Delta t)$
  4. At those x_i where $c'(u) > 0$ create new walkers if $\xi_i < \Delta t c'(u)$ with ξ_i $U[0,1]$
  5. At those x_i where $c'(u) < 0$ destroy walkers if $\xi_i < -\Delta t c'(u)$ with ξ_i $U[0,1]$
  6. Over the remaining points, $u(x, t + \Delta t) = \frac{1}{N}\sum_{\{i|x_i \leq x\}} 1$
- This is the nonlinear scalar reaction diffusion version of the RGM

The RGM for Nonlinear Parabolic IVPs

- Summary of the algorithm for the RGM to advance the solution of (3.3a) one time step using Δt:
    1. Represent $v(x,t) = \frac{1}{N}\sum_{i=1}^{N}\delta(x - x_i)$
    2. Choose Δt s.t. $\Delta t\,(\sup_u |c'(u)|) < 1$
    3. Move x_i to $x_i + \eta_i$ where η_i is $N(0, \Delta t)$
    4. At those x_i where $c'(u) > 0$ create new walkers if $\xi_i < \Delta t c'(u)$ with ξ_i $U[0,1]$
    5. At those x_i where $c'(u) < 0$ destroy walkers if $\xi_i < -\Delta t c'(u)$ with ξ_i $U[0,1]$
    6. Over the remaining points, $u(x, t + \Delta t) = \frac{1}{N}\sum_{\{i | x_i \leq x\}} 1$
- This is the nonlinear scalar reaction diffusion version of the RGM

The RGM for Nonlinear Parabolic IVPs

- Differences/advantages of the RGM and conventional (finite-difference/finite-element) methods:
  1. It is computationally easy to sample from $N(0, \Delta t)$
  2. Only costly operation each time step is to sort the remaining cohort of walkers by their position
  3. Can use ensemble averaging to reduce the variance of the solution
  4. Can choose to use either gradient "particles" of equal mass or allow the mass of each particle to change
  5. The RGM is adaptive, computational elements (pieces of the gradient) are created where $c'(u)$ is greatest, this is where sharp fronts appear and so fewer total computational elements are needed to spatially resolve jump-like solutions

The RGM for Nonlinear Parabolic IVPs

- Differences/advantages of the RGM and conventional (finite-difference/finite-element) methods:
  1. It is computationally easy to sample from $N(0, \Delta t)$
  2. Only costly operation each time step is to sort the remaining cohort of walkers by their position
  3. Can use ensemble averaging to reduce the variance of the solution
  4. Can choose to use either gradient "particles" of equal mass or allow the mass of each particle to change
  5. The RGM is adaptive, computational elements (pieces of the gradient) are created where $c'(u)$ is greatest, this is where sharp fronts appear and so fewer total computational elements are needed to spatially resolve jump-like solutions

The RGM for Nonlinear Parabolic IVPs

- Differences/advantages of the RGM and conventional (finite-difference/finite-element) methods:
  1. It is computationally easy to sample from $N(0, \Delta t)$
  2. Only costly operation each time step is to sort the remaining cohort of walkers by their position
  3. Can use ensemble averaging to reduce the variance of the solution
  4. Can choose to use either gradient “particles” of equal mass or allow the mass of each particle to change
  5. The RGM is adaptive, computational elements (pieces of the gradient) are created where $c'(u)$ is greatest, this is where sharp fronts appear and so fewer total computational elements are needed to spatially resolve jump-like solutions

The RGM for Nonlinear Parabolic IVPs

- Differences/advantages of the RGM and conventional (finite-difference/finite-element) methods:
  1. It is computationally easy to sample from $N(0, \Delta t)$
  2. Only costly operation each time step is to sort the remaining cohort of walkers by their position
  3. Can use ensemble averaging to reduce the variance of the solution
  4. Can choose to use either gradient "particles" of equal mass or allow the mass of each particle to change
  5. The RGM is adaptive, computational elements (pieces of the gradient) are created where $c'(u)$ is greatest, this is where sharp fronts appear and so fewer total computational elements are needed to spatially resolve jump-like solutions

The RGM for Nonlinear Parabolic IVPs

- Differences/advantages of the RGM and conventional (finite-difference/finite-element) methods:
  1. It is computationally easy to sample from $N(0, \Delta t)$
  2. Only costly operation each time step is to sort the remaining cohort of walkers by their position
  3. Can use ensemble averaging to reduce the variance of the solution
  4. Can choose to use either gradient “particles” of equal mass or allow the mass of each particle to change
  5. The RGM is adaptive, computational elements (pieces of the gradient) are created where $c'(u)$ is greatest, this is where sharp fronts appear and so fewer total computational elements are needed to spatially resolve jump-like solutions

The RGM for Nonlinear Parabolic IVPs

- Differences/advantages of the RGM and conventional (finite-difference/finite-element) methods:
  1. It is computationally easy to sample from $N(0, \Delta t)$
  2. Only costly operation each time step is to sort the remaining cohort of walkers by their position
  3. Can use ensemble averaging to reduce the variance of the solution
  4. Can choose to use either gradient “particles” of equal mass or allow the mass of each particle to change
  5. The RGM is adaptive, computational elements (pieces of the gradient) are created where $c'(u)$ is greatest, this is where sharp fronts appear and so fewer total computational elements are needed to spatially resolve jump-like solutions

The RGM in 2-Dimensions

- The RGM in 2D is the same as in 1D *except* for recovery from the gradient
- Write $u(\mathbf{x}) = G(\mathbf{x},\mathbf{y}) * \Delta u(\mathbf{y}) = \nabla_{\mathbf{n}} G(\mathbf{x},\mathbf{y}) * \nabla_{\mathbf{n}} u(\mathbf{y})$ (integration by parts)
- If $\nabla_{\mathbf{n}} u(\mathbf{y}) = \delta(\mathbf{y}-\mathbf{y}_0)$ then $u(\mathbf{x}) = \nabla_{\mathbf{n}} G(\mathbf{x},\mathbf{y}_0) = \frac{-1}{2\pi} \frac{(\mathbf{x}-\mathbf{y}_0)\cdot\mathbf{n}}{\|\mathbf{x}-\mathbf{y}_0\|^2}$
- Thus gradient recovery can be done via 2 n-body evaluations with charges n_1 and n_2 or with 1 Hilbert matrix application to the complex vector with $\mathbf{n}$
- Can (and do) use the Rokhlin-Greengard fast multipole algorithm for gradient recovery
- Initial gradient distribution comes from a detailed contour plot

The RGM in 2-Dimensions

- The RGM in 2D is the same as in 1D *except* for recovery from the gradient
- Write $u(\mathbf{x}) = G(\mathbf{x}, \mathbf{y}) * \Delta u(\mathbf{y}) = \nabla_{\mathbf{n}} G(\mathbf{x}, \mathbf{y}) * \nabla_{\mathbf{n}} u(\mathbf{y})$ (integration by parts)
- If $\nabla_{\mathbf{n}} u(\mathbf{y}) = \delta(\mathbf{y} - \mathbf{y}_0)$ then $u(\mathbf{x}) = \nabla_{\mathbf{n}} G(\mathbf{x}, \mathbf{y}_0) = \frac{-1}{2\pi} \frac{(\mathbf{x}-\mathbf{y}_0)\cdot\mathbf{n}}{\|\mathbf{x}-\mathbf{y}_0\|^2}$
- Thus gradient recovery can be done via 2 n-body evaluations with charges n_1 and n_2 or with 1 Hilbert matrix application to the complex vector with $\mathbf{n}$
- Can (and do) use the Rokhlin-Greengard fast multipole algorithm for gradient recovery
- Initial gradient distribution comes from a detailed contour plot

The RGM in 2-Dimensions

- The RGM in 2D is the same as in 1D *except* for recovery from the gradient
- Write $u(\mathbf{x}) = G(\mathbf{x}, \mathbf{y}) * \Delta u(\mathbf{y}) = \nabla_{\mathbf{n}} G(\mathbf{x}, \mathbf{y}) * \nabla_{\mathbf{n}} u(\mathbf{y})$ (integration by parts)
- If $\nabla_{\mathbf{n}} u(\mathbf{y}) = \delta(\mathbf{y} - \mathbf{y}_0)$ then $u(\mathbf{x}) = \nabla_{\mathbf{n}} G(\mathbf{x}, \mathbf{y}_0) = \frac{-1}{2\pi} \frac{(\mathbf{x}-\mathbf{y}_0)\cdot\mathbf{n}}{\|\mathbf{x}-\mathbf{y}_0\|^2}$
- Thus gradient recovery can be done via 2 n-body evaluations with charges n_1 and n_2 or with 1 Hilbert matrix application to the complex vector with $\mathbf{n}$
- Can (and do) use the Rokhlin-Greengard fast multipole algorithm for gradient recovery
- Initial gradient distribution comes from a detailed contour plot

The RGM in 2-Dimensions

- The RGM in 2D is the same as in 1D *except* for recovery from the gradient
- Write $u(\mathbf{x}) = G(\mathbf{x}, \mathbf{y}) * \Delta u(\mathbf{y}) = \nabla_{\mathbf{n}} G(\mathbf{x}, \mathbf{y}) * \nabla_{\mathbf{n}} u(\mathbf{y})$ (integration by parts)
- If $\nabla_{\mathbf{n}} u(\mathbf{y}) = \delta(\mathbf{y} - \mathbf{y}_0)$ then $u(\mathbf{x}) = \nabla_{\mathbf{n}} G(\mathbf{x}, \mathbf{y}_0) = \frac{-1}{2\pi} \frac{(\mathbf{x}-\mathbf{y}_0)\cdot\mathbf{n}}{\|\mathbf{x}-\mathbf{y}_0\|^2}$
- Thus gradient recovery can be done via 2 n-body evaluations with charges n_1 and n_2 or with 1 Hilbert matrix application to the complex vector with **n**
- Can (and do) use the Rokhlin-Greengard fast multipole algorithm for gradient recovery
- Initial gradient distribution comes from a detailed contour plot

The RGM in 2-Dimensions

- The RGM in 2D is the same as in 1D *except* for recovery from the gradient
- Write $u(\mathbf{x}) = G(\mathbf{x}, \mathbf{y}) * \Delta u(\mathbf{y}) = \nabla_{\mathbf{n}} G(\mathbf{x}, \mathbf{y}) * \nabla_{\mathbf{n}} u(\mathbf{y})$ (integration by parts)
- If $\nabla_{\mathbf{n}} u(\mathbf{y}) = \delta(\mathbf{y} - \mathbf{y}_0)$ then $u(\mathbf{x}) = \nabla_{\mathbf{n}} G(\mathbf{x}, \mathbf{y}_0) = \frac{-1}{2\pi} \frac{(\mathbf{x}-\mathbf{y}_0)\cdot\mathbf{n}}{\|\mathbf{x}-\mathbf{y}_0\|^2}$
- Thus gradient recovery can be done via 2 n-body evaluations with charges n_1 and n_2 or with 1 Hilbert matrix application to the complex vector with $\mathbf{n}$
- Can (and do) use the Rokhlin-Greengard fast multipole algorithm for gradient recovery
- Initial gradient distribution comes from a detailed contour plot

The RGM in 2-Dimensions

- The RGM in 2D is the same as in 1D *except* for recovery from the gradient
- Write $u(\mathbf{x}) = G(\mathbf{x}, \mathbf{y}) * \Delta u(\mathbf{y}) = \nabla_{\mathbf{n}} G(\mathbf{x}, \mathbf{y}) * \nabla_{\mathbf{n}} u(\mathbf{y})$ (integration by parts)
- If $\nabla_{\mathbf{n}} u(\mathbf{y}) = \delta(\mathbf{y} - \mathbf{y}_0)$ then $u(\mathbf{x}) = \nabla_{\mathbf{n}} G(\mathbf{x}, \mathbf{y}_0) = \frac{-1}{2\pi} \frac{(\mathbf{x}-\mathbf{y}_0)\cdot\mathbf{n}}{\|\mathbf{x}-\mathbf{y}_0\|^2}$
- Thus gradient recovery can be done via 2 n-body evaluations with charges n_1 and n_2 or with 1 Hilbert matrix application to the complex vector with $\mathbf{n}$
- Can (and do) use the Rokhlin-Greengard fast multipole algorithm for gradient recovery
- Initial gradient distribution comes from a detailed contour plot

Another MCM for a Nonlinear Parabolic PDE from Fluid Dynamics

- A model equation for fluid dynamics is Berger's equation in one-dimension, as an IVP:

$$\frac{\partial u}{\partial t} + u\frac{\partial u}{\partial x} = \frac{\epsilon}{2}\frac{\partial^2 u}{\partial x^2}, \quad u(x,0) = u_0(x)$$

- The substitution $\phi = e^{-\frac{1}{\epsilon}\int u\,dx} \iff u = -\epsilon\frac{\partial(\ln\phi)}{\partial x} = -\epsilon\frac{1}{\phi}\frac{\partial\phi}{\partial x}$ converts Berger's equation to the heat equation (Hopf, 1950):

$$\frac{\partial \phi}{\partial t} = \frac{\epsilon}{2}\frac{\partial^2 \phi}{\partial x^2}, \quad \phi(x,0) = e^{-\frac{1}{\epsilon}\int_0^x u_0(\xi)\,d\xi}$$

- Using the Feynman-Kac formula for the IVP for the heat equation one gets that $\phi(x,t) = E_x[e^{-\frac{1}{\epsilon}\int_0^{\sqrt{\epsilon}\beta(t)} u_0(\xi)\,d\xi}]$, which determines $u(x,t)$ via the above inversion formula

Another MCM for a Nonlinear Parabolic PDE from Fluid Dynamics

- A model equation for fluid dynamics is Berger's equation in one-dimension, as an IVP:

$$\frac{\partial u}{\partial t} + u\frac{\partial u}{\partial x} = \frac{\epsilon}{2}\frac{\partial^2 u}{\partial x^2}, \quad u(x,0) = u_0(x)$$

- The substitution $\phi = e^{-\frac{1}{\epsilon}\int u\,dx} \iff u = -\epsilon\frac{\partial(\ln\phi)}{\partial x} = -\epsilon\frac{1}{\phi}\frac{\partial\phi}{\partial x}$ converts Berger's equation to the heat equation (Hopf, 1950):

$$\frac{\partial \phi}{\partial t} = \frac{\epsilon}{2}\frac{\partial^2 \phi}{\partial x^2}, \quad \phi(x,0) = e^{-\frac{1}{\epsilon}\int_0^x u_0(\xi)\,d\xi}$$

- Using the Feynman-Kac formula for the IVP for the heat equation one gets that $\phi(x,t) = E_x[e^{-\frac{1}{\epsilon}\int_0^{\sqrt{\epsilon}\beta(t)} u_0(\xi)\,d\xi}]$, which determines $u(x,t)$ via the above inversion formula

Another MCM for a Nonlinear Parabolic PDE from Fluid Dynamics

- A model equation for fluid dynamics is Berger's equation in one-dimension, as an IVP:

$$\frac{\partial u}{\partial t} + u\frac{\partial u}{\partial x} = \frac{\epsilon}{2}\frac{\partial^2 u}{\partial x^2}, \quad u(x,0) = u_0(x)$$

- The substitution $\phi = e^{-\frac{1}{\epsilon}\int u\,dx} \iff u = -\epsilon\frac{\partial(\ln\phi)}{\partial x} = -\epsilon\frac{1}{\phi}\frac{\partial\phi}{\partial x}$ converts Berger's equation to the heat equation (Hopf, 1950):

$$\frac{\partial \phi}{\partial t} = \frac{\epsilon}{2}\frac{\partial^2 \phi}{\partial x^2}, \quad \phi(x,0) = e^{-\frac{1}{\epsilon}\int_0^x u_0(\xi)\,d\xi}$$

- Using the Feynman-Kac formula for the IVP for the heat equation one gets that $\phi(x,t) = E_x[e^{-\frac{1}{\epsilon}\int_0^{\sqrt{\epsilon}\beta(t)} u_0(\xi)\,d\xi}]$, which determines $u(x,t)$ via the above inversion formula

MCMs for the Schrödinger Equation (Brief)

- The Schrödinger equation is given by:

$$-i\frac{\hbar}{2\pi}u_\tau = \Delta u(x) - V(x)u(x)$$

- Can replace $-i\frac{\hbar}{2\pi}\tau = t$ (imaginary time), to give a linear parabolic PDE
- Usually $x \in \mathbb{R}^{3n}$ where there are n quantum particles, thus we are in a very high-dimensional case
- As in the above, use walks, killing, and importance sampling
- Interesting variants:
  1. *Diffusion Monte Carlo*
  2. *Greens Function Monte Carlo*
  3. *Path Integral Monte Carlo*

MCMs for the Schrödinger Equation (Brief)

- The Schrödinger equation is given by:

$$-i\frac{\hbar}{2\pi}u_\tau = \Delta u(x) - V(x)u(x)$$

- Can replace $-i\frac{\hbar}{2\pi}\tau = t$ (imaginary time), to give a linear parabolic PDE
- Usually $x \in \mathbb{R}^{3n}$ where there are n quantum particles, thus we are in a very high-dimensional case
- As in the above, use walks, killing, and importance sampling
- Interesting variants:
  1. *Diffusion Monte Carlo*
  2. *Greens Function Monte Carlo*
  3. *Path Integral Monte Carlo*

MCMs for the Schrödinger Equation (Brief)

- The Schrödinger equation is given by:

$$-i\frac{\hbar}{2\pi}u_\tau = \Delta u(x) - V(x)u(x)$$

- Can replace $-i\frac{\hbar}{2\pi}\tau = t$ (imaginary time), to give a linear parabolic PDE
- Usually $x \in \mathbb{R}^{3n}$ where there are n quantum particles, thus we are in a very high-dimensional case
- As in the above, use walks, killing, and importance sampling
- Interesting variants:
  1. *Diffusion Monte Carlo*
  2. *Greens Function Monte Carlo*
  3. *Path Integral Monte Carlo*

MCMs for the Schrödinger Equation (Brief)

- The Schrödinger equation is given by:

$$-i\frac{\hbar}{2\pi}u_\tau = \Delta u(x) - V(x)u(x)$$

- Can replace $-i\frac{\hbar}{2\pi}\tau = t$ (imaginary time), to give a linear parabolic PDE
- Usually $x \in \mathbb{R}^{3n}$ where there are n quantum particles, thus we are in a very high-dimensional case
- As in the above, use walks, killing, and importance sampling
- Interesting variants:
  1. *Diffusion Monte Carlo*
  2. *Greens Function Monte Carlo*
  3. *Path Integral Monte Carlo*

MCMs for the Schrödinger Equation (Brief)

- The Schrödinger equation is given by:

$$-i\frac{\hbar}{2\pi}u_\tau = \Delta u(x) - V(x)u(x)$$

- Can replace $-i\frac{\hbar}{2\pi}\tau = t$ (imaginary time), to give a linear parabolic PDE
- Usually $x \in \mathbb{R}^{3n}$ where there are n quantum particles, thus we are in a very high-dimensional case
- As in the above, use walks, killing, and importance sampling
- Interesting variants:
  1. *Diffusion Monte Carlo*
  2. *Greens Function Monte Carlo*
  3. *Path Integral Monte Carlo*

MCMs for the Schrödinger Equation (Brief)

- The Schrödinger equation is given by:

$$-i\frac{\hbar}{2\pi}u_\tau = \Delta u(x) - V(x)u(x)$$

- Can replace $-i\frac{\hbar}{2\pi}\tau = t$ (imaginary time), to give a linear parabolic PDE
- Usually $x \in \mathbb{R}^{3n}$ where there are n quantum particles, thus we are in a very high-dimensional case
- As in the above, use walks, killing, and importance sampling
- Interesting variants:
  1. *Diffusion Monte Carlo*
  2. *Greens Function Monte Carlo*
  3. *Path Integral Monte Carlo*

MCMs for the Schrödinger Equation (Brief)

- The Schrödinger equation is given by:

$$-i\frac{\hbar}{2\pi}u_\tau = \Delta u(x) - V(x)u(x)$$

- Can replace $-i\frac{\hbar}{2\pi}\tau = t$ (imaginary time), to give a linear parabolic PDE
- Usually $x \in \mathbb{R}^{3n}$ where there are n quantum particles, thus we are in a very high-dimensional case
- As in the above, use walks, killing, and importance sampling
- Interesting variants:
  1. *Diffusion Monte Carlo*
  2. *Greens Function Monte Carlo*
  3. *Path Integral Monte Carlo*

MCMs for the Schrödinger Equation (Brief)

- The Schrödinger equation is given by:

$$-i\frac{\hbar}{2\pi}u_\tau = \Delta u(x) - V(x)u(x)$$

- Can replace $-i\frac{\hbar}{2\pi}\tau = t$ (imaginary time), to give a linear parabolic PDE
- Usually $x \in \mathbb{R}^{3n}$ where there are n quantum particles, thus we are in a very high-dimensional case
- As in the above, use walks, killing, and importance sampling
- Interesting variants:
  1. *Diffusion Monte Carlo*
  2. *Greens Function Monte Carlo*
  3. *Path Integral Monte Carlo*

Another MCM for a Nonlinear Parabolic PDE from Fluid Dynamics

- Note when $L = \frac{\epsilon}{2}\Delta$ the scaling $x \to \sqrt{\epsilon}x$ converts L into the pure Laplacian
- Thus can sample from L with $\sqrt{\epsilon}\beta(\cdot)$ as scaled sample paths instead of ordinary Brownian motion, this is Brownian scaling
- Unlike the reaction-diffusion problems solved by the RGM this equation is an hyperbolic conservation law: $u_t + (u^2/2)_x = \frac{\epsilon}{2}u_{xx}$, these equations often have jumps that sharpen into discontinuous shocks as $\epsilon \to 0$
- The MCM for Berger's equation derived above was only possible because the Hopf-Cole transformation could be used to convert this problem to the heat equation

Another MCM for a Nonlinear Parabolic PDE from Fluid Dynamics

- Note when $L = \frac{\epsilon}{2}\Delta$ the scaling $x \to \sqrt{\epsilon}x$ converts L into the pure Laplacian
- Thus can sample from L with $\sqrt{\epsilon}\beta(\cdot)$ as scaled sample paths instead of ordinary Brownian motion, this is Brownian scaling
- Unlike the reaction-diffusion problems solved by the RGM this equation is an hyperbolic conservation law: $u_t + (u^2/2)_x = \frac{\epsilon}{2}u_{xx}$, these equations often have jumps that sharpen into discontinuous shocks as $\epsilon \to 0$
- The MCM for Berger's equation derived above was only possible because the Hopf-Cole transformation could be used to convert this problem to the heat equation

Another MCM for a Nonlinear Parabolic PDE from Fluid Dynamics

- Note when $L = \frac{\epsilon}{2}\Delta$ the scaling $x \to \sqrt{\epsilon}x$ converts L into the pure Laplacian
- Thus can sample from L with $\sqrt{\epsilon}\beta(\cdot)$ as scaled sample paths instead of ordinary Brownian motion, this is Brownian scaling
- Unlike the reaction-diffusion problems solved by the RGM this equation is an hyperbolic conservation law: $u_t + (u^2/2)_x = \frac{\epsilon}{2}u_{xx}$, these equations often have jumps that sharpen into discontinuous shocks as $\epsilon \to 0$
- The MCM for Berger's equation derived above was only possible because the Hopf-Cole transformation could be used to convert this problem to the heat equation

Another MCM for a Nonlinear Parabolic PDE from Fluid Dynamics

- Note when $L = \frac{\epsilon}{2}\Delta$ the scaling $x \to \sqrt{\epsilon}x$ converts L into the pure Laplacian
- Thus can sample from L with $\sqrt{\epsilon}\beta(\cdot)$ as scaled sample paths instead of ordinary Brownian motion, this is Brownian scaling
- Unlike the reaction-diffusion problems solved by the RGM this equation is an hyperbolic conservation law: $u_t + (u^2/2)_x = \frac{\epsilon}{2}u_{xx}$, these equations often have jumps that sharpen into discontinuous shocks as $\epsilon \to 0$
- The MCM for Berger's equation derived above was only possible because the Hopf-Cole transformation could be used to convert this problem to the heat equation

Two-Dimensional Incompressible Fluid Dynamics

- The equations of two dimensional incompressible fluid dynamics are:

$$\frac{\partial \mathbf{u}}{\partial t} + (\mathbf{u} \cdot \nabla)\mathbf{u} = -\frac{1}{\rho}\nabla p + \gamma \Delta \mathbf{u}$$
$$\nabla \cdot \mathbf{u} = 0, \quad \mathbf{u} = (u, v)^T \tag{4.1a}$$

1. Inviscid fluid: $\gamma = 0 \implies$ Euler equations
2. Viscous fluid: $\gamma \neq 0 \implies$ Navier-Stokes equations

- Since $\nabla \cdot \mathbf{u} = 0$ there is a stream function ψ s.t. $\mathbf{u} = (-\frac{\partial \psi}{\partial y}, \frac{\partial \psi}{\partial x})^T$, with the vorticity $\xi = \nabla \times \mathbf{u}$ we can rewrite as:

$$\frac{\partial \xi}{\partial t} + (\mathbf{u} \cdot \nabla)\xi = \gamma \Delta \xi, \quad \Delta \psi = -\xi \tag{4.1b}$$

Two-Dimensional Incompressible Fluid Dynamics

- The equations of two dimensional incompressible fluid dynamics are:

$$\frac{\partial \mathbf{u}}{\partial t} + (\mathbf{u} \cdot \nabla)\mathbf{u} = -\frac{1}{\rho}\nabla p + \gamma \Delta \mathbf{u}$$
$$\nabla \cdot \mathbf{u} = 0, \quad \mathbf{u} = (u, v)^T \tag{4.1a}$$

1. Inviscid fluid: $\gamma = 0 \implies$ Euler equations
2. Viscous fluid: $\gamma \neq 0 \implies$ Navier-Stokes equations

- Since $\nabla \cdot \mathbf{u} = 0$ there is a stream function ψ s.t. $\mathbf{u} = (-\frac{\partial \psi}{\partial y}, \frac{\partial \psi}{\partial x})^T$, with the vorticity $\xi = \nabla \times \mathbf{u}$ we can rewrite as:

$$\frac{\partial \xi}{\partial t} + (\mathbf{u} \cdot \nabla)\xi = \gamma \Delta \xi, \quad \Delta \psi = -\xi \tag{4.1b}$$

Two-Dimensional Incompressible Fluid Dynamics

- The equations of two dimensional incompressible fluid dynamics are:

$$\frac{\partial \mathbf{u}}{\partial t} + (\mathbf{u} \cdot \nabla)\mathbf{u} = -\frac{1}{\rho}\nabla p + \gamma \Delta \mathbf{u}$$

$$\nabla \cdot \mathbf{u} = 0, \quad \mathbf{u} = (u, v)^T \tag{4.1a}$$

1. Inviscid fluid: $\gamma = 0 \implies$ Euler equations
2. Viscous fluid: $\gamma \neq 0 \implies$ Navier-Stokes equations

- Since $\nabla \cdot \mathbf{u} = 0$ there is a stream function ψ s.t. $\mathbf{u} = (-\frac{\partial \psi}{\partial y}, \frac{\partial \psi}{\partial x})^T$, with the vorticity $\xi = \nabla \times \mathbf{u}$ we can rewrite as:

$$\frac{\partial \xi}{\partial t} + (\mathbf{u} \cdot \nabla)\xi = \gamma \Delta \xi, \quad \Delta \psi = -\xi \tag{4.1b}$$

Two-Dimensional Incompressible Fluid Dynamics

- The equations of two dimensional incompressible fluid dynamics are:

$$\frac{\partial \mathbf{u}}{\partial t} + (\mathbf{u}\cdot\nabla)\mathbf{u} = -\frac{1}{\rho}\nabla p + \gamma\Delta\mathbf{u}$$
$$\nabla\cdot\mathbf{u} = 0, \quad \mathbf{u} = (u,v)^T \tag{4.1a}$$

1. Inviscid fluid: $\gamma = 0 \implies$ Euler equations
2. Viscous fluid: $\gamma \neq 0 \implies$ Navier-Stokes equations

- Since $\nabla\cdot\mathbf{u} = 0$ there is a stream function ψ s.t. $\mathbf{u} = (-\frac{\partial\psi}{\partial y}, \frac{\partial\psi}{\partial x})^T$, with the vorticity $\xi = \nabla\times\mathbf{u}$ we can rewrite as:

$$\frac{\partial \xi}{\partial t} + (\mathbf{u}\cdot\nabla)\xi = \gamma\Delta\xi, \quad \Delta\psi = -\xi \tag{4.1b}$$

Two-Dimensional Incompressible Fluid Dynamics

- Recall the material (or total) derivative of z: $\frac{Dz}{Dt} = \frac{\partial z}{\partial t} + (\mathbf{u} \cdot \nabla)z$, this is the time rate of change in a quantity that is being advected in a fluid with velocity $\mathbf{u}$
- We can rewrite the vorticity above formulation as:

$$\frac{D\xi}{Dt} = \gamma \Delta \xi, \quad \Delta \psi = -\xi \tag{4.1c}$$

- If we neglect boundary conditions, can represent $\xi = \sum_{n=1}^{N} \xi_i \delta(\mathbf{x} - \mathbf{x}_i)$ (spike discretization of a gradient), since in 2D the fundamental solution of the Poisson equation is $\Delta^{-1}\delta(\mathbf{x} - \mathbf{x}_i) = -\frac{1}{2\pi} \log |\mathbf{x} - \mathbf{x}_i|$ we have $\psi = \frac{1}{2\pi} \sum_{n=1}^{N} \xi_i \log |\mathbf{x} - \mathbf{x}_i|$
- Can prove that if ξ is a sum of delta functions at some time t_0, it remains so for all time $t \geq t_0$

Two-Dimensional Incompressible Fluid Dynamics

- Recall the material (or total) derivative of z: $\frac{Dz}{Dt} = \frac{\partial z}{\partial t} + (\mathbf{u} \cdot \nabla) z$, this is the time rate of change in a quantity that is being advected in a fluid with velocity **u**
- We can rewrite the vorticity above formulation as:

$$\frac{D\xi}{Dt} = \gamma \Delta \xi, \quad \Delta \psi = -\xi \tag{4.1c}$$

- If we neglect boundary conditions, can represent $\xi = \sum_{n=1}^{N} \xi_i \delta(\mathbf{x} - \mathbf{x}_i)$ (spike discretization of a gradient), since in 2D the fundamental solution of the Poisson equation is $\Delta^{-1}\delta(\mathbf{x} - \mathbf{x}_i) = -\frac{1}{2\pi} \log |\mathbf{x} - \mathbf{x}_i|$ we have $\psi = \frac{1}{2\pi} \sum_{n=1}^{N} \xi_i \log |\mathbf{x} - \mathbf{x}_i|$
- Can prove that if ξ is a sum of delta functions at some time t_0, it remains so for all time $t \geq t_0$

Two-Dimensional Incompressible Fluid Dynamics

- Recall the material (or total) derivative of z: $\frac{Dz}{Dt} = \frac{\partial z}{\partial t} + (\mathbf{u} \cdot \nabla) z$, this is the time rate of change in a quantity that is being advected in a fluid with velocity $\mathbf{u}$
- We can rewrite the vorticity above formulation as:

$$\frac{D\xi}{Dt} = \gamma \Delta \xi, \quad \Delta \psi = -\xi \tag{4.1c}$$

- If we neglect boundary conditions, can represent $\xi = \sum_{n=1}^{N} \xi_i \delta(\mathbf{x} - \mathbf{x}_i)$ (spike discretization of a gradient), since in 2D the fundamental solution of the Poisson equation is $\Delta^{-1} \delta(\mathbf{x} - \mathbf{x}_i) = -\frac{1}{2\pi} \log |\mathbf{x} - \mathbf{x}_i|$ we have $\psi = \frac{1}{2\pi} \sum_{n=1}^{N} \xi_i \log |\mathbf{x} - \mathbf{x}_i|$
- Can prove that if ξ is a sum of delta functions at some time t_0, it remains so for all time $t \geq t_0$

Two-Dimensional Incompressible Fluid Dynamics

- Recall the material (or total) derivative of z: $\frac{Dz}{Dt} = \frac{\partial z}{\partial t} + (\mathbf{u} \cdot \nabla) z$, this is the time rate of change in a quantity that is being advected in a fluid with velocity $\mathbf{u}$
- We can rewrite the vorticity above formulation as:

$$\frac{D\xi}{Dt} = \gamma \Delta \xi, \quad \Delta \psi = -\xi \tag{4.1c}$$

- If we neglect boundary conditions, can represent $\xi = \sum_{n=1}^{N} \xi_i \delta(\mathbf{x} - \mathbf{x}_i)$ (spike discretization of a gradient), since in 2D the fundamental solution of the Poisson equation is $\Delta^{-1}\delta(\mathbf{x} - \mathbf{x}_i) = -\frac{1}{2\pi} \log|\mathbf{x} - \mathbf{x}_i|$ we have $\psi = \frac{1}{2\pi} \sum_{n=1}^{N} \xi_i \log|\mathbf{x} - \mathbf{x}_i|$
- Can prove that if ξ is a sum of delta functions at some time t_0, it remains so for all time $t \geq t_0$

The Vortex Method for the Incompressible Euler's Equation

- These observations on the vortex form of the Euler equations help extend ideas of a method first proposed by Chorin (Chorin, 1971):
  1. Represent $\xi(\mathbf{x}, t) = \sum_{i=1}^{N} \xi_i \delta(\mathbf{x} - \mathbf{x}_i)$, and so $\psi = \frac{1}{2\pi} \sum_{n=1}^{N} \xi_i \log |\mathbf{x} - \mathbf{x}_i|$
  2. Move each of the vortex "blobs" via the ODEs for the x and y components of ξ_i:

$$\frac{dx_i}{dt} = -\frac{1}{2\pi} \sum_{j \neq i} \xi_j \frac{\partial \log |\mathbf{x}_i - \mathbf{x}_j|}{\partial y}, \; i = 1, \dots, N$$

$$\frac{dy_i}{dt} = \frac{1}{2\pi} \sum_{j \neq i} \xi_j \frac{\partial \log |\mathbf{x}_i - \mathbf{x}_j|}{\partial x}, \; i = 1, \dots, N$$

- Note, no time step size constraint is *a priori* imposed but depends on the choice of numerical ODE scheme

The Vortex Method for the Incompressible Euler's Equation

- These observations on the vortex form of the Euler equations help extend ideas of a method first proposed by Chorin (Chorin, 1971):
    1. Represent $\xi(\mathbf{x}, t) = \sum_{i=1}^{N} \xi_i \delta(\mathbf{x} - \mathbf{x}_i)$, and so $\psi = \frac{1}{2\pi} \sum_{n=1}^{N} \xi_i \log |\mathbf{x} - \mathbf{x}_i|$
    2. Move each of the vortex "blobs" via the ODEs for the x and y components of ξ_i:

$$\begin{aligned}
\frac{dx_i}{dt} &= -\frac{1}{2\pi} \sum_{j \neq i} \xi_j \frac{\partial \log |\mathbf{x}_i - \mathbf{x}_j|}{\partial y}, \ i = 1, \dots, N \\
\frac{dy_i}{dt} &= \frac{1}{2\pi} \sum_{j \neq i} \xi_j \frac{\partial \log |\mathbf{x}_i - \mathbf{x}_j|}{\partial x}, \ i = 1, \dots, N
\end{aligned}$$

- Note, no time step size constraint is *a priori* imposed but depends on the choice of numerical ODE scheme

The Vortex Method for the Incompressible Euler's Equation

- These observations on the vortex form of the Euler equations help extend ideas of a method first proposed by Chorin (Chorin, 1971):
  1. Represent $\xi(\mathbf{x},t) = \sum_{i=1}^{N} \xi_i \delta(\mathbf{x}-\mathbf{x}_i)$, and so $\psi = \frac{1}{2\pi}\sum_{n=1}^{N} \xi_i \log|\mathbf{x}-\mathbf{x}_i|$
  2. Move each of the vortex "blobs" via the ODEs for the x and y components of ξ_i:

$$\begin{aligned} \frac{dx_i}{dt} &= -\frac{1}{2\pi}\sum_{j\neq i} \xi_j \frac{\partial \log|\mathbf{x}_i-\mathbf{x}_j|}{\partial y}, \; i=1,\dots,N \\ \frac{dy_i}{dt} &= \frac{1}{2\pi}\sum_{j\neq i} \xi_j \frac{\partial \log|\mathbf{x}_i-\mathbf{x}_j|}{\partial x}, \; i=1,\dots,N \end{aligned}$$

- Note, no time step size constraint is *a priori* imposed but depends on the choice of numerical ODE scheme

The Vortex Method for the Incompressible Euler's Equation

- These observations on the vortex form of the Euler equations help extend ideas of a method first proposed by Chorin (Chorin, 1971):
  1. Represent $\xi(\mathbf{x}, t) = \sum_{i=1}^{N} \xi_i \delta(\mathbf{x} - \mathbf{x}_i)$, and so $\psi = \frac{1}{2\pi} \sum_{n=1}^{N} \xi_i \log |\mathbf{x} - \mathbf{x}_i|$
  2. Move each of the vortex "blobs" via the ODEs for the x and y components of ξ_i:

$$\begin{aligned}\frac{dx_i}{dt} &= -\frac{1}{2\pi} \sum_{j \neq i} \xi_j \frac{\partial \log |\mathbf{x}_i - \mathbf{x}_j|}{\partial y}, \ i = 1, \dots, N \\ \frac{dy_i}{dt} &= \frac{1}{2\pi} \sum_{j \neq i} \xi_j \frac{\partial \log |\mathbf{x}_i - \mathbf{x}_j|}{\partial x}, \ i = 1, \dots, N\end{aligned}$$

- Note, no time step size constraint is *a priori* imposed but depends on the choice of numerical ODE scheme

The Vortex Method for the Incompressible Euler's Equation

- This is not a MCM but only a method for converting the 2D Euler equations (PDEs) to a system of ODEs which are mathematically equivalent to the N-body problem (gravitation, particle dynamics)
- It is common to use other functional forms for the vortex "blobs," and hence the stream function is changed, in our case the stream "blob" function is $\psi(\mathbf{x}) = \sum_{n=1}^{N} \xi_i \psi^0(\mathbf{x} - \mathbf{x}_i)$ where $\psi^0(\mathbf{x} - \mathbf{x}_i) = \frac{1}{2\pi} \log |\mathbf{x} - \mathbf{x}_i|$
- Other choices of stream "blob" functions are $\psi^0(\mathbf{x})$ s.t. $\psi^0(\mathbf{x}) \sim \frac{1}{2\pi} \log(\mathbf{x})$ for $|\mathbf{x}| \gg 0$, and $\psi^0(\mathbf{x}) \to 0$ as $|\mathbf{x}| \to 0$

The Vortex Method for the Incompressible Euler's Equation

- This is not a MCM but only a method for converting the 2D Euler equations (PDEs) to a system of ODEs which are mathematically equivalent to the N-body problem (gravitation, particle dynamics)
- It is common to use other functional forms for the vortex "blobs," and hence the stream function is changed, in our case the stream "blob" function is $\psi(\mathbf{x}) = \sum_{n=1}^{N} \xi_i \psi^0(\mathbf{x} - \mathbf{x}_i)$ where $\psi^0(\mathbf{x} - \mathbf{x}_i) = \frac{1}{2\pi} \log |\mathbf{x} - \mathbf{x}_i|$
- Other choices of stream "blob" functions are $\psi^0(\mathbf{x})$ s.t. $\psi^0(\mathbf{x}) \sim \frac{1}{2\pi} \log(\mathbf{x})$ for $|\mathbf{x}| \gg 0$, and $\psi^0(\mathbf{x}) \to 0$ as $|\mathbf{x}| \to 0$

The Vortex Method for the Incompressible Euler's Equation

- This is not a MCM but only a method for converting the 2D Euler equations (PDEs) to a system of ODEs which are mathematically equivalent to the N-body problem (gravitation, particle dynamics)
- It is common to use other functional forms for the vortex "blobs," and hence the stream function is changed, in our case the stream "blob" function is $\psi(\mathbf{x}) = \sum_{n=1}^{N} \xi_i \psi^0(\mathbf{x} - \mathbf{x}_i)$ where $\psi^0(\mathbf{x} - \mathbf{x}_i) = \frac{1}{2\pi} \log |\mathbf{x} - \mathbf{x}_i|$
- Other choices of stream "blob" functions are $\psi^0(\mathbf{x})$ s.t. $\psi^0(\mathbf{x}) \sim \frac{1}{2\pi} \log(\mathbf{x})$ for $|\mathbf{x}| \gg 0$, and $\psi^0(\mathbf{x}) \to 0$ as $|\mathbf{x}| \to 0$

Chorin's Random Vortex Method for the Incompressible Navier-Stokes Equations

- When $\gamma \neq 0$, Euler $\implies$ Navier-Stokes and (4.1b) can be thought of as an equation where vorticity is advected via fluid (the l.h.s. material derivative), and diffuses due to viscosity (the r.h.s term)
- As above, diffusion can be sampled by moving diffusing particles via the Gaussian (fundamental soln. of the diffusion eqn.)
- Thus we can extend the inviscid random vortex method (RVM) to Navier-Stokes through a fractional step approach, do the vorticity advection via the inviscid RVM and then treat the diffusion of vorticity equation by moving the vortex randomly
- This addition for diffusion of vorticity make Chorin's RGM into a MCM
- Note that the RGM is also fractional step, but both steps are MCMs

Chorin's Random Vortex Method for the Incompressible Navier-Stokes Equations

- When $\gamma \neq 0$, Euler $\implies$ Navier-Stokes and (4.1b) can be thought of as an equation where vorticity is advected via fluid (the l.h.s. material derivative), and diffuses due to viscosity (the r.h.s term)
- As above, diffusion can be sampled by moving diffusing particles via the Gaussian (fundamental soln. of the diffusion eqn.)
- Thus we can extend the inviscid random vortex method (RVM) to Navier-Stokes through a fractional step approach, do the vorticity advection via the inviscid RVM and then treat the diffusion of vorticity equation by moving the vortex randomly
- This addition for diffusion of vorticity make Chorin's RGM into a MCM
- Note that the RGM is also fractional step, but both steps are MCMs

Chorin's Random Vortex Method for the Incompressible Navier-Stokes Equations

- When $\gamma \neq 0$, Euler $\implies$ Navier-Stokes and (4.1b) can be thought of as an equation where vorticity is advected via fluid (the l.h.s. material derivative), and diffuses due to viscosity (the r.h.s term)
- As above, diffusion can be sampled by moving diffusing particles via the Gaussian (fundamental soln. of the diffusion eqn.)
- Thus we can extend the inviscid random vortex method (RVM) to Navier-Stokes through a fractional step approach, do the vorticity advection via the inviscid RVM and then treat the diffusion of vorticity equation by moving the vortex randomly
- This addition for diffusion of vorticity make Chorin's RGM into a MCM
- Note that the RGM is also fractional step, but both steps are MCMs

Chorin's Random Vortex Method for the Incompressible Navier-Stokes Equations

- When $\gamma \neq 0$, Euler $\implies$ Navier-Stokes and (4.1b) can be thought of as an equation where vorticity is advected via fluid (the l.h.s. material derivative), and diffuses due to viscosity (the r.h.s term)
- As above, diffusion can be sampled by moving diffusing particles via the Gaussian (fundamental soln. of the diffusion eqn.)
- Thus we can extend the inviscid random vortex method (RVM) to Navier-Stokes through a fractional step approach, do the vorticity advection via the inviscid RVM and then treat the diffusion of vorticity equation by moving the vortex randomly
- This addition for diffusion of vorticity make Chorin's RGM into a MCM
- Note that the RGM is also fractional step, but both steps are MCMs

Chorin's Random Vortex Method for the Incompressible Navier-Stokes Equations

- When $\gamma \neq 0$, Euler $\implies$ Navier-Stokes and (4.1b) can be thought of as an equation where vorticity is advected via fluid (the l.h.s. material derivative), and diffuses due to viscosity (the r.h.s term)
- As above, diffusion can be sampled by moving diffusing particles via the Gaussian (fundamental soln. of the diffusion eqn.)
- Thus we can extend the inviscid random vortex method (RVM) to Navier-Stokes through a fractional step approach, do the vorticity advection via the inviscid RVM and then treat the diffusion of vorticity equation by moving the vortex randomly
- This addition for diffusion of vorticity make Chorin's RGM into a MCM
- Note that the RGM is also fractional step, but both steps are MCMs

Chorin's RVM for the Incompressible Navier-Stokes Equations

- Chorin's RVM for 2D Navier-Stokes:
  1. Using stream function "blobs" represent $\psi(\mathbf{x}) = \sum_{n=1}^{N} \xi_i \psi^0(\mathbf{x} - \mathbf{x}_i)$
  2. Advect each of the vortex "blobs" Δt forward in time via the ODEs for the x and y components of ξ_i, use your favorite (stable) ODE method:
  $$\frac{dx_i}{dt} = -\sum_{j \neq i} \xi_j \frac{\partial \psi^0(\mathbf{x}_i - \mathbf{x}_j)}{\partial y}, \; i = 1, \ldots, N$$
  $$\frac{dy_i}{dt} = \sum_{j \neq i} \xi_j \frac{\partial \psi^0(\mathbf{x}_i - \mathbf{x}_j)}{\partial x}, \; i = 1, \ldots, N$$
  3. Update each $\mathbf{x}_i(t + \Delta t) \to \mathbf{x}_i(t + \Delta t) + \eta_i$ where η_i is $N(0, 2\Delta t \gamma)$

Chorin's RVM for the Incompressible Navier-Stokes Equations

- Chorin's RVM for 2D Navier-Stokes:
  1. Using stream function "blobs" represent $\psi(\mathbf{x}) = \sum_{n=1}^{N} \xi_i \psi^0(\mathbf{x} - \mathbf{x}_i)$
  2. Advect each of the vortex "blobs" Δt forward in time via the ODEs for the x and y components of ξ_i, use your favorite (stable) ODE method:
  $$\frac{dx_i}{dt} = -\sum_{j \neq i} \xi_j \frac{\partial \psi^0(\mathbf{x}_i - \mathbf{x}_j)}{\partial y}, \; i = 1, \ldots, N$$
  $$\frac{dy_i}{dt} = \sum_{j \neq i} \xi_j \frac{\partial \psi^0(\mathbf{x}_i - \mathbf{x}_j)}{\partial x}, \; i = 1, \ldots, N$$
  3. Update each $\mathbf{x}_i(t + \Delta t) \to \mathbf{x}_i(t + \Delta t) + \eta_i$ where η_i is $N(0, 2\Delta t \gamma)$

Chorin's RVM for the Incompressible Navier-Stokes Equations

- Chorin's RVM for 2D Navier-Stokes:
  1. Using stream function "blobs" represent $\psi(\mathbf{x}) = \sum_{n=1}^{N} \xi_i \psi^0(\mathbf{x} - \mathbf{x}_i)$
  2. Advect each of the vortex "blobs" Δt forward in time via the ODEs for the x and y components of ξ_i, use your favorite (stable) ODE method:
  $$\frac{dx_i}{dt} = -\sum_{j \neq i} \xi_j \frac{\partial \psi^0(\mathbf{x}_i - \mathbf{x}_j)}{\partial y}, \; i = 1, \dots, N$$
  $$\frac{dy_i}{dt} = \sum_{j \neq i} \xi_j \frac{\partial \psi^0(\mathbf{x}_i - \mathbf{x}_j)}{\partial x}, \; i = 1, \dots, N$$
  3. Update each $\mathbf{x}_i(t + \Delta t) \rightarrow \mathbf{x}_i(t + \Delta t) + \eta_i$ where η_i is $N(0, 2\Delta t \gamma)$

Chorin's RVM for the Incompressible Navier-Stokes Equations

- Chorin's RVM for 2D Navier-Stokes:
  1. Using stream function "blobs" represent $\psi(\mathbf{x}) = \sum_{n=1}^{N} \xi_i \psi^0(\mathbf{x} - \mathbf{x}_i)$
  2. Advect each of the vortex "blobs" Δt forward in time via the ODEs for the x and y components of ξ_i, use your favorite (stable) ODE method:
  $$\frac{dx_i}{dt} = -\sum_{j \neq i} \xi_j \frac{\partial \psi^0(\mathbf{x}_i - \mathbf{x}_j)}{\partial y}, \; i = 1, \ldots, N$$
  $$\frac{dy_i}{dt} = \sum_{j \neq i} \xi_j \frac{\partial \psi^0(\mathbf{x}_i - \mathbf{x}_j)}{\partial x}, \; i = 1, \ldots, N$$
  3. Update each $\mathbf{x}_i(t + \Delta t) \rightarrow \mathbf{x}_i(t + \Delta t) + \eta_i$ where η_i is $N(0, 2\Delta t \gamma)$

Chorin's RVM for the Incompressible Navier-Stokes Equations

- Shortcoming of method is irrotational flows, i.e. $\nabla \times \mathbf{u} = 0$, i.e. $\xi = 0$
  1. Irrotational flow implies $\exists \phi$ s.t. $\mathbf{u} = \nabla \phi$, i.e. $\mathbf{u}$ is a potential flow
  2. 2D potential flows reduce to solving $\nabla \cdot \mathbf{u} = \Delta \phi = 0$ with boundary conditions
  3. Can use this to enforce boundary conditions, as can add a potential flow that corrects for the rotational flow at the boundaries
- As with RGM the RVM is adaptive in that regions of high vorticity (rotation) get a high density of vortex "blobs"

Chorin's RVM for the Incompressible Navier-Stokes Equations

- Shortcoming of method is irrotational flows, i.e. $\nabla \times \mathbf{u} = 0$, i.e. $\xi = 0$
  1. Irrotational flow implies $\exists \phi$ s.t. $\mathbf{u} = \nabla \phi$, i.e. $\mathbf{u}$ is a potential flow
  2. 2D potential flows reduce to solving $\nabla \cdot \mathbf{u} = \Delta \phi = 0$ with boundary conditions
  3. Can use this to enforce boundary conditions, as can add a potential flow that corrects for the rotational flow at the boundaries
- As with RGM the RVM is adaptive in that regions of high vorticity (rotation) get a high density of vortex "blobs"

Chorin's RVM for the Incompressible Navier-Stokes Equations

- Shortcoming of method is irrotational flows, i.e. $\nabla \times \mathbf{u} = 0$, i.e. $\xi = 0$
  1. Irrotational flow implies $\exists \phi$ s.t. $\mathbf{u} = \nabla \phi$, i.e. **u** is a potential flow
  2. 2D potential flows reduce to solving $\nabla \cdot \mathbf{u} = \Delta \phi = 0$ with boundary conditions
  3. Can use this to enforce boundary conditions, as can add a potential flow that corrects for the rotational flow at the boundaries
- As with RGM the RVM is adaptive in that regions of high vorticity (rotation) get a high density of vortex "blobs"

Chorin's RVM for the Incompressible Navier-Stokes Equations

- Shortcoming of method is irrotational flows, i.e. $\nabla \times \mathbf{u} = 0$, i.e. $\xi = 0$
  1. Irrotational flow implies $\exists \phi$ s.t. $\mathbf{u} = \nabla \phi$, i.e. $\mathbf{u}$ is a potential flow
  2. 2D potential flows reduce to solving $\nabla \cdot \mathbf{u} = \Delta \phi = 0$ with boundary conditions
  3. Can use this to enforce boundary conditions, as can add a potential flow that corrects for the rotational flow at the boundaries
- As with RGM the RVM is adaptive in that regions of high vorticity (rotation) get a high density of vortex "blobs"

Chorin's RVM for the Incompressible Navier-Stokes Equations

- Shortcoming of method is irrotational flows, i.e. $\nabla \times \mathbf{u} = 0$, i.e. $\xi = 0$
  1. Irrotational flow implies $\exists \phi$ s.t. $\mathbf{u} = \nabla \phi$, i.e. $\mathbf{u}$ is a potential flow
  2. 2D potential flows reduce to solving $\nabla \cdot \mathbf{u} = \Delta \phi = 0$ with boundary conditions
  3. Can use this to enforce boundary conditions, as can add a potential flow that corrects for the rotational flow at the boundaries
- As with RGM the RVM is adaptive in that regions of high vorticity (rotation) get a high density of vortex "blobs"

MCMs for Other PDEs

- We have constructed MCMs for both elliptic and parabolic PDEs but have not considered MCMs for hyperbolic PDEs except for Berger's equation (was a very special case)
- In general MCMs for hyperbolic PDEs (like the wave equation: $u_{tt} = c^2 u_{xx}$) are hard to derive as Brownian motion is fundamentally related to diffusion (parabolic PDEs) and to the equilibrium of diffusion processes (elliptic PDEs), in contrast hyperbolic problems model distortion free information propagation which is fundamentally nonrandom
- A famous special case of an hyperbolic MCM for the telegrapher's equation (Kac, 1956):

$$\frac{1}{c^2}\frac{\partial^2 F}{\partial t^2} + \frac{2a}{c^2}\frac{\partial F}{\partial t} = \Delta F,$$

$$F(\mathbf{x},0) = \phi(\mathbf{x}), \quad \frac{\partial F(\mathbf{x},0)}{\partial t} = 0$$

MCMs for Other PDEs

- We have constructed MCMs for both elliptic and parabolic PDEs but have not considered MCMs for hyperbolic PDEs except for Berger's equation (was a very special case)
- In general MCMs for hyperbolic PDEs (like the wave equation: $u_{tt} = c^2 u_{xx}$) are hard to derive as Brownian motion is fundamentally related to diffusion (parabolic PDEs) and to the equilibrium of diffusion processes (elliptic PDEs), in contrast hyperbolic problems model distortion free information propagation which is fundamentally nonrandom
- A famous special case of an hyperbolic MCM for the telegrapher's equation (Kac, 1956):

$$\begin{aligned} \frac{1}{c^2}\frac{\partial^2 F}{\partial t^2} + \frac{2a}{c^2}\frac{\partial F}{\partial t} &= \Delta F, \\ F(\mathbf{x},0) = \phi(\mathbf{x}), \quad \frac{\partial F(\mathbf{x},0)}{\partial t} &= 0 \end{aligned}$$

MCMs for Other PDEs

- We have constructed MCMs for both elliptic and parabolic PDEs but have not considered MCMs for hyperbolic PDEs except for Berger's equation (was a very special case)
- In general MCMs for hyperbolic PDEs (like the wave equation: $u_{tt} = c^2 u_{xx}$) are hard to derive as Brownian motion is fundamentally related to diffusion (parabolic PDEs) and to the equilibrium of diffusion processes (elliptic PDEs), in contrast hyperbolic problems model distortion free information propagation which is fundamentally nonrandom
- A famous special case of an hyperbolic MCM for the telegrapher's equation (Kac, 1956):

$$\frac{1}{c^2}\frac{\partial^2 F}{\partial t^2} + \frac{2a}{c^2}\frac{\partial F}{\partial t} = \Delta F,$$

$$F(\mathbf{x},0) = \phi(\mathbf{x}), \quad \frac{\partial F(\mathbf{x},0)}{\partial t} = 0$$

MCMs for Other PDEs

- The telegrapher's equation approaches both the wave and heat equations in different limiting cases
  1. *Wave equation:* $a \to 0$
  2. *Heat equation:* $a, c \to \infty, 2a/c^2 \to \frac{1}{D}$
- Consider the one-dimensional telegrapher's equation, when $a = 0$ we know the solution is given by $F(x,t) = \frac{\phi(x+ct)+\phi(x-ct)}{2}$
- If we think of a as the probability per unit time of a Poisson process then $N(t) = \#$ of events occurring up to time t has the distribution $P\{N(t) = k\} = e^{-at}\frac{(at)^k}{k!}$
- If a particle moves with velocity c for time t it travels $ct = \int_0^t c\, d\tau$, if it undergoes random Poisson distributed direction reversal with probability per unit time a, the distance traveled in time t is $\int_0^t c(-1)^{N(\tau)}\, d\tau$

MCMs for Other PDEs

- The telegrapher's equation approaches both the wave and heat equations in different limiting cases
  1. *Wave equation:* $a \to 0$
  2. *Heat equation:* $a, c \to \infty, 2a/c^2 \to \frac{1}{D}$
- Consider the one-dimensional telegrapher's equation, when $a = 0$ we know the solution is given by $F(x, t) = \frac{\phi(x+ct)+\phi(x-ct)}{2}$
- If we think of a as the probability per unit time of a Poisson process then $N(t) = \#$ of events occurring up to time t has the distribution $P\{N(t) = k\} = e^{-at}\frac{(at)^k}{k!}$
- If a particle moves with velocity c for time t it travels $ct = \int_0^t c\, d\tau$, if it undergoes random Poisson distributed direction reversal with probability per unit time a, the distance traveled in time t is $\int_0^t c(-1)^{N(\tau)}\, d\tau$

MCMs for Other PDEs

- The telegrapher's equation approaches both the wave and heat equations in different limiting cases
  1. *Wave equation:* $a \to 0$
  2. *Heat equation:* $a, c \to \infty, 2a/c^2 \to \frac{1}{D}$
- Consider the one-dimensional telegrapher's equation, when $a = 0$ we know the solution is given by $F(x,t) = \frac{\phi(x+ct)+\phi(x-ct)}{2}$
- If we think of a as the probability per unit time of a Poisson process then $N(t) = \#$ of events occurring up to time t has the distribution $P\{N(t) = k\} = e^{-at}\frac{(at)^k}{k!}$
- If a particle moves with velocity c for time t it travels $ct = \int_0^t c\, d\tau$, if it undergoes random Poisson distributed direction reversal with probability per unit time a, the distance traveled in time t is $\int_0^t c(-1)^{N(\tau)}\, d\tau$

MCMs for Other PDEs

- The telegrapher's equation approaches both the wave and heat equations in different limiting cases
  1. *Wave equation:* $a \to 0$
  2. *Heat equation:* $a, c \to \infty, 2a/c^2 \to \frac{1}{D}$
- Consider the one-dimensional telegrapher's equation, when $a = 0$ we know the solution is given by $F(x,t) = \frac{\phi(x+ct)+\phi(x-ct)}{2}$
- If we think of a as the probability per unit time of a Poisson process then $N(t) = \#$ of events occurring up to time t has the distribution $P\{N(t) = k\} = e^{-at}\frac{(at)^k}{k!}$
- If a particle moves with velocity c for time t it travels $ct = \int_0^t c \, d\tau$, if it undergoes random Poisson distributed direction reversal with probability per unit time a, the distance traveled in time t is $\int_0^t c(-1)^{N(\tau)} \, d\tau$

MCMs for Other PDEs

- The telegrapher's equation approaches both the wave and heat equations in different limiting cases
  1. *Wave equation:* $a \to 0$
  2. *Heat equation:* $a, c \to \infty, 2a/c^2 \to \frac{1}{D}$
- Consider the one-dimensional telegrapher's equation, when $a = 0$ we know the solution is given by $F(x,t) = \frac{\phi(x+ct)+\phi(x-ct)}{2}$
- If we think of a as the probability per unit time of a Poisson process then $N(t) = \#$ of events occurring up to time t has the distribution $P\{N(t) = k\} = e^{-at}\frac{(at)^k}{k!}$
- If a particle moves with velocity c for time t it travels $ct = \int_0^t c\, d\tau$, if it undergoes random Poisson distributed direction reversal with probability per unit time a, the distance traveled in time t is $\int_0^t c(-1)^{N(\tau)}\, d\tau$

MCMs for Other PDEs

- The telegrapher's equation approaches both the wave and heat equations in different limiting cases
  1. *Wave equation:* $a \to 0$
  2. *Heat equation:* $a, c \to \infty, 2a/c^2 \to \frac{1}{D}$
- Consider the one-dimensional telegrapher's equation, when $a = 0$ we know the solution is given by $F(x,t) = \frac{\phi(x+ct)+\phi(x-ct)}{2}$
- If we think of a as the probability per unit time of a Poisson process then $N(t) = \#$ of events occurring up to time t has the distribution $P\{N(t) = k\} = e^{-at}\frac{(at)^k}{k!}$
- If a particle moves with velocity c for time t it travels $ct = \int_0^t c\, d\tau$, if it undergoes random Poisson distributed direction reversal with probability per unit time a, the distance traveled in time t is $\int_0^t c(-1)^{N(\tau)}\, d\tau$

MCMs for Other PDEs

- If we replace *ct* in the exact solution to the 1D wave equation by the randomized distance traveled average over all Poisson reversing paths we get:

$$F(x,t) = \frac{1}{2}E\left[\phi\left(x + \int_0^t c(-1)^{N(\tau)}\,d\tau\right)\right]$$

$$\frac{1}{2}E\left[\phi\left(x - \int_0^t c(-1)^{N(\tau)}\,d\tau\right)\right]$$

  which can be proven to solve the above IVP for the telegrapher's equation
- In any dimension, an exact solution for the wave equation can be converted into a solution to the telegrapher's equation by replacing *t* in the wave equation ansatz by the randomized time $\int_0^t (-1)^{N(\tau)}\,d\tau$ and averaging
- This is the basis of a MCM for the telegrapher's equation, one can also construct MCMs for finite-difference approximations to the telegrapher's equation

MCMs for Other PDEs

- If we replace ct in the exact solution to the 1D wave equation by the randomized distance traveled average over all Poisson reversing paths we get:

$$F(x,t) = \frac{1}{2}E\left[\phi\left(x + \int_0^t c(-1)^{N(\tau)}\, d\tau\right)\right]$$

$$\frac{1}{2}E\left[\phi\left(x - \int_0^t c(-1)^{N(\tau)}\, d\tau\right)\right]$$

  which can be proven to solve the above IVP for the telegrapher's equation
- In any dimension, an exact solution for the wave equation can be converted into a solution to the telegrapher's equation by replacing t in the wave equation ansatz by the randomized time $\int_0^t (-1)^{N(\tau)}\, d\tau$ and averaging
- This is the basis of a MCM for the telegrapher's equation, one can also construct MCMs for finite-difference approximations to the telegrapher's equation

MCMs for Other PDEs

- If we replace *ct* in the exact solution to the 1D wave equation by the randomized distance traveled average over all Poisson reversing paths we get:

$$F(x,t) = \frac{1}{2}E\left[\phi\left(x + \int_0^t c(-1)^{N(\tau)}\,d\tau\right)\right]$$

$$\frac{1}{2}E\left[\phi\left(x - \int_0^t c(-1)^{N(\tau)}\,d\tau\right)\right]$$

  which can be proven to solve the above IVP for the telegrapher's equation
- In any dimension, an exact solution for the wave equation can be converted into a solution to the telegrapher's equation by replacing t in the wave equation ansatz by the randomized time $\int_0^t (-1)^{N(\tau)}\,d\tau$ and averaging
- This is the basis of a MCM for the telegrapher's equation, one can also construct MCMs for finite-difference approximations to the telegrapher's equation

Nonlinear Equations and the Feynman-Kac Formula

- Recall that in the Feynman-Kac formula the operator L enters in through the SDE which generates the sample paths over which expected values are taken
- If one replaces the sample paths from solutions of linear SDEs with paths derived from branching processes one can sample certain nonlinear parabolic PDEs directly (McKean, 1988)
- Recall that the solution to the IVP for the heat equation is represented via Feynman-Kac as: $u(x, t) = E_x[u_0(\beta(t))]$

Nonlinear Equations and the Feynman-Kac Formula

- Recall that in the Feynman-Kac formula the operator L enters in through the SDE which generates the sample paths over which expected values are taken
- If one replaces the sample paths from solutions of linear SDEs with paths derived from branching processes one can sample certain nonlinear parabolic PDEs directly (McKean, 1988)
- Recall that the solution to the IVP for the heat equation is represented via Feynman-Kac as: $u(x, t) = E_x[u_0(\beta(t))]$

Nonlinear Equations and the Feynman-Kac Formula

- Recall that in the Feynman-Kac formula the operator L enters in through the SDE which generates the sample paths over which expected values are taken
- If one replaces the sample paths from solutions of linear SDEs with paths derived from branching processes one can sample certain nonlinear parabolic PDEs directly (McKean, 1988)
- Recall that the solution to the IVP for the heat equation is represented via Feynman-Kac as: $u(x,t) = E_x[u_0(\beta(t))]$

Nonlinear Equations and the Feynman-Kac Formula

Consider normal Brownian motion with exponentially distributed branching with unit branching probability per unit time, then the Feynman-Kac representation above with expectations taken over this branching process instead of normal Brownian motion solves the IVP for the Kolmogorov-Petrovskii-Piskunov equation:

$$\frac{\partial u}{\partial t} = \frac{1}{2}\Delta u + \left(u^2 - u\right), \quad u(x,0) = u_0(x)$$

has solution

$$u(x,t) = E_x\left[\prod_{i=1}^{N(t)} u_0(x_i(t))\right]$$

where the branching Brownian motion started at x at $t = 0$ leads to $N(t)$ particles at time t with locations $x_i(t), i = 1, \ldots, N(t)$

Nonlinear Equations and the Feynman-Kac Formula

- If instead of binary branching at exponentially distributed branching times there are n with probability p_n, $\sum_{n=2}^{\infty} p_n = 1$, then the branching Feynman-Kac formula solves the IVP for:

$$\frac{\partial u}{\partial t} = \frac{1}{2}\Delta u + \left(\sum_{n=2}^{\infty} p_n u^n - u \right)$$

- Can also have $p_n < 0$ with $\sum_{n=2}^{\infty} |p_n| = 1$ with same result except that must have two representations for the normal and the "negative" walkers

Nonlinear Equations and the Feynman-Kac Formula

- If instead of binary branching at exponentially distributed branching times there are n with probability p_n, $\sum_{n=2}^{\infty} p_n = 1$, then the branching Feynman-Kac formula solves the IVP for:

$$\frac{\partial u}{\partial t} = \frac{1}{2}\Delta u + \left(\sum_{n=2}^{\infty} p_n u^n - u \right)$$

- Can also have $p_n < 0$ with $\sum_{n=2}^{\infty} |p_n| = 1$ with same result except that must have two representations for the normal and the “negative” walkers

Nonlinear Equations and the Feynman-Kac Formula

- If $\lambda =$ the probability per unit time of branching then we can solve the IVP for:

$$\frac{\partial u}{\partial t} = \frac{1}{2}\Delta u + \lambda \left(\sum_{n=2}^{\infty} p_n u^n - u \right)$$

- If we have $k(x,t)$ as inhomogeneous branching probability per unit time, then we solve the IVP for:

$$\frac{\partial u}{\partial t} = \frac{1}{2}\Delta u + \int_0^t k(x,\tau)\, d\tau \left(\sum_{n=2}^{\infty} p_n u^n - u \right)$$

Nonlinear Equations and the Feynman-Kac Formula

- If $\lambda =$ the probability per unit time of branching then we can solve the IVP for:

$$\frac{\partial u}{\partial t} = \frac{1}{2}\Delta u + \lambda \left(\sum_{n=2}^{\infty} p_n u^n - u \right)$$

- If we have $k(x, t)$ as inhomogeneous branching probability per unit time, then we solve the IVP for:

$$\frac{\partial u}{\partial t} = \frac{1}{2}\Delta u + \int_0^t k(x, \tau)\, d\tau \left(\sum_{n=2}^{\infty} p_n u^n - u \right)$$

Monte Carlo Methods and Linear Algebra

- Monte Carlo has been, and continues to be used in linear algebra
- 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 $x = (I - H)^{-1} b$
- Note: $||H||$ can be defined in many ways, e. g.:
  1. $||H|| = \max_i \left(\sum_j |h_{ij}| \right)$
  2. $||H|| = \max_i (|\lambda_i(H)|) = \rho(H)$ (spectral radius)

Monte Carlo Methods and Linear Algebra

- Monte Carlo has been, and continues to be used in linear algebra
- 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 $x = (I - H)^{-1} b$
- Note: $||H||$ can be defined in many ways, e. g.:
  1. $||H|| = \max_i \left(\sum_j |h_{ij}| \right)$
  2. $||H|| = \max_i (|\lambda_i(H)|) = \rho(H)$ (spectral radius)

Monte Carlo Methods and Linear Algebra

- Monte Carlo has been, and continues to be used in linear algebra
- 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 $x = (I - H)^{-1} b$
- Note: $||H||$ can be defined in many ways, e. g.:
  1. $||H|| = \max_i \left(\sum_j |h_{ij}| \right)$
  2. $||H|| = \max_i (|\lambda_i(H)|) = \rho(H)$ (spectral radius)

Monte Carlo Methods and Linear Algebra

- Monte Carlo has been, and continues to be used in linear algebra
- 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 $x = (I - H)^{-1} b$
- Note: $||H||$ can be defined in many ways, e. g.:
  1. $||H|| = \max_i \left(\sum_j |h_{ij}| \right)$
  2. $||H|| = \max_i (|\lambda_i(H)|) = \rho(H)$ (spectral radius)

Monte Carlo Methods and Linear Algebra

- Monte Carlo has been, and continues to be used in linear algebra
- 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:

$$N = \sum_{i=0}^{\infty} H^i = (I - H)^{-1}, \quad ||N|| = \sum_{i=0}^{\infty} ||H^i|| \le \sum_{i=0}^{\infty} \mathbb{H}^i = \frac{1}{1 - \mathbb{H}}$$

- Formally, the solution is $x = (I - H)^{-1} b$
- Note: $||H||$ can be defined in many ways, e. g.:
  1. $||H|| = \max_i \left(\sum_j |h_{ij}| \right)$
  2. $||H|| = \max_i (|\lambda_i(H)|) = \rho(H)$ (spectral radius)

Monte Carlo Methods and Linear Algebra

- Monte Carlo has been, and continues to be used in linear algebra
- 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 $x = (I - H)^{-1} b$
- Note: $||H||$ can be defined in many ways, e. g.:
  1. $||H|| = \max_i \left(\sum_j |h_{ij}| \right)$
  2. $||H|| = \max_i \left(|\lambda_i(H)| \right) = \rho(H)$ (spectral radius)

Monte Carlo Methods and Linear Algebra

- Recall: Monte Carlo can be used to form a sum: $S = \sum_{i=1}^{M} a_i$ as follows
  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 $E[a_i/p_i] = \sum_{i=1}^{M} \left(\frac{a_i}{p_i}\right) p_i = S$
  3. $Var[a_i/p_i] \propto \sum_{i=1}^{M} \left(\frac{a_i^2}{p_i}\right)$, so optimal choice is $p_i \propto a_i^2$
- Given $||H|| = \mathbb{H} < 1$, solving $x = Hx + b$ via Neumann series requires successive matrix-vector multiplication
- We can use the above sampling of a sum to construct a sample of the Neumann series

Monte Carlo Methods and Linear Algebra

- Recall: Monte Carlo can be used to form a sum: $S = \sum_{i=1}^{M} a_i$ as follows
  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 $E[a_i/p_i] = \sum_{i=1}^{M} \left(\frac{a_i}{p_i}\right) p_i = S$
  3. $Var[a_i/p_i] \propto \sum_{i=1}^{M} \left(\frac{a_i^2}{p_i}\right)$, so optimal choice is $p_i \propto a_i^2$
- Given $||H|| = \mathbb{H} < 1$, solving $x = Hx + b$ via Neumann series requires successive matrix-vector multiplication
- We can use the above sampling of a sum to construct a sample of the Neumann series

Monte Carlo Methods and Linear Algebra

- Recall: Monte Carlo can be used to form a sum: $S = \sum_{i=1}^{M} a_i$ as follows
  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 $E[a_i/p_i] = \sum_{i=1}^{M} \left(\frac{a_i}{p_i}\right) p_i = S$
  3. $Var[a_i/p_i] \propto \sum_{i=1}^{M} \left(\frac{a_i^2}{p_i}\right)$, so optimal choice is $p_i \propto a_i^2$
- Given $||H|| = \mathbb{H} < 1$, solving $x = Hx + b$ via Neumann series requires successive matrix-vector multiplication
- We can use the above sampling of a sum to construct a sample of the Neumann series

Monte Carlo Methods and Linear Algebra

- Recall: Monte Carlo can be used to form a sum: $S = \sum_{i=1}^{M} a_i$ as follows
  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 $E[a_i/p_i] = \sum_{i=1}^{M} \left(\frac{a_i}{p_i}\right) p_i = S$
  3. $Var[a_i/p_i] \propto \sum_{i=1}^{M} \left(\frac{a_i^2}{p_i}\right)$, so optimal choice is $p_i \propto a_i^2$
- Given $||H|| = \mathbb{H} < 1$, solving $x = Hx + b$ via Neumann series requires successive matrix-vector multiplication
- We can use the above sampling of a sum to construct a sample of the Neumann series

Monte Carlo Methods and Linear Algebra

- Recall: Monte Carlo can be used to form a sum: $S = \sum_{i=1}^{M} a_i$ as follows
  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 $E[a_i/p_i] = \sum_{i=1}^{M} \left(\frac{a_i}{p_i}\right) p_i = S$
  3. $Var[a_i/p_i] \propto \sum_{i=1}^{M} \left(\frac{a_i^2}{p_i}\right)$, so optimal choice is $p_i \propto a_i^2$
- Given $||H|| = \mathbb{H} < 1$, solving $x = Hx + b$ via Neumann series requires successive matrix-vector multiplication
- We can use the above sampling of a sum to construct a sample of the Neumann series

Monte Carlo Methods and Linear Algebra

- Recall: Monte Carlo can be used to form a sum: $S = \sum_{i=1}^{M} a_i$ as follows
  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 $E[a_i/p_i] = \sum_{i=1}^{M} \left(\frac{a_i}{p_i}\right) p_i = S$
  3. $Var[a_i/p_i] \propto \sum_{i=1}^{M} \left(\frac{a_i^2}{p_i}\right)$, so optimal choice is $p_i \propto a_i^2$
- Given $||H|| = \mathbb{H} < 1$, solving $x = Hx + b$ via Neumann series requires successive matrix-vector multiplication
- We can use the above sampling of a sum to construct a sample of the Neumann series

Monte Carlo Methods and Linear Algebra

The Ulam-von Neumann Method

- We first construct a Markov chain based on H and b to sample a ℓ-fold product matrices as follows
- Define the transition probability matrix, P
  1. $p_{ij} \geq 0$, and $p_{ij} > 0$ when $h_{ij} \neq 0$
  2. Define $p_i = 1 - \sum_j p_{ij}$
- Now define a Markov chain on states $\{0, 1, \ldots, n\}$ with transition probability, p_{ij}, and termination probability from state i, p_i
- Also define

$$v_{ij} = \begin{cases} \frac{h_{ij}}{p_{ij}}, & h_{ij} \neq 0 \\ 0, & \text{otherwise} \end{cases}$$

Monte Carlo Methods and Linear Algebra

The Ulam-von Neumann Method

- We first construct a Markov chain based on H and b to sample a ℓ-fold product matrices as follows
- Define the transition probability matrix, P
  1. $p_{ij} \geq 0$, and $p_{ij} > 0$ when $h_{ij} \neq 0$
  2. Define $p_i = 1 - \sum_j p_{ij}$
- Now define a Markov chain on states $\{0, 1, \ldots, n\}$ with transition probability, p_{ij}, and termination probability from state i, p_i
- Also define

$$v_{ij} = \begin{cases} \frac{h_{ij}}{p_{ij}}, & h_{ij} \neq 0 \\ 0, & \text{otherwise} \end{cases}$$

Monte Carlo Methods and Linear Algebra

The Ulam-von Neumann Method

- We first construct a Markov chain based on H and b to sample a ℓ-fold product matrices as follows
- Define the transition probability matrix, P
  1. $p_{ij} \geq 0$, and $p_{ij} > 0$ when $h_{ij} \neq 0$
  2. Define $p_i = 1 - \sum_j p_{ij}$
- Now define a Markov chain on states $\{0, 1, \dots, n\}$ with transition probability, p_{ij}, and termination probability from state i, p_i
- Also define

$$v_{ij} = \begin{cases} \frac{h_{ij}}{p_{ij}}, & h_{ij} \neq 0 \\ 0, & \text{otherwise} \end{cases}$$

Monte Carlo Methods and Linear Algebra

The Ulam-von Neumann Method

- We first construct a Markov chain based on H and b to sample a ℓ-fold product matrices as follows
- Define the transition probability matrix, P
  1. $p_{ij} \geq 0$, and $p_{ij} > 0$ when $h_{ij} \neq 0$
  2. Define $p_i = 1 - \sum_j p_{ij}$
- Now define a Markov chain on states $\{0, 1, \dots, n\}$ with transition probability, p_{ij}, and termination probability from state i, p_i
- Also define

$$v_{ij} = \begin{cases} \frac{h_{ij}}{p_{ij}}, & h_{ij} \neq 0 \\ 0, & \text{otherwise} \end{cases}$$

Monte Carlo Methods and Linear Algebra

The Ulam-von Neumann Method

- We first construct a Markov chain based on H and b to sample a ℓ-fold product matrices as follows
- Define the transition probability matrix, P
  1. $p_{ij} \geq 0$, and $p_{ij} > 0$ when $h_{ij} \neq 0$
  2. Define $p_i = 1 - \sum_j p_{ij}$
- Now define a Markov chain on states $\{0, 1, \dots, n\}$ with transition probability, p_{ij}, and termination probability from state i, p_i
- Also define

$$v_{ij} = \begin{cases} \frac{h_{ij}}{p_{ij}}, & h_{ij} \neq 0 \\ 0, & \text{otherwise} \end{cases}$$

Monte Carlo Methods and Linear Algebra

The Ulam-von Neumann Method

- We first construct a Markov chain based on H and b to sample a ℓ-fold product matrices as follows
- Define the transition probability matrix, P
  1. $p_{ij} \geq 0$, and $p_{ij} > 0$ when $h_{ij} \neq 0$
  2. Define $p_i = 1 - \sum_j p_{ij}$
- Now define a Markov chain on states $\{0, 1, \dots, n\}$ with transition probability, p_{ij}, and termination probability from state i, p_i
- Also define

$$v_{ij} = \begin{cases} \frac{h_{ij}}{p_{ij}}, & h_{ij} \neq 0 \\ 0, & \text{otherwise} \end{cases}$$

Monte Carlo Methods and Linear Algebra

- Given the desire to sample x_i we create the following estimator based on
  1. Generating a Markov chain $\gamma(i_0, i_1, \dots, i_k)$ using p_{ij} and p_i, where state i_k is the penultimate before absorbtion
  2. Form the estimator

$$\mathbb{X}(\gamma) = \mathbb{V}_m(\gamma)\frac{b_{i_k}}{p_{i_k}}, \text{ where } \mathbb{V}_m(\gamma) = \sum_{j=1}^{m} v_{i_{j-1}i_j}, \quad m \leq k$$

Monte Carlo Methods and Linear Algebra

- Given the desire to sample x_i we create the following estimator based on
  1. Generating a Markov chain $\gamma(i_0, i_1, \dots, i_k)$ using p_{ij} and p_i, where state i_k is the penultimate before absorbtion
  2. Form the estimator

$$\mathbb{X}(\gamma) = \mathbb{V}_m(\gamma)\frac{b_{i_k}}{p_{i_k}}, \text{ where } \mathbb{V}_m(\gamma) = \sum_{j=1}^{m} v_{i_{j-1} i_j}, \; m \le k$$

Monte Carlo Methods and Linear Algebra

- Given the desire to sample x_i we create the following estimator based on
  1. Generating a Markov chain $\gamma(i_0, i_1, \dots, i_k)$ using p_{ij} and p_i, where state i_k is the penultimate before absorbtion
  2. Form the estimator

$$\mathbb{X}(\gamma) = \mathbb{V}_m(\gamma)\frac{b_{i_k}}{p_{i_k}}, \text{ where } \mathbb{V}_m(\gamma) = \sum_{j=1}^{m} v_{i_{j-1}i_j}, \; m \le k$$

Monte Carlo Methods and Linear Algebra

- We have

$$E\left[\mathbb{X}(\gamma)|i_0 = i\right] = \sum_{k=0}^{\infty}\left(\sum_{i_1}\cdots\sum_{i_k} p_{ii_1}\dots p_{i_{k-1}i_k} v_{ii_1}\dots v_{i_{k-1}i_k}\frac{b_{i_k}}{p_{i_k}}\right) =$$

$$\sum_{k=0}^{\infty}\left(\sum_{i_1}\cdots\sum_{i_k} p_{ii_1} v_{ii_1}\dots p_{i_{k-1}i_k} v_{i_{k-1}i_k}\frac{b_{i_k}}{p_{i_k}}\right) =$$

$$\sum_{k=0}^{\infty}\left(\sum_{i_1}\cdots\sum_{i_k} h_{ii_1} h_{i_{k-1}i_k} b_{i_k}\right) = \text{the } i\text{th component of} \sum_{k=0}^{\infty} H^k b$$

Monte Carlo Methods and Linear Algebra

Elaborations on Monte Carlo for Linear Algebra

- Backward walks (Adjoint sampling)
- The lower variance Wasow estimator

$$\mathbb{X}^*(\gamma) = \sum_{m=0}^{k} \mathbb{V}_m(\gamma) b_{i_k}$$

- Again with only matrix-vector multiplication can do the power method and shifted variants for sampling eigenvalues
- Newer, more highly convergent methods are based on randomly sampling smaller problems
- All these have integral equation and Feynman-Kac analogs

Monte Carlo Methods and Linear Algebra

Elaborations on Monte Carlo for Linear Algebra

- Backward walks (Adjoint sampling)
- The lower variance Wasow estimator

$$\mathbb{X}^*(\gamma) = \sum_{m=0}^{k} \mathbb{V}_m(\gamma) b_{i_k}$$

- Again with only matrix-vector multiplication can do the power method and shifted variants for sampling eigenvalues
- Newer, more highly convergent methods are based on randomly sampling smaller problems
- All these have integral equation and Feynman-Kac analogs

Monte Carlo Methods and Linear Algebra

Elaborations on Monte Carlo for Linear Algebra

- Backward walks (Adjoint sampling)
- The lower variance Wasow estimator

$$\mathbb{X}^*(\gamma) = \sum_{m=0}^{k} \mathbb{V}_m(\gamma) b_{i_k}$$

- Again with only matrix-vector multiplication can do the power method and shifted variants for sampling eigenvalues
- Newer, more highly convergent methods are based on randomly sampling smaller problems
- All these have integral equation and Feynman-Kac analogs

Monte Carlo Methods and Linear Algebra

Elaborations on Monte Carlo for Linear Algebra

- Backward walks (Adjoint sampling)
- The lower variance Wasow estimator

$$\mathbb{X}^*(\gamma) = \sum_{m=0}^{k} \mathbb{V}_m(\gamma) b_{i_k}$$

- Again with only matrix-vector multiplication can do the power method and shifted variants for sampling eigenvalues
- Newer, more highly convergent methods are based on randomly sampling smaller problems
- All these have integral equation and Feynman-Kac analogs

Monte Carlo Methods and Linear Algebra

Elaborations on Monte Carlo for Linear Algebra

- Backward walks (Adjoint sampling)
- The lower variance Wasow estimator

$$\mathbb{X}^*(\gamma) = \sum_{m=0}^{k} \mathbb{V}_m(\gamma) b_{i_k}$$

- Again with only matrix-vector multiplication can do the power method and shifted variants for sampling eigenvalues
- Newer, more highly convergent methods are based on randomly sampling smaller problems
- All these have integral equation and Feynman-Kac analogs

Parallel Computing Overview

- Now that we know all these MCMs, we must discuss how to implement these methods on parallel (and vector) computers
- There are two major classes of parallel computer being commercially produced
    1. Single Instruction Multiple Data machines:
- Only one data stream so the same instruction is broadcast to all processors
- Usually these machines have many simple processors, often they are bit serial (fine grained)
- Usually these machines have distributed memory
- Connection Machine and vector-processing units are “classical” examples
- Modern examples include GPGPUs

Parallel Computing Overview

- Now that we know all these MCMs, we must discuss how to implement these methods on parallel (and vector) computers
- There are two major classes of parallel computer being commercially produced
  1. **S**ingle **I**nstruction **M**ultiple **D**ata machines:
- Only one data stream so the same instruction is broadcast to all processors
- Usually these machines have many simple processors, often they are bit serial (fine grained)
- Usually these machines have distributed memory
- Connection Machine and vector-processing units are “classical” examples
- Modern examples include GPGPUs

Parallel Computing Overview

- Now that we know all these MCMs, we must discuss how to implement these methods on parallel (and vector) computers
- There are two major classes of parallel computer being commercially produced
  1. **S**ingle **I**nstruction **M**ultiple **D**ata machines:
- Only one data stream so the same instruction is broadcast to all processors
- Usually these machines have many simple processors, often they are bit serial (fine grained)
- Usually these machines have distributed memory
- Connection Machine and vector-processing units are “classical” examples
- Modern examples include GPGPUs

Parallel Computing Overview

- Now that we know all these MCMs, we must discuss how to implement these methods on parallel (and vector) computers
- There are two major classes of parallel computer being commercially produced
  1. **S**ingle **I**nstruction **M**ultiple **D**ata machines:
- Only one data stream so the same instruction is broadcast to all processors
- Usually these machines have many simple processors, often they are bit serial (fine grained)
- Usually these machines have distributed memory
- Connection Machine and vector-processing units are “classical” examples
- Modern examples include GPGPUs

Parallel Computing Overview

- Now that we know all these MCMs, we must discuss how to implement these methods on parallel (and vector) computers
- There are two major classes of parallel computer being commercially produced
  1. **S**ingle **I**nstruction **M**ultiple **D**ata machines:
- Only one data stream so the same instruction is broadcast to all processors
- Usually these machines have many simple processors, often they are bit serial (fine grained)
- Usually these machines have distributed memory
- Connection Machine and vector-processing units are “classical” examples
- Modern examples include GPGPUs

Parallel Computing Overview

- Now that we know all these MCMs, we must discuss how to implement these methods on parallel (and vector) computers
- There are two major classes of parallel computer being commercially produced
  1. **S**ingle **I**nstruction **M**ultiple **D**ata machines:
- Only one data stream so the same instruction is broadcast to all processors
- Usually these machines have many simple processors, often they are bit serial (fine grained)
- Usually these machines have distributed memory
- Connection Machine and vector-processing units are “classical” examples
- Modern examples include GPGPUs

Parallel Computing Overview

- Now that we know all these MCMs, we must discuss how to implement these methods on parallel (and vector) computers
- There are two major classes of parallel computer being commercially produced
  1. **S**ingle **I**nstruction **M**ultiple **D**ata machines:
- Only one data stream so the same instruction is broadcast to all processors
- Usually these machines have many simple processors, often they are bit serial (fine grained)
- Usually these machines have distributed memory
- Connection Machine and vector-processing units are “classical” examples
- Modern examples include GPGPUs

Parallel Computing Overview

- Now that we know all these MCMs, we must discuss how to implement these methods on parallel (and vector) computers
- There are two major classes of parallel computer being commercially produced
   1. **S**ingle **I**nstruction **M**ultiple **D**ata machines:
- Only one data stream so the same instruction is broadcast to all processors
- Usually these machines have many simple processors, often they are bit serial (fine grained)
- Usually these machines have distributed memory
- Connection Machine and vector-processing units are “classical” examples
- Modern examples include GPGPUs

Parallel Computing Overview

1. **M**ultiple **I**nstruction **M**ultiple **D**ata machines:
    - These machines are a collection of conventional computers with an interconnection network
    - Each processor can run its own program
    - Processors are usually large grained
    - Can have shared or distributed memory
    - Shared memory limits the number of processors though bus technology
    - Distributed memory can be implemented with many different interconnection topologies
    - Modern examples:
        1.1 *Multicore machines*
        1.2 *Clustered machines*
        1.3 *Shared-memory machines*

Parallel Computing Overview

1. **M**ultiple **I**nstruction **M**ultiple **D**ata machines:
   - These machines are a collection of conventional computers with an interconnection network
   - Each processor can run its own program
   - Processors are usually large grained
   - Can have shared or distributed memory
   - Shared memory limits the number of processors though bus technology
   - Distributed memory can be implemented with many different interconnection topologies
   - Modern examples:

     1.1 *Multicore machines*

     1.2 *Clustered machines*

     1.3 *Shared-memory machines*

Parallel Computing Overview

1. **M**ultiple **I**nstruction **M**ultiple **D**ata machines:
   - These machines are a collection of conventional computers with an interconnection network
   - Each processor can run its own program
   - Processors are usually large grained
   - Can have shared or distributed memory
   - Shared memory limits the number of processors though bus technology
   - Distributed memory can be implemented with many different interconnection topologies
   - Modern examples:
     1.1 *Multicore machines*
     1.2 *Clustered machines*
     1.3 *Shared-memory machines*

Parallel Computing Overview

1. **M**ultiple **I**nstruction **M**ultiple **D**ata machines:
   - These machines are a collection of conventional computers with an interconnection network
   - Each processor can run its own program
   - Processors are usually large grained
   - Can have shared or distributed memory
   - Shared memory limits the number of processors though bus technology
   - Distributed memory can be implemented with many different interconnection topologies
   - Modern examples:
     1.1 *Multicore machines*
     1.2 *Clustered machines*
     1.3 *Shared-memory machines*

Parallel Computing Overview

1. **M**ultiple **I**nstruction **M**ultiple **D**ata machines:
   - These machines are a collection of conventional computers with an interconnection network
   - Each processor can run its own program
   - Processors are usually large grained
   - Can have shared or distributed memory
   - Shared memory limits the number of processors though bus technology
   - Distributed memory can be implemented with many different interconnection topologies
   - Modern examples:
     1.1 *Multicore machines*
     1.2 *Clustered machines*
     1.3 *Shared-memory machines*

Parallel Computing Overview

1. **M**ultiple **I**nstruction **M**ultiple **D**ata machines:
   - These machines are a collection of conventional computers with an interconnection network
   - Each processor can run its own program
   - Processors are usually large grained
   - Can have shared or distributed memory
   - Shared memory limits the number of processors though bus technology
   - Distributed memory can be implemented with many different interconnection topologies
   - Modern examples:
     1.1 *Multicore machines*
     1.2 *Clustered machines*
     1.3 *Shared-memory machines*

Parallel Computing Overview

1. **M**ultiple **I**nstruction **M**ultiple **D**ata machines:
   - These machines are a collection of conventional computers with an interconnection network
   - Each processor can run its own program
   - Processors are usually large grained
   - Can have shared or distributed memory
   - Shared memory limits the number of processors though bus technology
   - Distributed memory can be implemented with many different interconnection topologies
   - Modern examples:
     1.1 *Multicore machines*
     1.2 *Clustered machines*
     1.3 *Shared-memory machines*

Parallel Computing Overview

1. **M**ultiple **I**nstruction **M**ultiple **D**ata machines:
   - These machines are a collection of conventional computers with an interconnection network
   - Each processor can run its own program
   - Processors are usually large grained
   - Can have shared or distributed memory
   - Shared memory limits the number of processors though bus technology
   - Distributed memory can be implemented with many different interconnection topologies
   - Modern examples:
     1.1 *Multicore machines*
     1.2 *Clustered machines*
     1.3 *Shared-memory machines*

Parallel Computing Overview

1. **M**ultiple **I**nstruction **M**ultiple **D**ata machines:
   - These machines are a collection of conventional computers with an interconnection network
   - Each processor can run its own program
   - Processors are usually large grained
   - Can have shared or distributed memory
   - Shared memory limits the number of processors though bus technology
   - Distributed memory can be implemented with many different interconnection topologies
   - Modern examples:
     1.1 *Multicore machines*
     1.2 *Clustered machines*
     1.3 *Shared-memory machines*

Parallel Computing Overview

1. **M**ultiple **I**nstruction **M**ultiple **D**ata machines:
   - These machines are a collection of conventional computers with an interconnection network
   - Each processor can run its own program
   - Processors are usually large grained
   - Can have shared or distributed memory
   - Shared memory limits the number of processors though bus technology
   - Distributed memory can be implemented with many different interconnection topologies
   - Modern examples:
     1.1 *Multicore machines*
     1.2 *Clustered machines*
     1.3 *Shared-memory machines*

Parallel Computing Overview

1. **M**ultiple **I**nstruction **M**ultiple **D**ata machines:
   - These machines are a collection of conventional computers with an interconnection network
   - Each processor can run its own program
   - Processors are usually large grained
   - Can have shared or distributed memory
   - Shared memory limits the number of processors though bus technology
   - Distributed memory can be implemented with many different interconnection topologies
   - Modern examples:
     1.1 *Multicore machines*
     1.2 *Clustered machines*
     1.3 *Shared-memory machines*

General Principles for Constructing Parallel Algorithms

- Use widely replicated aspects of a given problem as the basis for parallelism
  1. In MCMs can map each independent statistical sample to an independent process with essentially no interprocessor communication
  2. With spatial systems, spatial subdomains (domain decomposition) can be used but implicit schemes and global conditions (elliptic BVPs) will require considerable communication
  3. In some cases solutions over time are desired as in ODE problems, and the iterative solution of the entire time trajectory (continuation methods) can be distributed by time point

General Principles for Constructing Parallel Algorithms

- Use widely replicated aspects of a given problem as the basis for parallelism
  1. In MCMs can map each independent statistical sample to an independent process with essentially no interprocessor communication
  2. With spatial systems, spatial subdomains (domain decomposition) can be used but implicit schemes and global conditions (elliptic BVPs) will require considerable communication
  3. In some cases solutions over time are desired as in ODE problems, and the iterative solution of the entire time trajectory (continuation methods) can be distributed by time point

General Principles for Constructing Parallel Algorithms

- Use widely replicated aspects of a given problem as the basis for parallelism
  1. In MCMs can map each independent statistical sample to an independent process with essentially no interprocessor communication
  2. With spatial systems, spatial subdomains (domain decomposition) can be used but implicit schemes and global conditions (elliptic BVPs) will require considerable communication
  3. In some cases solutions over time are desired as in ODE problems, and the iterative solution of the entire time trajectory (continuation methods) can be distributed by time point

General Principles for Constructing Parallel Algorithms

- Use widely replicated aspects of a given problem as the basis for parallelism
  1. In MCMs can map each independent statistical sample to an independent process with essentially no interprocessor communication
  2. With spatial systems, spatial subdomains (domain decomposition) can be used but implicit schemes and global conditions (elliptic BVPs) will require considerable communication
  3. In some cases solutions over time are desired as in ODE problems, and the iterative solution of the entire time trajectory (continuation methods) can be distributed by time point

General Approaches to the Construction of MCMs for Elliptic BVPs

- As a simple example, consider the Dirichlet problem for the Laplace equation (1)
- The Wiener integral representation for the solution to (1) is $u(x) = E_x[g(\beta(\tau_{\partial\Omega}))]$
- General approaches are to use (i) “exact” samples of the Wiener integral, (ii) sample from a discretization of the Wiener integral
- Can sample with spherical processes, use the MVP and spheres to walk until ϵ from the boundary, other elliptic L’s lead to uniformity on ellipses instead of spheres (i)

General Approaches to the Construction of MCMs for Elliptic BVPs

- As a simple example, consider the Dirichlet problem for the Laplace equation (1)
- The Wiener integral representation for the solution to (1) is $u(x) = E_x[g(\beta(\tau_{\partial\Omega}))]$
- General approaches are to use (i) “exact” samples of the Wiener integral, (ii) sample from a discretization of the Wiener integral
- Can sample with spherical processes, use the MVP and spheres to walk until ϵ from the boundary, other elliptic L’s lead to uniformity on ellipses instead of spheres (i)

General Approaches to the Construction of MCMs for Elliptic BVPs

- As a simple example, consider the Dirichlet problem for the Laplace equation (1)
- The Wiener integral representation for the solution to (1) is $u(x) = E_x[g(\beta(\tau_{\partial\Omega}))]$
- General approaches are to use (i) “exact” samples of the Wiener integral, (ii) sample from a discretization of the Wiener integral
- Can sample with spherical processes, use the MVP and spheres to walk until ϵ from the boundary, other elliptic L’s lead to uniformity on ellipses instead of spheres (i)

General Approaches to the Construction of MCMs for Elliptic BVPs

- As a simple example, consider the Dirichlet problem for the Laplace equation (1)
- The Wiener integral representation for the solution to (1) is $u(x) = E_x[g(\beta(\tau_{\partial\Omega}))]$
- General approaches are to use (i) "exact" samples of the Wiener integral, (ii) sample from a discretization of the Wiener integral
- Can sample with spherical processes, use the MVP and spheres to walk until ϵ from the boundary, other elliptic L's lead to uniformity on ellipses instead of spheres (i)

General Approaches to the Construction of MCMs for Elliptic BVPs

- Random Fourier series relationship to Brownian motion can be used to generate walks via a truncated series, in some cases this gives an exact random series solution which is then sampled, with complicated Ω this approach is not practical (ii)
- Can use a high dimensional integral approximation of the infinite-dimensional (Wiener) integral, the finite-dimensional integral is then evaluated via analytic or MCMs (ii)
- Spatially discretize the region and sample from the discrete Wiener integral (ii)
- This last is a very fruitful approach especially w.r.t. parallel computers

General Approaches to the Construction of MCMs for Elliptic BVPs

- Random Fourier series relationship to Brownian motion can be used to generate walks via a truncated series, in some cases this gives an exact random series solution which is then sampled, with complicated Ω this approach is not practical (ii)
- Can use a high dimensional integral approximation of the infinite-dimensional (Wiener) integral, the finite-dimensional integral is then evaluated via analytic or MCMs (ii)
- Spatially discretize the region and sample from the discrete Wiener integral (ii)
- This last is a very fruitful approach especially w.r.t. parallel computers

General Approaches to the Construction of MCMs for Elliptic BVPs

- Random Fourier series relationship to Brownian motion can be used to generate walks via a truncated series, in some cases this gives an exact random series solution which is then sampled, with complicated Ω this approach is not practical (ii)
- Can use a high dimensional integral approximation of the infinite-dimensional (Wiener) integral, the finite-dimensional integral is then evaluated via analytic or MCMs (ii)
- Spatially discretize the region and sample from the discrete Wiener integral (ii)
- This last is a very fruitful approach especially w.r.t. parallel computers

General Approaches to the Construction of MCMs for Elliptic BVPs

- Random Fourier series relationship to Brownian motion can be used to generate walks via a truncated series, in some cases this gives an exact random series solution which is then sampled, with complicated Ω this approach is not practical (ii)
- Can use a high dimensional integral approximation of the infinite-dimensional (Wiener) integral, the finite-dimensional integral is then evaluated via analytic or MCMs (ii)
- Spatially discretize the region and sample from the discrete Wiener integral (ii)
- This last is a very fruitful approach especially w.r.t. parallel computers

Discrete Wiener Integrals

- All of the theory for continuous sample path Wiener integrals mentioned above carries over to the discrete cases
  1. Can replace the continuous region Ω with a discretization Ω_h where h the characteristic discretization parameter
  2. Replace $\beta(\cdot)$ with random walks on Ω_h, $\beta_h(\cdot)$, this requires a transition probability matrix for the walks on the grid $[\mathbf{P}]_{ij} = p_{ij}$
  3. E.g. the discrete Wiener integral solution to equation (1): $u_h(x) = E_x^h[g(\beta_h(\tau_{\partial\Omega_h}))]$
  4. In this case if one has elliptic regularity of $u(x)$ and a nonsingular discretization, Ω_h then $u_h(x) = u(x) + O(h^2)$

Discrete Wiener Integrals

- All of the theory for continuous sample path Wiener integrals mentioned above carries over to the discrete cases
  1. Can replace the continuous region Ω with a discretization Ω_h where h the characteristic discretization parameter
  2. Replace $\beta(\cdot)$ with random walks on Ω_h, $\beta_h(\cdot)$, this requires a transition probability matrix for the walks on the grid $[\mathbf{P}]_{ij} = p_{ij}$
  3. E.g. the discrete Wiener integral solution to equation (1): $u_h(x) = E_x^h[g(\beta_h(\tau_{\partial\Omega_h}))]$
  4. In this case if one has elliptic regularity of $u(x)$ and a nonsingular discretization, Ω_h then $u_h(x) = u(x) + O(h^2)$

Discrete Wiener Integrals

- All of the theory for continuous sample path Wiener integrals mentioned above carries over to the discrete cases
  1. Can replace the continuous region Ω with a discretization Ω_h where h the characteristic discretization parameter
  2. Replace $\beta(\cdot)$ with random walks on Ω_h, $\beta_h(\cdot)$, this requires a transition probability matrix for the walks on the grid $[\mathbf{P}]_{ij} = p_{ij}$
  3. E.g. the discrete Wiener integral solution to equation (1): $u_h(x) = E_x^h[g(\beta_h(\tau_{\partial\Omega_h}))]$
  4. In this case if one has elliptic regularity of $u(x)$ and a nonsingular discretization, Ω_h then $u_h(x) = u(x) + O(h^2)$

Discrete Wiener Integrals

- All of the theory for continuous sample path Wiener integrals mentioned above carries over to the discrete cases
  1. Can replace the continuous region Ω with a discretization Ω_h where h the characteristic discretization parameter
  2. Replace $\beta(\cdot)$ with random walks on Ω_h, $\beta_h(\cdot)$, this requires a transition probability matrix for the walks on the grid $[\mathbf{P}]_{ij} = p_{ij}$
  3. E.g. the discrete Wiener integral solution to equation (1): $u_h(x) = E_x^h[g(\beta_h(\tau_{\partial\Omega_h}))]$
  4. In this case if one has elliptic regularity of $u(x)$ and a nonsingular discretization, Ω_h then $u_h(x) = u(x) + O(h^2)$

Discrete Wiener Integrals

- All of the theory for continuous sample path Wiener integrals mentioned above carries over to the discrete cases
  1. Can replace the continuous region Ω with a discretization Ω_h where h the characteristic discretization parameter
  2. Replace $\beta(\cdot)$ with random walks on Ω_h, $\beta_h(\cdot)$, this requires a transition probability matrix for the walks on the grid $[\mathbf{P}]_{ij} = p_{ij}$
  3. E.g. the discrete Wiener integral solution to equation (1): $u_h(x) = E_x^h[g(\beta_h(\tau_{\partial\Omega_h}))]$
  4. In this case if one has elliptic regularity of $u(x)$ and a nonsingular discretization, Ω_h then $u_h(x) = u(x) + O(h^2)$

Parallel N-body Potential Evaluation

- N-body potential problems are common in biochemistry, stellar dynamics, fluid dynamics, materials simulation, the boundary element method for elliptic PDEs, etc.
- The solution of N-body potential problems requires evaluation of function of the form: $\Phi(\mathbf{x}) = \sum_{n=1}^{N} \phi(\mathbf{x} - \mathbf{x}_i)$ for all values of $\mathbf{x} = \mathbf{x}_j, j = 1, \dots, N$
- One heuristic solution is to replace $\phi(\mathbf{x})$ with a cutoff version $\phi_{co}(\mathbf{x}) = \phi(\mathbf{x})$ for $|\mathbf{x}| \le r$ and $\phi_{co}(\mathbf{x}) = 0$ otherwise, this reduces the problem to only keeping track of r-neighborhood points
- Can use the $\mathbf{x}_i$, $\mathbf{x}_j$ interaction as the basis for parallelism and use N^2 processors to calculate the $N(N-1)/2$ terms in parallel, initialization of the coordinates and accumulation the results requires $O(\log_2 N)$ operations

Parallel N-body Potential Evaluation

- N-body potential problems are common in biochemistry, stellar dynamics, fluid dynamics, materials simulation, the boundary element method for elliptic PDEs, etc.
- The solution of N-body potential problems requires evaluation of function of the form: $\Phi(\mathbf{x}) = \sum_{n=1}^{N} \phi(\mathbf{x} - \mathbf{x}_i)$ for all values of $\mathbf{x} = \mathbf{x}_j$, $j = 1, \ldots, N$
- One heuristic solution is to replace $\phi(\mathbf{x})$ with a cutoff version $\phi_{co}(\mathbf{x}) = \phi(\mathbf{x})$ for $|\mathbf{x}| \leq r$ and $\phi_{co}(\mathbf{x}) = 0$ otherwise, this reduces the problem to only keeping track of r-neighborhood points
- Can use the $\mathbf{x}_i$, $\mathbf{x}_j$ interaction as the basis for parallelism and use N^2 processors to calculate the $N(N-1)/2$ terms in parallel, initialization of the coordinates and accumulation the results requires $O(\log_2 N)$ operations

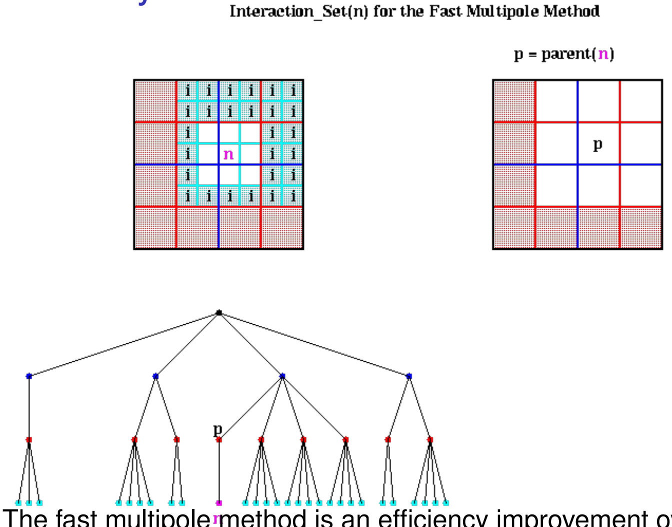
Parallel N-body Potential Evaluation

- N-body potential problems are common in biochemistry, stellar dynamics, fluid dynamics, materials simulation, the boundary element method for elliptic PDEs, etc.
- The solution of N-body potential problems requires evaluation of function of the form: $\Phi(\mathbf{x}) = \sum_{n=1}^{N} \phi(\mathbf{x} - \mathbf{x}_i)$ for all values of $\mathbf{x} = \mathbf{x}_j$, $j = 1, \dots, N$
- One heuristic solution is to replace $\phi(\mathbf{x})$ with a cutoff version $\phi_{co}(\mathbf{x}) = \phi(\mathbf{x})$ for $|\mathbf{x}| \leq r$ and $\phi_{co}(\mathbf{x}) = 0$ otherwise, this reduces the problem to only keeping track of r-neighborhood points
- Can use the $\mathbf{x}_i$, $\mathbf{x}_j$ interaction as the basis for parallelism and use N^2 processors to calculate the $N(N-1)/2$ terms in parallel, initialization of the coordinates and accumulation the results requires $O(\log_2 N)$ operations

Parallel N-body Potential Evaluation

- N-body potential problems are common in biochemistry, stellar dynamics, fluid dynamics, materials simulation, the boundary element method for elliptic PDEs, etc.
- The solution of N-body potential problems requires evaluation of function of the form: $\Phi(\mathbf{x}) = \sum_{n=1}^{N} \phi(\mathbf{x} - \mathbf{x}_i)$ for all values of $\mathbf{x} = \mathbf{x}_j$, $j = 1, \dots, N$
- One heuristic solution is to replace $\phi(\mathbf{x})$ with a cutoff version $\phi_{co}(\mathbf{x}) = \phi(\mathbf{x})$ for $|\mathbf{x}| \leq r$ and $\phi_{co}(\mathbf{x}) = 0$ otherwise, this reduces the problem to only keeping track of r-neighborhood points
- Can use the $\mathbf{x}_i$, $\mathbf{x}_j$ interaction as the basis for parallelism and use N^2 processors to calculate the $N(N-1)/2$ terms in parallel, initialization of the coordinates and accumulation the results requires $O(\log_2 N)$ operations

Parallel N-body Potential Evaluation



- The fast multipole method is an efficiency improvement over direct methods

The Rokhlin-Greengard Fast Multipole Method

- Algorithm is based on multipole expansion and some theory from complex variable series, consider the electrostatic description
- If $z \in \mathbb{C}$ then a charge of intensity q at z_0 results in a complex potential via Laurent series for $|z| > |z_0|$:

$$\phi_{z_0}(z) = q \ln(z - z_0) = q \left[\ln(z) - \sum_{k=1}^{\infty} \frac{1}{k} \left(\frac{z_0}{z} \right)^k \right]$$

1. If we have m charges q_i at locations z_i then the potential induced by them is given by the multipole expansion for $|z| > r = \max_i |z_i|$:

$$\begin{aligned} \phi(z) &= Q \ln(z) + \sum_{k=1}^{\infty} \frac{a_k}{z^k}, \\ \text{where } Q &= \sum_{i=1}^{m} q_i, \text{ and } a_k = \sum_{i=1}^{m} \frac{-q_i z_i^k}{k} \end{aligned}$$

The Rokhlin-Greengard Fast Multipole Method

- Algorithm is based on multipole expansion and some theory from complex variable series, consider the electrostatic description
- If $z \in \mathbb{C}$ then a charge of intensity q at z_0 results in a complex potential via Laurent series for $|z| > |z_0|$:

$$\phi_{z_0}(z) = q \ln(z - z_0) = q \left[\ln(z) - \sum_{k=1}^{\infty} \frac{1}{k} \left(\frac{z_0}{z} \right)^k \right]$$

1. If we have m charges q_i at locations z_i then the potential induced by them is given by the multipole expansion for $|z| > r = \max_i |z_i|$:

$$\begin{aligned} \phi(z) &= Q \ln(z) + \sum_{k=1}^{\infty} \frac{a_k}{z^k}, \\ \text{where } Q &= \sum_{i=1}^{m} q_i, \text{ and } a_k = \sum_{i=1}^{m} \frac{-q_i z_i^k}{k} \end{aligned}$$

The Rokhlin-Greengard Fast Multipole Method

- Algorithm is based on multipole expansion and some theory from complex variable series, consider the electrostatic description
- If $z \in \mathbb{C}$ then a charge of intensity q at z_0 results in a complex potential via Laurent series for $|z| > |z_0|$:

$$\phi_{z_0}(z) = q\ln(z - z_0) = q\left[\ln(z) - \sum_{k=1}^{\infty} \frac{1}{k}\left(\frac{z_0}{z}\right)^k\right]$$

1. If we have m charges q_i at locations z_i then the potential induced by them is given by the multipole expansion for $|z| > r = \max_i |z_i|$:

$$\begin{aligned} \phi(z) &= Q\ln(z) + \sum_{k=1}^{\infty} \frac{a_k}{z^k}, \\ \text{where } Q &= \sum_{i=1}^{m} q_i, \text{ and } a_k = \sum_{i=1}^{m} \frac{-q_i z_i^k}{k} \end{aligned}$$

The Rokhlin-Greengard Fast Multipole Method

- Given an accuracy, ϵ, one can truncate the multipole expansion to a fixed number, $p = \lceil -\log_c \epsilon \rceil$, of terms, where $c = |\frac{z}{r}|$
- With p determined one can store a multipole expansion as $\{a_1, a_2 \dots, a_p\}$

1. We can move a multipole's center from z_0 to the origin with new coefficients:

$$b_l = -\frac{Qz_0^l}{l} + \sum_{k=1}^{l} a_k z_0^{l-k} \binom{l-1}{k-1}$$

- Note that $\{a_1, a_2 \dots, a_p\}$ can be used to exactly compute $\{b_1, b_2 \dots, b_p\}$

The Rokhlin-Greengard Fast Multipole Method

- Given an accuracy, ϵ, one can truncate the multipole expansion to a fixed number, $p = \lceil -\log_c \epsilon \rceil$, of terms, where $c = |\frac{z}{r}|$
- With p determined one can store a multipole expansion as $\{a_1, a_2 \ldots, a_p\}$

1. We can move a multipole's center from z_0 to the origin with new coefficients:

$$b_l = -\frac{Qz_0^l}{l} + \sum_{k=1}^{l} a_k z_0^{l-k} \binom{l-1}{k-1}$$

- Note that $\{a_1, a_2 \ldots, a_p\}$ can be used to exactly compute $\{b_1, b_2 \ldots, b_p\}$

The Rokhlin-Greengard Fast Multipole Method

- Given an accuracy, ϵ, one can truncate the multipole expansion to a fixed number, $p = \lceil -\log_c \epsilon \rceil$, of terms, where $c = |\frac{z}{r}|$
- With p determined one can store a multipole expansion as $\{a_1, a_2 \ldots, a_p\}$

1. We can move a multipole's center from z_0 to the origin with new coefficients:

$$b_l = -\frac{Qz_0^l}{l} + \sum_{k=1}^{l} a_k z_0^{l-k} \binom{l-1}{k-1}$$

- Note that $\{a_1, a_2 \ldots, a_p\}$ can be used to exactly compute $\{b_1, b_2 \ldots, b_p\}$

The Rokhlin-Greengard Fast Multipole Method

- Given an accuracy, ϵ, one can truncate the multipole expansion to a fixed number, $p = \lceil -\log_c \epsilon \rceil$, of terms, where $c = |\frac{z}{r}|$
- With p determined one can store a multipole expansion as $\{a_1, a_2 \dots, a_p\}$

1. We can move a multipole's center from z_0 to the origin with new coefficients:

$$b_l = -\frac{Qz_0^l}{l} + \sum_{k=1}^{l} a_k z_0^{l-k} \binom{l-1}{k-1}$$

- Note that $\{a_1, a_2 \dots, a_p\}$ can be used to exactly compute $\{b_1, b_2 \dots, b_p\}$

The Rokhlin-Greengard Fast Multipole Method

1. Can also convert a multipole Laurent expansion into a local Taylor expansion:

$$\begin{aligned}\phi(z) &= \sum_{l=0}^{\infty} c_l z^l, \text{ where}\\ c_0 &= Q\ln(-z_0) + \sum_{k=0}^{\infty} \frac{a_k}{z_0^k}(-1)^k, \text{ and}\\ c_l &= -\frac{Q}{l\dot{z}_0^l} + \frac{1}{z_0^l}\sum_{k=1}^{\infty} \frac{a_k}{z_0^k}\binom{l+k-1}{k-1}(-1)^k\end{aligned}$$

2. And translate local expansions:

$$\sum_{k=0}^{n} a_k(z-z_0)^k = \sum_{l=0}^{n}\left(\sum_{k=l}^{n} a_k\binom{k}{l}(-z_0)^{k-l}\right)z^l$$

The Rokhlin-Greengard Fast Multipole Method

1. Can also convert a multipole Laurent expansion into a local Taylor expansion:

$$\begin{aligned}
\phi(z) &= \sum_{l=0}^{\infty} c_l z^l, \text{ where} \\
c_0 &= Q\ln(-z_0) + \sum_{k=0}^{\infty} \frac{a_k}{z_0^k}(-1)^k, \text{ and} \\
c_l &= -\frac{Q}{l\dot{z}_0^l} + \frac{1}{z_0^l}\sum_{k=1}^{\infty}\frac{a_k}{z_0^k}\binom{l+k-1}{k-1}(-1)^k
\end{aligned}$$

2. And translate local expansions:

$$\sum_{k=0}^{n} a_k (z-z_0)^k = \sum_{l=0}^{n}\left(\sum_{k=l}^{n} a_k \binom{k}{l}(-z_0)^{k-l}\right) z^l$$

The Rokhlin-Greengard Fast Multipole Method

- Items (1)-(4) above are the machinery required to allow the construction and use of multipole expansions, and given a multipole expansion, it requires $O(N)$ operations to evaluate it at N points, thus an algorithm for the construction of a multipole expansion from N point charges that requires $O(N)$ operations reduces the complexity of N-body problems to $O(N)$ complexity
- The Rokhlin-Greengard algorithm achieves this by using a multiscale approach and (1)-(4)
- Consider a box enclosing $z_1, z_2, \dots, z_N$, and $n \approx \lceil \log_4 N \rceil$ refinements of the box, in 2D one parent box becomes four children boxes

The Rokhlin-Greengard Fast Multipole Method

- Items (1)-(4) above are the machinery required to allow the construction and use of multipole expansions, and given a multipole expansion, it requires $O(N)$ operations to evaluate it at N points, thus an algorithm for the construction of a multipole expansion from N point charges that requires $O(N)$ operations reduces the complexity of N-body problems to $O(N)$ complexity
- The Rokhlin-Greengard algorithm achieves this by using a multiscale approach and (1)-(4)
- Consider a box enclosing $z_1, z_2, \dots, z_N$, and $n \approx \lceil \log_4 N \rceil$ refinements of the box, in 2D one parent box becomes four children boxes

The Rokhlin-Greengard Fast Multipole Method

- Items (1)-(4) above are the machinery required to allow the construction and use of multipole expansions, and given a multipole expansion, it requires $O(N)$ operations to evaluate it at N points, thus an algorithm for the construction of a multipole expansion from N point charges that requires $O(N)$ operations reduces the complexity of N-body problems to $O(N)$ complexity
- The Rokhlin-Greengard algorithm achieves this by using a multiscale approach and (1)-(4)
- Consider a box enclosing $z_1, z_2, \dots, z_N$, and $n \approx \lceil \log_4 N \rceil$ refinements of the box, in 2D one parent box becomes four children boxes

The Rokhlin-Greengard Fast Multipole Method

- Goal is to construct all p-term multipole expansions due to the particles in each box and level (upward pass) and then use these to construct local expansions in each box and level (downward pass)
  1. The upward pass:
     - At the finest level construct box-centered p-term multipole expansions due to the particles in each box using (1)
     - At each coarser level shift child p-term multipole expansions to build box-centered p-term multipole expansions due to the particles in the parent boxes using (2)
  2. The downward pass:
     - Construct local Taylor expansion in each box at the finest level by converting the p-term multipole expansions of boxes in the "interaction list" via (3)

The Rokhlin-Greengard Fast Multipole Method

- Goal is to construct all p-term multipole expansions due to the particles in each box and level (upward pass) and then use these to construct local expansions in each box and level (downward pass)
  1. The upward pass:
     - At the finest level construct box-centered p-term multipole expansions due to the particles in each box using (1)
     - At each coarser level shift child p-term multipole expansions to build box-centered p-term multipole expansions due to the particles in the parent boxes using (2)
  2. The downward pass:
     - Construct local Taylor expansion in each box at the finest level by converting the p-term multipole expansions of boxes in the "interaction list" via (3)

The Rokhlin-Greengard Fast Multipole Method

- Goal is to construct all p-term multipole expansions due to the particles in each box and level (upward pass) and then use these to construct local expansions in each box and level (downward pass)
  1. The upward pass:
     - At the finest level construct box-centered p-term multipole expansions due to the particles in each box using (1)
     - At each coarser level shift child p-term multipole expansions to build box-centered p-term multipole expansions due to the particles in the parent boxes using (2)
  2. The downward pass:
     - Construct local Taylor expansion in each box at the finest level by converting the p-term multipole expansions of boxes in the “interaction list” via (3)

The Rokhlin-Greengard Fast Multipole Method

- Goal is to construct all p-term multipole expansions due to the particles in each box and level (upward pass) and then use these to construct local expansions in each box and level (downward pass)
  1. The upward pass:
     - At the finest level construct box-centered p-term multipole expansions due to the particles in each box using (1)
     - At each coarser level shift child p-term multipole expansions to build box-centered p-term multipole expansions due to the particles in the parent boxes using (2)
  2. The downward pass:
     - Construct local Taylor expansion in each box at the finest level by converting the p-term multipole expansions of boxes in the "interaction list" via (3)

The Rokhlin-Greengard Fast Multipole Method

- Goal is to construct all p-term multipole expansions due to the particles in each box and level (upward pass) and then use these to construct local expansions in each box and level (downward pass)
  1. The upward pass:
     - At the finest level construct box-centered p-term multipole expansions due to the particles in each box using (1)
     - At each coarser level shift child p-term multipole expansions to build box-centered p-term multipole expansions due to the particles in the parent boxes using (2)
  2. The downward pass:
     - Construct local Taylor expansion in each box at the finest level by converting the p-term multipole expansions of boxes in the "interaction list" via (3)

The Rokhlin-Greengard Fast Multipole Method

- Goal is to construct all p-term multipole expansions due to the particles in each box and level (upward pass) and then use these to construct local expansions in each box and level (downward pass)
  1. The upward pass:
     - At the finest level construct box-centered p-term multipole expansions due to the particles in each box using (1)
     - At each coarser level shift child p-term multipole expansions to build box-centered p-term multipole expansions due to the particles in the parent boxes using (2)
  2. The downward pass:
     - Construct local Taylor expansion in each box at the finest level by converting the p-term multipole expansions of boxes in the “interaction list” via (3)

The Rokhlin-Greengard Fast Multipole Method

1. The downward pass (cont.):
   - Via (4) add these together to get local box-centered expansions for all the particles outside the box's neighborhood using coarser level *p*-term multipole expansions
   - Evaluate the above local expansions at each particle and add in the directly compute nearest-neighbor interactions
   - This algorithm is overall $O(N)$ complex and has many parallel aspects

The Rokhlin-Greengard Fast Multipole Method

1. The downward pass (cont.):
   - Via (4) add these together to get local box-centered expansions for all the particles outside the box's neighborhood using coarser level *p*-term multipole expansions
   - Evaluate the above local expansions at each particle and add in the directly compute nearest-neighbor interactions
   - This algorithm is overall $O(N)$ complex and has many parallel aspects

The Rokhlin-Greengard Fast Multipole Method

1. The downward pass (cont.):
   - Via (4) add these together to get local box-centered expansions for all the particles outside the box's neighborhood using coarser level p-term multipole expansions
   - Evaluate the above local expansions at each particle and add in the directly compute nearest-neighbor interactions
   - This algorithm is overall $O(N)$ complex and has many parallel aspects

The Rokhlin-Greengard Fast Multipole Method

1. The downward pass (cont.):
    - Via (4) add these together to get local box-centered expansions for all the particles outside the box's neighborhood using coarser level *p*-term multipole expansions
    - Evaluate the above local expansions at each particle and add in the directly compute nearest-neighbor interactions
    - This algorithm is overall $O(N)$ complex and has many parallel aspects

The Rokhlin-Greengard Fast Multipole Method

- Need only store the *p*-term multipole/local Taylor expansion coefficients $\{a_1, a_2, \dots, a_p\}$, giving a trivial data structure with which to implementationally deal
- This version of the multipole algorithm depends on rather uniform distribution of particles, however there is an adaptive version where the “finest boxes” are allowed to be different sizes s.t. each box has approximately one particle

The Rokhlin-Greengard Fast Multipole Method

- Need only store the p-term multipole/local Taylor expansion coefficients $\{a_1, a_2, \dots, a_p\}$, giving a trivial data structure with which to implementationally deal
- This version of the multipole algorithm depends on rather uniform distribution of particles, however there is an adaptive version where the “finest boxes” are allowed to be different sizes s.t. each box has approximately one particle

Parallel Implementation of the Fast Multipole Method

- In all the steps described in both the upward and downward passes of the multipole method the multipole or local expansions can all be done in parallel
- Thus the parallel complexity is proportional to the number of levels, i.e. $O(\log_4 N) = O(n)$
- For the nonadaptive version there a serious load balancing problem due to whether at the each level, l, each box contains exactly N/n^l particles
- This is a multigrid algorithm and so there will be the problem of idle processors at coarse levels on parallel machines with fine grainsize
- One can also implement the adaptive version of the multipole method in parallel
- For more detail on the parallel version see (Greengard and Gropp, 1990)

Parallel Implementation of the Fast Multipole Method

- In all the steps described in both the upward and downward passes of the multipole method the multipole or local expansions can all be done in parallel
- Thus the parallel complexity is proportional to the number of levels, i.e. $O(\log_4 N) = O(n)$
- For the nonadaptive version there a serious load balancing problem due to whether at the each level, l, each box contains exactly N/n^l particles
- This is a multigrid algorithm and so there will be the problem of idle processors at coarse levels on parallel machines with fine grainsize
- One can also implement the adaptive version of the multipole method in parallel
- For more detail on the parallel version see (Greengard and Gropp, 1990)

Parallel Implementation of the Fast Multipole Method

- In all the steps described in both the upward and downward passes of the multipole method the multipole or local expansions can all be done in parallel
- Thus the parallel complexity is proportional to the number of levels, i.e. $O(\log_4 N) = O(n)$
- For the nonadaptive version there a serious load balancing problem due to whether at the each level, l, each box contains exactly N/n^l particles
- This is a multigrid algorithm and so there will be the problem of idle processors at coarse levels on parallel machines with fine grainsize
- One can also implement the adaptive version of the multipole method in parallel
- For more detail on the parallel version see (Greengard and Gropp, 1990)

Parallel Implementation of the Fast Multipole Method

- In all the steps described in both the upward and downward passes of the multipole method the multipole or local expansions can all be done in parallel
- Thus the parallel complexity is proportional to the number of levels, i.e. $O(\log_4 N) = O(n)$
- For the nonadaptive version there a serious load balancing problem due to whether at the each level, l, each box contains exactly N/n^l particles
- This is a multigrid algorithm and so there will be the problem of idle processors at coarse levels on parallel machines with fine grainsize
- One can also implement the adaptive version of the multipole method in parallel
- For more detail on the parallel version see (Greengard and Gropp, 1990)

Parallel Implementation of the Fast Multipole Method

- In all the steps described in both the upward and downward passes of the multipole method the multipole or local expansions can all be done in parallel
- Thus the parallel complexity is proportional to the number of levels, i.e. $O(\log_4 N) = O(n)$
- For the nonadaptive version there a serious load balancing problem due to whether at the each level, l, each box contains exactly N/n^l particles
- This is a multigrid algorithm and so there will be the problem of idle processors at coarse levels on parallel machines with fine grainsize
- One can also implement the adaptive version of the multipole method in parallel
- For more detail on the parallel version see (Greengard and Gropp, 1990)

Parallel Implementation of the Fast Multipole Method

- In all the steps described in both the upward and downward passes of the multipole method the multipole or local expansions can all be done in parallel
- Thus the parallel complexity is proportional to the number of levels, i.e. $O(\log_4 N) = O(n)$
- For the nonadaptive version there a serious load balancing problem due to whether at the each level, l, each box contains exactly N/n^l particles
- This is a multigrid algorithm and so there will be the problem of idle processors at coarse levels on parallel machines with fine grainsize
- One can also implement the adaptive version of the multipole method in parallel
- For more detail on the parallel version see (Greengard and Gropp, 1990)

Bibliography

Booth, T. E. (1981) “Exact Monte Carlo solutions of elliptic partial differential equations," *J. Comp. Phys.*, **39**: 396–404.

Brandt A. (1977) “Multi-level adaptive solutions to boundary value problems," *Math. Comp.*, **31**: 333–390.

Chorin, A. J. (1973) "Numerical study of slightly viscous flow," *J. Fluid Mech.*, **57**: 785–796.

Chorin, Alexandre J. and Hald, Ole H. (2006) *Stochastic Tools in Mathematics and Science*, Surveys and Tutorials in the Applied Mathematical Sciences, Vol. 1, viii+147, Springer, New York.

Courant, R., K. O. Friedrichs, and H. Lewy (1928) “Über die partiellen Differenzengleichungen der mathematischen Physik," *Math. Ann.*, **100**: 32–74 (In German), Reprinted in *I.B.M. J. Res. and Dev.*, **11**: 215–234 (In English).

Curtiss, J. H. (1953) “Monte Carlo methods for the iteration of linear operators," *J. Math. and Phys.*, **32**: 209–232.

Bibliography

- Booth, T. E. (1981) “Exact Monte Carlo solutions of elliptic partial differential equations," *J. Comp. Phys.*, **39**: 396–404.
- Brandt A. (1977) “Multi-level adaptive solutions to boundary value problems," *Math. Comp.*, **31**: 333–390.
- Chorin, A. J. (1973) "Numerical study of slightly viscous flow," *J. Fluid Mech.*, **57**: 785–796.
- Chorin, Alexandre J. and Hald, Ole H. (2006) *Stochastic Tools in Mathematics and Science*, Surveys and Tutorials in the Applied Mathematical Sciences, Vol. 1, viii+147, Springer, New York.
- Courant, R., K. O. Friedrichs, and H. Lewy (1928) “Über die partiellen Differenzengleichungen der mathematischen Physik," *Math. Ann.*, **100**: 32–74 (In German), Reprinted in *I.B.M. J. Res. and Dev.*, **11**: 215–234 (In English).
- Curtiss, J. H. (1953) “Monte Carlo methods for the iteration of linear operators," *J. Math. and Phys.*, **32**: 209–232.

Bibliography

- Booth, T. E. (1981) “Exact Monte Carlo solutions of elliptic partial differential equations," *J. Comp. Phys.*, **39**: 396–404.
- Brandt A. (1977) “Multi-level adaptive solutions to boundary value problems," *Math. Comp.*, **31**: 333–390.
- Chorin, A. J. (1973) "Numerical study of slightly viscous flow," *J. Fluid Mech.*, **57**: 785–796.
- Chorin, Alexandre J. and Hald, Ole H. (2006) *Stochastic Tools in Mathematics and Science*, Surveys and Tutorials in the Applied Mathematical Sciences, Vol. 1, viii+147, Springer, New York.
- Courant, R., K. O. Friedrichs, and H. Lewy (1928) “Über die partiellen Differenzengleichungen der mathematischen Physik," *Math. Ann.*, **100**: 32–74 (In German), Reprinted in *I.B.M. J. Res. and Dev.*, **11**: 215–234 (In English).
- Curtiss, J. H. (1953) “Monte Carlo methods for the iteration of linear operators," *J. Math. and Phys.*, **32**: 209–232.

Bibliography

- Booth, T. E. (1981) “Exact Monte Carlo solutions of elliptic partial differential equations," *J. Comp. Phys.*, **39**: 396–404.
- Brandt A. (1977) “Multi-level adaptive solutions to boundary value problems," *Math. Comp.*, **31**: 333–390.
- Chorin, A. J. (1973) "Numerical study of slightly viscous flow," *J. Fluid Mech.*, **57**: 785–796.
- Chorin, Alexandre J. and Hald, Ole H. (2006) *Stochastic Tools in Mathematics and Science*, Surveys and Tutorials in the Applied Mathematical Sciences, Vol. 1, viii+147, Springer, New York.
- Courant, R., K. O. Friedrichs, and H. Lewy (1928) “Über die partiellen Differenzengleichungen der mathematischen Physik," *Math. Ann.*, **100**: 32–74 (In German), Reprinted in *I.B.M. J. Res. and Dev.*, **11**: 215–234 (In English).
- Curtiss, J. H. (1953) “Monte Carlo methods for the iteration of linear operators," *J. Math. and Phys.*, **32**: 209–232.

Bibliography

- Booth, T. E. (1981) “Exact Monte Carlo solutions of elliptic partial differential equations," *J. Comp. Phys.*, **39**: 396–404.
- Brandt A. (1977) “Multi-level adaptive solutions to boundary value problems," *Math. Comp.*, **31**: 333–390.
- Chorin, A. J. (1973) "Numerical study of slightly viscous flow," *J. Fluid Mech.*, **57**: 785–796.
- Chorin, Alexandre J. and Hald, Ole H. (2006) *Stochastic Tools in Mathematics and Science*, Surveys and Tutorials in the Applied Mathematical Sciences, Vol. 1, viii+147, Springer, New York.
- Courant, R., K. O. Friedrichs, and H. Lewy (1928) “Über die partiellen Differenzengleichungen der mathematischen Physik," *Math. Ann.*, **100**: 32–74 (In German), Reprinted in *I.B.M. J. Res. and Dev.*, **11**: 215–234 (In English).
- Curtiss, J. H. (1953) “Monte Carlo methods for the iteration of linear operators," *J. Math. and Phys.*, **32**: 209–232.

Bibliography

- Booth, T. E. (1981) “Exact Monte Carlo solutions of elliptic partial differential equations," *J. Comp. Phys.*, **39**: 396–404.
- Brandt A. (1977) “Multi-level adaptive solutions to boundary value problems," *Math. Comp.*, **31**: 333–390.
- Chorin, A. J. (1973) "Numerical study of slightly viscous flow," *J. Fluid Mech.*, **57**: 785–796.
- Chorin, Alexandre J. and Hald, Ole H. (2006) *Stochastic Tools in Mathematics and Science*, Surveys and Tutorials in the Applied Mathematical Sciences, Vol. 1, viii+147, Springer, New York.
- Courant, R., K. O. Friedrichs, and H. Lewy (1928) “Über die partiellen Differenzengleichungen der mathematischen Physik," *Math. Ann.*, **100**: 32–74 (In German), Reprinted in *I.B.M. J. Res. and Dev.*, **11**: 215–234 (In English).
- Curtiss, J. H. (1953) “Monte Carlo methods for the iteration of linear operators," *J. Math. and Phys.*, **32**: 209–232.

Bibliography

Curtiss, J. H. (1956) “A theoretical comparison of the efficiencies of two classical methods and a Monte Carlo method for computing one component of the solution of a set of linear algebraic equations," in *Symp. Monte Carlo Methods*, H. A. Meyer, Ed., Wiley: New York, pp. 191–233.

Donsker, M. D., and M. Kac (1951) “A sampling method for determining the lowest eigenvalue and the principle eigenfunction of Schrödinger’s equation," *J. Res. Nat. Bur. Standards*, **44**: 551–557.

Ermakov, S. M. and G. A. Mikhailov (1982) *Statistical Modeling*, Nauka, Moscow, (in Russian).

Ermakov, S. M., V. V. Nekrutkin and A. S. Sipin (1989) *Random Processes for Classical Equations of Mathematical Physics*, Kluwer Academic Publishers: Dordrecht.

Forsythe, G. E., and R. A. Leibler (1950) “Matrix inversion by a Monte Carlo method," *Math. Tab. Aids Comput.*, **4**: 127–129.

Bibliography

Curtiss, J. H. (1956) “A theoretical comparison of the efficiencies of two classical methods and a Monte Carlo method for computing one component of the solution of a set of linear algebraic equations," in *Symp. Monte Carlo Methods*, H. A. Meyer, Ed., Wiley: New York, pp. 191–233.

Donsker, M. D., and M. Kac (1951) “A sampling method for determining the lowest eigenvalue and the principle eigenfunction of Schrödinger’s equation," *J. Res. Nat. Bur. Standards*, **44**: 551–557.

Ermakov, S. M. and G. A. Mikhailov (1982) *Statistical Modeling*, Nauka, Moscow, (in Russian).

Ermakov, S. M., V. V. Nekrutkin and A. S. Sipin (1989) *Random Processes for Classical Equations of Mathematical Physics*, Kluwer Academic Publishers: Dordrecht.

Forsythe, G. E., and R. A. Leibler (1950) “Matrix inversion by a Monte Carlo method," *Math. Tab. Aids Comput.*, **4**: 127–129.

Bibliography

Curtiss, J. H. (1956) “A theoretical comparison of the efficiencies of two classical methods and a Monte Carlo method for computing one component of the solution of a set of linear algebraic equations," in *Symp. Monte Carlo Methods*, H. A. Meyer, Ed., Wiley: New York, pp. 191–233.

Donsker, M. D., and M. Kac (1951) “A sampling method for determining the lowest eigenvalue and the principle eigenfunction of Schrödinger’s equation," *J. Res. Nat. Bur. Standards*, **44**: 551–557.

Ermakov, S. M. and G. A. Mikhailov (1982) *Statistical Modeling*, Nauka, Moscow, (in Russian).

Ermakov, S. M., V. V. Nekrutkin and A. S. Sipin (1989) *Random Processes for Classical Equations of Mathematical Physics*, Kluwer Academic Publishers: Dordrecht.

Forsythe, G. E., and R. A. Leibler (1950) “Matrix inversion by a Monte Carlo method," *Math. Tab. Aids Comput.*, **4**: 127–129.

Bibliography

Curtiss, J. H. (1956) “A theoretical comparison of the efficiencies of two classical methods and a Monte Carlo method for computing one component of the solution of a set of linear algebraic equations," in *Symp. Monte Carlo Methods*, H. A. Meyer, Ed., Wiley: New York, pp. 191–233.

Donsker, M. D., and M. Kac (1951) “A sampling method for determining the lowest eigenvalue and the principle eigenfunction of Schrödinger’s equation," *J. Res. Nat. Bur. Standards*, **44**: 551–557.

Ermakov, S. M. and G. A. Mikhailov (1982) *Statistical Modeling*, Nauka, Moscow, (in Russian).

Ermakov, S. M., V. V. Nekrutkin and A. S. Sipin (1989) *Random Processes for Classical Equations of Mathematical Physics*, Kluwer Academic Publishers: Dordrecht.

Forsythe, G. E., and R. A. Leibler (1950) “Matrix inversion by a Monte Carlo method," *Math. Tab. Aids Comput.*, **4**: 127–129.

Bibliography

Curtiss, J. H. (1956) “A theoretical comparison of the efficiencies of two classical methods and a Monte Carlo method for computing one component of the solution of a set of linear algebraic equations," in *Symp. Monte Carlo Methods*, H. A. Meyer, Ed., Wiley: New York, pp. 191–233.

Donsker, M. D., and M. Kac (1951) “A sampling method for determining the lowest eigenvalue and the principle eigenfunction of Schrödinger’s equation," *J. Res. Nat. Bur. Standards*, **44**: 551–557.

Ermakov, S. M. and G. A. Mikhailov (1982) *Statistical Modeling*, Nauka, Moscow, (in Russian).

Ermakov, S. M., V. V. Nekrutkin and A. S. Sipin (1989) *Random Processes for Classical Equations of Mathematical Physics*, Kluwer Academic Publishers: Dordrecht.

Forsythe, G. E., and R. A. Leibler (1950) “Matrix inversion by a Monte Carlo method," *Math. Tab. Aids Comput.*, **4**: 127–129.

Bibliography

Freidlin, M. (1985) *Functional Integration and Partial Differential Equations*, Princeton University Press: Princeton.

Ghoniem, A. F. and F. S. Sherman (1985) “Grid-free simulation of diffusion using random walk methods," *J. Comp. Phys.*, **61**: 1-37.

Greengard, L. F. (1988) *The Rapid Evaluation of Potential Fields in Particle Systems*, MIT Press: Cambridge, MA.

Greengard, L. F. and W. Gropp (1990) “A parallel version of the fast multipole method," *Computers Math. Applic.*, **20**: 63-71.

Hall, A. (1873) “On an experimental determination of π," *Messeng. Math.*, **2**: 113–114.

Halton, J. H. (1970), “A Retrospective and Prospective Survey of the Monte Carlo Method," *SIAM Review*, **12(1)**: 1–63,.

Hammersley, J. M., and D. C. Handscomb (1964) *Monte Carlo Methods*, Chapman and Hall, London.

Bibliography

Freidlin, M. (1985) *Functional Integration and Partial Differential Equations*, Princeton University Press: Princeton.

Ghoniem, A. F. and F. S. Sherman (1985) “Grid-free simulation of diffusion using random walk methods," *J. Comp. Phys.*, **61**: 1-37.

Greengard, L. F. (1988) *The Rapid Evaluation of Potential Fields in Particle Systems*, MIT Press: Cambridge, MA.

Greengard, L. F. and W. Gropp (1990) “A parallel version of the fast multipole method," *Computers Math. Applic.*, **20**: 63-71.

Hall, A. (1873) “On an experimental determination of π," *Messeng. Math.*, **2**: 113–114.

Halton, J. H. (1970), “A Retrospective and Prospective Survey of the Monte Carlo Method," *SIAM Review*, **12(1)**: 1–63,.

Hammersley, J. M., and D. C. Handscomb (1964) *Monte Carlo Methods*, Chapman and Hall, London.

Bibliography

Freidlin, M. (1985) *Functional Integration and Partial Differential Equations*, Princeton University Press: Princeton.

Ghoniem, A. F. and F. S. Sherman (1985) “Grid-free simulation of diffusion using random walk methods," *J. Comp. Phys.*, **61**: 1-37.

Greengard, L. F. (1988) *The Rapid Evaluation of Potential Fields in Particle Systems*, MIT Press: Cambridge, MA.

Greengard, L. F. and W. Gropp (1990) “A parallel version of the fast multipole method," *Computers Math. Applic.*, **20**: 63-71.

Hall, A. (1873) “On an experimental determination of π," *Messeng. Math.*, **2**: 113–114.

Halton, J. H. (1970), “A Retrospective and Prospective Survey of the Monte Carlo Method," *SIAM Review*, **12(1)**: 1–63,.

Hammersley, J. M., and D. C. Handscomb (1964) *Monte Carlo Methods*, Chapman and Hall, London.

Bibliography

- Freidlin, M. (1985) *Functional Integration and Partial Differential Equations*, Princeton University Press: Princeton.
- Ghoniem, A. F. and F. S. Sherman (1985) “Grid-free simulation of diffusion using random walk methods," *J. Comp. Phys.*, **61**: 1-37.
- Greengard, L. F. (1988) *The Rapid Evaluation of Potential Fields in Particle Systems*, MIT Press: Cambridge, MA.
- Greengard, L. F. and W. Gropp (1990) “A parallel version of the fast multipole method," *Computers Math. Applic.*, **20**: 63-71.
- Hall, A. (1873) “On an experimental determination of π," *Messeng. Math.*, **2**: 113–114.
- Halton, J. H. (1970), “A Retrospective and Prospective Survey of the Monte Carlo Method," *SIAM Review*, **12(1)**: 1–63,.
- Hammersley, J. M., and D. C. Handscomb (1964) *Monte Carlo Methods*, Chapman and Hall, London.

Bibliography

- Freidlin, M. (1985) *Functional Integration and Partial Differential Equations*, Princeton University Press: Princeton.
- Ghoniem, A. F. and F. S. Sherman (1985) “Grid-free simulation of diffusion using random walk methods," *J. Comp. Phys.*, **61**: 1-37.
- Greengard, L. F. (1988) *The Rapid Evaluation of Potential Fields in Particle Systems*, MIT Press: Cambridge, MA.
- Greengard, L. F. and W. Gropp (1990) “A parallel version of the fast multipole method," *Computers Math. Applic.*, **20**: 63-71.
- Hall, A. (1873) “On an experimental determination of π," *Messeng. Math.*, **2**: 113–114.
- Halton, J. H. (1970), “A Retrospective and Prospective Survey of the Monte Carlo Method," *SIAM Review*, **12(1)**: 1–63,.
- Hammersley, J. M., and D. C. Handscomb (1964) *Monte Carlo Methods*, Chapman and Hall, London.

Bibliography

- Freidlin, M. (1985) *Functional Integration and Partial Differential Equations*, Princeton University Press: Princeton.
- Ghoniem, A. F. and F. S. Sherman (1985) “Grid-free simulation of diffusion using random walk methods," *J. Comp. Phys.*, **61**: 1-37.
- Greengard, L. F. (1988) *The Rapid Evaluation of Potential Fields in Particle Systems*, MIT Press: Cambridge, MA.
- Greengard, L. F. and W. Gropp (1990) “A parallel version of the fast multipole method," *Computers Math. Applic.*, **20**: 63-71.
- Hall, A. (1873) “On an experimental determination of π," *Messeng. Math.*, **2**: 113–114.
- Halton, J. H. (1970), “A Retrospective and Prospective Survey of the Monte Carlo Method," *SIAM Review*, **12(1)**: 1–63,.
- Hammersley, J. M., and D. C. Handscomb (1964) *Monte Carlo Methods*, Chapman and Hall, London.

Bibliography

- Freidlin, M. (1985) *Functional Integration and Partial Differential Equations*, Princeton University Press: Princeton.
- Ghoniem, A. F. and F. S. Sherman (1985) “Grid-free simulation of diffusion using random walk methods," *J. Comp. Phys.*, **61**: 1-37.
- Greengard, L. F. (1988) *The Rapid Evaluation of Potential Fields in Particle Systems*, MIT Press: Cambridge, MA.
- Greengard, L. F. and W. Gropp (1990) “A parallel version of the fast multipole method," *Computers Math. Applic.*, **20**: 63-71.
- Hall, A. (1873) “On an experimental determination of π," *Messeng. Math.*, **2**: 113–114.
- Halton, J. H. (1970), “A Retrospective and Prospective Survey of the Monte Carlo Method," *SIAM Review*, **12(1)**: 1–63,.
- Hammersley, J. M., and D. C. Handscomb (1964) *Monte Carlo Methods*, Chapman and Hall, London.

Bibliography

Halton, J. H. (1989) “Pseudo-random trees: multiple independent sequence generators for parallel and branching computations," *J. Comp. Phys.*, **84**: 1–56.

Hillis, D. (1985) *The Connection Machine*, M.I.T. University Press: Cambridge, MA.

Hopf, E. (1950) “The partial differential equation $u_t + uu_x = \mu_{xx}$," *Comm. Pure Applied Math.*, **3**: 201–230.

Itô, K. and H. P. McKean, Jr. (1965) *Diffusion Processes and Their Sample Paths*, Springer-Verlag: Berlin, New York.

Kac, M. (1947) “Random Walk and the Theory of Brownian Motion,"*The American Mathematical Monthly*, **54(7)**: 369–391.

Kac, M. (1956) *Some Stochastic Problems in Physics and Mathematics*, Colloquium Lectures in the Pure and Applied Sciences, No. 2, Magnolia Petroleum Co., Hectographed.

Kac, M. (1980) *Integration in Function Spaces and Some of Its Applications*, Lezioni Fermiane, Accademia Nazionale Dei Lincei Scuola Normale Superiore, Pisa.

Bibliography

- Halton, J. H. (1989) “Pseudo-random trees: multiple independent sequence generators for parallel and branching computations," *J. Comp. Phys.*, **84**: 1–56.
- Hillis, D. (1985) *The Connection Machine*, M.I.T. University Press: Cambridge, MA.
- Hopf, E. (1950) “The partial differential equation $u_t + uu_x = \mu_{xx}$," *Comm. Pure Applied Math.*, **3**: 201–230.
- Itô, K. and H. P. McKean, Jr. (1965) *Diffusion Processes and Their Sample Paths*, Springer-Verlag: Berlin, New York.
- Kac, M. (1947) “Random Walk and the Theory of Brownian Motion,"*The American Mathematical Monthly*, **54(7)**: 369–391.
- Kac, M. (1956) *Some Stochastic Problems in Physics and Mathematics*, Colloquium Lectures in the Pure and Applied Sciences, No. 2, Magnolia Petroleum Co., Hectographed.
- Kac, M. (1980) *Integration in Function Spaces and Some of Its Applications*, Lezioni Fermiane, Accademia Nazionale Dei Lincei Scuola Normale Superiore, Pisa.

Bibliography

Halton, J. H. (1989) “Pseudo-random trees: multiple independent sequence generators for parallel and branching computations," *J. Comp. Phys.*, **84**: 1–56.

Hillis, D. (1985) *The Connection Machine*, M.I.T. University Press: Cambridge, MA.

Hopf, E. (1950) “The partial differential equation $u_t + uu_x = \mu_{xx}$," *Comm. Pure Applied Math.*, **3**: 201–230.

Itô, K. and H. P. McKean, Jr. (1965) *Diffusion Processes and Their Sample Paths*, Springer-Verlag: Berlin, New York.

Kac, M. (1947) “Random Walk and the Theory of Brownian Motion,"*The American Mathematical Monthly*, **54(7)**: 369–391.

Kac, M. (1956) *Some Stochastic Problems in Physics and Mathematics*, Colloquium Lectures in the Pure and Applied Sciences, No. 2, Magnolia Petroleum Co., Hectographed.

Kac, M. (1980) *Integration in Function Spaces and Some of Its Applications*, Lezioni Fermiane, Accademia Nazionale Dei Lincei Scuola Normale Superiore, Pisa.

Bibliography

- Halton, J. H. (1989) “Pseudo-random trees: multiple independent sequence generators for parallel and branching computations," *J. Comp. Phys.*, **84**: 1–56.
- Hillis, D. (1985) *The Connection Machine*, M.I.T. University Press: Cambridge, MA.
- Hopf, E. (1950) “The partial differential equation $u_t + uu_x = \mu_{xx}$," *Comm. Pure Applied Math.*, **3**: 201–230.
- Itô, K. and H. P. McKean, Jr. (1965) *Diffusion Processes and Their Sample Paths*, Springer-Verlag: Berlin, New York.
- Kac, M. (1947) “Random Walk and the Theory of Brownian Motion,"*The American Mathematical Monthly*, **54(7)**: 369–391.
- Kac, M. (1956) *Some Stochastic Problems in Physics and Mathematics*, Colloquium Lectures in the Pure and Applied Sciences, No. 2, Magnolia Petroleum Co., Hectographed.
- Kac, M. (1980) *Integration in Function Spaces and Some of Its Applications*, Lezioni Fermiane, Accademia Nazionale Dei Lincei Scuola Normale Superiore, Pisa.

Bibliography

Halton, J. H. (1989) “Pseudo-random trees: multiple independent sequence generators for parallel and branching computations," *J. Comp. Phys.*, **84**: 1–56.

Hillis, D. (1985) *The Connection Machine*, M.I.T. University Press: Cambridge, MA.

Hopf, E. (1950) “The partial differential equation $u_t + uu_x = \mu_{xx}$," *Comm. Pure Applied Math.*, **3**: 201–230.

Itô, K. and H. P. McKean, Jr. (1965) *Diffusion Processes and Their Sample Paths*, Springer-Verlag: Berlin, New York.

Kac, M. (1947) “Random Walk and the Theory of Brownian Motion,"*The American Mathematical Monthly*, **54(7)**: 369–391.

Kac, M. (1956) *Some Stochastic Problems in Physics and Mathematics*, Colloquium Lectures in the Pure and Applied Sciences, No. 2, Magnolia Petroleum Co., Hectographed.

Kac, M. (1980) *Integration in Function Spaces and Some of Its Applications*, Lezioni Fermiane, Accademia Nazionale Dei Lincei Scuola Normale Superiore, Pisa.

Bibliography

- Halton, J. H. (1989) “Pseudo-random trees: multiple independent sequence generators for parallel and branching computations," *J. Comp. Phys.*, **84**: 1–56.
- Hillis, D. (1985) *The Connection Machine*, M.I.T. University Press: Cambridge, MA.
- Hopf, E. (1950) “The partial differential equation $u_t + uu_x = \mu_{xx}$," *Comm. Pure Applied Math.*, **3**: 201–230.
- Itô, K. and H. P. McKean, Jr. (1965) *Diffusion Processes and Their Sample Paths*, Springer-Verlag: Berlin, New York.
- Kac, M. (1947) “Random Walk and the Theory of Brownian Motion,"*The American Mathematical Monthly*, **54(7)**: 369–391.
- Kac, M. (1956) *Some Stochastic Problems in Physics and Mathematics*, Colloquium Lectures in the Pure and Applied Sciences, No. 2, Magnolia Petroleum Co., Hectographed.
- Kac, M. (1980) *Integration in Function Spaces and Some of Its Applications*, Lezioni Fermiane, Accademia Nazionale Dei Lincei Scuola Normale Superiore, Pisa.

Bibliography

- Halton, J. H. (1989) “Pseudo-random trees: multiple independent sequence generators for parallel and branching computations," *J. Comp. Phys.*, **84**: 1–56.
- Hillis, D. (1985) *The Connection Machine*, M.I.T. University Press: Cambridge, MA.
- Hopf, E. (1950) “The partial differential equation $u_t + uu_x = \mu_{xx}$," *Comm. Pure Applied Math.*, **3**: 201–230.
- Itô, K. and H. P. McKean, Jr. (1965) *Diffusion Processes and Their Sample Paths*, Springer-Verlag: Berlin, New York.
- Kac, M. (1947) “Random Walk and the Theory of Brownian Motion,"*The American Mathematical Monthly*, **54(7)**: 369–391.
- Kac, M. (1956) *Some Stochastic Problems in Physics and Mathematics*, Colloquium Lectures in the Pure and Applied Sciences, No. 2, Magnolia Petroleum Co., Hectographed.
- Kac, M. (1980) *Integration in Function Spaces and Some of Its Applications*, Lezioni Fermiane, Accademia Nazionale Dei Lincei Scuola Normale Superiore, Pisa.

Bibliography

Knuth, D. E. (1981) *The Art of Computer Programming, Vol. 2: Seminumerical Algorithms*, Second edition, Addison-Wesley: Reading, MA.

Marsaglia, G. and A. Zaman “A new class of random number generators," submitted to *SIAM J. Sci. Stat. Comput.*

Mascagni, M. (1991) “High dimensional numerical integration and massively parallel computing," *Contemp. Math.*, , **115**: 53–73.

Mascagni, M. (1990) “A tale of two architectures: parallel Wiener integral methods for elliptic boundary value problems," *SIAM News*, **23**: 8,12.

McKean, H. P. (1975)“Application of Brownian Motion to the Equation of Kolmogorov-Petrovskii-Piskunov" *Communications on Pure and Applied Mathematics*, **XXVIII**: 323–331.

Bibliography

- Knuth, D. E. (1981) *The Art of Computer Programming, Vol. 2: Seminumerical Algorithms*, Second edition, Addison-Wesley: Reading, MA.
- Marsaglia, G. and A. Zaman “A new class of random number generators," submitted to *SIAM J. Sci. Stat. Comput.*
- Mascagni, M. (1991) “High dimensional numerical integration and massively parallel computing," *Contemp. Math.*, , **115**: 53–73.
- Mascagni, M. (1990) “A tale of two architectures: parallel Wiener integral methods for elliptic boundary value problems," *SIAM News*, **23**: 8,12.
- McKean, H. P. (1975)“Application of Brownian Motion to the Equation of Kolmogorov-Petrovskii-Piskunov" *Communications on Pure and Applied Mathematics*, **XXVIII**: 323–331.

Bibliography

- Knuth, D. E. (1981) *The Art of Computer Programming, Vol. 2: Seminumerical Algorithms*, Second edition, Addison-Wesley: Reading, MA.
- Marsaglia, G. and A. Zaman “A new class of random number generators," submitted to *SIAM J. Sci. Stat. Comput.*
- Mascagni, M. (1991) “High dimensional numerical integration and massively parallel computing," *Contemp. Math.*, , **115**: 53–73.
- Mascagni, M. (1990) “A tale of two architectures: parallel Wiener integral methods for elliptic boundary value problems," *SIAM News*, **23**: 8,12.
- McKean, H. P. (1975)“Application of Brownian Motion to the Equation of Kolmogorov-Petrovskii-Piskunov" *Communications on Pure and Applied Mathematics*, **XXVIII**: 323–331.

Bibliography

Knuth, D. E. (1981) *The Art of Computer Programming, Vol. 2: Seminumerical Algorithms*, Second edition, Addison-Wesley: Reading, MA.

Marsaglia, G. and A. Zaman “A new class of random number generators," submitted to *SIAM J. Sci. Stat. Comput.*

Mascagni, M. (1991) “High dimensional numerical integration and massively parallel computing," *Contemp. Math.*, , **115**: 53–73.

Mascagni, M. (1990) “A tale of two architectures: parallel Wiener integral methods for elliptic boundary value problems," *SIAM News*, **23**: 8,12.

McKean, H. P. (1975)“Application of Brownian Motion to the Equation of Kolmogorov-Petrovskii-Piskunov" *Communications on Pure and Applied Mathematics*, **XXVIII**: 323–331.

Bibliography

- Knuth, D. E. (1981) *The Art of Computer Programming, Vol. 2: Seminumerical Algorithms*, Second edition, Addison-Wesley: Reading, MA.
- Marsaglia, G. and A. Zaman “A new class of random number generators," submitted to *SIAM J. Sci. Stat. Comput.*
- Mascagni, M. (1991) “High dimensional numerical integration and massively parallel computing," *Contemp. Math.*, , **115**: 53–73.
- Mascagni, M. (1990) “A tale of two architectures: parallel Wiener integral methods for elliptic boundary value problems," *SIAM News*, **23**: 8,12.
- McKean, H. P. (1975)“Application of Brownian Motion to the Equation of Kolmogorov-Petrovskii-Piskunov" *Communications on Pure and Applied Mathematics*, **XXVIII**: 323–331.

Bibliography

McKean, H. P. (1988) Private communication. M. E. Muller, "Some Continuous Monte Carlo Methods for the Dirichlet Problem," The Annals of Mathematical Statistics, 27(3): 569-589, 1956.

Mikhailov, G. A. (1995) *New Monte Carlo Methods With Estimating Derivatives*, V. S. P. Publishers.

Niederreiter, H. (1978) “Quasi-Monte Carlo methods and pseudo-random numbers," *Bull. Amer. Math. Soc.*, **84**: 957–1041.

Rubenstein, M. (1981) *Simulation and the Monte Carlo Method*, Wiley-Interscience: New York.

Sabelfeld, K. K. (1991), *Monte Carlo Methods in Boundary Value Problems*, Springer-Verlag, Berlin, Heidelberg, New York.

Sabelfeld, K. and N. Mozartova (2009) “Sparsified Randomization Algorithms for large systems of linear equations and a new version of the Random Walk on Boundary method," *Monte Carlo Methods and Applications*, **15(3)**: 257–284.

Bibliography

- McKean, H. P. (1988) Private communication. M. E. Muller, "Some Continuous Monte Carlo Methods for the Dirichlet Problem," The Annals of Mathematical Statistics, 27(3): 569-589, 1956.
- Mikhailov, G. A. (1995) *New Monte Carlo Methods With Estimating Derivatives*, V. S. P. Publishers.
- Niederreiter, H. (1978) “Quasi-Monte Carlo methods and pseudo-random numbers," *Bull. Amer. Math. Soc.*, **84**: 957–1041.
- Rubenstein, M. (1981) *Simulation and the Monte Carlo Method*, Wiley-Interscience: New York.
- Sabelfeld, K. K. (1991), *Monte Carlo Methods in Boundary Value Problems*, Springer-Verlag, Berlin, Heidelberg, New York.
- Sabelfeld, K. and N. Mozartova (2009) “Sparsified Randomization Algorithms for large systems of linear equations and a new version of the Random Walk on Boundary method," *Monte Carlo Methods and Applications*, **15(3)**: 257–284.

Bibliography

- McKean, H. P. (1988) Private communication. M. E. Muller, "Some Continuous Monte Carlo Methods for the Dirichlet Problem," The Annals of Mathematical Statistics, 27(3): 569-589, 1956.
- Mikhailov, G. A. (1995) *New Monte Carlo Methods With Estimating Derivatives*, V. S. P. Publishers.
- Niederreiter, H. (1978) “Quasi-Monte Carlo methods and pseudo-random numbers," *Bull. Amer. Math. Soc.*, **84**: 957–1041.
- Rubenstein, M. (1981) *Simulation and the Monte Carlo Method*, Wiley-Interscience: New York.
- Sabelfeld, K. K. (1991), *Monte Carlo Methods in Boundary Value Problems*, Springer-Verlag, Berlin, Heidelberg, New York.
- Sabelfeld, K. and N. Mozartova (2009) “Sparsified Randomization Algorithms for large systems of linear equations and a new version of the Random Walk on Boundary method," *Monte Carlo Methods and Applications*, **15(3)**: 257–284.

Bibliography

- McKean, H. P. (1988) Private communication. M. E. Muller, "Some Continuous Monte Carlo Methods for the Dirichlet Problem," The Annals of Mathematical Statistics, 27(3): 569-589, 1956.
- Mikhailov, G. A. (1995) *New Monte Carlo Methods With Estimating Derivatives*, V. S. P. Publishers.
- Niederreiter, H. (1978) “Quasi-Monte Carlo methods and pseudo-random numbers," *Bull. Amer. Math. Soc.*, **84**: 957–1041.
- Rubenstein, M. (1981) *Simulation and the Monte Carlo Method*, Wiley-Interscience: New York.
- Sabelfeld, K. K. (1991), *Monte Carlo Methods in Boundary Value Problems*, Springer-Verlag, Berlin, Heidelberg, New York.
- Sabelfeld, K. and N. Mozartova (2009) “Sparsified Randomization Algorithms for large systems of linear equations and a new version of the Random Walk on Boundary method," *Monte Carlo Methods and Applications*, **15(3)**: 257–284.

Bibliography

- McKean, H. P. (1988) Private communication. M. E. Muller, "Some Continuous Monte Carlo Methods for the Dirichlet Problem," The Annals of Mathematical Statistics, 27(3): 569-589, 1956.
- Mikhailov, G. A. (1995) *New Monte Carlo Methods With Estimating Derivatives*, V. S. P. Publishers.
- Niederreiter, H. (1978) “Quasi-Monte Carlo methods and pseudo-random numbers," *Bull. Amer. Math. Soc.*, **84**: 957–1041.
- Rubenstein, M. (1981) *Simulation and the Monte Carlo Method*, Wiley-Interscience: New York.
- Sabelfeld, K. K. (1991), *Monte Carlo Methods in Boundary Value Problems*, Springer-Verlag, Berlin, Heidelberg, New York.
- Sabelfeld, K. and N. Mozartova (2009) “Sparsified Randomization Algorithms for large systems of linear equations and a new version of the Random Walk on Boundary method," *Monte Carlo Methods and Applications*, **15(3)**: 257–284.

Bibliography

McKean, H. P. (1988) Private communication. M. E. Muller, "Some Continuous Monte Carlo Methods for the Dirichlet Problem," The Annals of Mathematical Statistics, 27(3): 569-589, 1956.

Mikhailov, G. A. (1995) *New Monte Carlo Methods With Estimating Derivatives*, V. S. P. Publishers.

Niederreiter, H. (1978) "Quasi-Monte Carlo methods and pseudo-random numbers," *Bull. Amer. Math. Soc.*, **84**: 957–1041.

Rubenstein, M. (1981) *Simulation and the Monte Carlo Method*, Wiley-Interscience: New York.

Sabelfeld, K. K. (1991), *Monte Carlo Methods in Boundary Value Problems*, Springer-Verlag, Berlin, Heidelberg, New York.

Sabelfeld, K. and N. Mozartova (2009) "Sparsified Randomization Algorithms for large systems of linear equations and a new version of the Random Walk on Boundary method," *Monte Carlo Methods and Applications*, **15(3)**: 257–284.

Bibliography

Sherman, A. S., and C. S. Peskin (1986) “A Monte Carlo method for scalar reaction diffusion equations”, *SIAM J. Sci. Stat. Comput.*, **7**: 1360–1372.

Sherman, A. S., and C. S. Peskin (1988) “Solving the Hodgkin-Huxley equations by a random walk method”, *SIAM J. Sci. Stat. Comput.*, **9**: 170–190.

Shreider, Y. A. (1966) *The Monte Carlo Method. The Method of Statistical Trial*, Pergamon Press: New York.

Spanier, J. and E. M. Gelbard (1969) *Monte Carlo Principles and Neutron Transport Problems*, Addison-Wesley: Reading, MA.

Wasow, W. R. (1952), “A Note on the Inversion of Matrices by Random Walks," *Mathematical Tables and Other Aids to Computation*, **6(38)**: 78–81.

Bibliography

Sherman, A. S., and C. S. Peskin (1986) “A Monte Carlo method for scalar reaction diffusion equations”, *SIAM J. Sci. Stat. Comput.*, **7**: 1360–1372.

Sherman, A. S., and C. S. Peskin (1988) “Solving the Hodgkin-Huxley equations by a random walk method”, *SIAM J. Sci. Stat. Comput.*, **9**: 170–190.

Shreider, Y. A. (1966) *The Monte Carlo Method. The Method of Statistical Trial*, Pergamon Press: New York.

Spanier, J. and E. M. Gelbard (1969) *Monte Carlo Principles and Neutron Transport Problems*, Addison-Wesley: Reading, MA.

Wasow, W. R. (1952), “A Note on the Inversion of Matrices by Random Walks," *Mathematical Tables and Other Aids to Computation*, **6(38)**: 78–81.

Bibliography

- Sherman, A. S., and C. S. Peskin (1986) “A Monte Carlo method for scalar reaction diffusion equations”, *SIAM J. Sci. Stat. Comput.*, **7**: 1360–1372.
- Sherman, A. S., and C. S. Peskin (1988) “Solving the Hodgkin-Huxley equations by a random walk method”, *SIAM J. Sci. Stat. Comput.*, **9**: 170–190.
- Shreider, Y. A. (1966) *The Monte Carlo Method. The Method of Statistical Trial*, Pergamon Press: New York.
- Spanier, J. and E. M. Gelbard (1969) *Monte Carlo Principles and Neutron Transport Problems*, Addison-Wesley: Reading, MA.
- Wasow, W. R. (1952), “A Note on the Inversion of Matrices by Random Walks," *Mathematical Tables and Other Aids to Computation*, **6(38)**: 78–81.

Bibliography

- Sherman, A. S., and C. S. Peskin (1986) “A Monte Carlo method for scalar reaction diffusion equations”, *SIAM J. Sci. Stat. Comput.*, **7**: 1360–1372.
- Sherman, A. S., and C. S. Peskin (1988) “Solving the Hodgkin-Huxley equations by a random walk method”, *SIAM J. Sci. Stat. Comput.*, **9**: 170–190.
- Shreider, Y. A. (1966) *The Monte Carlo Method. The Method of Statistical Trial*, Pergamon Press: New York.
- Spanier, J. and E. M. Gelbard (1969) *Monte Carlo Principles and Neutron Transport Problems*, Addison-Wesley: Reading, MA.
- Wasow, W. R. (1952), “A Note on the Inversion of Matrices by Random Walks," *Mathematical Tables and Other Aids to Computation*, **6(38)**: 78–81.

Bibliography

Sherman, A. S., and C. S. Peskin (1986) “A Monte Carlo method for scalar reaction diffusion equations”, *SIAM J. Sci. Stat. Comput.*, **7**: 1360–1372.

Sherman, A. S., and C. S. Peskin (1988) “Solving the Hodgkin-Huxley equations by a random walk method”, *SIAM J. Sci. Stat. Comput.*, **9**: 170–190.

Shreider, Y. A. (1966) *The Monte Carlo Method. The Method of Statistical Trial*, Pergamon Press: New York.

Spanier, J. and E. M. Gelbard (1969) *Monte Carlo Principles and Neutron Transport Problems*, Addison-Wesley: Reading, MA.

Wasow, W. R. (1952), “A Note on the Inversion of Matrices by Random Walks," *Mathematical Tables and Other Aids to Computation*, **6(38)**: 78–81.