Grid-Free Monte Carlo for PDEs with Spatially Varying Coefficients

ROHAN SAWHNEY[*], Carnegie Mellon University, USA
DARIO SEYB[*], Dartmouth College, USA
WOJCIECH JAROSZ[†], Dartmouth College, USA
KEENAN CRANE[†], Carnegie Mellon University, USA

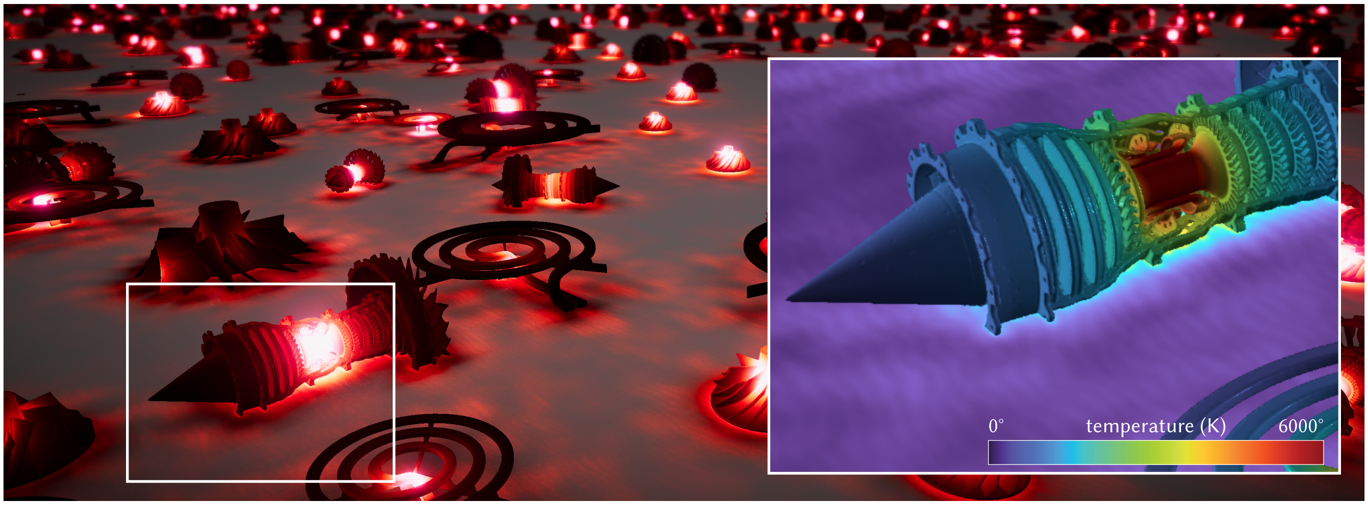


Fig. 1. Distribution of heat (inset) radiating from infinitely many blackbodies—about 600M effective boundary vertices are visible from this viewpoint alone. (Here we visualize a 2D slice of the full 3D solution.) Our Monte Carlo PDE solver directly captures fine geometric detail and intricate spatially varying coefficients without meshing, sampling, or homogenizing the 3D domain, by building on techniques from volumetric rendering.

Partial differential equations (PDEs) with spatially varying coefficients arise throughout science and engineering, modeling rich heterogeneous material behavior. Yet conventional PDE solvers struggle with the immense complexity found in nature, since they must first discretize the problem—leading to spatial aliasing, and global meshing/sampling that is costly and error-prone. We describe a method that approximates neither the domain geometry, the problem data, nor the solution space, providing the exact solution (in expectation) even for problems with extremely detailed geometry and intricate coefficients. Our main contribution is to extend the *walk on spheres (WoS)* algorithm from constant- to variable-coefficient problems, by drawing on techniques from volumetric rendering. In particular, an approach inspired by *null-scattering* yields unbiased Monte Carlo estimators for a large class of 2nd order elliptic PDEs, which share many attractive features with Monte Carlo rendering: no meshing, trivial parallelism, and the ability to evaluate the solution at any point without solving a global system of equations.

CCS Concepts: • **Mathematics of computing** → **Partial differential equations**.

Additional Key Words and Phrases: integral equations, Monte Carlo methods

[*]and [†] indicate equal contribution.

Authors' addresses: Rohan Sawhney, rohansawhney@cs.cmu.edu, Carnegie Mellon University, USA; Dario Seyb, dario.r.seyb.gr@dartmouth.edu, Dartmouth College, USA; Wojciech Jarosz, wojciech.k.jarosz@dartmouth.edu, Dartmouth College, USA; Keenan Crane, kmcrane@cs.cmu.edu, Carnegie Mellon University, USA.

ACM Reference Format:
Rohan Sawhney, Dario Seyb, Wojciech Jarosz, and Keenan Crane. 2022. Grid-Free Monte Carlo for PDEs with Spatially Varying Coefficients. *ACM Trans. Graph.* 41, 4, Article 53 (July 2022), 17 pages. https://doi.org/10.1145/3528223.3530134

1 INTRODUCTION

PDEs with spatially varying coefficients describe a rich variety of phenomena. In thermodynamics, for example, variable coefficients model how heterogeneous composite materials conduct or insulate heat. Much as early algorithms for photorealistic rendering were motivated by predictive lighting design [Ward and Shakespeare 1998], such models can be used to predict and improve thermal efficiency in building design [Zalewski et al. 2010]. Likewise, variable permittivity in electrostatics impacts the design of antennas [Ozdemir 2005] and the simulation of biomolecules [Fahrenberger et al. 2014]; in hydrology, variable transmissivity of water through soil impacts remediation strategies for groundwater pollution [Willmann et al. 2010]. More directly connected to our work, variable coefficients in the light transport equation are used to model heterogeneity in participating media [Novák et al. 2018]. Beyond spatially varying materials, variable coefficients can also be used to model curved geometry by using PDE coefficients on a flat domain to encode an alternative *Riemannian metric* (see Sec. 6.5).

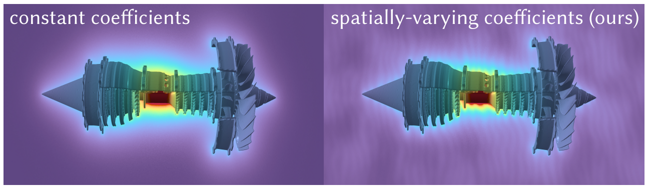


Fig. 2. We directly resolve the detailed effects of, e.g., spatially varying material density—without resorting to homogenization of PDE coefficients.

Our method computes the exact solution (in expectation) to 2nd order linear elliptic equations of the form

$$\boxed{\begin{array}{rcll} \nabla\cdot(\alpha\nabla u)+\vec{\omega}\cdot\nabla u-\sigma u &=& -f & \text{on } \Omega,\\ u &=& g & \text{on } \partial\Omega.\end{array}} \tag{1}$$

Here Ω is a region in $\mathbb{R}^n$, $\alpha : \Omega \rightarrow \mathbb{R}_{>0}$ and $\vec{\omega} : \Omega \rightarrow \mathbb{R}^n$ are a twice-differentiable function and a vector field expressible as the gradient of a scalar field (resp.), and $\sigma : \Omega \rightarrow \mathbb{R}_{\geq 0}$ is continuous (C^0). These conditions are sufficient to ensure ellipticity. The *source term* $f : \Omega \rightarrow \mathbb{R}$ and *boundary values* $g : \partial\Omega \rightarrow \mathbb{R}$ need not be continuous. Figure 5 illustrates the effect of each term on the solution u; see Sec. 2.2 for further background on PDEs. Note that unlike methods for *numerical homogenization* [Desbrun et al. 2013], we aim to directly resolve the original, detailed solution (Fig. 2).

Though many methods consider such PDEs, they all suffer from a common problem: the need to spatially discretize (e.g., mesh or point-sample) the domain interior. Even so-called *meshless methods (MFEM)* must carefully distribute interior nodes (Fig. 4); *boundary element methods (BEM)* must be integrated with volumetric methods to handle interior terms (see Sec. 7). For problems with intricate geometry, discretization is hence a major burden on designers, scientists, and engineers: even state-of-the-art methods are error-prone, can take hours of preprocessing, and can destroy application-critical features due to spatial aliasing (Fig. 3). Such problems are further compounded by variable coefficients, since now the discretization must also be carefully adapted to regions where coefficients exhibit fine detail (Fig. 23). Moreover, meshing algorithms (and code) for spatially-varying PDEs lag far behind technology for constant-coefficient problems [Alauzet and Loseille 2016].

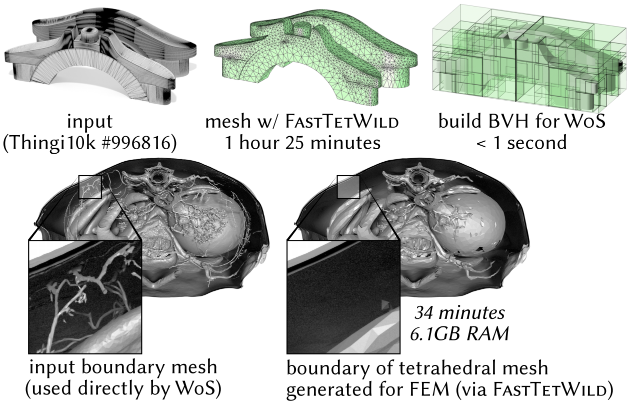


Fig. 3. *Top:* The bottleneck in conventional methods is often not the solve itself, but rather the cost of meshing (here, via Hu et al. [2020]). As in rendering, WoS need only build a simple bounding volume hierarchy (BVH). *Bottom:* Conventional methods also sacrifice spatial detail—here destroying key features like blood vessels. *Figures from Sawhney and Crane [2020].*

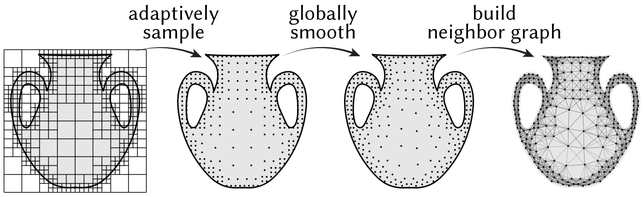


Fig. 4. So-called "meshless" methods still perform a process akin to global meshing, which can result in spatial aliasing of fine features. One ends up with a mesh-like structure which must satisfy stringent sampling criteria to avoid numerical blowup, and must still solve a large globally coupled linear system. (Figure adapted from [Pauly et al. 2005, Figure 6].)

Overall, these challenges make it difficult (if not impossible) to analyze large, heterogeneous systems (as in Fig. 1) which commonly arise in real applications—say, directly analyzing a *building information model (BIM)* that includes heating ducts, plumbing, insulation, etc., rather than a simplified geometric proxy. Yet scenes of this size and complexity are commonplace in rendering. Why such a big gap between our ability to *visualize* and *analyze* complex scenes? A major reason is rendering has moved away from methods like *finite element radiosity* [Cohen and Wallace 1993] and toward Monte Carlo methods—both to handle more intricate light transport phenomena and to avoid difficult meshing problems [Christensen and Jarosz 2016; Jensen 2001, Chapter 1]. For PDEs however, Monte Carlo techniques have received comparatively little attention.

Grid-free Monte Carlo methods solve PDEs without discretizing the problem domain, nor the space of functions used to represent the solution. The starting point is the *walk on spheres (WoS)* method of Muller [1956], which uses a recursive integral formulation akin to the classic *rendering equation* [Kajiya 1986]. This approach sidesteps many challenges faced by conventional solvers: it can evaluate the solution at any point without solving a global system, is trivial to parallelize, and works directly with any boundary representation (implicit surfaces, spline patches, etc.), including coarse or low-quality meshes unsuitable for finite-element analysis. Following Sawhney and Crane [2020], recent work in computer graphics explores how to generalize and accelerate WoS, by drawing inspiration from geometry processing and Monte Carlo rendering [Krayer and Müller 2021; Marschner et al. 2021; Mossberg 2021; Nabizadeh et al. 2021]. To date, however, WoS still handles only a small class of constant-coefficient PDEs, limiting its use in applications.

1.1 Contributions

We generalize WoS to a large set of variable-coefficient PDEs, by establishing a link with recent *null-scattering* techniques for rendering heterogeneous participating media [Novák et al. 2018]. To our knowledge, there is no other way to solve such PDEs without discretizing space—whether by previous WoS methods, or any other means. Our method is appropriate for finding the steady-state solution for a diffusive process (in contrast to, say, the dynamics of large scale deformations). Specifically, we provide:

- A novel integral formulation of variable-coefficient 2nd order linear elliptic PDEs that makes them amenable to grid-free Monte Carlo methods.
- Efficient WoS algorithms inspired by the *delta tracking* [Woodcock et al. 1965] and *next-flight* [Cramer 1978] methods from volume rendering.
- A variance reduction strategy that significantly reduces noise in problems with high-frequency coefficients, based on *weight windows* from neutron transport.

In the process, we obtain a precise mathematical picture of the relationship between diffusive variable-coefficient PDEs and heterogeneous participating media. Since we transform the original problem into a constant-coefficient PDE with a variable source term, we get the usual convergence guarantee for WoS algorithms: the variance of an N-sample estimate decreases at a rate $1/N$, with a negligible amount of bias due to the ε-shell (see [Sawhney and Crane 2020, Section 6.1]). Numerical experiments on several thousand models empirically verify convergence behavior (see Figs. 16, 17, 24 and 28). A implementation of our method in both C++ and detailed pseudocode can be found in the supplemental material.

Limitations. WoS is an emerging class of methods that does not yet support all the features of more mature methods such as FEM—for example, it is still not known how to handle general Neumann or Robin boundary conditions (as needed in, e.g., linear elasticity). These questions are largely orthogonal to the issues addressed here, and are left as future work (see Sec. 8 for further discussion).

2 BACKGROUND

The derivation of our method depends on concepts from PDE theory, the theory of integral equations, stochastic calculus, and volumetric rendering. Since we expect few readers will be familiar with all these topics, we provide essential background here. Sec. 3 then describes the basic WoS algorithm, which is the starting point for our variable-coefficient algorithm in Sec. 4. For a gentler introduction to WoS, see Sawhney and Crane [2020, Section 2].

2.1 Notation

For any region $A \subset \mathbb{R}^n$, we use $|A|$ to denote its volume and ∂A for its boundary. For a point $x \in A$, $\overline{x}$ denotes the point closest to x on the boundary ∂A. Throughout, $\Omega \subset \mathbb{R}^n$ is the domain of interest, $B(x) \subset \Omega$ is a ball centered on x, and $\partial\Omega_\varepsilon := \{x \in \Omega : |x - \overline{x}| < \varepsilon\}$ is an *epsilon shell* around $\partial\Omega$. We use $\vec{u}$ to denote a vector field on $\mathbb{R}^n$, and ∇ and $\nabla\cdot$ for the gradient and divergence operators (resp.), so that $\Delta := \nabla \cdot \nabla$ is the negative-semidefinite Laplacian. We use $\mathcal{U}$ to denote the uniform distribution on $[0, 1] \subset \mathbb{R}$, $\mathcal{N}(x, v)$ for the n-dimensional normal distribution with mean x and variance v, and $\mathbb{E}[X]$ for the expected value of a random variable X.

2.2 Differential Equations

A *partial differential equation (PDE)* describes a function u implicitly, via relationships between partial derivatives in space. One must ultimately solve for an explicit function satisfying this equation—which is the *raison d'être* for numerical PDE solvers. Here we describe the terms in our main PDE (Eq. 1), which are also visualized in Fig. 5.

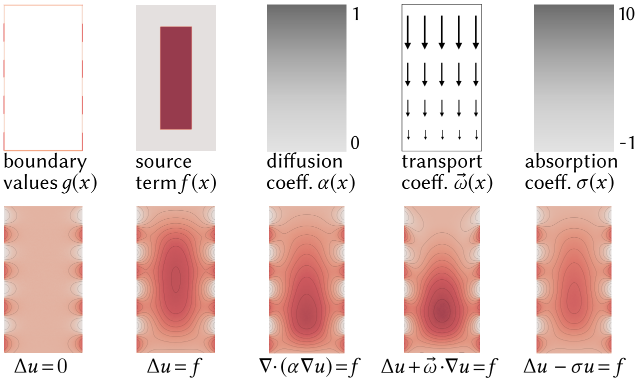


Fig. 5. Effect of each term of our main PDE (Eq. 1) on the solution.

A standard PDE is the *Laplace equation* $\Delta u = 0$, which describes the steady-state of a diffusion process, i.e., the way heat diffuses smoothly from the domain boundary into the interior. Like Eq. 1, the Laplace equation is *2nd order* in space since it involves spatial derivatives no higher than degree two; it is *linear* since it is a linear polynomial equation with respect to the function and its derivatives.

Boundary conditions. We primarily consider *Dirichlet conditions*, which fix the solution at points where the value of u is known (e.g., the temperature on the boundary). For instance, a Laplace equation with Dirichlet boundary conditions has the form

$$\begin{array}{rcll} \Delta u(x) & = & 0 & \text{on } \Omega, \\ u(x) & = & g(x) & \text{on } \partial\Omega, \end{array} \tag{2}$$

where $g : \partial\Omega \to \mathbb{R}$ is a given function. *Neumann conditions* instead specify derivatives (e.g., heat flow through the boundary). Though our final algorithms handle only Dirichlet conditions, the transformations in Sec. 4 make no assumptions about the type of boundary conditions—and could in principle be applied to Neumann problems.

Source term. Continuing with the heat analogy, a *source term* $f : \Omega \to \mathbb{R}$ adds additional "background temperature" to a PDE. For instance, a *Poisson equation* has the form

$$\Delta u(x) \quad = \quad -f(x) \quad \text{on } \Omega \tag{3}$$

subject to, e.g., Dirichlet or Neumann boundary conditions.

Diffusion coefficient. The rate of diffusion in a spatially-varying medium is modeled by replacing Δ with the operator $\nabla \cdot (\alpha(x)\nabla)$, where $\alpha : \Omega \to \mathbb{R}_{>0}$ is the *diffusion coefficient* (Fig. 5, *center*).

Absorption. An *absorption (or screening) coefficient* $\sigma : \Omega \to \mathbb{R}$ models "cooling" of the solution due to the background medium; positive/negative coefficients dampen/magnify the solution, resp. E.g., a *screened Poisson equation*, seen in Fig. 5, *far right*, is given by

$$\Delta u(x) - \sigma(x)u(x) \quad = \quad -f(x) \quad \text{on } \Omega, \tag{4}$$

again subject to boundary conditions.

Drift. Finally, a *drift coefficient*, given by a vector field $\vec{\omega} : \Omega \to \mathbb{R}^n$, models the motion of a material in a particular direction. For instance, the *steady-state advection equation* $\vec{\omega}(x) \cdot \nabla u(x) = 0$ describes a quantity u that is unchanged as it flows along $\vec{\omega}$; adding this term to a Poisson equation causes heat to drift as it diffuses.

2.3 Integral Equations

One can often reformulate elliptic PDEs, such as the equations from the previous section, as *recursive integral equations*—akin to the classic *rendering equation* [Kajiya 1986]. As in rendering, such equations can be solved without discretizing space, by recursively applying Monte Carlo integration (as will be discussed in Sec. 3).

Boundary conditions. Our starting point is the *mean value principle*, which says that at each point $x \in \Omega$, the solution u to a Laplace equation (Eq. 2) equals its mean value over any sphere around x:

$$u(x) = \frac{1}{|\partial B(x)|} \int_{\partial B(x)} u(z) \; \mathrm{d}z. \tag{5}$$

Notice that Eq. 5 is recursive: the value of u at x depends on (unknown) values at other points z. The "base case" is effectively provided by the (known) boundary values $g(x)$—see especially Eq. 19.

Source term. For a PDE with a source term f, such as the Poisson equation (Eq. 3), the integral representation of u gains a term

$$\int_{B(x)} f(y)\; G(x,y) \; \mathrm{d}y, \tag{6}$$

where $G(x,y)$ is the *Green's function* for the PDE. In general, for a linear PDE $Lu = f$, the Green's function G satisfies $LG = \delta$, where δ is a Dirac delta. When L is the Laplacian Δ, G is called the *harmonic Green's function*, and describes a source f with a single "spike" of heat. Importantly, the Green's function depends on the shape of the domain Ω: it will be different for, e.g., a ball versus all of $\mathbb{R}^n$. This representation is attractive because many common Green's functions are known in closed form (see Sec. 1 of the supplement).

Constant diffusion, absorption, and drift. For a constant diffusion coefficient $\alpha \in \mathbb{R}$, the Green's function G simply scales by a factor α. Adding an absorption term $\sigma u(x)$ with constant coefficient $\sigma \in \mathbb{R}$ replaces the harmonic Green's function with the *Yukawa potential* G^σ (Fig. 6), given explicitly in the supplement. Finally, drift along a constant direction $\vec{\omega} \in \mathbb{R}^n$ can be captured via the *von Mises–Fisher distribution* [Gatto 2013; Sabelfeld 2018], though we will not need this distribution in our formulation.

Variable coefficients. The constant-coefficient representations described in this section cannot immediately be applied to PDEs with variable coefficients. In Sec. 4 we therefore transform our main PDE (Eq. 1) into a *constant*-coefficient screened Poisson equation (Eq. 4), but with a recursive right-hand side. Combining several of the terms above, we can then write the solution to Eq. 4 in integral form:

$$u(x) = \int_{B(c)} f(y)\; G^\sigma(x,y)\, \mathrm{d}y \; + \int_{\partial B(c)} u(z)\; P^\sigma(x,z)\, \mathrm{d}z. \tag{7}$$

Here we also use a more general *off-centered* formulation, where the point of evaluation x need not coincide with the center $c \in \Omega$ of the ball $B(c)$ [Duffy 2015; Hwang et al. 2015]. In particular, the boundary term now incorporates the *Poisson kernel* $P^\sigma(x,z)$. Like the Green's function, the Poisson kernel describes how a "spike" on the boundary affects the solution, and in general is given by the normal derivative of the Green's function at the boundary (see supplemental material for explicit expressions). For $x = c$ and $\sigma = 0$, $P^\sigma(x,z)$ reduces to $1/|\partial B(x)|$, recovering the usual mean value property.

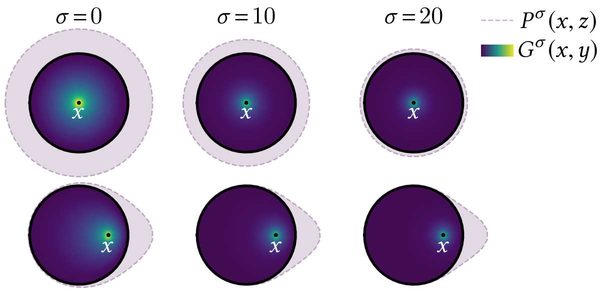


Fig. 6. *Top*: As σ increases, the Green's function $G^\sigma(x,y)$ for a screened Poisson equation becomes more localized around the point x; the magnitude of the Poisson kernel $P^\sigma(x,z)$ shrinks. *Bottom*: When x is not at the center of the ball, G^σ and P^σ are no longer rotationally symmetric; as $\sigma \to 0$, they approach the harmonic Green's function and Poisson kernel (resp.).

2.4 Stochastic Equations

The solution to an elliptic PDE can also be described in terms of random walks, via the *Feynman-Kac formula* from stochastic calculus [Øksendal 2003, Ch. 8]. This formula will provide a critical starting point for our method in Sec. 4, since it is more general than known deterministic integral formulas. Moreover, Feynman-Kac has close parallels with volume rendering (Sec. 2.5), providing us with key techniques for numerical integration in Sec. 5.

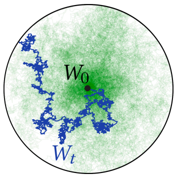


Fig. 7. *Brownian motion* is a random walk (blue) which on average models diffusion (green).

2.4.1 Stochastic Processes. A *continuous stochastic process* describes the trajectory of a particle taking a continuous "random walk." A central example is a *Brownian motion* W_t, characterized by the property that increments $W_{t+s} - W_t$ follow a normal distribution $\mathcal{N}(0,s)$, and are independent of past values of W_t (Fig. 7). By symmetry, a Brownian walk starting at the center x of a ball $B(x)$ exits on all points on its boundary $\partial B(x)$ with equal probability.

More generally, a *diffusion process* (Fig. 8) describes a particle with velocity $\vec{\omega}(x)$ subject to random displacements of strength $\alpha(x)$, and obeys a *stochastic differential equation (SDE)*

$$\mathrm{d}X_t = \vec{\omega}(X_t)\, \mathrm{d}t \; + \; \sqrt{\alpha(X_t)} \;\, \mathrm{d}W_t. \tag{8}$$

As in volume rendering (Sec. 2.5), an *extinction coefficient* $\sigma(x)$ can be used to model absorption into a background medium.

2.4.2 Feynman-Kac Formula. Notice that the parameters $\alpha, \vec{\omega}, \sigma$ of a diffusion process resemble the coefficients of our main PDE (Eq. 1). Feynman-Kac makes this relationship explicit by expressing the solution to Eq. 1 as an expectation over random trajectories of X_t.

Boundary term. We can build up the relationship between PDEs and stochastic processes by first considering a basic Laplace equation $\Delta u = 0$ with Dirichlet boundary values g (Eq. 2). In this case, *Kakutani's principle* [Kakutani 1944] states that

$$u(x) = \mathbb{E}[g(W_\tau)], \tag{9}$$

where τ is the (random) time when W_t first hits the domain boundary $\partial\Omega$. In other words, the solution to a Laplace equation is just the average boundary value "seen" by random walkers starting at x.

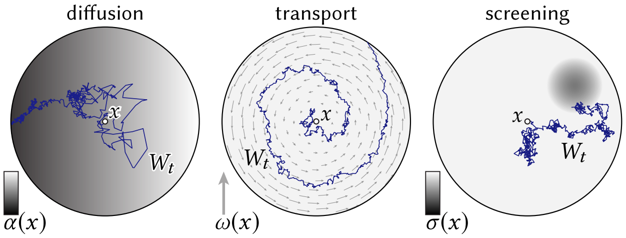


Fig. 8. Components of a diffusion process. Note that unlike Brownian motion, a diffusion process will not in general have a uniform exit distribution over $\partial B(x)$. *Left*: the diffusion coefficient $\alpha(x)$ modulates the size of random increments. *Center*: the drift coefficient $\vec{\omega}(x)$ adds deterministic offsets to the trajectory. *Right*: the screening coefficient $\sigma(x) > 0$ describes the probability of W_t being absorbed; $\sigma(x) < 0$ essentially describes emission of new random walkers.

Source term. For PDEs with a source term f, such as the Poisson equation $\Delta u = -f$, the solution u also picks up a term capturing the average heat "felt" by a random walker along its path (see Øksendal [2003, Ch. 9]):

$$\mathbb{E}\left[\int_0^\tau f(W_t)\ \mathrm{d}t\right]. \tag{10}$$

Absorption. To model the effect of absorption, as in a screened Poisson equation $\Delta u - \sigma u = -f$, we can incorporate the absorption coefficient $\sigma \in \mathbb{R}$ into the boundary and source terms to get

$$\mathbb{E}\left[\mathrm{e}^{-\sigma\tau} g(W_\tau)\right] \qquad \text{and} \qquad \mathbb{E}\left[\int_0^\tau \mathrm{e}^{-\sigma t} f(W_t)\,\mathrm{d}t\right], \tag{11}$$

respectively [Øksendal 2003, Ch. 8]. Notice that *larger* values of σ yield *smaller* contributions. For a spatially varying coefficient $\sigma(x)$, we can simply replace $-\sigma\tau$ with $-\int_0^\tau \sigma(W_t)\,\mathrm{d}t$ in both terms.

Feynman-Kac. Finally, to account for spatially varying diffusion $\alpha(x)$ and drift $\vec{\omega}(x)$, we can replace the Brownian motion W_t with a general diffusion process X_t, *à la* Eq. 8. Combining the expressions for boundary, source, and absorption terms from Eqs. 9–11, we arrive at the Feynman-Kac formula

$$u(x) = \mathbb{E}\left[\int_0^\tau \mathrm{e}^{-\int_0^t \sigma(X_s)\,\mathrm{d}s} f(X_t)\,\mathrm{d}t \ + \ \mathrm{e}^{-\int_0^\tau \sigma(X_t)\,\mathrm{d}t} g(X_\tau)\right]. \tag{12}$$

Though this formula can be approximated via direct simulation of small, explicit time steps (Eq. 31), it is not amenable to efficient, bias-free walk on spheres methods—see Sec. 7.1.3 for more discussion.

2.4.3 Relationship Between Stochastic and Integral Equations. The stochastic viewpoint also provides a useful interpretation of the Green's function G^σ and Poisson kernel P^σ. On a ball $B(x)$, $G^\sigma(x, y)$ can be viewed as an (unnormalized) probability density, describing how likely it is that a random walker starting at x passes through any point $y \in B(x)$; $P^\sigma(x, z)$ likewise gives the probability that a walker exits through any point $z \in \partial B(x)$ (assuming it is not first absorbed). For instance, when $\sigma = 0$, a Brownian random walk W_t starting at the ball center $x = c$ will exit through all points on the boundary sphere $\partial B(x)$ with equal probability. In this case, the Poisson kernel also reduces to a constant function, in accordance with the mean value property (Eq. 5).

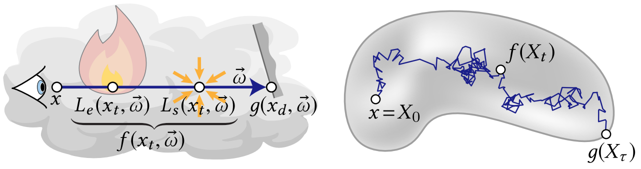


Fig. 9. *Left*: The VRE describes the radiance $L(x, \vec{\omega})$ along a ray as a function of scattering and emission f, as well as the radiance g leaving the boundary. *Right*: Feynman-Kac likewise describes how a source term f and boundary data g contribute to the solution of a diffusive PDE with variable coefficients, but along the trajectory of a random process X_t instead of a light path.

2.5 Volume Rendering

In graphics, the radiative transport equation (RTE) [Chandrasekhar 1960] is used to describe the behavior of light in heterogeneous media that absorb, scatter and emit radiation (Fig. 9, *left*). Unlike Eq. 1, the RTE is only 1st order in space. It states that the *radiance* $L(x, \vec{\omega})$ at each point x and in each direction $\vec{\omega} \in \mathbb{R}^n$ satisfies

$$\begin{array}{rcll} \vec{\omega} \cdot \nabla L(x, \vec{\omega}) - \sigma(x) L(x, \vec{\omega}) & = & -f(x, \vec{\omega}, L) & \text{on } \Omega, \\ L(x, \vec{\omega}) & = & g(x, \vec{\omega}, L) & \text{on } \partial\Omega. \end{array} \tag{13}$$

This equation is recursive, since the source term $f(x, \vec{\omega}, L)$ depends on the radiance $L_s(x, \vec{\omega})$ *in-scattered* at x (as well as any *emission* $L_e(x, \vec{\omega})$); likewise, the function $g(x, \vec{\omega}, L)$ describes radiance leaving the boundary. The spatially varying *extinction coefficient* $\sigma(x)$ specifies the density of scattering or absorbing particles at x.

The integral representation of the RTE is called the *volume rendering equation (VRE)* [Pharr et al. 2016, Ch. 15.1], which gives the radiance $L(x, \vec{\omega})$ as an integral along a ray $x_t := x - \vec{\omega}t$ of length d:

$$L(x, \vec{\omega}) = \int_0^d \mathrm{e}^{-\int_0^t \sigma(x_s)\,\mathrm{d}s} f(x_t, \vec{\omega}, L)\,\mathrm{d}t \ + \\ \mathrm{e}^{-\int_0^d \sigma(x_t)\,\mathrm{d}t} g(x_d, \vec{\omega}, L). \tag{14}$$

Delta tracking. The VRE is typically solved using *volumetric path tracing (VPT)* [Lafortune and Willems 1996], but a spatially varying $\sigma(x)$ presents challenges akin to those for Feynman-Kac: approximating the *transmittance function* $\exp(-\int_0^d \sigma(x_t)\,\mathrm{d}t)$ via explicit steps along x_t can yield significant error. *Delta tracking* [Raab et al. 2008; Woodcock et al. 1965] instead rewrites Eq. 13 so that all spatial variation in the extinction coefficient $\sigma(x)$ is captured by a source term on the right-hand side—leaving only a constant absorption coefficient $\bar{\sigma} := \max(\sigma(x))$ [Galtier et al. 2013; Kutz et al. 2017]:

$$\vec{\omega} \cdot \nabla L(x, \vec{\omega}) - \bar{\sigma} L(x, \vec{\omega}) = -\underbrace{\left(f(x, \vec{\omega}, L) + (\bar{\sigma} - \sigma(x)) L(x, \vec{\omega})\right)}_{=:\, f'(x, \vec{\omega}, L)}. \tag{15}$$

Conceptually, fictitious *null matter* is added to the initially heterogeneous medium so that it now has a constant density (Fig. 10, *left*). Eq. 15 then has an integral representation

$$L(x, \vec{\omega}) = \int_0^d \mathrm{e}^{-\bar{\sigma} t} f'(x_t, \vec{\omega}, L)\,\mathrm{d}t \ + \ \mathrm{e}^{-\bar{\sigma} d} g(x_d, \vec{\omega}, L). \tag{16}$$

This representation is more amenable to Monte Carlo integration, since the transmittance function $\mathrm{e}^{-\bar{\sigma} t}$ can be evaluated in closed form. Spatial variations in $\sigma(x)$ are now accounted for by weighting the radiance inside the source term f' in Eq. 16 by $\bar{\sigma} - \sigma(x_t)$.

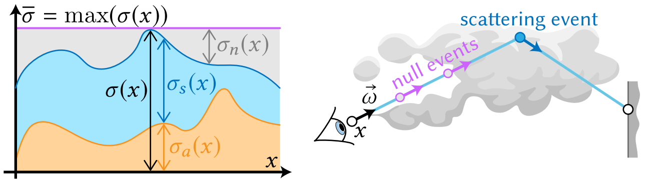


Fig. 10. *Left*: The delta tracking method in volume rendering artificially fills a heterogeneous medium with fictitious null matter (indicated by $\sigma_n(x)$) so that the combined density $\sigma(x)$ is constant everywhere. *Right*: *Null-events* probabilistically sampled inside the medium by VPT [Fong et al. 2017, Algorithm 2] then re-weight the radiance by $\bar{\sigma} - \sigma(x)$ to account for the original heterogeneity in $\sigma(x)$. Likewise, the WoS algorithm in Fig. 13 solves variable-coefficient PDEs by introducing null-events into the random walk.

In Sec. 4, we derive a generalized mean value expression for the PDE in Eq. 1 by applying the delta tracking transformation to the Feynman-Kac formula, resulting in an integral representation amenable to walk on spheres.

3 WALK ON SPHERES

We next describe the basic *walk on spheres (WoS)* algorithm. WoS was developed by Muller [1956] to solve the Laplace equation (Eq. 2), but has since been extended to a broader class of PDEs. Section 3.2 reviews WoS estimators for constant-coefficient PDEs, which serve as building blocks for our variable-coefficient extension in Sec. 4.

3.1 Monte Carlo Integration

WoS is a Monte Carlo estimator for the solution to a PDE. In general, a *Monte Carlo estimator* approximates an integral using random samples of the integrand. In particular, for any (L^1) integrable function $\phi : \Omega \rightarrow \mathbb{R}$, the quantity

$$I := \int_{\Omega} \phi(x)\,\mathrm{d}x \tag{17}$$

can be approximated by the sum

$$\widehat{I}_N := \frac{1}{N}\sum_{i=1}^{N} \frac{\phi(X_i)}{p(X_i)}, \quad X_i \sim p, \tag{18}$$

where the X_i are independent random samples drawn from any probability density p that is nonzero on the support of ϕ. In this paper we will express all our estimators as *single-sample estimators* $\widehat{I}$ (dropping the subscript $N = 1$ for brevity), with the expectation that their values will be averaged over many trials to improve accuracy.

Importantly, although $\widehat{I}_N$ is called an “estimator”, it does not provide merely an estimate—rather, a well-designed estimator will give the *exact* value of the integral, in expectation. More precisely, $\widehat{I}_N$ is *unbiased* if $\mathbb{E}[\widehat{I}_N] = I$ for *any* number of samples N, and *consistent* if the error $\widehat{I}_N - I$ goes to zero as $N \longrightarrow \infty$ with probability one [Veach 1997, Section 1.4.4]. Error is more often quantified by the *variance* $\mathrm{Var}[\widehat{I}_N] := \mathbb{E}[(\widehat{I}_N - \mathbb{E}[\widehat{I}_N])^2]$, i.e., the average squared deviation from the expected value. As long as ϕ has finite variance, an unbiased estimator is automatically consistent (by the central limit theorem), with variance going to zero at a rate $O(1/N)$.

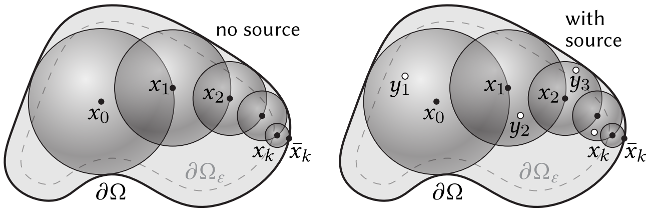


Fig. 11. *Left*: The walk on spheres algorithm repeatedly jumps to a random point on the largest sphere centered at the current point x_k, until it gets within an ε distance to the boundary. *Right*: An additional random sample y_{k+1} inside each ball $B(x_k)$ is used to evaluate the source term f.

3.2 The Walk on Spheres Algorithm

Suppose we want to evaluate the solution to a basic Laplace equation $\Delta u = 0$ with Dirichlet boundary conditions g (Eq. 2) at some point $x_0 \in \Omega$. The mean value formula (Eq. 5) says that $u(x_0)$ is equal to the average of u over any ball $B(x_0) \subset \Omega$; alternatively, Kakutani's principle (Eq. 9) says that $u(x_0)$ equals the expected value of u where trajectories of random walkers first hit the ball boundary:

$$u(x_0) = \frac{1}{|\partial B(x_0)|}\int_{\partial B(x_0)} u(z)\ \mathrm{d}z = \mathbb{E}[u(W_\tau)].$$

Both perspectives point to the same strategy for estimating $u(x_0)$: uniformly sample a point x_1 on a ball around x_0. If x_1 is extremely close to the domain boundary (i.e., within the ε-shell Ω_ε), grab the boundary value $g(\bar{x}_1)$. Otherwise, evaluate $u(x_1)$. This reasoning leads to the recursive WoS estimator

$$\widehat{u}(x_k) := \begin{cases} g(\bar{x}_k) & x_k \in \partial\Omega_\varepsilon, \\ \widehat{u}(x_{k+1}) & \text{otherwise.} \end{cases} \tag{19}$$

The point x_{k+1} is drawn from a uniform distribution on the *largest* sphere centered at x_k, helping us reach the boundary in a small number of steps—Fig. 11, *left* depicts one possible “walk.” As shown in Fig. 28 (and in [Sawhney and Crane 2020, Figure 14]), terminating this walk in the ε-shell introduces negligible bias which diminishes at a rate of $O(1/\log\varepsilon)$ [Binder and Braverman 2012].

Source term. To incorporate a source term f, we must also add an estimate of the integral over $B(x_k)$ from Eq. 6 to $\widehat{u}(x_k)$ at each step of the walk. Here again we use a single-sample estimate at a point y_{k+1} (Fig. 11, *right*). Though y_{k+1} could be sampled uniformly, Sawhney and Crane [2020, Section 4.2] achieve better results by importance sampling the source term f, Green's function G, or by combining strategies via *multiple importance sampling* [Veach and Guibas 1995].

Absorption. Finally, to incorporate a constant absorption term σu, as in Eq. 4, we simply need to adopt the corresponding Green's function G^σ and Poisson kernel P^σ (given in supplemental material).

Unfortunately, an integral representation of our main PDE (Eq. 1) is not available, making it unclear how to apply WoS. In the next section we exploit structural similarities between the Feynman-Kac formula (Eq. 12) and the volume rendering equation (Eq. 14) to derive a new integral representation for PDEs with variable coefficients.

4 OUR METHOD

The standard WoS algorithm cannot handle the spatially varying coefficients $\alpha(x)$, $\vec{\omega}(x)$ and $\sigma(x)$ in Eq. 1, but it can solve a PDE with a spatially varying source term $f(x)$. We hence apply a series of transformations (Fig. 12) that convert Eq. 1 into an equivalent constant-coefficient screened Poisson equation (Eq. 23); we then use the integral version of screened Poisson (Eq. 7) to define a variable-coefficient WoS estimator. From a stochastic perspective, these transformations are equivalent to writing Feynman-Kac purely in terms of Brownian motion, rather than a generic diffusion process.

Our method applies whenever Eq. 1 is elliptic—which holds if the diffusion coefficient $\alpha(x)$ is strictly positive and the screening coefficient $\sigma(x)$ is nonnegative [Evans 1998; Friedman and Fu 1975]. For brevity, we here omit the drift term $\vec{\omega}(x) \cdot \nabla u(x)$, though the approach is unchanged for equations with drift (see App. A.3). For readers not interested in the derivation, Eq. 26 gives the final integral formulation of Eq. 1; Sec. 5 describes our final algorithms.

4.1 Transformations

Second order. We first expand the 2nd order term $\nabla \cdot (\alpha(x)\nabla u)$ in Eq. 1 via the product rule. We then divide the resulting equation by $\alpha(x)$, and apply the identity $\nabla \ln(\alpha(x)) = \nabla\alpha(x)\,/\alpha(x)$ to get

$$\Delta u(x) + \nabla \ln(\alpha(x)) \cdot \nabla u(x) - \frac{\sigma(x)}{\alpha(x)} u(x) = -\frac{f(x)}{\alpha(x)}. \tag{20}$$

At this point the 2nd order term Δu no longer has variable coefficients, but spatial variation in the lower order terms remains.

First order. A *Girsanov transformation* re-expresses a random process under a change of probability measure, e.g., from a generic diffusion process X_t to an ordinary Brownian motion W_t [Øksendal 2003, Ch. 8]. As shown in App. A.1, applying this transformation to Eq. 20 eliminates the 1st order operator from Eq. 20, shifting all spatial variation into the 0th order term:

$$\begin{aligned} \Delta U(x) - \sigma'(x)U(x) &= -f'(x) && \text{on } \Omega, \\ U(x) &= g'(x) && \text{on } \partial\Omega. \end{aligned} \tag{21}$$

Here,

$$U(x) \coloneqq \sqrt{\alpha(x)}\, u(x), \quad g'(x) \coloneqq \sqrt{\alpha(x)}\, g(x), \quad f'(x) \coloneqq \frac{\sqrt{\alpha(x)}}{\alpha(x)} f(x),$$

$$\text{and} \quad \sigma'(x) \coloneqq \frac{\sigma(x)}{\alpha(x)} + \frac{1}{2}\left(\frac{\Delta\alpha(x)}{\alpha(x)} - \frac{|\nabla \ln(\alpha(x))|^2}{2}\right).$$

Equation 21 is equivalent to our original PDE with variable coefficients in Eq. 1, which can be verified by substituting the expressions for U, g', f' and σ' back into this equation.

Unlike Feynman-Kac, which involves a diffusion process X_t, the stochastic formula for Eq. 21 uses only simple Brownian motion W_t:

$$U(x) = \mathbb{E}\left[\int_0^\tau e^{-\int_0^t \sigma'(W_s)\,ds} f'(W_t)\,dt \; + \; e^{-\int_0^\tau \sigma'(W_t)\,dt}\, g'(W_\tau)\right]. \tag{22}$$

Zeroth Order. The only remaining term on the left-hand side of Eq. 21 with spatially varying coefficients is the 0th order screening term $\sigma'(x)U$. We hence apply a transformation inspired by delta tracking (Sec. 2.5) to shift this heterogeneity to a source term on the right-hand side. In doing so, we introduce a coefficient $\bar{\sigma} > 0$ by subtracting $\bar{\sigma}U$ from both sides of Eq. 21. The result is a PDE with the same basic form as a screened Poisson equation:

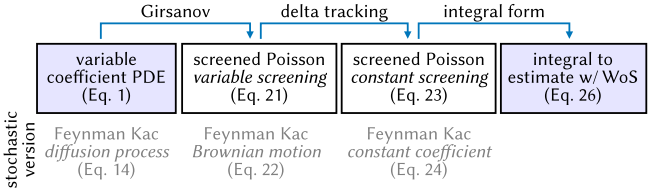

Girsanov
delta tracking
integral form
variable coefficient PDE (Eq. 1)
screened Poisson *variable screening* (Eq. 21)
screened Poisson *constant screening* (Eq. 23)
integral to estimate w/ WoS (Eq. 26)
stochastic version
Feynman Kac *diffusion process* (Eq. 14)
Feynman Kac *Brownian motion* (Eq. 22)
Feynman Kac *constant coefficient* (Eq. 24)

Fig. 12. An overview of the transformations we apply to Eq. 1 to derive an integral formulation amenable to Monte Carlo estimation with WoS.

$$\begin{aligned} \Delta U(x) - \bar{\sigma}U(x) &= -\underbrace{\big(f'(x) + (\bar{\sigma} - \sigma'(x))U(x)\big)}_{=:\, f'(x,\,U)} \quad \text{on } \Omega, \\ U(x) &= g'(x) \quad \text{on } \partial\Omega. \end{aligned} \tag{23}$$

Though only constant coefficients now appear on the left-hand side, *no approximation of any kind has been introduced.* Unlike a typical linear PDE however, the solution U appears on the right-hand side. As in volume rendering, we account for this recursive dependence by applying recursive Monte Carlo estimation (Sec. 5)—a strategy not available in the traditional setting of, e.g., finite element methods.

Like the transformed VRE in Eq. 16, and the stochastic formulas in Eq. 11, the stochastic expression for Eq. 23 also has a transmittance function $e^{-\bar{\sigma}t}$ that no longer varies spatially:

$$U(x) = \mathbb{E}\left[\int_0^\tau e^{-\bar{\sigma}t} f'(W_t, U)\,dt \; + \; e^{-\bar{\sigma}\tau}\, g'(W_\tau)\right]. \tag{24}$$

4.2 Integral Representation

We can now express the solution to Eq. 23 using the integral form of the constant coefficient screened Poisson equation (Eq. 7):

$$U(x) = \int_{B(c)} f'(y, U)\, G^{\bar{\sigma}}(x,y)\,dy \; + \; \int_{\partial B(c)} U(z)\, P^{\bar{\sigma}}(x,z)\,dz. \tag{25}$$

(Recall that here x can be any point inside $B(c)$—not just its center.) Finally, we make the substitution $U(x) = \sqrt{\alpha(x)}\, u(x)$ from Eq. 21 to write this integral in terms of the original function u:

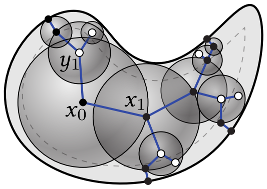
$$\boxed{\begin{aligned} u(x) = \tfrac{1}{\sqrt{\alpha(x)}} \Big(\textstyle\int_{B(c)} f'(y, \sqrt{\alpha}\, u)\, G^{\bar{\sigma}}(x,y)\,dy \; + \\ \textstyle\int_{\partial B(c)} \sqrt{\alpha(z)}\, u(z)\, P^{\bar{\sigma}}(x,z)\,dz\Big). \end{aligned}} \tag{26}$$

Unlike Eq. 7, we now have a *recursive* integral equation (due to the presence of u in both volume and boundary terms), which can be solved via recursive application of Monte Carlo integration.

4.3 Monte Carlo Estimator

A single-sample estimator for Eq. 26 at a point $x_k \in B(c)$ is given by

$$\widehat{u}(x_k) := \frac{1}{\sqrt{\alpha(x_k)}}\left(\overbrace{\frac{f'(y_{k+1}, \sqrt{\alpha}\,\widehat{u})\; G^{\overline{\sigma}}(x_k, y_{k+1})}{p^B(y_{k+1})\;\mathbb{P}^B}}^{\text{evaluate volume term with probability } \mathbb{P}^B} + \overbrace{\frac{\sqrt{\alpha(x_{k+1})}\;\widehat{u}(x_{k+1})\;P^{\overline{\sigma}}(x_k, x_{k+1})}{p^{\partial B}(x_{k+1})\;\mathbb{P}^{\partial B}}}^{\text{evaluate boundary term with probability } \mathbb{P}^{\partial B}}\right), \quad (27)$$



where y_{k+1} and x_{k+1} are points sampled inside and on the surface of $B(c)$ according to probability densities p^B and $p^{\partial B}$ (resp.). Values $\mathbb{P}^B, \mathbb{P}^{\partial B} \in (0, 1]$ control the probability of sampling the volume and boundary terms (resp.). Letting $\mathbb{P}^B = \mathbb{P}^{\partial B} = 1$ yields exponential growth in the number of steps, since each walk branches into two (see inset), and walks do not terminate until both y_{k+1} and x_{k+1} are contained in $\partial\Omega_\varepsilon$. In Sec. 5, we develop two WoS algorithms that avoid branching via careful choice of $\mathbb{P}^B$ and $\mathbb{P}^{\partial B}$.

5 ALGORITHMS

Due to the diversity of heterogeneous phenomena in nature, different algorithms for solving the VRE adopt different strategies to trade off between variance, bias, and computational cost [Novák et al. 2018]. Likewise, an algorithm for solving diffusive PDEs will be more effective when it is well-matched to the way coefficients are distributed in space. We provide a unified integral framework, based on Eq. 27, which enables us to explore WoS variants appropriate for different problems—akin to the unidirectional estimator in Georgiev et al. [2019, Eq. 14]. In particular, we devise two estimators inspired by *delta tracking* (Sec. 2.5) and *next-flight* [Cramer 1978] methods from volume rendering. Sec. 3 of the supplement provides pseudo-code for both methods; Sec. 2 shows how these methods can also be used to compute the solution gradient $\nabla u(x)$ needed in a variety of applications (see Sawhney and Crane [2020, Section 3]).

5.1 Delta Tracking Variant

To avoid branching, our delta tracking variant of WoS uses a special property of the Poisson kernel $P^{\overline{\sigma}}$ of a screened Poisson equation when x_k is at the ball center. Assuming $\overline{\sigma} > 0$, and letting $|G^{\overline{\sigma}}(x)|$ be the integral of $G^{\overline{\sigma}}$ over $B(x)$ (supplemental, Sec. 1.1), we have

$$P^{\overline{\sigma}}(x_k, x_{k+1}) = \frac{1 - \overline{\sigma}|G^{\overline{\sigma}}(x_k)|}{|\partial B(x_k)|}.$$

Since $\overline{\sigma}|G^{\overline{\sigma}}(x_k)| \in (0, 1)$ (see Eq. 4 of the supplement), we can sample the boundary and volume terms with probability $\mathbb{P}^{\partial B} := 1 - \overline{\sigma}|G^{\overline{\sigma}}(x_k)|$ and $\mathbb{P}^B := 1 - \mathbb{P}^{\partial B}$, yielding a non-branching estimator

$$\widehat{u}(x_k) := \begin{cases} g(\overline{x}_k), & \overline{x}_k \in \partial\Omega_\varepsilon, \\ \frac{1}{\overline{\sigma}\sqrt{\alpha(x_k)}} f'(y_{k+1}, \sqrt{\alpha}\,\widehat{u}), & \mu \sim \mathcal{U} \le \overline{\sigma}|G^{\overline{\sigma}}(x_k)|, \\ \sqrt{\alpha(x_{k+1})/\alpha(x_k)}\;\widehat{u}(x_{k+1}) & \text{otherwise.} \end{cases} \quad (28)$$

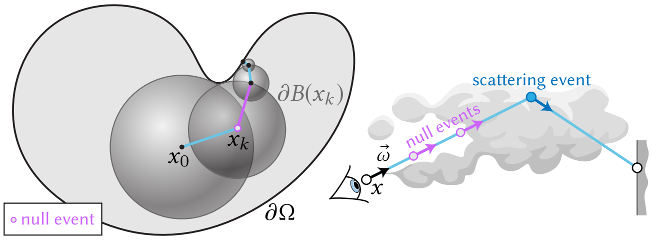
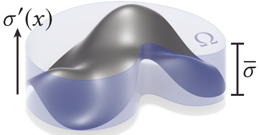


Fig. 13. *Left*: Unlike standard WoS for constant coefficient problems, the delta tracking variant jumps to a random point either inside or on the surface of the largest ball $B(x_k)$ centered at x_k. As in rendering (*right*), null-events sampled inside $B(x_k)$ re-weight the solution estimate by $\overline{\sigma} - \sigma'(x)$ (see definition of f' in Eq. 23) to account for spatial variations in the PDE.

This estimator importance samples y_{k+1} and x_{k+1} via the densities $p^B := G^{\overline{\sigma}}(x_k, y_{k+1})/|G^{\overline{\sigma}}(x_k)|$ and $p^{\partial B} := 1/|\partial B(x_k)|$, respectively. Use of constant-coefficient kernels $G^{\overline{\sigma}}$ and $P^{\overline{\sigma}}$ is critical, since the kernels for the original coefficient function $\sigma'(x)$ (Eq. 21) are not known in closed form. However, spatial variation in $\sigma'(x)$ is still accounted for by f', which corresponds to probabilistically sampling null-events (Fig. 13).



The coefficient $\overline{\sigma}$ is the only free parameter in this algorithm, and must be strictly positive to ensure that $\mathbb{P}^B > 0$. In volume rendering one typically lets $\overline{\sigma} = \max(\sigma(x))$, which enables closed-form sampling of volumetric events (absorption, scattering, or null scattering) and boundary reflections. We instead let $\overline{\sigma} = \max(\sigma'(x)) - \min(\sigma'(x))$, since in general $\sigma'(x)$ (from Eq. 21) can have both positive and negative values at different points $x \in \Omega$ (see inset). More recent volume rendering research [Georgiev et al. 2019; Novák et al. 2014] treats $\overline{\sigma}$ as a control variate rather than a bound, to reduce variance based on the profile of the coefficients. In conjunction with clever choices for p^B and $p^{\partial B}$, these strategies can be more efficient than delta tracking; we leave such extensions to future work.

5.2 Next-Flight Variant

Our delta tracking variant of WoS takes far more steps as $\overline{\sigma}$ increases (see Fig. 14), since the Green's function becomes more localized (Fig. 6) and it becomes more likely that we sample a point in the volume than on the boundary ($\mathbb{P}^B > \mathbb{P}^{\partial B}$). Longer walks are ultimately more expensive, since distance queries are usually the bottleneck for WoS (much like ray intersections in path tracing).

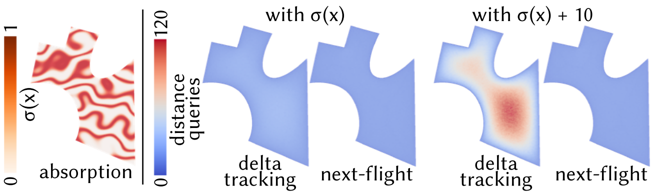


Fig. 14. As $\overline{\sigma}$ increases, the delta tracking variant requires far more distance queries (reducing run-time performance), while the number of queries for the next-flight strategy is unchanged.

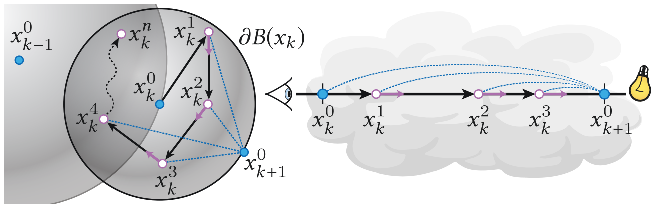


Fig. 15. *Left*: Our next-flight variant of WoS addresses heterogeneity in PDE coefficients by evaluating *off-centered* versions of the Green's function and Poisson kernel for every ball $B(x_k^0)$ in a walk. Short walks in $B(x_k^0)$ all terminate at a point $x_{k+1}^0 \in \partial B(x_k^0)$ to avoid branching. *Right*: The next-flight method for volume rendering likewise estimates transmittance along a ray with a predetermined endpoint.

We hence propose a variant based on the *next-flight* scheme of Cramer [1978], which takes big steps even when $\bar{\sigma}$ is large (Fig. 15). As usual we walk along points $x_0^0, x_1^0, \ldots$ sampled from successive spheres $\partial B(x_{k-1})$, always estimating both boundary and volume terms ($\mathbb{P}^{\partial B} = \mathbb{P}^B = 1$). But rather than start a new walk to the boundary for the volume term, we take a "short off-center walk" $x_k^1, x_k^2, \ldots, x_k^M$ within each ball $B(x_k^0)$, and re-use the estimate of the boundary contribution at x_{k+1}^0 for all steps in this short walk. An expression for this estimator is obtained by recursively expanding the definition of $\widehat{u}$ in the volume term of Eq. 27:

$$\widehat{u}(x_k^0) := \frac{1}{\sqrt{\alpha(x_k^0)}}\left(\sqrt{\alpha(x_{k+1}^0)}\;\widehat{u}(x_{k+1}^0)\;\widehat{T}(x_k^0, x_{k+1}^0) \;+\; \widehat{S}(x_k^0)\right), \tag{29}$$

where

$$\widehat{T}(x_k^0, z) := \sum_{j=0}^{M} \frac{P^{\bar{\sigma}}(x_k^j, z)}{p^{\partial B}(z)} \prod_{l=0}^{j-1} W(l),$$

$$\widehat{S}(x_k^0) := \sum_{j=1}^{M} \frac{f(x_k^j)}{\sqrt{\alpha(x_k^j)}\;(\bar{\sigma} - \sigma'(x_k^j))} \prod_{l=0}^{j-1} W(l),$$

$$W(l) := \frac{G^{\bar{\sigma}}(x_k^l, x_k^{l+1})\;(\bar{\sigma} - \sigma'(x_k^{l+1}))}{p^B(x_k^{l+1})}.$$

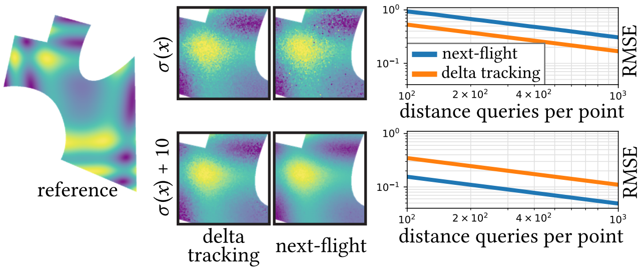
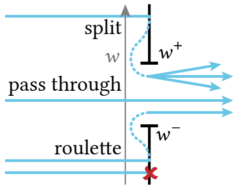


Fig. 16. Comparison of WoS variants for a screened Poisson problem with spatially-varying absorption. *Top:* For smaller coefficients $\sigma(x)$, next-flight exhibits higher variance due to greater correlation among samples. *Bottom:* for larger coefficients $\sigma(x) + 10$ delta tracking now exhibits more variance for equal compute time, since it requires far more distance queries.

For each point x_k^l, the subscript k indexes steps in the walk; the superscript l indexes points along the short walk in $B(x_k^0)$. The number of terms M is determined by using the "throughput" $\prod_{l=0}^{j-1} W(l)$ as a probability for *Russian roulette* [Pharr et al. 2016, Ch. 13.7]. Parameters $\bar{\sigma}$, p^B and $p^{\partial B}$ are the same as in Sec. 5.1, but the Green's function $G^{\bar{\sigma}}$ and Poisson kernel $P^{\bar{\sigma}}$ must now be evaluated and sampled using general off-centered formulas (see Sec. 1.2 and Sec. 1.4 of the supplemental).

A key benefit of the next-flight scheme is that it does not need additional distance queries to evaluate $\widehat{T}$ and $\widehat{S}$ within $B(x_k)$. Decreased computation does however come at the cost of increased correlation in $\widehat{T}$ due to the reuse of $\widehat{u}(x_{k+1})$; see Fig. 16.

5.3 Variance Reduction



Both WoS variants exhibit the standard rate of convergence for Monte Carlo (see plots of root mean squared error (RSME) in Figs. 16, 17, 24 and 28). However, we can further reduce variance using a *weight window* strategy from neutron transport [Hoogenboom and Légrády 2005]. The basic observation is that different walks can have very different cumulative weights, leading to high estimator variance—especially for long walks. For instance, in each delta tracking step the estimate gets scaled by one of the coefficients of $\widehat{u}$ in Eq. 28, leading to very small (or large) weights when $\bar{\sigma}$ is large. A weight window (inset) helps ensure that all walks have a similar

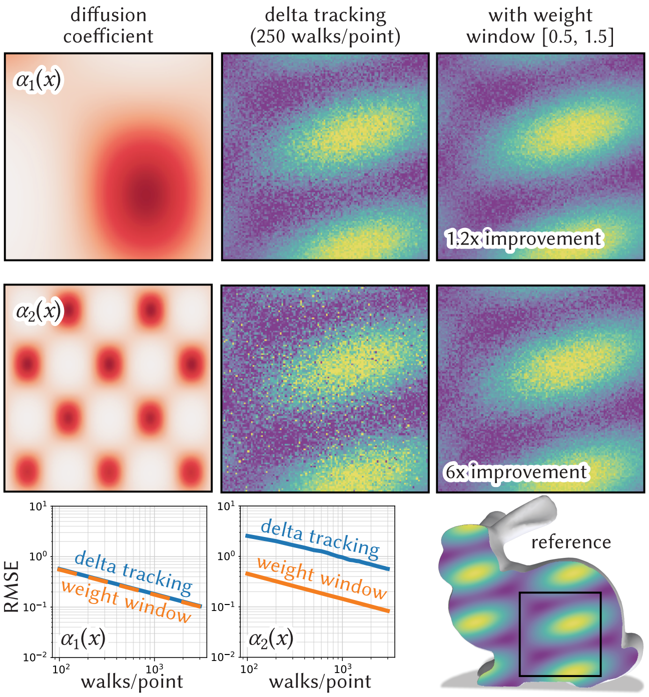


Fig. 17. Weight windows significantly reduce variance for problems with large variation in coefficients, as seen here for the problem $\nabla \cdot (\alpha(x)\nabla u) = -f(x)$ with two different coefficient functions $\alpha_1(x), \alpha_2(x)$.

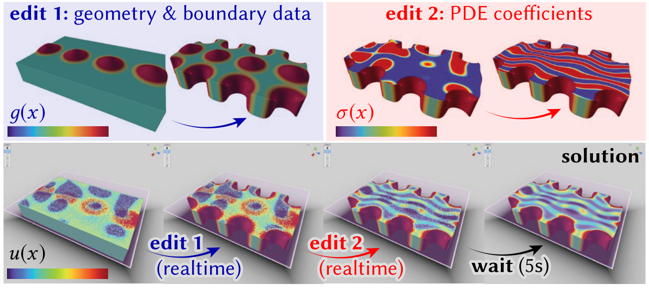


Fig. 18. Our approach is ideal for interactive editing since it operates directly on the original scene representation (here, signed distance fields composed via CSG operations) and provides instant feedback after updates to scene geometry and boundary conditions (edit 1) or PDE coefficients (edit 2).

weight by either using Russian roulette to terminate low-weight walks (which contribute little to the overall estimate), or *splitting* high-weight walks into multiple lower-weight walks (promoting better exploration of the domain).

At each step, a walk is allowed to continue if its weight w is within the window $[w^-, w^+]$ (we use a static window $[0.5, 1.5]$). If $w < w^-$, the walk is terminated with a Russian roulette probability of w/w^-. If $w > w^+$, it is split into roughly $m := w/w^+$ new walks, each of weight w/m. Since m is generally not an integer, we use the *expected value splits* approach of Booth [1985] which spawns $n := \lfloor m \rfloor$ walks with probability $n+1-m$, and $n+1$ walks otherwise.

Figure 17 highlights the effectiveness of a static weight window for reducing variance in problems with high frequency coefficients and large parameter bounds $\bar{\sigma}$. Prior work in neutron transport and rendering has shown that choosing the window size adaptively can improve efficiency by up to an order of magnitude [Booth and Hendricks 1984; Vorba and Křivánek 2016; Wagner and Haghighat 1998]. We leave such optimization to future work.

6 IMPLEMENTATION AND RESULTS

We next discuss practical considerations pertaining to our modified WoS algorithms. We also highlight the unique benefits of our approach on a range of example problems inspired by engineering and design applications.

6.1 Implementation

Similar to a scene in a renderer, a PDE is encoded by a description of the domain geometry Ω, and the functions $\alpha, \vec{\omega}, \sigma, f, g$ in Eq. 1.

Functions are provided as arbitrary callback routines that return a value for any query point x; unlike FEM, finite difference methods, etc., they need not be discretized or approximated in a finite-dimensional basis. The gradient and Laplacian of coefficients $\alpha(x)$ and $\vec{\omega}(x)$ (see Sec. 4.1 and App. A.3) can be evaluated via any standard technique (e.g., automatic differentiation); the bounding parameter $\bar{\sigma} := \max(\sigma'(x)) - \min(\sigma'(x))$ is computed as in volume rendering [Novák et al. 2018], e.g., by random or regular sampling.

Geometry is encoded by a function that gives the point $\bar{x} \in \partial\Omega$ closest to any query point $x \in \Omega$ (hence the radius of the largest empty ball $B(x)$). Such *closest point queries* are easily evaluated for a wide variety of boundary representations (polygonal meshes, spline patches, implicit surfaces, etc.) and can be accelerated via standard spatial hierarchies such as a *bounding volume hierarchy (BVH)* or *octree* [Ericson 2004; Intel 2013; Museth 2013]. Unlike finite element mesh generation (or node placement for meshless FEM), an acceleration hierarchy uses very little memory and can be built almost instantaneously even for very detailed models (Fig. 3, *top*). Moreover, unlike a bad mesh, a poorly-constructed hierarchy harms only performance—not correctness or accuracy. For experiments we use a basic CPU-based axis aligned BVH for triangle meshes [Sawhney et al. 2020], though ray tracing hardware in recent GPUs shows potential for further acceleration of other queries [Burgess 2020].

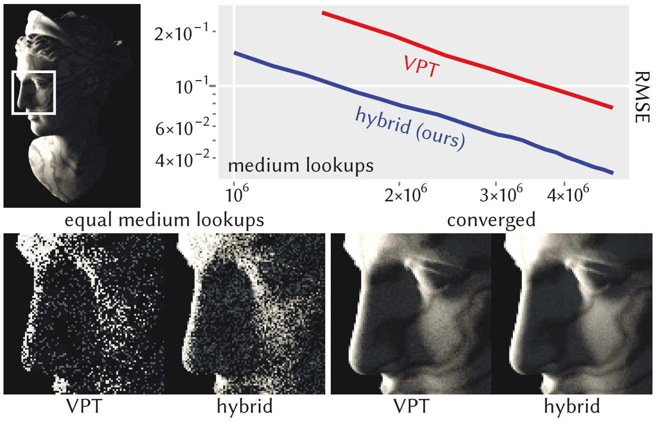


Fig. 19. Our hybrid subsurface scattering strategy yields lower relative mean squared error than VPT for an equal number of medium lookups.

6.2 Interactive Editing

Monte Carlo methods are popular in rendering because they provide immediate feedback that can be progressively improved—enabling engineers and artists to quickly iterate on designs. We explored this modality for PDEs by implementing a GPU version of our solver in Unity (Fig. 18) where domain geometry is encoded by a *signed distance function (SDF)*, and visualized via *sphere tracing* [Hart 1996]; WoS need only estimate the solution at points visible to the camera. This setup allows us to interactively explore problems of immense geometric complexity—for instance, Fig. 1 shows a scene with an infinite, aperiodic arrangement of detailed models with high-frequency material coefficients, which would be impractical or impossible for conventional PDE solvers. In fact, as shown in Fig. 23, meshing even a small region of this scene is prohibitively expensive. Fig. 18 shows interactive editing via constructive solid geometry (CSG) operations, which are easily represented *without meshing* via implicit functions. The solution is visualized on a 2D slice plane, and samples are accumulated progressively until the scene changes—in this case our solver runs at 60 frames per second.

6.3 Subsurface Scattering

Our Monte Carlo PDE solvers are easily combined with Monte Carlo renderers, since both support pointwise evaluation. One interesting use case is *subsurface scattering* in dense, high-albedo media like milk or marble. Since light in such media experiences thousands of scattering events, a *diffusion approximation* [Jensen et al. 2001] is

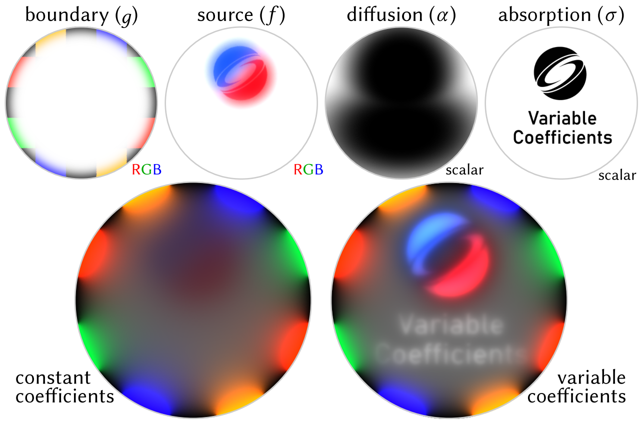


Fig. 20. Our method enables us to augment classic diffusion curve images with spatially varying coefficients that provide greater artistic control. For instance, source terms f have limited use in classic diffusion curves, since they get severely blurred out *(left)*. By locally adjusting the diffusion strength α, we can decorate the image with interesting textures or decals *(right)*.

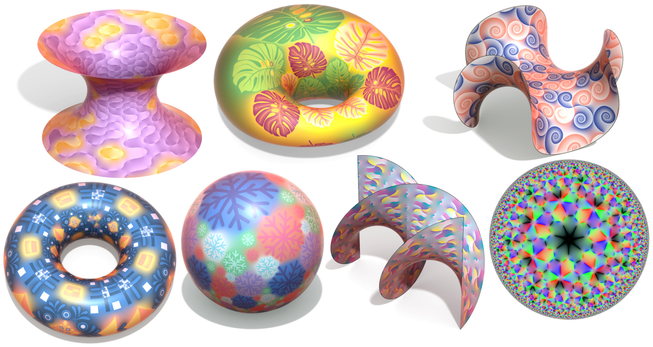


Fig. 21. Variable coefficients enable us to extend WoS to curved domains (not previously possible). Here we directly resolve intricate boundary conditions for diffusion curves—without generating a fine surface mesh that conforms to boundary curves or a spatially-adaptive grid in the parameter domain.

often much faster than direct estimation via volumetric path tracing (VPT). Popular methods based on variants of *dipole approximation* make further simplifying assumptions, such as homogeneity and local geometric planarity, which often induce more error than the diffusion assumption itself [d'Eon and Irving 2011; Donner et al. 2008; Habel et al. 2013; Jensen et al. 2001]. Other methods use standard PDE solvers to evaluate diffusion [Arbree et al. 2011; Koerner et al. 2014], but face the usual challenges of volumetric meshing and global solves—and the difficulty of coupling with a renderer.

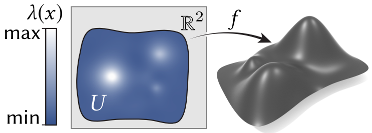
We avoid such challenges by combining VPT and WoS via a simple rule: at each path vertex we generate a free-flight distance (as in VPT) *and* compute the distance to the medium boundary (as in WoS). We then continue with either VPT or WoS based on the larger distance. Unlike dipole techniques, this strategy directly models heterogeneity in the medium; it is conceptually similar to shell tracing [Moon et al. 2008; Müller et al. 2016] or condensed history neutron transport [Fleck and Canfield 1984], since each WoS step effectively aggregates a large number of VPT steps. In practice, it tends to take more accurate VPT steps near the boundary and large WoS deeper in the medium, yielding significantly less variance than pure VPT for an equal number of medium lookups (Fig. 19).

6.4 Heterogeneous Diffusion Curves

WoS is attractive for *diffusion curves* [Orzan et al. 2008], since it provides real-time progressive previews that are easily implemented on the GPU [Quilez 2020], can be applied to a zoom-in without first computing a coarse global solution (as done by Orzan et al. [2008, Section 3.2.4]), and avoids aliasing of fine features due to grid resolution [Sawhney and Crane 2020, Figure 16]. Our method makes it possible to generalize classic diffusion curves by also painting a source $f(x)$, diffusion coefficient $\alpha(x)$, and absorption coefficient $\sigma(x)$—Fig. 20 shows one example. The added benefit is that, unlike constant-coefficient diffusion images, the source term $f(x)$ is no longer severely blurred—enabling one to add interesting decals or background texture, while still smoothly diffusing color over regions without such details. Variable absorption $\sigma(x)$ helps to further emphasize detail, since the strength of the source contribution is roughly $1/\sigma(x)$. Akin to "texture shaders" [Bowers et al. 2011; Prévost et al. 2015], this enables an enriched design space spanning diffusion curves and traditional 2D graphics, which previous approaches achieved via simpler alpha blending.

6.5 Walk on Curved Surfaces

A basic hypothesis of the original WoS algorithm is that a random walk exits every point on the boundary of a ball with equal probability (Sec. 3.2). However, on surfaces with non-constant curvature this hypothesis no longer holds: intuitively, more walkers will escape through the "valleys" than through the "mountains." As a result, standard WoS cannot be used for many algorithms in geometric and scientific computing that need to solve equations on a surface/shell.



Our variable-coefficient scheme enables WoS to be applied to curved surfaces for the first time. In particular, consider any surface expressed as a *conformal parameterization* $f : \mathbb{R}^2 \supset U \longrightarrow \mathbb{R}^3$; conformal means that f distorts the surface by a positive scaling $\lambda(x)$ at each point $x \in U$, i.e., $J_f^T J_f = \lambda(x)I$, where I is the identity and J_f is the Jacobian of f. The Laplacian Δ_f of the curved surface is then related to the ordinary Laplacian via $\Delta = \lambda\Delta_f$. Hence, we can solve PDEs on the curved surface by replacing the usual diffusion coefficient $\alpha(x)$ with $\lambda(x)$. Fig. 21 shows several examples; for periodic domains (like the torus) our walks simply "wrap around." In theory, *every* surface admits a conformal parameterization (by the uniformization theorem [Abikoff 1981]), but in practice many important surfaces used in engineering (such as NURBS or other spline patches) are expressed in non-conformal coordinates. To directly handle such patches, we would need to extend our method to *anisotropic* diffusion coefficients—an important topic for future work. Note also that earlier work on diffusion curves for surfaces uses free-space 2D Green's functions [Sun et al. 2012], which provide only a rough proxy for a curved surface's true Green's function.

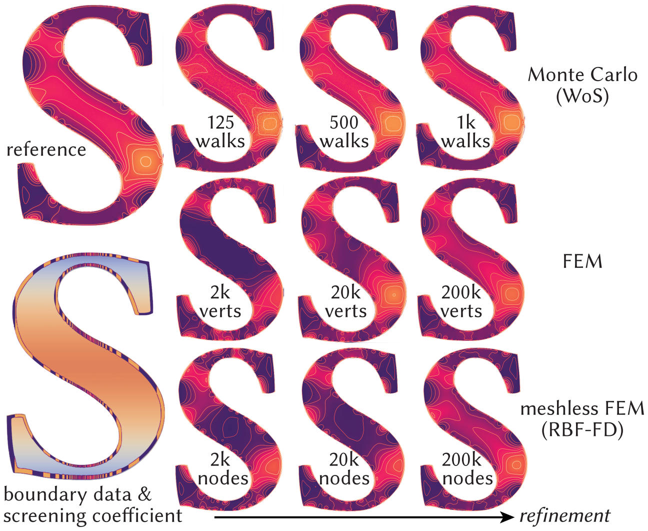


Fig. 22. With both FEM and MFEM, local aliasing of high-frequency boundary data yields large global errors in the solution, demanding significant refinement. In contrast, WoS always captures the global solution (even with very few samples); error instead manifests as high-frequency noise.

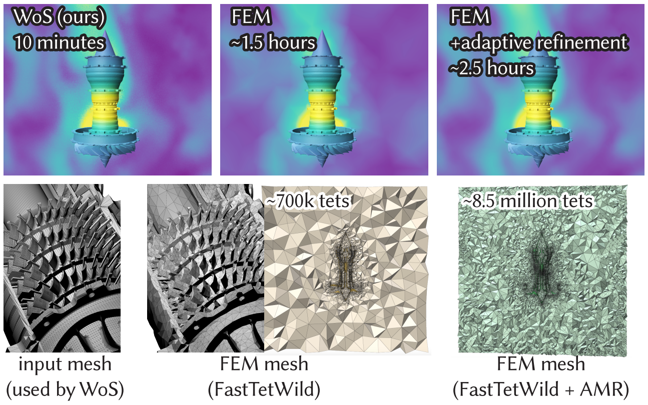


Fig. 23. Even when starting with a coarse approximation of just a single black body from Fig. 1, FEM takes immense time and memory to resolve detailed variations due to PDE coefficients. Here, initial coarse meshing by *FastTetWild* [Hu et al. 2020] takes about 1.5 hours to produce a mesh that cannot resolve fine details in solution or geometry (despite being quite large already). After 1 hour more of AMR via Anderson et al. [2021] the solution is better resolved, but the mesh size has blown up. Our CPU-based WoS implementation takes about 10 minutes total, on the same machine.

7 RELATED WORK AND COMPARISONS

7.1 PDE Solvers

There are many methods for solving variable-coefficient PDEs—far too many to review here. Instead, we examine the unique benefits of Monte Carlo in terms of the basic properties shared by all methods within each major class. Since our goal is to directly resolve fine details, we omit discussion of *homogenization* schemes, which approximate fine-scale behavior via coarse-scale models [Abdulle et al. 2012; Durlofsky 1991; Dykaar and Kitanidis 1992; Efendiev and Hou 2009; March et al. 2021]; Monte Carlo methods like ours may in fact help to accurately determine parameters for such schemes.

7.1.1 Finite Element Methods (FEM). All finite element methods (including BEM and meshless FEM) adopt a common framework: for a linear PDE $Lu = f$, find the best approximation to u in a finite-dimensional function space. E.g., standard *Galerkin FEM* uses an approximation $\widehat{u} := \sum_{i=1}^{n} u_i \phi_i$ in a basis $\phi_1, \ldots, \phi_n : \Omega \to \mathbb{R}$, where $u_i \in \mathbb{R}$ are unknown coefficients. Letting $\langle u, v \rangle := \int_\Omega u(x)v(x)\,\mathrm{d}x$ denote the L^2 inner product, one then seeks a $\widehat{u}$ satisfying

$$\langle L\widehat{u}, \phi_j \rangle = \langle \hat{f}, \phi_j \rangle, \quad j = 1, \ldots, n, \tag{30}$$

i.e., such that $\widehat{u}$ agrees with the true solution u when restricted to the subspace $V := \mathrm{span}(\{\phi_i\})$. To solve Eq. 30, one rewrites it as

$$\textstyle\sum_{i=1}^{n} u_i \langle L\phi_i, \phi_j \rangle = \sum_{i=1}^{n} f_i \langle \phi_i, \phi_j \rangle.$$

The inner products on the left- and right-hand side define *mass* and *stiffness* matrices (resp.), and are often further approximated via numerical quadrature. From this perspective, the only difference between flavors of FEM is the choice of basis functions ϕ_i (and the difficulty of integrating them). Otherwise, all finite element methods share a common set of challenges:

- They must all solve a globally coupled system of equations.
- They are all prone to spatial aliasing in the geometry, solution, boundary conditions, source terms, and/or coefficients, since any finite basis $\{\phi_i\}$ provides limited spatial resolution.
- They all demand spatial discretization (meshing or sampling) to define bases ϕ_i, which can be costly and error prone (Fig. 3).

In contrast, WoS can directly evaluate the solution at any point without meshing or global node placement, and without a global solve. Moreover, it does not suffer from aliasing in the solution or problem data, since functions are not restricted to a finite-dimensional subspace V (see Fig. 22). For these reasons, it is also difficult to make an exact performance comparison between WoS and traditional solvers in terms of target accuracy, since performance is contingent on several factors beyond just rates of convergence, such as the cost of mesh generation and refinement, parallel scalability and the need for local versus global evaluation of the solution.

Mesh-Based FEM. Most often, FEM bases ϕ_i are defined via polyhedral mesh elements. Quickly and robustly meshing large, detailed and/or imperfect geometry (e.g., with self-intersections) is an ongoing “grand challenge,” where even state-of-the-art methods can struggle (Fig. 3). This problem gets harder if the mesh must also be refined for spatially varying coefficients: even with intelligent *adaptive mesh refinement (AMR)* [Zienkiewicz and Zhu 1992a,b], meshing quickly becomes prohibitive (Fig. 23). More recent *a priori* p-refinement does not help, since it considers only element quality and not spatial frequencies in the solution or problem data [Schneider et al. 2018]. WoS bypasses meshing entirely, needing only a BVH (for closest point queries), which uses minimal memory and can be built in a fraction of a second—even for totally degenerate geometry (see [Sawhney and Crane 2020, Figure 2]).

Meshless FEM. Though *meshless FEM (MFEM)* seems like a natural alternative to grid-free Monte Carlo, the term “meshless” is a bit of misnomer: MFEM does not need a polyhedral mesh, but

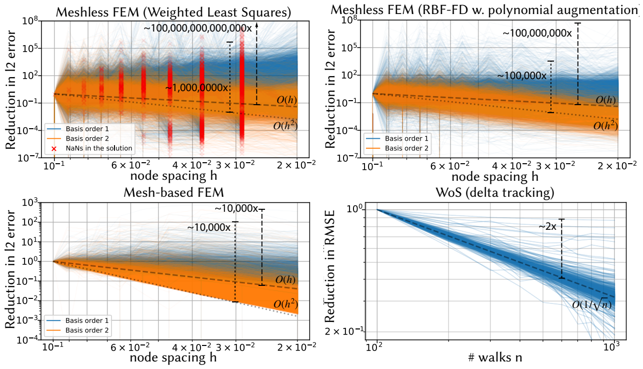


Fig. 24. Convergence of meshless FEM is not well-understood: such methods often fail to converge under refinement and/or show extremely large variation in error, as seen here for a variable-coefficient problem on models from the *Thingi10k* dataset [Hu et al. 2020; Zhou and Jacobson 2016]). Both mesh-based FEM and Monte Carlo methods come with rigorous convergence guarantees and behave much more predictably under refinement.

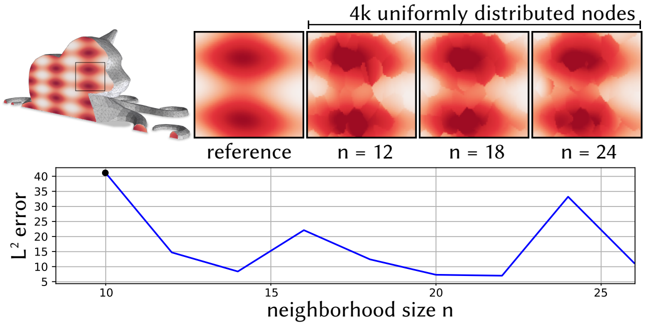


Fig. 25. In practice, it can be difficult to find reliable parameters for meshless FEM—for instance, increasing neighborhood size often *increases* error in an unpredictable way. In contrast, WoS requires no parameter tuning.

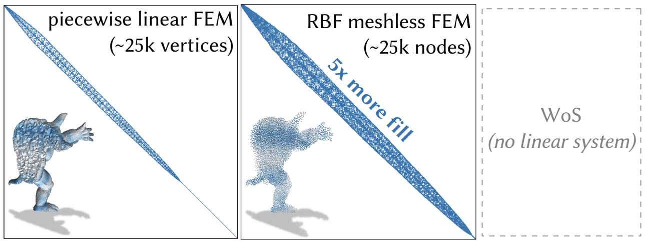


Fig. 26. Meshless FEM must solve a much denser linear system than even standard FEM—whereas WoS avoids solving a global system altogether.

must still discretize the domain by carefully arranging a collection of nodes (Fig. 4). Nodes are then associated with bases ϕ_i, such as *radial basis functions (RBFs)*, to build mass and stiffness matrices. To couple bases with overlapping support one must identify neighbors—forming a global graph structure similar in size and complexity to a polyhedral mesh. Node locations must satisfy criteria that are often just as difficult and delicate to enforce as mesh quality criteria [Li and Liu 2007, Ch. 3], and often require global optimization akin to mesh smoothing [Slak and Kosec 2019]. Moreover, just as one bad element can ruin an FEM solution, bad node placement can lead to catastrophic failure (e.g., NaNs in the solution—see Fig. 24, *top left*). One can adaptively sample nodes to mitigate spatial aliasing—but unlike mesh-based FEM, adaptive refinement for MFEM is poorly understood (lacking, e.g., rigorous convergence guarantees). Finally, typical MFEM bases are *approximating* rather than *interpolating*, complicating enforcement of boundary conditions [Fries et al. 2004; Nguyen et al. 2008]; some methods hence modulate bases by distance-like functions [Shapiro and Tsukanov 1999], but still effectively discretize functions on the interior by choosing a finite basis $\{\phi_i\}$.

As shown in Fig. 24, a more serious challenge with MFEM is *stagnation*: only until very recently [Bayona et al. 2019, 2017; Flyer et al. 2016], MFEM methods might fail to converge without careful problem-specific tuning of parameters such as neighborhood size. More damning is that increasing the neighborhood size does not always make the solution better (see Fig. 25). Moreover, whereas convergence of *adaptive* FEM is rigorously understood [Mekchay and Nochetto 2005], there is a dearth of corresponding results for adaptive MFEM schemes—especially important for problems with detailed geometry and coefficients. In practice, MFEM also requires denser mass/stiffness matrices (Fig. 26) than those used in mesh-based FEM, while often providing less accurate results. For instance, methods such as *RBF-FD with polynomial augmentation* [Flyer et al. 2016] that converge under refinement require at least order-2 bases.

On the whole, MFEM is not known for its reliability—in stark contrast, WoS guarantees that the expected solution equals the true solution of the smooth PDE without any parameter tuning whatsoever. Moreover, unlike MFEM, WoS is truly "meshless": at no point does one require a global sampling or meshing of the domain. Finally, though MFEM has been around for a long time, it has not seen nearly as much use in practice as mesh-based methods (e.g., with very few open source or commercial packages available).

Boundary Element Methods (BEM). Boundary element methods approximate the solution using bases functions ϕ_i associated only with elements of a boundary mesh (such as free-space Green's functions). These methods draw a natural comparison with WoS, since they need not discretize the interior of the domain. However, there is a significant difference in capabilities: whereas WoS easily handles problems with source terms and spatially varying coefficients on the domain interior, basic BEM ignores these terms altogether (see Fig. 27). In order to handle general interior terms, one must couple BEM with a second interior solver such as FEM, MFEM, or FD—inheriting all the same challenges [Coleman et al. 1991; Costabel 1987; Partridge et al. 2012]. Moreover, even for problems involving only boundary terms, BEM must discretize the boundary geometry, leading to spatial aliasing in both boundary data and geometry. Unlike FEM/MFEM, BEM must solve a globally coupled *dense* system of equations, demanding special techniques like *hierarchical matrix approximation* [Hackbusch 2015] to obtain reasonable performance.

7.1.2 Finite Difference (FD) Methods.

The main conceptual difference between finite difference and finite element methods is that degrees of freedom now represent point samples of the unknown function at nodal points, rather than coefficients in a finite basis.

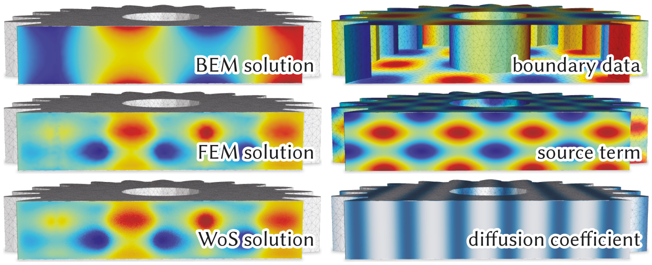


Fig. 27. Unlike FEM and Monte Carlo, traditional BEM ignores volumetric functions (e.g.coefficients and source terms) that affect the PDE solution.

Derivatives are likewise evaluated via Taylor series approximation (e.g., using finite difference formulas), rather than by taking derivatives of basis functions.

On the one hand, FD schemes are attractive due to the simplicity of implementation on a regular grid. They are however less than ideal for PDEs with nonuniform coefficients, where the uniformity of grid cells can lead to significant numerical diffusion [Umansky et al. 2005], spurious negative values [Sharma and Hammett 2007], and locking/stagnation [Babuška and Suri 1992]. Another major challenge is spatial adaptivity: for many elliptic PDEs, refining the whole domain is overkill. Hierarchical structures like *octrees* can be used to adaptively refine solutions [Gibou et al. 2018; Losasso et al. 2006], yet come with their own challenges (e.g., less coherent memory access, and increased complexity of implementation). Enforcement of boundary conditions may also not be straightforward, since cell boundaries are typically axis aligned [Causon and Mingham 2010].

On the whole, finite differences suffer from the same basic challenges as finite element methods: one must spatially discretize the domain, boundary conditions, source term, and coefficient functions, leading to either aliasing or oversampling. Moreover, one must solve a globally coupled system of equations over the entire domain, rather than concentrating computational effort only at points or regions of interest (as with Monte Carlo).

Material Point Methods. Material point methods [Jiang et al. 2016], such as PIC [Harlow and Welch 1965], FLIP [Brackbill and Ruppel 1986; Zhu and Bridson 2005], APIC [Jiang et al. 2015], and MPM [Sulsky et al. 1995] are popular for time-dependent computational mechanics problems involving large-scale deformation (fluids, plasticity, etc.). These methods are also sometimes referred to as "meshless", but they are not (in general) MFEM schemes as defined in Sec. 7.1.1. Rather, these methods use particles to approximate advection, and a background grid to solve elliptic problems (such as pressure projection in fluids). Critically, for the problems we consider here (time-*independent* elliptic PDEs), there is no advection component, and MPM reduces to simply solving elliptic equations on a grid—with the same trade-offs discussed above.

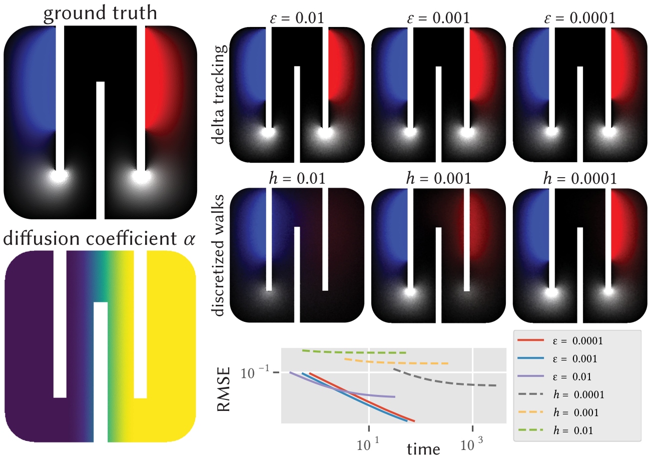


Fig. 28. *Top Row, Right*: Our WoS algorithms correctly resolve the boundary conditions for any value of ε; shrinking the ε-shell reduces bias in the solution in a predicable manner, with little impact on performance (*Bottom Right*). *Middle Row, Right*: Solving the same PDE with the integration scheme in Eq. 31 eventually resolves the boundary conditions with a finer step-size, though at the detriment of run-time performance (*Bottom Right*).

7.1.3 *Stochastic Methods.*

Not all PDE solvers need to discretize space—the notable exception are Monte Carlo methods based on continuous random processes such as Brownian motion. The stochastic approach to deterministic boundary value problems centers on the simulation of random walks that in aggregate solve a large class of elliptic PDEs [Øksendal 2003]. Pointwise evaluation of PDE solutions has allowed this formulation to find extensive use in scientific disciplines such as mathematical finance [Black and Scholes 1973; Cox et al. 1985; Merton 1971; Merton and Samuelson 1992], computational physics and chemistry [Gillespie 1977; Grebenkov 2007; Mascagni and Simonov 2004a,b] and optimal control [Kalman 1960; Kappen 2007] (albeit often on simple geometric domains).

Discretized Random Walks. In place of WoS, one might try approximating the Feynman-Kac formula by directly simulating a diffusion process X_t with explicit time stepping [Higham 2001; Kloeden and Platen 2013], akin to *ray marching* [Tuy and Tuy 1984]:

$$X_{k+1} = X_k + \vec{\omega}(X_k)h + \sqrt{\alpha(X_k)}\,(W_{k+1} - W_k). \tag{31}$$

However, this approach introduces several sources of error. E.g., walks often leave the domain and must be clamped to the boundary (Fig. 29); shrinking h reduces error, but significantly slows down computation (Fig. 28). Also, nonlinear functions φ do not in general commute with an expectation ($\mathbb{E}[\varphi(X)] \neq \varphi(\mathbb{E}[X])$), hence it is not clear how to estimate the function $\exp(-\int_0^\tau \sigma(X_t)\,dt)$ in Eq. 12 in an unbiased way. Bias is exacerbated in problems with variable diffusion and drift coefficients which implicitly modify the ideal step size. In contrast, the ε-shell in WoS incurs only minuscule bias at the very end of a walk, leading to far less error overall (Fig. 28, *top*); the size of ε also has little effect on performance (Fig. 28, *bottom*). Note that as with ray marching in rendering [Kettunen et al. 2021], discretized walks do still exhibit fairly predictable and low variance, as long as some bias is tolerable.

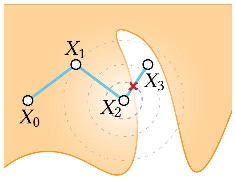


Fig. 29. Discretized random walks can leave the domain, biasing estimates.

Continuous Random Walks. A variety of so-called *grid-free Monte Carlo methods* have been developed for simulating random walks without spatial discretization. The chief example is the WoS method from Sec. 3; variants include *walk on rectangles* [Deaconu and Lejay 2006] and *walk on boundaries* [Sabelfeld and Simonov 2013]. Zagajac [1995] describes an alternative strategy based on shooting rays, albeit only for the basic Laplace equation. WoS has been applied to a limited set of problems with piecewise constant coefficients [Lejay and Maire 2013; Maire and Nguyen 2016]; ours is the first grid-free method for fairly general continuously-varying coefficients.

7.2 Volume rendering

While Monte Carlo is little-used for PDEs, it is the method of choice for the RTE in both neutron transport [Spanier and Gelbard 1969] and rendering [Novák et al. 2018; Pharr et al. 2016]. Our approach is closest to null-scattering methods, such as the *delta tracking* method detailed in Sec. 2.5 [Coleman 1968; Raab et al. 2008; Woodcock et al. 1965]. Early null-scattering methods were physically motivated, but lacked rigorous mathematical justification. More recent work re-derives null-scattering in a framework that allows construction of estimators without the need for a direct physical interpretation [Galtier et al. 2013; Georgiev et al. 2019; Jonsson et al. 2020; Kettunen et al. 2021; Kutz et al. 2017; Miller et al. 2019], and enables known properties of the medium to be exploited, e.g., using control variates [Kutz et al. 2017; Novák et al. 2014]. A clear mathematical foundation provides rich opportunities for improvement—we likewise expect the framework introduced here will make future work in volume rendering more easily transferable to methods for diffusive PDEs.

8 LIMITATIONS AND FUTURE WORK

Our method is not without limitations. As in rendering, coefficients with large spatial variation can lead to increased variance—unlike rendering, large variations in $\sigma'(x)$ (Eq. 21) typically stem from *derivatives* of diffusion and drift coefficients, making it difficult to deal with, e.g., sharp changes in material density. Adapting further techniques from volume rendering may help address such situations—e.g., local bounds on coefficient functions [Szirmay-Kalos et al. 2011; Yue et al. 2011], low-variance estimators for the VRE [Georgiev et al. 2019; Novák et al. 2018], or adaptive weight windows (as discussed in Sec. 5.3). Likewise, techniques from the SDE literature such as *multi-level Monte Carlo* [Giles 2015] may help accelerate run-time performance.

More broadly, the WoS framework still lacks support for many basic features of schemes like FEM, such as Neumann or Robin boundary conditions on general domains. Our hope is that the framework introduced here provides building blocks for extending WoS to a much larger class of PDEs, such as those with anisotropic coefficients or certain forms of nonlinearity. In the long term we are optimistic that grid-free Monte Carlo will develop into a mature technology, enabling engineers and scientists to analyze systems of extreme complexity without having to worry about meshing or discretization—just as in rendering today.

ACKNOWLEDGMENTS

This work was generously supported by nTopology and Disney Research, NSF awards 1943123, 1812796 and 1844538, a Neukom Institute CompX faculty grant, a Packard Fellowship, and an NVIDIA Graduate Fellowship.

REFERENCES

Assyr Abdulle, E Weinan, Björn Engquist, and Eric Vanden-Eijnden. 2012. The heterogeneous multiscale method. *Acta Numerica* 21 (2012), 1–87.

William Abikoff. 1981. The Uniformization Theorem. *Amer. Math. Monthly* 88, 8 (1981).

Frédéric Alauzet and Adrien Loseille. 2016. A decade of progress on anisotropic mesh adaptation for computational fluid dynamics. *Computer-Aided Design* 72 (2016).

Robert Anderson, Julian Andrej, Andrew Barker, et al. 2021. MFEM: A modular finite element methods library. *Computers & Mathematics with Applications* 81 (2021).

Adam Arbree, Bruce Walter, and Kavita Bala. 2011. Heterogeneous Subsurface Scattering Using the Finite Element Method. *IEEE TVCG* 17, 7 (July 2011), 956–969.

Ivo Babuška and Manil Suri. 1992. On locking and robustness in the finite element method. *SIAM J. Numer. Anal.* 29, 5 (1992), 1261–1293.

Victor Bayona, Natasha Flyer, and Bengt Fornberg. 2019. On the role of polynomials in RBF-FD approximations. *J. Comput. Phys.* 380 (2019), 378–399.

Victor Bayona, Natasha Flyer, Bengt Fornberg, and Gregory A Barnett. 2017. On the role of polynomials in RBF-FD approximations. *J. Comp. Phys.* 332 (2017).

Ilia Binder and Mark Braverman. 2012. The rate of convergence of the walk on spheres algorithm. *Geometric and Functional Analysis* 22, 3 (2012), 558–587.

Fischer Black and Myron Scholes. 1973. The Pricing of Options and Corporate Liabilities. *J. Pol. Econ.* 81, 3 (1973), 637–654. http://www.jstor.org/stable/1831029

Thomas E Booth. 1985. Monte Carlo variance comparison for expected-value versus sampled splitting. *Nucl. Sci. Eng.* 89, 4 (1985), 305–309.

Thomas E Booth and John S Hendricks. 1984. Importance estimation in forward Monte Carlo calculations. *Nuclear Technology-Fusion* 5, 1 (1984), 90–100.

John C. Bowers, Jonathan Leahey, and Rui Wang. 2011. A Ray Tracing Approach to Diffusion Curves. *Proc. EGSR* 30, 4 (2011), 1345–1352.

Jeremiah U Brackbill and Hans M Ruppel. 1986. FLIP: A method for adaptively zoned, particle-in-cell calculations of fluid flows. *J. Comp. Phys.* 65, 2 (1986), 314–343.

John Burgess. 2020. RTX on—the NVIDIA Turing GPU. *IEEE Micro* 40, 2 (2020), 36–44.

DM Causon and CG Mingham. 2010. *Introductory finite difference methods for PDEs.*

Subrahmanyan Chandrasekhar. 1960. *Radiative Transfer.* Dover Publications, NY.

Per H. Christensen and Wojciech Jarosz. 2016. The Path to Path-Traced Movies. *Foundations and Trends in Computer Graphics and Vision* 10, 2 (Oct. 2016), 103–175.

Michael F. Cohen and John R. Wallace. 1993. *Radiosity and Realistic Image Synthesis.*

CJ Coleman, DL Tullock, and N Phan-Thien. 1991. An effective boundary element method for inhomogeneous PDEs. *J. App. Math. Phys. (ZAMP)* 42, 5 (1991).

W. A. Coleman. 1968. Mathematical Verification of a Certain Monte Carlo Sampling Technique and Applications of the Technique to Radiation Transport Problems. *Nucl. Sci. Eng.* 32, 1 (April 1968), 76–81.

Martin Costabel. 1987. Principles of boundary element methods. *Computer Physics Reports* 6, 1-6 (1987), 243–274.

John C. Cox, Jonathan E. Ingersoll, and Stephen A. Ross. 1985. A Theory of the Term Structure of Interest Rates. *Econometrica* 53, 2 (1985), 385–407.

S. N. Cramer. 1978. Application of the Fictitious Scattering Radiation Transport Model for Deep-Penetration Monte Carlo Calculations. *Nucl. Sci. Eng.* 65, 2 (1978).

Madalina Deaconu and Antoine Lejay. 2006. A random walk on rectangles algorithm. *Methodology and Computing in Applied Probability* 8, 1 (2006), 135–151.

Eugene d'Eon and Geoffrey Irving. 2011. A Quantized-Diffusion Model for Rendering Translucent Materials. *Proc. SIGGRAPH* 30, 4 (July 2011), 56:1–56:14.

Mathieu Desbrun, Roger D Donaldson, and Houman Owhadi. 2013. Modeling Across Scales: Discrete Geometric Structures in Homogenization and Inverse Homogenization. *Multiscale Analysis and Nonlinear Dynamics* (2013), 19–64.

Craig Donner, Tim Weyrich, Eugene d'Eon, Ravi Ramamoorthi, and Szymon Rusinkiewicz. 2008. A Layered, Heterogeneous Reflectance Model for Acquiring and Rendering Human Skin. *Proc. SIGGRAPH Asia* 27, 5 (2008), 140:1–140:12.

Dean G Duffy. 2015. *Green's functions with applications.* Chapman and Hall/CRC.

Louis J Durlofsky. 1991. Numerical calculation of equivalent grid block permeability tensors for heterogeneous porous media. *Water resources research* 27, 5 (1991).

Bruce B Dykaar and Peter K Kitanidis. 1992. Determination of the Effective Hydraulic Conductivity for Heterogeneous Porous Media. *Water Res. R.* 28, 4 (1992).

Yalchin Efendiev and Thomas Y Hou. 2009. *Multiscale finite element methods: theory and applications.* Vol. 4.

Christer Ericson. 2004. *Real-time collision detection.* Crc Press.

Lawrence C Evans. 1998. *Partial differential equations.* Vol. 19. Rhode Island, USA.

Florian Fahrenberger, Zhenli Xu, and Christian Holm. 2014. Simulation of electric double layers around charged colloids in aqueous solution of variable permittivity. *J. Chem. Phys.* 141, 6 (2014).

J. A Fleck and E. H Canfield. 1984. A Random Walk Procedure for Improving the Computational Efficiency of the Implicit Monte Carlo Method for Nonlinear Radiation Transport. *J. Comput. Phys.* 54, 3 (June 1984), 508–523.

Natasha Flyer, Bengt Fornberg, Victor Bayona, and Gregory A Barnett. 2016. On the role of polynomials in RBF-FD approximations: I. Interpolation and accuracy. *J. Comput. Phys.* 321 (2016), 21–38.

Julian Fong, Magnus Wrenninge, Christopher Kulla, and Ralf Habel. 2017. Production Volume Rendering. In *ACM SIGGRAPH Courses.* ACM Press, New York, NY, USA.

A. Friedman and K.S. Fu. 1975. *Stochastic Differential Equations and Applications.*

Thomas-Peter Fries, Hermann Matthies, et al. 2004. Classification and overview of meshfree methods. (2004).

M. Galtier, S. Blanco, C. Caliot, C. Coustet, J. Dauchet, M. El Hafi, V. Eymet, R. Fournier, J. Gautrais, A. Khuong, B. Piaud, and G. Terrée. 2013. Integral Formulation of Null-Collision Monte Carlo Algorithms. 125 (Aug. 2013), 57–68.

Riccardo Gatto. 2013. The von Mises–Fisher distribution of the first exit point from the hypersphere of the drifted Brownian motion and the density of the first exit time. *Statistics & Probability Letters* 83, 7 (2013), 1669–1676.

Iliyan Georgiev, Zackary Misso, Toshiya Hachisuka, Derek Nowrouzezahrai, Jaroslav Křivánek, and Wojciech Jarosz. 2019. Integral Formulations of Volumetric Transmittance. *Proc. SIGGRAPH Asia* 38, 6 (Nov. 2019), 154:1–154:17.

Frederic Gibou, Ronald Fedkiw, and Stanley Osher. 2018. A review of level-set methods and some recent applications. *J. Comput. Phys.* 353 (2018), 82–109.

Michael B Giles. 2015. Multilevel Monte Carlo methods. *Acta Numer.* 24 (2015), 259–328.

Daniel T Gillespie. 1977. Exact stochastic simulation of coupled chemical reactions. *The journal of physical chemistry* 81, 25 (1977), 2340–2361.

Denis S Grebenkov. 2007. NMR survey of reflected Brownian motion. *Reviews of Modern Physics* 79, 3 (2007), 1077.

Ralf Habel, Per H. Christensen, and Wojciech Jarosz. 2013. Photon Beam Diffusion: A Hybrid Monte Carlo Method for Subsurface Scattering. *Proc. EGSR* 32, 4 (2013).

Wolfgang Hackbusch. 2015. *Hierarchical matrices: algorithms and analysis.* Vol. 49.

Francis H Harlow and J Eddie Welch. 1965. Numerical calculation of time-dependent viscous incompressible flow of fluid with free surface. *Phys. Fluids* 8, 12 (1965).

John C. Hart. 1996. Sphere Tracing: A Geometric Method for the Antialiased Ray Tracing of Implicit Surfaces. *The Visual Computer* 12, 10 (Dec. 1996), 527–545.

Desmond J Higham. 2001. An algorithmic introduction to numerical simulation of stochastic differential equations. *SIAM review* 43, 3 (2001), 525–546.

J Eduard Hoogenboom and Dávid Légrády. 2005. A critical review of the weight window generator in MCNP. In *Proceedings of Monte Carlo Topical Meeting*. 17–21.

Yixin Hu, Teseo Schneider, Bolun Wang, Denis Zorin, and Daniele Panozzo. 2020. Fast Tetrahedral Meshing in the Wild. *ACM Trans. Graph.* 39, 4 (July 2020).

Chi-Ok Hwang, Sungpyo Hong, and Jinwoo Kim. 2015. Off-centered Walk-on-Spheres algorithm. *J. Comp. Phys.* 303 (2015), 331–335.

Intel. 2013. Embree: High Performance Ray Tracing Kernels. http://embree.github.io/

Henrik Wann Jensen. 2001. State of the Art in Monte Carlo Ray Tracing for Realistic Image Synthesis. In *SIGGRAPH Course Notes*.

Henrik W. Jensen, Frank Suykens, and Per H. Christensen. 2001. A Practical Guide to Global Illumination Using Photon Mapping. In *ACM SIGGRAPH Courses*.

Chenfanfu Jiang, Craig Schroeder, Andrew Selle, Joseph Teran, and Alexey Stomakhin. 2015. The affine particle-in-cell method. *ACM TOG* 34, 4 (2015), 1–10.

Chenfanfu Jiang, Craig Schroeder, Joseph Teran, Alexey Stomakhin, and Andrew Selle. 2016. The material point method for simulating continuum materials. In *ACM SIGGRAPH 2016 Courses*. 1–52.

Daniel Jonsson, Joel Kronander, Jonas Unger, Thomas B. Schon, and Magnus Wrenninge. 2020. Direct Transmittance Estimation in Heterogeneous Participating Media Using Approximated Taylor Expansions. *IEEE TVCG* (2020), 1–1.

James T. Kajiya. 1986. The Rendering Equation. *Proc. SIGGRAPH* 20, 4 (Aug. 1986).

S. Kakutani. 1944. Two-dimensional Brownian Motion and Harmonic Functions. *Proceedings of the Imperial Academy* 20, 10 (1944), 706–714.

Rudolph Emil Kalman. 1960. A New Approach to Linear Filtering and Prediction Problems. *Trans. ASME–Journal of Basic Engineering* 82, Series D (1960), 35–45.

Hilbert J Kappen. 2007. An introduction to stochastic control theory, path integrals and reinforcement learning. In *AIP conference proceedings*, Vol. 887. Amer. Inst. Phys.

Markus Kettunen, Eugene d'Eon, Jacopo Pantaleoni, and Jan Novak. 2021. An Unbiased Ray-Marching Transmittance Estimator. (Feb. 2021). arXiv:2102.10294 [cs.GR]

P. Kloeden and E. Platen. 2013. *Num. Sol. of Stoc. Diff. Eq.* Vol. 23.

David Koerner, Jamie Portsmouth, Filip Sadlo, Thomas Ertl, and Bernd Eberhardt. 2014. Flux-Limited Diffusion for Multiple Scattering in Participating Media. *CGF* 33, 6 (Sept. 2014), 178–189.

Bastian Krayer and Stefan Müller. 2021. Hierarchical Point Distance Fields. In *International Symposium on Visual Computing*. Springer, 435–446.

Peter Kutz, Ralf Habel, Yining Karl Li, and Jan Novák. 2017. Spectral and Decomposition Tracking for Rendering Heterogeneous Volumes. *Proc. SIGGRAPH* 36, 4 (July 2017).

Eric P. Lafortune and Yves D. Willems. 1996. Rendering Participating Media with Bidirectional Path Tracing. In *Proc. EGWR*. Springer-Verlag, Vienna, 91–100.

Antoine Lejay and Sylvain Maire. 2013. New Monte Carlo schemes for simulating diffusions in discontinuous media. *J. Comp. and Appl. Mathematics* 245 (2013).

Shaofan Li and Wing Kam Liu. 2007. *Meshfree particle methods.*

Frank Losasso, Ronald Fedkiw, and Stanley Osher. 2006. Spatially adaptive techniques for level set methods and incompressible flow. *Computers & Fluids* 35, 10 (2006).

Sylvain Maire and Giang Nguyen. 2016. Stochastic finite differences for elliptic diffusion equations in stratified domains. *Mathematics & Comp. in Simulation* 121 (2016).

Nathan G March, Elliot J Carr, and Ian W Turner. 2021. Numerical investigation into coarse-scale models of diffusion in complex heterogeneous media. *Transport in Porous Media* 139, 3 (2021), 467–489.

Zoë Marschner, Paul Zhang, David Palmer, and Justin Solomon. 2021. Sum-of-squares geometry processing. *ACM Transactions on Graphics (TOG)* 40, 6 (2021), 1–13.

Michael Mascagni and Nikolai A Simonov. 2004a. Monte Carlo methods for calculating some physical properties of large molecules. *SIAM J. sc. comp.* 26, 1 (2004), 339–357.

Michael Mascagni and Nikolai A Simonov. 2004b. The random walk on the boundary method for calculating capacitance. *J. Comput. Phys.* 195, 2 (2004), 465–473.

Khamron Mekchay and Ricardo H Nochetto. 2005. Convergence of adaptive finite element methods for general second order linear elliptic PDEs. *SIAM J. Numer. Anal.* 43, 5 (2005), 1803–1827.

Robert C Merton. 1971. Optimum consumption and portfolio rules in a continuous-time model. In *Stochastic optimization models in finance*. Elsevier, 621–661.

Robert C Merton and Paul Anthony Samuelson. 1992. Continuous-time finance. (1992).

Bailey Miller, Iliyan Georgiev, and Wojciech Jarosz. 2019. A Null-Scattering Path Integral Formulation of Light Transport. *Proc. SIGGRAPH* 38, 4 (July 2019), 44:1–44:13.

Jonathan T. Moon, Bruce Walter, and Steve Marschner. 2008. Efficient Multiple Scattering in Hair Using Spherical Harmonics. *Proc. SIGGRAPH* 27, 3 (Aug. 2008).

Linus Mossberg. 2021. *GPU-Accelerated Monte Carlo Geometry Processing for Gradient-Domain Methods.* Ph. D. Dissertation. Linköping University, Linköping, Sweden.

Mervin E. Muller. 1956. Some Continuous Monte Carlo Methods for the Dirichlet Problem. *Annals of Mathematical Statistics* 27, 3 (Sept. 1956), 569–589.

Thomas Müller, Marios Papas, Markus Gross, Wojciech Jarosz, and Jan Novák. 2016. Efficient Rendering of Heterogeneous Polydisperse Granular Media. *Proc. SIGGRAPH Asia* 35, 6 (Nov. 2016), 168:1–168:14.

Ken Museth. 2013. VDB: High-Resolution Sparse Volumes with Dynamic Topology. *ACM TOG* 32, 3 (July 2013), 27:1–27:22.

Mohammad Sina Nabizadeh, Ravi Ramamoorthi, and Albert Chern. 2021. Kelvin Transformations for Simulations on Infinite Domains. (July 2021).

Vinh Phu Nguyen, Timon Rabczuk, Stéphane Bordas, and Marc Duflot. 2008. Meshless methods: a review and computer implementation aspects. *Mathematics and computers in simulation* 79, 3 (2008), 763–813.

Jan Novák, Iliyan Georgiev, Johannes Hanika, and Wojciech Jarosz. 2018. Monte Carlo Methods for Volumetric Light Transport Simulation. *Computer Graphics Forum (Proc. Eurographics State of the Art Reports)* 37, 2 (May 2018), 551–576.

Jan Novák, Andrew Selle, and Wojciech Jarosz. 2014. Residual Ratio Tracking for Estimating Attenuation in Participating Media. *ACM TOG* 33, 6 (Nov. 2014).

Bernt Øksendal. 2003. *Stochastic Differential Equations: An Introduction with Applications.*

Alexandrina Orzan, Adrien Bousseau, Holger Winnemöller, Pascal Barla, Joëlle Thollot, and David Salesin. 2008. Diffusion Curves: A Vector Representation for Smooth-Shaded Images. *Proc. SIGGRAPH* 27, 3 (Aug. 2008), 1.

Ozgur Ozdemir. 2005. Variable permittivity dielectric material loaded stepped-horn antenna. (2005).

Paul William Partridge, Carlos Alberto Brebbia, et al. 2012. *Dual reciprocity boundary element method.*

Mark Pauly, Richard Keiser, Bart Adams, Philip Dutré, Markus Gross, and Leonidas J Guibas. 2005. Meshless animation of fracturing solids. *ACM TOG* 24, 3 (2005).

Matt Pharr, Wenzel Jakob, and Greg Humphreys. 2016. *Physically Based Rendering: From Theory to Implementation* (3rd ed.). Morgan Kaufmann, Cambridge, MA.

Romain Prévost, Wojciech Jarosz, and Olga Sorkine-Hornung. 2015. A Vectorial Framework for Ray Traced Diffusion Curves. *CGF* 34, 1 (Feb. 2015), 253–264.

Inigo Quilez. 2020. Monte Carlo PDE Shader. https://www.shadertoy.com/view/WdXfzl.

Matthias Raab, Daniel Seibert, and Alexander Keller. 2008. Unbiased Global Illumination with Participating Media. In *Monte Carlo and Quasi-Monte Carlo Methods*, Alexander Keller, Stefan Heinrich, and Harald Niederreiter (Eds.). Springer-Verlag, 591–605.

Karl K Sabelfeld. 2018. Application of the von Mises–Fisher distribution to Random Walk on Spheres method for solving high-dimensional diffusion–advection–reaction equations. *Statistics & Probability Letters* 138 (2018), 137–142.

Karl K Sabelfeld and Nikolai A Simonov. 2013. *Random walks on boundary for solving PDEs.* De Gruyter.

Rohan Sawhney and Keenan Crane. 2020. Monte Carlo Geometry Processing: A Grid-Free Approach to PDE-Based Methods on Volumetric Domains. *Proc. SIGGRAPH* 39, 4 (2020).

Rohan Sawhney, Ruihao Ye, Johann Korndoerfer, and Keenan Crane. 2020. FCPW: Fastest Closest Points in the West. https://github.com/rohan-sawhney/fcpw.

Teseo Schneider, Yixin Hu, Jérémie Dumas, Xifeng Gao, Daniele Panozzo, and Denis Zorin. 2018. Decoupling simulation accuracy from mesh quality. *ACM TOG* (2018).

Vadim Shapiro and Igor Tsukanov. 1999. Meshfree simulation of deforming domains. *Computer-Aided Design* 31, 7 (1999), 459–471.

Prateek Sharma and Gregory W Hammett. 2007. Preserving monotonicity in anisotropic diffusion. *J. Comput. Phys.* 227, 1 (2007), 123–142.

Jure Slak and Gregor Kosec. 2019. On generation of node distributions for meshless PDE discretizations. *SIAM Journal on Scientific Computing* 41, 5 (2019), A3202–A3229.

Jerome Spanier and Ely Meyer Gelbard. 1969. *Monte Carlo Principles and Neutron Transport Problems.* Addison-Wesley.

Deborah Sulsky, Shi-Jian Zhou, and Howard L Schreyer. 1995. Application of a particle-in-cell method to solid mechanics. *Comp. phys. comms.* 87, 1-2 (1995), 236–252.

Xin Sun, Guofu Xie, Yue Dong, Stephen Lin, Weiwei Xu, Wencheng Wang, Xin Tong, and Baining Guo. 2012. Diffusion Curve Textures for Resolution Independent Texture Mapping. *Proc. SIGGRAPH* 31, 4 (July 2012), 74:1–74:9.

László Szirmay-Kalos, Balázs Tóth, and Milán Magdics. 2011. Free Path Sampling in High Resolution Inhomogeneous Participating Media. *CGF* 30, 1 (2011), 85–97.

Heang K Tuy and Lee Tan Tuy. 1984. Direct 2-D display of 3-D objects. *IEEE Computer Graphics and Applications* 4, 10 (1984), 29–34.

MV Umansky, MS Day, and TD Rognlien. 2005. On numerical solution of strongly anisotropic diffusion equation on misaligned grids. *Numerical Heat Transfer, Part B: Fundamentals* 47, 6 (2005), 533–554.

Eric Veach. 1997. *Robust Monte Carlo Methods for Light Transport Simulation.* Ph.D. Thesis. Stanford University.

Eric Veach and Leonidas J. Guibas. 1995. Optimally Combining Sampling Techniques for Monte Carlo Rendering. In *Proc. SIGGRAPH*, Vol. 29. ACM Press, 419–428.

Jiří Vorba and Jaroslav Křivánek. 2016. Adjoint-Driven Russian Roulette and Splitting in Light Transport Simulation. *Proc. SIGGRAPH* 35, 4 (July 2016), 42:1–42:11.

John C Wagner and Alireza Haghighat. 1998. Automated variance reduction of Monte Carlo shielding calculations using the discrete ordinates adjoint function. *Nucl. Sci. Eng.* 128, 2 (1998).

Greg Ward and Rob Shakespeare. 1998. Rendering with Radiance: the art and science of lighting visualization. (1998).

Mathias Willmann, Jesus Carrera, Xavier Sanchez-Vila, O Silva, and Marco Dentz. 2010. Coupling of mass transfer and reactive transport for nonlinear reactions in heterogeneous media. *Water resources research* 46, 7 (2010).

E. R. Woodcock, T. Murphy, P. J. Hemmings, and T. C. Longworth. 1965. Techniques Used in the GEM Code for Monte Carlo Neutronics Calculations in Reactors and Other Systems of Complex Geometry. In *Applications of Computing Methods to Reactor Problems.* Argonne National Laboratory.

Yonghao Yue, Kei Iwasaki, Bing-Yu Chen, Yoshinori Dobashi, and Tomoyuki Nishita. 2011. Toward Optimal Space Partitioning for Unbiased, Adaptive Free Path Sampling of Inhomogeneous Participating Media. *CGF* 30, 7 (2011), 1911–1919.

Jovan Zagajac. 1995. A fast method for estimating discrete field values in early engineering design. In *Proc. the ACM symp. on Solid modeling and applications.* 420–430.

Laurent Zalewski, Stéphane Lassue, Daniel Rousse, and Kamel Boukhalfa. 2010. Experimental and numerical characterization of thermal bridges in prefabricated building walls. *Energy Conversion and Management* 51, 12 (2010), 2869–2877.

Qingnan Zhou and Alec Jacobson. 2016. Thingi10K: A Dataset of 10,000 3D-Printing Models. *arXiv preprint arXiv:1605.04797* (2016).

Yongning Zhu and Robert Bridson. 2005. Animating sand as a fluid. *ACM Transactions on Graphics (TOG)* 24, 3 (2005), 965–972.

Olgierd Cecil Zienkiewicz and Jian Zhong Zhu. 1992a. The superconvergent patch recovery and a posteriori error estimates. Part 1: The recovery technique. *Internat. J. Numer. Methods Engrg.* 33, 7 (1992), 1331–1364.

Olgierd Cecil Zienkiewicz and Jian Zhong Zhu. 1992b. The superconvergent patch recovery and a posteriori error estimates. Part 2: Error estimates and adaptivity. *Internat. J. Numer. Methods Engrg.* 33, 7 (1992), 1365–1382.

A REMOVING THE FIRST ORDER TERM

Sec. 4 transforms a 2nd order PDE with variable drift into an equivalent PDE without a 1st order term (Eqs. 20 and 21). Here we give the details of this transformation; note that App. A.1 assumes $\alpha := 1$, since a variable diffusion coefficient $\alpha(x)$ can be absorbed into the source function $f(x)$ and screening coefficient $\sigma(x)$, as in Eq. 20.

A.1 Girsanov Transformation

In stochastic calculus, a *Girsanov transformation* describes how the dynamics of a continuous random process transform under a change of probability measure. Let X_t be the diffusion process governed by the SDE $\mathrm{d}X_t = \vec{\omega}(X_t)\,\mathrm{d}t + \mathrm{d}W_t$. The solution to the PDE

$$\begin{aligned} \frac{1}{2}\Delta u(x) + \vec{\omega}(x)\cdot\nabla u(x) - \sigma(x)u(x) &= -f(x) \text{ on } \Omega, \\ u(x) &= g(x) \text{ on } \partial\Omega. \end{aligned} \tag{32}$$

can then be expressed via the expectation

$$u(x) = \mathbb{E}\left[\int_0^\tau \mathrm{e}^{-\int_0^t \sigma(X_s)\,\mathrm{d}s}\, f(X_t)\,\mathrm{d}t \;+ \mathrm{e}^{-\int_0^\tau \sigma(X_t)\,\mathrm{d}t} g(X_\tau)\right] \tag{33}$$

over random trajectories of X_t, conditioned on $X_0 = x$. The Girsanov transformation modifies the drift coefficient $\vec{\omega}(x)$ of X_t by introducing an *importance sampling weight*

$$Z(W_t) := \mathrm{e}^{\int_0^t \vec{\omega}(W_s)\cdot\mathrm{d}W_s \,-\, \frac{1}{2}\int_0^t |\vec{\omega}(W_s)|^2\,\mathrm{d}s}. \tag{34}$$

into Eq. 33. The term $\int_0^t \vec{\omega}(W_s)\cdot\mathrm{d}W_s$ is called a *stochastic integral* since it is defined with respect to variations of a Brownian process W_s (App. A.2 describes how to evaluate this integral; see Øksendal [2003] for a formal definition). Reweighting by Z yields an *equivalent* stochastic formula involving only a Brownian random walk W_t:

$$\begin{aligned} u(x) = \mathbb{E}\Bigg[\int_0^\tau \mathrm{e}^{-\int_0^t \sigma(W_s)\,\mathrm{d}s}\, Z(W_t)\, f(W_t)\,\mathrm{d}t \,+ \\ \mathrm{e}^{-\int_0^\tau \sigma(W_t)\,\mathrm{d}t}\, Z(W_\tau)\, g(W_\tau)\,\Bigg|\, W_0 = x\Bigg]. \end{aligned} \tag{35}$$

A.2 The Chain Rule of Stochastic Calculus

Itô's lemma is the stochastic counterpart of the chain rule: given a twice differential function $\gamma(x) : \mathbb{R}^n \mapsto \mathbb{R}$, it says that the differential $\mathrm{d}\gamma$, as a function of a Brownian process W_s, is given by

$$\mathrm{d}\gamma(W_s) = \nabla\gamma(W_s)\cdot\mathrm{d}W_s \;+\; \frac{1}{2}\Delta\gamma(W_s)\,\mathrm{d}s. \tag{36}$$

Integrating over time and rearranging the terms then yields:

$$\int_0^t \nabla\gamma(W_s)\cdot\mathrm{d}W_s = \gamma(W_t) - \gamma(W_0) - \int_0^t \tfrac{1}{2}\Delta\gamma(W_s)\,\mathrm{d}s. \tag{37}$$

This integrated version of Itô's lemma allows us to re-express the importance weight Z without a stochastic integral. In particular, if the drift coefficient takes the form $\vec{\omega}(x) = \nabla\gamma(x)$ for some scalar field γ, then an alternative expression for Z is given by:

$$Z(W_t) = \mathrm{e}^{\gamma(W_t) \,-\, \gamma(W_0) \,-\, \frac{1}{2}\int_0^t (\Delta\gamma(W_s) \,+\, |\nabla\gamma(W_s)|^2)\,\mathrm{d}s}. \tag{38}$$

A.3 Derivation of Eq. 21

With this new expression for Z, Eq. 35 now takes the form

$$\begin{aligned} u(x) = \mathrm{e}^{-\gamma(x)}\,\mathbb{E}\Bigg[\int_0^\tau \mathrm{e}^{-\int_0^t \sigma'(W_s)\,\mathrm{d}s}\, f'(W_t)\,\mathrm{d}t \,+ \\ \mathrm{e}^{-\int_0^\tau \sigma'(W_t)\,\mathrm{d}t} g'(W_\tau)\,\Bigg|\, W_0 = x\Bigg], \end{aligned} \tag{39}$$

where

$$\begin{gathered} f'(x) := \mathrm{e}^{\gamma(x)} f(x), \qquad g'(x) := \mathrm{e}^{\gamma(x)} g(x), \\ \sigma'(x) := \sigma(x) + \frac{1}{2}\left(\Delta\gamma(x) + |\nabla\gamma(x)|^2\right). \end{gathered}$$

Setting $U(x) := \mathrm{e}^{\gamma(x)}u(x)$, the PDE corresponding to Eq. 39 no longer has a 1st order drift term:

$$\begin{aligned} \frac{1}{2}\Delta U(x) - \sigma'(x)U(x) &= -f'(x) \text{ on } \Omega, \\ U(x) &= g'(x) \text{ on } \partial\Omega, \end{aligned} \tag{40}$$

Note in particular that we obtain Eq. 21 from Eq. 20 by letting $\gamma(x) = \frac{1}{2}\ln(\alpha(x))$.

Received January 2022