GRID-FREE MONTE CARLO FOR PDES WITH SPATIALLY VARYING COEFFICIENTS

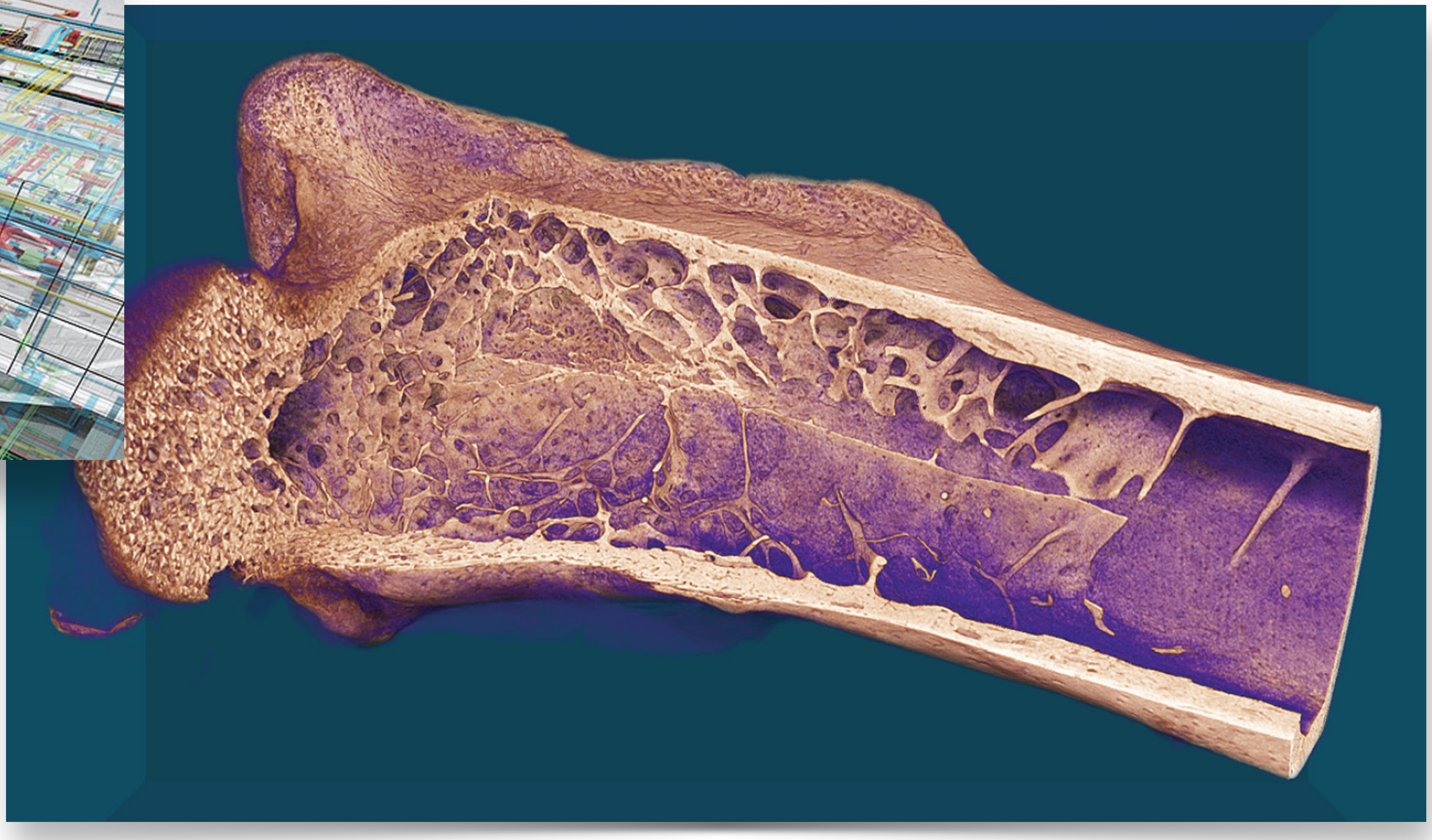
ROHAN SAWHNEY*, CARNEGIE MELLON UNIVERSITY
DARIO SEYB*, DARTMOUTH COLLEGE
WOJCIECH JAROSZ†, DARTMOUTH COLLEGE
KEENAN CRANE†, CARNEGIE MELLON UNIVERSITY

The symbols * and † indicate equal contribution.

Geometric & material complexity in science & engineering

building information model

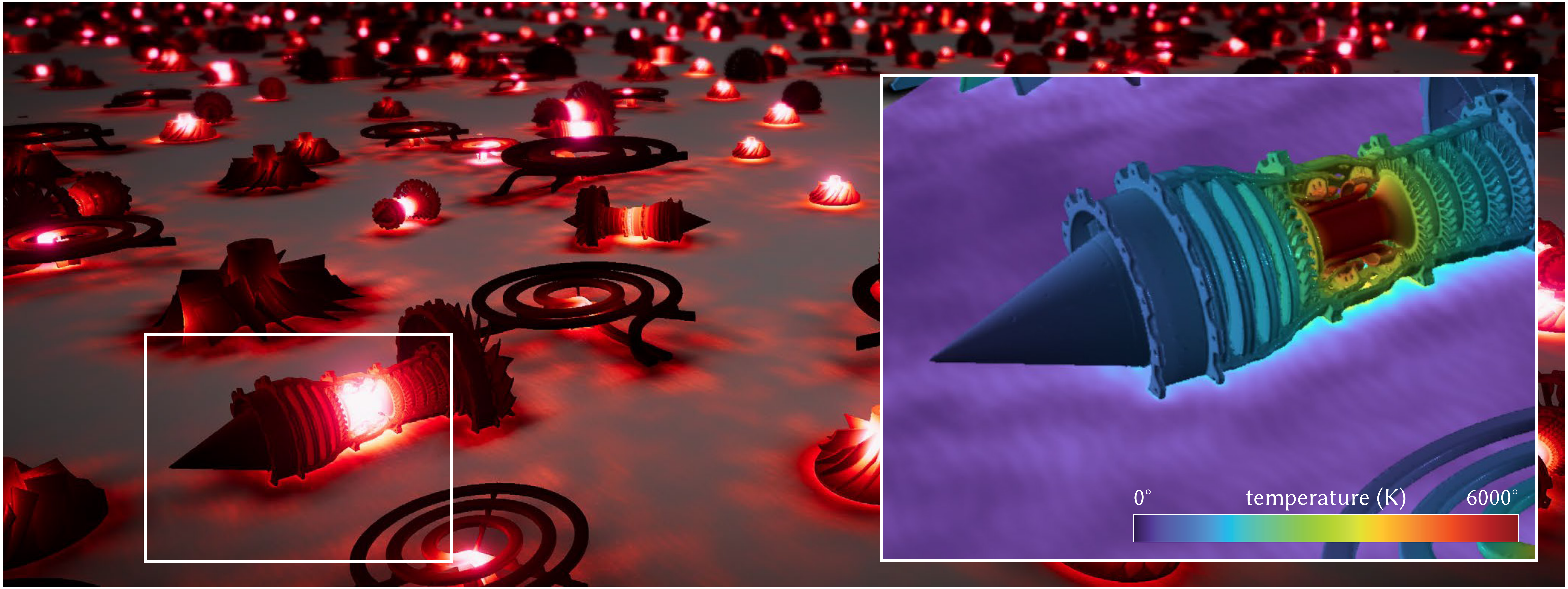
microCT scan



Photorealistic rendering of complex geometry & materials

geometry + materials ⇨ **rendered output**

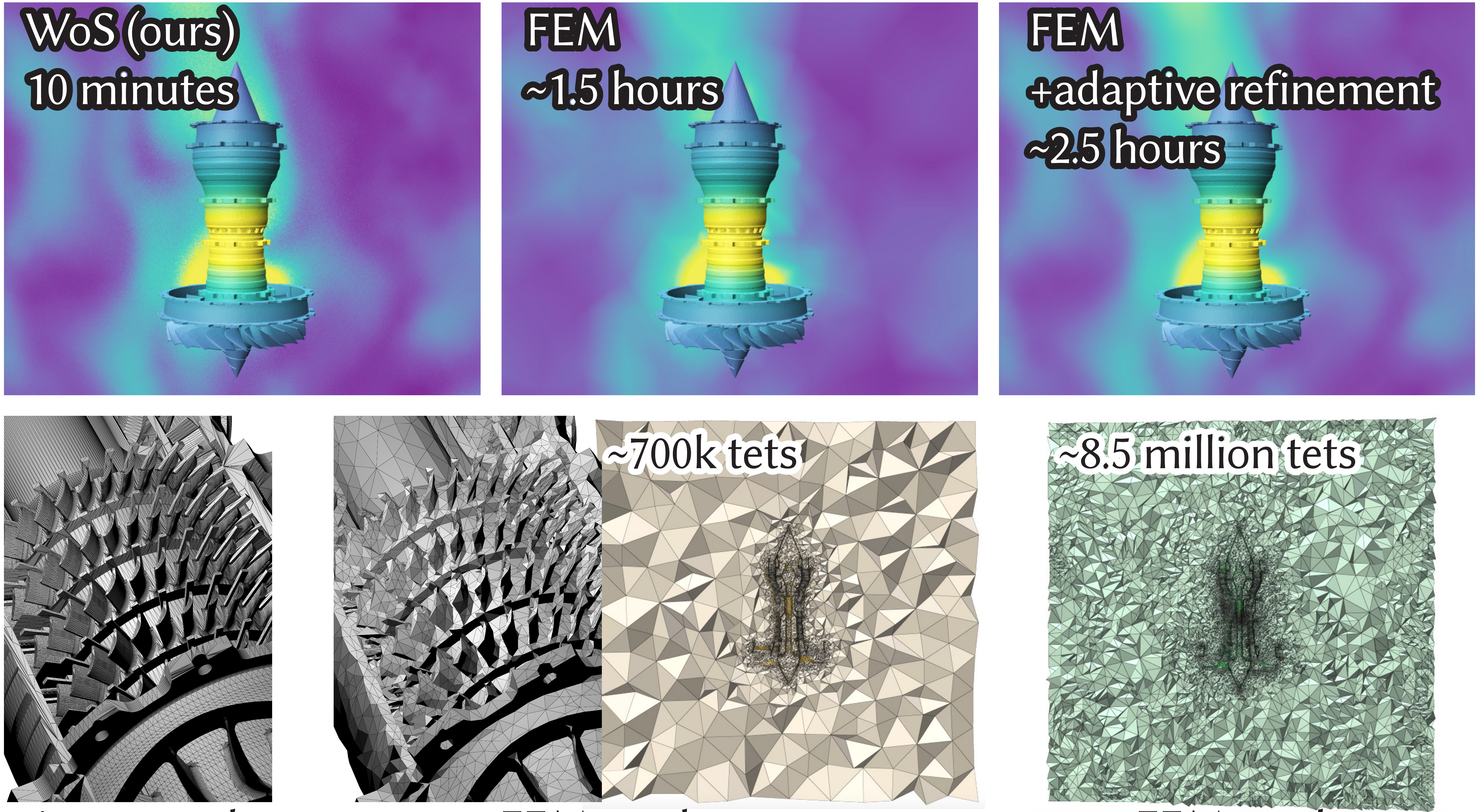
Physical analysis of complex geometry & materials



Heat radiating from infinitely many blackbodies in a heterogenous medium

About 600 million effective vertices from visible viewpoint

Challenge with conventional PDE solvers: scalability

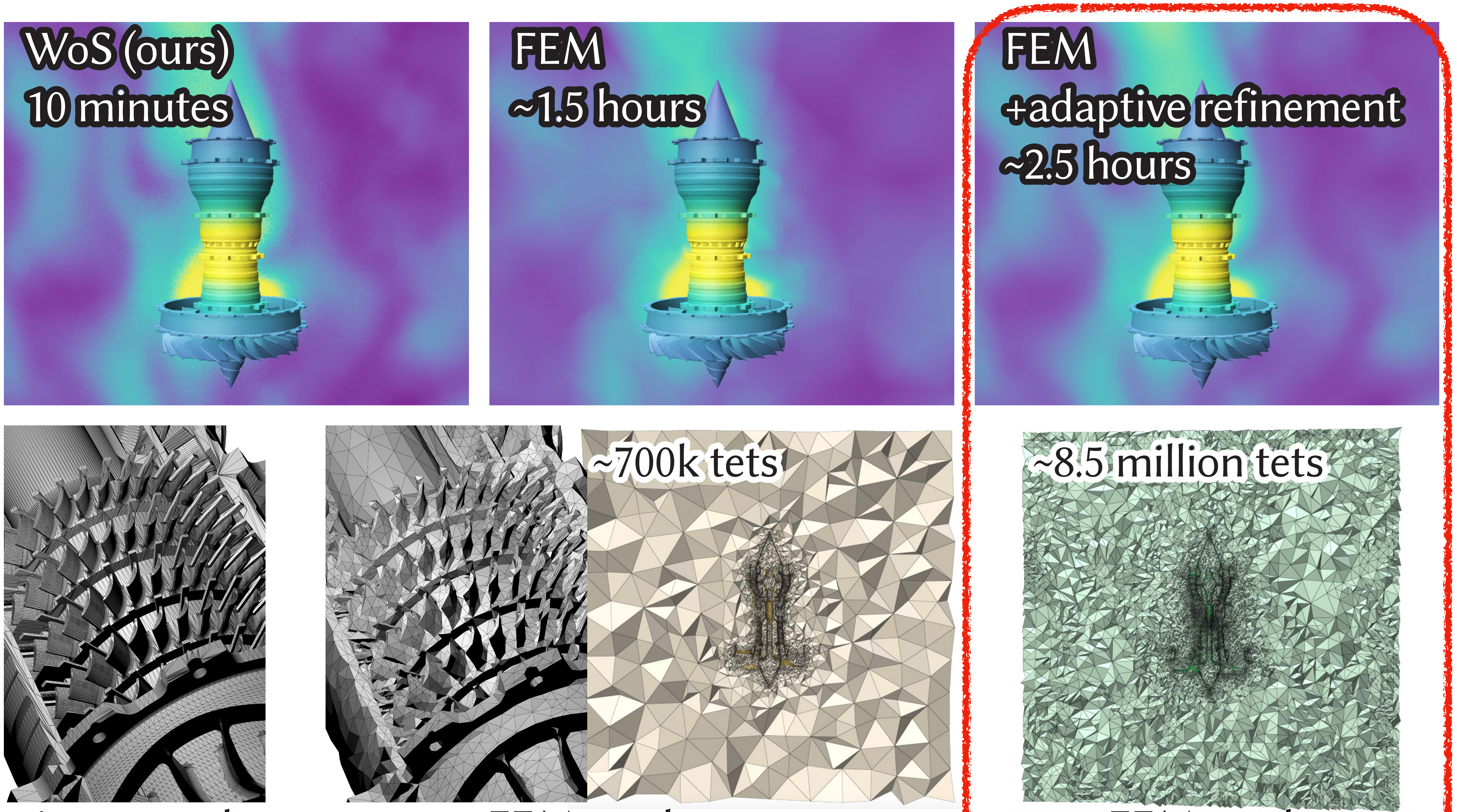


input mesh
(used by WoS)

FEM mesh
(FastTetWild)

FEM mesh
(FastTetWild + AMR)

Challenge with conventional PDE solvers: scalability

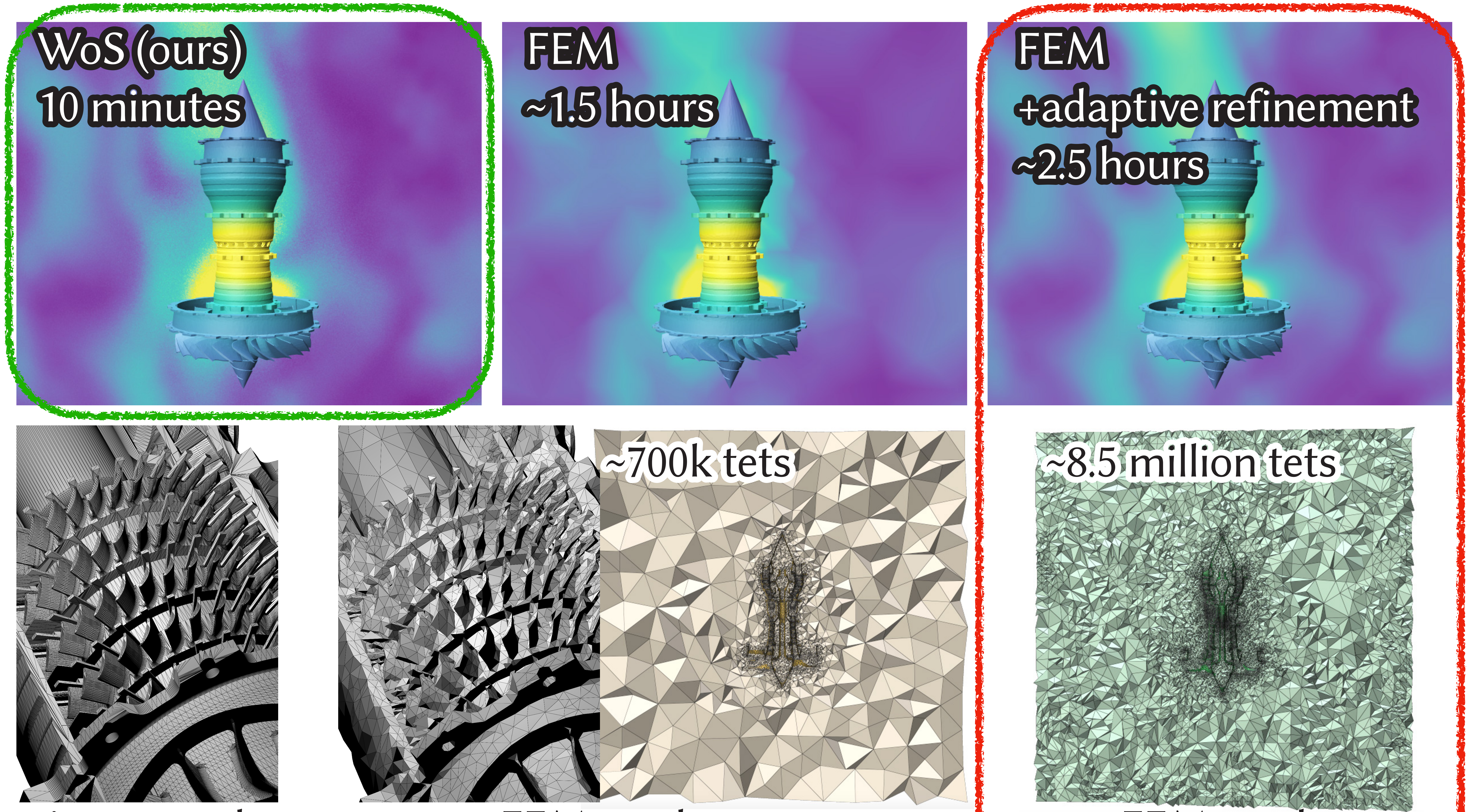


input mesh
(used by WoS)

FEM mesh
(FastTetWild)

FEM mesh
(FastTetWild + AMR)

Challenge with conventional PDE solvers: scalability



input mesh
(used by WoS)

FEM mesh
(FastTetWild)

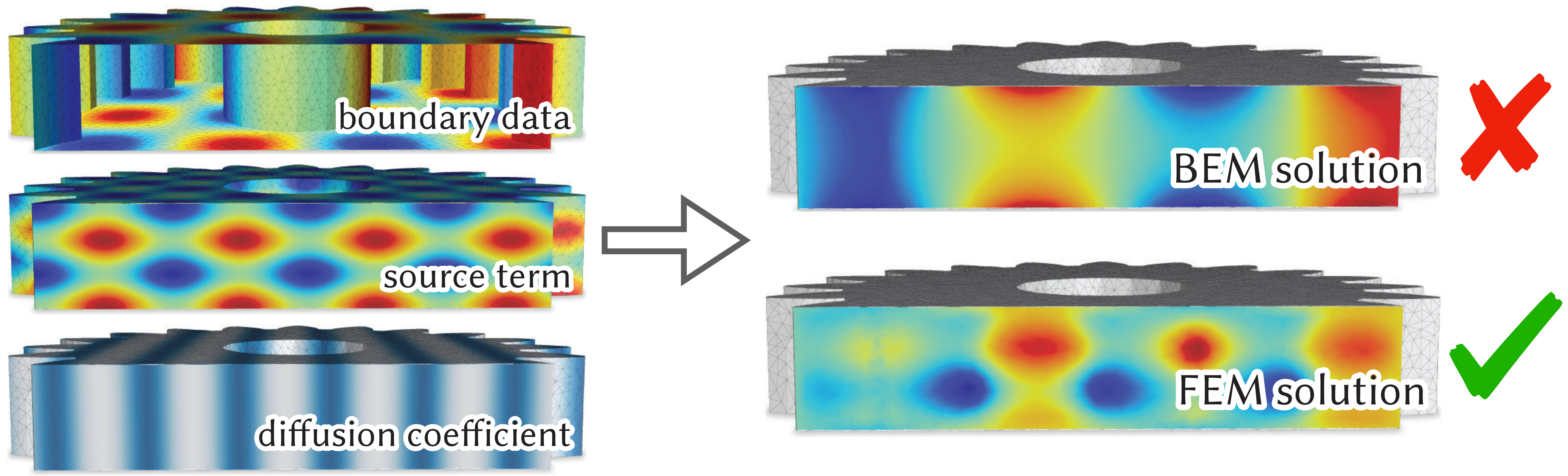
FEM mesh
(FastTetWild + AMR)

Challenge with conventional “mesh free” PDE solvers

Boundary Element Method (BEM) and Meshless Finite Element (MFEM)

Challenge with conventional "mesh free" PDE solvers

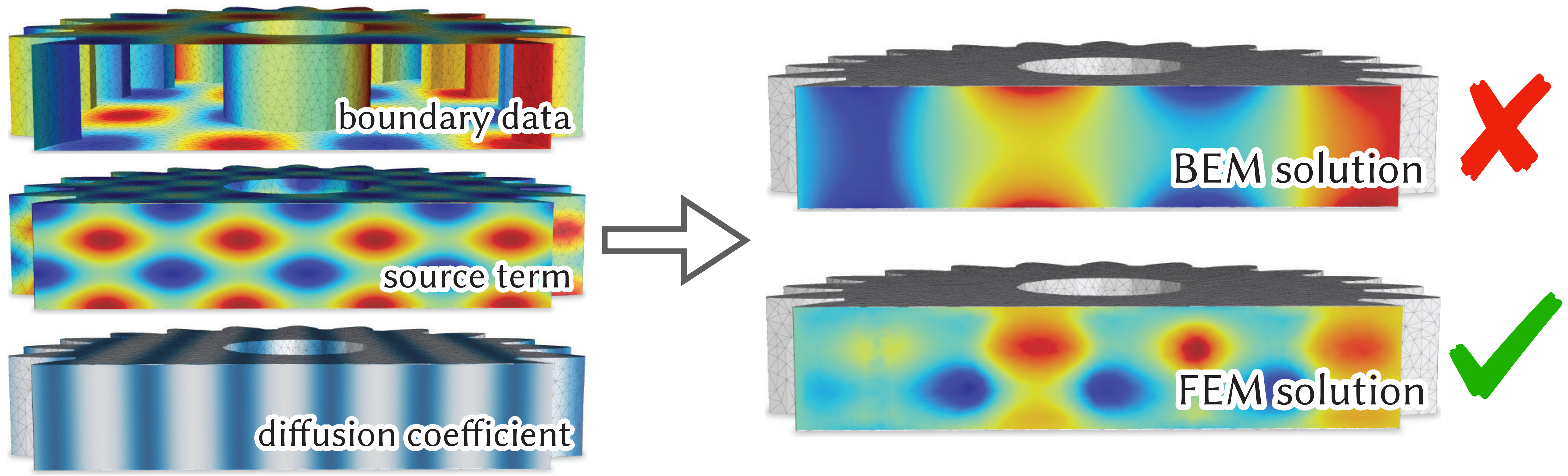
Boundary Element Method (BEM) and Meshless Finite Element (MFEM)



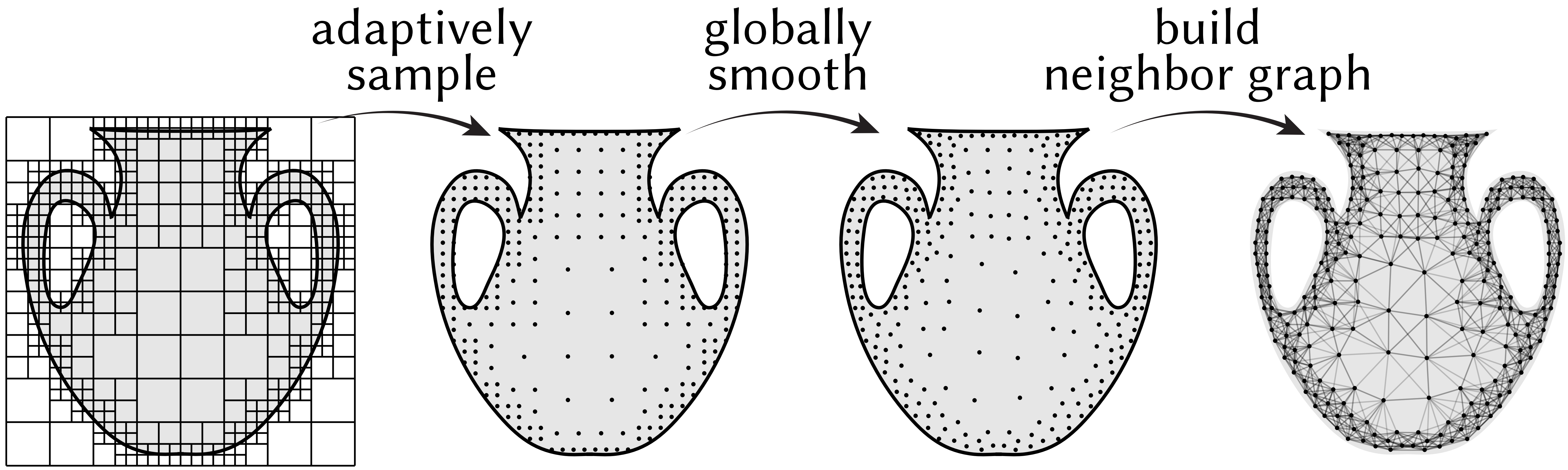
BEM: No support for spatially-varying coefficients

Challenge with conventional "mesh free" PDE solvers

Boundary Element Method (BEM) and Meshless Finite Element (MFEM)



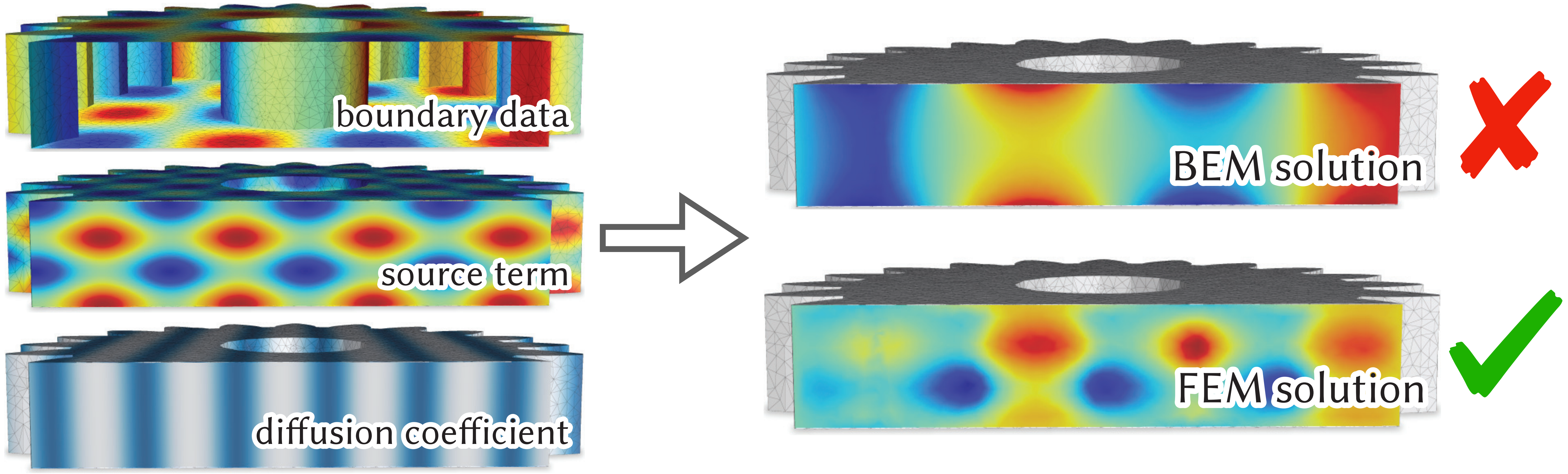
BEM: No support for spatially-varying coefficients



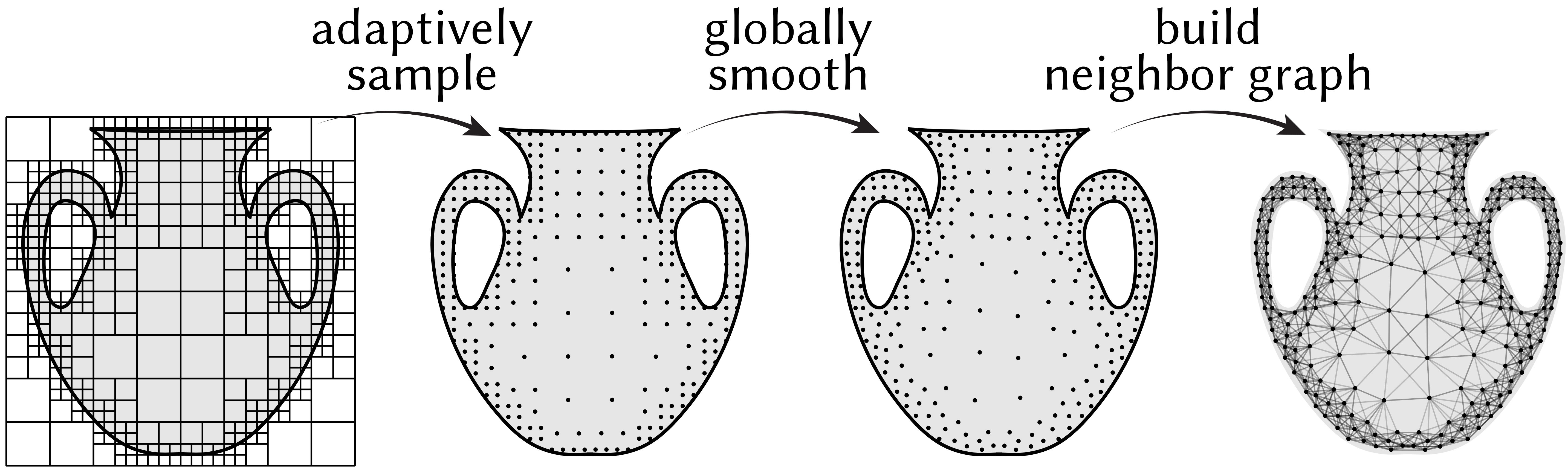
MFEM: Careful node placement & connectivity

Challenge with conventional "mesh free" PDE solvers

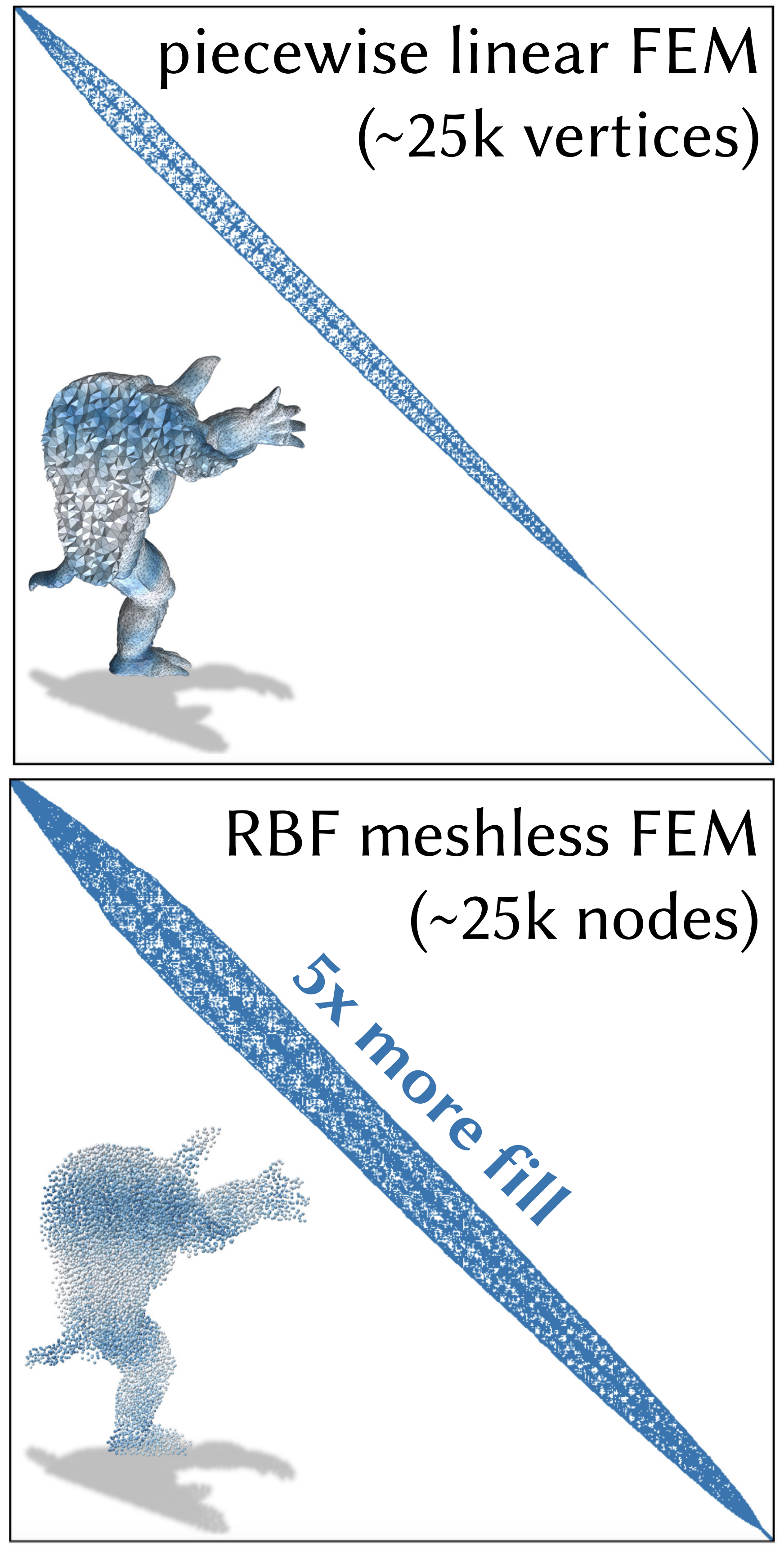
Boundary Element Method (BEM) and Meshless Finite Element (MFEM)



BEM: No support for spatially-varying coefficients



MFEM: Careful node placement & connectivity



MFEM & BEM: Global system solves

Monte Carlo Geometry Processing [SIGGRAPH 2020]

Monte Carlo Geometry Processing:
A Grid-Free Approach to PDE-Based Methods on Volumetric Domains

ROHAN SAWHNEY and KEENAN CRANE, Carnegie Mellon University

Fig. 1. Real-world geometry has not only rich surface detail *(left)* but also intricate internal structure *(center)*. On such domains, FEM-based geometric algorithms struggle to mesh, setup, and solve PDEs—in this case taking more than 14 hours and 30GB of memory just for a basic Poisson equation. Our Monte Carlo solver uses about 1GB of memory and takes less than a minute to provide a preview *(center right)* that can then be progressively refined *(far right)*. *[Boundary mesh of Fijian strumigenys FJ13 used courtesy of the Economo Lab at OIST.]*

This paper explores how core problems in PDE-based geometry processing can be efficiently and reliably solved via grid-free Monte Carlo methods. Modern geometric algorithms often need to solve Poisson-like equations on geometrically intricate domains. Conventional methods most often mesh the domain, which is both challenging and expensive for geometry with fine details or imperfections (holes, self-intersections, *etc.*). In contrast, grid-free Monte Carlo methods avoid mesh generation entirely, and instead just evaluate closest point queries. They hence do not discretize space, time, nor even function spaces, and provide the exact solution (in expectation) even on extremely challenging models. More broadly, they share many benefits with Monte Carlo methods from photorealistic rendering: excellent scaling, trivial parallel implementation, view-dependent evaluation, and the ability to work with any kind of geometry (including implicit or procedural descriptions). We develop a complete "black box" solver that encompasses integration, variance reduction, and visualization, and explore how it can be used for various geometry processing tasks. In particular, we consider several fundamental linear elliptic PDEs with constant coefficients on solid regions of $\mathbb{R}^n$. Overall we find that Monte Carlo methods significantly broaden the horizons of geometry processing, since they easily handle problems of size and complexity that are essentially hopeless for conventional methods.

CCS Concepts: • **Computing methodologies → Shape analysis.**

Additional Key Words and Phrases: numerical methods, stochastic solvers

ACM Reference Format:
Rohan Sawhney and Keenan Crane. 2020. Monte Carlo Geometry Processing: A Grid-Free Approach to PDE-Based Methods on Volumetric Domains. *ACM Trans. Graph.* 39, 4, Article 123 (July 2020), 18 pages. https://doi.org/10.1145/3386569.3392374

1 INTRODUCTION

The complexity of geometric models has increased dramatically in recent years, but is still far from matching the complexity found in nature—consider, for instance, detailed microstructures that give rise to physical or biological behavior (Fig. 1). PDE-based methods provide powerful tools for processing and analyzing such data, but have not yet reached a point where algorithms "just work": even basic tasks still entail careful preprocessing or parameter tuning, and robust algorithms can exhibit poor scaling in time or memory. Monte Carlo methods provide new opportunities for geometry processing, making a sharp break with traditional finite element methods (FEM). In particular, by avoiding the daunting challenge of *mesh generation* they offer a framework that is highly scalable, parallelizable, and numerically robust, and significantly expands the kind of geometry that can be used in PDE-based algorithms.

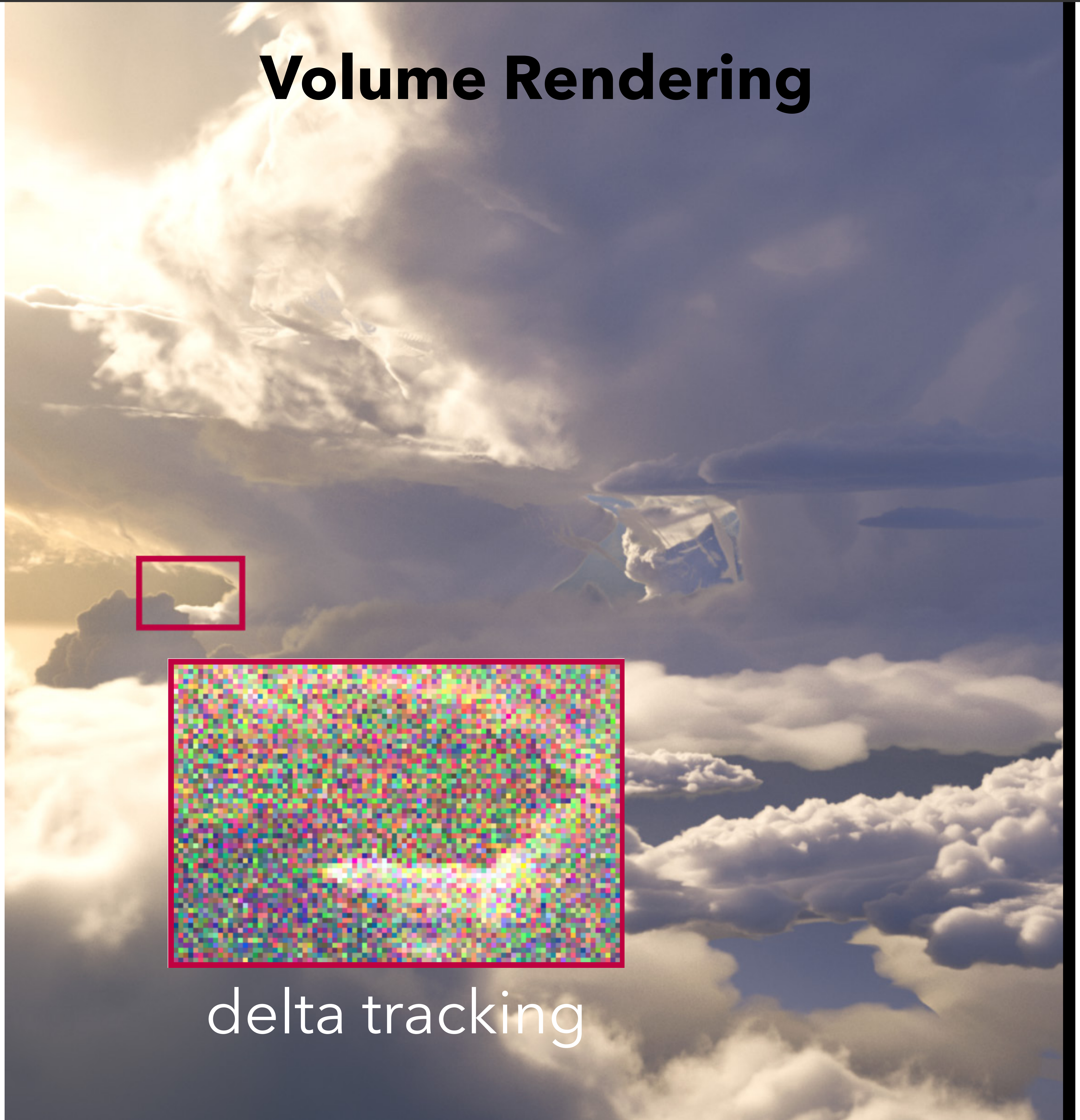
Photorealistic rendering experienced an analogous development around the 1990s: finite element radiosity [Goral et al. 1984] gave way to Monte Carlo integration of the light transport equation [Kajiya 1986], for reasons that are nicely summarized by Wann Jensen [2001, Chapter 1]. Although this shift was motivated in part by a desire for more complex illumination, it has also made it possible to work with scenes of extreme *geometric* complexity—modern renderers handle trillions of effective polygons [Georgiev et al. 2018] and, in stark contrast to FEM, yield high-quality results even for

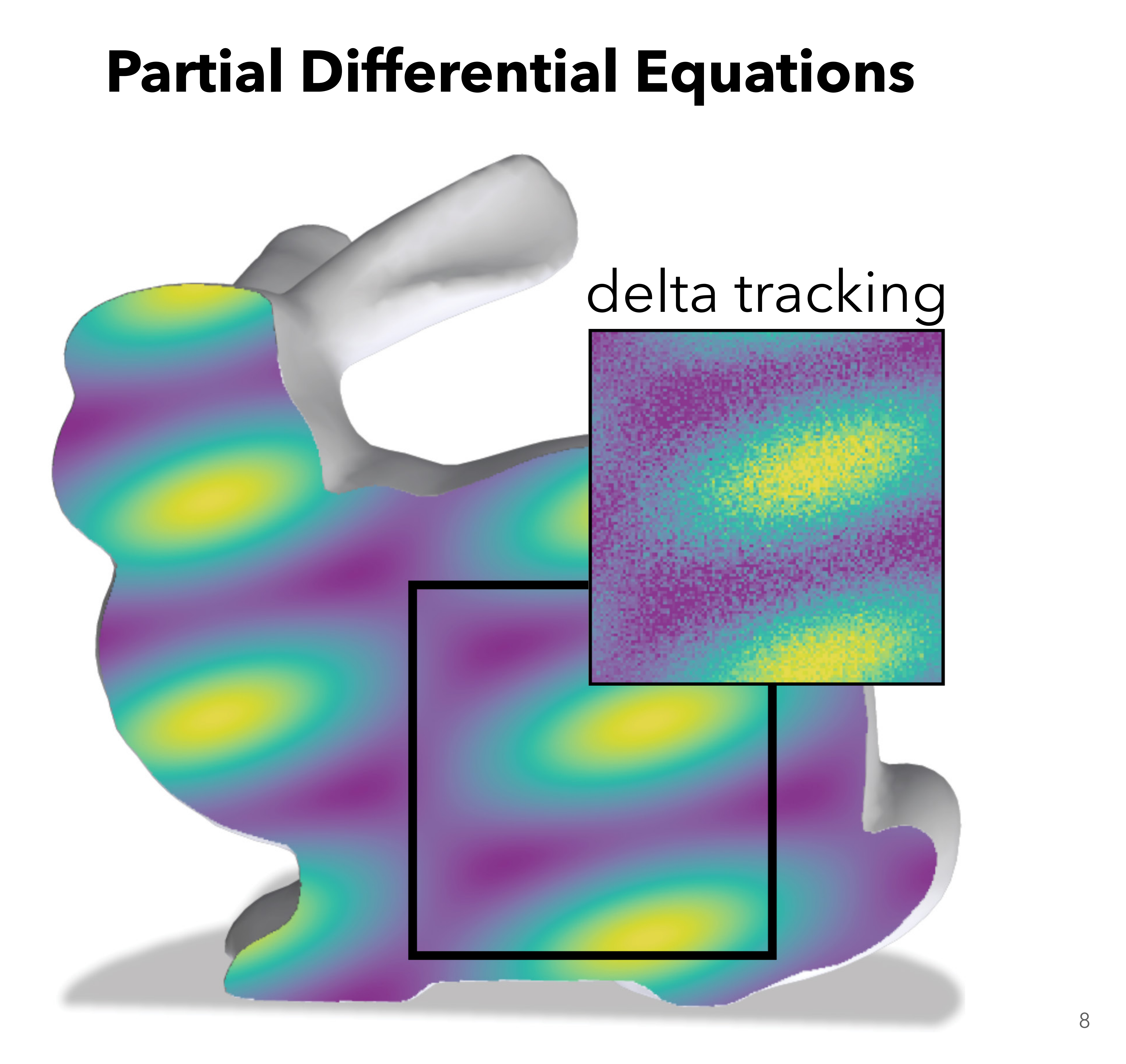
Authors' address: Rohan Sawhney; Keenan Crane, Carnegie Mellon University, 5000 Forbes Ave, Pittsburgh, PA, 15213.

Permission to make digital or hard copies of part or all of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for third-party components of this work must be honored. For all other uses, contact the owner/author(s).
© 2020 Copyright held by the owner/author(s).
0730-0301/2020/7-ART123
https://doi.org/10.1145/3386569.3392374

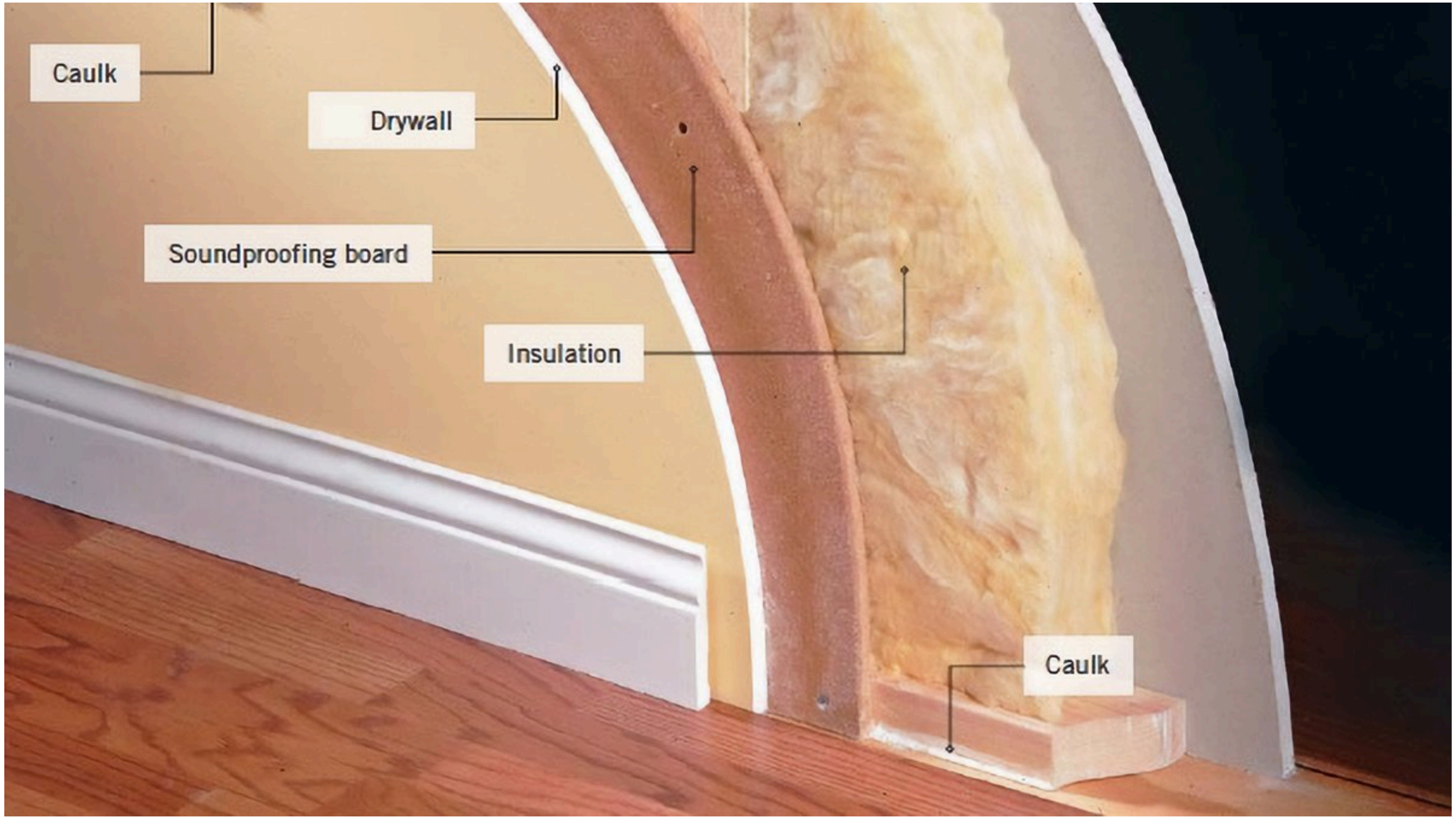
ACM Trans. Graph., Vol. 39, No. 4, Article 123. Publication date: July 2020.

Contribution: Bridge between PDEs & Volume Rendering





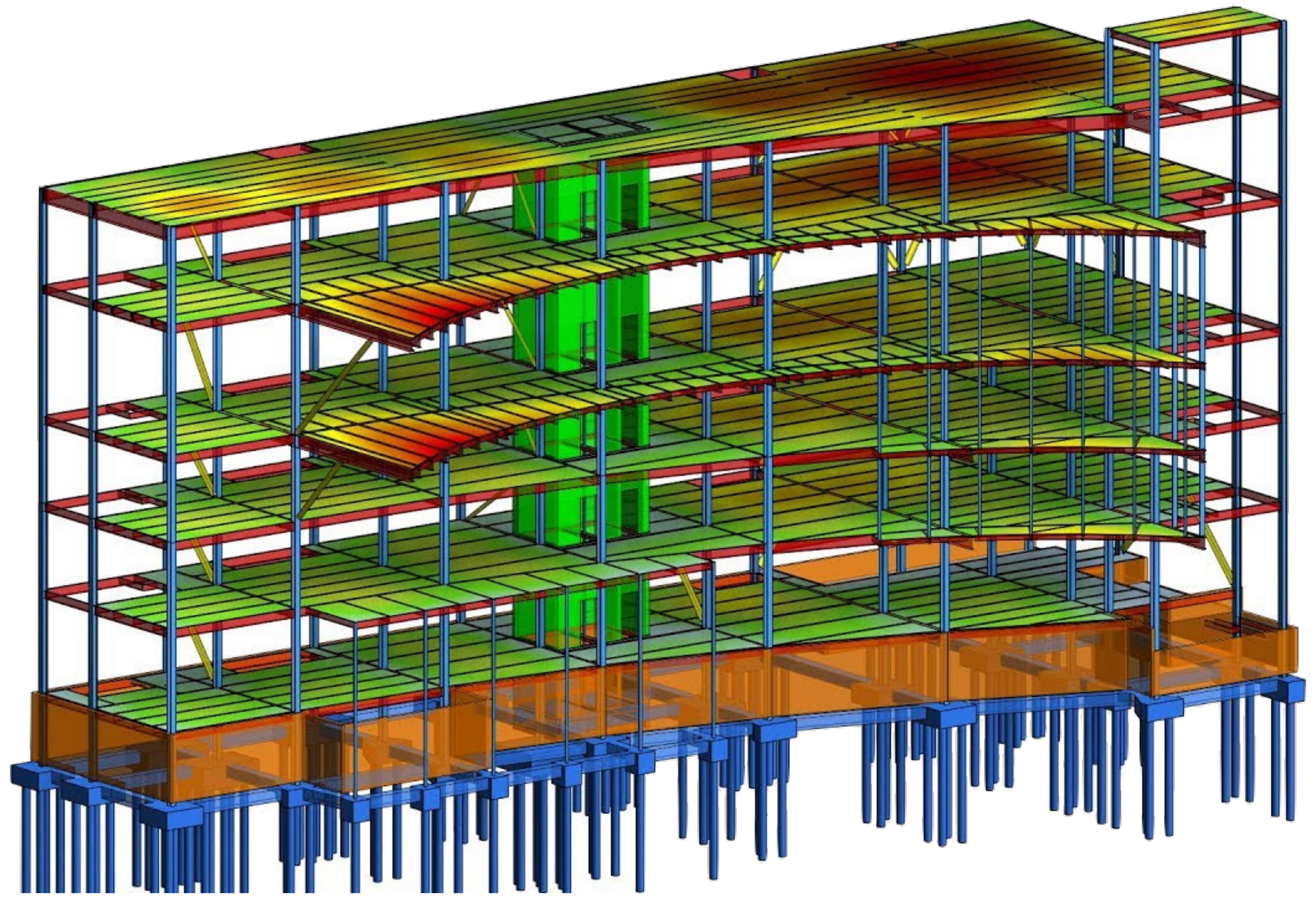
Spatial heterogeneity is everywhere!



Rich material properties e.g., wall with thermal insulation, sound proofing et.c

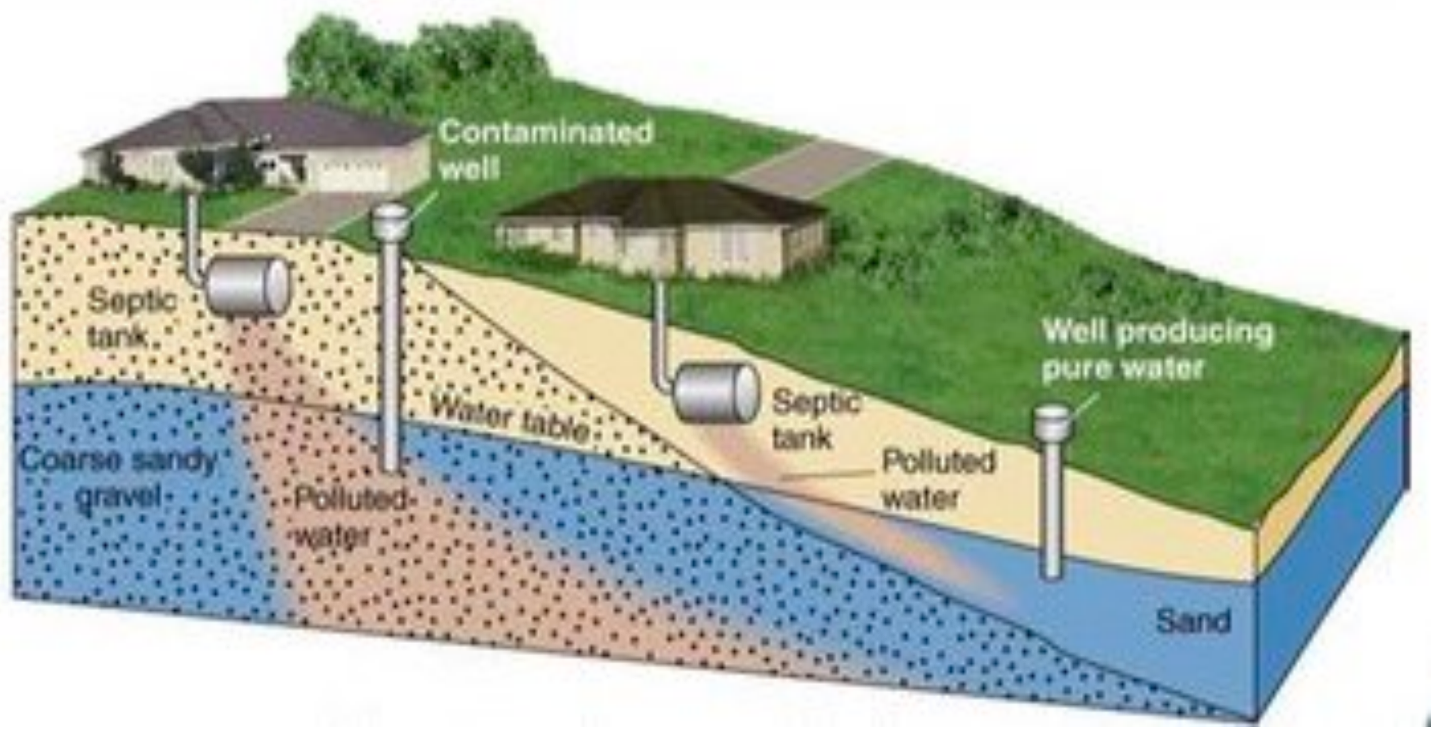
Spatial heterogeneity is everywhere!

thermal performance

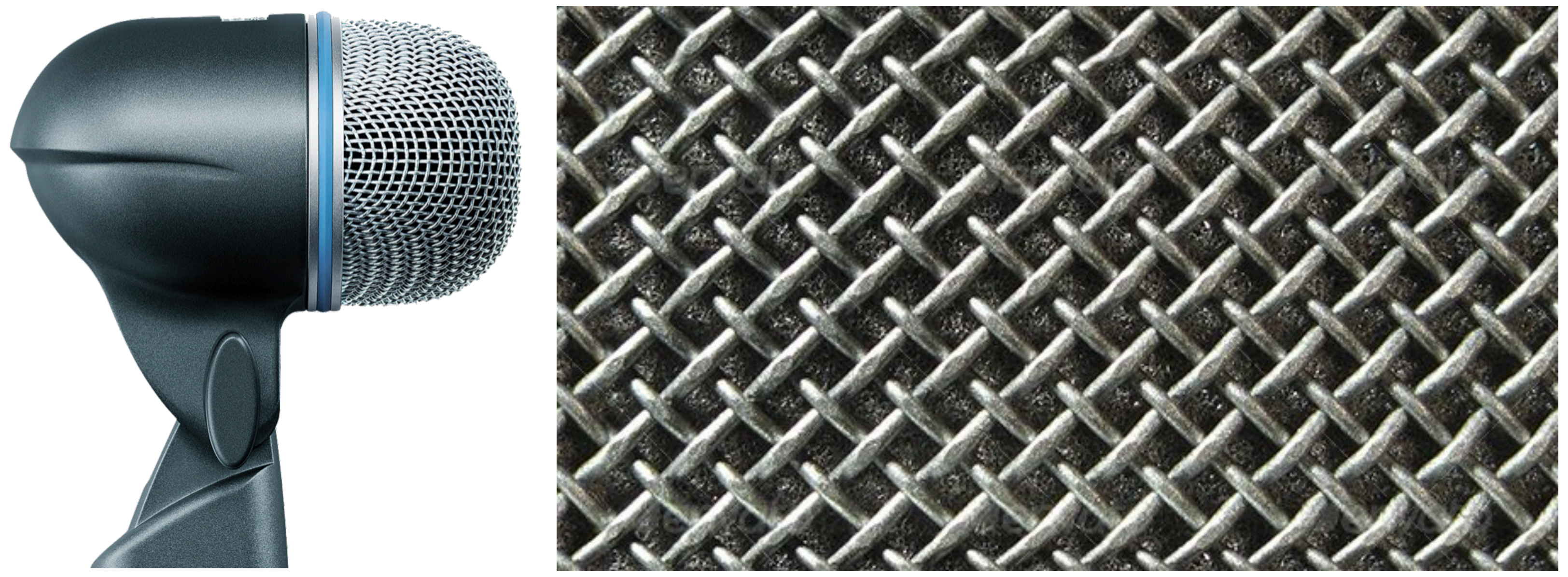


structural analysis

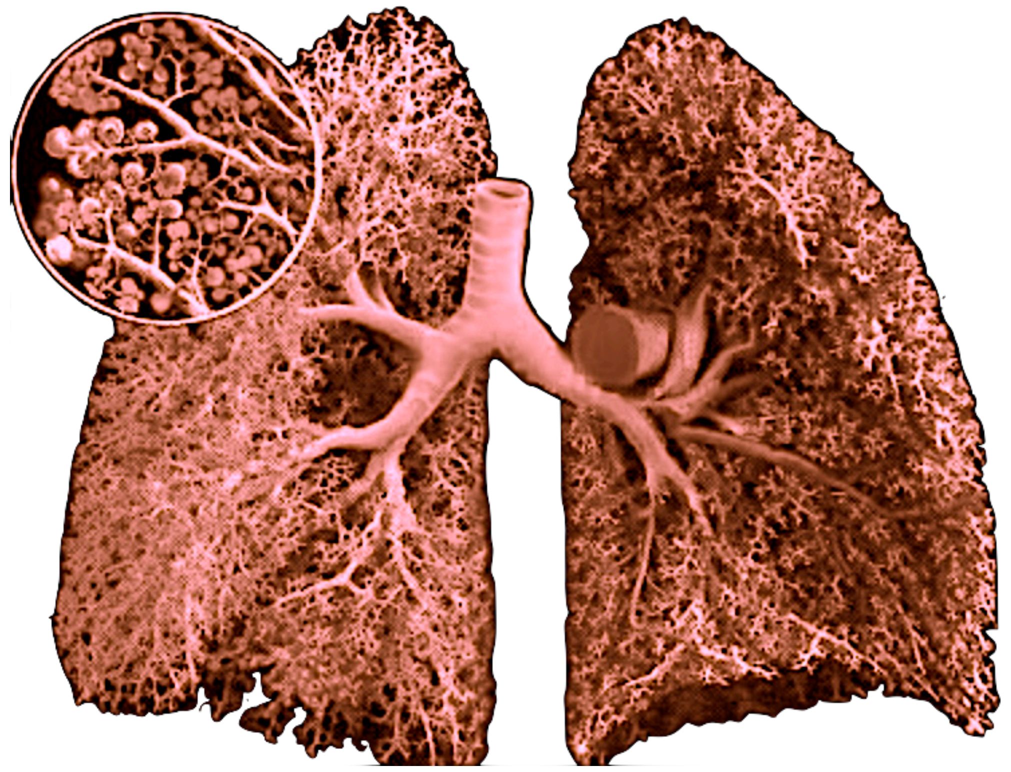
electrical capacitance



geological modeling



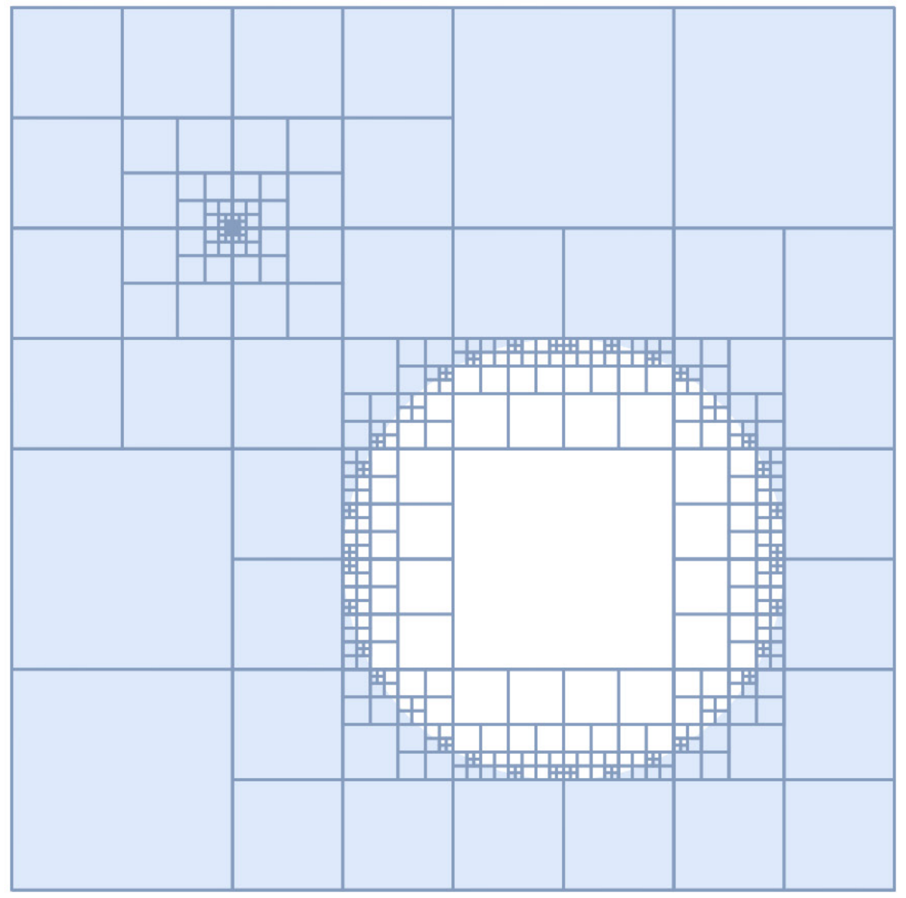
acoustic performance



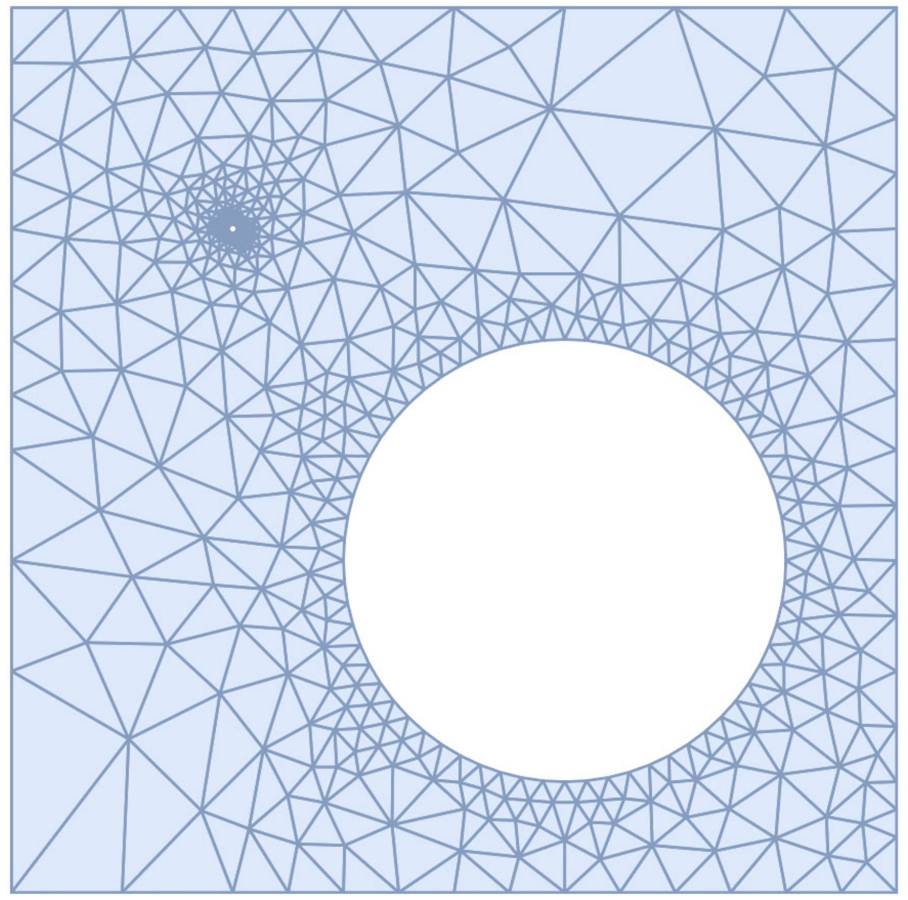
biological modeling

Shortcoming of conventional PDE solvers

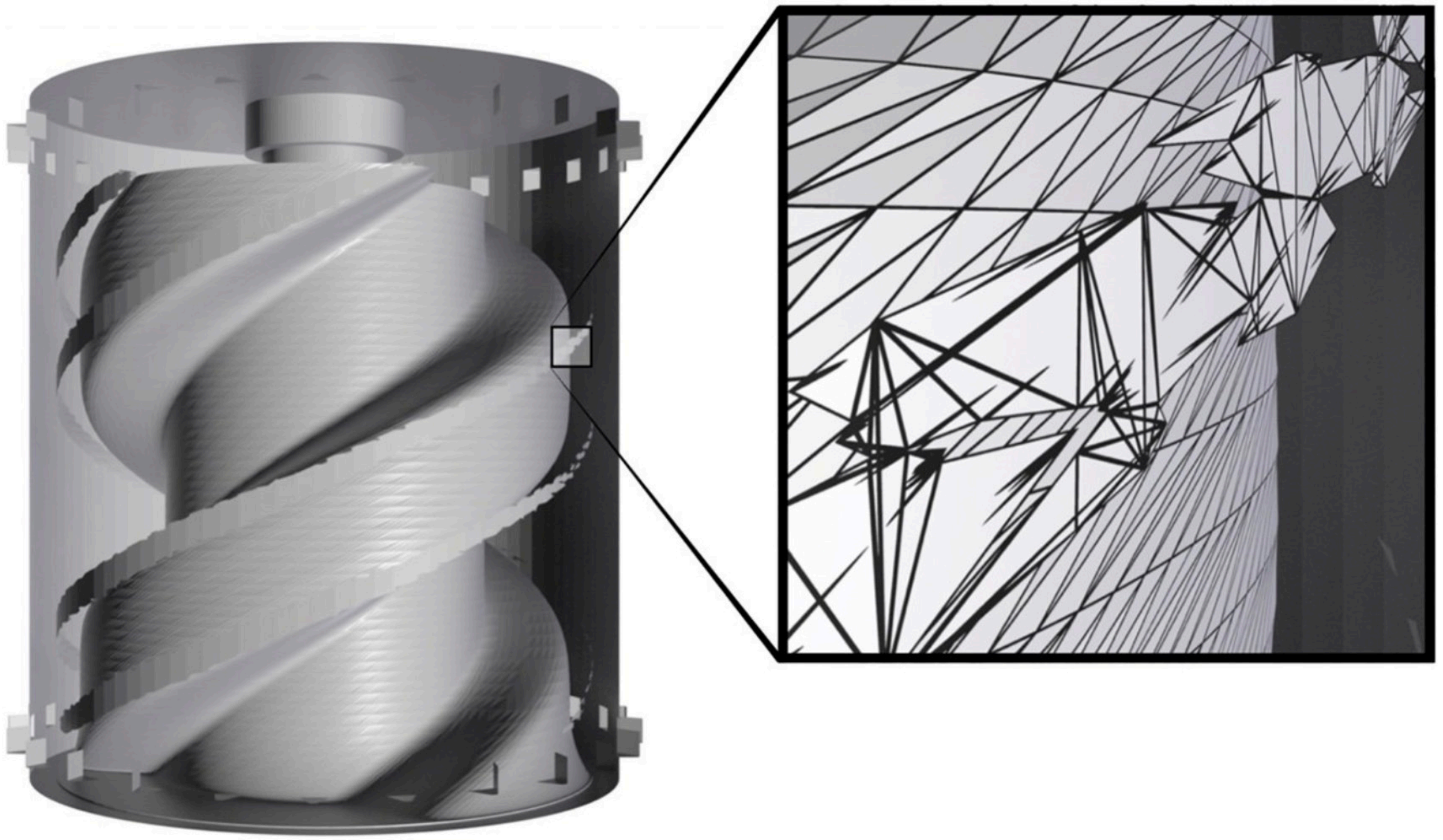
Spatial discretization: expensive and error-prone



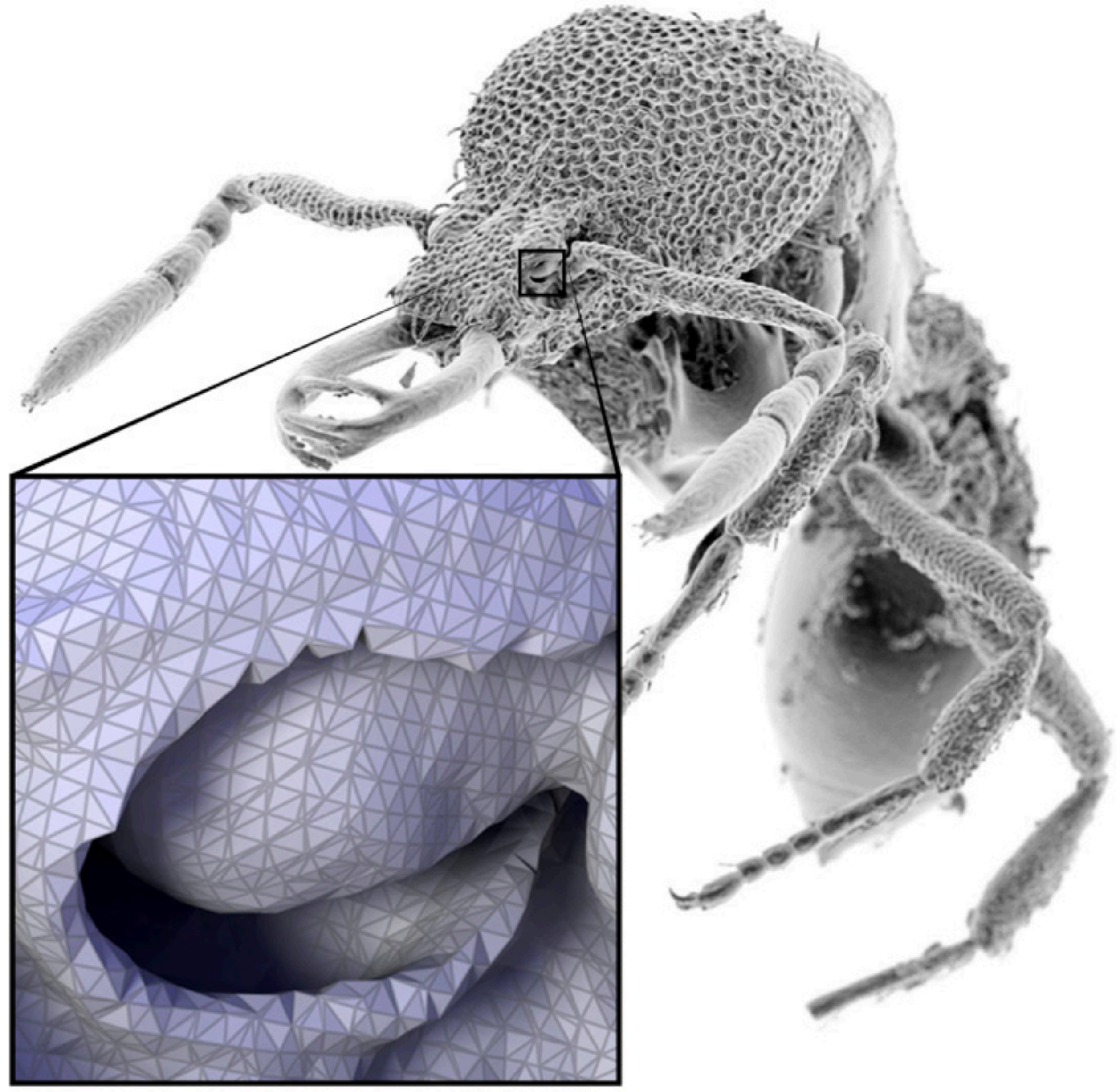
finite difference



finite element



Defective geometry, e.g.,
self-intersections, non-manifold elements

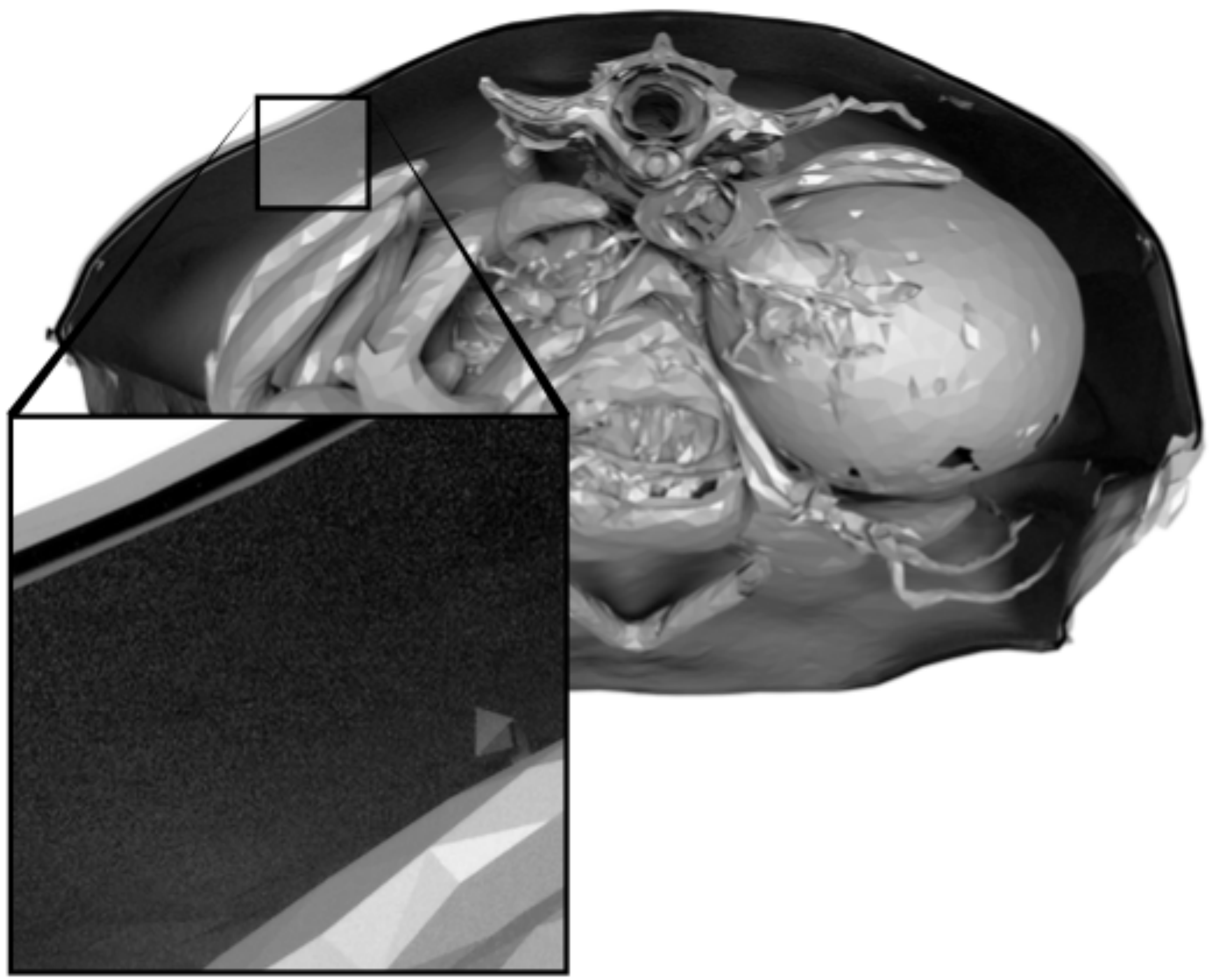


14 hrs / 30 GB RAM
to generate FEM mesh

Shortcoming of conventional PDE solvers

Spatial discretization: destroys geometric features

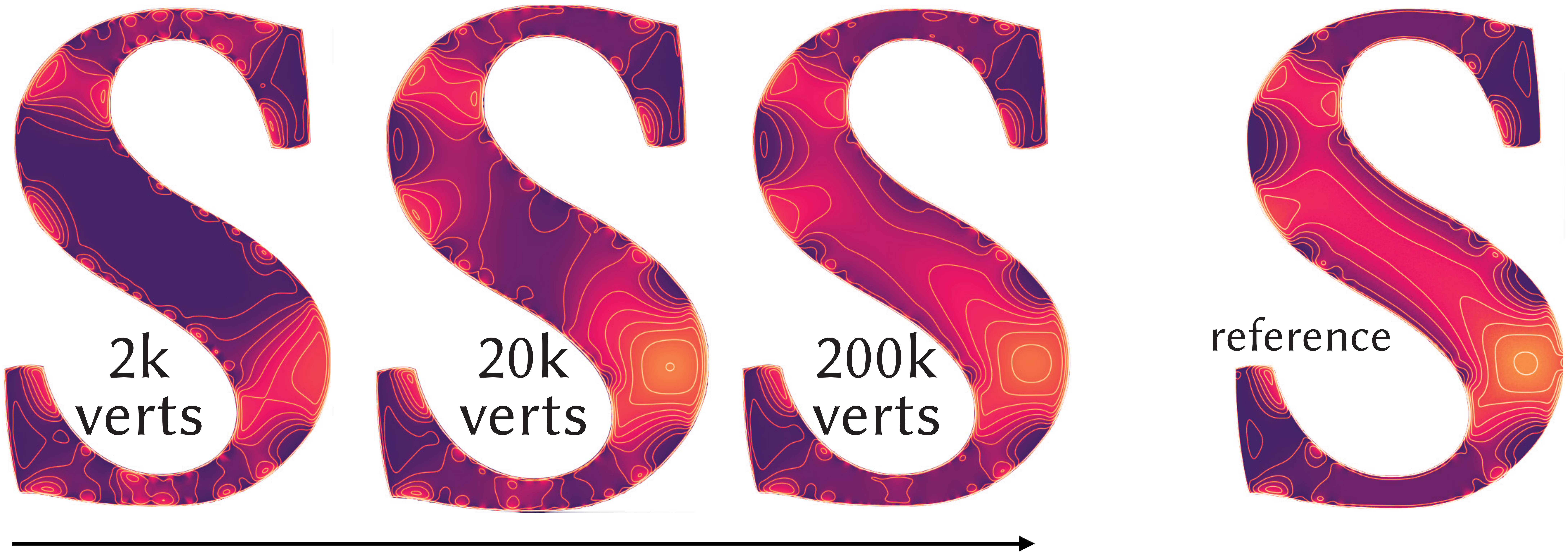
boundary mesh (input)



34 minutes / 6.1 GB RAM to generate FEM mesh (missing blood vessels)

Shortcoming of conventional PDE solvers

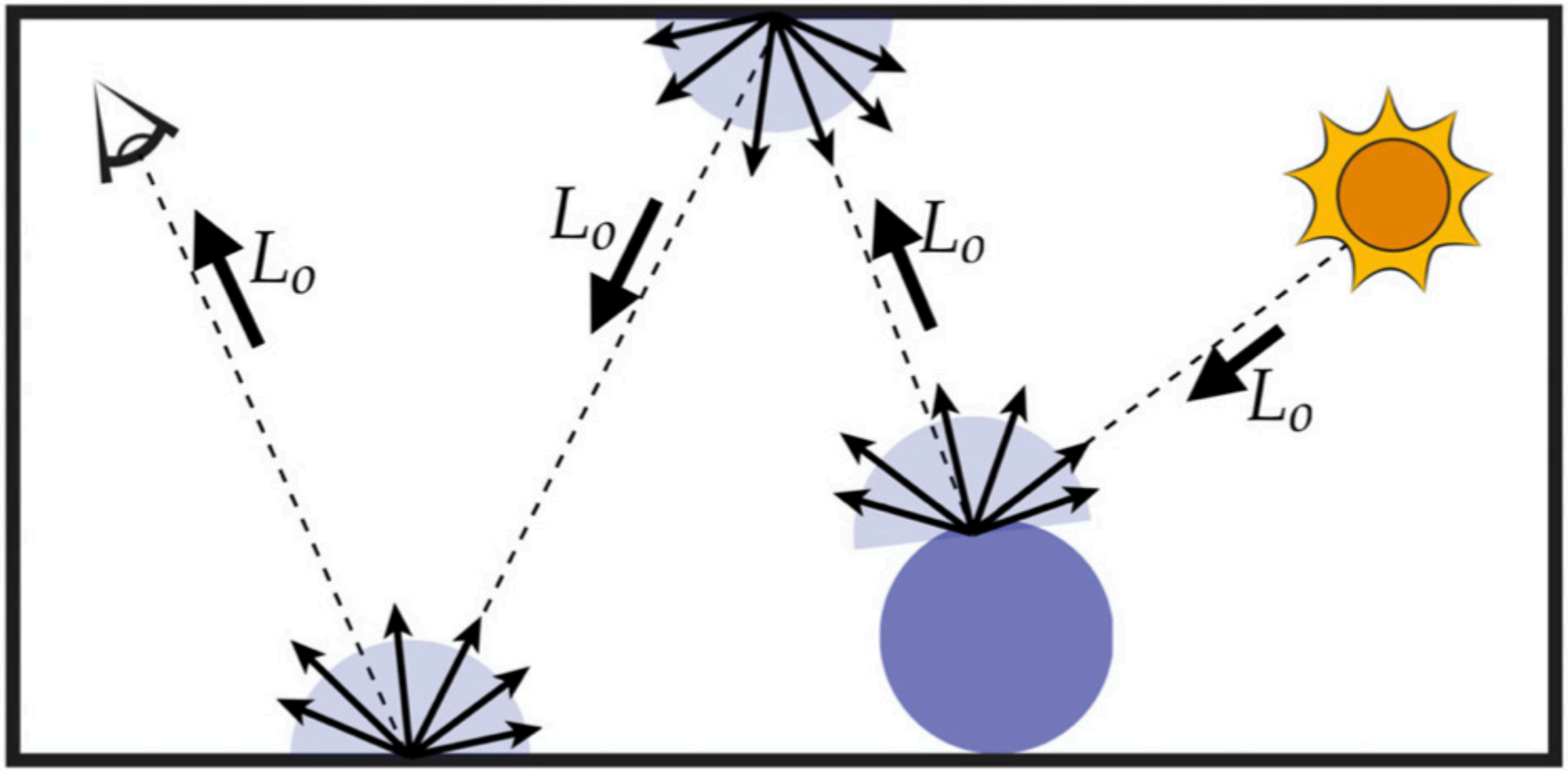
Spatial discretization: causes aliasing in the PDE inputs and solution

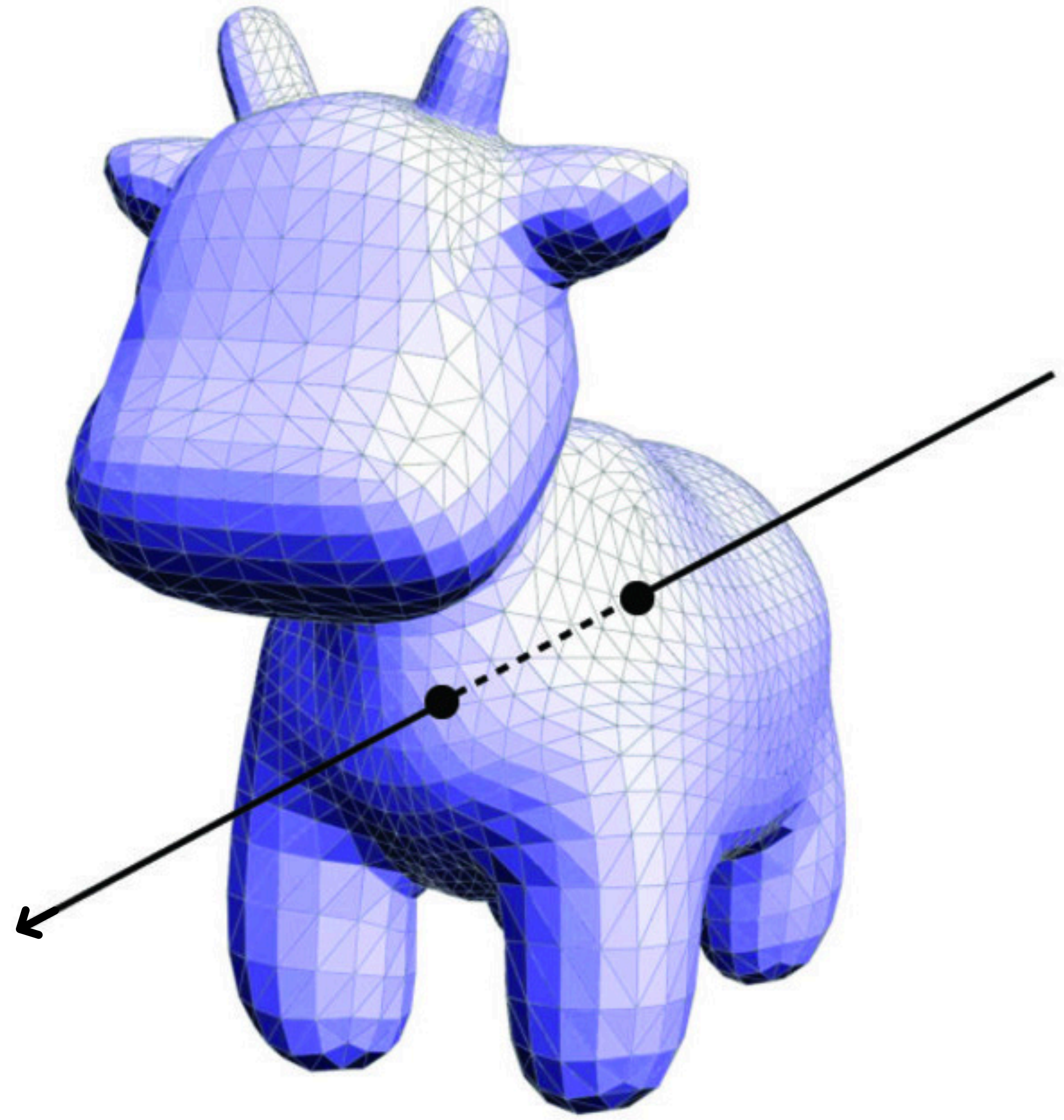


FEM requires significant **mesh refinement** to match reference

Monte Carlo Rendering

Does not require high quality meshing & solving global systems

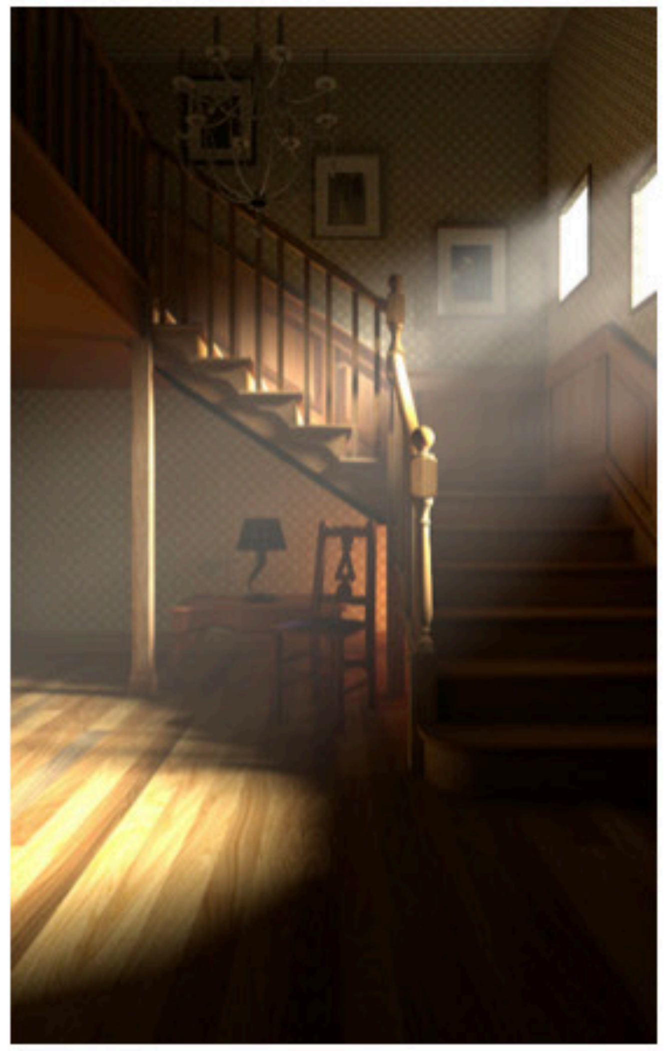


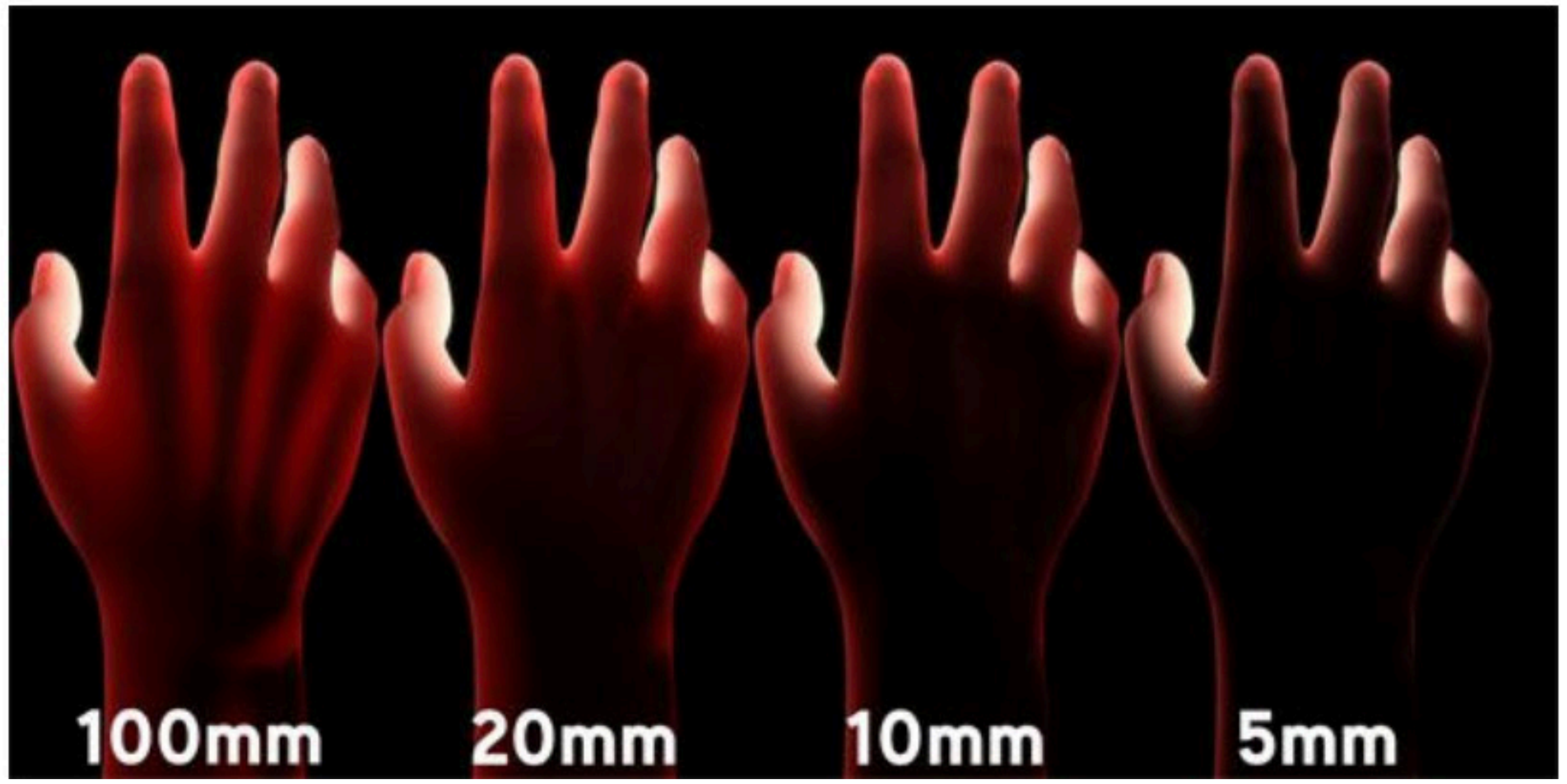


Ray intersection query

Monte Carlo Rendering

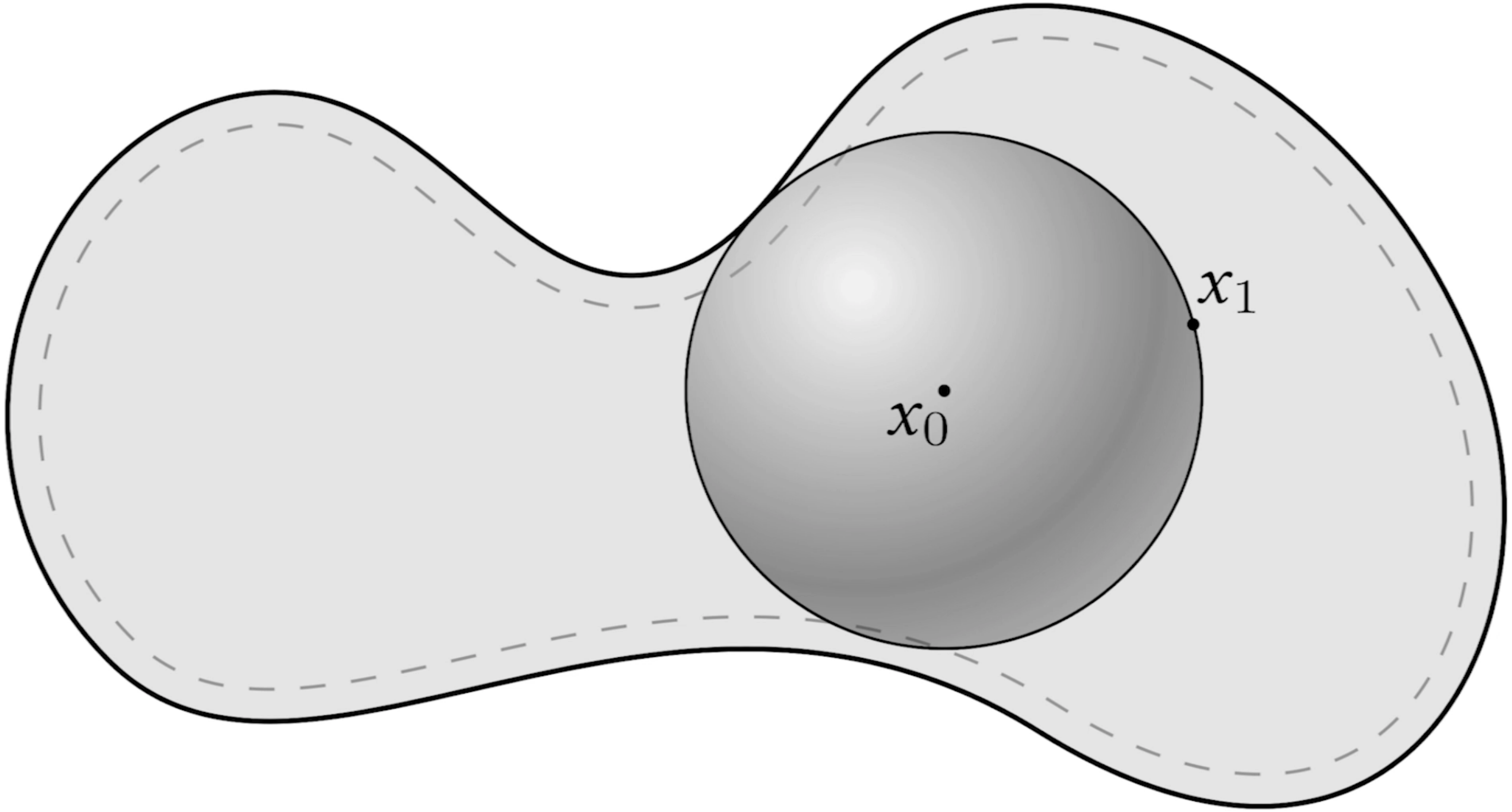
Photorealistic image generation of participating and granular media





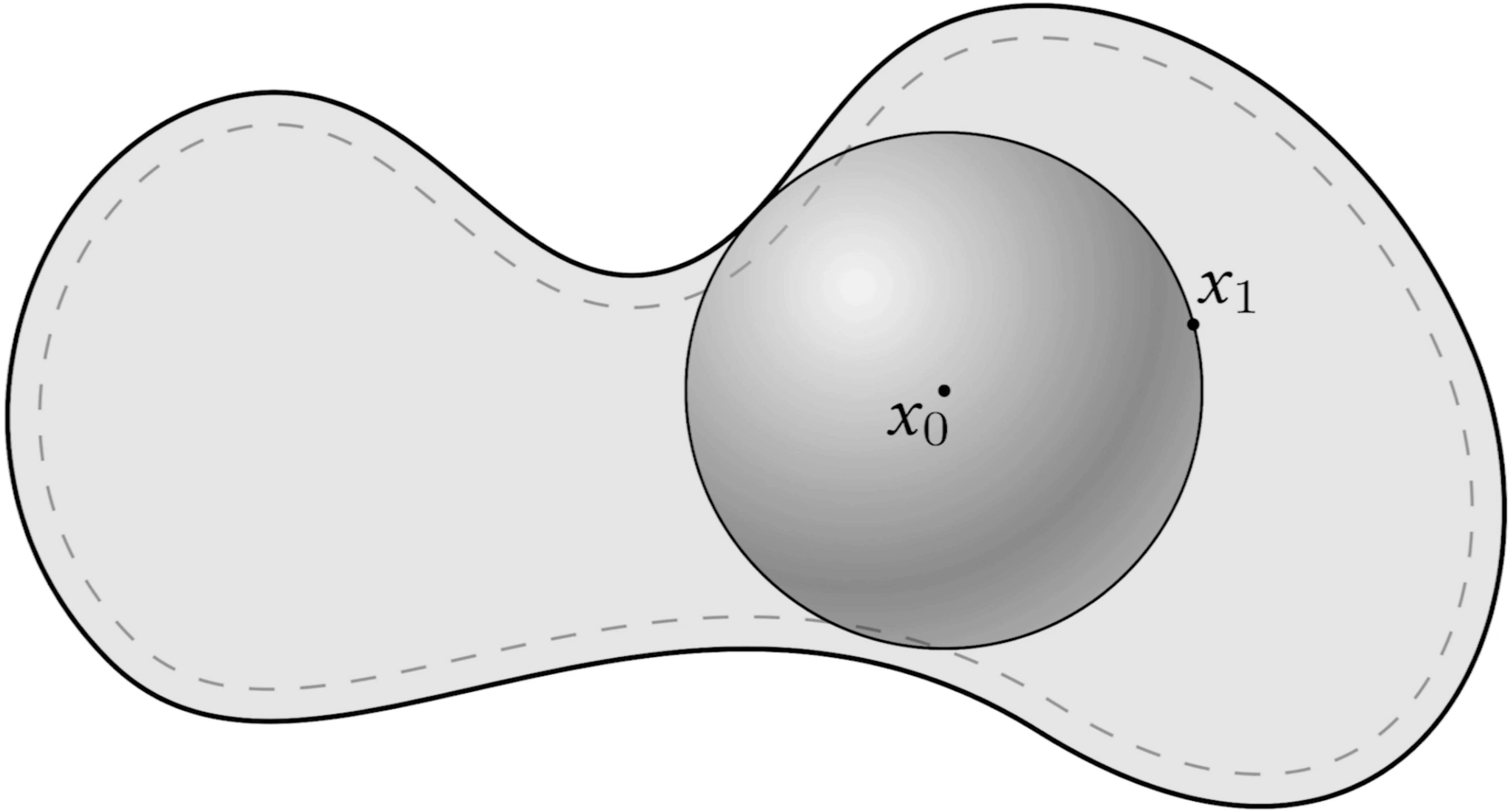
The walk on spheres (WoS) algorithm [Muller 1956]

Does not require high quality meshing & solving global systems



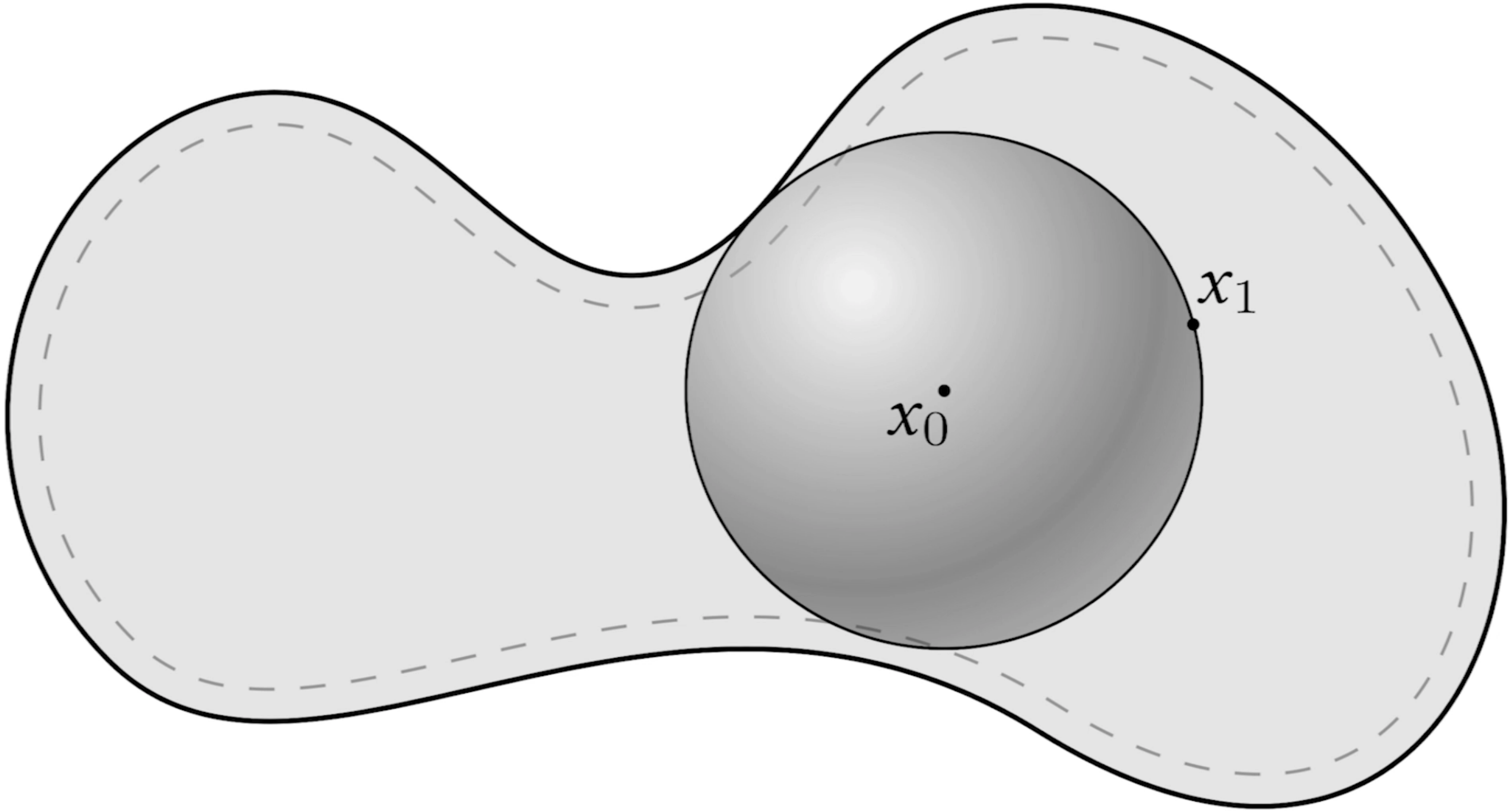
The walk on spheres (WoS) algorithm [Muller 1956]

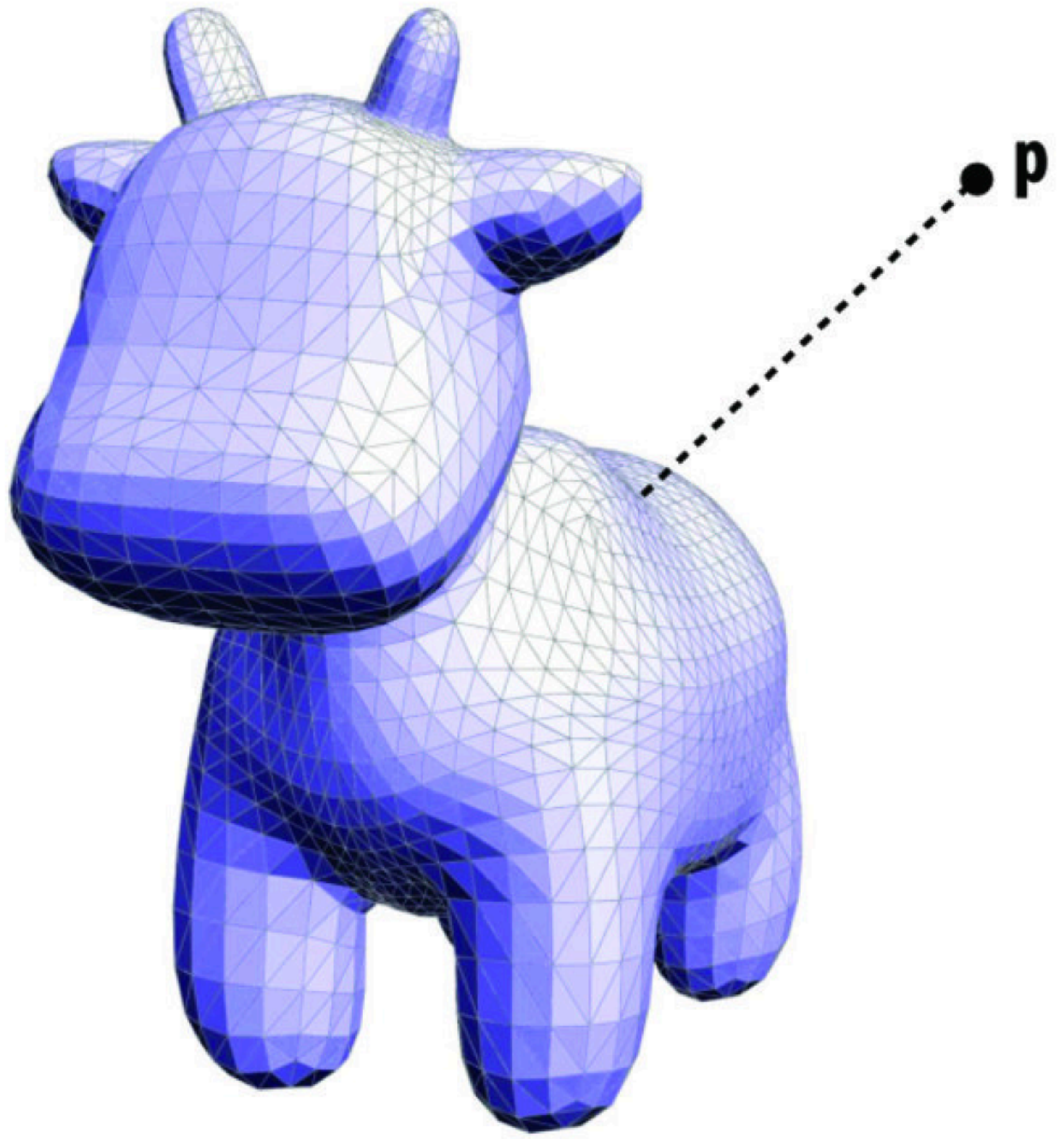
Does not require high quality meshing & solving global systems



The walk on spheres (WoS) algorithm [Muller 1956]

Does not require high quality meshing & solving global systems





Closest point query

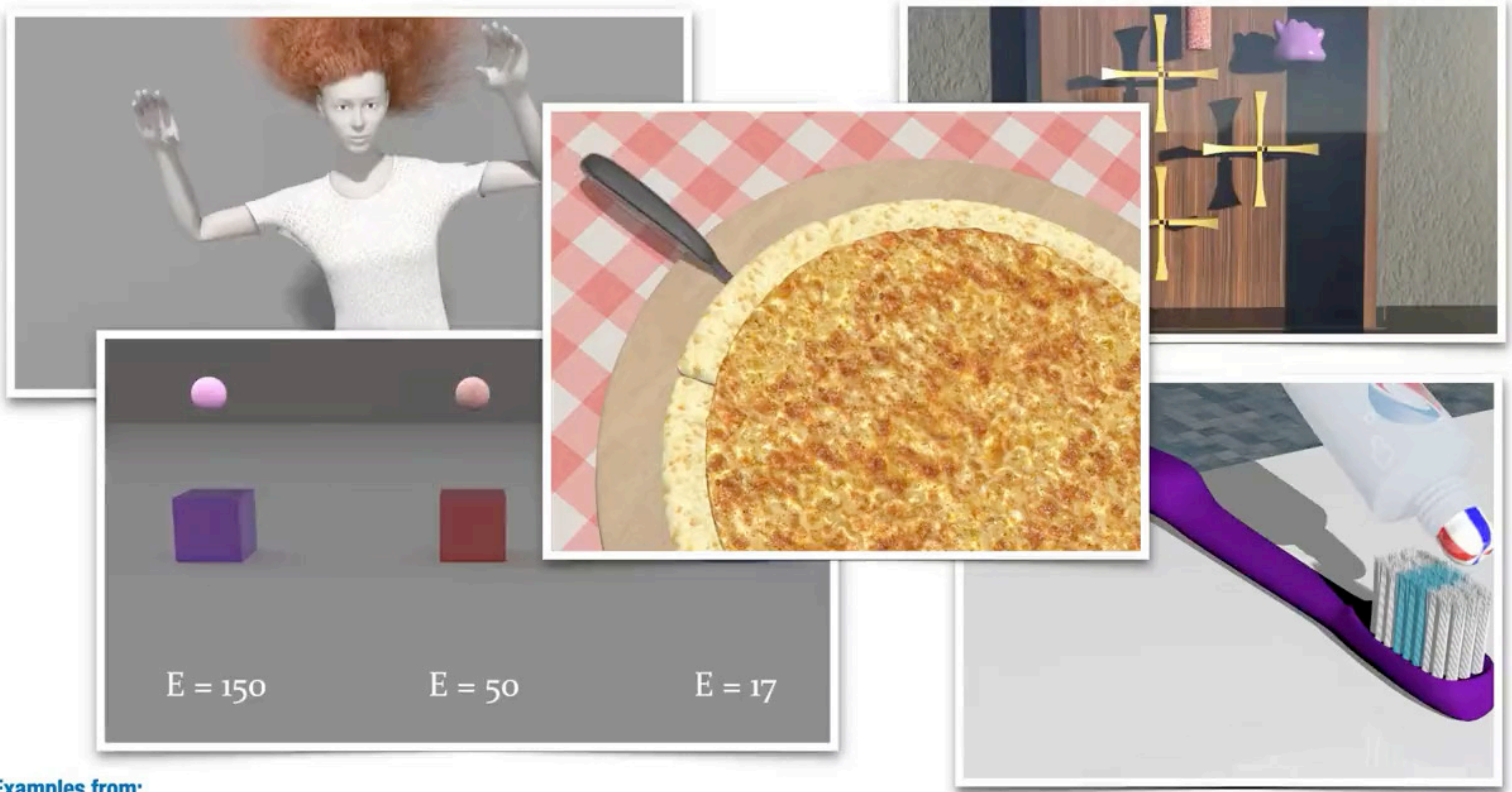
Multi-material physical simulation in graphics

Examples from:

Han et al, "A Hybrid Material Point Method for Frictional Contact with Diverse Materials" (2019)

Zhu et al, "Codimensional Non-Newtonian Fluids" (2015)

Multi-material physical simulation in graphics

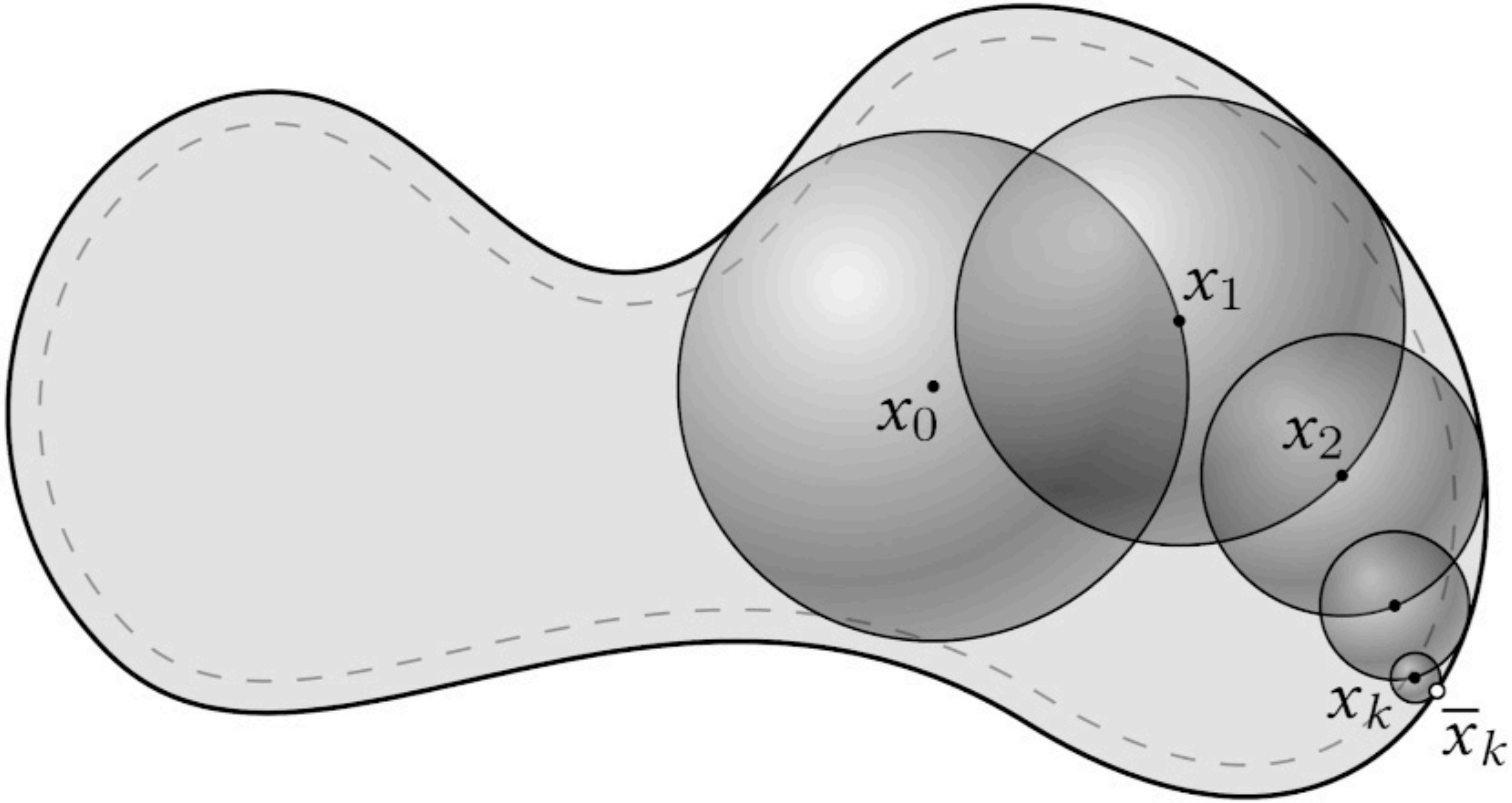


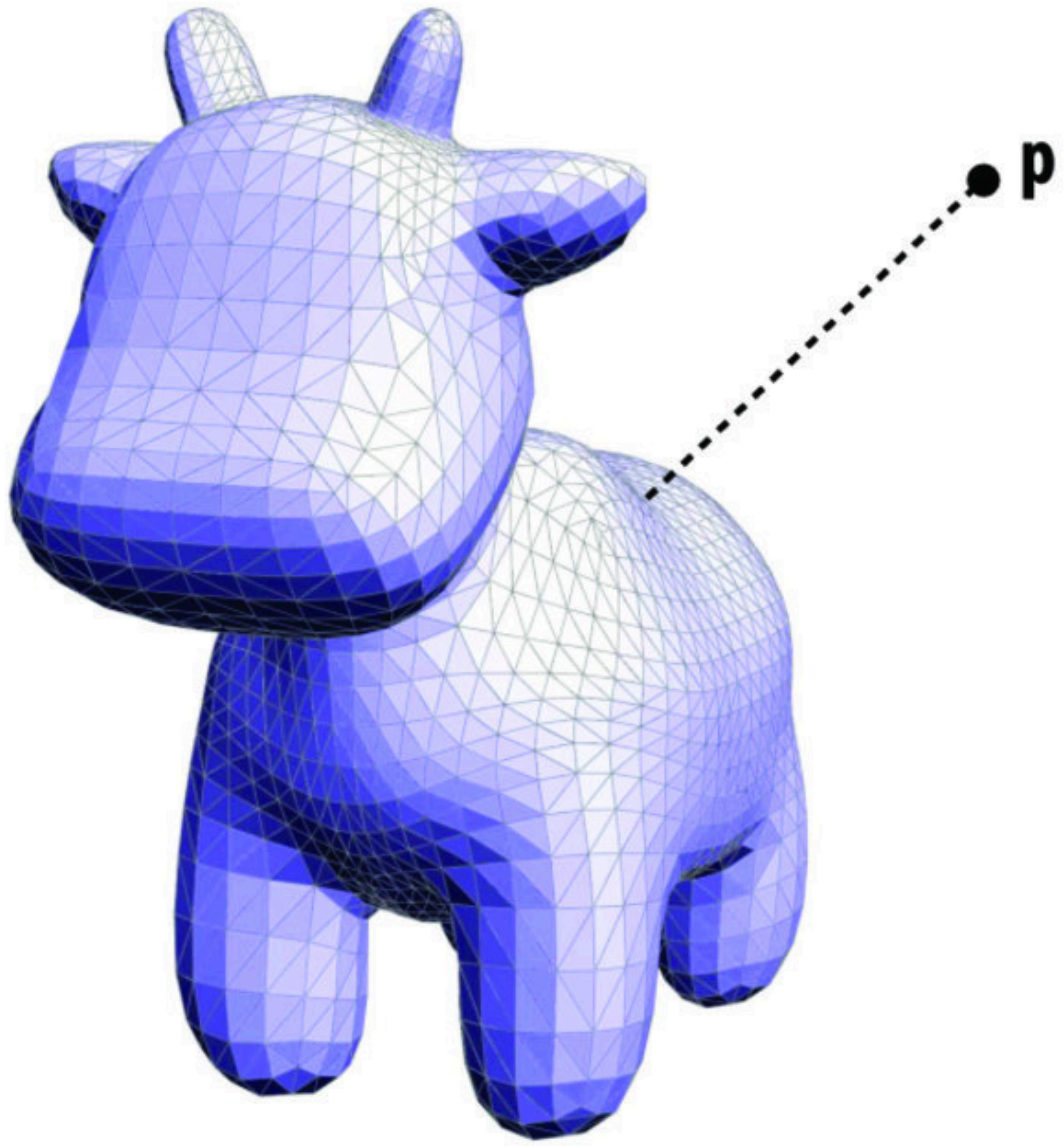
Examples from:

Han et al, "A Hybrid Material Point Method for Frictional Contact with Diverse Materials" (2019)
Zhu et al, "Codimensional Non-Newtonian Fluids" (2015)

The walk on spheres (WoS) algorithm [Muller 1956]

Goal: extend to broader class of problems $\Longrightarrow$ PDEs with variable coefficients

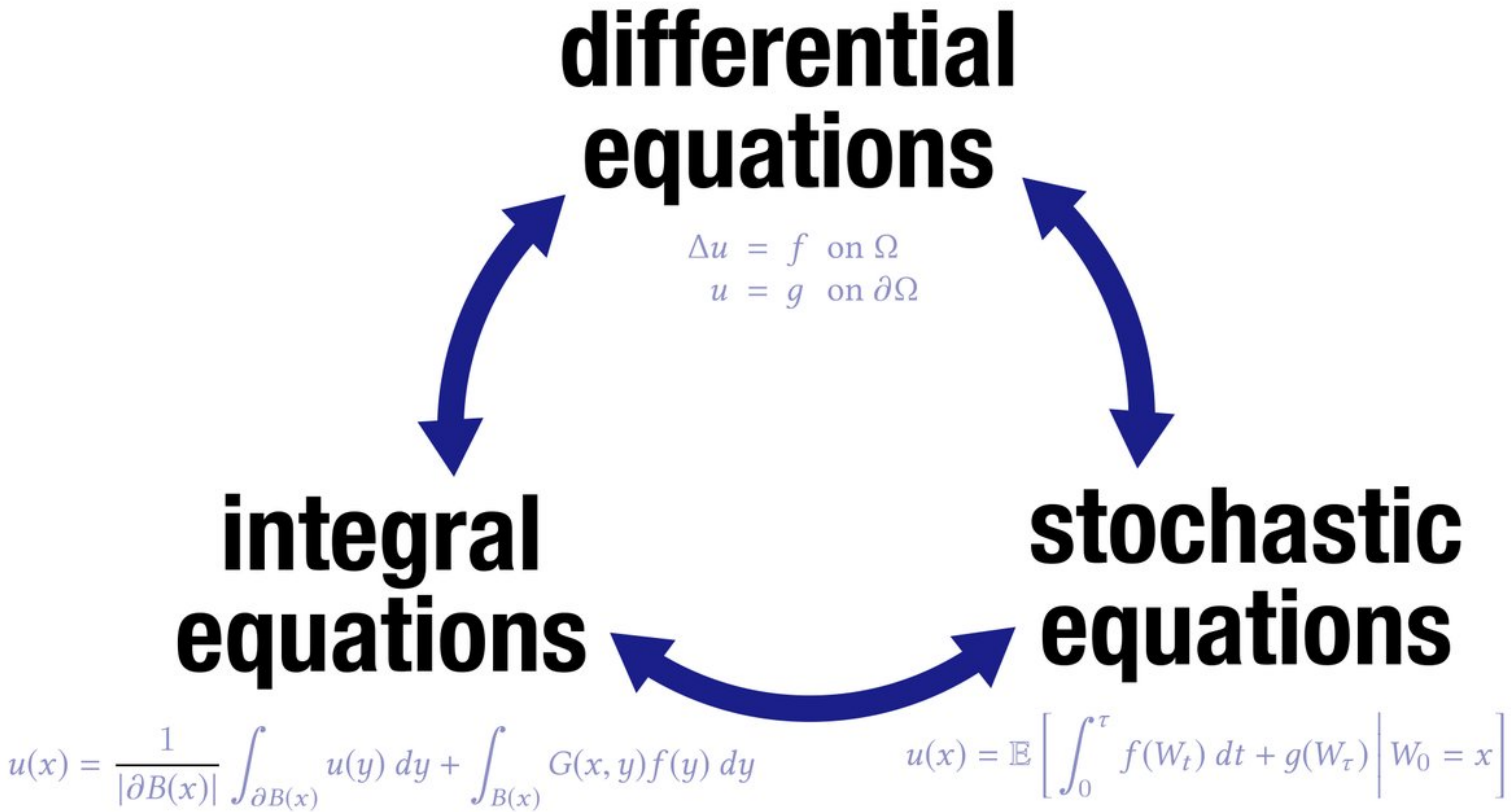




Closest point query

BACKGROUND

A tale of three equations…



2nd order linear elliptic PDEs

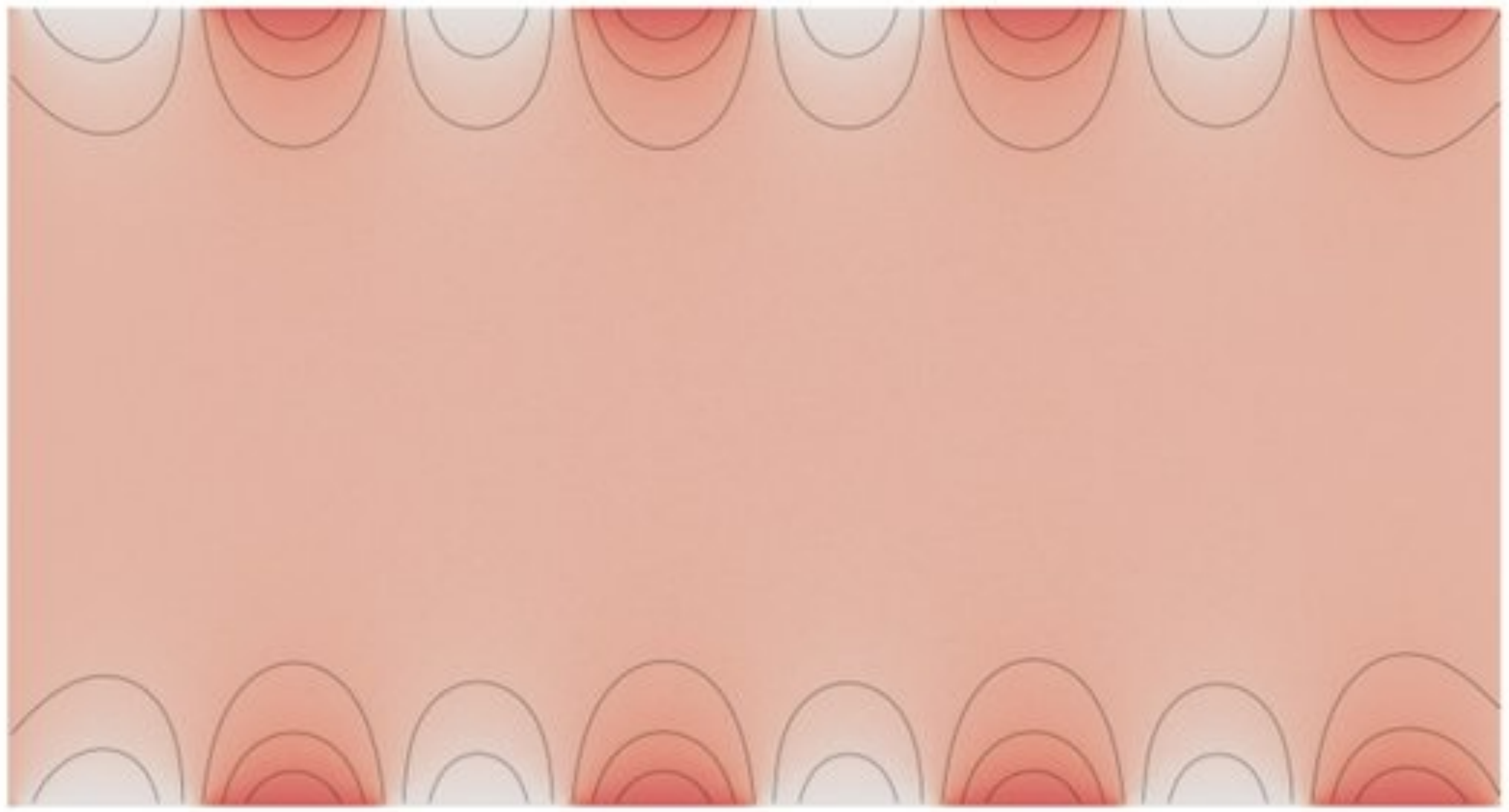
$$\overbrace{\nabla \cdot (\alpha \nabla u)}^{\text{diffusion}} + \overbrace{\vec{\omega} \cdot \nabla u}^{\text{drift}} - \overbrace{\sigma u}^{\text{absorption}} = \overbrace{-f}^{\text{source}} \text{ on } \overbrace{\Omega}^{\text{domain}}$$

$$\underbrace{u}_{\text{solution}} = \underbrace{g}_{\text{boundary values}} \text{ on } \underbrace{\partial\Omega}_{\text{domain boundary}}$$

2nd order linear elliptic PDEs

Laplace equation

boundary values $g(x)$



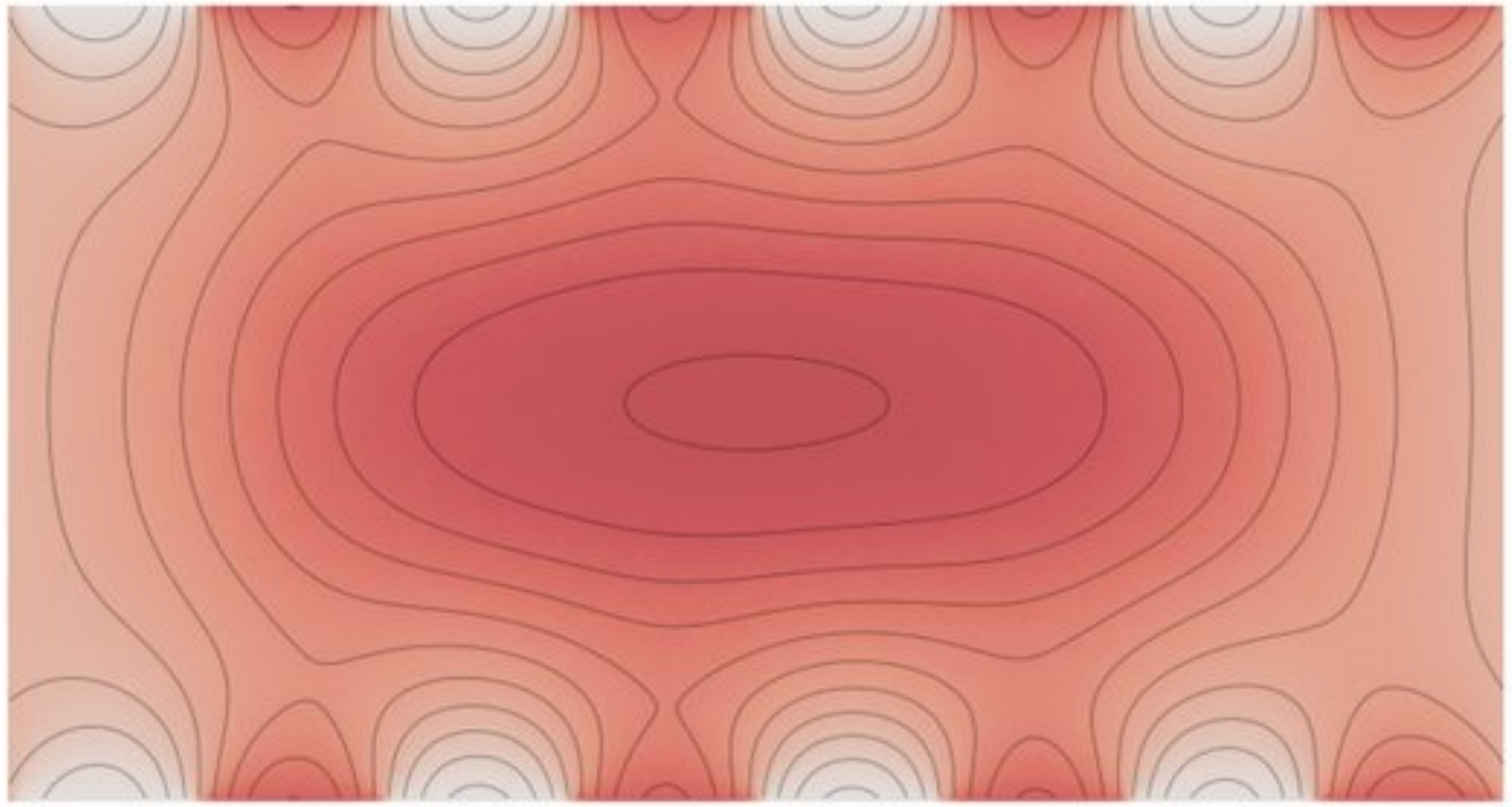
$$\Delta u = 0$$

$$\Delta := \frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial y^2} + \frac{\partial^2}{\partial z^2}$$

2nd order linear elliptic PDEs

Poisson equation

source
term $f(x)$

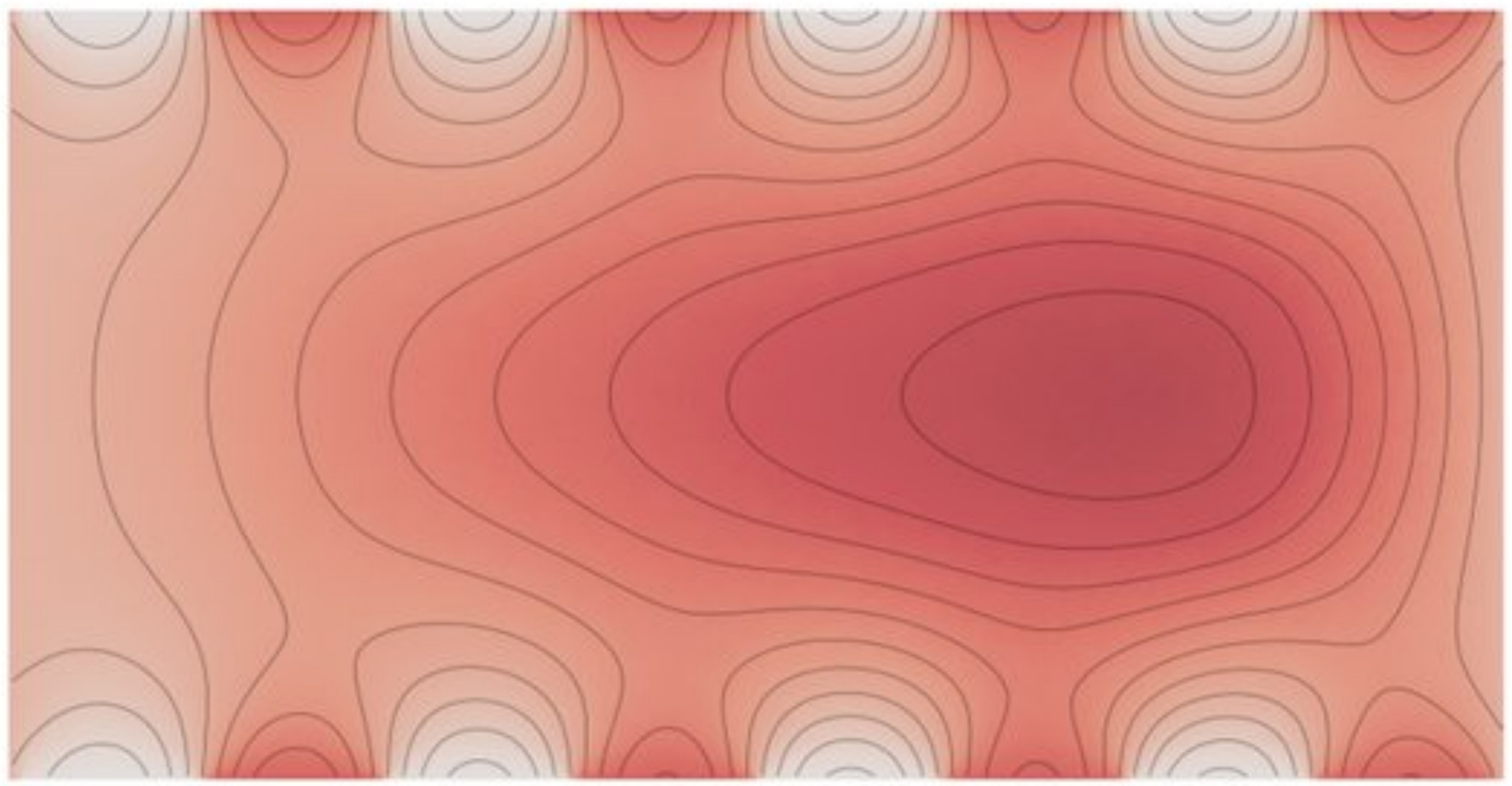


$\Delta u = f$

2nd order linear elliptic PDEs

variable diffusion Poisson equation

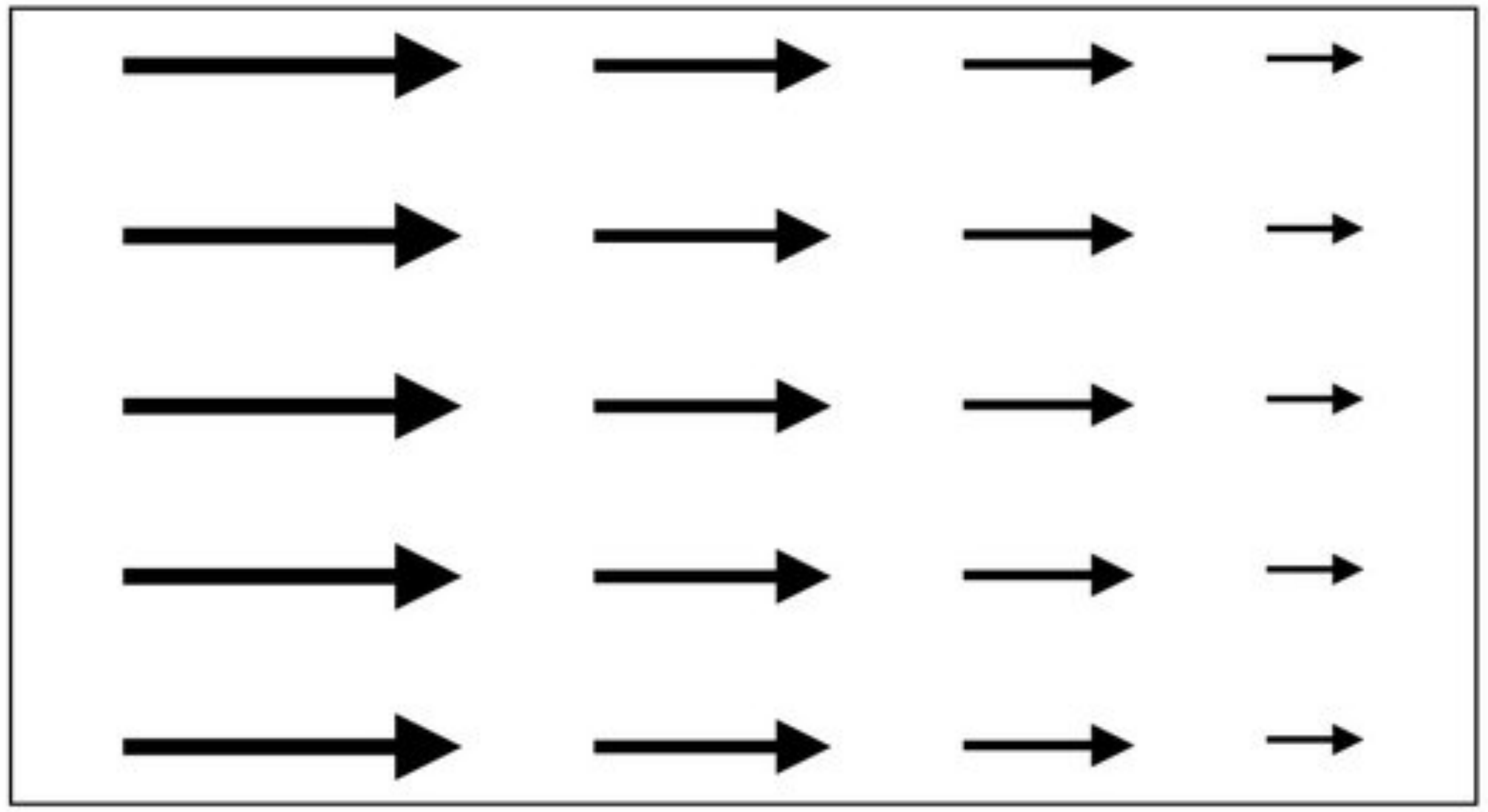
diffusion
coeff. $\alpha(x)$



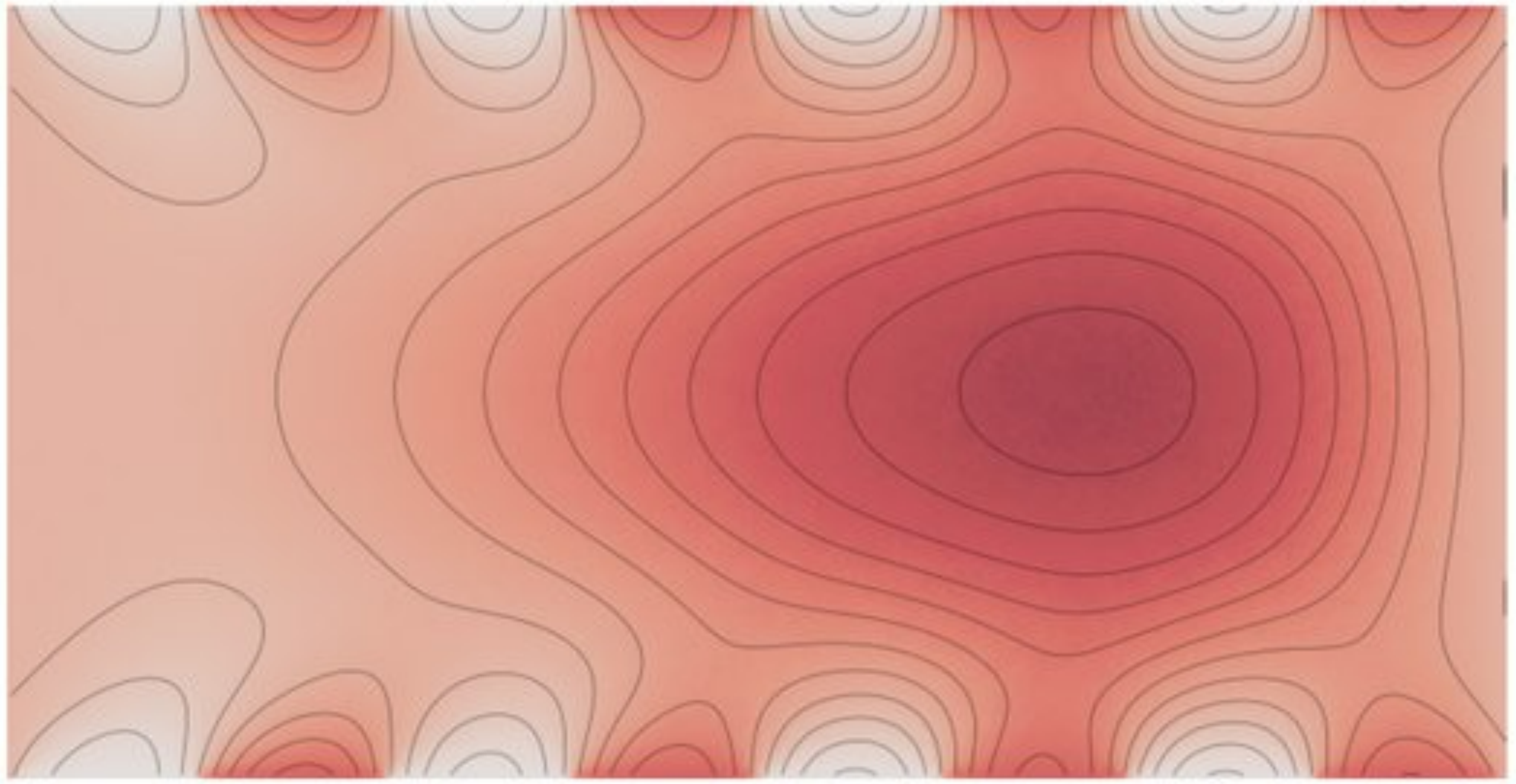
$\nabla \cdot (\alpha \nabla u) = f$

2nd order linear elliptic PDEs

stationary advection-diffusion equation



transport
coeff. $\vec{\omega}(x)$



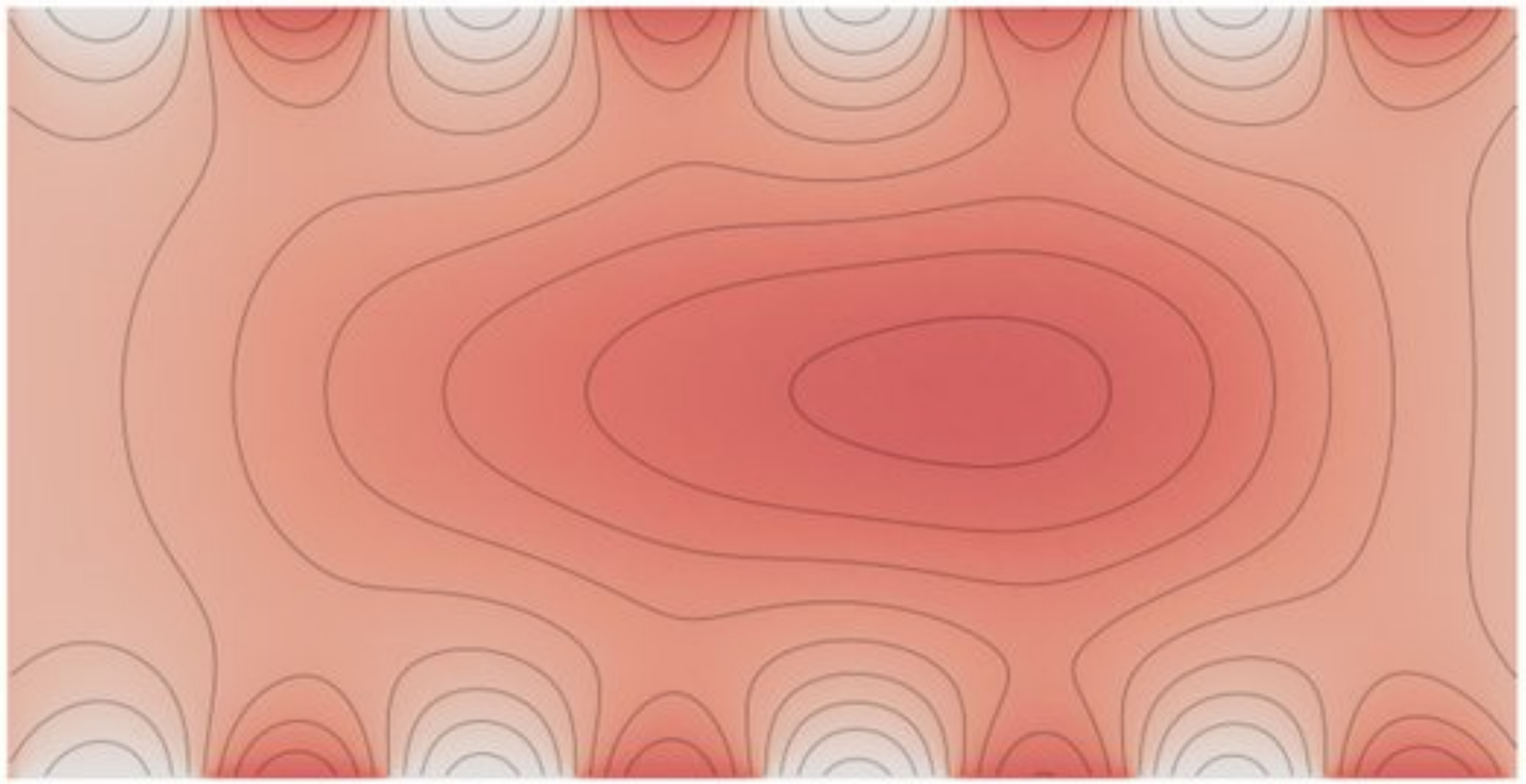
$$\Delta u + \vec{\omega} \cdot \nabla u = f$$

2nd order linear elliptic PDEs

screened Poisson equation

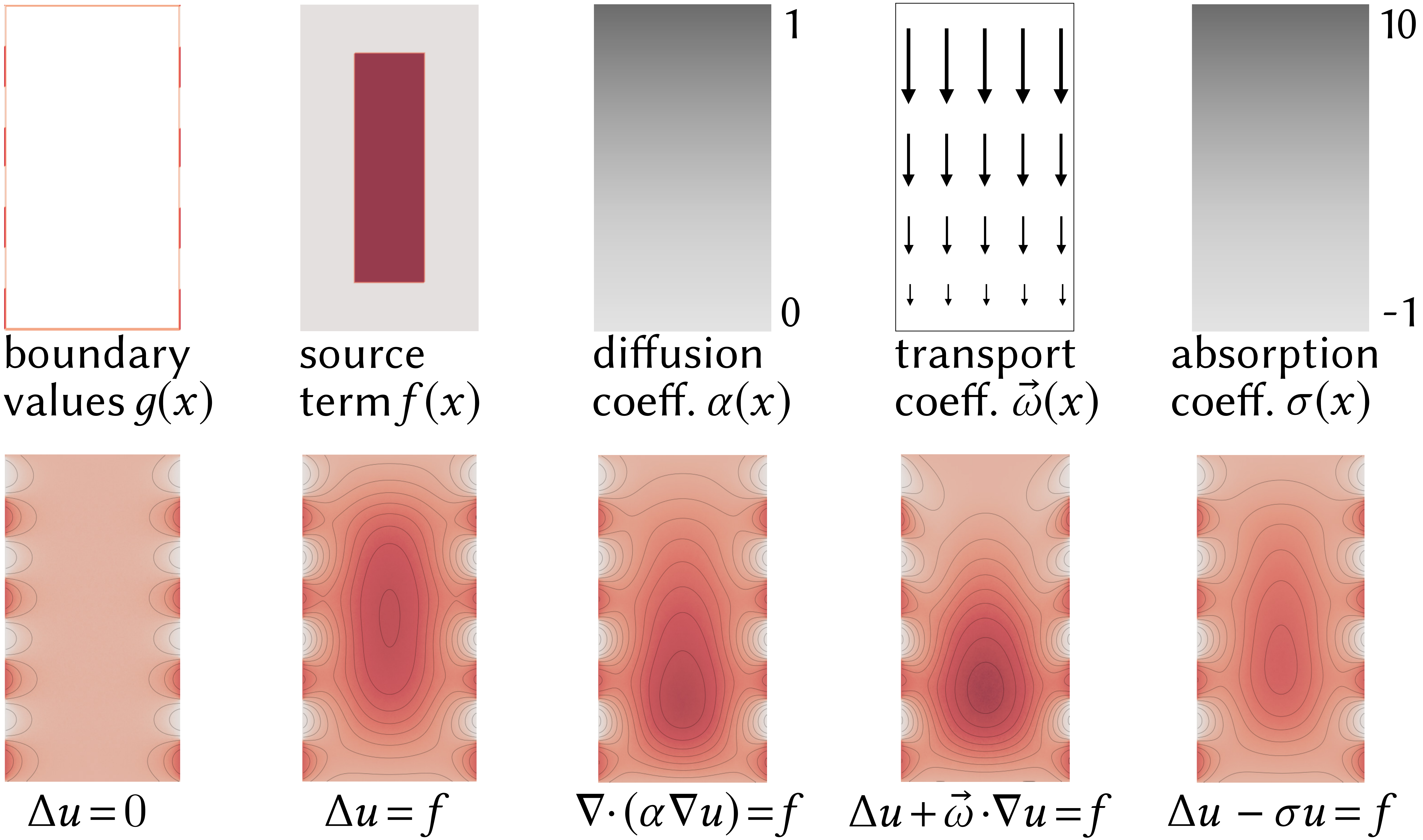
10 -1

absorption coeff. $\sigma(x)$

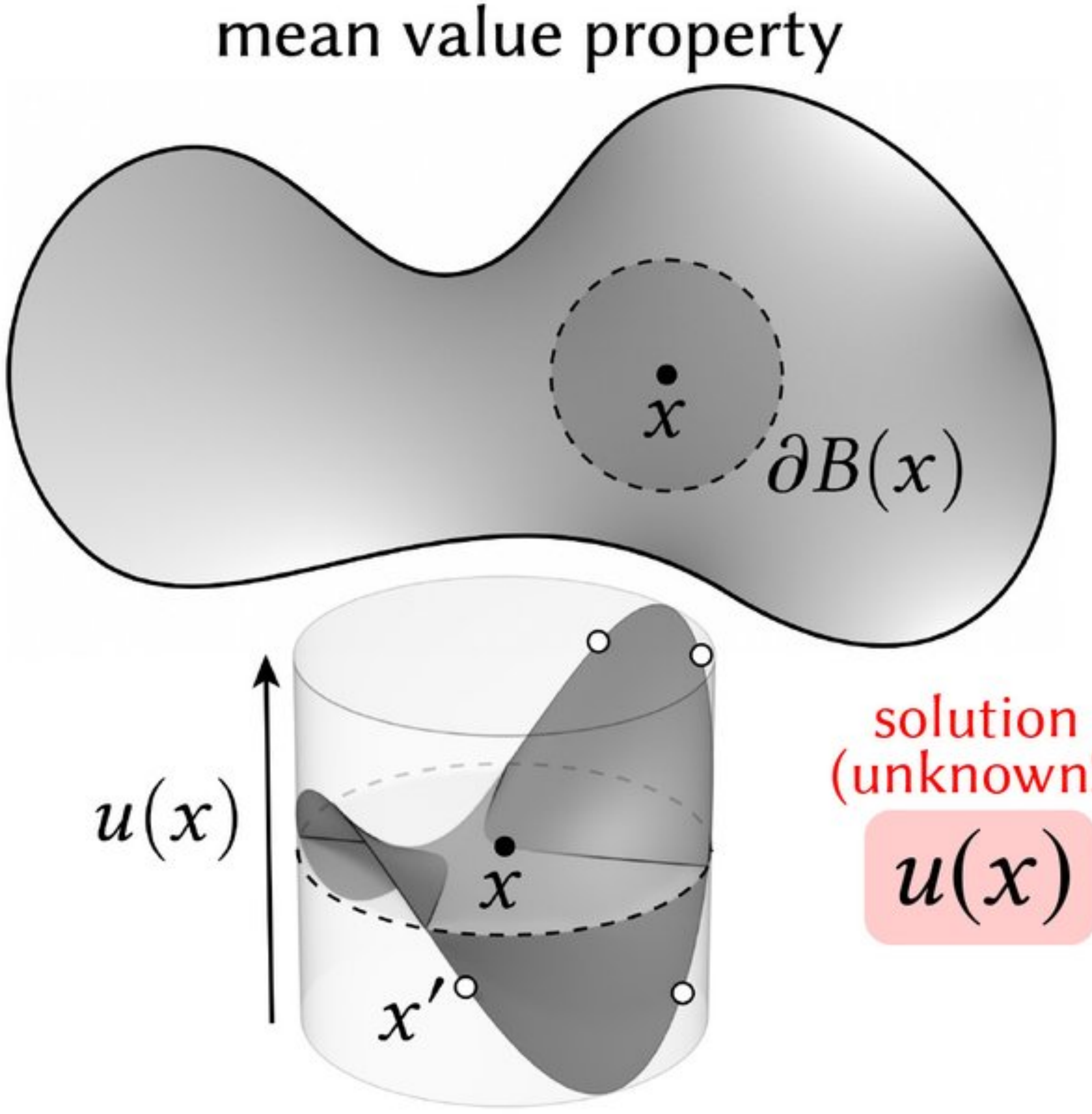


$$\Delta u - \sigma u = f$$

2nd order linear elliptic PDEs

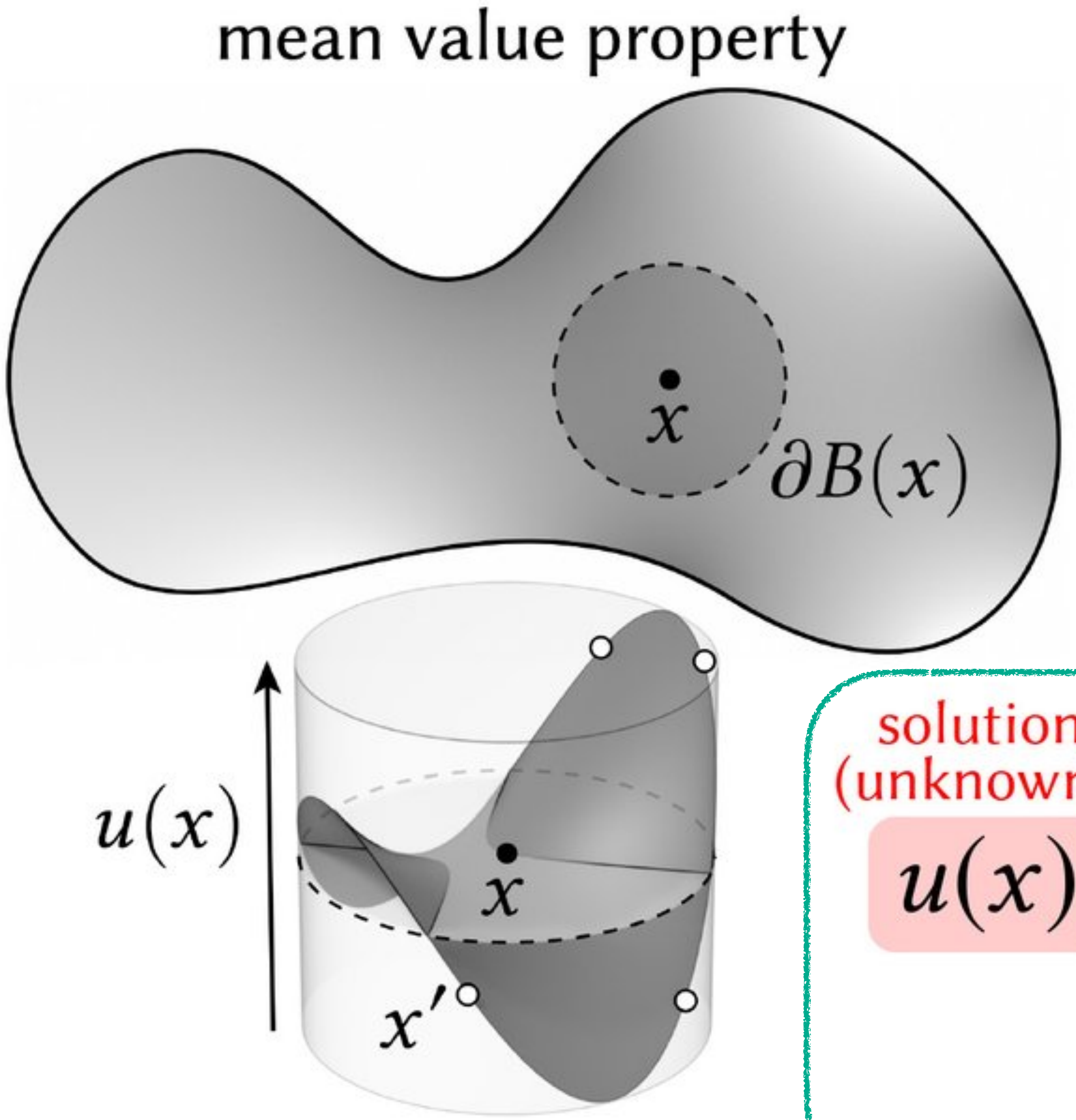


Integral for Laplace equation



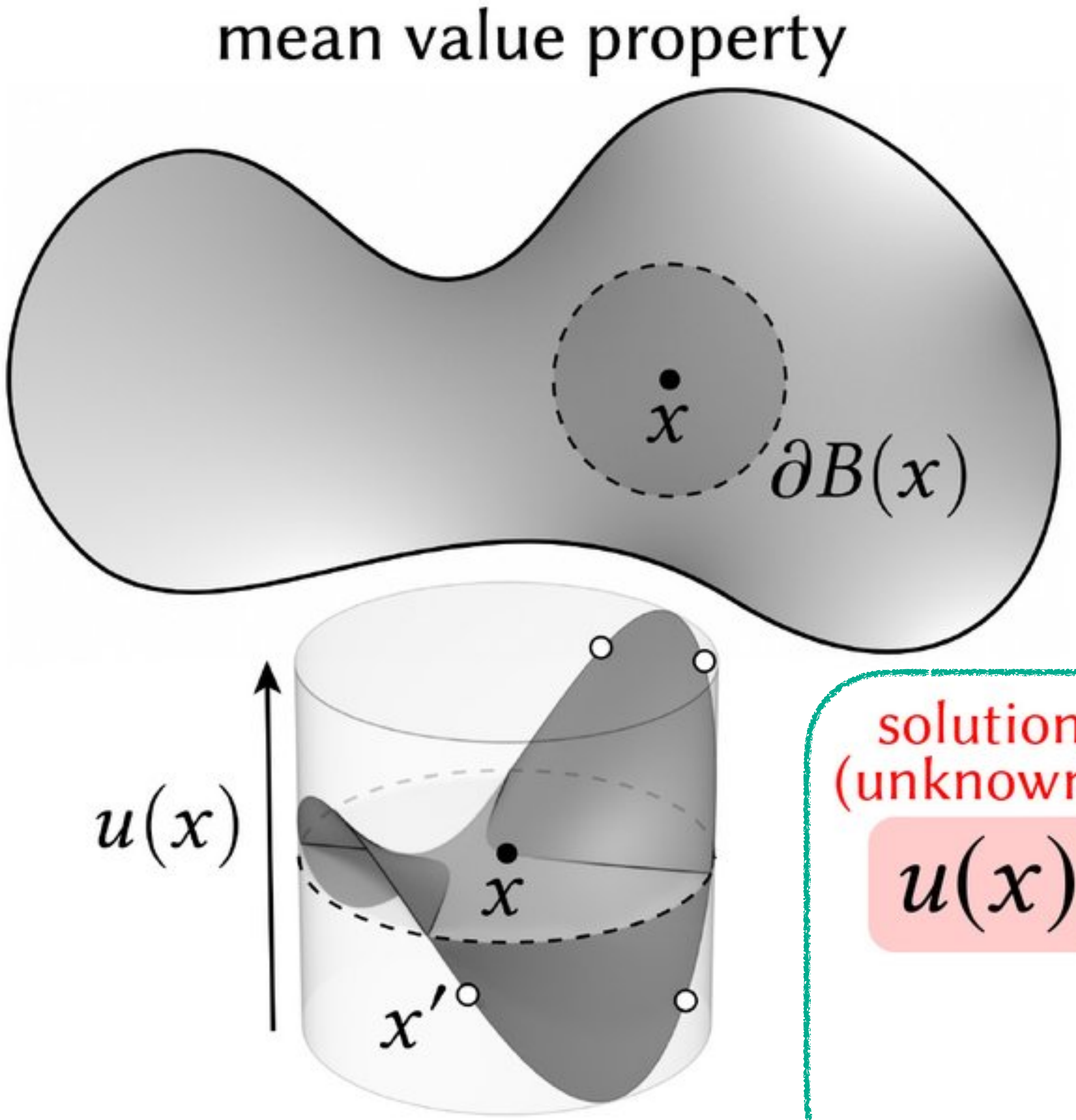
$$\overbrace{u(x)}^{\text{solution (unknown!)}} = \underbrace{\frac{1}{|\partial B(x)|}}_{\text{volume of bounding sphere}} \int_{\underbrace{\partial B(x)}_{\text{ball around point } x}} \overbrace{u(y)}^{\text{solution (unknown!)}} \, dy$$

Integral for Laplace equation



$$\underset{\text{solution (unknown!)}}{u(x)} = \frac{1}{\underbrace{|\partial B(x)|}_{\text{volume of bounding sphere}}} \int_{\underbrace{\partial B(x)}_{\text{ball around point } x}} \underset{\text{solution (unknown!)}}{u(y)}\, dy$$

Integral for Laplace equation



basis for WoS!

solution (unknown!)

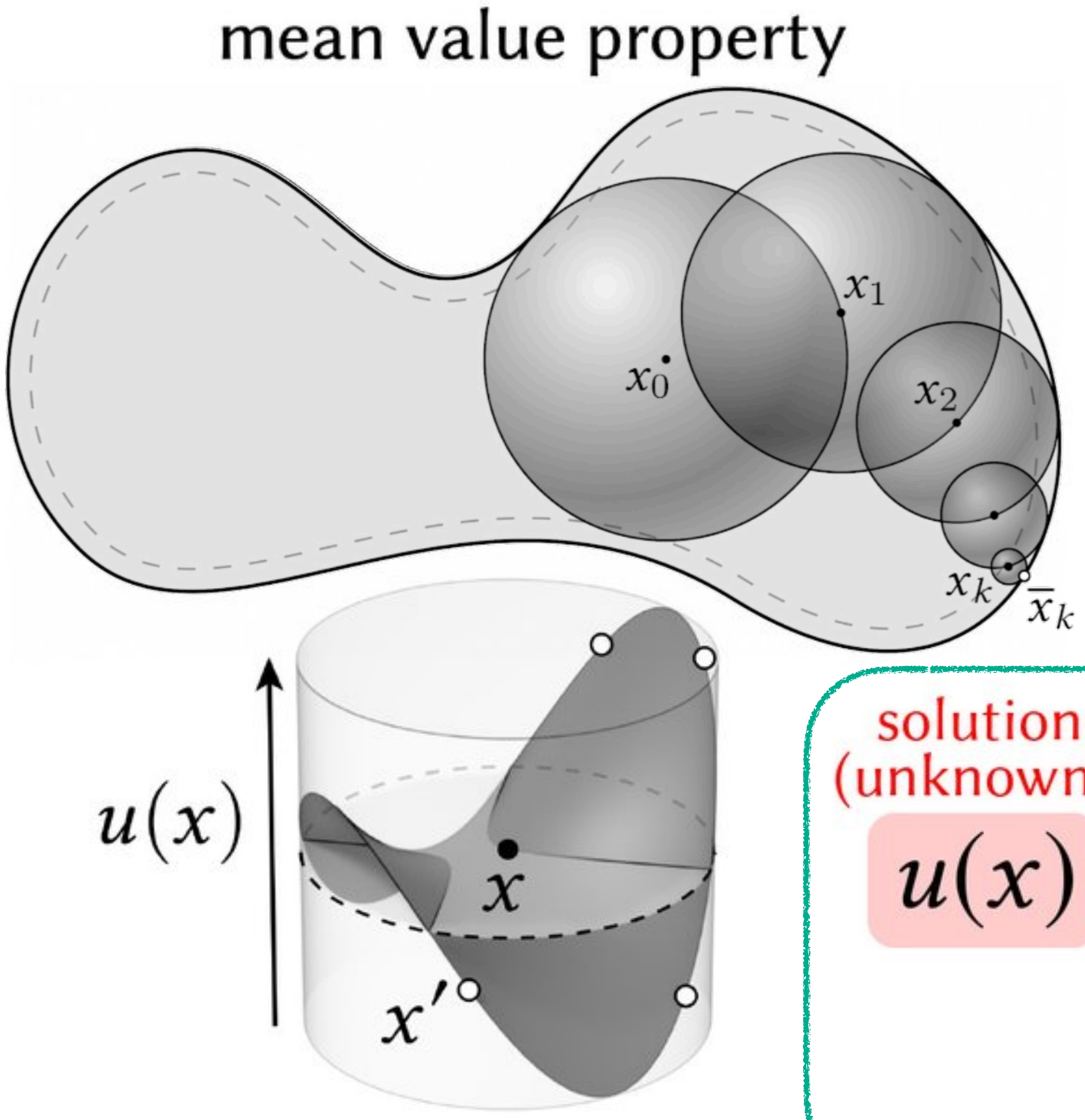
solution (unknown!)

$$u(x) = \frac{1}{|\partial B(x)|} \int_{\partial B(x)} u(y)\, dy$$

volume of bounding sphere

ball around point x

Integral for Laplace equation



basis for WoS!

$$\overset{\text{solution (unknown!)}}{\boxed{u(x)}} = \underbrace{\frac{1}{|\partial B(x)|}}_{\text{volume of bounding sphere}} \int_{\underbrace{\partial B(x)}_{\text{ball around point } x}} \overset{\text{solution (unknown!)}}{\boxed{u(y)}}\, dy$$

Integral for screened Poisson equation (constant absorption)

screened Poisson — PDE

$$\overbrace{\Delta u}^{\text{diffusion}} - \overbrace{\sigma u}^{\text{absorption}} = \overbrace{-f}^{\text{source term}} \quad \text{on} \quad \Omega$$

$$\text{solution} \ \ u = g \quad \text{on} \quad \partial\Omega$$

screened Poisson — IE

$$\underbrace{u(x)}_{\text{solution}} = \int_{\underbrace{B(c)}_{\text{ball containing } x}} \overbrace{f(y)}^{\text{source term}} \underbrace{G^{\sigma}(x,y)}_{\text{Green's function}} \, dy + \int_{\partial B(c)} \overbrace{u(z)}^{\text{solution}} \underbrace{P^{\sigma}(x,z)}_{\text{Poisson kernel}} \, dz$$

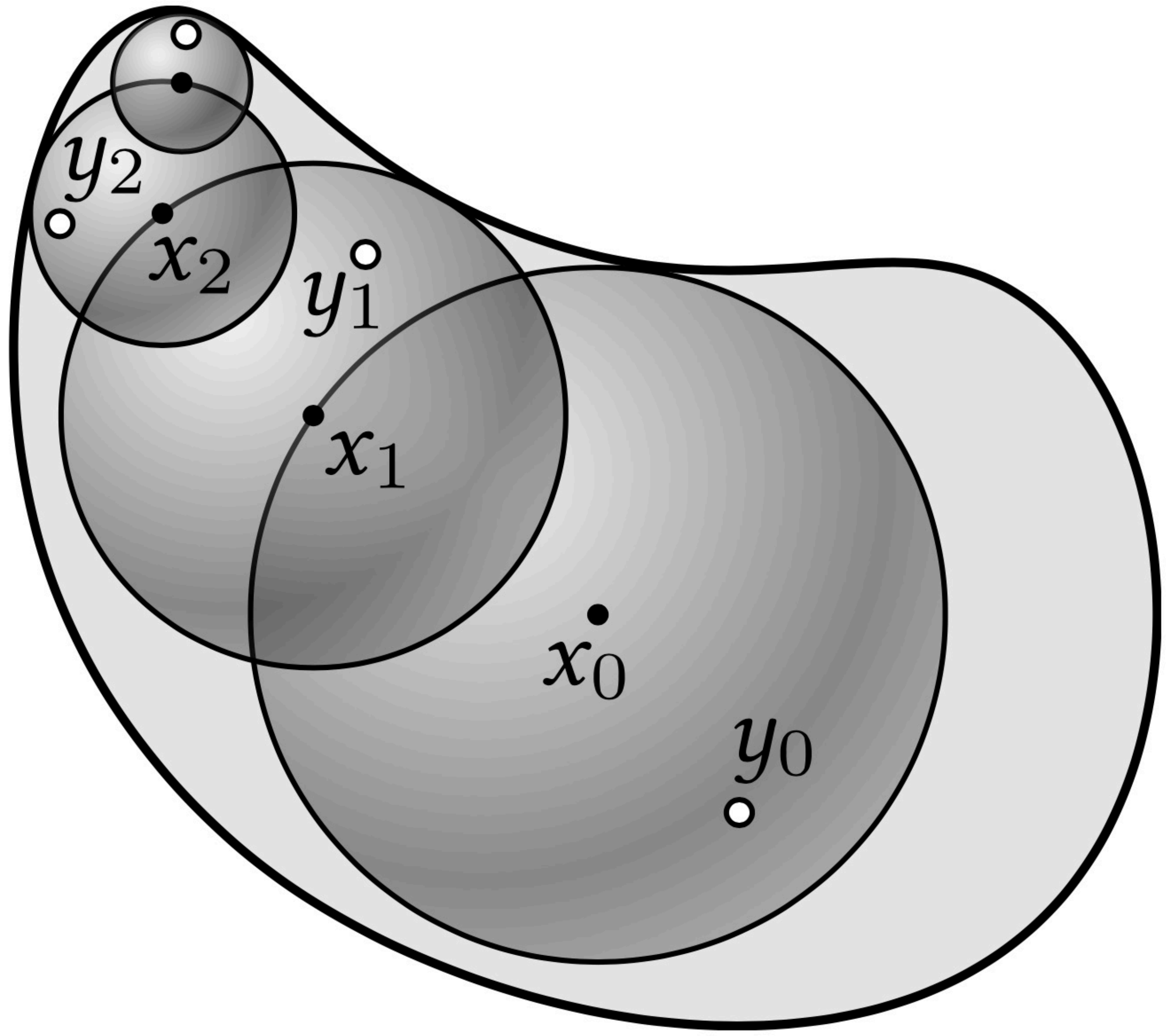
Integral for screened Poisson equation (constant absorption)

screened Poisson — PDE

diffusion absorption source term

$$\Delta u - \sigma u = -f \quad \text{on} \quad \Omega$$

$$\text{solution } u = g \quad \text{on} \quad \partial\Omega$$

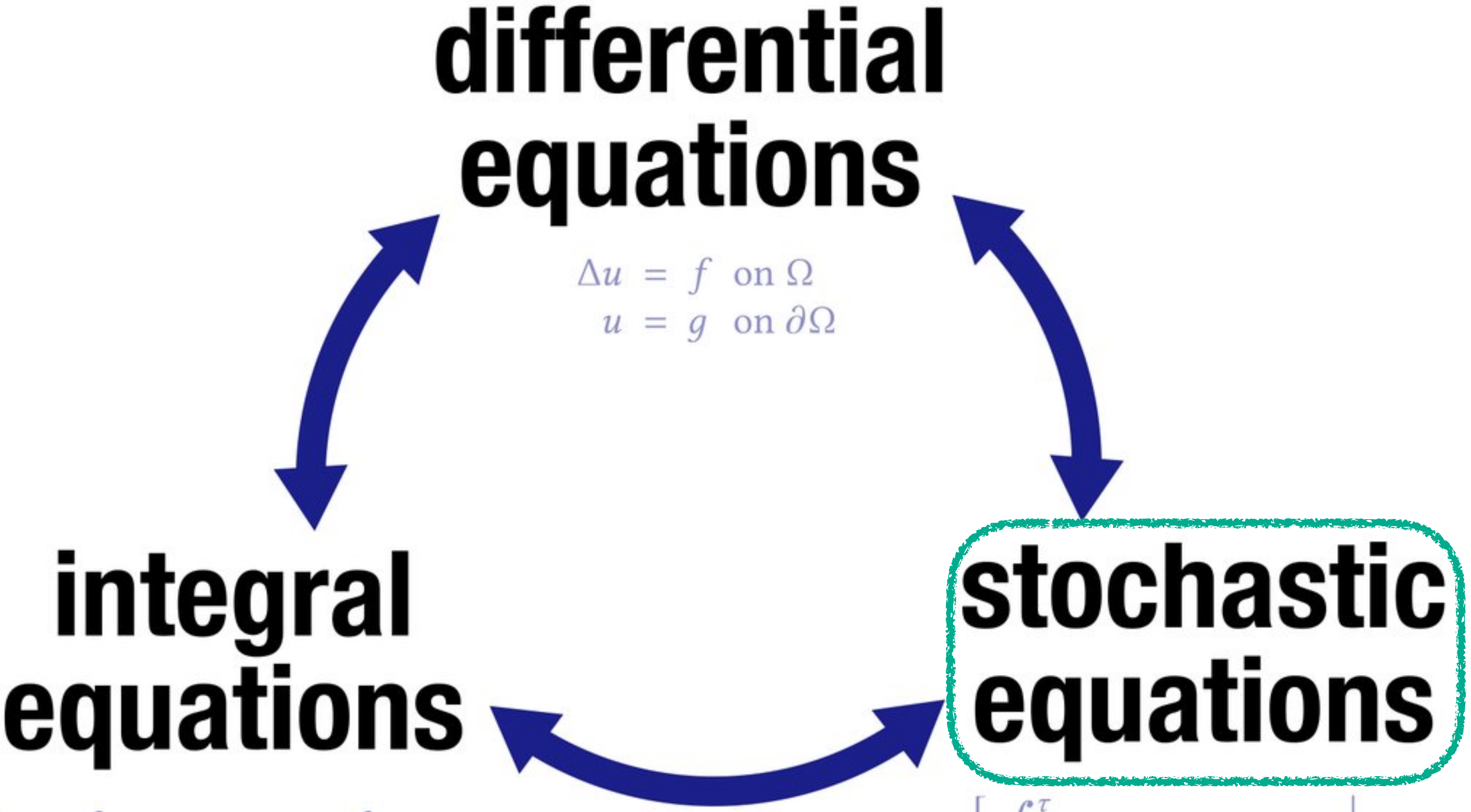


WoS: sample f inside each ball

screened Poisson — IE

$$\underbrace{u(x)}_{\text{solution}} = \int_{\underbrace{B(c)}_{\text{ball containing } x}} \overbrace{f(y)}^{\text{source term}} \underbrace{G^\sigma(x,y)}_{\text{Green's function}} \, dy + \int_{\partial B(c)} \overbrace{u(z)}^{\text{solution}} \underbrace{P^\sigma(x,z)}_{\text{Poisson kernel}} \, dz$$

A tale of three equations...

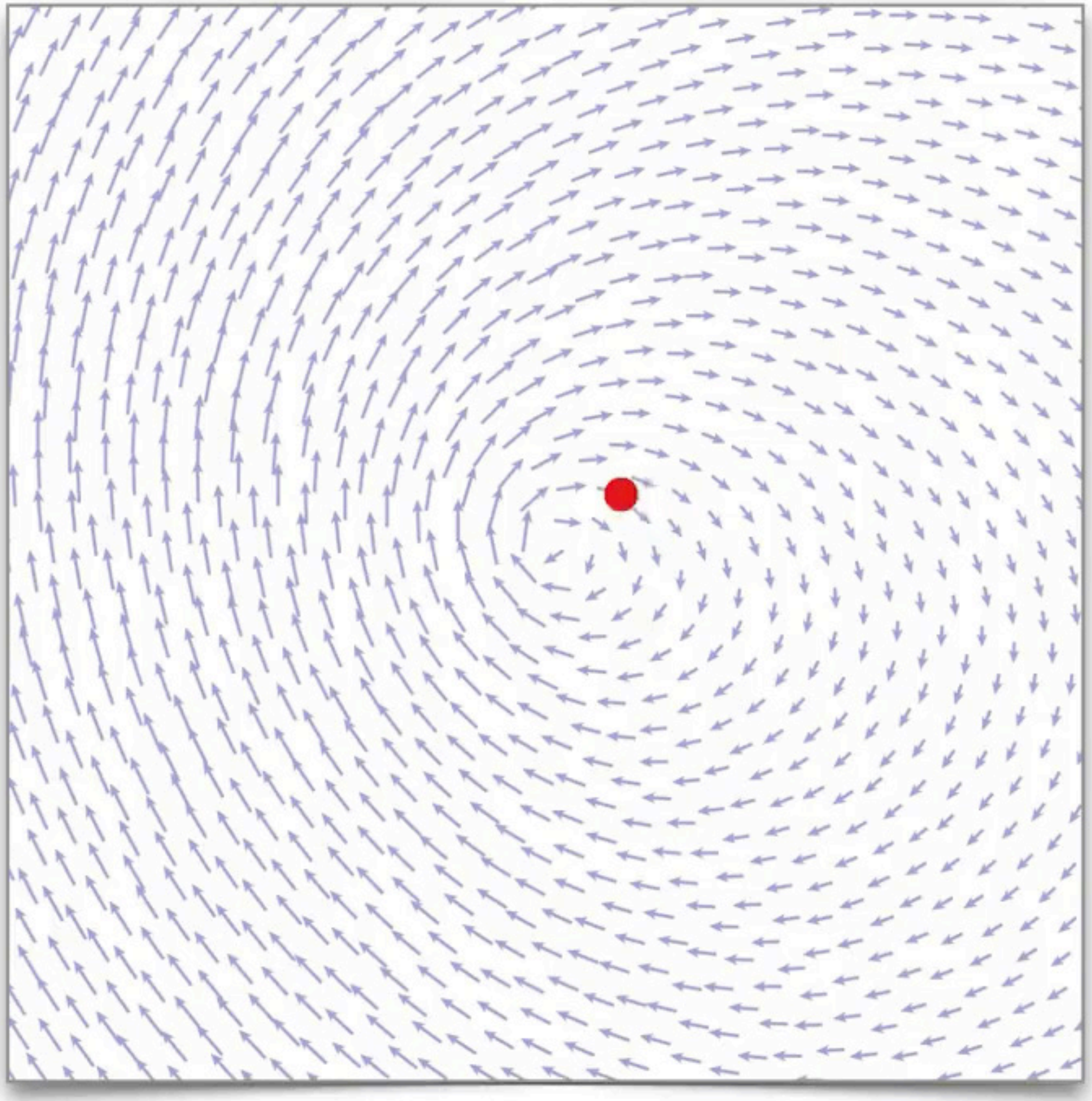


$$\begin{aligned} \Delta u &= f \quad \text{on } \Omega \\ u &= g \quad \text{on } \partial\Omega \end{aligned}$$

$$u(x) = \frac{1}{|\partial B(x)|} \int_{\partial B(x)} u(y)\, dy + \int_{B(x)} G(x,y) f(y)\, dy$$

$$u(x) = \mathbb{E}\left[\int_0^{\tau} f(W_t)\, dt + g(W_\tau) \,\middle|\, W_0 = x \right]$$

Ordinary differential equation (ODE)



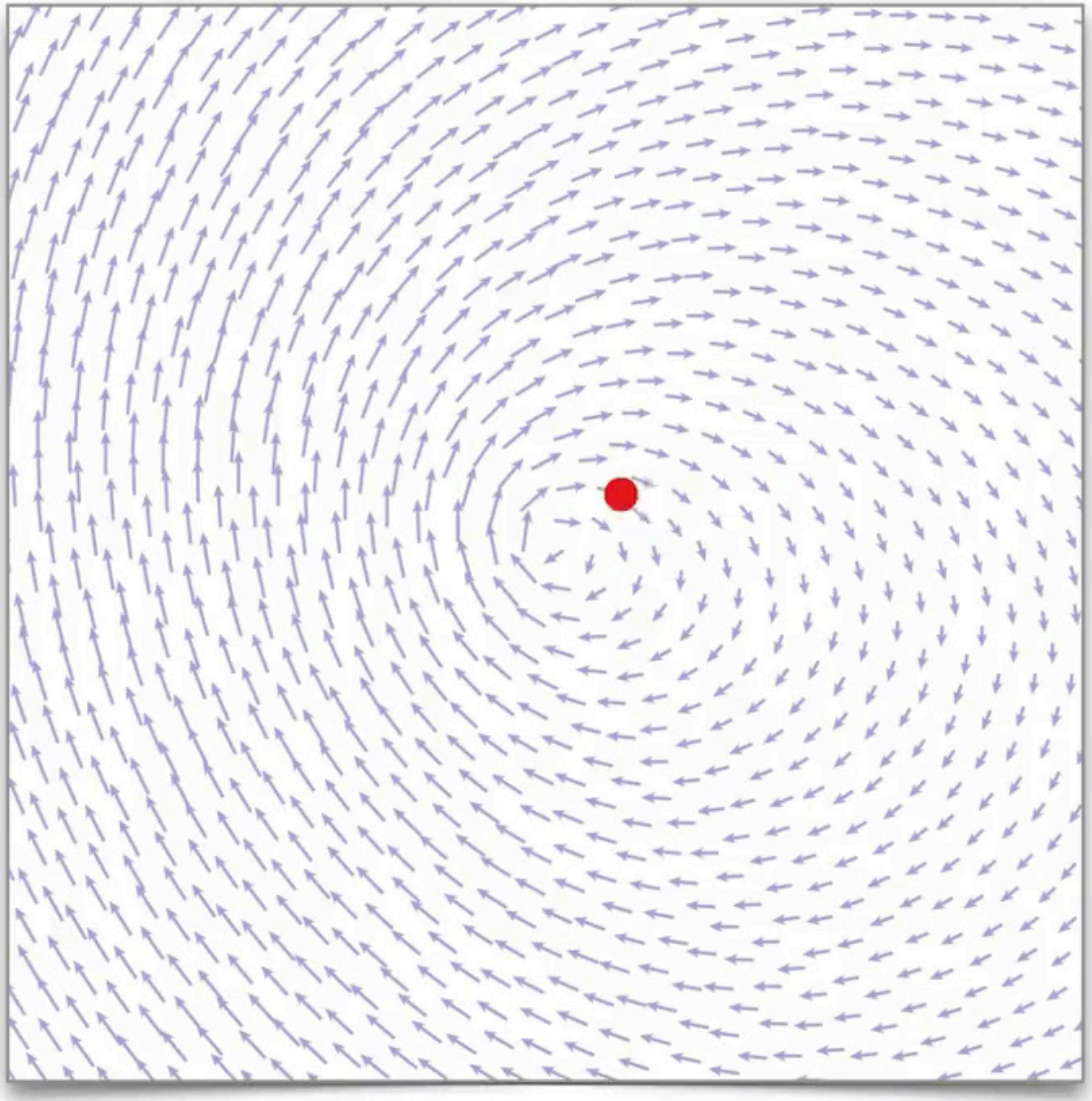
DETERMINISTIC MOTION

$$dX_t = \vec{\omega}(X_t)\,dt$$

● trajectory (X_t)

→ drift direction $(\vec{\omega})$

Ordinary differential equation (ODE)



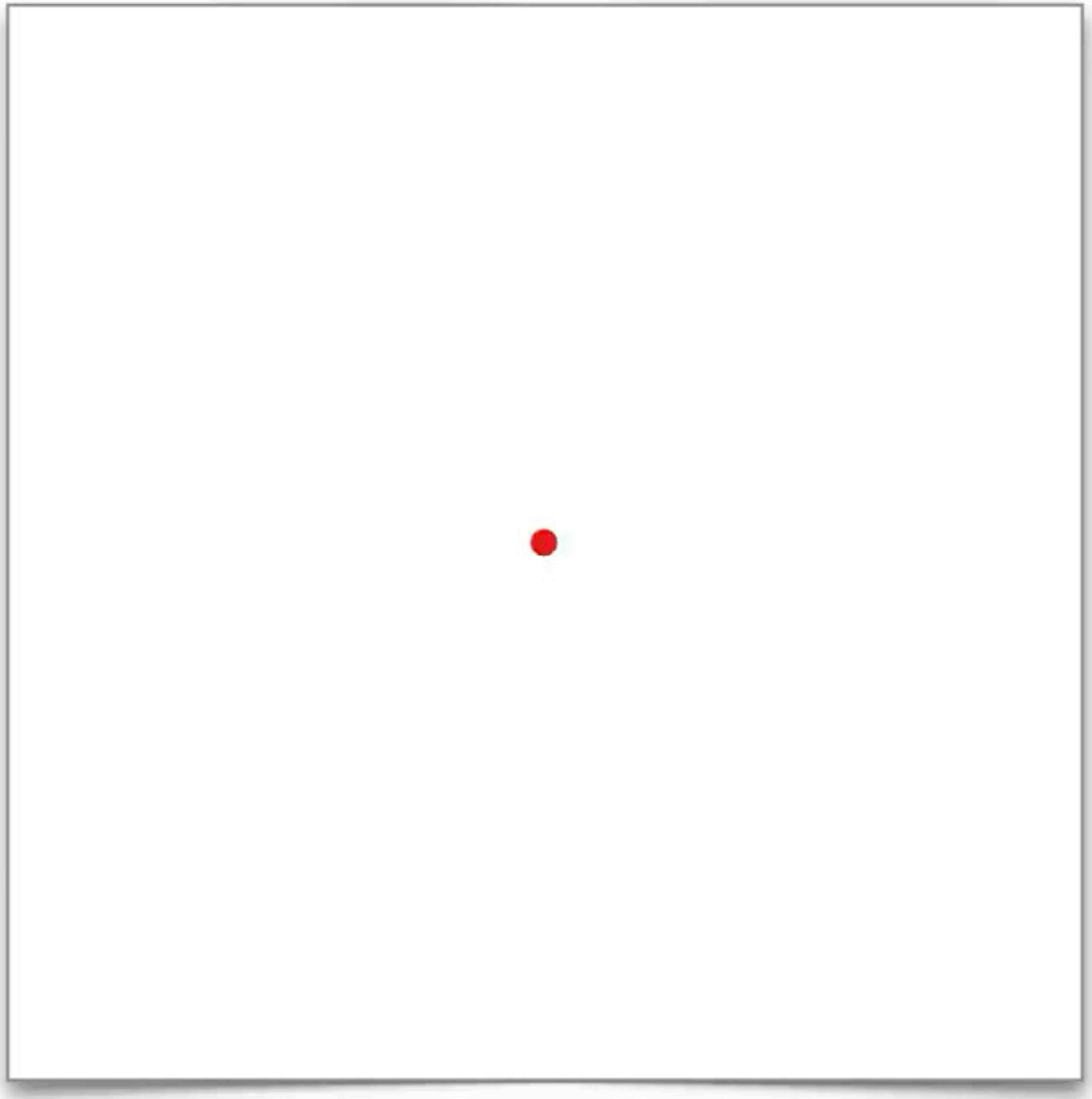
DETERMINISTIC MOTION

$$dX_t = \vec{\omega}(X_t)\,dt$$

● trajectory (X_t)

→ drift direction $(\vec{\omega})$

Stochastic differential equation (SDE)

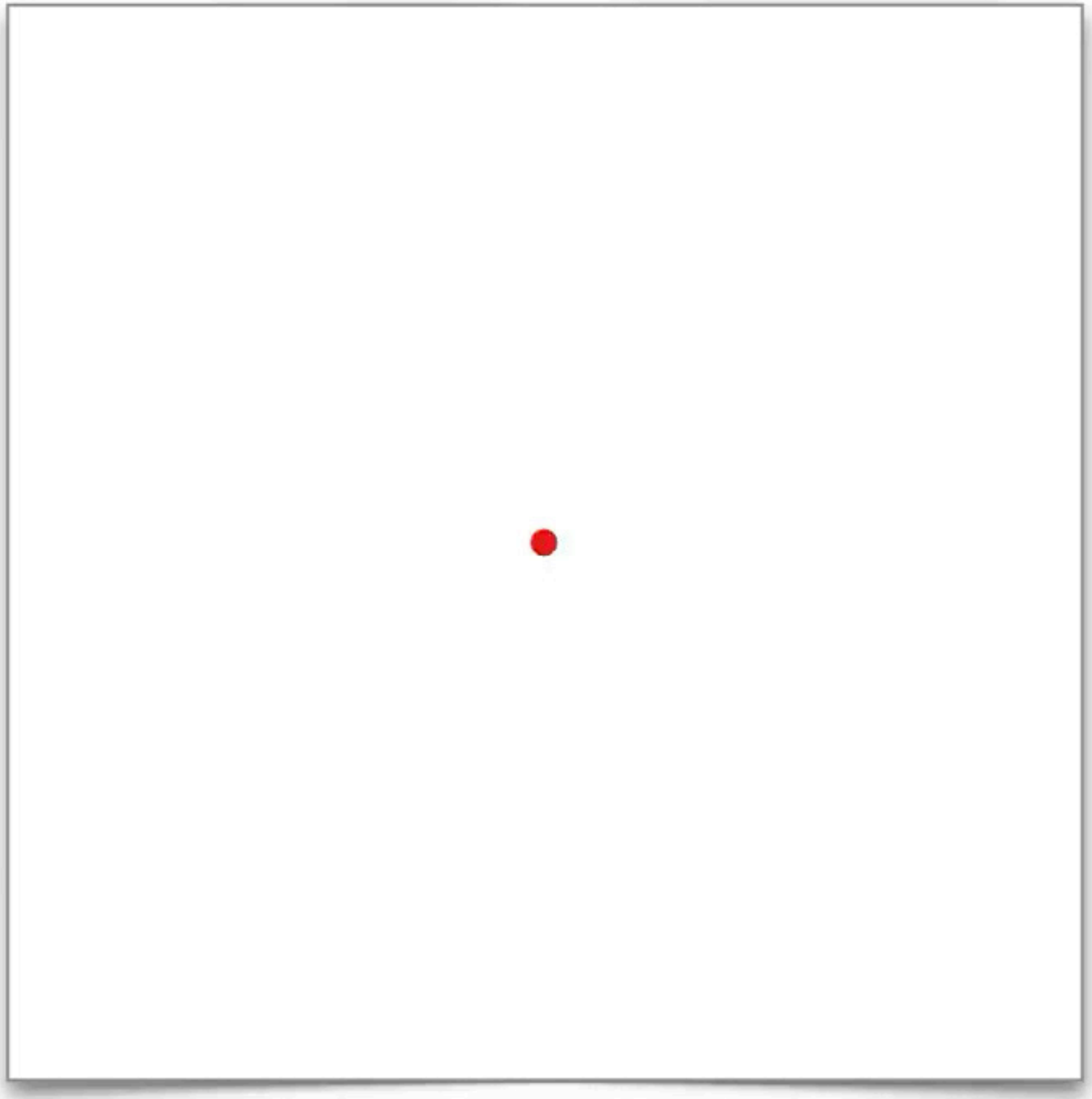


Brownian Motion

$$dX_t = dW_t$$

● trajectory (X_t)

Stochastic differential equation (SDE)

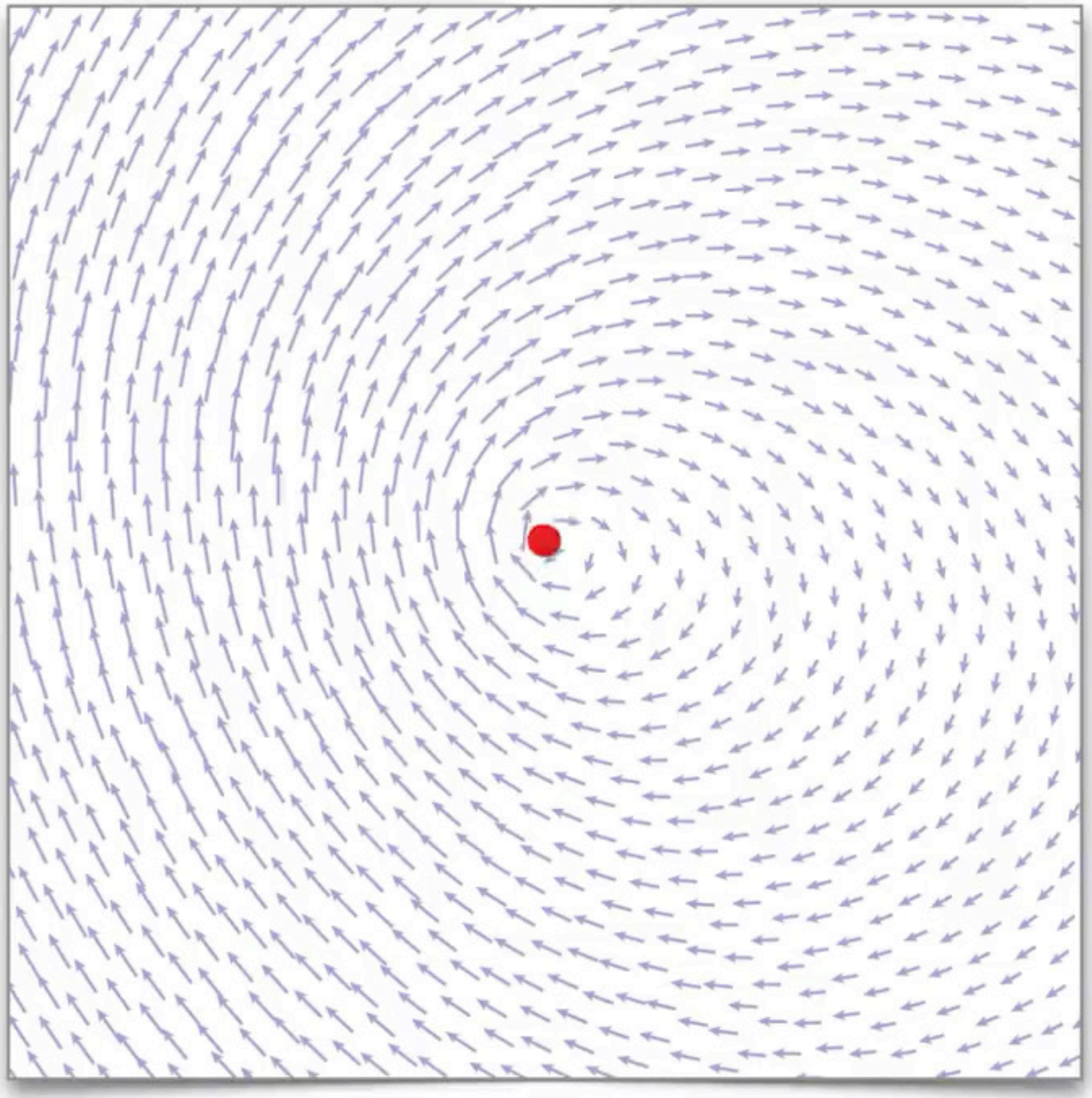


BROWNIAN MOTION

$$dX_t = dW_t$$

● trajectory (X_t)

Stochastic differential equation (SDE)



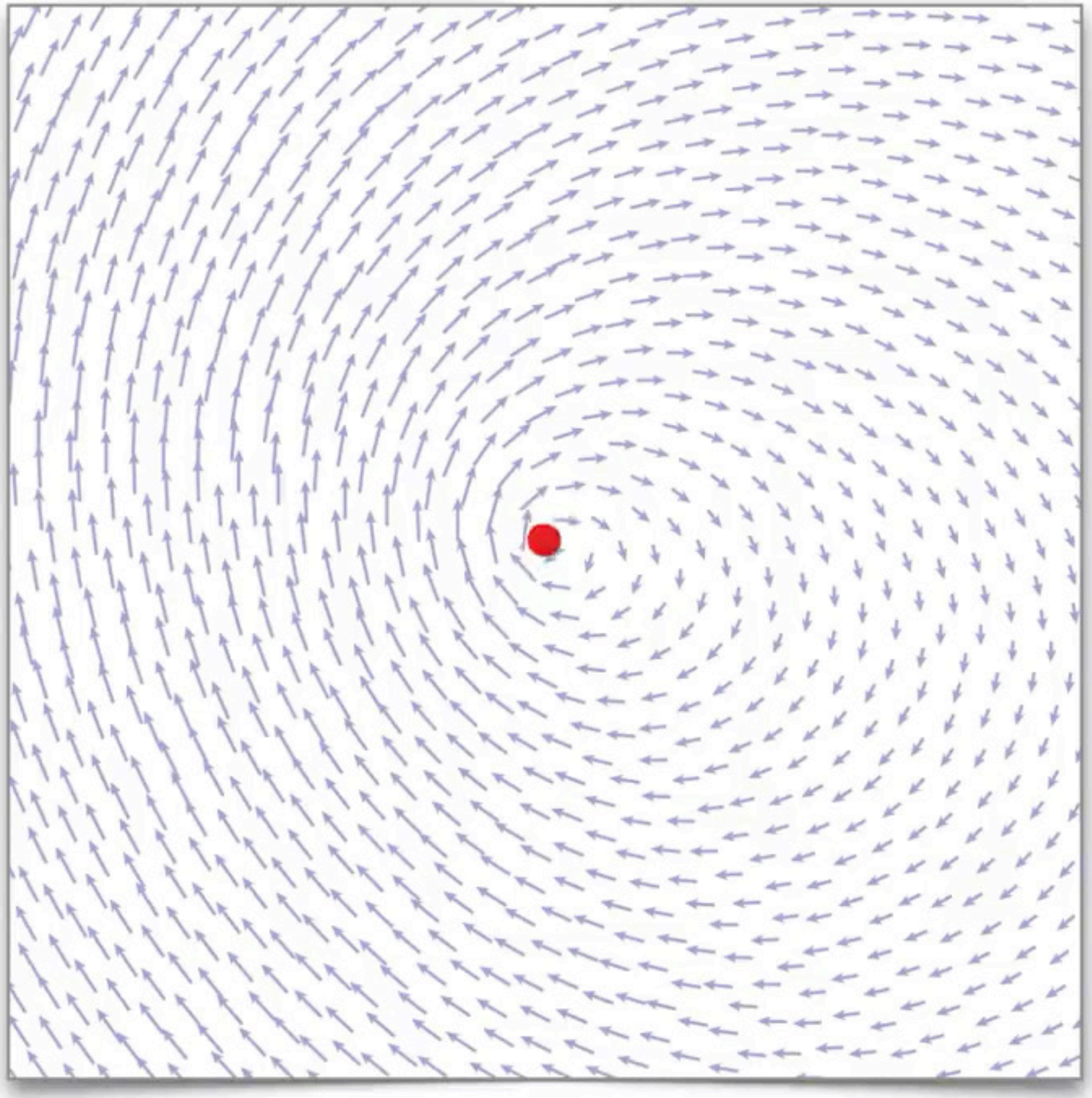
Brownian Motion with Drift

$$dX_t = \vec{\omega}(X_t)\, dt + dW_t$$

trajectory (X_t)

drift direction ($\vec{\omega}$)

Stochastic differential equation (SDE)



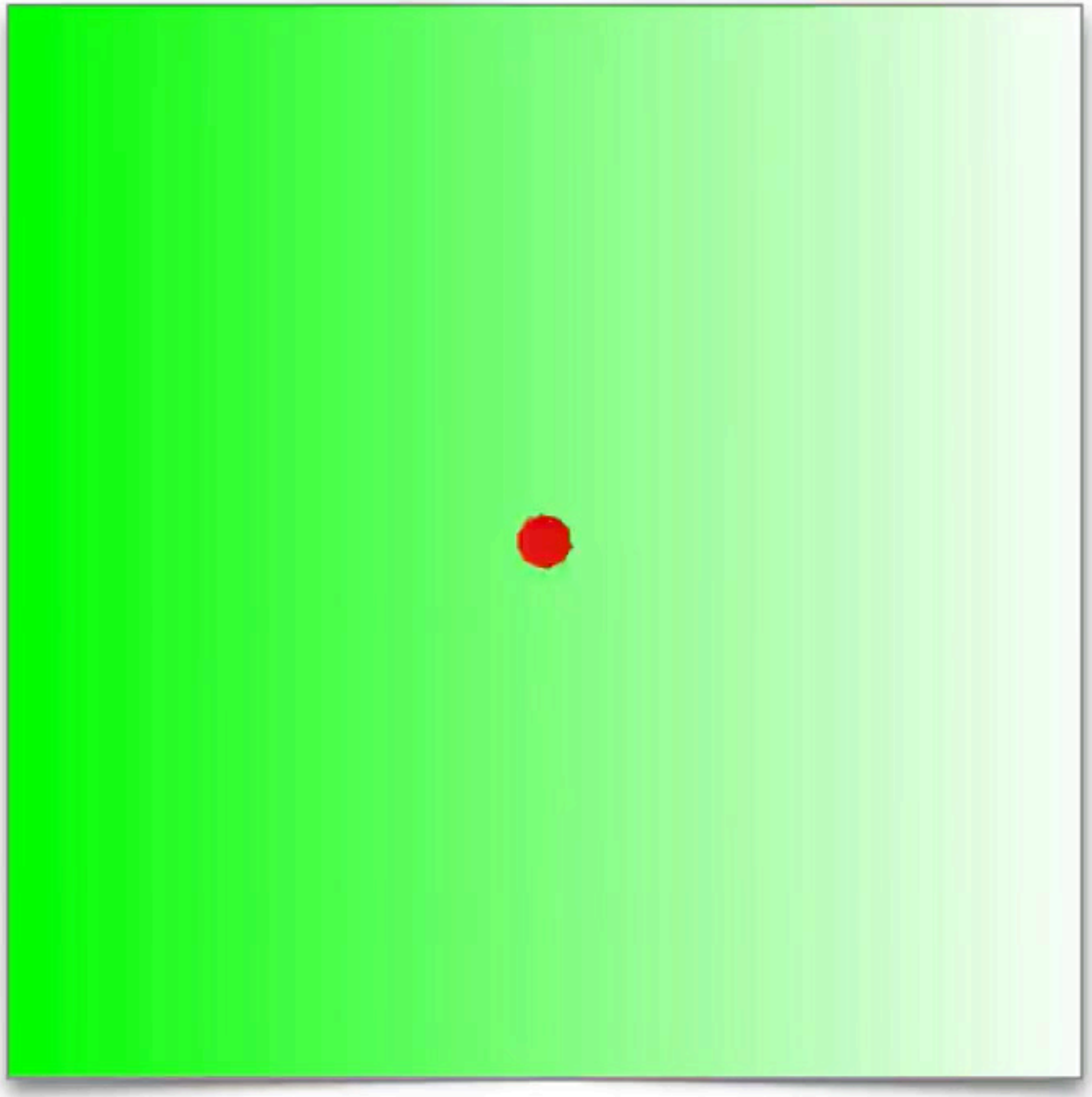
BROWNIAN MOTION WITH DRIFT

$$dX_t = \vec{\omega}(X_t)\,dt + dW_t$$

● trajectory (X_t)

→ drift direction $(\vec{\omega})$

Stochastic differential equation (SDE)



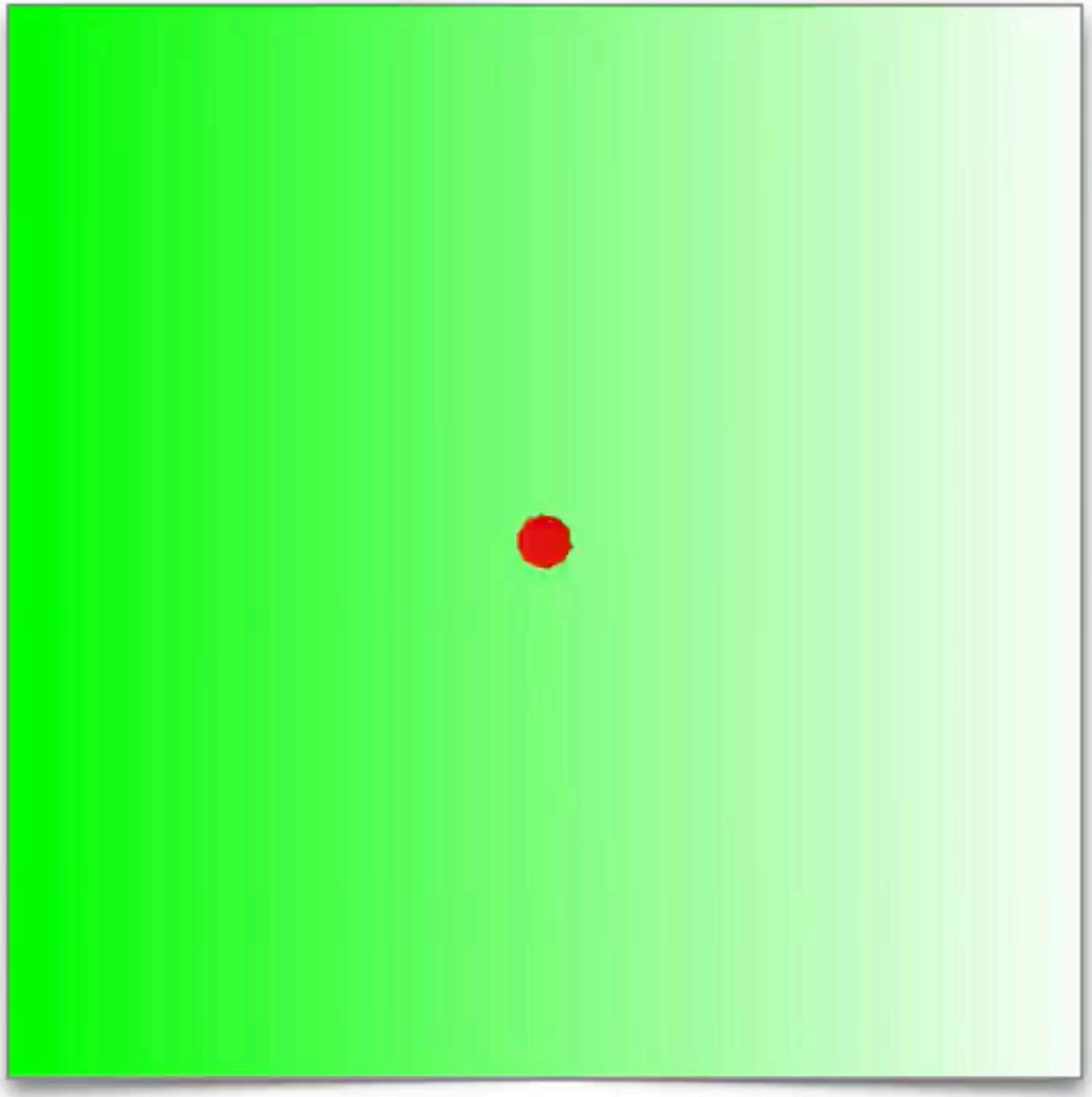
BROWNIAN MOTION WITH VARIABLE SCALE

$$dX_t = \sqrt{\alpha(X_t)}\, dW_t$$

● trajectory (X_t)

diffusivity (α)

Stochastic differential equation (SDE)



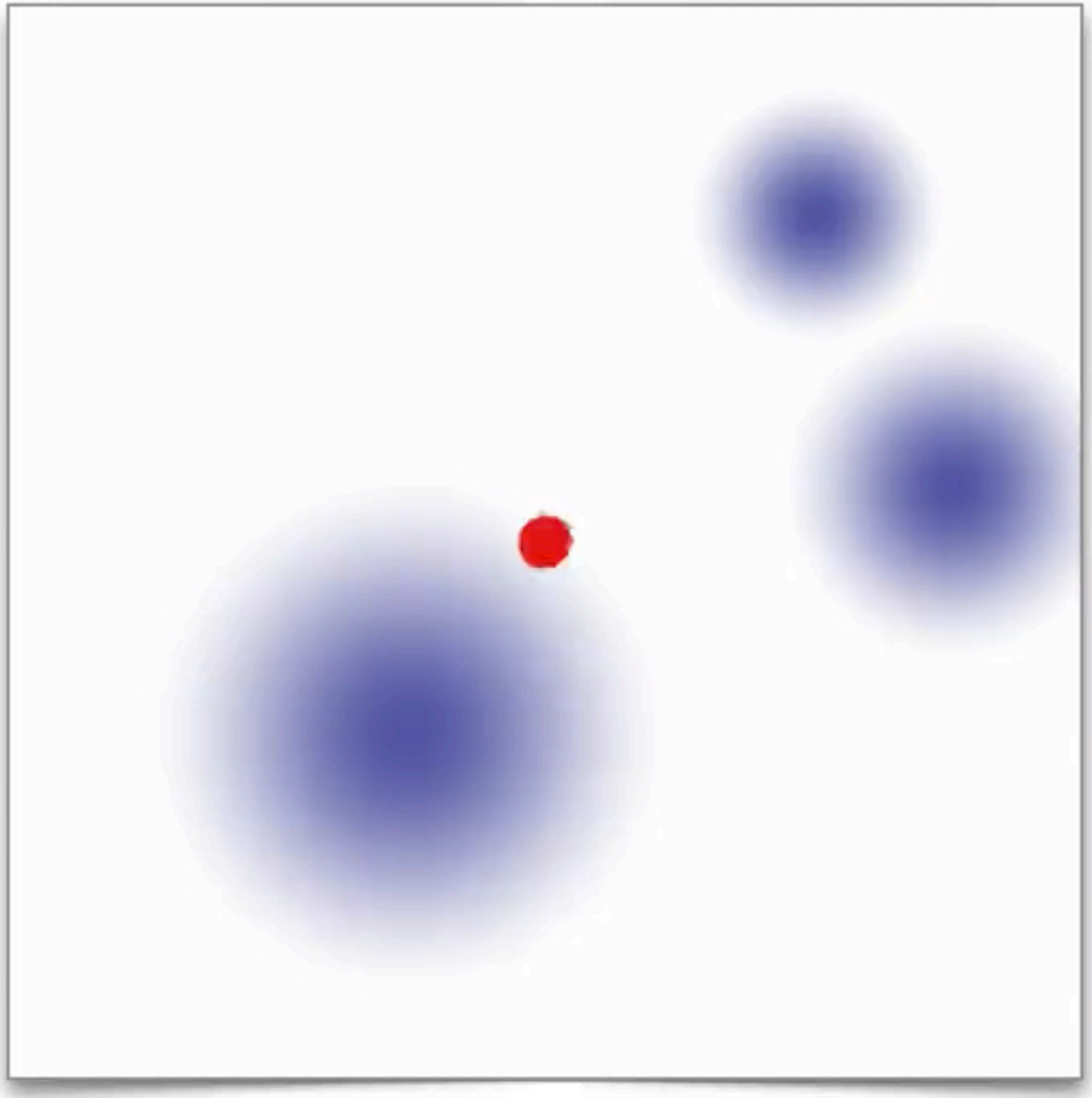
BROWNIAN MOTION WITH VARIABLE SCALE

$$dX_t = \sqrt{\alpha(X_t)}\, dW_t$$

trajectory (X_t)

diffusivity (α)

Stochastic differential equation (SDE)



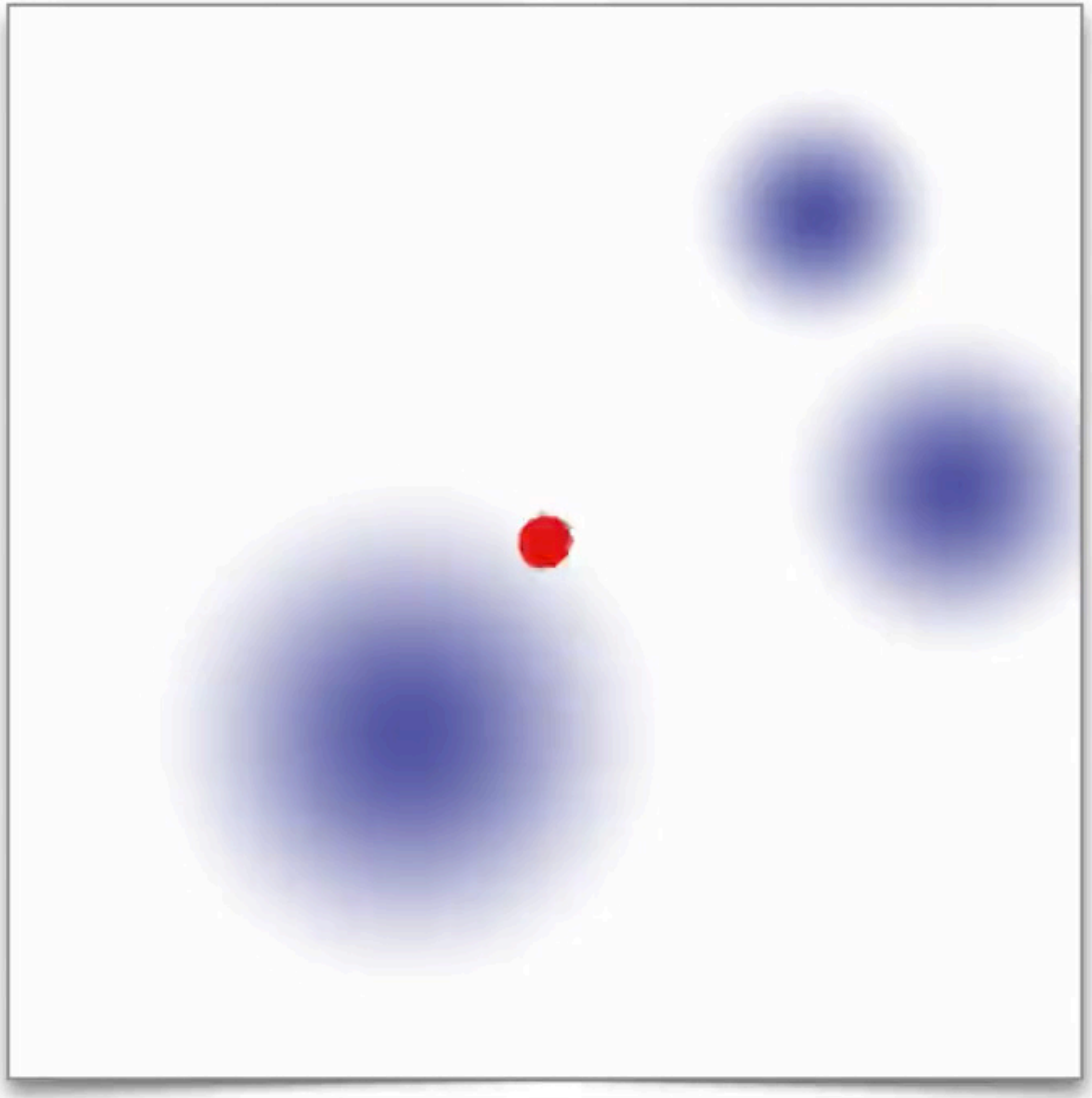
BROWNIAN MOTION IN ABSORBING MEDIUM

$$dX_t = dW_t$$

trajectory (X_t)

absorption (σ)

Stochastic differential equation (SDE)



BROWNIAN MOTION IN ABSORBING MEDIUM

$$dX_t = dW_t$$

trajectory (X_t)

absorption (σ)

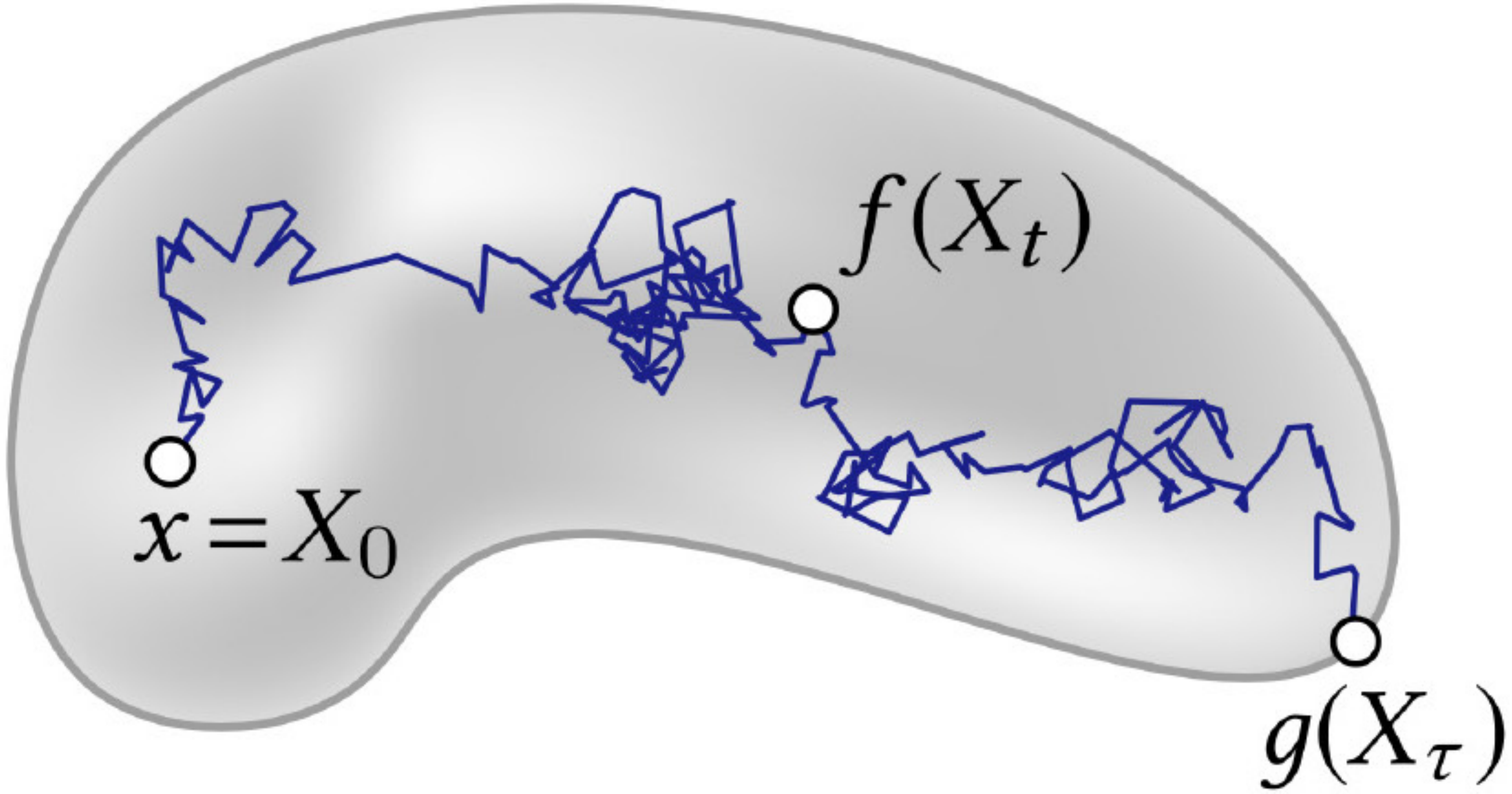
Feynman-Kac formula

time to reach boundary

absorption

random walk

$$\underbrace{u(x)}_{\text{solution}} = \mathbb{E}\left[\int_0^{\tau} e^{-\int_0^t \sigma(X_s)ds} \underbrace{f}_{\text{source}}(X_t)\, dt + e^{-\int_0^{\tau} \sigma(X_t)dt}\, \underbrace{g}_{\text{boundary values}}(X_\tau)\right]$$

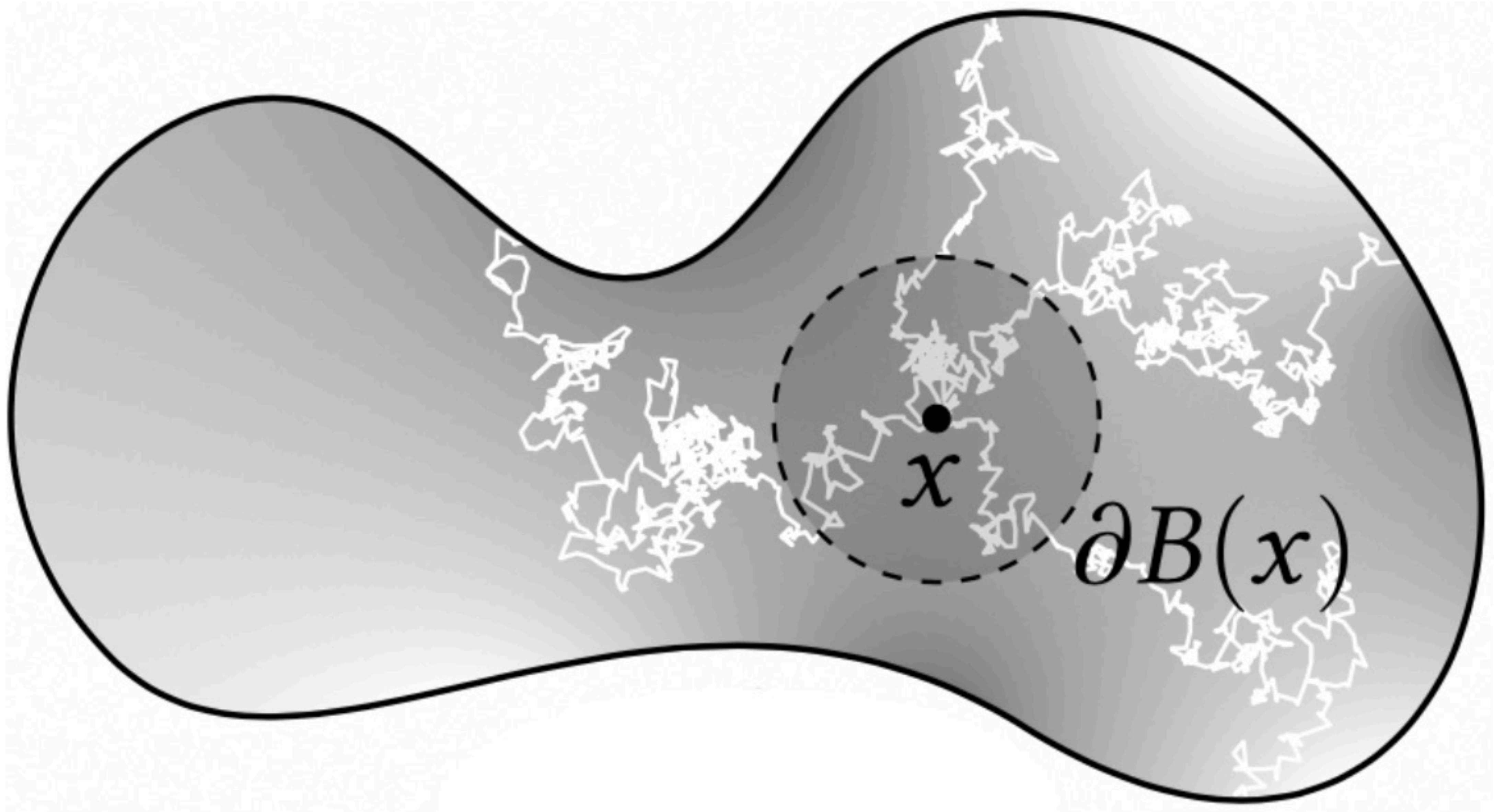


random walk

Brownian motion

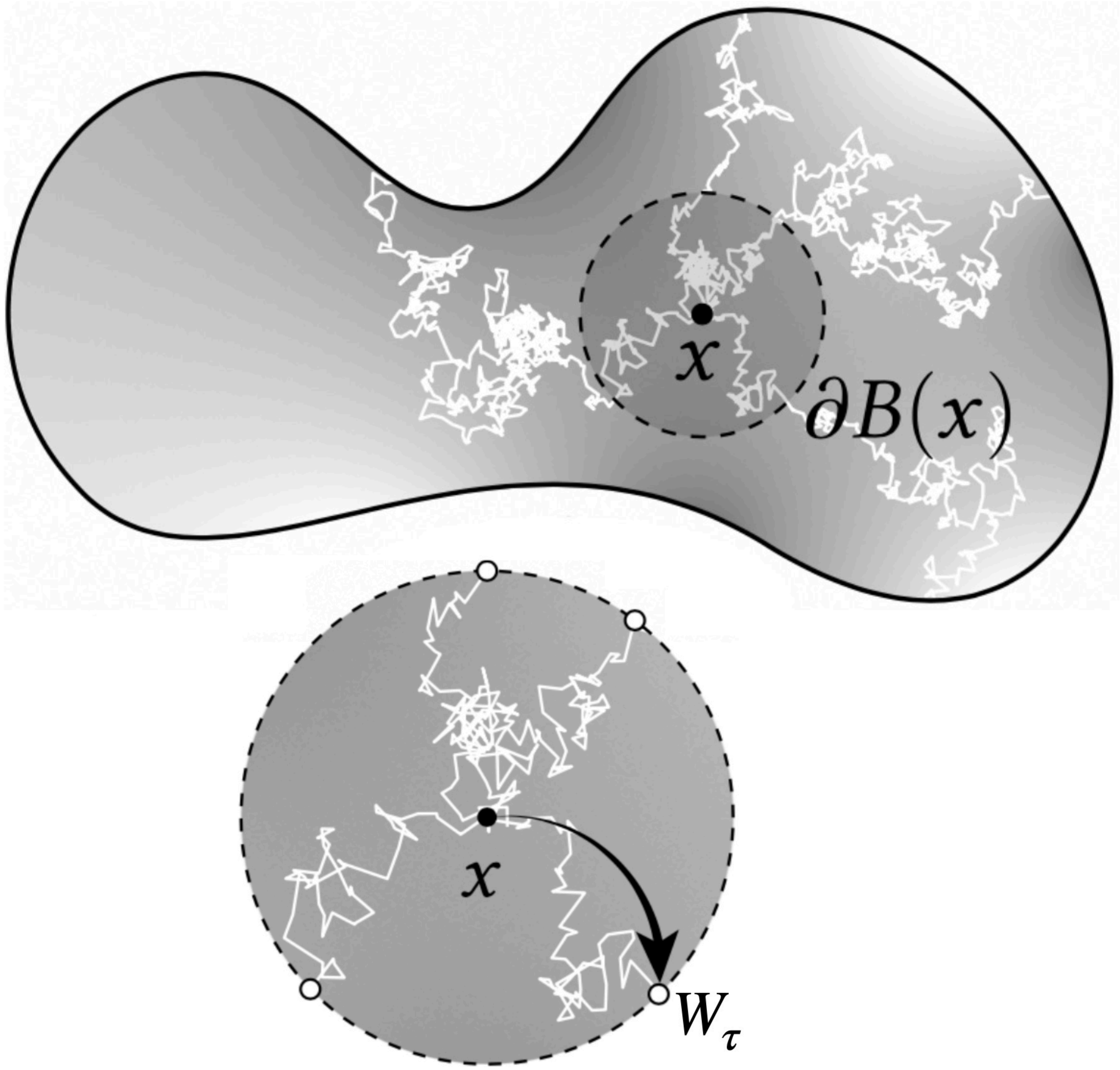
$$dX_t = \underbrace{\vec{\omega}(X_t)}_{\text{velocity}}\, dt + \underbrace{\sqrt{\alpha(X_t)}}_{\text{diffusion rate}}\, dW_t$$

Special case: Kakutani's principle



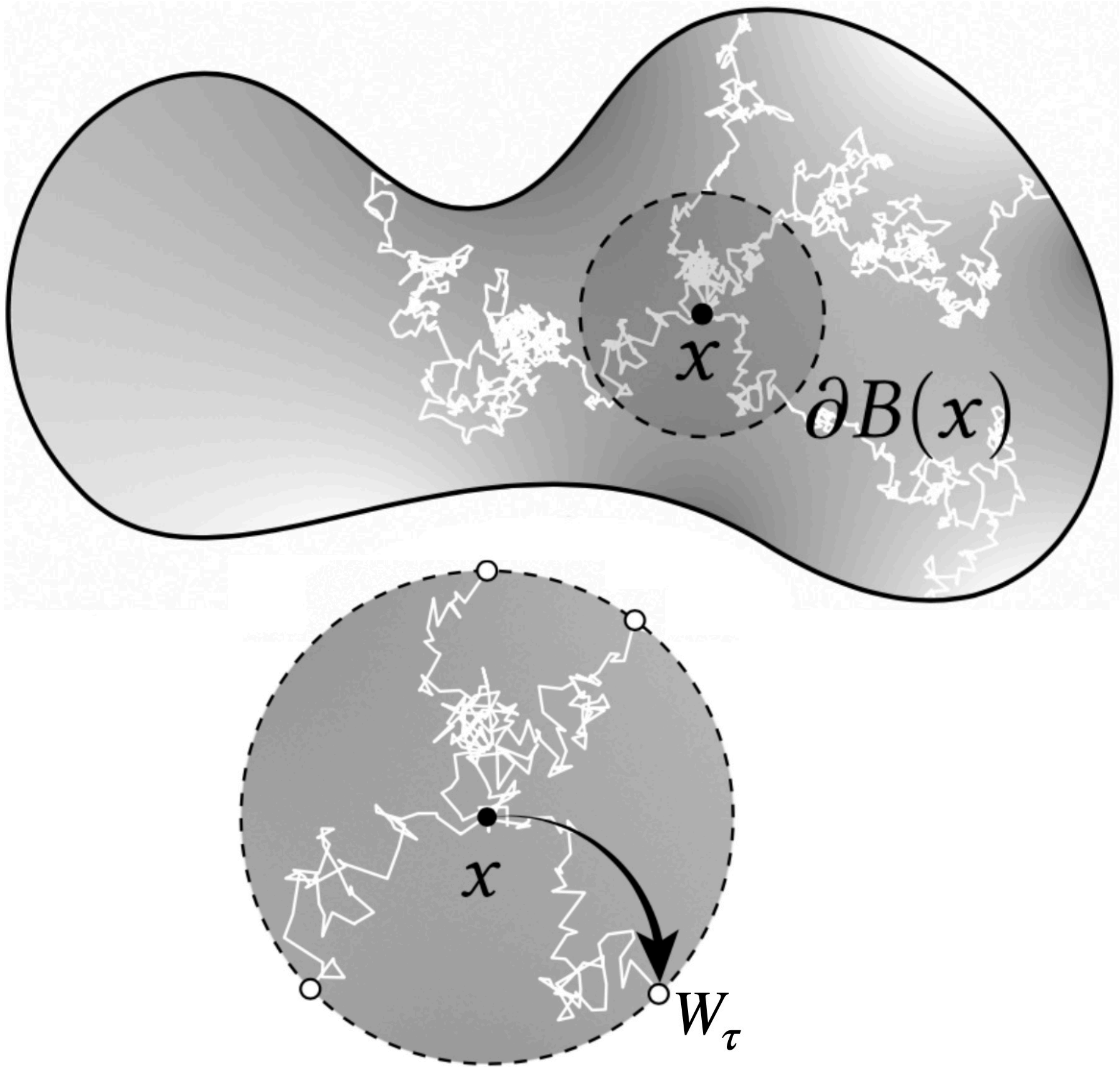
$$u(x) = \mathbb{E}[u(W_\tau)]$$

Special case: Kakutani's principle



$$u(x) = \mathbb{E}[u(W_\tau)]$$

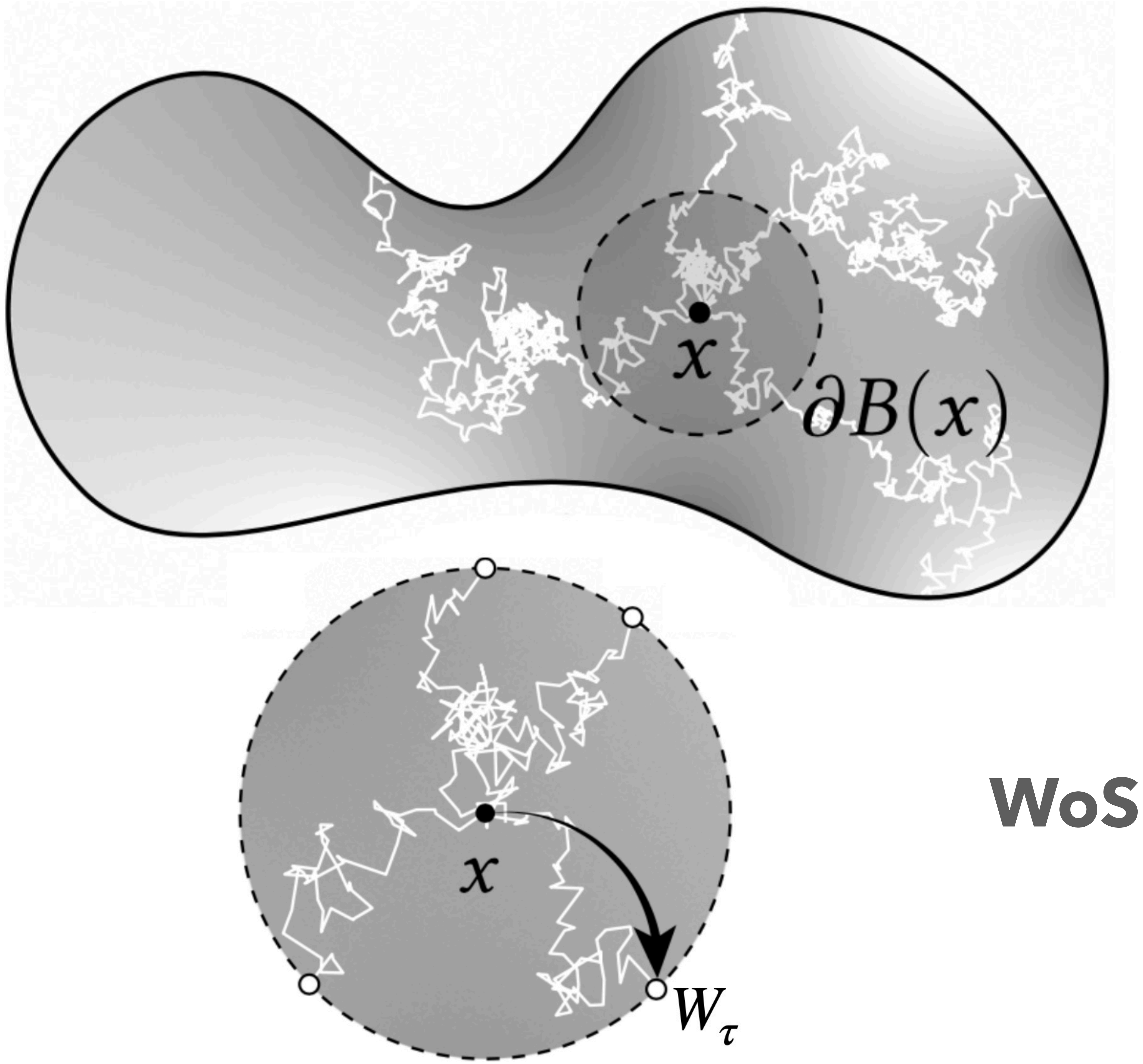
Special case: Kakutani's principle



$$u(x) = \mathbb{E}[u(W_\tau)]$$

$$= \frac{1}{|\partial B(x)|} \int_{\partial B(x)} u(y)\ dy$$

Special case: Kakutani's principle



$$u(x) = \mathbb{E}[u(W_\tau)]$$

$$= \frac{1}{|\partial B(x)|} \int_{\partial B(x)} u(y)\ dy$$

WoS simulates Brownian motion efficiently!

Feynman-Kac formula

time to reach boundary

absorption

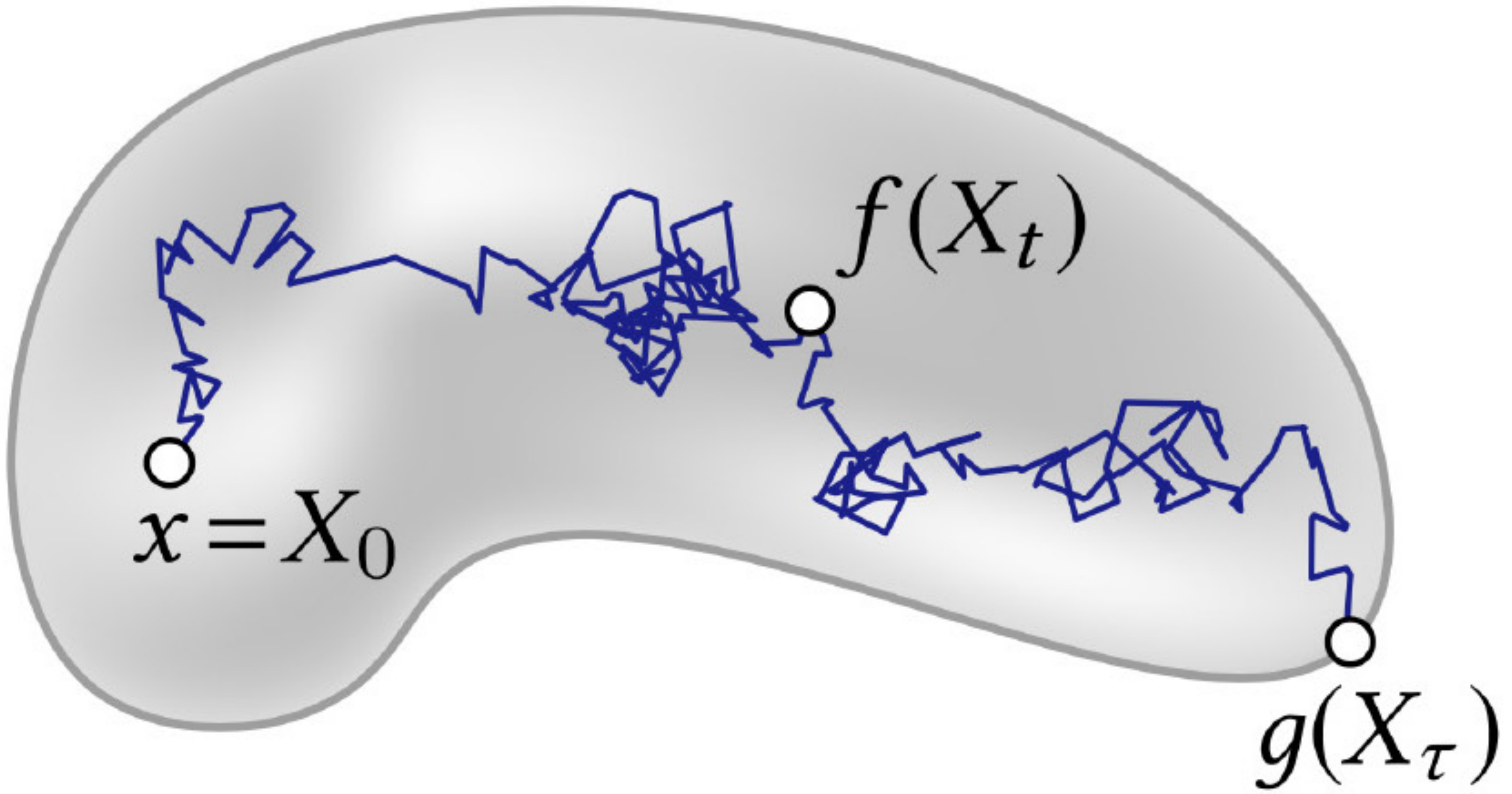
random walk

$$u(x) = \mathbb{E}\left[\int_0^{\tau} e^{-\int_0^t \sigma(X_s)ds} f(X_t)\, dt + e^{-\int_0^{\tau} \sigma(X_t)dt} g(X_\tau)\right]$$

solution

source

boundary values



random walk

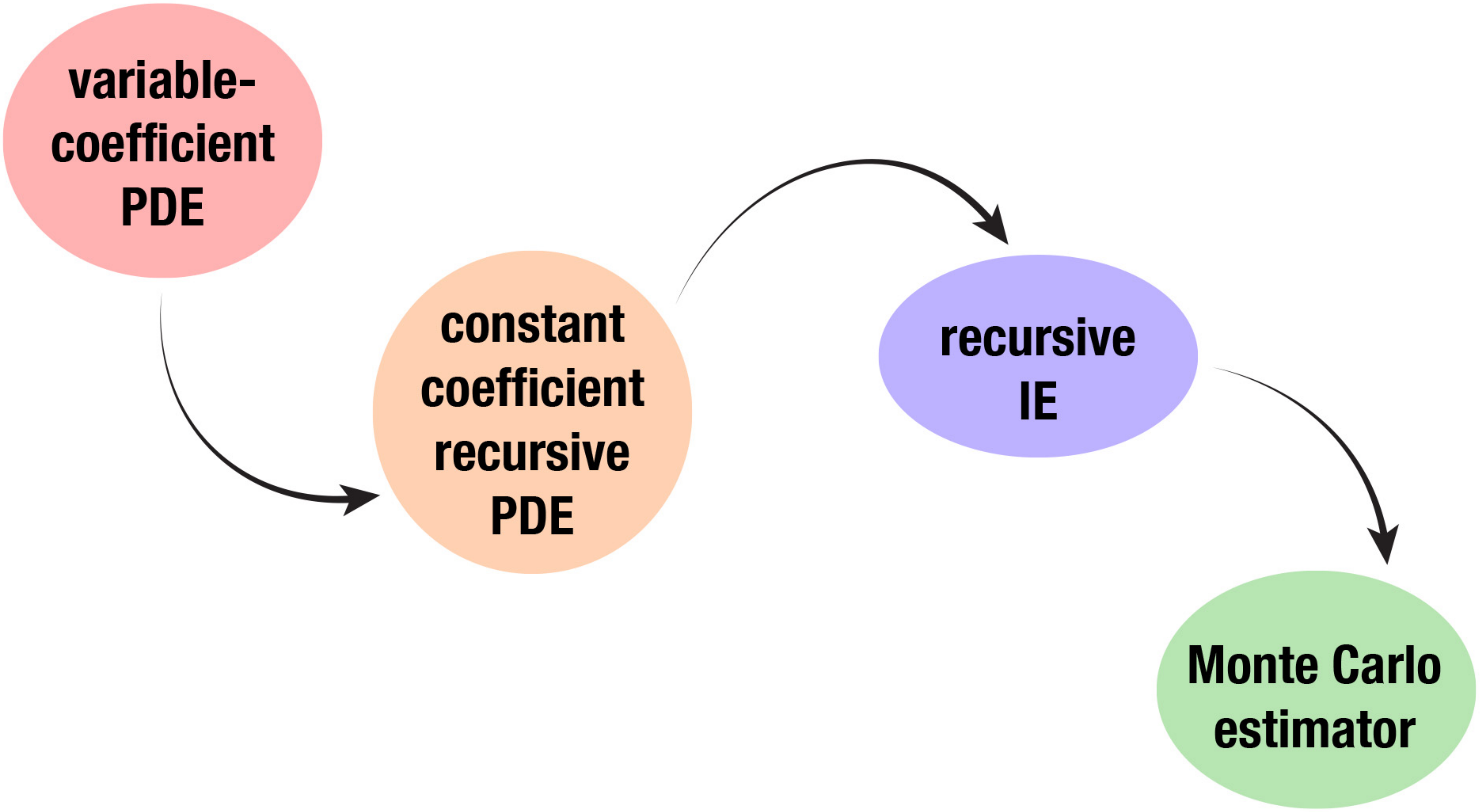
Brownian motion

$$dX_t = \vec{\omega}(X_t)\, dt + \sqrt{\alpha(X_t)}\, dW_t$$

velocity

diffusion rate

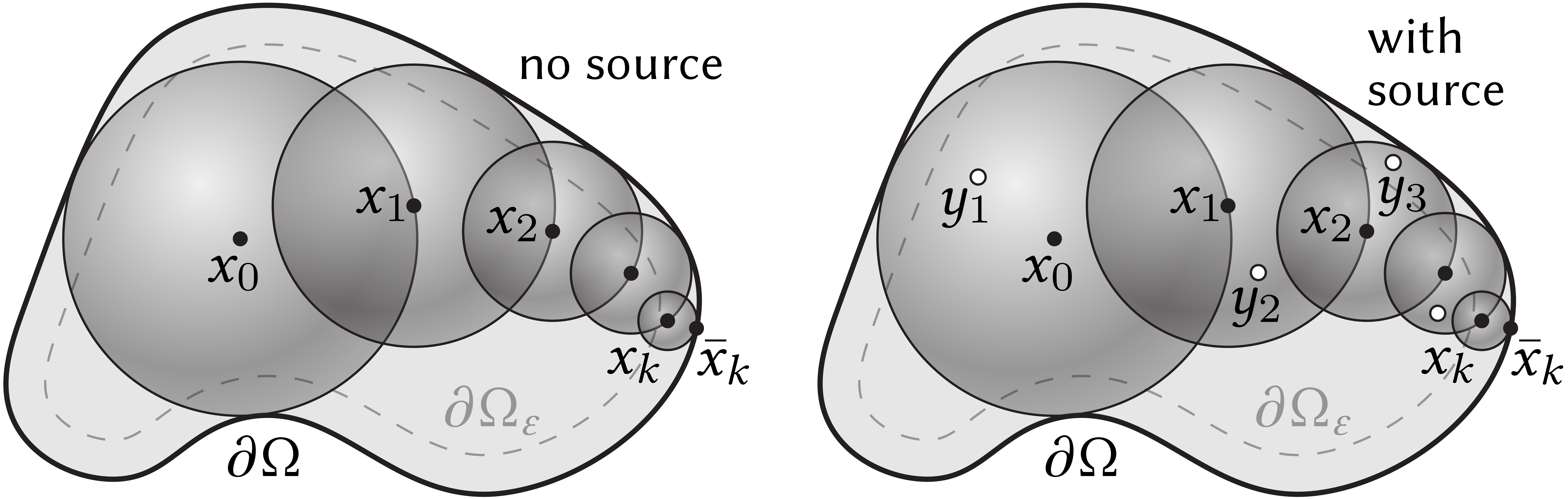
Next: recursive integral equation for variable coefficients



Method

WoS for PDEs with source terms

E.g., $\Delta u = f(x)$; sample the **spatially-varying source** f inside each ball



Transformations

Variable coefficient $\nabla \cdot (\alpha \nabla u) + \overrightarrow{\omega} \cdot \nabla u - \sigma\, u = -f$

Transformations

Variable coefficient $$\nabla \cdot (\alpha \nabla u) + \overrightarrow{\omega} \cdot \nabla u - \sigma\, u = -f$$

$\Downarrow$ Girsanov & delta tracking transformations

Transformations

Variable coefficient

$$\nabla \cdot (\alpha \nabla u) + \overrightarrow{\omega} \cdot \nabla u - \sigma\, u = -f$$

Girsanov & delta tracking transformations

Constant coefficient

(No approximation!)

$$\Delta u - \bar{\sigma}\, u = f(x, \alpha, \overrightarrow{\omega}, \sigma, u)$$

constant

Transformations

Variable coefficient

$$\nabla \cdot (\alpha \nabla u) + \overrightarrow{\omega} \cdot \nabla u - \sigma\, u = -f$$

Girsanov & delta tracking transformations

Constant coefficient

(No approximation!)

$$\Delta u - \bar{\sigma}\, u = \boxed{f(x, \alpha, \overrightarrow{\omega}, \sigma, u)}$$

constant

recursive

Transformations

Variable coefficient

$$\nabla \cdot (\alpha \nabla u) + \overrightarrow{\omega} \cdot \nabla u - \sigma\, u = -f$$

Girsanov & delta tracking transformations

Constant coefficient

(No approximation!)

$$\Delta u - \bar{\sigma}\, u = \boxed{f(x, \alpha, \overrightarrow{\omega}, \sigma, u)} \longleftarrow \text{recursive}$$

constant

Integral

$$\int_{B(x)} f(y, \alpha, \overrightarrow{\omega}, \sigma, u)\; G^{\bar{\sigma}}(x, y)\; dy + \int_{\partial B(x)} u(z)\; P^{\bar{\sigma}}(x, z)\; dz$$

Transformation 1: Girsanov

Re-express Feynman Kac in terms of Brownian motion

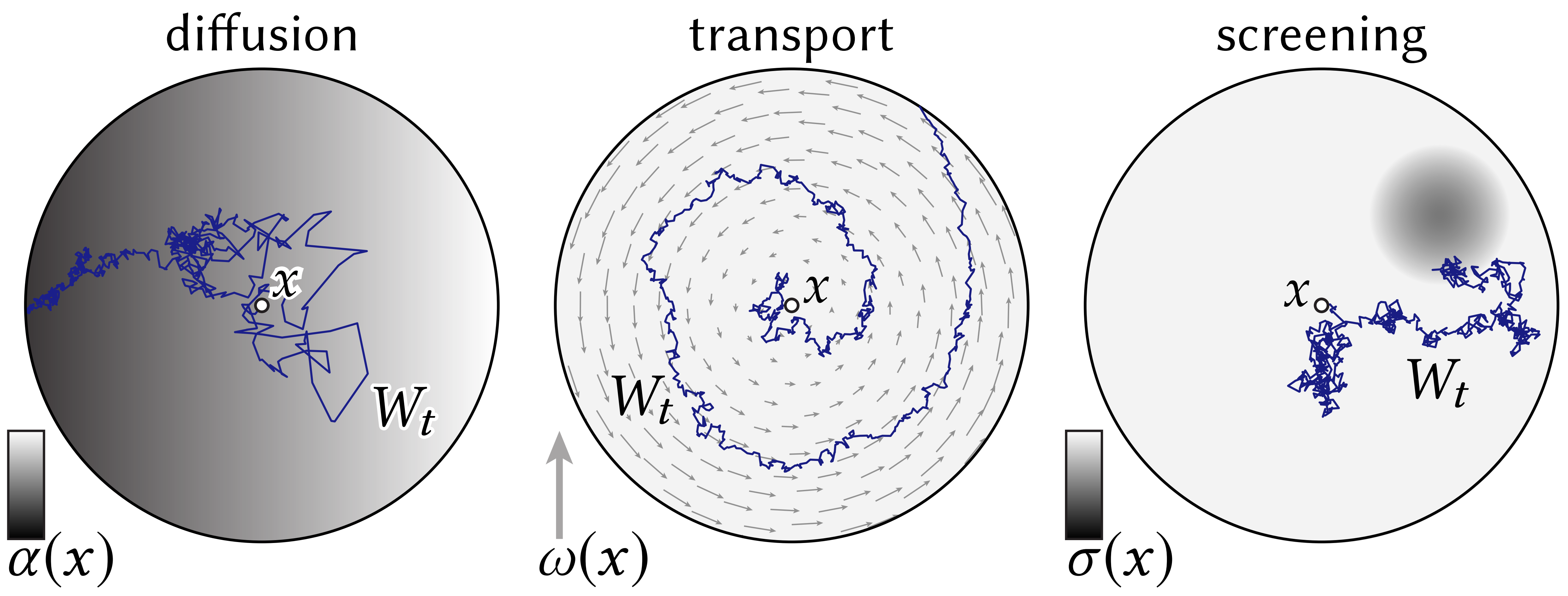
$$u(x) \;=\; \mathbb{E}\left[\int_0^{\tau} e^{-\int_0^t \sigma(W_s)\,ds}\, f(W_t)\,dt \;+\; e^{-\int_0^{\tau} \sigma(W_t)\,dt}\, g(W_\tau)\right]$$

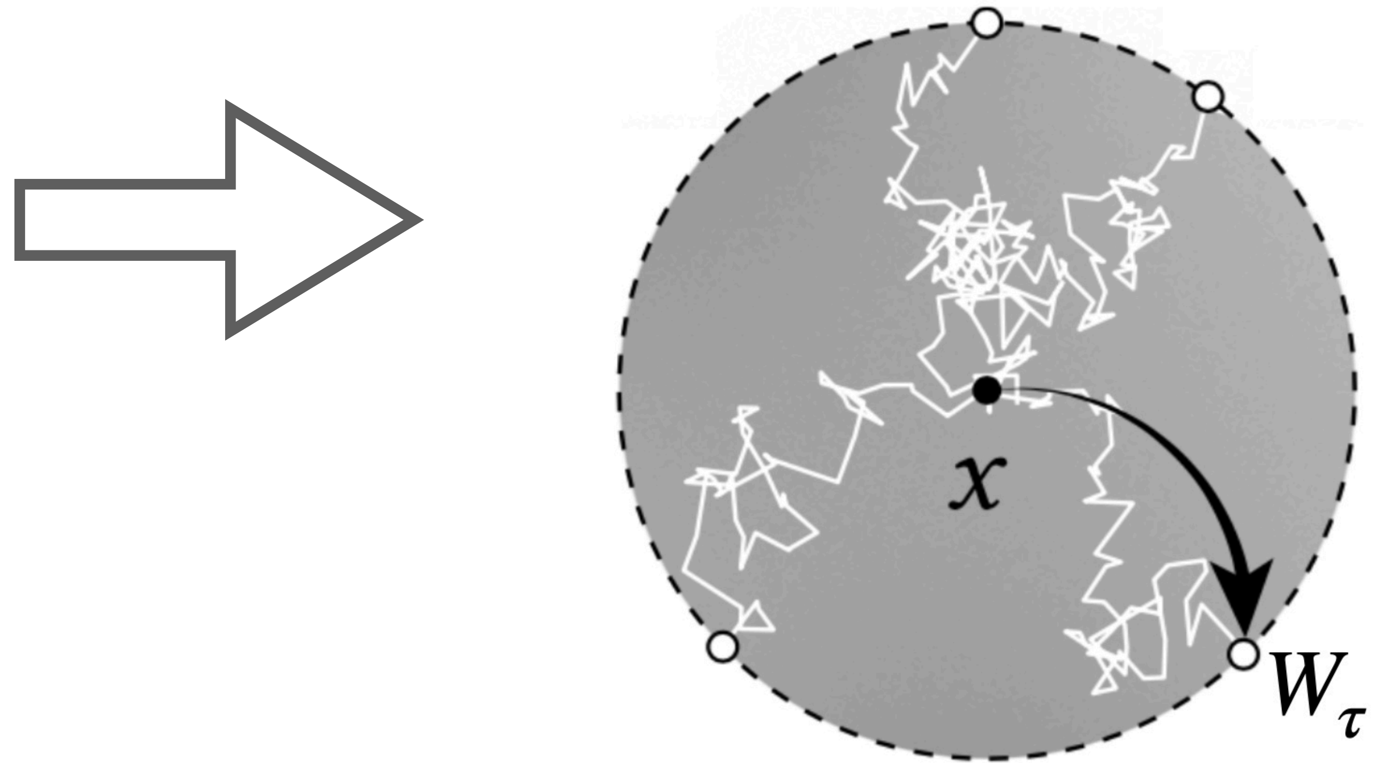
Transformation 1: Girsanov

Re-express Feynman Kac in terms of Brownian motion

$$u(x) \;=\; \mathbb{E}\left[\int_0^{\tau} e^{-\int_0^t \sigma(W_s)\,ds}\, f(W_t)\,dt \;+\; e^{-\int_0^\tau \sigma(W_t)\,dt}\, g(W_\tau)\right]$$

$$dX_t = \vec{\omega}(X_t)\,dt \;+\; \sqrt{\alpha(X_t)}\,dW_t \qquad\qquad dX_t = dW_t$$

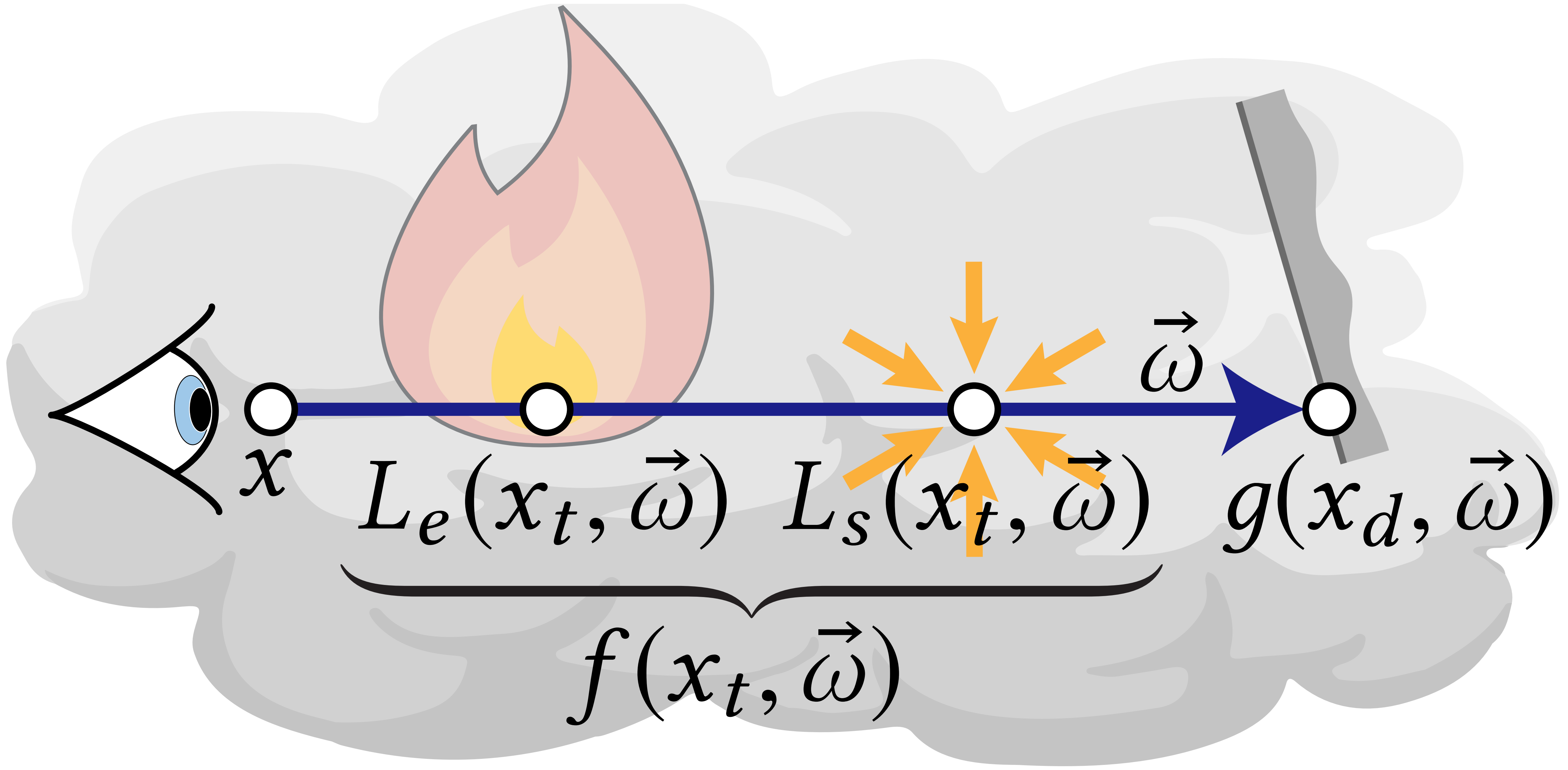




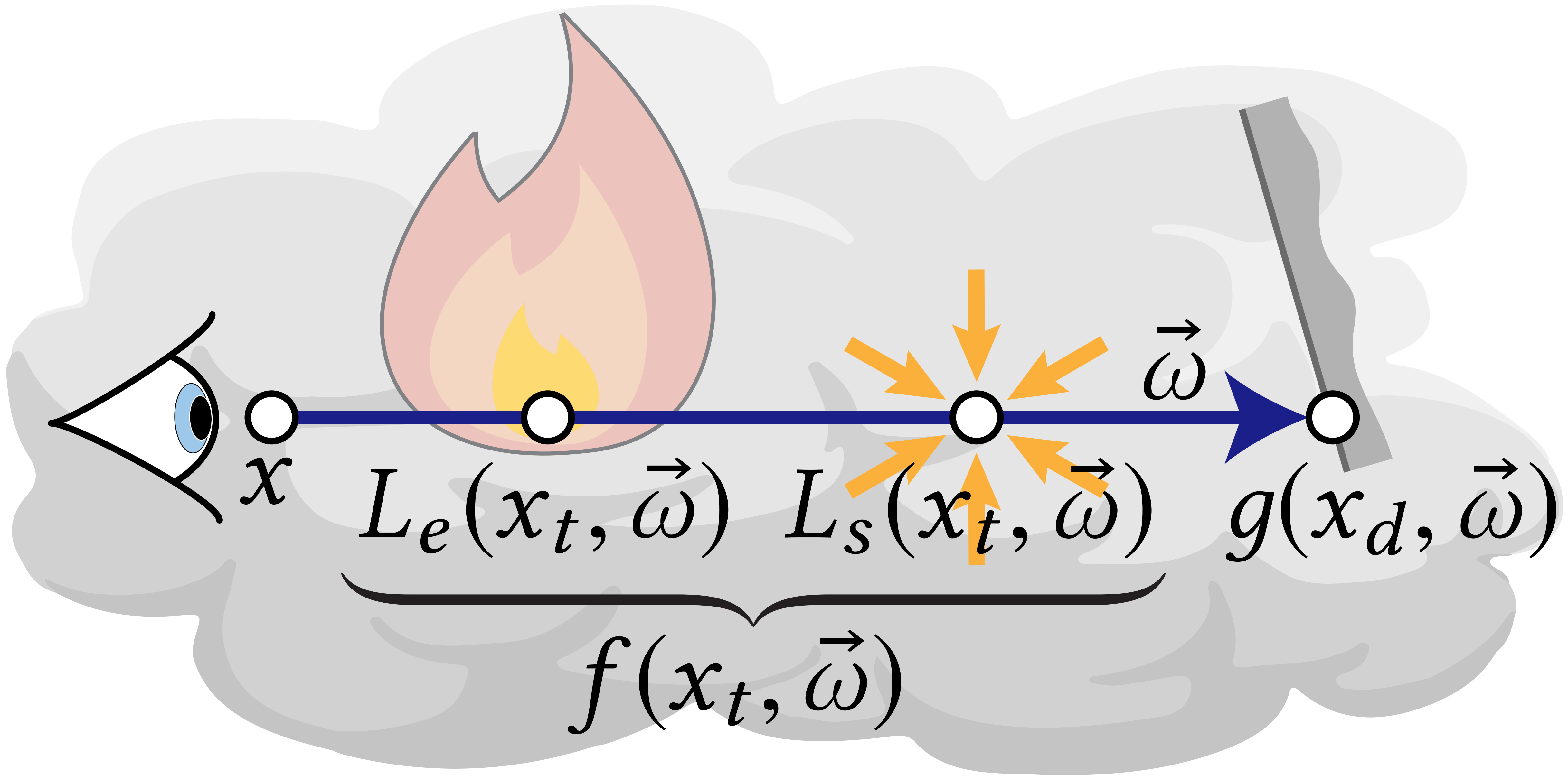
Volume Rendering Equation (VRE)

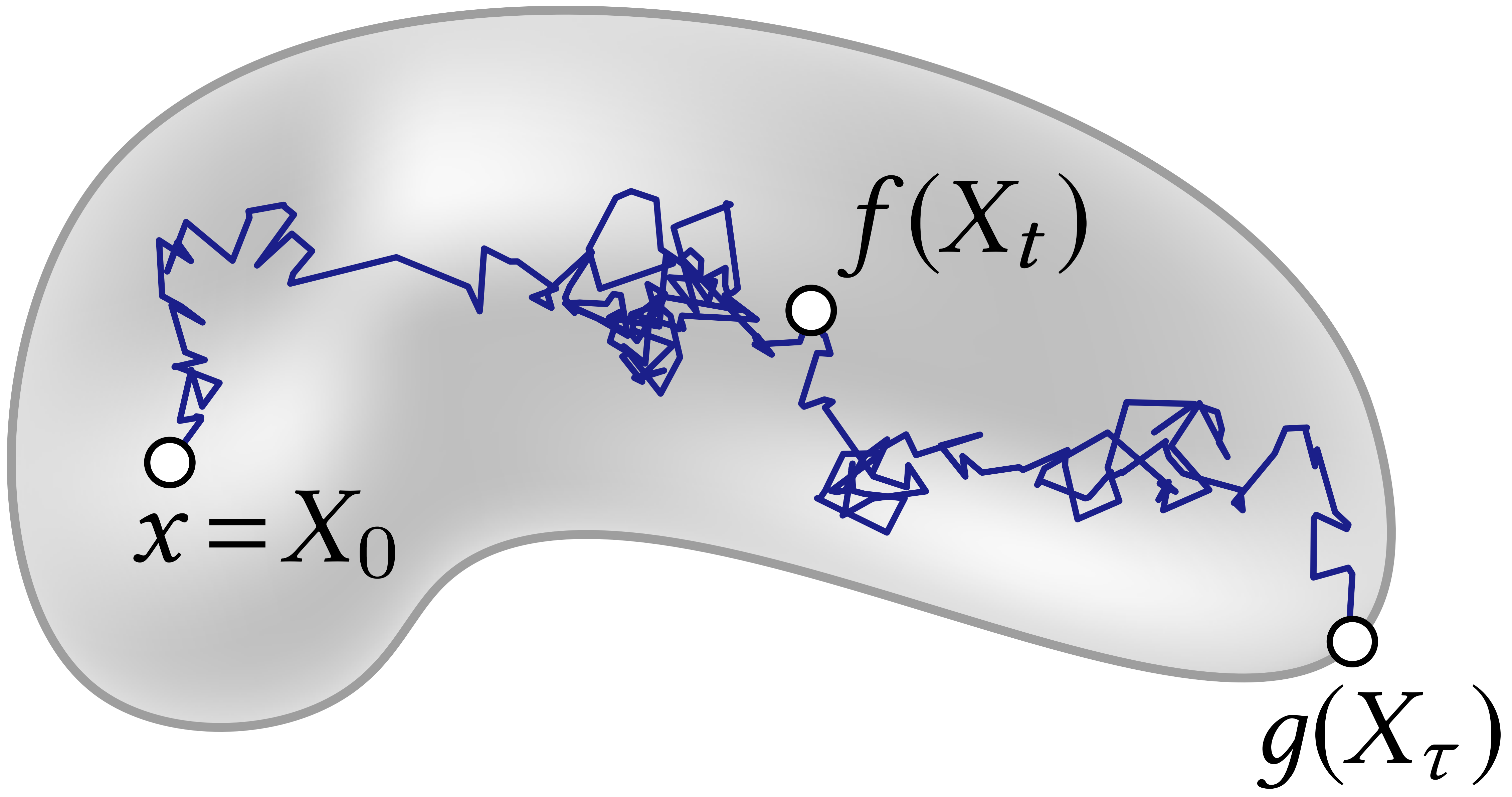
VRE describes the radiance in heterogeneous absorbing & scattering media

$$L(w, \overrightarrow{\omega}) = \int_0^d e^{-\int_0^t \sigma(x_s)\, ds} f(x_t, \overrightarrow{\omega})\, dt + e^{-\int_0^d \sigma(x_t)\, dt} g(x_d, \overrightarrow{\omega})$$



Structural connection between VRE & Feynman-Kac





$$L(w, \vec{\omega}) \;=\; \int_0^d e^{-\int_0^t \sigma(x_s)\,ds} f(x_t, \vec{\omega})\,dt \;+\; e^{-\int_0^d \sigma(x_t)\,dt}\, g(x_d, \vec{\omega})$$

$$u(x) \;=\; \mathbb{E}\left[\int_0^\tau e^{-\int_0^t \sigma(W_s)\,ds} f(W_t)\,dt \;+\; e^{-\int_0^\tau \sigma(W_t)\,dt}\, g(W_\tau)\right]$$

VRE describes the radiance in heterogeneous absorbing & scattering media

Feynman-Kac for 2nd order variable coefficient PDEs

Transformation 2: Delta tracking

Variable coefficient

$$u(x) \;=\; \mathbb{E}\left[\int_0^{\tau} e^{-\int_0^t \sigma(W_s)\,ds}\, f(W_t)\,dt \;+\; e^{-\int_0^{\tau} \sigma(W_t)\,dt}\; g(W_\tau)\right]$$

$$\Downarrow$$

Constant coefficient

(No approximation!)

$$u(x) \;=\; \mathbb{E}\left[\int_0^{\tau} e^{-\bar{\sigma}t}\, f(W_t, \sigma, u)\,dt \;+\; e^{-\bar{\sigma}\tau}\; g(W_\tau)\right]$$

Transformation 2: Delta tracking

Variable coefficient

$$u(x) \;=\; \mathbb{E}\left[\int_0^\tau e^{-\int_0^t \sigma(W_s)\,ds}\, f(W_t)\,dt \;+\; e^{-\int_0^\tau \sigma(W_t)\,dt}\, g(W_\tau)\right]$$

Constant coefficient

(No approximation!)

$$u(x) \;=\; \mathbb{E}\left[\int_0^\tau e^{-\bar{\sigma}t}\, \underbrace{\boxed{f(W_t, \sigma, u)}}_{\text{recursive}}\,dt \;+\; e^{-\bar{\sigma}\tau}\, g(W_\tau)\right]$$

Transformation 2: Delta tracking

Variable coefficient

$$u(x) = \mathbb{E}\left[\int_0^\tau e^{-\int_0^t \sigma(W_s)\,ds} f(W_t)\,dt + e^{-\int_0^\tau \sigma(W_t)\,dt} g(W_\tau)\right]$$

Constant coefficient

(No approximation!)

$$u(x) = \mathbb{E}\left[\int_0^\tau e^{-\bar{\sigma}t} \underbrace{\boxed{f(W_t, \sigma, u)}}_{\text{recursive}}\,dt + e^{-\underbrace{\bar{\sigma}}_{\text{constant}}\tau} g(W_\tau)\right]$$

Transformation 2: Delta tracking

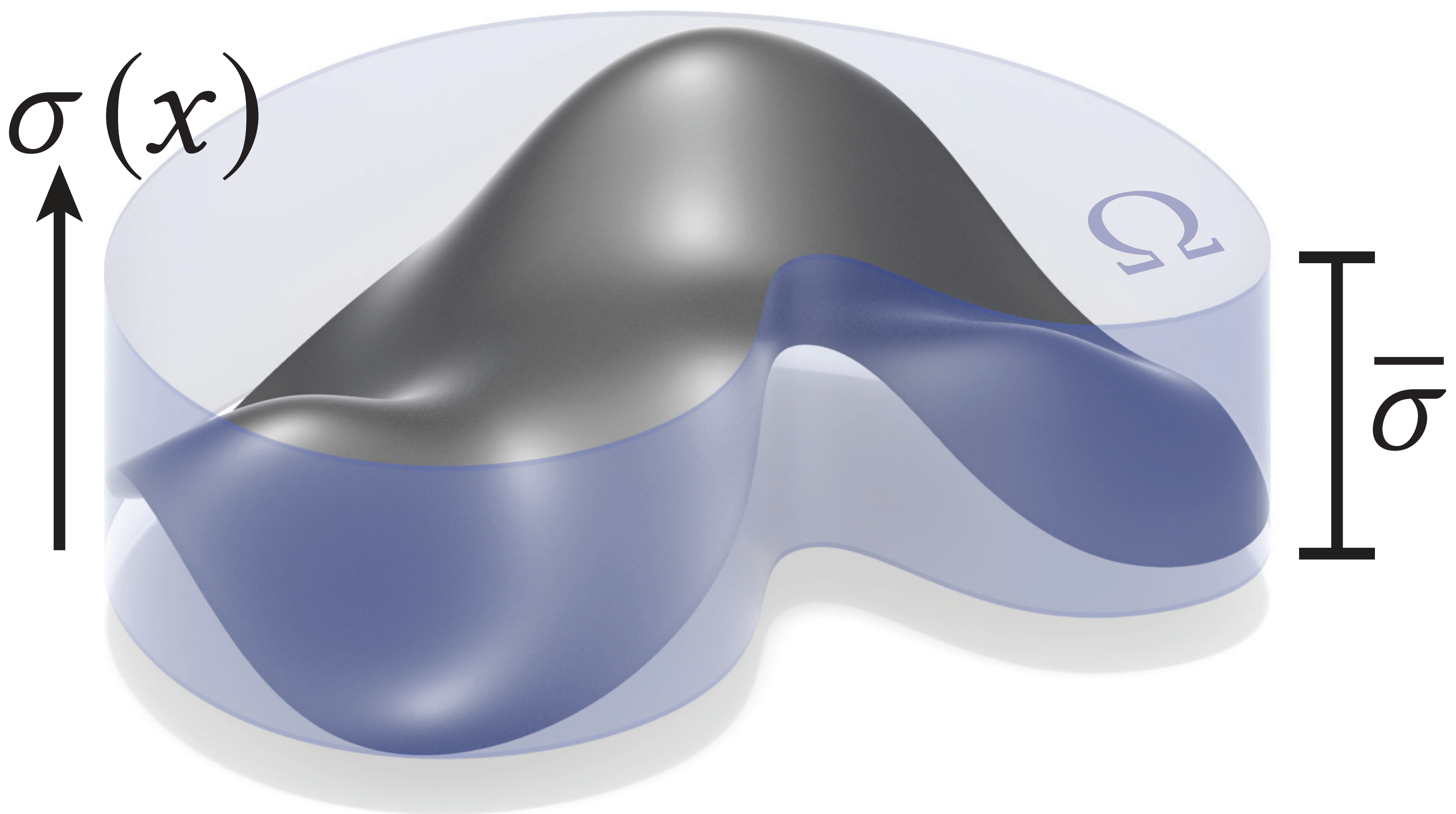
Variable coefficient

$$u(x) = \mathbb{E}\left[\int_0^\tau e^{-\int_0^t \sigma(W_s)\,ds} f(W_t)\,dt + e^{-\int_0^\tau \sigma(W_t)\,dt} g(W_\tau)\right]$$

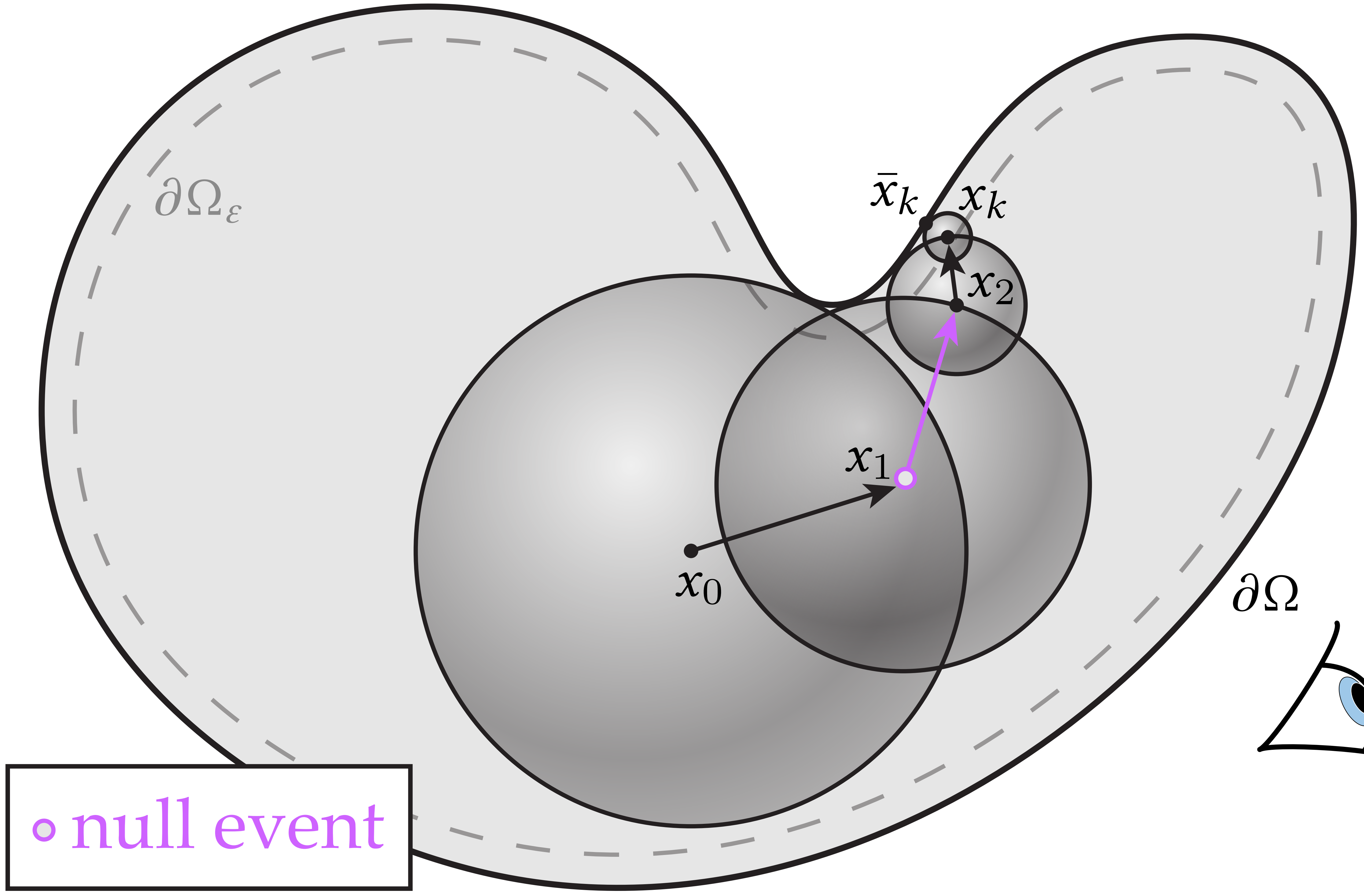
Constant coefficient

(No approximation!)

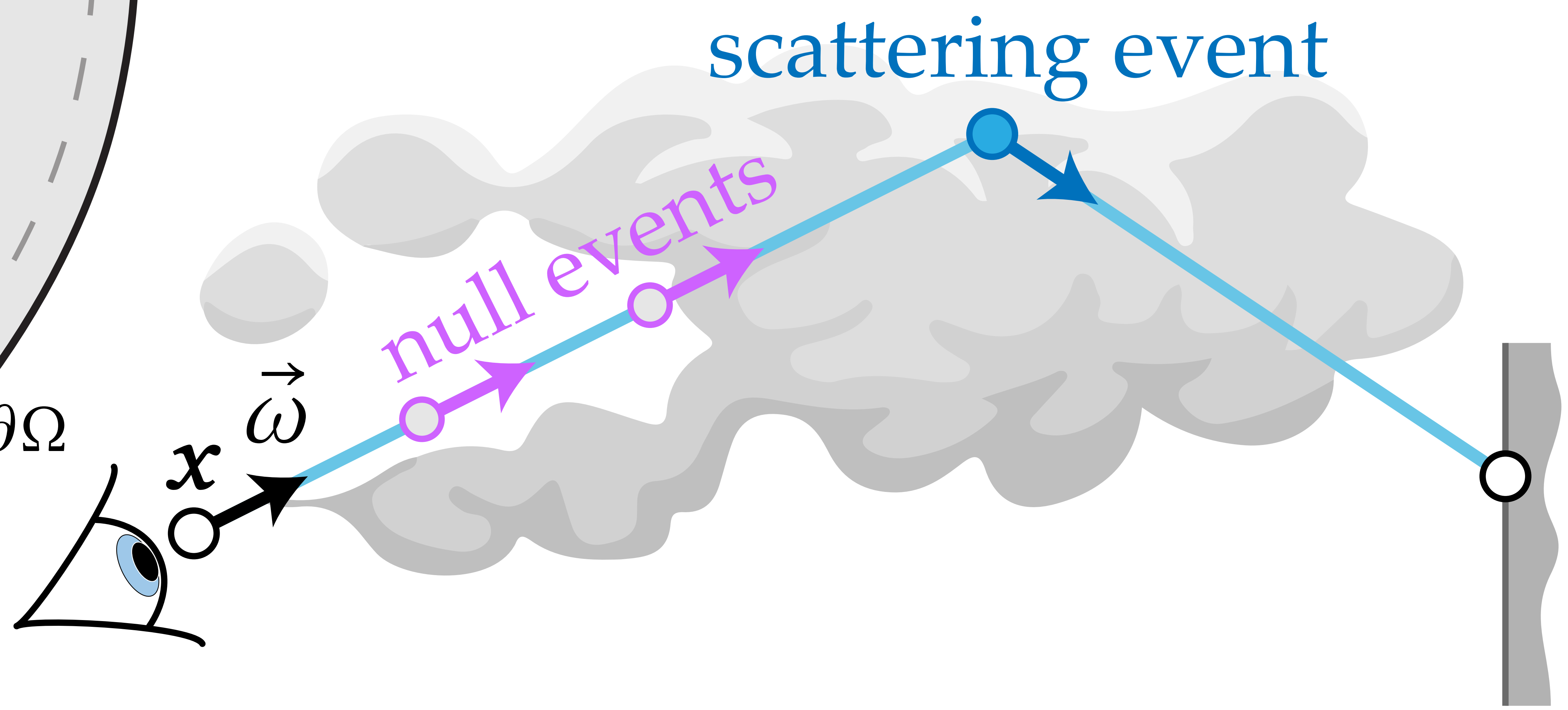
$$u(x) = \mathbb{E}\left[\int_0^\tau e^{-\bar{\sigma}t} \underbrace{\boxed{f(W_t, \sigma, u)}}_{\text{recursive}}\,dt + e^{-\underbrace{\bar{\sigma}}_{\text{constant}}\tau} g(W_\tau)\right]$$



Delta tracking variant of WoS

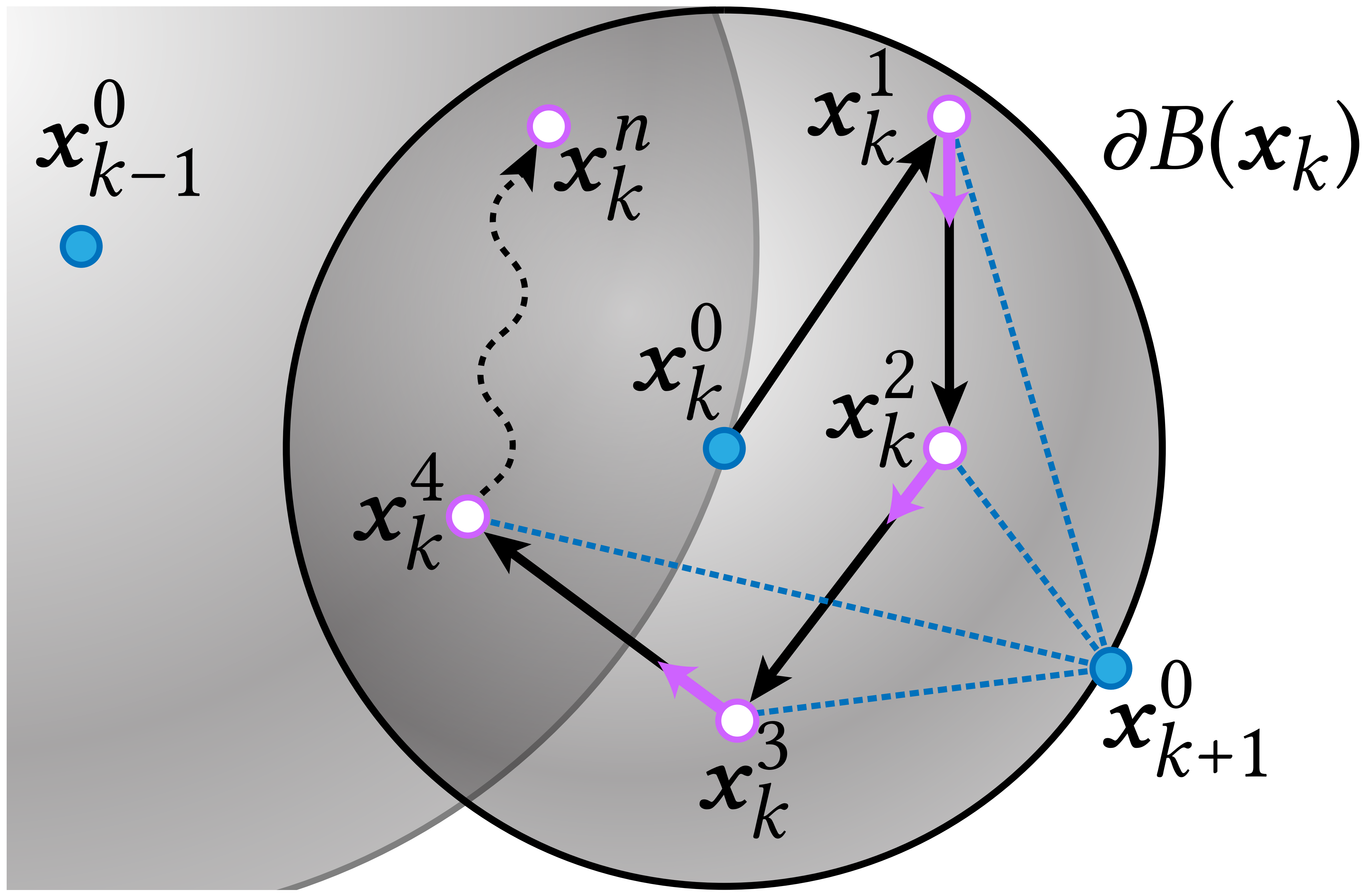


WoS delta tracking



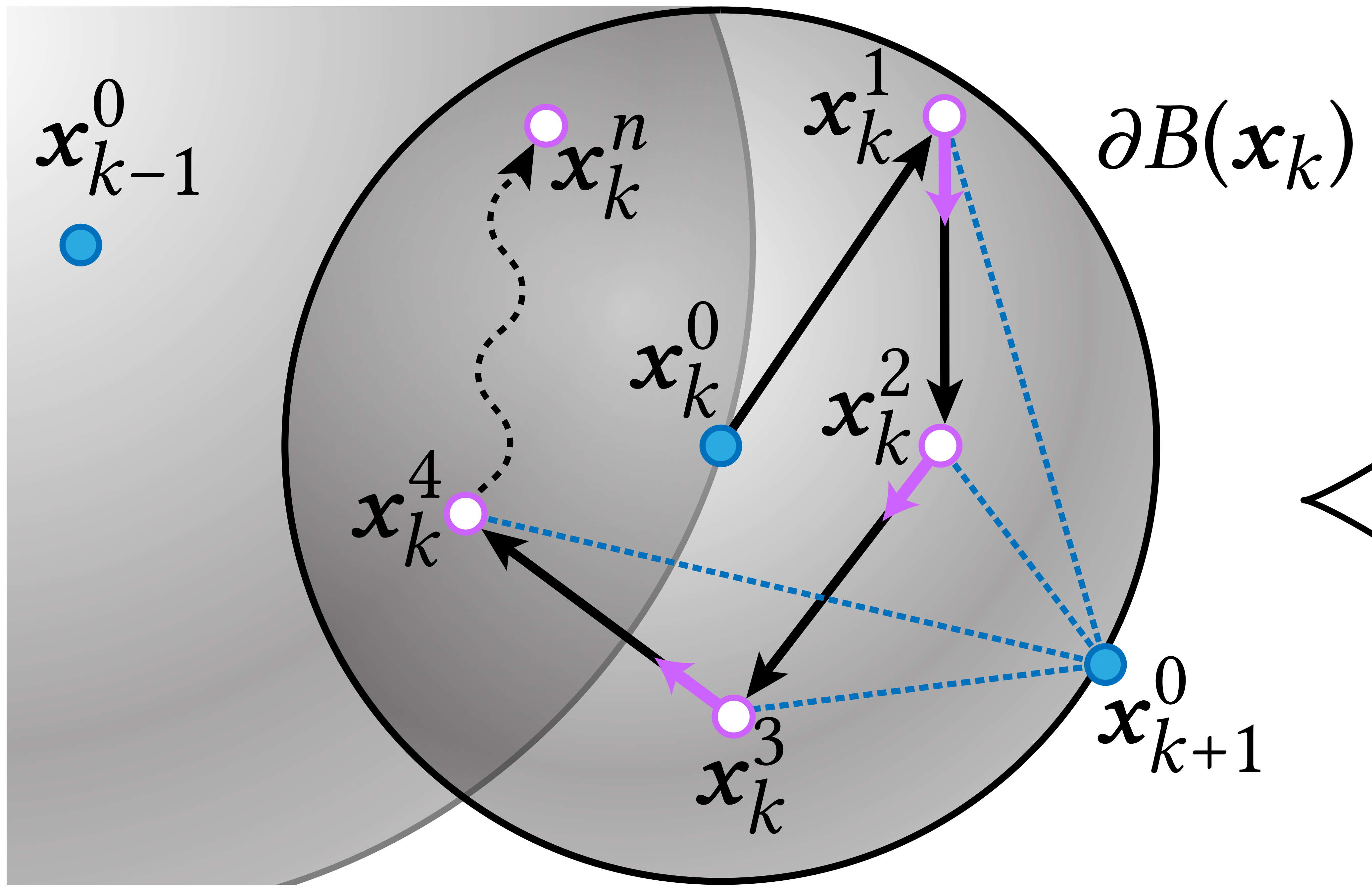
delta tracking method in volume rendering
[Woodcock et al., 1965, Raab et al. 2008]

Next-flight variant of WoS

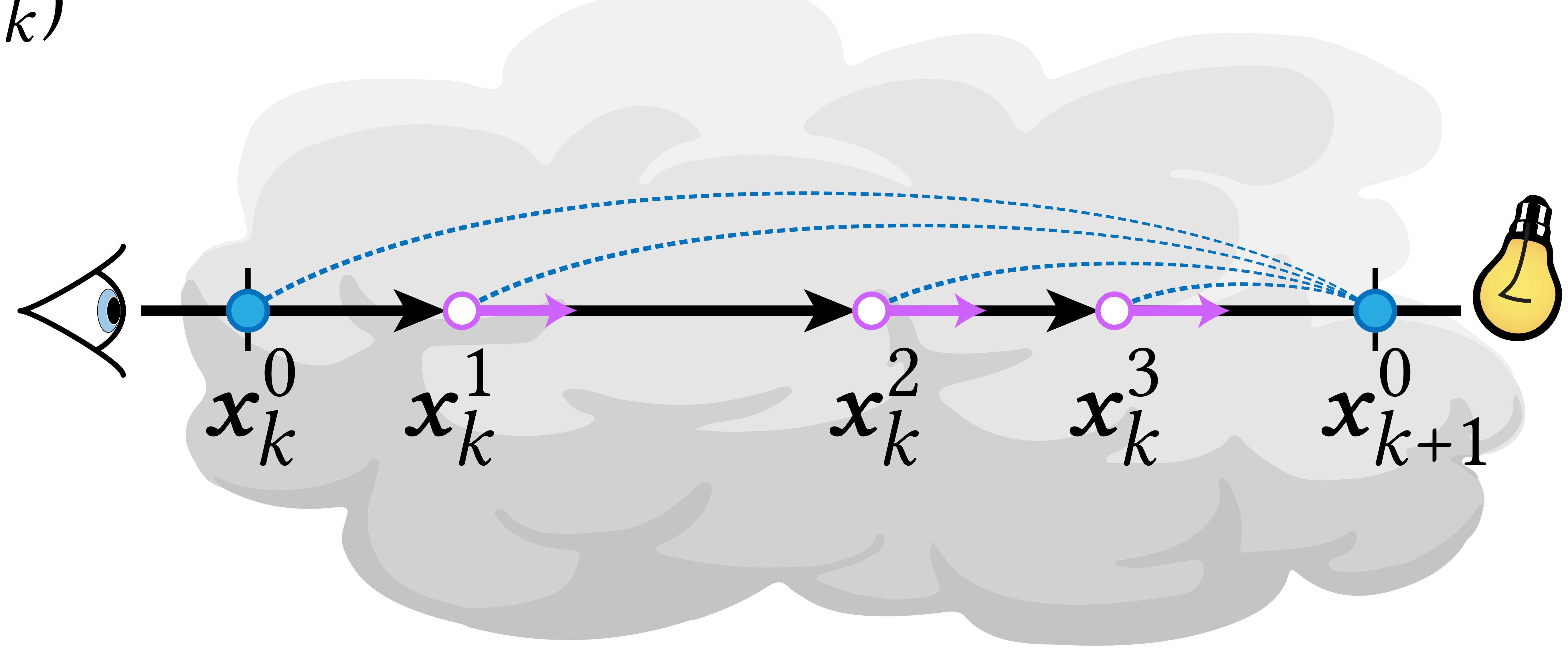


WoS next-flight

Next-flight variant of WoS

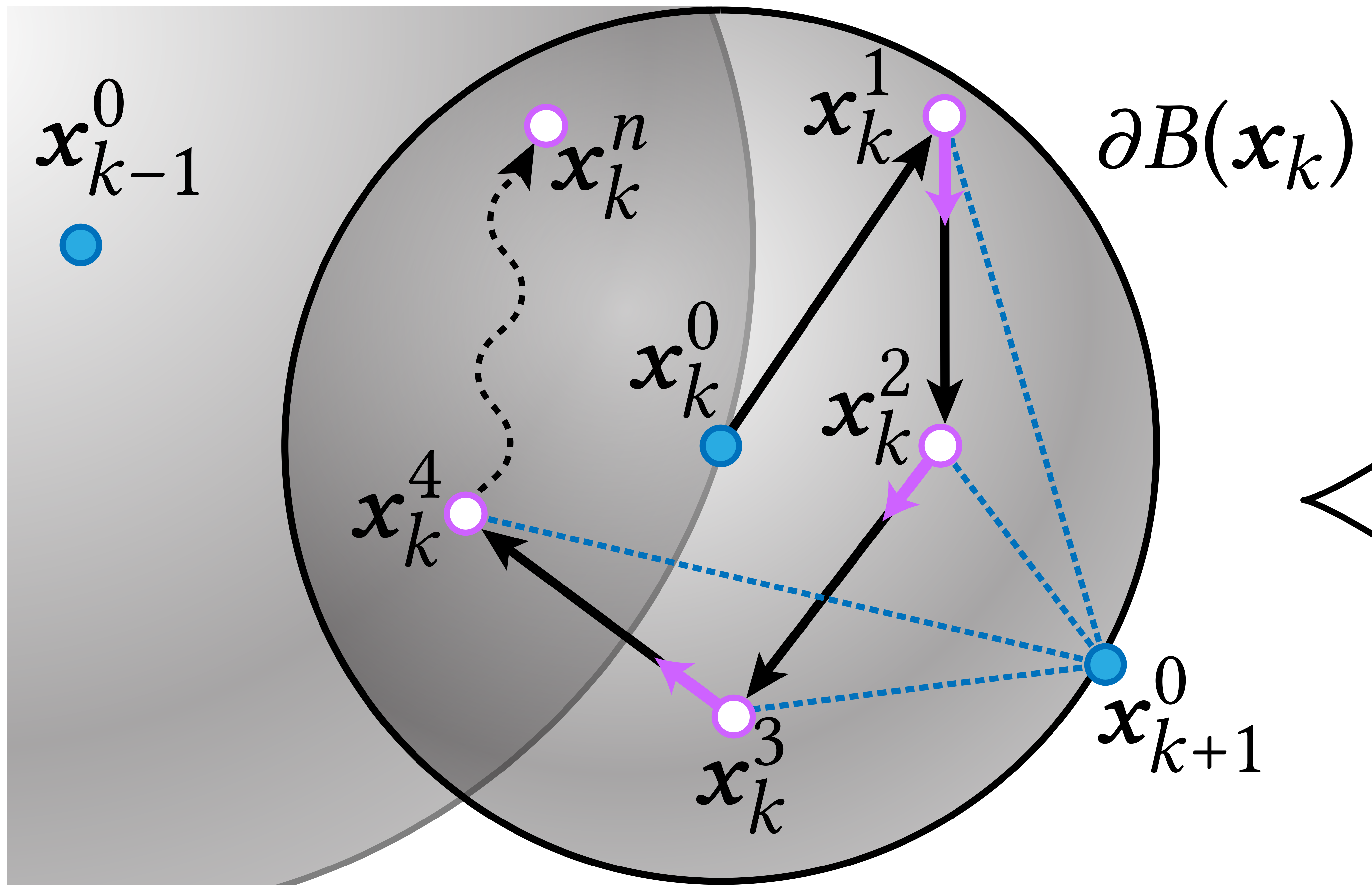


WoS next-flight

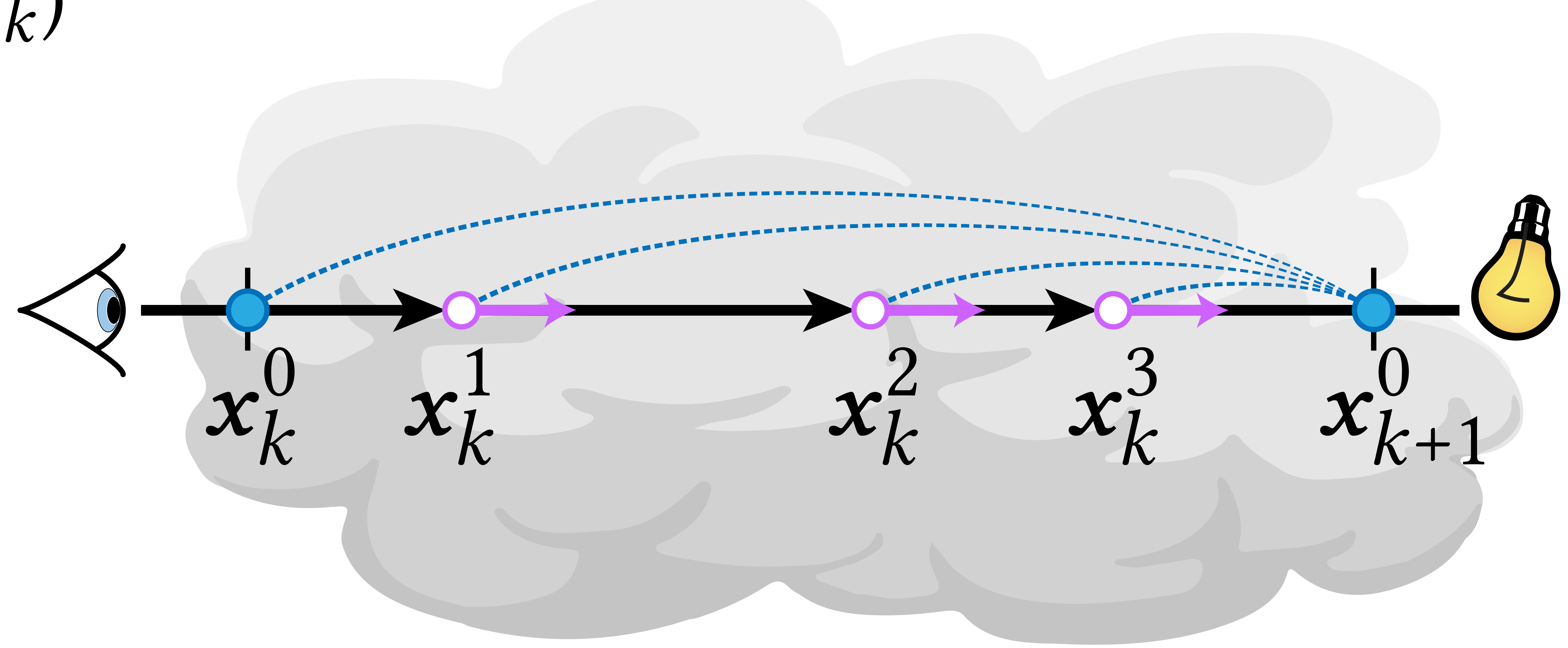


Next-flight algorithm in volume rendering [Cramer 1978]

Next-flight variant of WoS



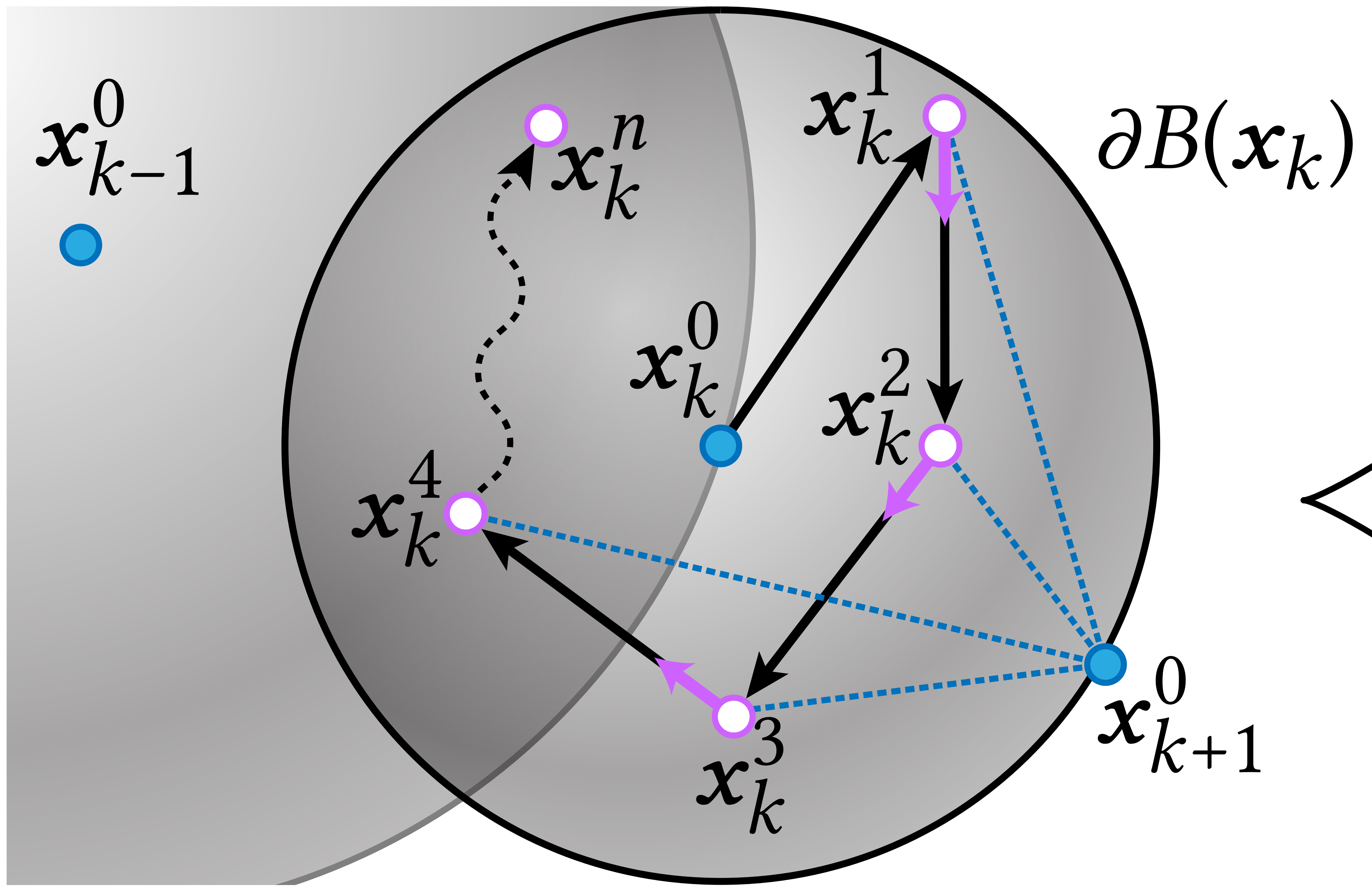
WoS next-flight



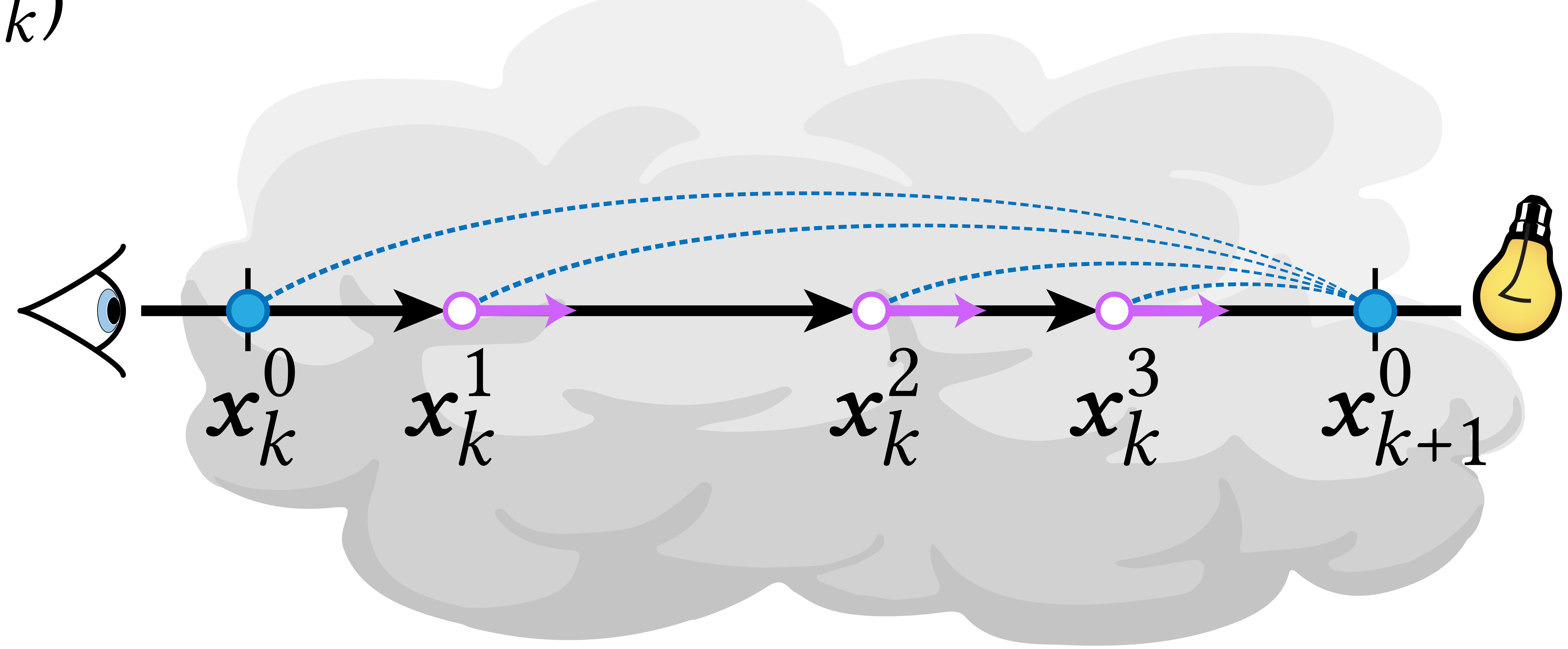
Next-flight algorithm in volume rendering [Cramer 1978]

Fewer distance queries higher correlation compared to delta tracking WoS

Next-flight variant of WoS



WoS next-flight



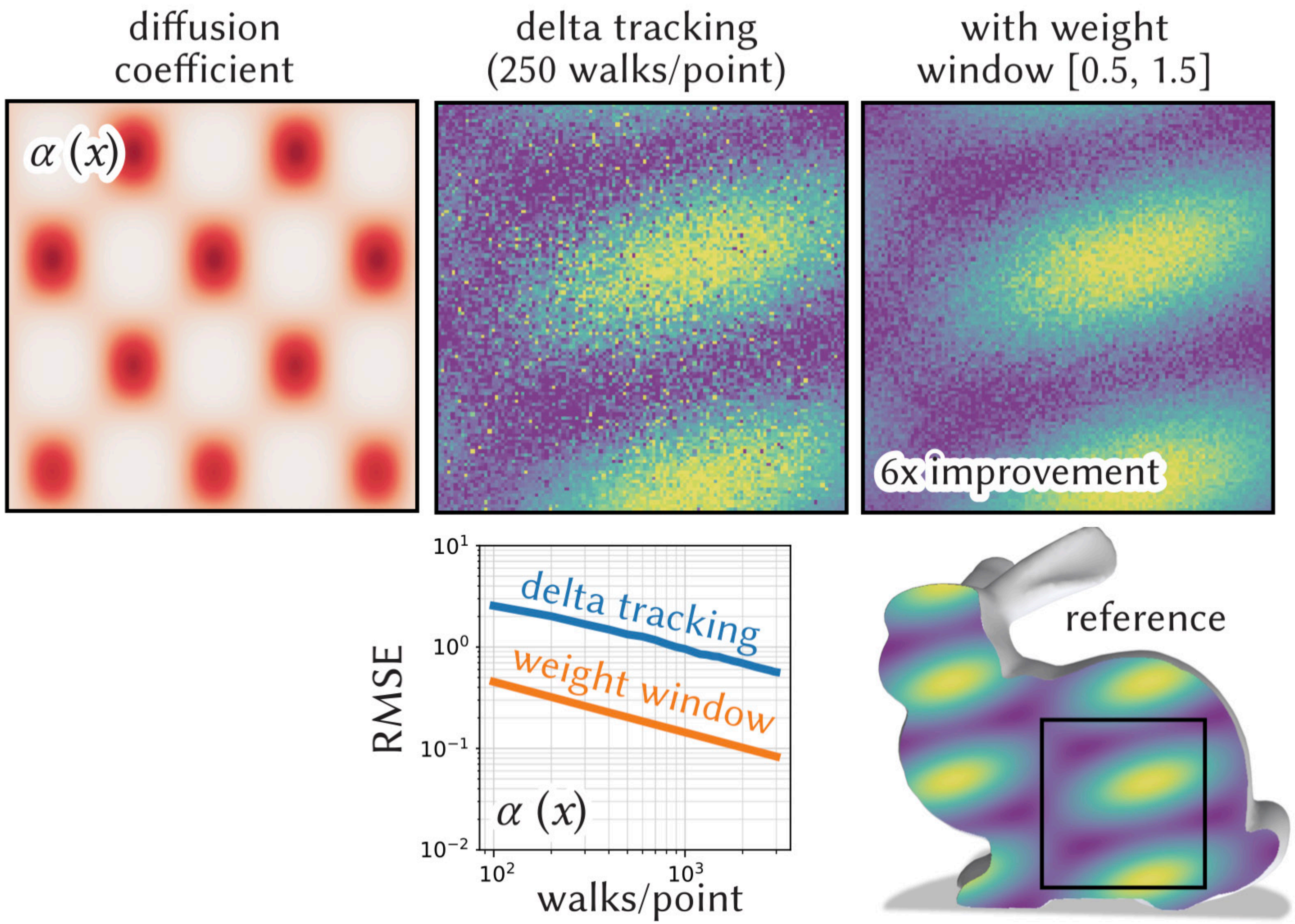
Next-flight algorithm in volume rendering [Cramer 1978]

Fewer distance queries higher correlation compared to delta tracking WoS

Similar variance & run-time characteristics as volume rendering counterparts

Weight window [Hoogenboom and Légrády 2005]

Uses splitting and Russian roulette to reduce noise



Implementation & Results

PDE inputs

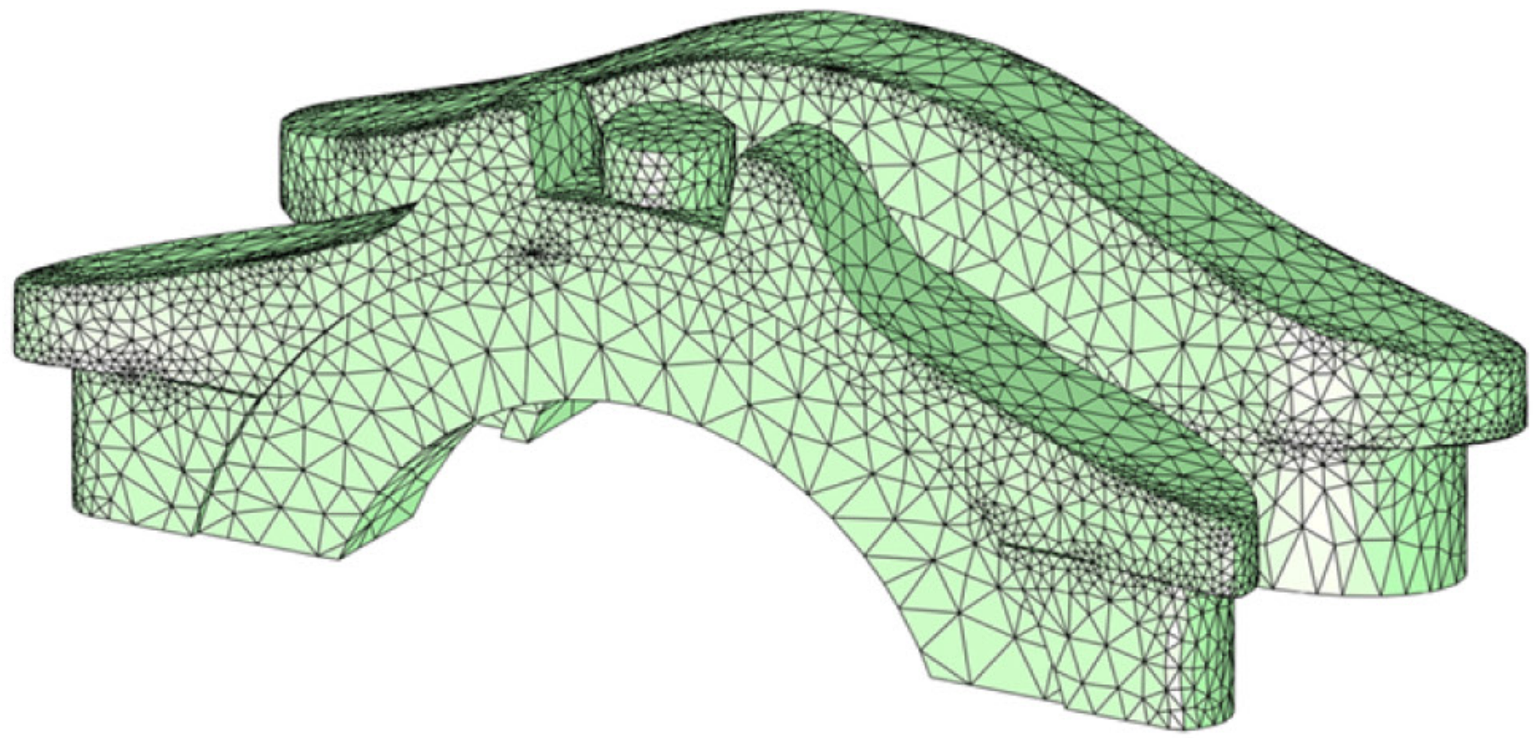
$$\overset{\text{diffusion}}{\boxed{\nabla \cdot (\alpha \nabla u)}} + \overset{\text{drift}}{\boxed{\vec{\omega} \cdot \nabla u}} - \overset{\text{absorption}}{\boxed{\sigma u}} = \overset{\text{source}}{\boxed{-f}} \text{ on } \overset{\text{domain}}{\boxed{\Omega}}$$

$$\underset{\text{solution}}{u} = \underset{\substack{\text{boundary}\\\text{values}}}{\boxed{g}} \text{ on } \underset{\substack{\text{domain}\\\text{boundary}}}{\boxed{\partial\Omega}}$$

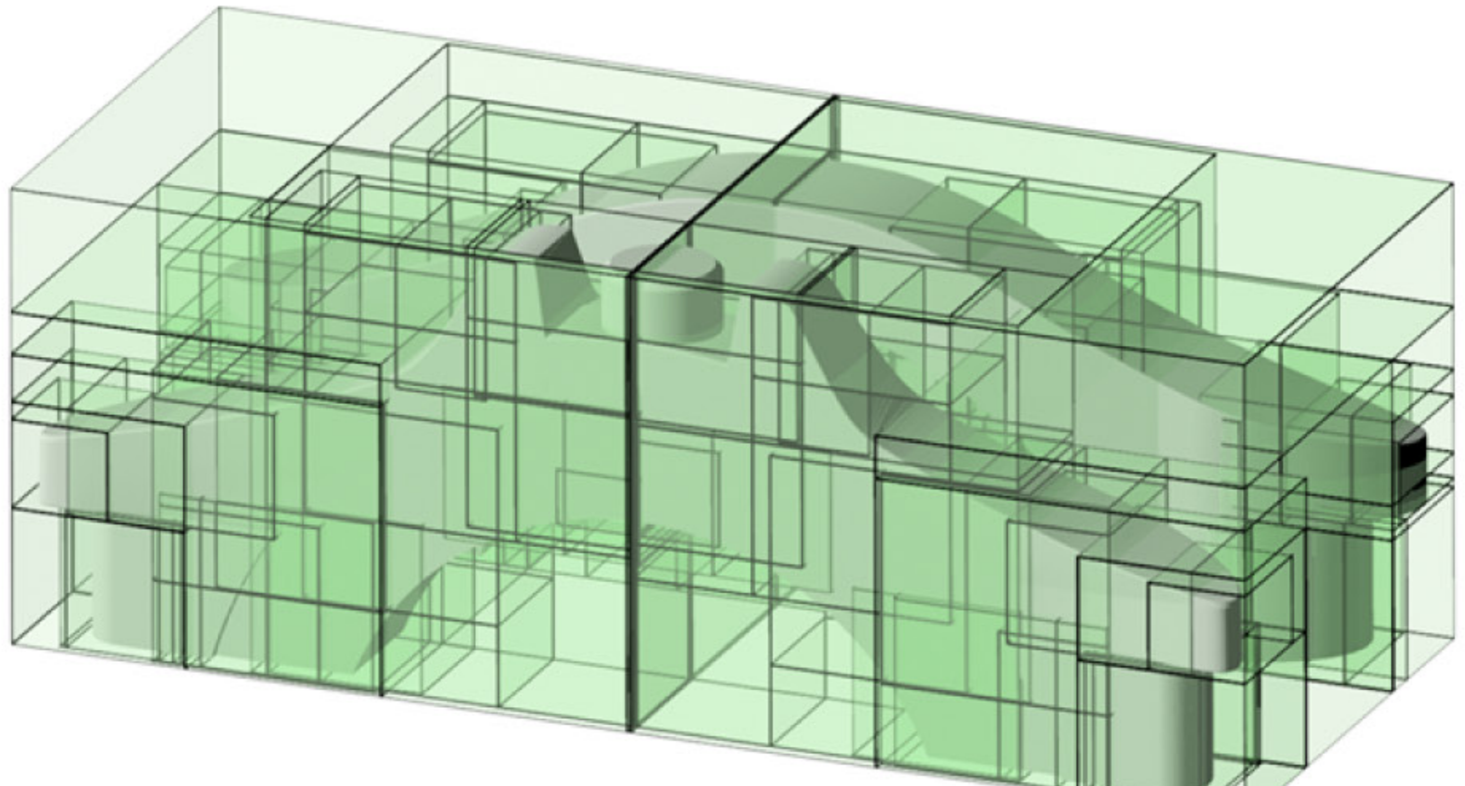
Acceleration of closest point queries

Accelerate closest point queries using BVH

input
(Thingi10k #996816)



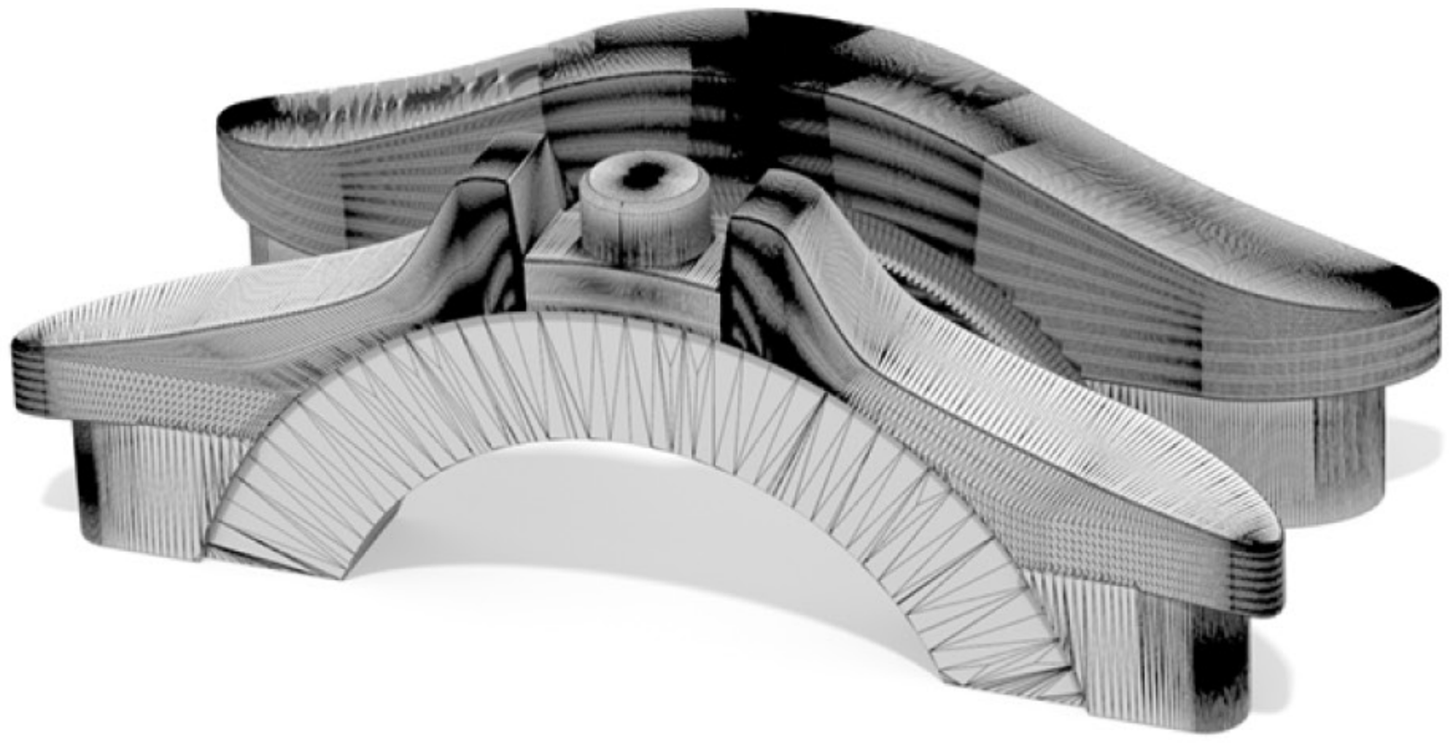
mesh w/ FastTetWild
1 hour 25 minutes



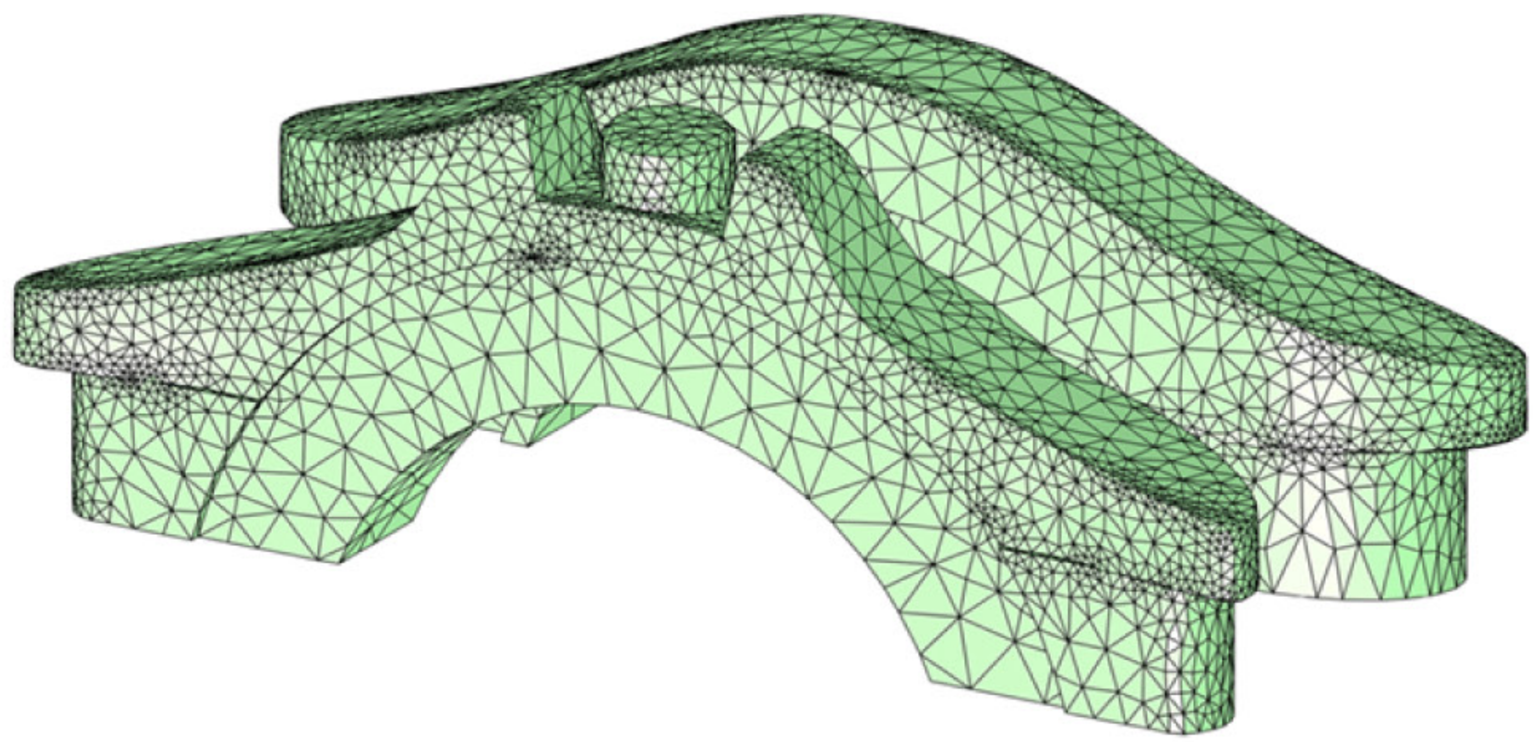
build BVH for WoS
< 1 second

Acceleration of closest point queries

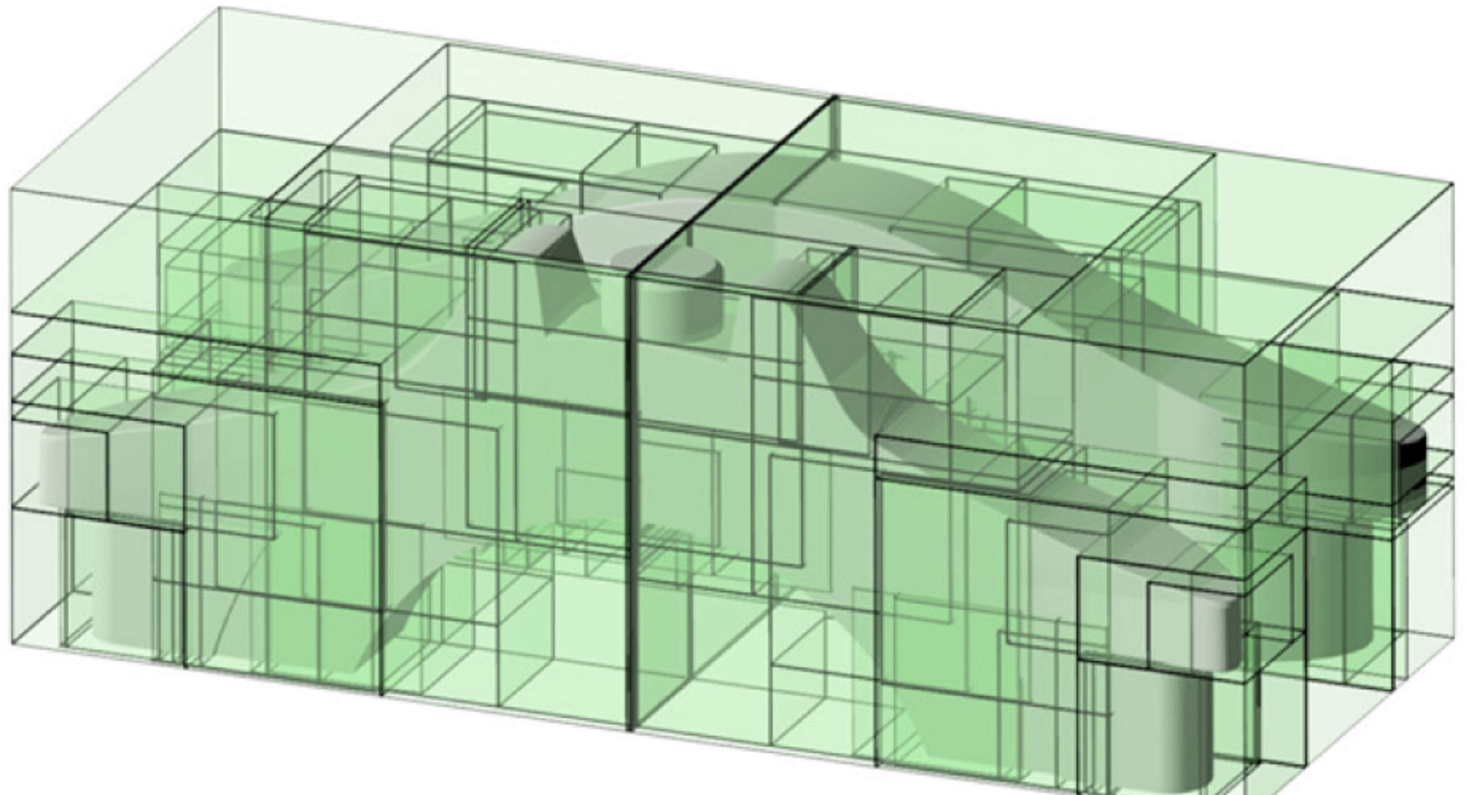
Accelerate closest point queries using BVH



input
(Thingi10k #996816)



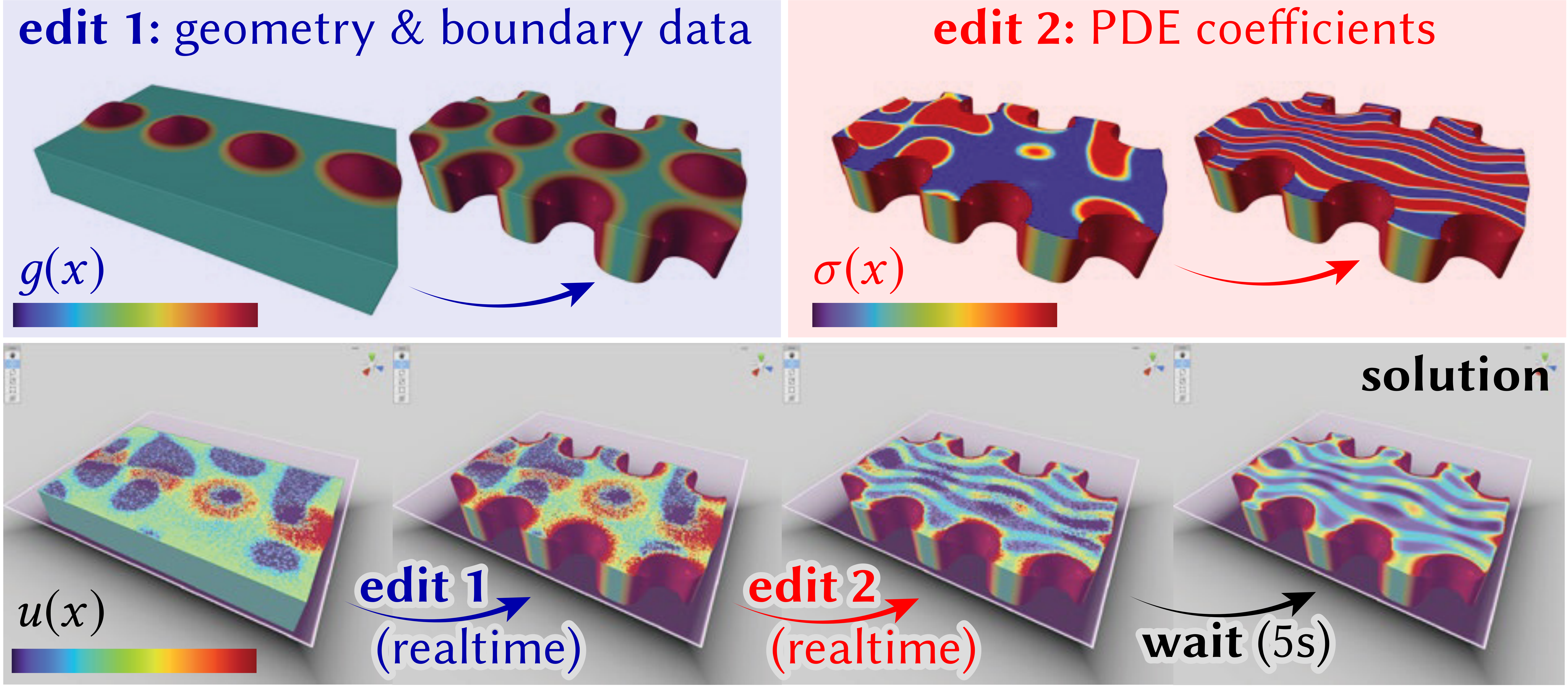
mesh w/ FastTetWild
1 hour 25 minutes



build BVH for WoS
< 1 second

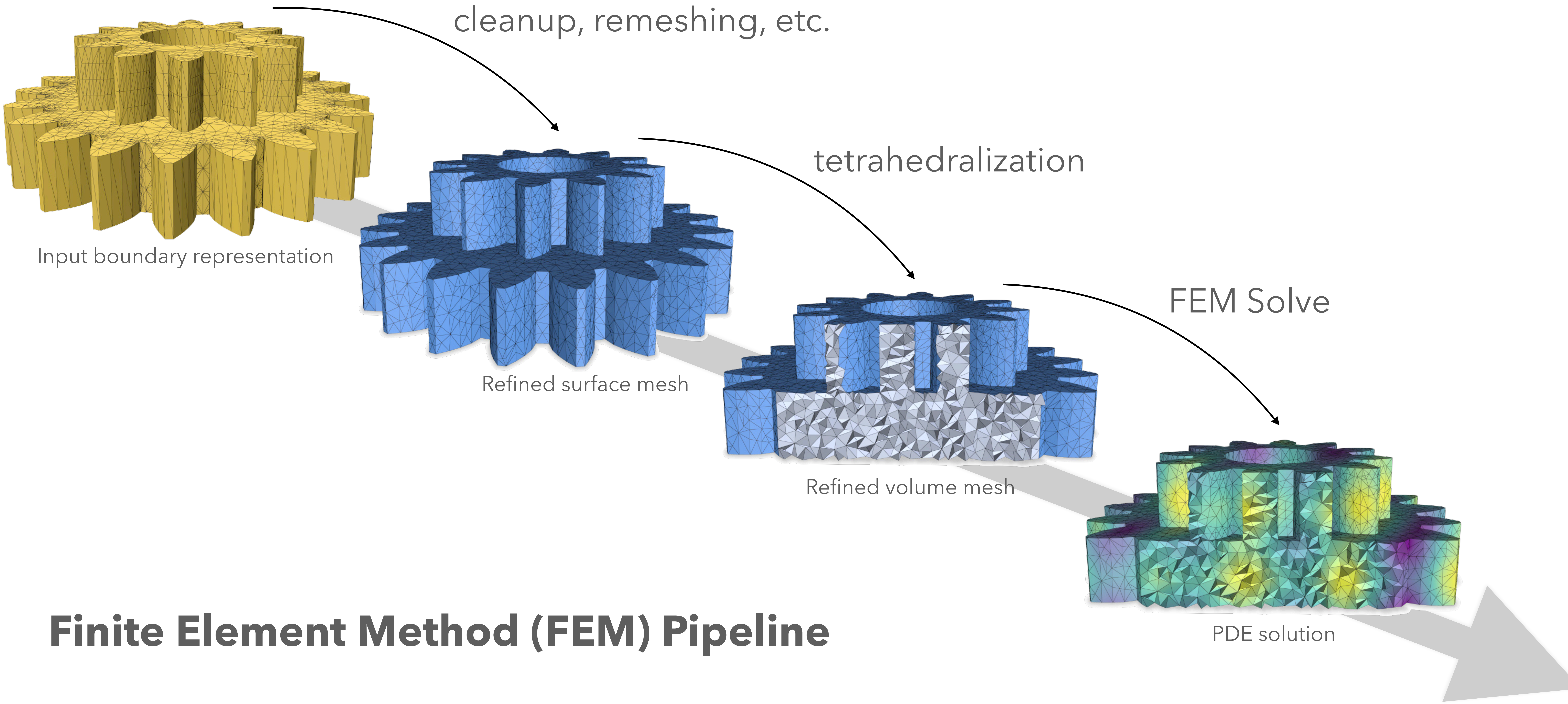
Unlike bad meshes, BVHs do not impact correctness/accuracy of PDE solution!

Interactive editing



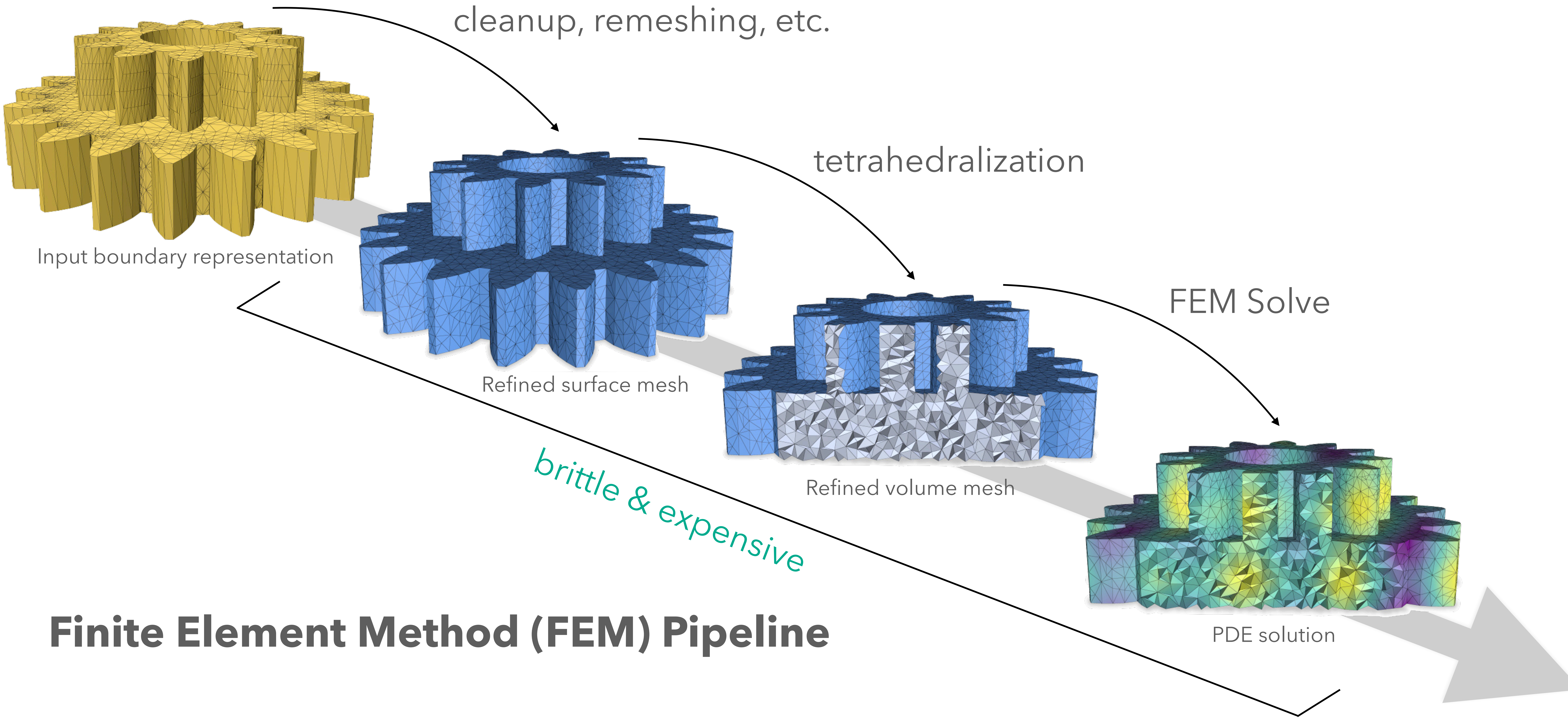

No model cleanup, reduction or homogenization!

End-to-end cost of conventional PDE solvers



Finite Element Method (FEM) Pipeline

End-to-end cost of conventional PDE solvers



Finite Element Method (FEM) Pipeline

Conventional PDE solvers can be brittle

Poor mesh quality completely throws off FEM solution

FEM solution

Reference solution

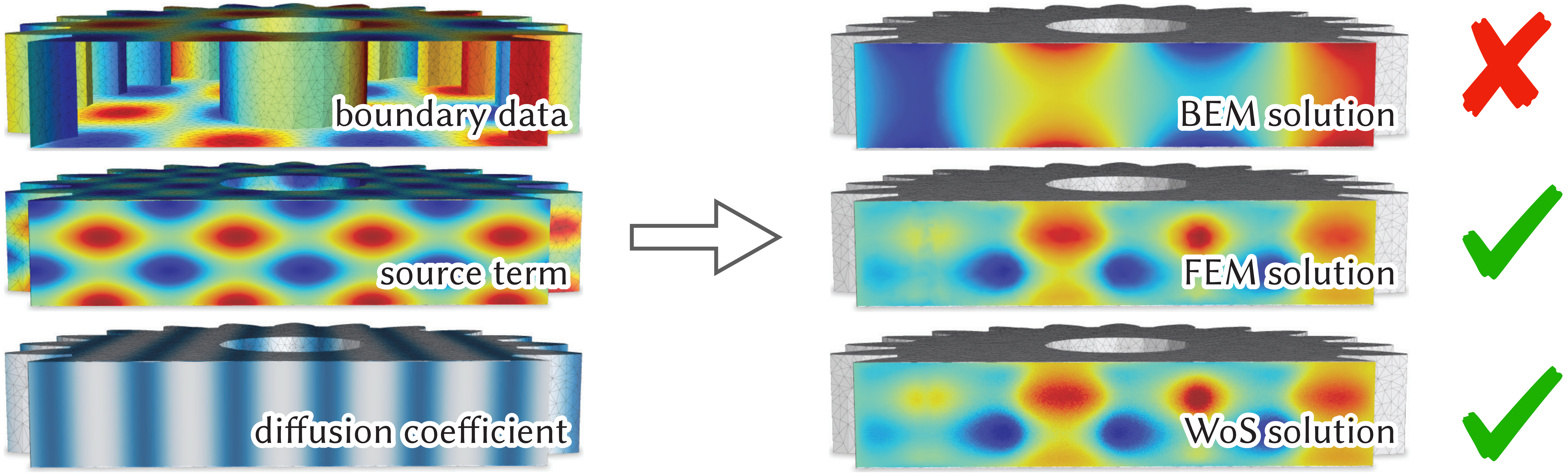
Source: Sharp and Crane, A Laplacian for Nonmanifold Triangle Meshes

Comparison with conventional solvers

The **boundary element method (BEM)** does not require volumetric meshing

Comparison with conventional solvers

The **boundary element method (BEM)** does not require volumetric meshing



BEM does not support problems with source terms or variable coefficients

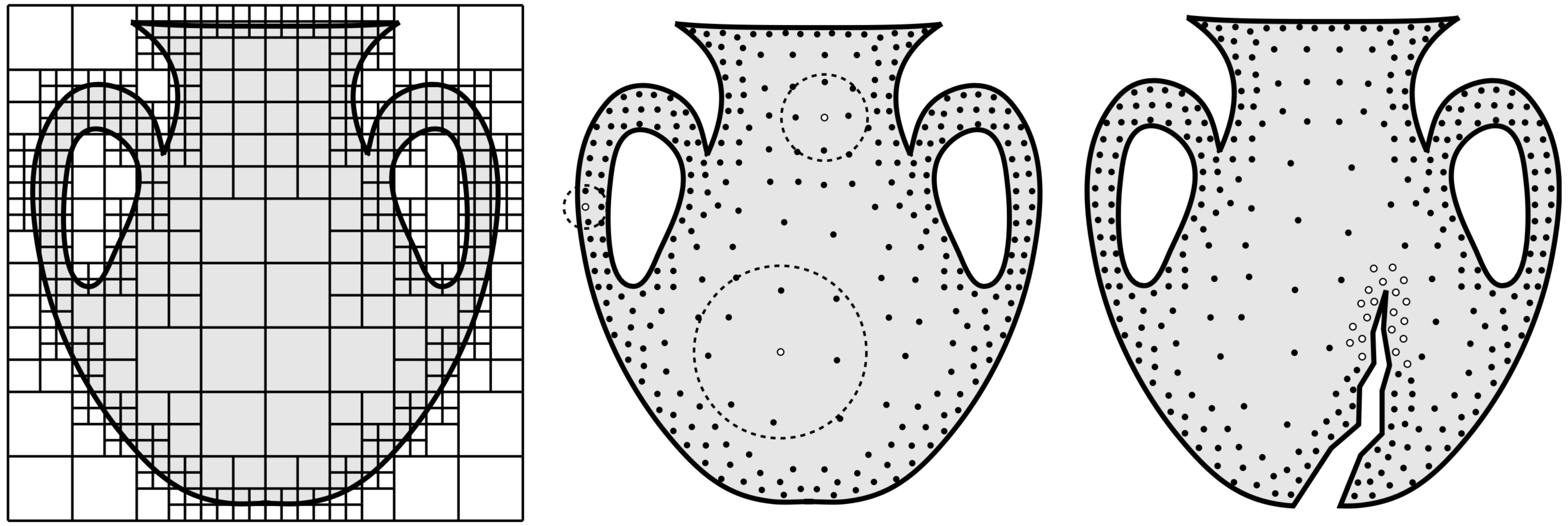
Comparison with conventional solvers

Meshless FEM solvers also do not require a volume mesh

Comparison with conventional solvers

Meshless FEM solvers also do not require a volume mesh

- Require dense sampling of the domain

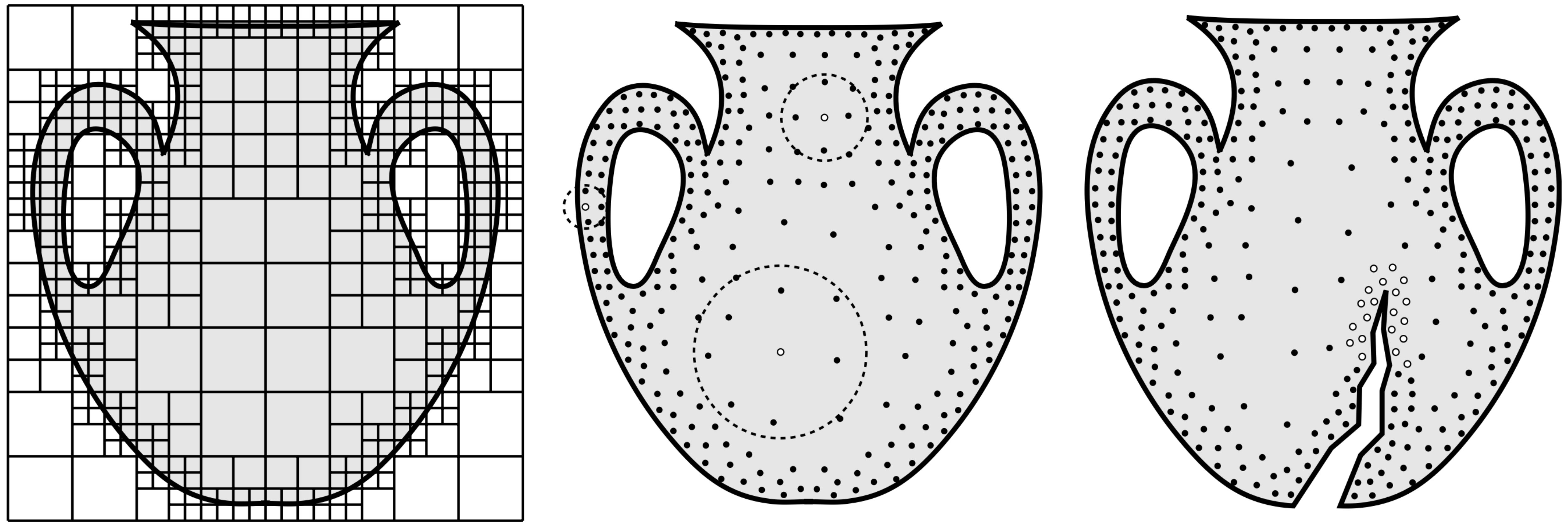


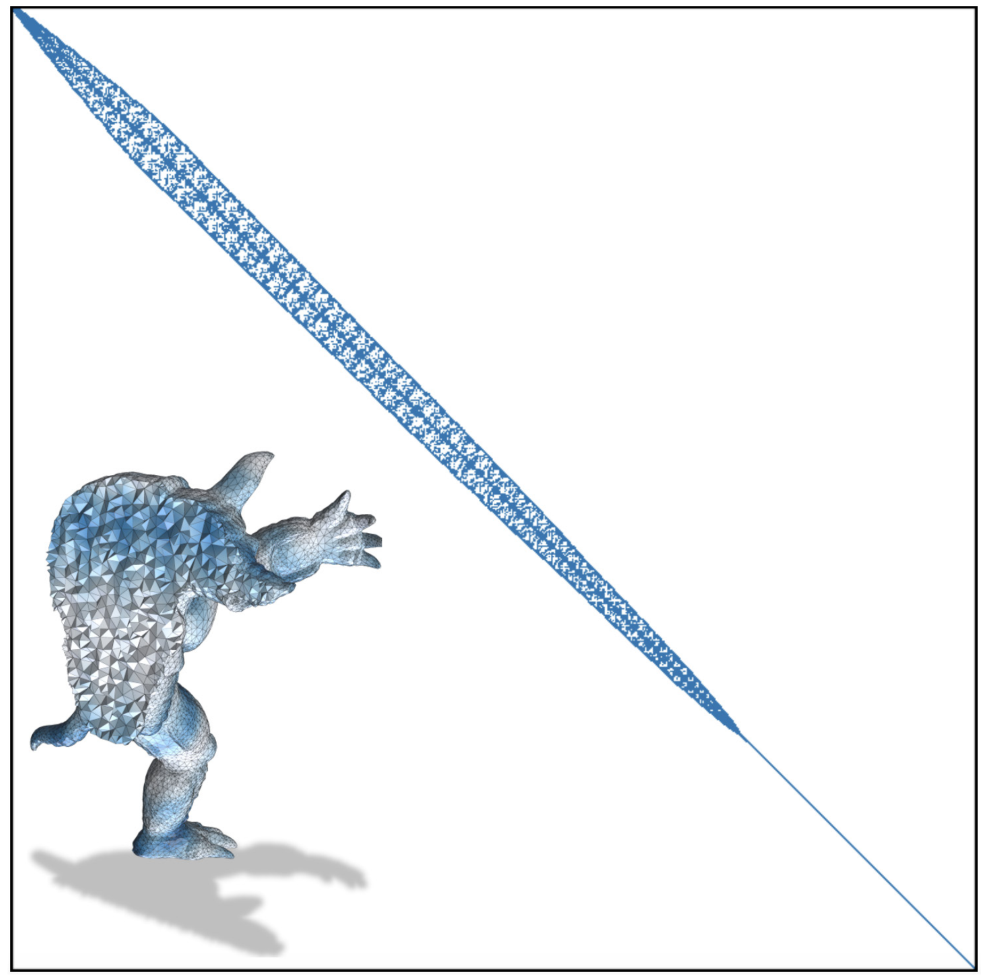
Comparison with conventional solvers

Meshless FEM solvers also do not require a volume mesh

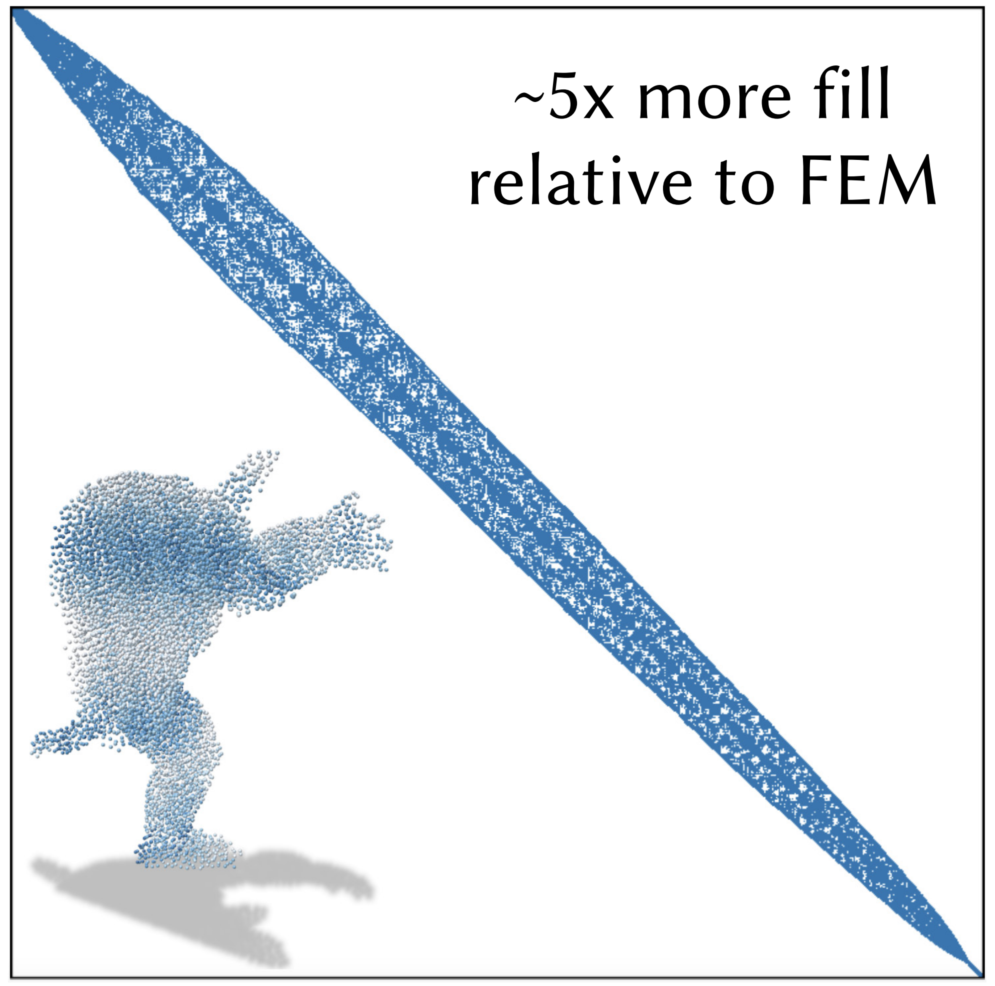
- Require dense sampling of the domain

- Require solving large linear systems





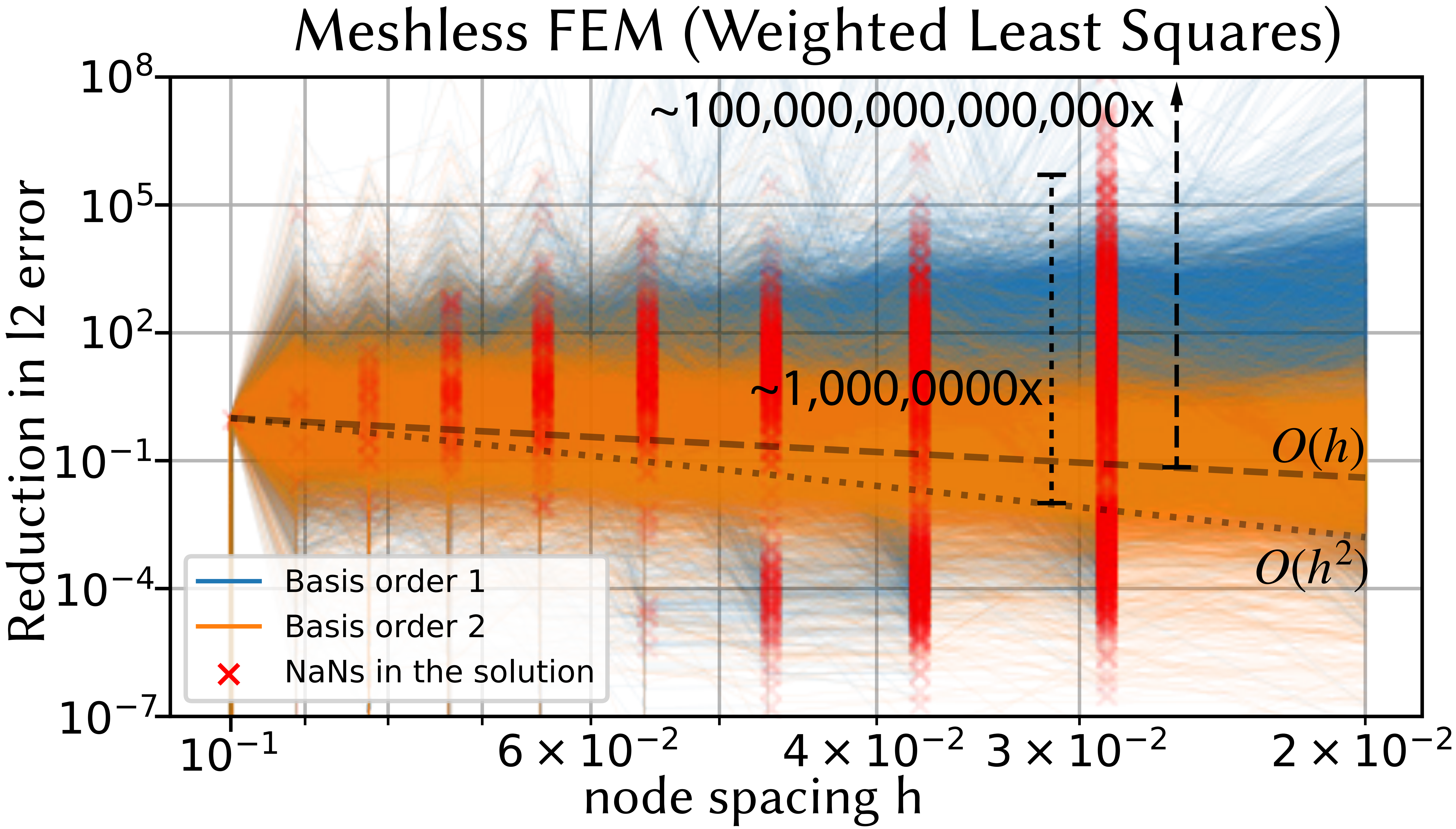
Typical FEM sparse matrix
(~25k vertices after reordering)



Typical Meshless FEM sparse matrix
(~25k nodes after reordering)

Meshless FEM is unreliable

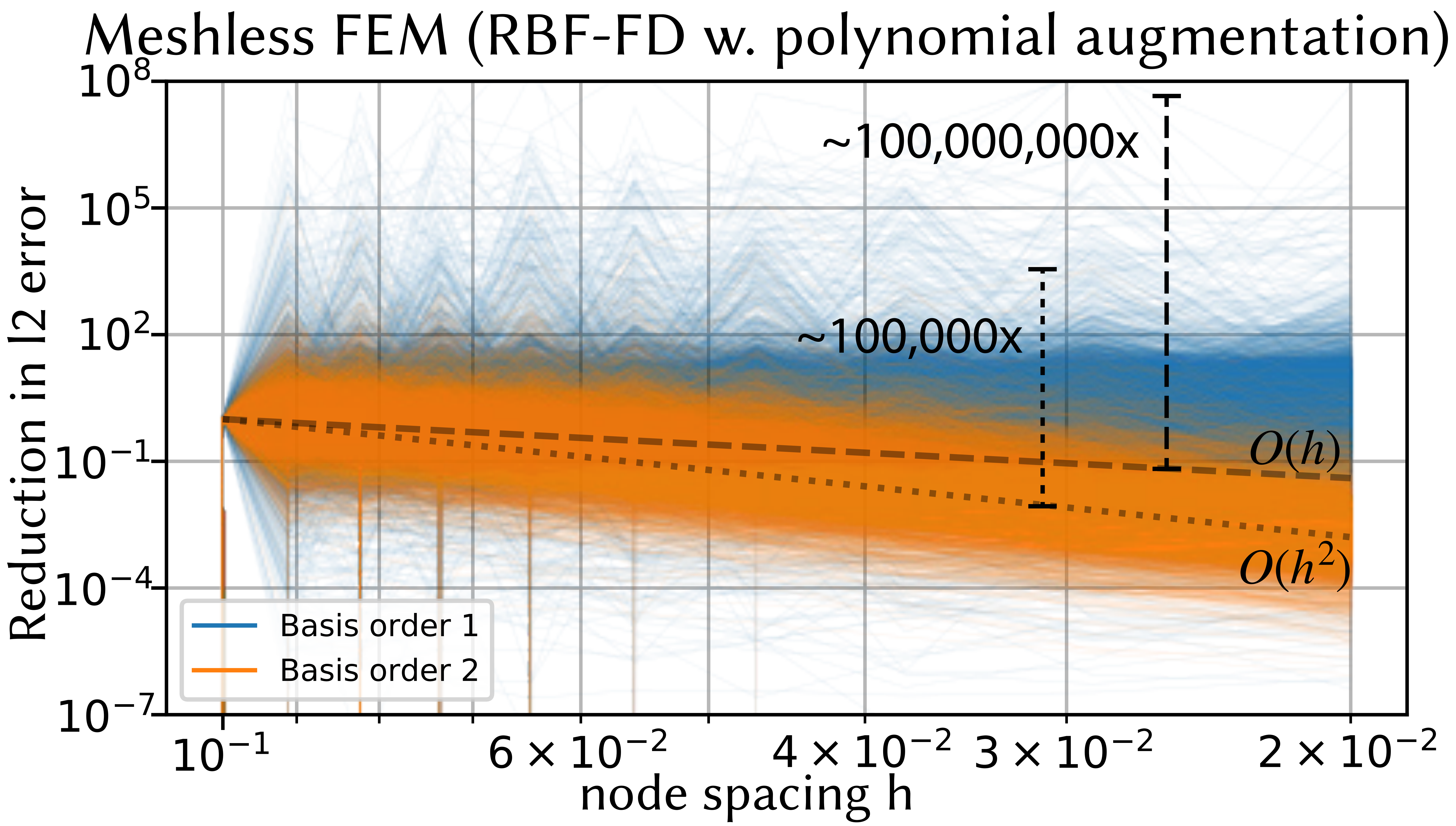
Solvers have unpredictable convergence under refinement



Tested on **10k models** from the Thingi10k dataset

Meshless FEM is unreliable

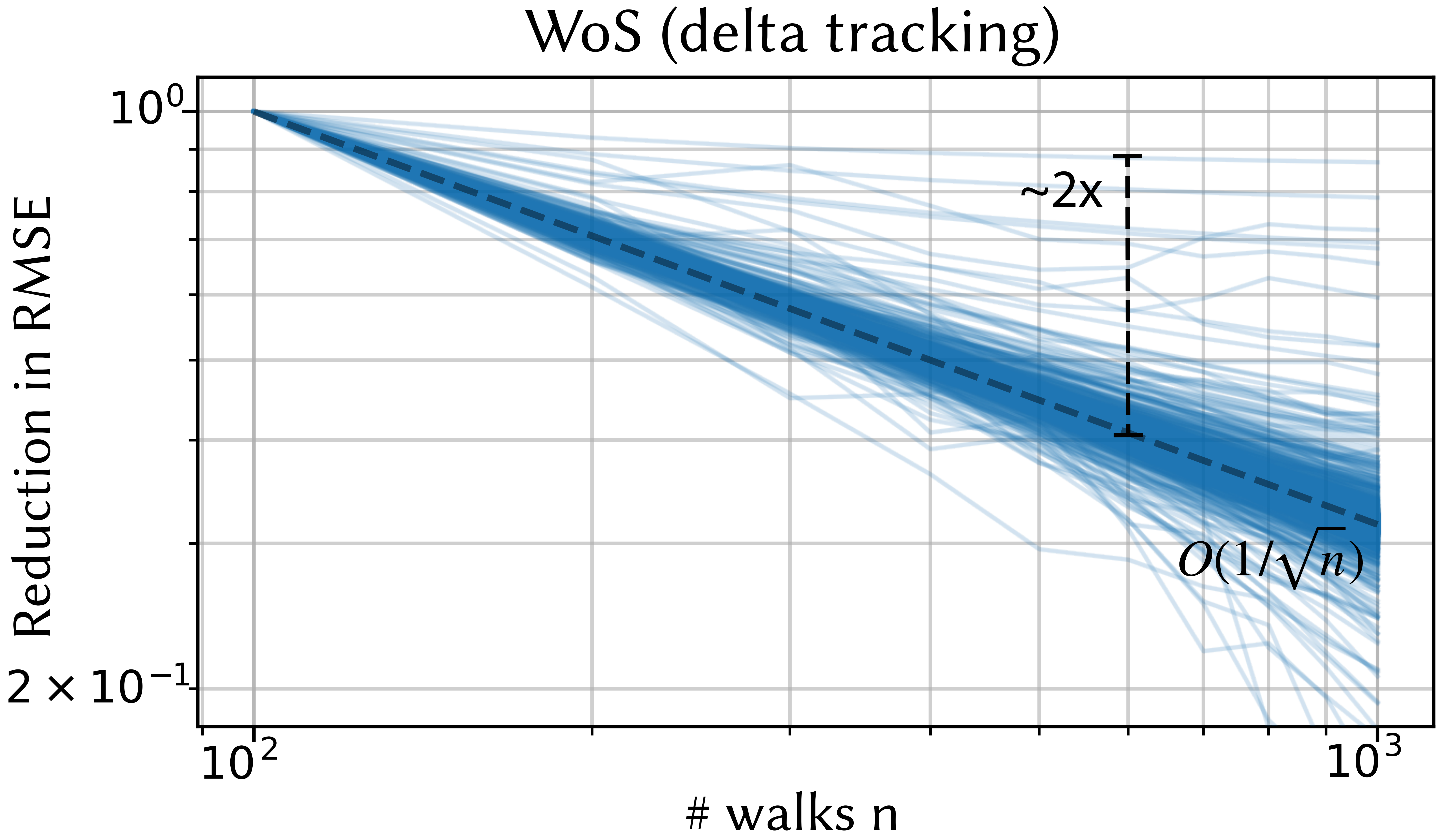
Solvers have unpredictable convergence under refinement



Tested on **10k models** from the Thingi10k dataset

Meshless FEM is unreliable

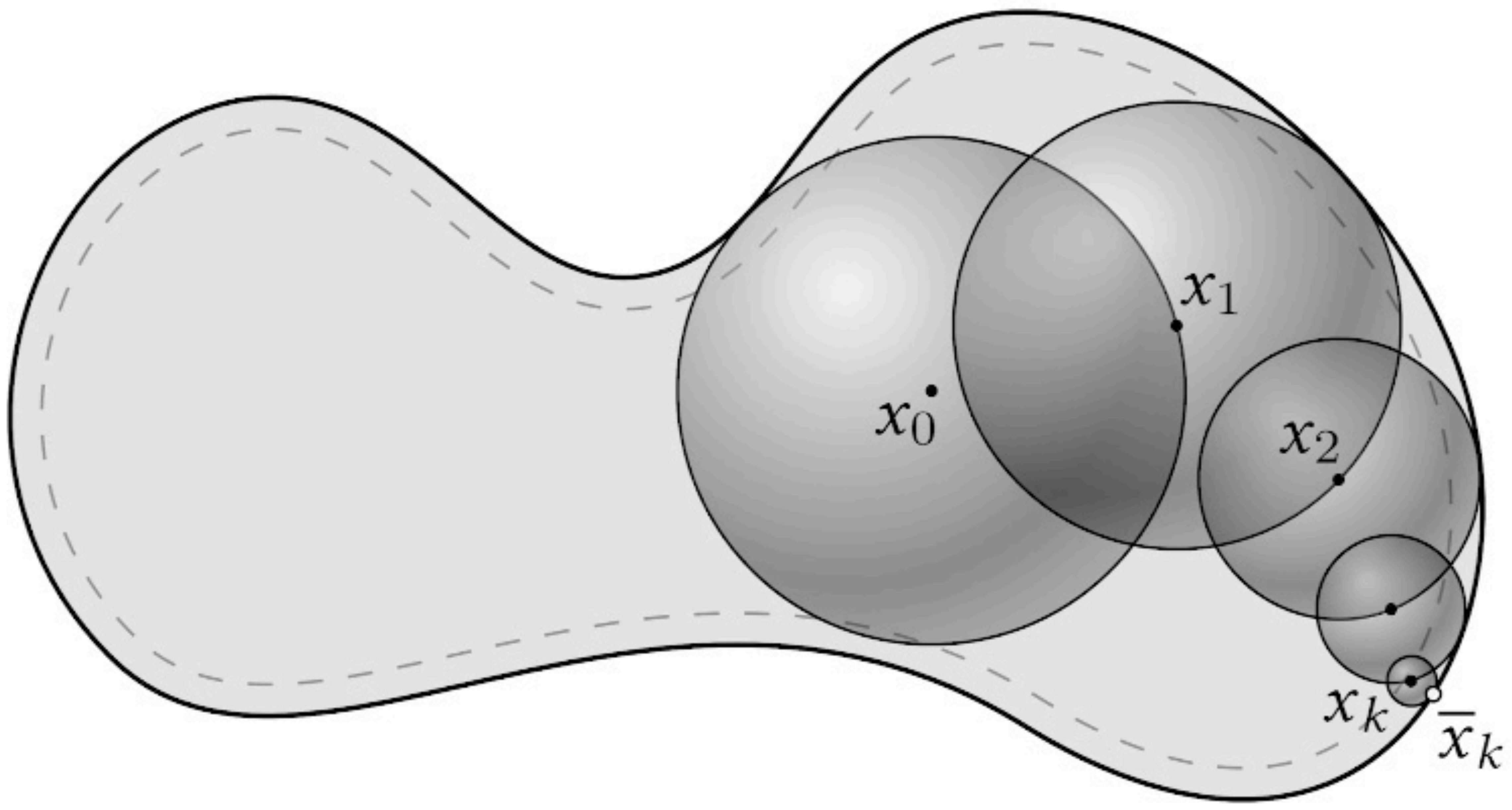
Walk on spheres converges predictably



Tested on **10k models** from the Thingi10k dataset

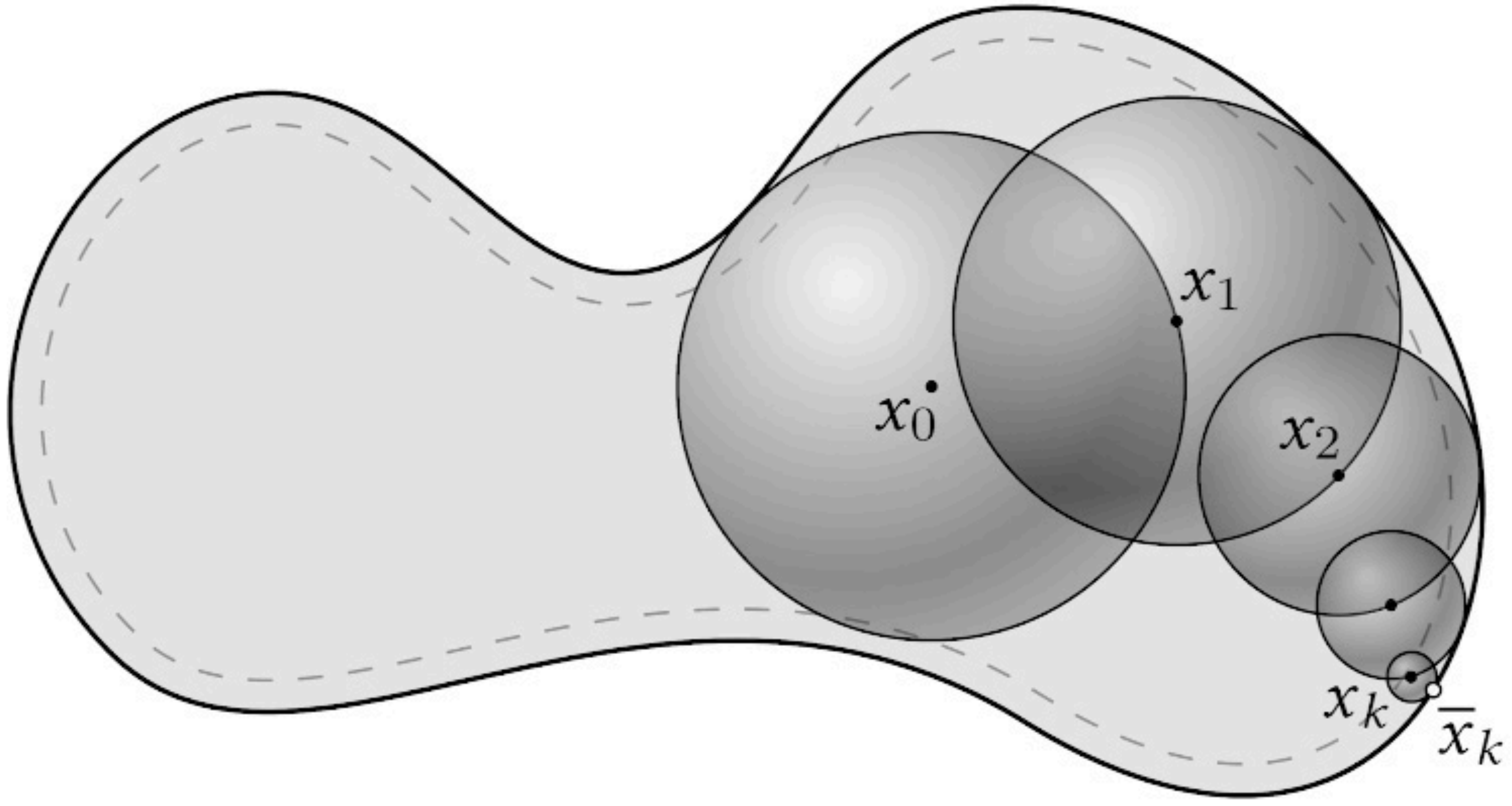
Stopping tolerance ε

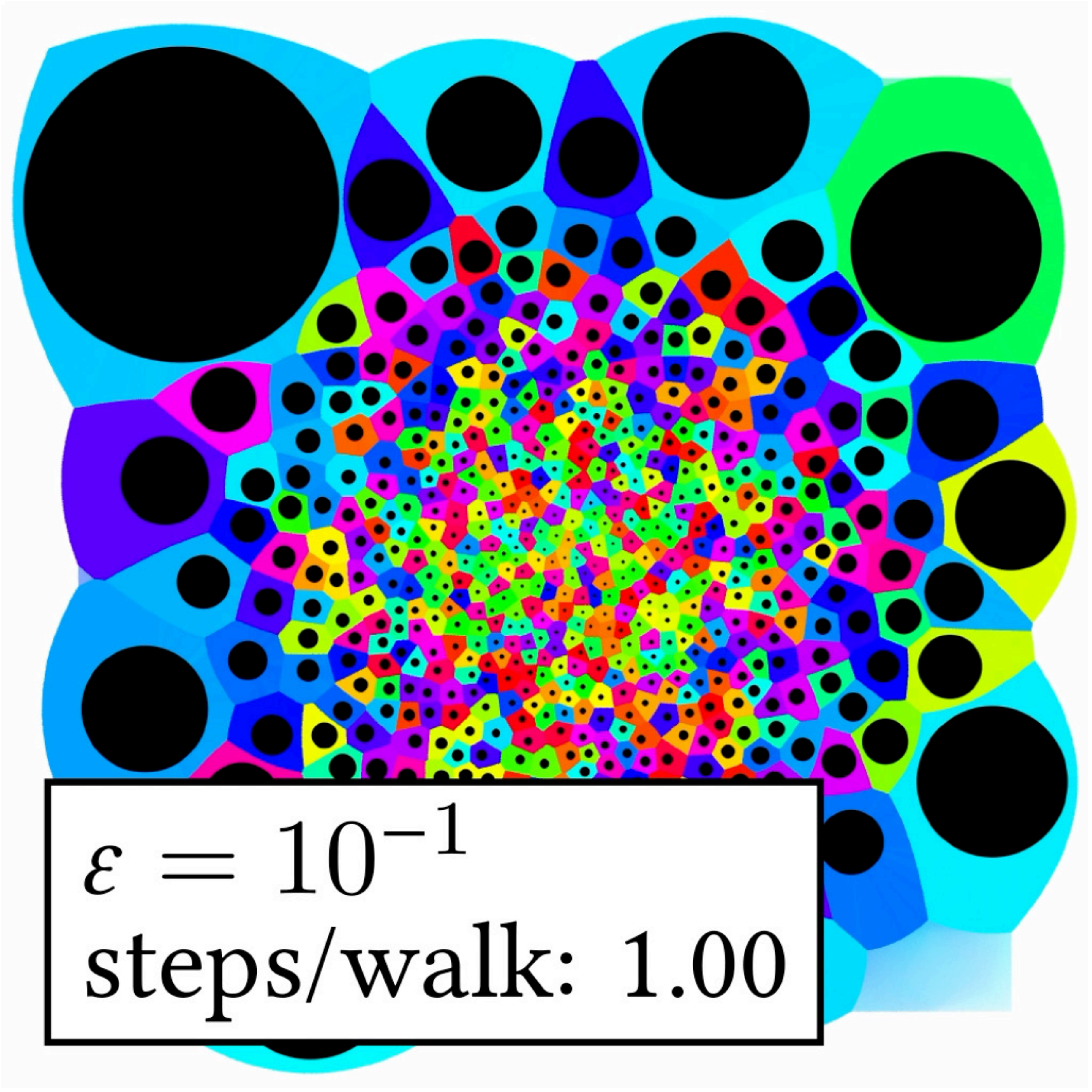
Introduces minimal bias and has little impact on performance



Stopping tolerance ε

Introduces minimal bias and has little impact on performance



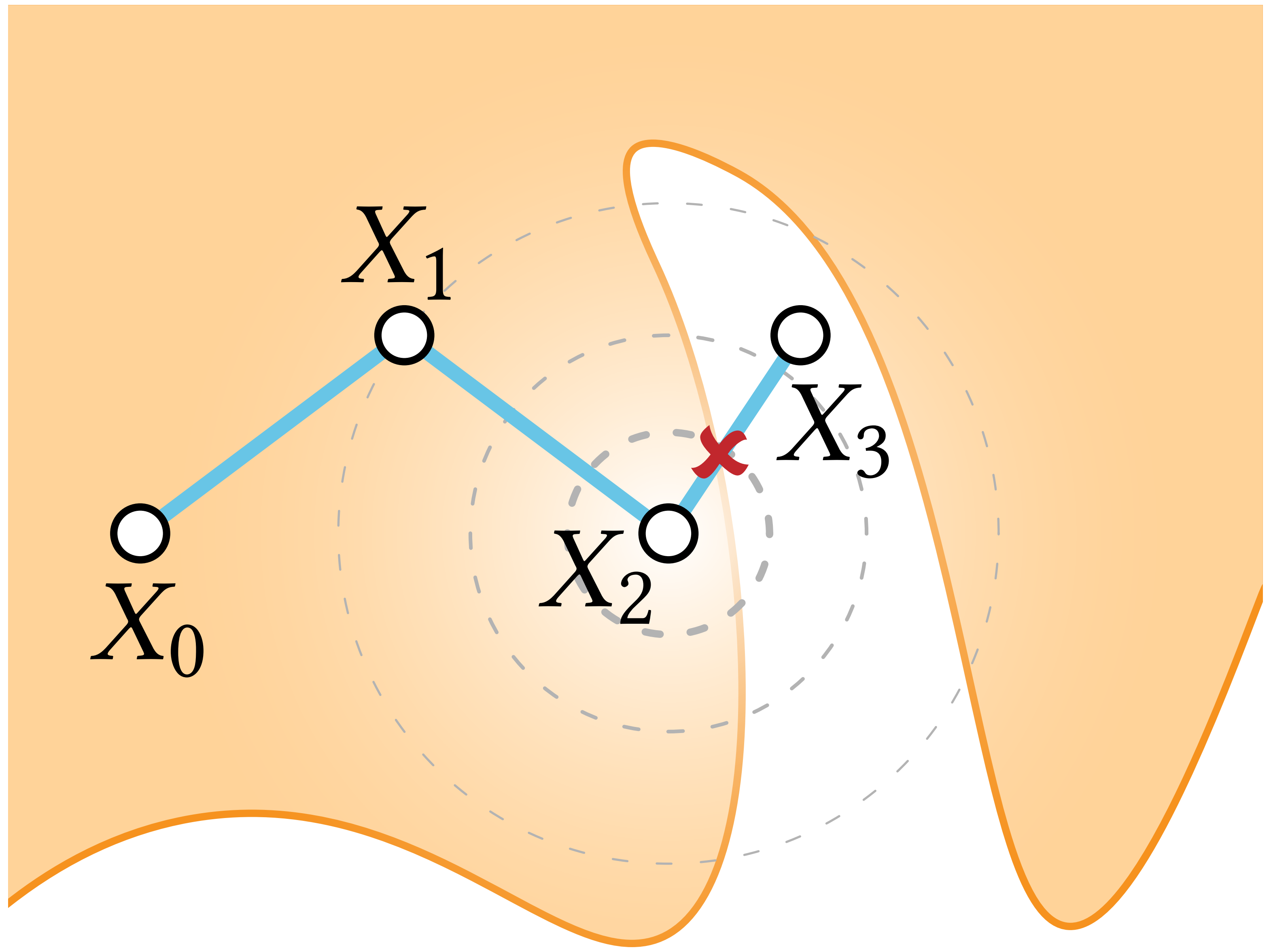


Discretized random walks

Explicit time stepping of diffusion process: $X_{k+1} = X_k + \vec{\omega}(X_k)\ h + \sqrt{\alpha(X_k)}\ (W_{k+1} - W_k)$

Discretized random walks

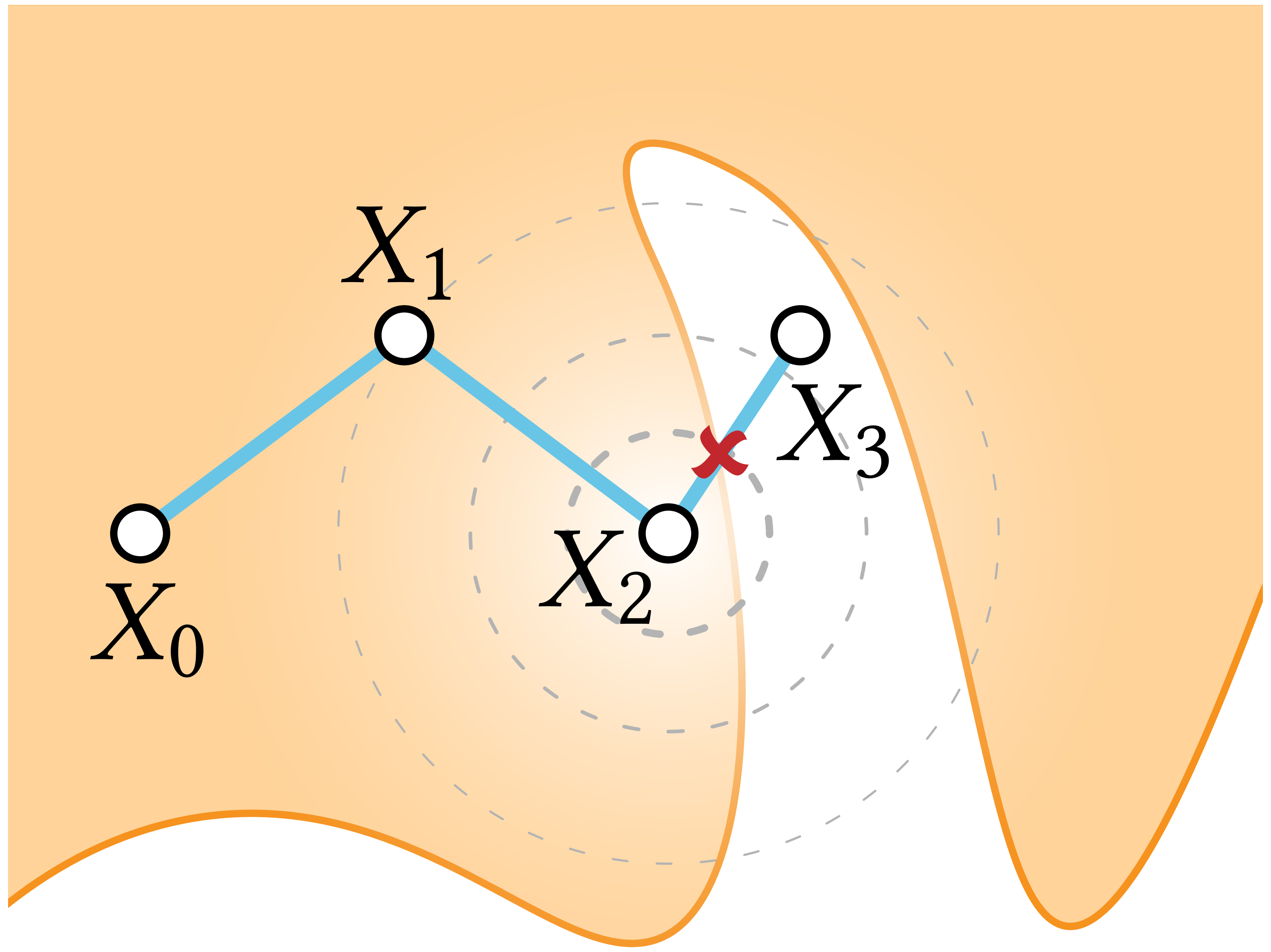
Explicit time stepping of diffusion process: $X_{k+1} = X_k + \vec{\omega}(X_k)\, h + \sqrt{\alpha(X_k)}\, (W_{k+1} - W_k)$



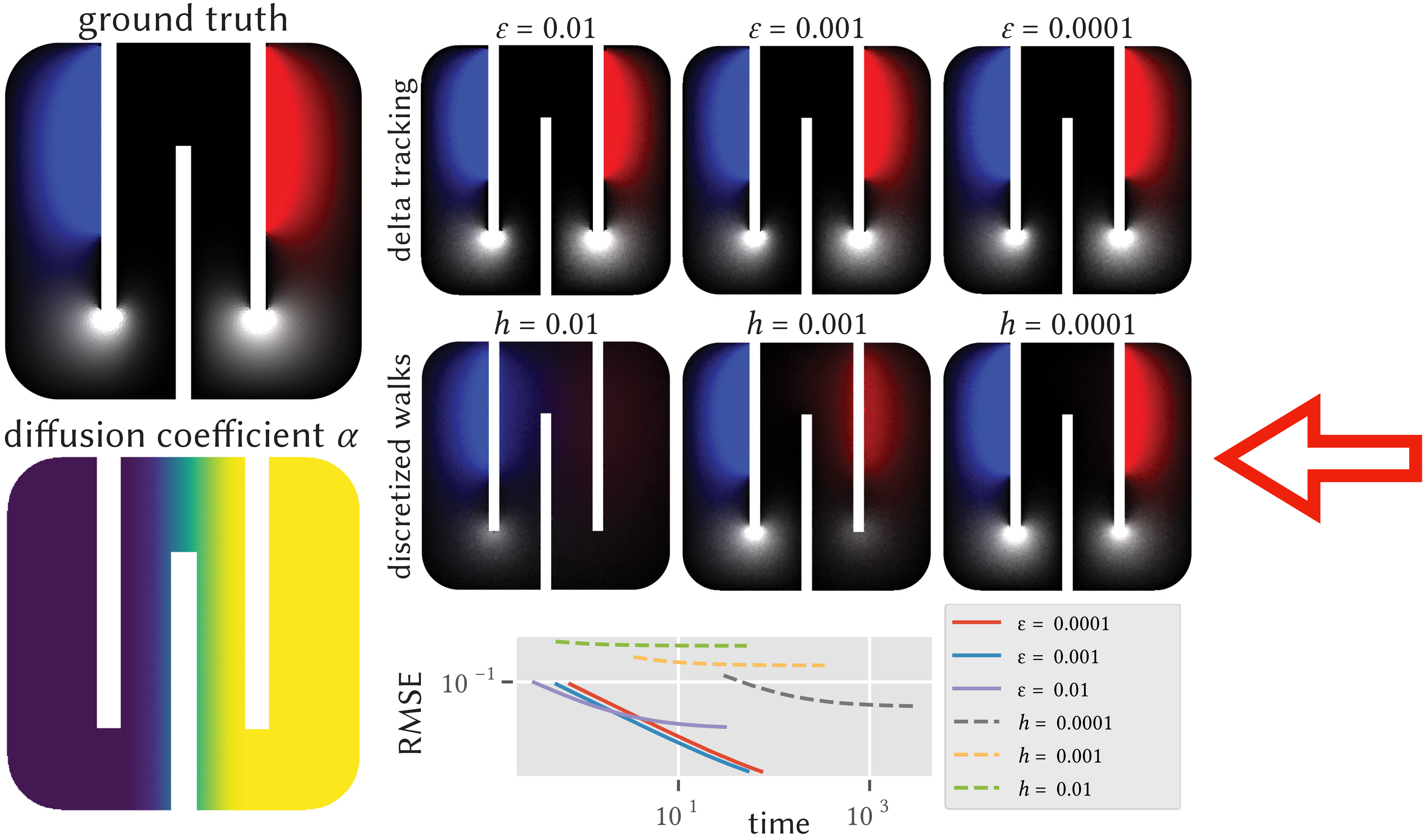
Discretized walks can leave the domain, biasing estimates

Discretized random walks

Explicit time stepping of diffusion process: $X_{k+1} = X_k + \vec{\omega}(X_k)\, h + \sqrt{\alpha(X_k)}\, (W_{k+1} - W_k)$

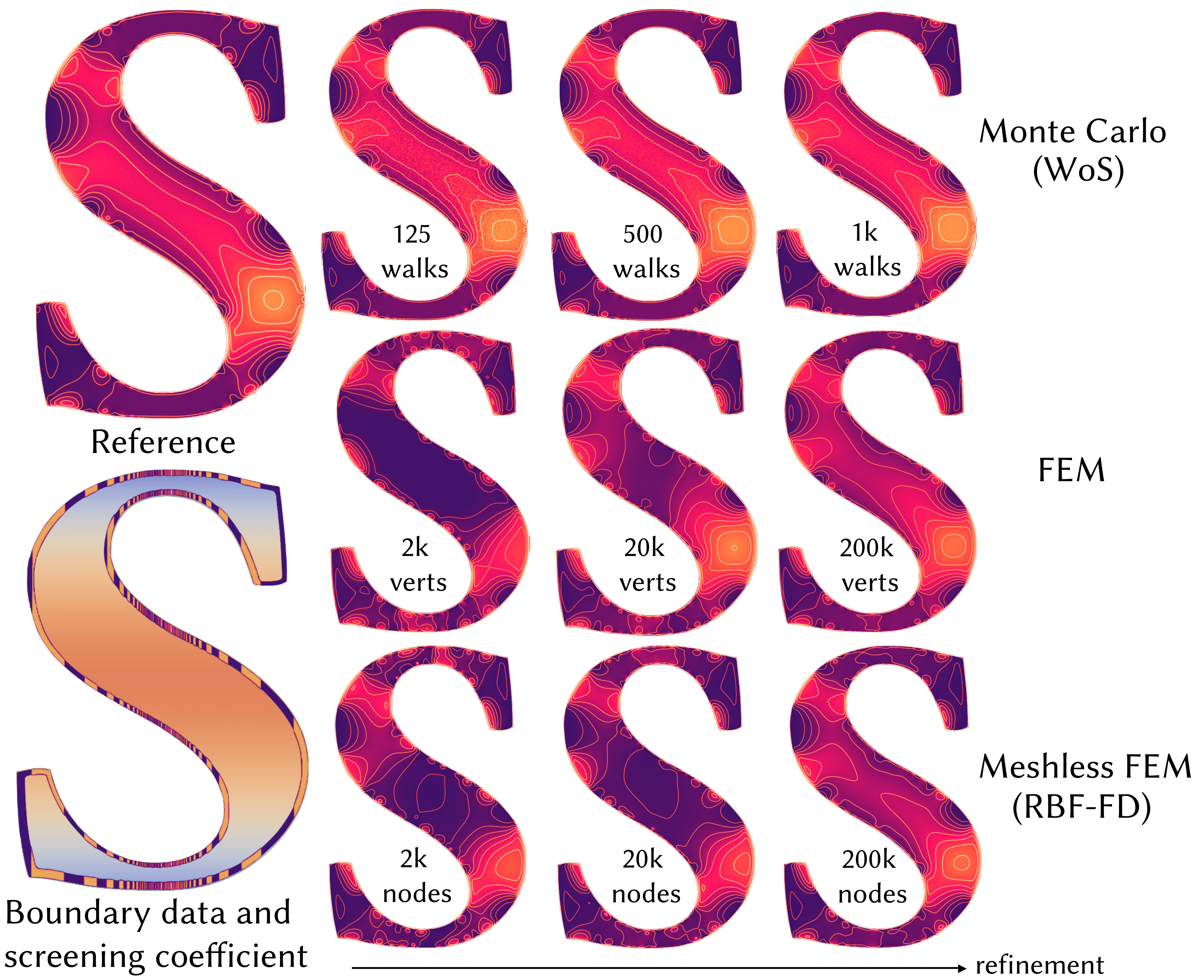


Discretized walks can leave the domain, biasing estimates



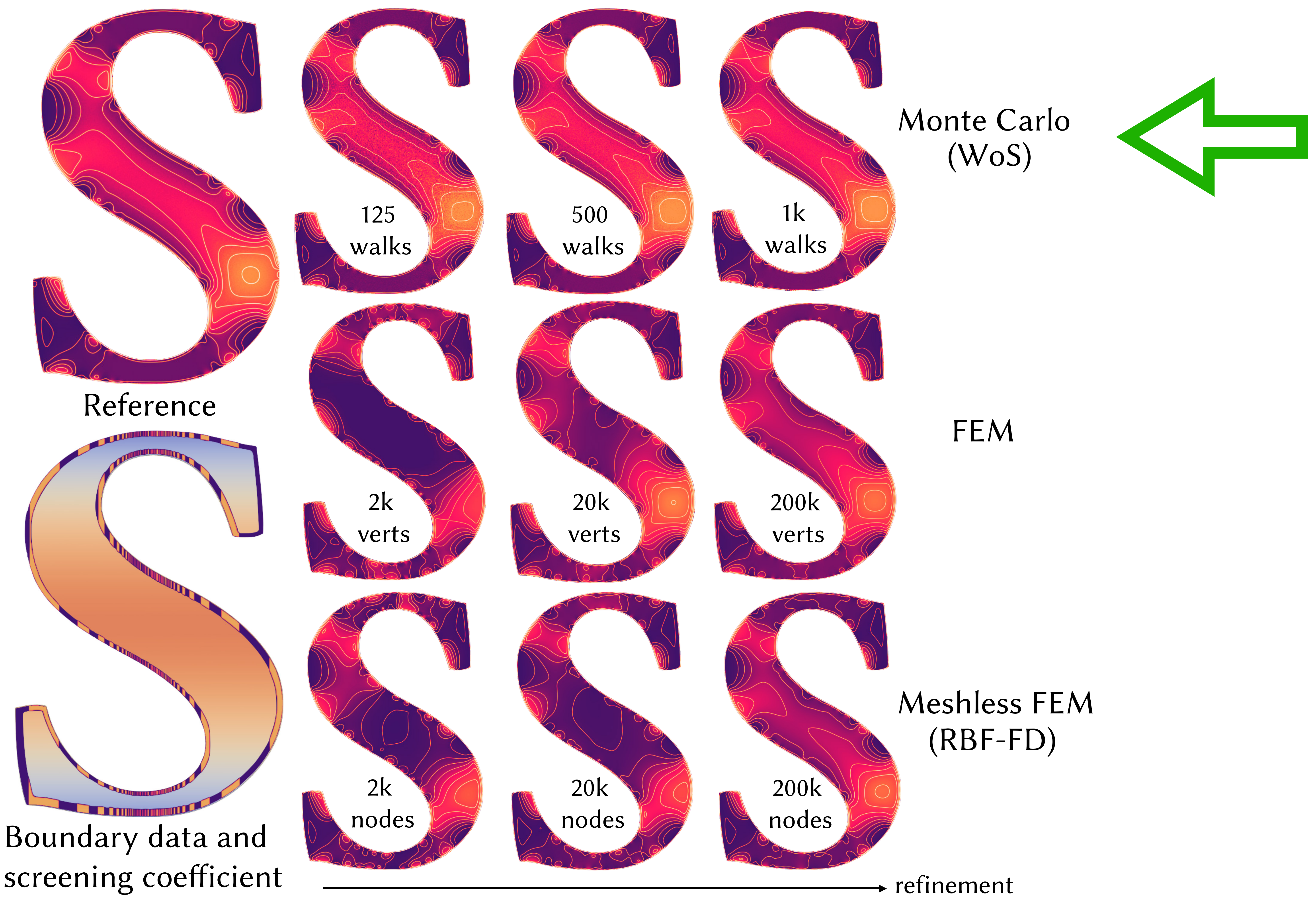
No spatial aliasing

Monte Carlo decouples boundary conditions/coefficients from geometry



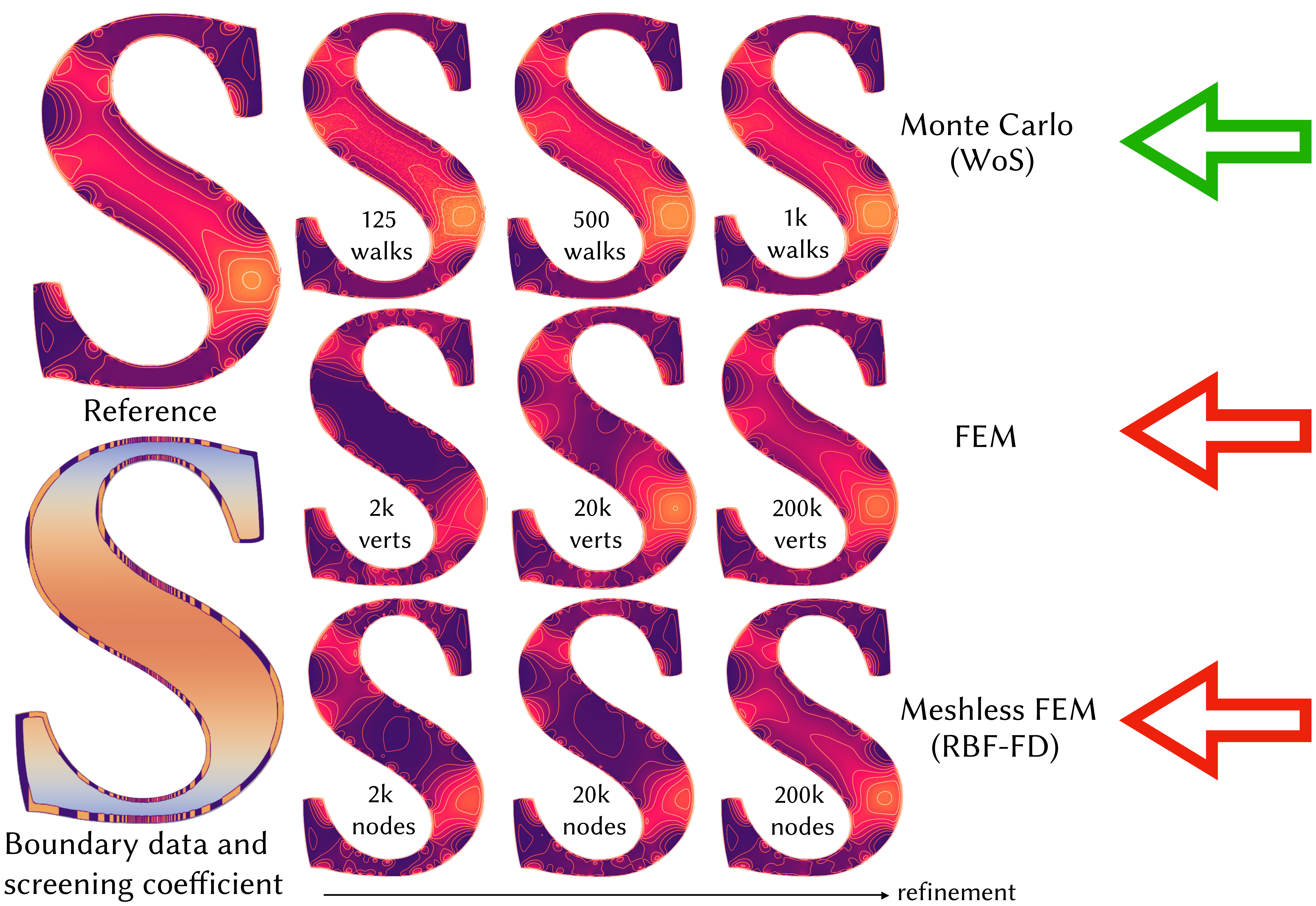
No spatial aliasing

Monte Carlo decouples boundary conditions/coefficients from geometry



No spatial aliasing

Monte Carlo decouples boundary conditions/coefficients from geometry

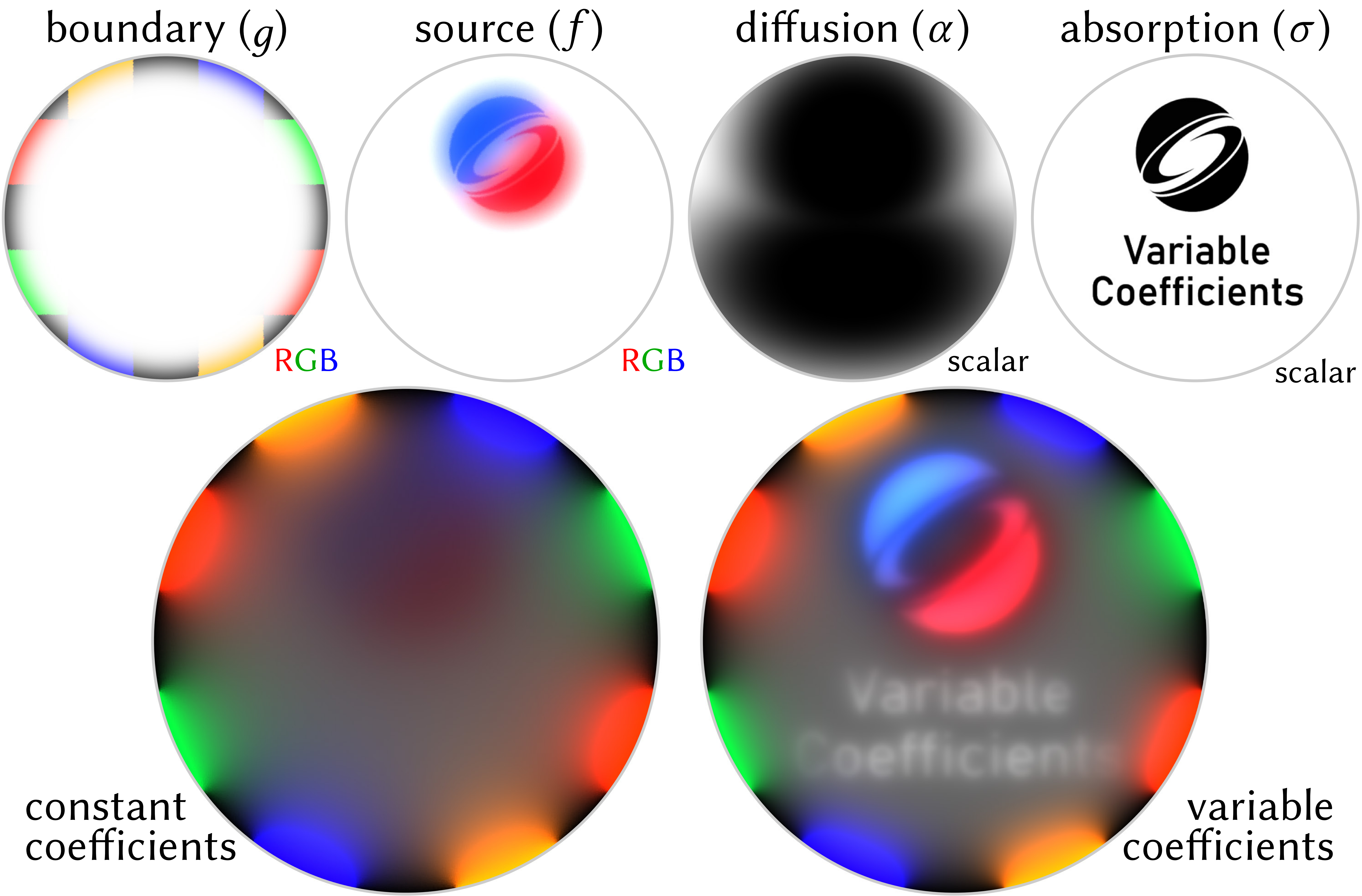


Physical analysis of complex geometry & materials

No homogenization of PDE coefficients!

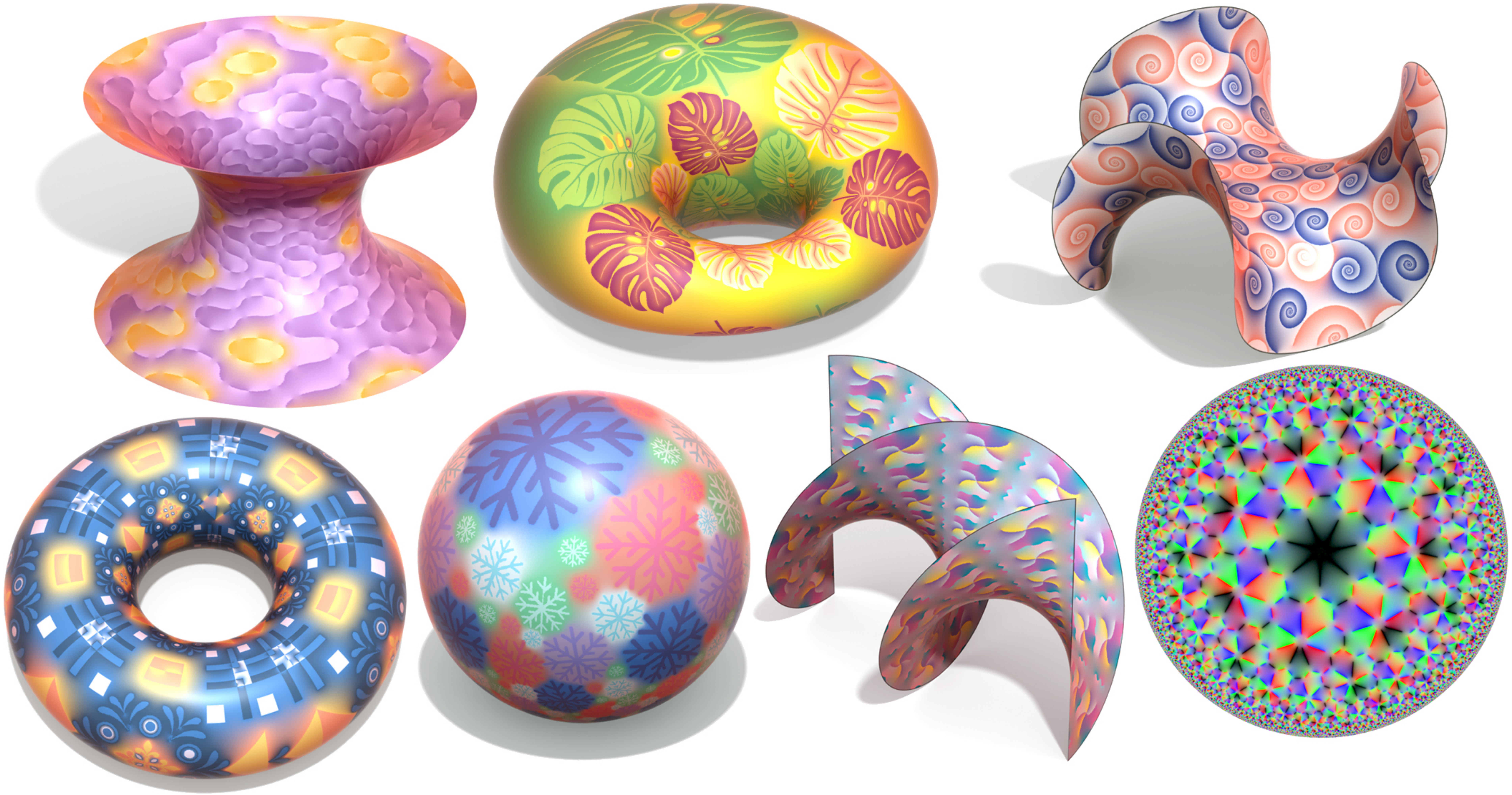
Example application: variable coefficient diffusion curves

Additional control over sharp details



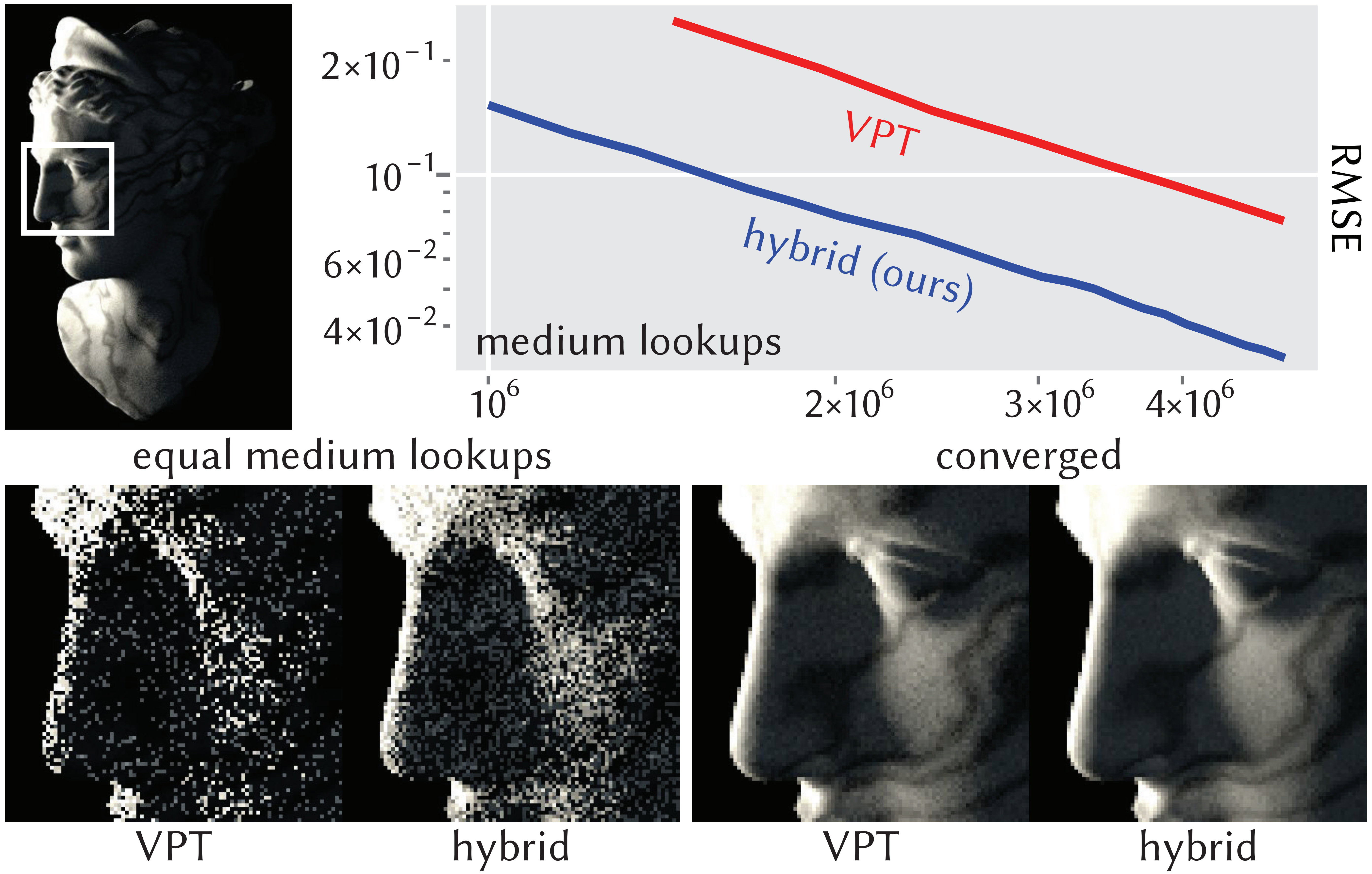
Example application: diffusion curves on surfaces

Use variable coeffs on flat domains to model constant coeffs on curved domains



Example application: subsurface scattering

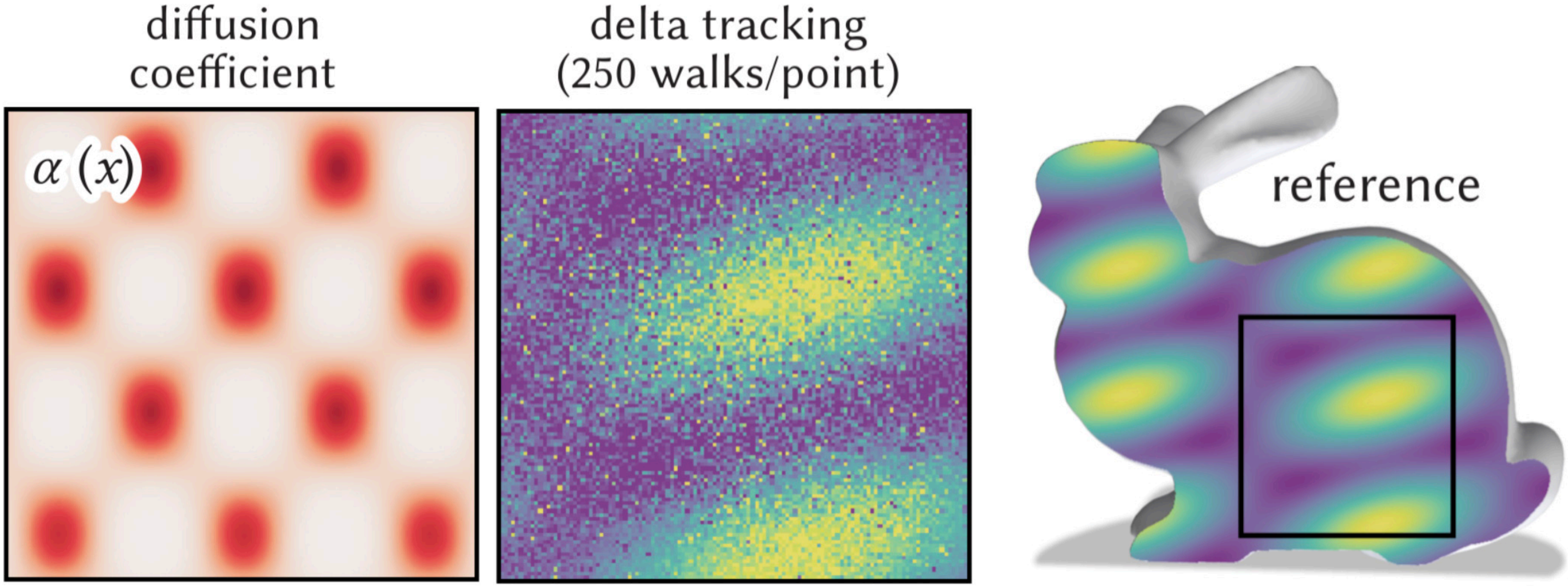
Easy to mix volumetric path tracing (VPT) and walk on spheres (WoS)



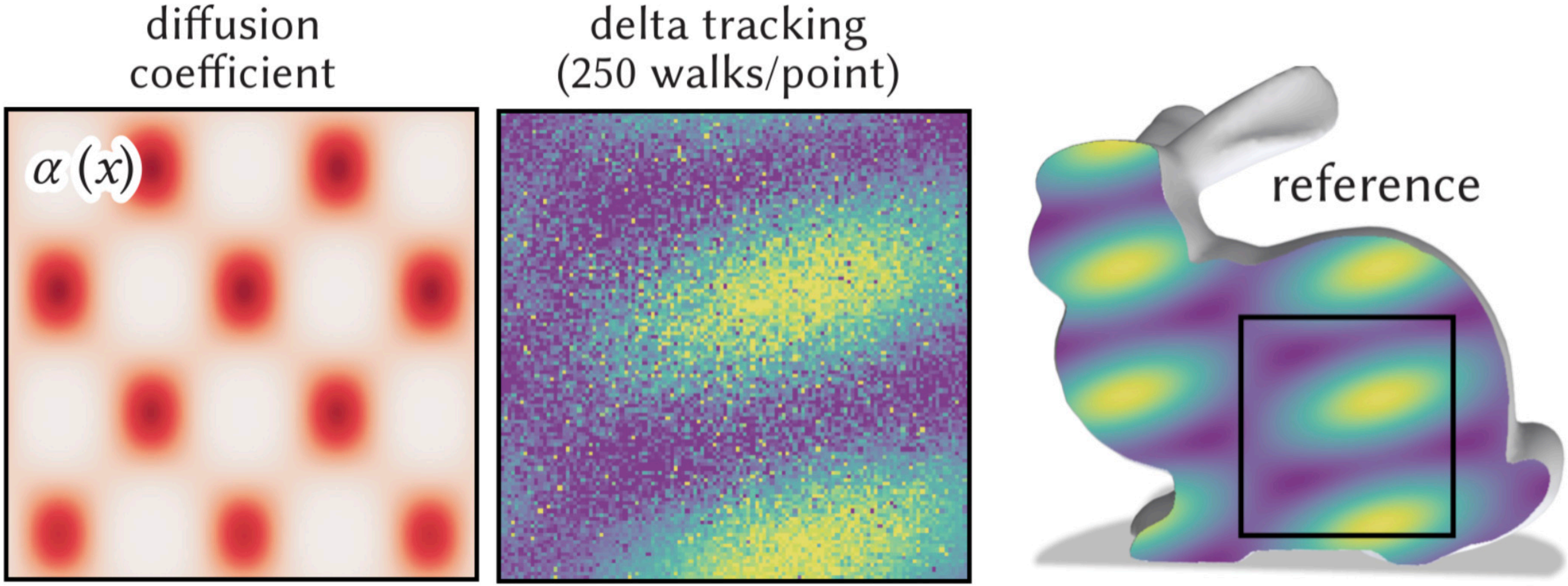
Hybrid strategy : VPT near boundary, WoS deeper inside volume

Limitations & Future Work

High variance due to large spatial variation



High variance due to large spatial variation



Future: local coefficient bounds, low-variance VRE estimators, adaptive weight window

Future: support for important features

Neumann & Robin boundary conditions

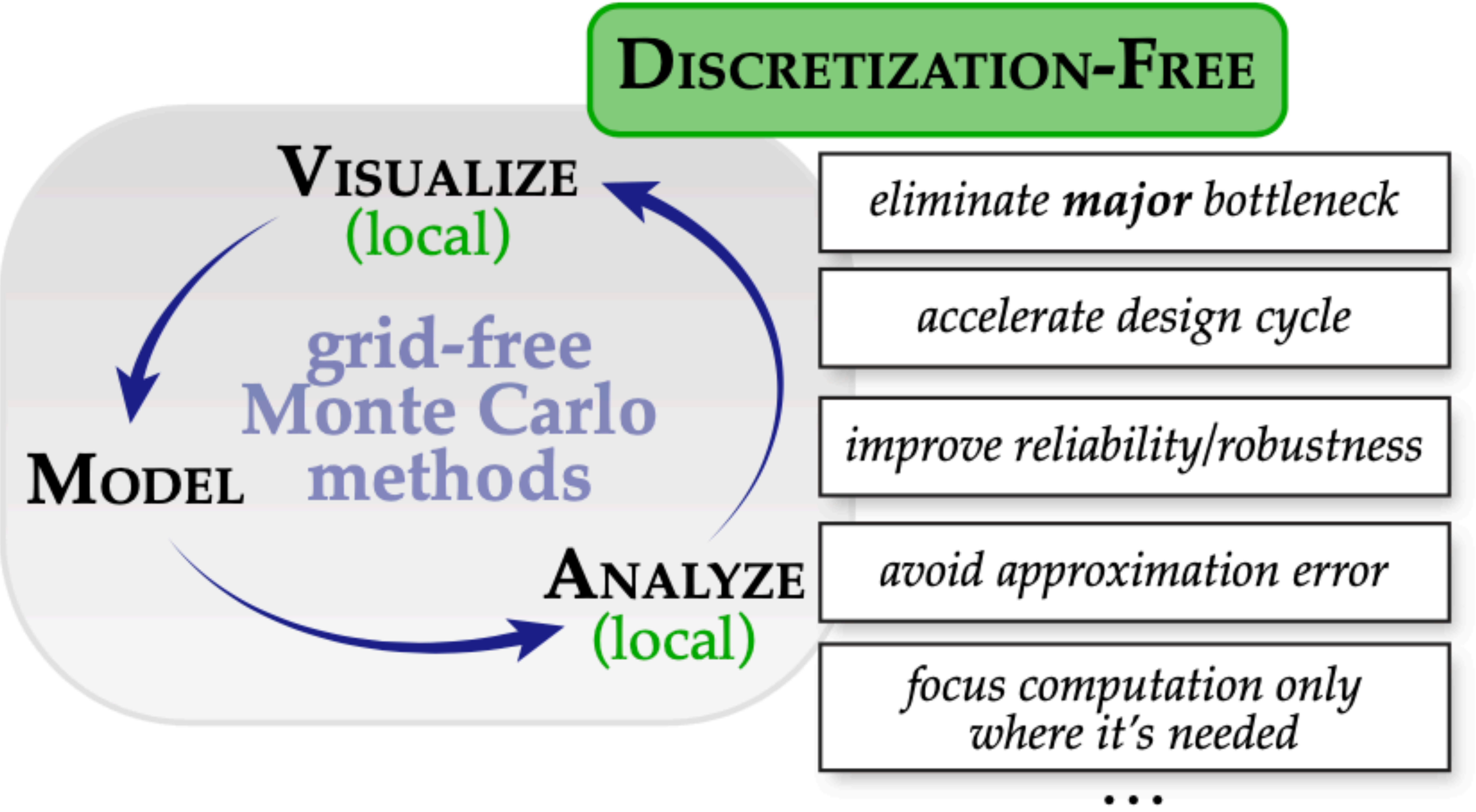
Anisotropic diffusion coefficients

Non-linear PDEs

High performance distance queries

Differentiable implementation

The promise of grid-free Monte Carlo



GRID-FREE MONTE CARLO FOR PDES WITH SPATIALLY VARYING COEFFICIENTS

ROHAN SAWHNEY*, CARNEGIE MELLON UNIVERSITY
DARIO SEYB*, DARTMOUTH COLLEGE
WOJCIECH JAROSZ†, DARTMOUTH COLLEGE
KEENAN CRANE†, CARNEGIE MELLON UNIVERSITY

The symbols * and † indicate equal contribution.

BACKUP

I don’t know…