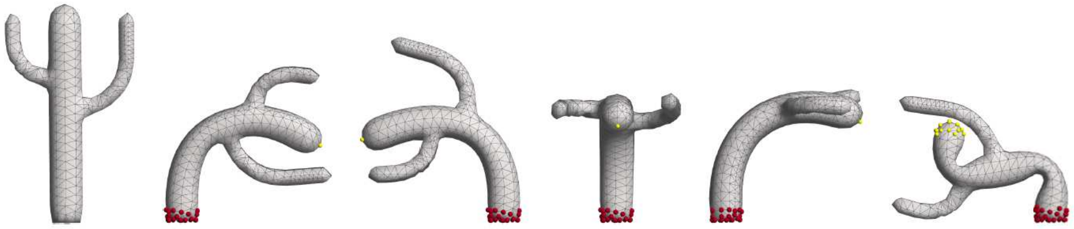
Computer Animation & Physical Simulation

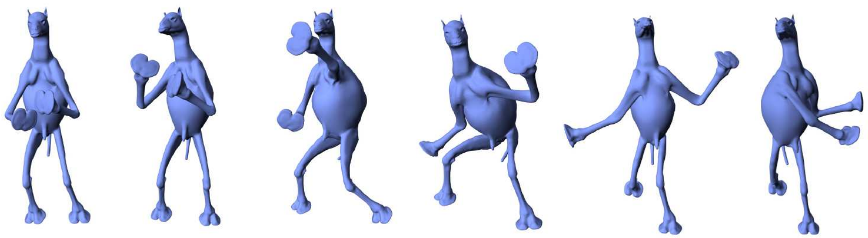
Lecture 5: Preliminaries for Physically-Based Animation

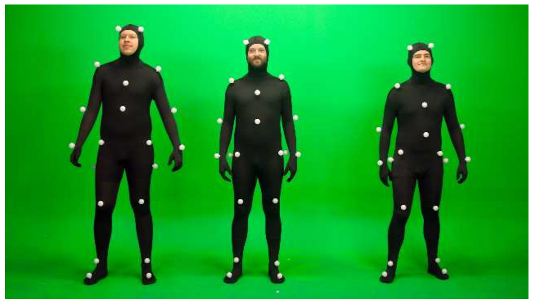
XIAOPEI LIU
School of Information Science and Technology
ShanghaiTech University

Limitations of Non-Physically-Based Animation

- **Manual adjustment**
  - Tedious and labor intensive
  - Cannot cover wide enough animations



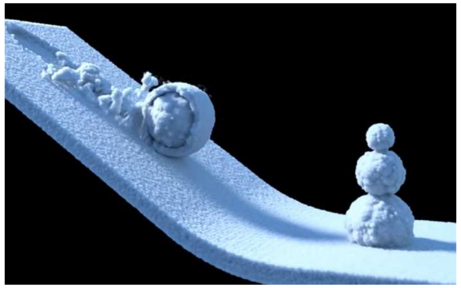




Physically-Based Animation

- **A simulation of real physical system**
  - Usually involve physical laws
  - Solve physical dynamic equations





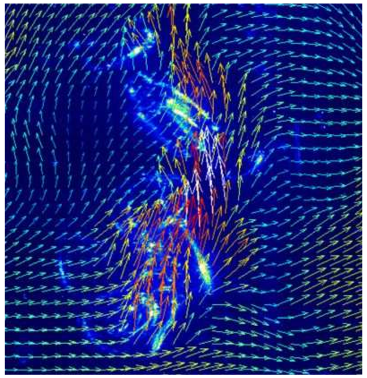
How to do physically-based animation?

- **Modeling**
  - The dynamic process described by partial differential equations
  - Particle dynamics $$\frac{d}{dt}\mathbf{Y}(t) = \frac{d}{dt}\begin{pmatrix} x(t) \\ v(t) \end{pmatrix} = \begin{pmatrix} v(t) \\ F(t)/m \end{pmatrix}$$
  - Rigid body dynamics $$\begin{aligned} m\dot{\mathbf{v}} &= \mathbf{f} \\ \mathbf{I}\dot{\omega} + \omega \times \mathbf{I}\omega &= \tau \end{aligned}$$
  - Soft-body dynamics $$\rho\,\ddot{\mathbf{u}} + \mathbf{f}_d(\dot{\mathbf{u}}) - \frac{\partial W_{int}(\mathbf{u})}{\partial \mathbf{u}} = \mathbf{f}_{ext}$$
  - Fluid dynamics $$\rho\left(\frac{\partial \mathbf{u}}{\partial t} + \mathbf{u}\cdot\nabla\mathbf{u}\right) = -\nabla p + \mu\nabla^2\mathbf{u} + \mathbf{g}$$ $$\nabla\cdot\mathbf{u} = 0$$

How to do physically-based animation?

- **Simulation (model solving)**
  - Analytical solutions are rare
  - Numerically solve the model equations

$$\rho\left(\frac{\partial \mathbf{u}}{\partial t} + \mathbf{u}\cdot\nabla\mathbf{u}\right) = -\nabla p + \mu\nabla^2\mathbf{u} + \mathbf{g}$$
$$\nabla\cdot\mathbf{u} = 0$$

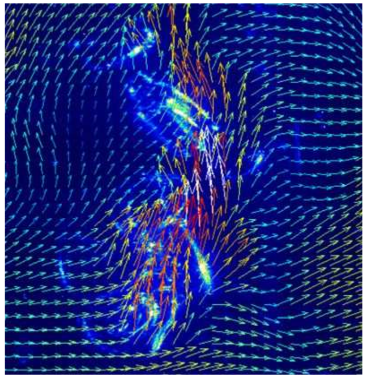


How to do physically-based animation?

- **Rendering**
  - Perform graphical rendering based on the simulated data

$$\rho\left(\frac{\partial \mathbf{u}}{\partial t} + \mathbf{u}\cdot\nabla\mathbf{u}\right) = -\nabla p + \mu\nabla^2\mathbf{u} + \mathbf{g}$$

$$\nabla\cdot\mathbf{u} = 0$$



I. Partial Differential Equations

Differential Equation

- **A differential equation**
  - Contain unknown single-/multi-variable functions and their (partial) derivatives
  - Formulate problems involving functions of single/several variables
  - Describe a wide variety of phenomena
    - e.g., sound, heat, electrodynamics, fluid dynamics, elasticity, or quantum mechanics

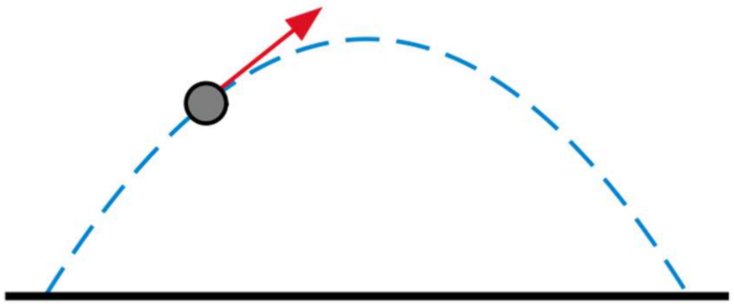
Ordinary Differential Equation

- **A Differential Equation**
  - Contain one or more functions of one independent variable and its derivatives
  - General form

$$a_0(x)y + a_1(x)y' + a_2(x)y'' + \cdots + a_n(x)y^{(n)} + b(x) = 0$$

  - A simple example

$$m\frac{\mathrm{d}^2 x(t)}{\mathrm{d}t^2} = F(x(t))$$

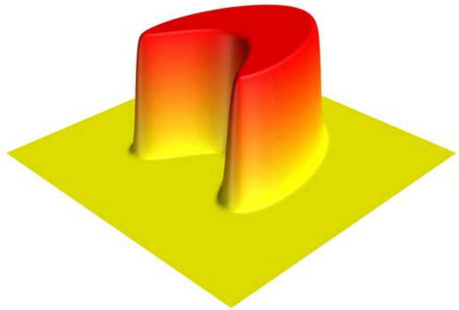


Partial Differential Equation

- **A Differential Equation**
  - Contain unknown multivariable functions and their partial derivatives
  - General form

$$f\left(x_1, \ldots, x_n, u, \frac{\partial u}{\partial x_1}, \ldots, \frac{\partial u}{\partial x_n}, \frac{\partial^2 u}{\partial x_1 \partial x_1}, \ldots, \frac{\partial^2 u}{\partial x_1 \partial x_n}, \ldots\right) = 0$$

  - A simple example



$$\frac{\partial \phi(\mathbf{r}, t)}{\partial t} = \nabla \cdot \left[D(\phi, \mathbf{r}) \, \nabla \phi(\mathbf{r}, t)\right]$$

Partial Differential Equation

- **Typical PDEs**
  - Diffusion equation (heat conduction)

$$\frac{\partial \phi(\mathbf{r},t)}{\partial t} = \nabla \cdot \left[D(\phi,\mathbf{r})\, \nabla \phi(\mathbf{r},t) \right]$$

  - Advection equation (flow problem)

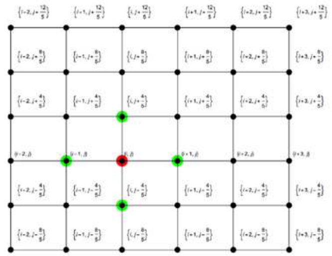
$$\frac{\partial \psi}{\partial t} + \nabla \cdot (\psi \mathbf{u}) = 0$$

  - Poisson equation (steady-state distribution)

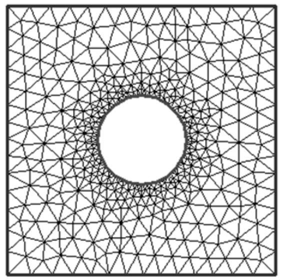
$$\nabla^2 \varphi = f$$

How to numerically solve PDEs?

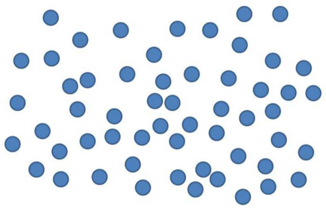
- Grid discretization



- Mesh discretization



- Particle discretization



How to numerically solve PDEs?

- **Finite difference methods**
  - Taylor series expansion
  - Explicit/implicit formulation for time integration
- **Weighted-residual-type methods**
  - Spectral method
  - Finite volume method
  - Finite element method
- **Meshless methods**
  - Smoothed particle hydrodynamics (SPH)
  - Moving-least-square based methods
  - Radial-basis-function based methods

II. Finite Difference Method

Finite Difference Method

- **Derivative Approximation**
  - First derivative
    - Taylor expansion

$$u(x+\Delta x) = u(x) + \Delta x \frac{\partial u(x)}{\partial x} + \frac{(\Delta x)^2}{2}\frac{\partial^2 u(x)}{\partial x^2} + \frac{(\Delta x)^3}{3!}\frac{\partial^3 u(x)}{\partial x^3} + \cdots$$

    - First order approximation

$$\frac{u(x+\Delta x) - u(x)}{\Delta x} = \frac{\partial u(x)}{\partial x} + \frac{\Delta x}{2}\frac{\partial^2 u(x)}{\partial x^2} + \cdots = \frac{\partial u(x)}{\partial x} + \mathrm{O}(\Delta x)$$

Finite Difference Method

- **Derivative Approximation**
  - First derivative
    - Second order approximation

$$u_{i+1} = u_i + \Delta x \left(\frac{\partial u}{\partial x}\right)_i + \frac{\Delta x^2}{2}\left(\frac{\partial^2 u}{\partial x^2}\right)_i + \frac{\Delta x^3}{3!}\left(\frac{\partial^3 u}{\partial x^3}\right)_i + \frac{\Delta x^4}{4!}\left(\frac{\partial^4 u}{\partial x^4}\right)_i + \cdots$$

$$u_{i-1} = u_i - \Delta x \left(\frac{\partial u}{\partial x}\right)_i + \frac{\Delta x^2}{2}\left(\frac{\partial^2 u}{\partial x^2}\right)_i - \frac{\Delta x^3}{3!}\left(\frac{\partial^3 u}{\partial x^3}\right)_i + \frac{\Delta x^4}{4!}\left(\frac{\partial^4 u}{\partial x^4}\right)_i + \cdots$$

$$\Downarrow$$

$$\left(\frac{\partial u}{\partial x}\right)_i = \frac{u_{i+1} - u_{i-1}}{2\Delta x} + \mathrm{O}(\Delta x^2)$$

Finite Difference Method

- **Derivative Approximation**
  - Second derivative
    - Second order approximation

$$u_{i+1} = u_i + \Delta x\left(\frac{\partial u}{\partial x}\right)_i + \frac{\Delta x^2}{2}\left(\frac{\partial^2 u}{\partial x^2}\right)_i + \frac{\Delta x^3}{3!}\left(\frac{\partial^3 u}{\partial x^3}\right)_i + \frac{\Delta x^4}{4!}\left(\frac{\partial^4 u}{\partial x^4}\right)_i + \cdots$$

$$u_{i-1} = u_i - \Delta x\left(\frac{\partial u}{\partial x}\right)_i + \frac{\Delta x^2}{2}\left(\frac{\partial^2 u}{\partial x^2}\right)_i - \frac{\Delta x^3}{3!}\left(\frac{\partial^3 u}{\partial x^3}\right)_i + \frac{\Delta x^4}{4!}\left(\frac{\partial^4 u}{\partial x^4}\right)_i + \cdots$$

$$\Downarrow$$

$$\left(\frac{\partial^2 u}{\partial x^2}\right)_i = \frac{u_{i+1} - 2u_i + u_{i-1}}{\Delta x^2} + \mathrm{O}(\Delta x^2)$$

Finite Difference Method

- **Derivative Approximation**
  - General methods

$$\left(\frac{\partial u}{\partial x}\right)_i = \frac{au_i + bu_{i-1} + cu_{i-2}}{\Delta x}$$

$$u_{i-1} = u_i + (-\Delta x)\left(\frac{\partial u}{\partial x}\right)_i + \frac{(-\Delta x)^2}{2}\left(\frac{\partial^2 u}{\partial x^2}\right)_i + \frac{(-\Delta x)^3}{3!}\left(\frac{\partial^3 u}{\partial x^3}\right)_i + \cdots$$

$$u_{i-2} = u_i + (-2\Delta x)\left(\frac{\partial u}{\partial x}\right)_i + \frac{(-2\Delta x)^2}{2}\left(\frac{\partial^2 u}{\partial x^2}\right)_i + \frac{(-2\Delta x)^3}{3!}\left(\frac{\partial^3 u}{\partial x^3}\right)_i + \cdots$$

$$\Downarrow$$

$$au_i + bu_{i-1} + cu_{i-2} = (a+b+c)u_i - \Delta x(b+2c)\left(\frac{\partial u}{\partial x}\right)_i + \frac{\Delta x^2}{2}(b+4c)\left(\frac{\partial^2 u}{\partial x^2}\right)_i + O(\Delta x^3)$$

Finite Difference Method

- **Derivative Approximation**
  - General methods

$$\left(\frac{\partial u}{\partial x}\right)_i = \frac{au_i + bu_{i-1} + cu_{i-2}}{\Delta x}$$

+

$$au_i + bu_{i-1} + cu_{i-2} = (a+b+c)u_i - \Delta x(b+2c)\left(\frac{\partial u}{\partial x}\right)_i + \frac{\Delta x^2}{2}(b+4c)\left(\frac{\partial^2 u}{\partial x^2}\right)_i + \mathrm{O}(\Delta x^3)$$

$$\begin{aligned} a+b+c&=0 \\ b+2c&=-1 \\ b+4c&=0 \end{aligned}$$

$$\begin{aligned} a+b+c&=0 \\ b+2c&=0 \\ b+4c&=2 \end{aligned}$$

$$\left(\frac{\partial u}{\partial x}\right)_i = \frac{3u_i - 4u_{i-1} + u_{i-2}}{2\Delta x} + \mathrm{O}(\Delta x^2)$$

$$\left(\frac{\partial u}{\partial x}\right)_i = \frac{-3u_i + 4u_{i+1} - u_{i+2}}{2\Delta x} + \mathrm{O}(\Delta x^2)$$

$$\left(\frac{\partial^2 u}{\partial x^2}\right)_i = \frac{u_i - 2u_{i-1} + u_{i-2}}{\Delta x^2} + \Delta x \frac{\partial^3 u}{\partial x^3} + \cdots$$

Similarly

Finite Difference Method

- **Various order finite difference formulas**

(a) Forward Difference, O(Δx)

	u_i	u_{i+1}	u_{i+2}	u_{i+3}	u_{i+4}
$\Delta x \frac{\partial u}{\partial x}$	−1	1			
$\Delta x^2 \frac{\partial^2 u}{\partial x^2}$	1	−2	1		
$\Delta x^3 \frac{\partial^3 u}{\partial x^3}$	−1	3	−3	1	
$\Delta x^4 \frac{\partial^4 u}{\partial x^4}$	1	−4	6	−4	1

(b) Backward Difference, O(Δx)

	u_{i-4}	u_{i-3}	u_{i-2}	u_{i-1}	u_i
$\Delta x \frac{\partial u}{\partial x}$				−1	1
$\Delta x^2 \frac{\partial^2 u}{\partial x^2}$			1	−2	1
$\Delta x^3 \frac{\partial^3 u}{\partial x^3}$		−1	3	−3	1
$\Delta x^4 \frac{\partial^4 u}{\partial x^4}$	1	−4	6	−4	1

(d) Forward Difference, O(Δx^2)

	u_i	u_{i+1}	u_{i+2}	u_{i+3}	u_{i+4}	u_{i+5}
$2\Delta x \frac{\partial u}{\partial x}$	−3	4	−1			
$\Delta x^2 \frac{\partial^2 u}{\partial x^2}$	2	−5	4	−1		
$2\Delta x^3 \frac{\partial^3 u}{\partial x^3}$	−5	18	−24	14	−3	
$\Delta x^4 \frac{\partial^4 u}{\partial x^4}$	3	−14	26	−24	11	−2

(e) Backward Difference, O(Δx^2)

	u_{i-5}	u_{i-4}	u_{i-3}	u_{i-2}	u_{i-1}	u_i
$2\Delta x \frac{\partial u}{\partial x}$				1	−4	3
$\Delta x^2 \frac{\partial^2 u}{\partial x^2}$			−1	4	−5	2
$2\Delta x^3 \frac{\partial^3 u}{\partial x^3}$		3	−14	24	−18	5
$\Delta x^4 \frac{\partial^4 u}{\partial x^4}$	−2	11	−24	26	−14	3

(c) Central Difference, O(Δx^2)

	u_{i-2}	u_{i-1}	u_i	u_{i+1}	u_{i+2}
$2\Delta x \frac{\partial u}{\partial x}$		−1	0	1	
$\Delta x^2 \frac{\partial^2 u}{\partial x^2}$		1	−2	1	
$2\Delta x^3 \frac{\partial^3 u}{\partial x^3}$	−1	2	0	2	1
$\Delta x^4 \frac{\partial^4 u}{\partial x^4}$	1	−4	6	−4	1

(f) Central Difference, O(Δx^4)

	u_{i-3}	u_{i-2}	u_{i-1}	u_i	u_{i+1}	u_{i+2}	u_{i+3}
$12\Delta x \frac{\partial u}{\partial x}$		1	−8	0	8	−1	
$12\Delta x^2 \frac{\partial^2 u}{\partial x^2}$		−1	16	−30	16	−1	
$8\Delta x^3 \frac{\partial^3 u}{\partial x^3}$	1	−8	13	0	−13	8	−1
$6\Delta x^4 \frac{\partial^4 u}{\partial x^4}$	−1	12	−39	56	−39	12	−1

II.a Solving Laplace Equation

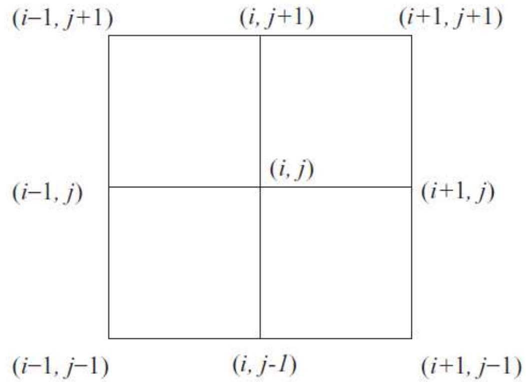
Solving 2D Laplace equation

- **Model equation**

$$\nabla^2 u = \frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} = 0$$

- **Central difference discretization**

$$\Delta u_{ij} = \left(\frac{\delta_x^2}{\Delta x^2} + \frac{\delta_y^2}{\Delta y^2}\right) u_{ij} = \frac{u_{i-1,j} - 2u_{i,j} + u_{i+1,j}}{\Delta x^2} + \frac{u_{i,j-1} - 2u_{i,j} + u_{i,j+1}}{\Delta y^2} + \mathrm{O}(\Delta x^2, \Delta y^2)$$



2D uniform mesh structure

Solving 2D Laplace equation

- **Five-point and nine-point finite differences**

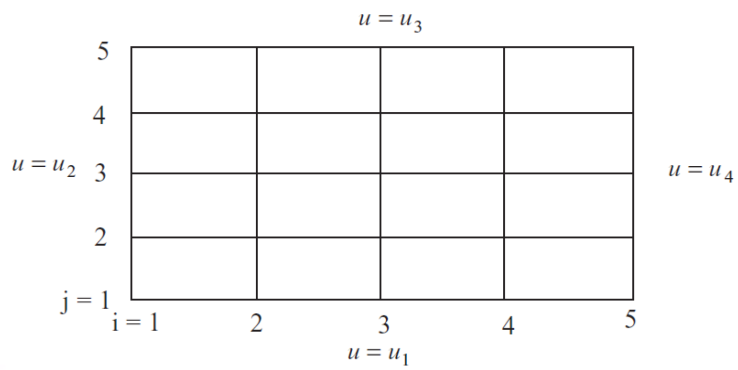
$$\frac{u_{i+1,j} - 2u_{i,j} + u_{i-1,j}}{\Delta x^2} + \frac{u_{i,j+1} - 2u_{i,j} + u_{i,j-1}}{\Delta y^2} = 0$$

$$\frac{-u_{i-2,j} + 16u_{i-1,j} - 30u_{i,j} + 16u_{i+1,j} - u_{i+2,j}}{12\Delta x^2} + \frac{-u_{i,j-2} + 16u_{i,j-1} - 30u_{i,j} + 16u_{i,j+1} - u_{i,j+2}}{12\Delta y^2} = 0$$

  - Consider the five-point scheme

$$u_{i+1,j} + u_{i-1,j} + \beta^2 u_{i,j+1} + \beta^2 u_{i,j-1} - 2(1+\beta^2)u_{i,j} = 0$$

$$\beta = \Delta x / \Delta y$$



Solving 2D Laplace equation

- Writing at all interior nodes and setting

$$\gamma = -2(1+\beta^2)$$

- We obtain for the discretization

$$\begin{bmatrix} \gamma & 1 & 0 & \beta^2 & 0 & 0 & 0 & 0 & 0 \\ 1 & \gamma & 1 & 0 & \beta^2 & 0 & 0 & 0 & 0 \\ 0 & 1 & \gamma & 0 & 0 & \beta^2 & 0 & 0 & 0 \\ \beta^2 & 0 & 0 & \gamma & 1 & 0 & \beta^2 & 0 & 0 \\ 0 & \beta^2 & 0 & 1 & \gamma & 1 & 0 & \beta^2 & 0 \\ 0 & 0 & \beta^2 & 0 & 1 & \gamma & 0 & 0 & \beta^2 \\ 0 & 0 & 0 & \beta^2 & 0 & 0 & \gamma & 1 & 0 \\ 0 & 0 & 0 & 0 & \beta^2 & 0 & 1 & \gamma & 1 \\ 0 & 0 & 0 & 0 & 0 & \beta^2 & 0 & 1 & \gamma \end{bmatrix} \begin{bmatrix} u_{2,2} \\ u_{3,2} \\ u_{4,2} \\ u_{2,3} \\ u_{3,3} \\ u_{4,3} \\ u_{2,4} \\ u_{3,4} \\ u_{4,4} \end{bmatrix} = \begin{bmatrix} -u_{1,2} - \beta^2 u_{2,1} \\ -\beta^2 u_{3,1} \\ -u_{5,2} - \beta^2 u_{4,1} \\ -u_{1,3} \\ 0 \\ -u_{5,3} \\ -u_{1,4} - \beta^2 u_{2,5} \\ -\beta^2 u_{3,5} \\ -u_{5,4} - \beta^2 u_{4,5} \end{bmatrix}$$

Solving Large Sparse Linear Systems

- **Direct methods**
  - Gaussian elimination with factorization (LU, Cholesky factorization etc.)
  - Generally applied, stable, but memory consuming

- **Iterative methods**
  - Jacobi/Gauss-Seidal iteration
  - Successive over-relaxation (SOR)
  - Alternating direction implicit (ADI)
  - Conjugate gradient (CG) and generalized minimal residual (GMRES) algorithms
  - Specific matrix form, convergence stability based on matrix structure, but much less memory usage

Solving Large Sparse Linear Systems

- **Iterative methods**
  - Jacobi iteration method

$$u_{i,j}^{k+1} = \frac{1}{2(1+\beta^2)}[u_{i+1,j}^{k} + u_{i-1,j}^{k} + \beta^2(u_{i,j+1}^{k} + u_{i,j-1}^{k})]$$

  - Point Gauss-Seidel Iteration Method

$$u_{i,j}^{k+1} = \frac{1}{2(1+\beta^2)}[u_{i+1,j}^{k} + u_{i-1,j}^{k+1} + \beta^2(u_{i,j+1}^{k} + u_{i,j-1}^{k+1})]$$

  - Line Gauss-Seidel Iteration Method

$$u_{i-1,j}^{k+1} - 2(1+\beta^2)u_{i,j}^{k+1} + u_{i+1,j}^{k+1} = -\beta^2(u_{i,j+1}^{k} + u_{i,j-1}^{k+1})$$

Solving Large Sparse Linear Systems

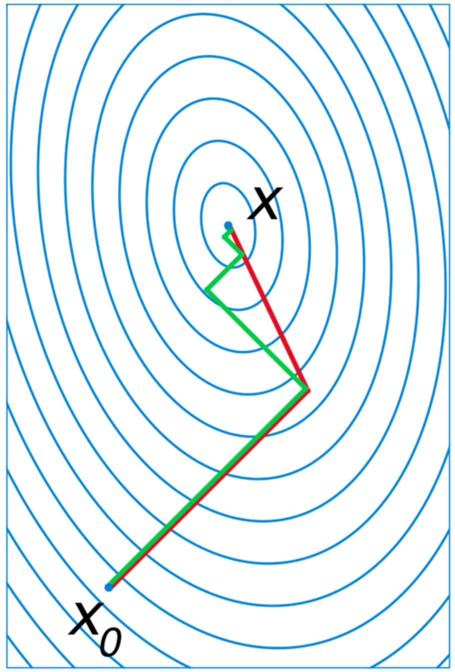
- **Iterative methods**
  - Conjugate gradient
    - Iterative formulation

$$\mathbf{p}_0 = \mathbf{b} - \mathbf{A}\mathbf{x}_0$$

$$\mathbf{p}_k = \mathbf{r}_k - \sum_{i<k} \frac{\mathbf{p}_i^\mathsf{T} \mathbf{A} \mathbf{r}_k}{\mathbf{p}_i^\mathsf{T} \mathbf{A} \mathbf{p}_i} \mathbf{p}_i$$

$$\mathbf{x}_{k+1} = \mathbf{x}_k + \alpha_k \mathbf{p}_k$$

$$\alpha_k = \frac{\mathbf{p}_k^\mathsf{T} \mathbf{b}}{\mathbf{p}_k^\mathsf{T} \mathbf{A} \mathbf{p}_k} = \frac{\mathbf{p}_k^\mathsf{T} (\mathbf{r}_{k-1} + \mathbf{A}\mathbf{x}_{k-1})}{\mathbf{p}_k^\mathsf{T} \mathbf{A} \mathbf{p}_k} = \frac{\mathbf{p}_k^\mathsf{T} \mathbf{r}_{k-1}}{\mathbf{p}_k^\mathsf{T} \mathbf{A} \mathbf{p}_k}$$



II.b Solving Diffusion Equation

1D Diffusion Equation

- **Solving 1D diffusion equation**
  - Model equation

$$\frac{\partial u}{\partial t} - \alpha \frac{\partial^2 u}{\partial x^2} = 0$$

  - Forward-Time/Central-Space (FTCS) Method
    - Explicit scheme

$$\frac{u_i^{n+1} - u_i^n}{\Delta t} = \frac{\alpha\left(u_{i+1}^n - 2u_i^n + u_{i-1}^n\right)}{\Delta x^2} + O(\Delta t, \Delta x^2)$$

or

$$u_i^{n+1} = u_i^n + d\left(u_{i+1}^n - 2u_i^n + u_{i-1}^n\right) \qquad d = \frac{\alpha \Delta t}{\Delta x^2}$$

1D Diffusion Equation

- **Solving 1D diffusion equation**
  - von Neumann stability analysis

$$\frac{u_i^{n+1} - u_i^n}{\Delta t} = \frac{\alpha\left(u_{i+1}^n - 2u_i^n + u_{i-1}^n\right)}{\Delta x^2} + O(\Delta t, \Delta x^2) \qquad + \qquad u_i^n = \bar{u}_i^n + \varepsilon_i^n$$

$$\Downarrow$$

$$\frac{\bar{u}_i^{n+1} - \bar{u}_i^n}{\Delta t} + \frac{\varepsilon_i^{n+1} - \varepsilon_i^n}{\Delta t} = \frac{\alpha}{(\Delta x)^2}\left(\bar{u}_{i+1}^n - 2\bar{u}_i^n + \bar{u}_{i-1}^n\right) + \frac{\alpha}{(\Delta x)^2}\left(\varepsilon_{i+1}^n - 2\varepsilon_i^n + \varepsilon_{i-1}^n\right)$$

$$\Downarrow$$

$$\frac{\varepsilon_i^{n+1} - \varepsilon_i^n}{\Delta t} = \frac{\alpha}{(\Delta x)^2}\left(\varepsilon_{i+1}^n - 2\varepsilon_i^n + \varepsilon_{i-1}^n\right)$$

1D Diffusion Equation

- **Solving 1D diffusion equation**
  - von Neumann stability analysis
    - Writing for the entire domain leads to

$$\mathbf{U}^n = \bar{\mathbf{U}}^n + \boldsymbol{\varepsilon}^n$$

$$\boldsymbol{\varepsilon}^n = \begin{bmatrix} \cdot \\ \varepsilon_{i-1}^n \\ \varepsilon_i^n \\ \varepsilon_{i+1}^n \\ \cdot \end{bmatrix}$$

$$\downarrow$$

$$\bar{\mathbf{U}}^{n+1} + \boldsymbol{\varepsilon}^{n+1} = \mathbf{C}(\bar{\mathbf{U}}^n + \boldsymbol{\varepsilon}^n)$$

$$\boldsymbol{\varepsilon}^{n+1} = \mathbf{C}\boldsymbol{\varepsilon}^n$$

$$C = 1 + d(E - 2 + E^{-1}) = \begin{bmatrix} \cdot & \cdot & \cdot & \cdot & \cdot \\ d & (1-2d) & d & 0 & 0 \\ \cdot & d & (1-2d) & d & 0 \\ \cdot & 0 & d & (1-2d) & d \\ \cdot & \cdot & \cdot & \cdot & \cdot \end{bmatrix}$$

1D Diffusion Equation

- **Solving 1D diffusion equation**
  - von Neumann stability analysis
    - If the boundary conditions are considered as periodic
    - Fourier series expansion in space

$$k_j = jk_{\min} = j\pi/L = j\pi/(N\Delta x), \quad j = 0, 1, \ldots N$$

$$\varepsilon_i^n = \sum_{j=-N}^{N} \bar{\varepsilon}_j^n e^{Ik_j(i\Delta x)} = \sum_{j=-N}^{N} \bar{\varepsilon}_j^n e^{Iji\pi/N} \qquad I = \sqrt{-1}$$

$$\phi = k_j \Delta x = j\pi/N \qquad \varepsilon_i^n = \sum_{j=-N}^{N} \bar{\varepsilon}_j^n e^{Ii\phi}$$

$$\frac{\bar{\varepsilon}^{n+1} - \bar{\varepsilon}^n}{\Delta t} e^{Ii\phi} = \frac{\alpha}{\Delta x^2}\left(\bar{\varepsilon}^n e^{I(i+1)\phi} - 2\bar{\varepsilon}^n e^{Ii\phi} + \bar{\varepsilon}^n e^{I(i-1)\phi}\right)$$

or

$$\bar{\varepsilon}^{n+1} - \bar{\varepsilon}^n - d\bar{\varepsilon}^n\left(e^{I\phi} - 2 + e^{-I\phi}\right) = 0$$

1D Diffusion Equation

- **Solving 1D diffusion equation**
  - von Neumann stability analysis
    - Stability condition

$$|g| = \left|\frac{\bar{\varepsilon}^{n+1}}{\bar{\varepsilon}^{n}}\right| \le 1 \quad \text{for all } \phi$$

$$\Downarrow$$

$$g = 1 + d(e^{I\phi} - 2 + e^{-I\phi})$$

or

$$g = 1 - 2d(1 - \cos\phi)$$

$$\Rightarrow$$

$$g \le 1$$

or

$$1 - 2d(1 - \cos\phi) \ge -1$$

$$\Downarrow$$

$$0 \le d \le 1/2$$

1D Diffusion Equation

- **Solving 1D diffusion equation**
  - Implicit scheme
    - Laasonen method

$$\frac{u_i^{n+1} - u_i^n}{\Delta t} = \frac{\alpha\left(u_{i+1}^{n+1} - 2u_i^{n+1} + u_{i-1}^{n+1}\right)}{\Delta x^2}, \quad O(\Delta t, \Delta x^2)$$

      - Unconditionally stable

    - Crank-Nicolson method

$$\frac{u_i^{n+1} - u_i^n}{\Delta t} = \frac{\alpha}{2}\left[\frac{u_{i+1}^{n+1} - 2u_i^{n+1} + u_{i-1}^{n+1}}{\Delta x^2} + \frac{u_{i+1}^{n} - 2u_i^{n} + u_{i-1}^{n}}{\Delta x^2}\right], \quad O(\Delta t^2, \Delta x^2)$$

      - Unconditionally stable

1D Diffusion Equation

- **Solving 1D diffusion equation**
  - Implicit scheme
    - β-method

$$\frac{u_i^{n+1} - u_i^n}{\Delta t} = \alpha\left[\frac{\beta\left(u_{i+1}^{n+1} - 2u_i^{n+1} + u_{i-1}^{n+1}\right)}{(\Delta x)^2} + \frac{(1-\beta)\left(u_{i+1}^n - 2u_i^n + u_{i-1}^n\right)}{(\Delta x)^2}\right]$$

      - For $\frac{1}{2} \le \beta \le 1$, unconditionally stable

2D Diffusion Equation

- **Solving 2D diffusion equation**
  - Model equation

$$\frac{\partial u}{\partial t} - \alpha\left(\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2}\right) = 0$$

  - Explicit scheme

$$\frac{u_{i,j}^{n+1} - u_{i,j}^{n}}{\Delta t} = \alpha\left(\frac{u_{i+1,j}^{n} - 2u_{i,j}^{n} + u_{i-1,j}^{n}}{\Delta x^2} + \frac{u_{i,j+1}^{n} - 2u_{i,j}^{n} + u_{i,j-1}^{n}}{\Delta y^2}\right), \quad O(\Delta t, \Delta x^2, \Delta y^2)$$

    - Stability condition

$$d_x + d_y \leq \frac{1}{2} \qquad\qquad d_x = \frac{\alpha \Delta t}{\Delta x^2}, \qquad d_y = \frac{\alpha \Delta t}{\Delta y^2}$$

2D Diffusion Equation

- **Solving 2D diffusion equation**
  - Implicit scheme
    - Alternating direction implicit (ADI) scheme
      - Unconditionally stable

$$\frac{u_{i,j}^{n+1} - u_{i,j}^{n}}{\Delta t} = \alpha\left(\frac{u_{i+1,j}^{n+1} - 2u_{i,j}^{n+1} + u_{i-1,j}^{n+1}}{\Delta x^2} + \frac{u_{i,j+1}^{n+1} - 2u_{i,j}^{n+1} + u_{i,j-1}^{n+1}}{\Delta y^2}\right)$$

$$\Downarrow$$

$$\frac{u_{i,j}^{n+\frac{1}{2}} - u_{i,j}^{n}}{\Delta t/2} = \alpha\left(\frac{u_{i+1,j}^{n+\frac{1}{2}} - 2u_{i,j}^{n+\frac{1}{2}} + u_{i-1,j}^{n+\frac{1}{2}}}{\Delta x^2} + \frac{u_{i,j+1}^{n} - 2u_{i,j}^{n} + u_{i,j-1}^{n}}{\Delta y^2}\right)$$

and

$$\frac{u_{i,j}^{n+1} - u_{i,j}^{n+\frac{1}{2}}}{\Delta t/2} = \alpha\left(\frac{u_{i+1,j}^{n+\frac{1}{2}} - 2u_{i,j}^{n+\frac{1}{2}} + u_{i-1,j}^{n+\frac{1}{2}}}{\Delta x^2} + \frac{u_{i,j+1}^{n+1} - 2u_{i,j}^{n+1} + u_{i,j-1}^{n+1}}{\Delta y^2}\right)$$

2D Diffusion Equation

- **Solving 2D diffusion equation**
  - Implicit scheme
    - Alternating direction implicit (ADI) scheme
      - These two equations can be written in a tridiagonal form

$$\underbrace{-d_1 u_{i+1,j}^{n+\frac{1}{2}} + (1+2d_1) u_{i,j}^{n+\frac{1}{2}} - d_1 u_{i-1,j}^{n+\frac{1}{2}}}_{\text{implicit in } x\text{-direction}} = \underbrace{d_2 u_{i,j+1}^{n} + (1-2d_2) u_{i,j}^{n} + d_2 u_{i,j-1}^{n}}_{\text{explicit in } y\text{-direction}}$$

$$\underbrace{-d_2 u_{i,j+1}^{n+1} + (1+2d_2) u_{i,j}^{n+1} - d_2 u_{i,j-1}^{n+1}}_{\text{unknown}} = \underbrace{d_1 u_{i+1,j}^{n+\frac{1}{2}} + (1-2d_1) u_{i,j}^{n+\frac{1}{2}} + d_1 u_{i-1,j}^{n+\frac{1}{2}}}_{\text{known}}$$

$$d_1 = \frac{1}{2} d_x = \frac{1}{2} \frac{\alpha \Delta t}{\Delta x^2}$$

$$d_2 = \frac{1}{2} d_y = \frac{1}{2} \frac{\alpha \Delta t}{\Delta y^2}$$

2D Diffusion Equation

- **Solving 2D diffusion equation**
  - Solving tridiagonal matrix system
    - Thomas algorithm

$$
\begin{bmatrix}
b_1 & c_1 & 0 & \cdot & \cdot & \cdot & \cdot \\
a_2 & b_2 & c_2 & 0 & \cdot & \cdot & \cdot \\
0 & a_3 & b_3 & c_3 & 0 & \cdot & \cdot \\
\cdot & \cdot & * & * & * & \cdot & \cdot \\
\cdot & \cdot & \cdot & * & * & * & \cdot \\
\cdot & \cdot & \cdot & \cdot & * & * & c_{NI-1} \\
0 & \cdot & \cdot & \cdot & \cdot & a_{NI} & b_{NI}
\end{bmatrix}
\begin{bmatrix}
T_1^{n+1} \\ T_2^{n+1} \\ T_3^{n+1} \\ * \\ * \\ * \\ T_{NI}^{n+1}
\end{bmatrix}
=
\begin{bmatrix}
g_1 \\ g_2 \\ g_3 \\ * \\ * \\ * \\ g_{NI}
\end{bmatrix}
$$

$$b_i = b_i - \frac{a_i}{b_{i-1}} c_{i-1} \quad i = 2, 3, \ldots NI$$

$$g_i = g_i - \frac{a_i}{b_{i-1}} g_{i-1} \quad i = 2, 3, \ldots NI$$

$$T_{NI} = \frac{g_{NI}}{b_{NI}}$$

$$T_j = \frac{g_j - c_j T_{j+1}}{b_j} \quad j = NI - 1, \quad NI - 2, \ldots, 1$$

II.b Solving Advection Equation

1D Advection Equation

- **Solving 1D advection equation**
  - Model equation

$$\frac{\partial u}{\partial t} + a\frac{\partial u}{\partial x} = 0, \quad a > 0$$

  - Forward time and forward space (FTFS) approximations

$$\frac{u_i^{n+1} - u_i^n}{\Delta t} = -a\frac{u_{i+1}^n - u_i^n}{\Delta x}$$

1D Advection Equation

- **Solving 1D advection equation**
  - Forward time and forward space (FTFS) approximations
    - von Neumann stability analysis
      - Amplification factor

$$g = 1 - C(e^{I\phi} - 1) = 1 - C(\cos\phi - 1) - IC\sin\phi = 1 + 2C\sin^2\frac{\phi}{2} - IC\sin\phi$$

$$C = \frac{a\Delta t}{\Delta x} \longleftarrow \text{CFL number}$$

$$|g|^2 = g\,g^* = \left(1 + 2C\sin^2\frac{\phi}{2}\right)^2 + C^2\sin^2\phi = 1 + 4C(1+C)\sin^2\frac{\phi}{2} \geq 1$$

      - Unconditionally unstable

1D Advection Equation

- **Solving 1D advection equation**
  - Forward time and backward space (FTBS) approximations
    - First order upwind scheme

$$\frac{u_i^{n+1} - u_i^n}{\Delta t} = -a\frac{u_i^n - u_{i-1}^n}{\Delta x}, \quad O(\Delta t, \Delta x)$$

    - von Neumann stability analysis
      - Amplification factor

$$g = 1 - C(1 - e^{-I\phi}) = 1 - C(1 - \cos\phi) - IC\sin\phi$$
$$= 1 - 2C\sin^2\frac{\phi}{2} - IC\sin\phi$$

1D Advection Equation

- **Solving 1D advection equation**
  - Forward time and backward space (FTBS) approximations
    - von Neumann stability analysis
      - Amplification factor

or

$$g = \xi + I\eta, \qquad |g| = \left[1 - 4C(1 - C)\sin^2\frac{\phi}{2}\right]^{1/2}$$

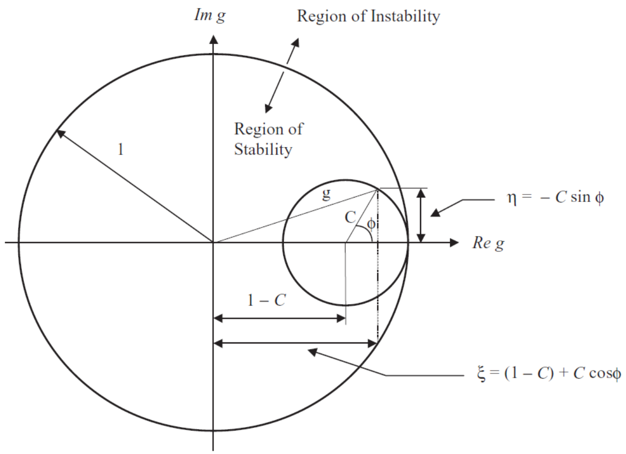
with

$$\xi = 1 - 2C\sin^2\frac{\phi}{2} = (1 - C) + C\cos\phi$$

$$\eta = -C\sin\phi$$

1D Advection Equation

- **Solving 1D advection equation**
  - Forward time and backward space (FTBS) approximations
    - von Neumann stability analysis



$$g = \xi + I\eta, \qquad |g| = \left[1 - 4C(1-C)\sin^2\frac{\phi}{2}\right]^{1/2}$$

$$\xi = 1 - 2C\sin^2\frac{\phi}{2} = (1-C) + C\cos\phi$$

$$\eta = -C\sin\phi$$

Conditionally stable:
0<C<1 (CFL condition)

1D Advection Equation

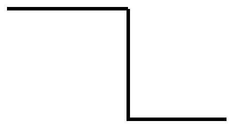
- **Solving 1D advection equation**
  - Dissipation error
    - Defined by the amplitude of g
  - Dispersion error
    - Defined by the angle (phase) of g
      - Phase by the numerical scheme

$$\Phi = \tan^{-1}\frac{\mathrm{Im}(g)}{\mathrm{Re}(g)} = \tan^{-1}\frac{\eta}{\xi} = \tan^{-1}\frac{-C\sin\phi}{1-C+C\cos\phi}$$

      - Ideal phase $\tilde{\Phi} = ka\Delta t = C\phi$

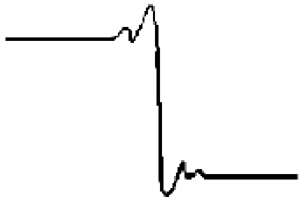
1D Advection Equation

- **Solving 1D advection equation**
  - Dissipation and dispersion error



Exact solution

Dissipation error



Dispersion error

1D Advection Equation

- **Solving 1D advection equation**
  - Choose computational schemes
    - Minimize both dissipation and dispersion errors
  - Lax method $$u_i^{n+1} = \frac{1}{2}\left(u_{i+1}^n + u_{i-1}^n\right) - \frac{C}{2}\left(u_{i+1}^n - u_{i-1}^n\right)$$
    - Stability condition: C≤1
  - Midpoint leapfrog method

$$\frac{u_i^{n+1} - u_i^{n-1}}{2\Delta t} = -\frac{a\left(u_{i+1}^n - u_{i-1}^n\right)}{2\Delta x}, \quad \mathrm{O}(\Delta t^2, \Delta x^2)$$

    - Stability condition: C≤1; two independent solutions, coupled only spatially.

1D Advection Equation

- **Solving 1D advection equation**
  - Lax-Wendroff method

$$u_i^{n+1} = u_i^n + \frac{\partial u}{\partial t}\Delta t + \frac{1}{2!}\frac{\partial^2 u}{\partial t^2}\Delta t^2 + \mathrm{O}(\Delta t^3)$$

$$\frac{\partial u}{\partial t} + a\frac{\partial u}{\partial x} = 0, \quad a > 0 \quad \Rightarrow \quad \frac{\partial^2 u}{\partial t^2} = -a\frac{\partial}{\partial x}\left(\frac{\partial u}{\partial t}\right) = a^2\frac{\partial^2 u}{\partial x^2}$$

$$u_i^{n+1} = u_i^n + \Delta t\left(-a\frac{\partial u}{\partial x}\right) + \frac{\Delta t^2}{2}\left(a^2\frac{\partial^2 u}{\partial x^2}\right)$$

$$u_i^{n+1} = u_i^n - a\Delta t\left(\frac{u_{i+1}^n - u_{i-1}^n}{2\Delta x}\right) + \frac{1}{2}(a\,\Delta t)^2\left(\frac{u_{i+1}^n - 2u_i^n + u_{i-1}^n}{(\Delta x)^2}\right), \quad \mathrm{O}(\Delta t^2, \Delta x^2)$$

Stability condition: C⩽1

1D Advection Equation

- **Solving 1D advection equation**
  - Implicit scheme
    - Unconditionally stable
    - Euler's FTCS method

$$\frac{u_i^{n+1} - u_i^n}{\Delta t} = \frac{-a}{2\Delta x}\left(u_{i+1}^{n+1} - u_{i-1}^{n+1}\right), \quad O(\Delta t, \Delta x^2)$$

    - Crank-Nicolson method

$$\frac{u_i^{n+1} - u_i^n}{\Delta t} = -\frac{a}{2}\left[\frac{u_{i+1}^{n+1} - u_{i-1}^{n+1}}{2\Delta x} + \frac{u_{i+1}^{n} - u_{i-1}^{n}}{2\Delta x}\right], \quad O(\Delta t^2, \Delta x^2)$$

1D Advection Equation

- **Solving 1D advection equation**
  - Predictor-corrector methods
    - Lax-Wendroff multistep scheme

Step 1

$$u_{i+\frac{1}{2}}^{n+\frac{1}{2}} = \frac{1}{2}\left(u_{i+1}^{n} + u_{i}^{n}\right) - \frac{C}{2}\left(u_{i+1}^{n} - u_{i}^{n}\right), \quad \mathrm{O}(\Delta t^2, \Delta x^2)$$

Step 2

$$u_{i}^{n+1} = u_{i}^{n} - C\left(u_{i+\frac{1}{2}}^{n+\frac{1}{2}} - u_{i-\frac{1}{2}}^{n+\frac{1}{2}}\right), \quad \mathrm{O}(\Delta t^2, \Delta x^2)$$

- Stability condition: $C \leq 1$

1D Advection Equation

- **Solving 1D advection equation**
  - Predictor-corrector methods
    - MacCormack multistep scheme
      - Consider an intermediate step

$$u_i^{n+\frac{1}{2}} = \frac{1}{2}(u_i^n + u_i^*)$$

Step 1

$$\frac{u_i^* - u_i^n}{\Delta t} = -a\frac{(u_{i+1}^n - u_i^n)}{\Delta x}$$

Predictor

$$u_i^* = u_i^n - C(u_{i+1}^n - u_i^n)$$

Step 2

$$\frac{u_i^{n+1} - u_i^{n+\frac{1}{2}}}{\Delta t/2} = -a\frac{(u_i^* - u_{i-1}^*)}{\Delta x}$$

Corrector

$$u_i^{n+1} = \frac{1}{2}[(u_i^n + u_i^*) - C(u_i^* - u_{i-1}^*)], \quad O(\Delta t^2, \Delta x^2)$$

Stability condition: C⩽1

III Spectral Methods

Spectral Method

- **Method of weighted residuals**
  - Basic principle
    - Consider partial differential equation $P[u] = 0$
      - On domain D, with boundary condition $B(u) = 0$
    - An ansatz for the approximate solution

$$u_N(x,t) = u_B(x,t) + \sum_{k=0}^{N} a_k(t) \cdot \phi_k(x)$$

      - $\phi_k(x)$ are called trial functions, usually fulfill homogeneous boundary conditions on ∂B
      - $a_k(t)$ are the corresponding time-dependent coefficients

Spectral Method

- **Method of weighted residuals**
  - Basic principle
    - Advantage of the ansatz
      - Temporal and spatial derivatives are decoupled
    - Spatial derivatives

$$\frac{\partial^p u_N}{\partial x^p} = \sum_{k=0}^{N} a_k(t) \cdot \frac{\mathrm{d}^p}{\mathrm{d}x^p} \phi_k(x)$$

    - Residual is defined as

$$R(x,t) := P(u_N(x,t))$$

Spectral Method

- **Method of weighted residuals**
  - Basic principle
    - How to determine the N + 1 unknown coefficients?
      - Residual R is required to be orthogonal to all test functions $w_j(x)$

$$\int_{\mathcal{D}} w_j(x) \cdot R(x,t) \mathrm{d}x = 0 \;, \quad j = 0, \ldots, N \;,$$

    - Choice of test functions
      - Galerkin method $w_j = \phi_j \;, \quad j = 0, \ldots, N$
      - Collocation method: the residual R is required to vanish on sample points

$$w_j = \delta(x - x_j) \;, \quad j = 0, \ldots, N$$

Spectral Method

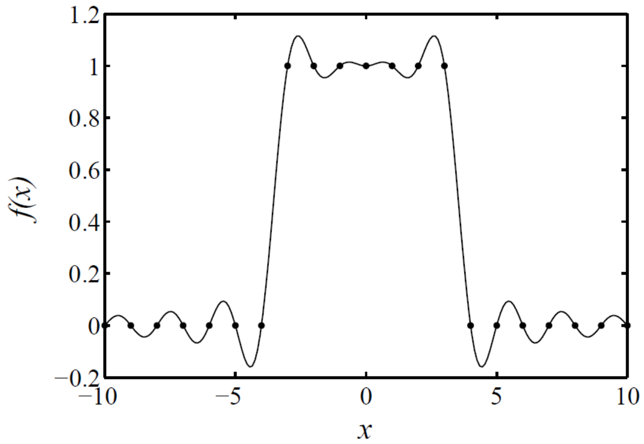
- **Method of weighted residuals**
  - Basic principle
    - Choice of trial functions
      - Fourier series

$$u_N(x) = \sum_{|k|\leq K} c_k e^{\mathrm{i}k\alpha x} = \sum_{|k|\leq K} c_k \Phi_k \ , \quad \text{with } c_k \in \mathbb{C}$$

      - Spectral convergence for smooth regions
      - Important properties of Fourier series
        - Orthogonality $(\Phi_k, \Phi_l) = \frac{1}{L}\int_0^L \Phi_k(x)\Phi_l^*(x)\mathrm{d}x = \frac{1}{L}\int_0^L \Phi_k(x)\Phi_{-l}(x)\mathrm{d}x = \delta_{kl}$
        - Differentiation $\Phi_k'(x) = \mathrm{i}k\alpha\Phi_k(x)$

Spectral Method

- **Method of weighted residuals**
  - Basic principle
    - Choice of trial functions
      - Gibbs phenomena for discontinuous functions



Spectral Method

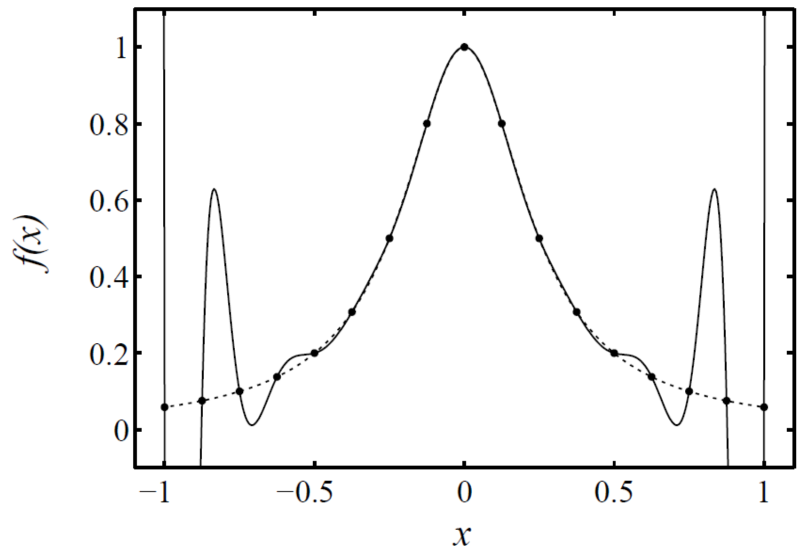
- **Method of weighted residuals**
  - Basic principle
    - Choice of trial functions
      - Chebyshev polynomials
        - Fourier series have problem with non-periodic functions
        - Adopt Chebyshev polynomials, defined on a domain |x| ≤ 1

$$T_k(x) = \cos(k \arccos x) \ , \quad k = 0, 1, 2, \ldots \ .$$

$$\begin{aligned} T_0(x) &= 1 \\ T_1(x) &= x \\ T_2(x) &= 2x^2 - 1 \\ T_3(x) &= 4x^3 - 3x \end{aligned}$$

Spectral Method

- **Method of weighted residuals**
  - Basic principle
    - Choice of trial functions
      - General problem for high-order polynomials interpolation: Runge phenomena
        - Interpolate over equal-distance samples

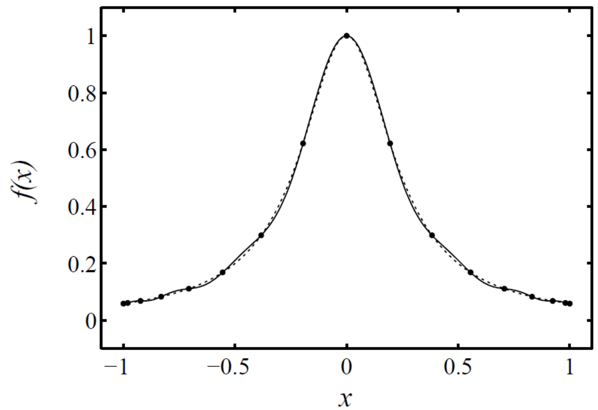


Spectral Method

- **Method of weighted residuals**
  - Basic principle
    - Choice of trial functions
      - non-equidistant distribution of points for exponential convergence
      - A common distribution of points: Gauss-Lobatto points

$$x_j = \cos \frac{\pi j}{N}\ , \quad j = 0, \ldots, N$$

      - Can employ FFT for computation



Spectral Method

- **Example**
  - Linear stationary case
    - Consider a linear problem of the form

$$P(u) \equiv \mathrm{L}u - r = 0$$

    - The residual is then given by

$$R(x) = \mathrm{L}u_N - r$$

    - Using the weighted residual formulation

$$\sum_{k=0}^{N} a_k \int_{\mathcal{D}} w_j \cdot \mathrm{L}\phi_k(x)\mathrm{d}x = \int_{\mathcal{D}} w_j(r - \mathrm{L}u_B)\mathrm{d}x \ , \quad j = 0, \ldots, N$$

    - In matrix formulation **Aa=s** $\quad A_{jk} = \int_{\mathcal{D}} w_j \cdot \mathrm{L}\phi_k(x)\mathrm{d}x \qquad s_j = \int_{\mathcal{D}} w_j \cdot (r - \mathrm{L}u_B)\mathrm{d}x$

Spectral Method

- **Example**
  - Linear stationary case
    - Galerkin method

$$A_{jk} = \int \phi_j \mathrm{L} \phi_k \mathrm{d}x \ , \quad s_j = \int \phi_j (r - \mathrm{L} u_B) \mathrm{d}x$$

    - Collocation method

$$A_{jk} = \mathrm{L}\phi_k(x_j) \ , \quad s_j = r(x_j) - \mathrm{L}u_B(x_j)$$

III.a Finite Element Methods

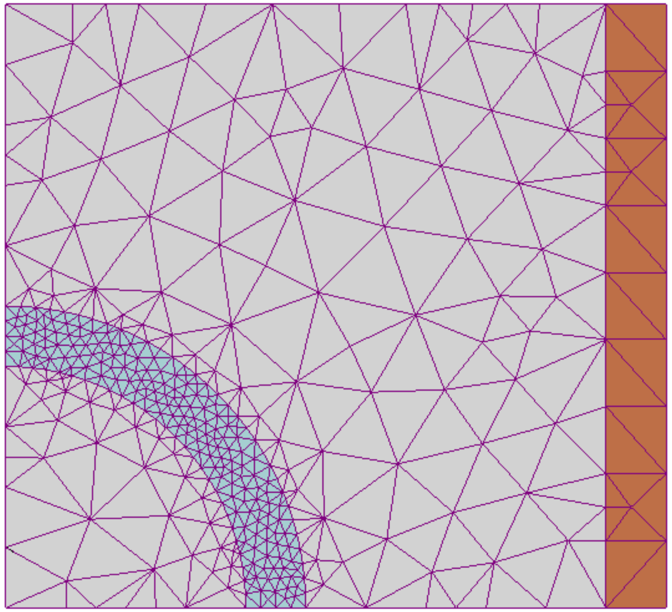
Finite Element Method

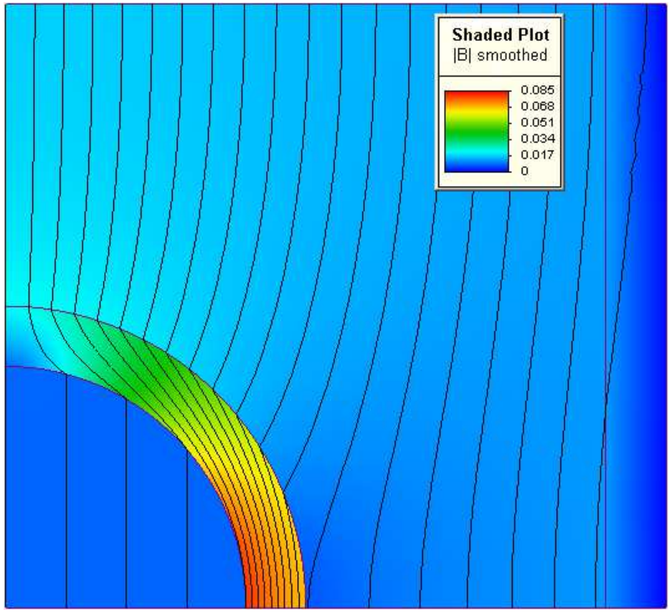
- **A numerical method to solve PDE**
  - Based on domain decomposition
    - Usually triangle/tetrahedron meshes
  - Function approximation over element domains

- **Advantage of domain subdivision**
  - Accurate representation of complex geometry
  - Inclusion of dissimilar material properties
  - Easy representation of the total solution
  - Capture of local effects

Finite Element Method

- **FEM Mesh and related solution**





Finite Element Method

- **1D Poisson problem**
  - Strong formulation

$$\begin{cases} u''(x) = f(x) \text{ in } (0,1) \\ u(0) = u(1) = 0 \end{cases}$$

  - Weak formulation
    - If ***u*** solves the problem, then for any smooth function *v*

$$\begin{aligned} \int_0^1 f(x)v(x)\,dx &= \int_0^1 u''(x)v(x)\,dx \\ &= u'(x)v(x)|_0^1 - \int_0^1 u'(x)v'(x)\,dx \\ &= -\int_0^1 u'(x)v'(x)\,dx \end{aligned}$$

assumption that $v(0) = v(1) = 0$

III.b Finite Volume Methods

Finite Volume Method

- **What is a finite volume**
  - The small volume surrounding each node point on a mesh

- **Finite volume formulation**
  - Volume integrals in a PDE containing a divergence term are converted to surface integrals
  - Evaluated as fluxes at the surfaces of each finite volume
  - Conservative
    - Flux entering a given volume is identical to that leaving the adjacent volume

Finite Volume Method

- **Conservative formulation**
  - The general conservation law problem

$$\frac{\partial \mathbf{u}}{\partial t} + \nabla \cdot \mathbf{f}\left(\mathbf{u}\right) = \mathbf{0}$$

  - Take the volume integral over the cell

$$\int_{v_i} \frac{\partial \mathbf{u}}{\partial t}\, dv + \int_{v_i} \nabla \cdot \mathbf{f}\left(\mathbf{u}\right)\, dv = \mathbf{0}$$

  - Apply divergence theorem

$$v_i \frac{d\bar{\mathbf{u}}_i}{dt} + \oint_{S_i} \mathbf{f}\left(\mathbf{u}\right) \cdot \mathbf{n}\, dS = \mathbf{0} \quad \Rightarrow \quad \frac{d\bar{\mathbf{u}}_i}{dt} + \frac{1}{v_i} \oint_{S_i} \mathbf{f}\left(\mathbf{u}\right) \cdot \mathbf{n}\, dS = \mathbf{0}$$

IV Meshless(Particle) Methods

Meshless Numerical Methods

- **Problem with mesh-based methods**
  - Difficulty in meshing and re-meshing
  - Difficulties when dealing with certain class of problems
    - Handling large deformation that leads to an extremely skewed mesh
    - Simulating the breakage of structures or components with large numbers of fragments
    - Solving dynamic contacts with moving boundaries
    - Solving multi-physics problems

Meshless Numerical Methods

- **Meshless methods**
  - Construct numerical solvers without mesh (only point samples)
  - Point sampling is much easier than meshing
- **Typical methods**
  - Smoothed particle hydrodynamics (SPH)
  - Moving least square (MLS)
  - Radial basis functions (RBF)

Smoothed-Particle Hydrodynamics

- **Function approximation with SPH**
  - Problem setting
    - Reconstructing an (unknown) function f from a set of irregular samples $f_i = f(x_i)$
    - Using the Dirac-delta function, we can rewrite f(x) as a convolution

$$f(\mathbf{x}) = \int_{\mathbf{x}'} f(\mathbf{x}')\delta(\|\mathbf{x}-\mathbf{x}'\|)\,\mathrm{d}V$$

    - Replace delta function with a kernel function w_h

$$\tilde{f}(\mathbf{x}) = \int_{\mathbf{x}'} f(\mathbf{x}')\omega_h(\|\mathbf{x}-\mathbf{x}'\|)\,\mathrm{d}V \qquad \int \omega_h = 1$$

Smoothed-Particle Hydrodynamics

- **Function approximation with SPH**
  - Discretize the integral into a sum over all sample points to obtain the SPH approximation

$$\tilde{f}(\mathbf{x}) = \int_{\mathbf{x}'} f(\mathbf{x}')\omega_h(\|\mathbf{x}-\mathbf{x}'\|)\,\mathrm{d}V \quad \Rightarrow \quad \langle f \rangle(\mathbf{x}) = \sum_i f_i \omega_h(\|\mathbf{x}_i - \mathbf{x}\|)V_i$$

  - How to compute volume V_i for each sample?
    - Associate with mass m_i

$$V_i = \frac{m_i}{\rho_i}$$

Smoothed-Particle Hydrodynamics

- **Function approximation with SPH**
  - How to compute density estimation?

$$\rho_i = \langle \rho \rangle (\mathbf{x}_i) = \sum_j \omega_h(\|\mathbf{x}_i - \mathbf{x}_j\|)\, \rho_j V_j \qquad + \qquad V_i = \frac{m_i}{\rho_i}$$

$$\Downarrow$$

$$\begin{aligned}
\rho_i = \langle \rho \rangle (\mathbf{x}_i) &= \sum_j \omega_h(\|\mathbf{x}_i - \mathbf{x}_j\|)\, \rho_j V_j \\
&= \sum_j \omega_h(\|\mathbf{x}_i - \mathbf{x}_j\|)\, \rho_j \frac{m_j}{\rho_j} \\
&= \sum_j \omega_h(\|\mathbf{x}_i - \mathbf{x}_j\|)\, m_j
\end{aligned}$$

Smoothed-Particle Hydrodynamics

- **Kernel functions**
  - Admissible kernel functions: they must be normalized

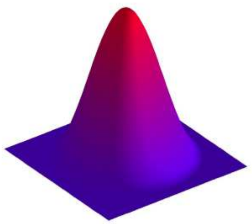
$$\int_{\mathbf{X}} \omega_h(||\mathbf{x}||)\mathrm{d}V = 1$$

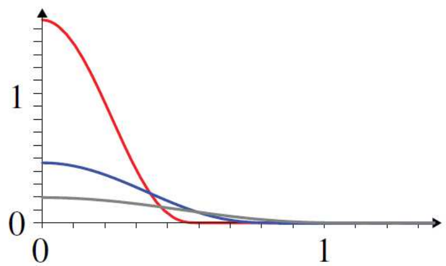
  - Smoothing parameter h
    - Allowing control over how far the influence of each sample point reaches (local support)
    - Too large values of h produce unnecessarily smooth reconstructions
    - Kernel function converges to a Dirac-delta function as h goes to zero

Smoothed-Particle Hydrodynamics

- **Kernel functions**
  - A good polynomial kernel function

$$\omega_h(d) = \begin{cases} \frac{315}{64\pi h^3}\left(1 - \frac{d^2}{h^2}\right)^3 & d < h, \\ 0 & \text{otherwise} \end{cases}$$





Smoothed-Particle Hydrodynamics

- **Approximation of differential operators**
  - Apply SPH approximations to the solution of partial differential equations
    - Not only a reconstruction of the continuous function *f* , but also the derivatives of the function
  - Sample values f_i are constants, we can write approximation of gradient as

$$\langle \nabla f \rangle (\mathbf{x}) = \sum_i f_i \nabla \omega_h(\|\mathbf{x} - \mathbf{x}_i\|) V_i$$

$$\nabla \omega_h(\|\mathbf{x} - \mathbf{x}_i\|) = \frac{\mathbf{x} - \mathbf{x}_i}{\|\mathbf{x} - \mathbf{x}_i\|} \omega_h'(\|\mathbf{x} - \mathbf{x}_i\|)$$

Smoothed-Particle Hydrodynamics

- **Approximation of differential operators**
  - Other linear operators can be treated similarly

$$\langle \Delta f \rangle (\mathbf{x}) = \sum_i f_i \Delta \omega_h (\|\mathbf{x} - \mathbf{x}_i\|) V_i$$

$$\langle \nabla \cdot \mathbf{f} \rangle (\mathbf{x}) = \sum_i \mathbf{f}_i \cdot \nabla \omega_h (\|\mathbf{x} - \mathbf{x}_i\|) V_i$$

  - Accuracy of the approximations of derivative
    - Strongly depends on the distribution of sample points within the support region
    - For highly irregular sample distributions, the differential properties can be very noisy

Smoothed-Particle Hydrodynamics

- **Approximation of differential operators**
  - Problem with previous estimation
    - Gradient approximation can yield non-zero values even if the function is constant
  - How to rectify?
    - Enforce a zero gradient for constant functions by subtracting the constant f_i

$$
\begin{aligned}
\nabla f(\mathbf{x}_i) &\approx \langle \nabla [f - f_i] \rangle (\mathbf{x}_i) \\
&= \sum_j \left(f_j - f_i\right) \nabla \omega_h(\|\mathbf{x}_i - \mathbf{x}_j\|) V_j
\end{aligned}
$$

Smoothed-Particle Hydrodynamics

- **Approximation of differential operators**
  - Same reasoning applied to the divergence and Laplace operators

$$
\begin{aligned}
\langle \nabla \cdot \mathbf{f} \rangle (\mathbf{x}_i) &= \sum_j \left(\mathbf{f}_j - \mathbf{f}_i\right) \cdot \nabla \omega_h(\|\mathbf{x}_i - \mathbf{x}_j\|)\, V_j \\
\langle \Delta f \rangle (\mathbf{x}_i) &= \sum_j \left(f_j - f_i\right) \Delta \omega_h(\|\mathbf{x}_i - \mathbf{x}_j\|)\, V_j
\end{aligned}
$$

Moving Least Square

- **Function approximation using moving least squares**
  - Why?
    - SPH method has in general poor accuracy
    - SPH method lacks zero order consistency
  - Shape function approximation

$$\langle f \rangle (\mathbf{x}) = \mathbf{p}^T (\mathbf{x}) \mathbf{a}$$

  - We wish to obtain the coefficient vector **a** that minimizes the error

$$E = \sum_i \omega_{h_i} (\|\mathbf{x} - \mathbf{x}_i\|) \left(\mathbf{p}^T (\mathbf{x}_i) \mathbf{a} - f_i \right)^2$$

Moving Least Square

- **Function approximation using moving least squares**
  - Minimization yields

$$\sum_i \omega_{h_i}(\|\mathbf{x}-\mathbf{x}_i\|)\mathbf{p}(\mathbf{x}_i)\left(\mathbf{p}^T(\mathbf{x}_i)\mathbf{a}-f_i\right)=\mathbf{0}$$

  - Solving the linear system

$$\mathbf{a}=\mathbf{M}(\mathbf{x})^{-1}\sum_i \omega_{h_i}(\|\mathbf{x}-\mathbf{x}_i\|)\mathbf{p}(\mathbf{x}_i)f_i$$

$$\mathbf{M}(\mathbf{x})=\sum_i \omega_{h_i}(\|\mathbf{x}-\mathbf{x}_i\|)\mathbf{p}(\mathbf{x}_i)\mathbf{p}^T(\mathbf{x}_i)$$

Moving Least Square

- **Function approximation using moving least squares**
  - Final approximation

$$\langle f \rangle (\mathbf{x}) = \mathbf{p}^T(\mathbf{x})\mathbf{a}$$
$$= \mathbf{p}^T(\mathbf{x})\mathbf{M}(\mathbf{x})^{-1}\sum_i \omega_{h_i}(\|\mathbf{x}-\mathbf{x}_i\|)\mathbf{p}(\mathbf{x}_i)f_i$$

$$\Downarrow$$

$$\langle f \rangle (\mathbf{x}) = \sum_i \Phi_i(\mathbf{x}) f_i$$

  - Shape function $\Phi_i(\mathbf{x}) = \omega_{h_i}(\|\mathbf{x}-\mathbf{x}_i\|)\mathbf{p}(\mathbf{x})^T\mathbf{M}(\mathbf{x})^{-1}\mathbf{p}(\mathbf{x}_i)$

Moving Least Square

- **Approximation of differential operators**
  - First-order derivatives of f_i are obtained as

$$\frac{\partial \langle f \rangle (\mathbf{x})}{\partial \mathbf{x}_{(k)}} = \sum_i \frac{\partial \Phi_i(\mathbf{x})}{\partial \mathbf{x}_{(k)}} f_i$$

$$\begin{aligned}\frac{\partial \Phi_i(\mathbf{x})}{\partial \mathbf{x}_{(k)}} = {} & \frac{\partial \omega_{h_i}(\|\mathbf{x}-\mathbf{x}_i\|)}{\partial \mathbf{x}_{(k)}} \mathbf{p}^T(\mathbf{x}) \mathbf{M}(\mathbf{x})^{-1} \mathbf{p}(\mathbf{x}_i) \\ & + \omega_{h_i}(\|\mathbf{x}-\mathbf{x}_i\|) \mathbf{p}^T(\mathbf{x}) \frac{\partial \mathbf{M}(\mathbf{x})^{-1}}{\partial \mathbf{x}_{(k)}} \mathbf{p}(\mathbf{x}_i) \\ & + \omega_{h_i}(\|\mathbf{x}-\mathbf{x}_i\|) \frac{\partial \mathbf{p}^T(\mathbf{x})}{\partial \mathbf{x}_{(k)}} \mathbf{M}(\mathbf{x})^{-1} \mathbf{p}(\mathbf{x}_i)\end{aligned}$$

  - The derivative of inverted matrix $\frac{\partial (\mathbf{M}^{-1})}{\partial \mathbf{x}_{(k)}} = -\mathbf{M}^{-1} (\frac{\partial \mathbf{M}}{\partial \mathbf{x}_{(k)}}) \mathbf{M}^{-1}$

Moving Least Square

- **Approximation of differential operators**
  - Spatial gradient

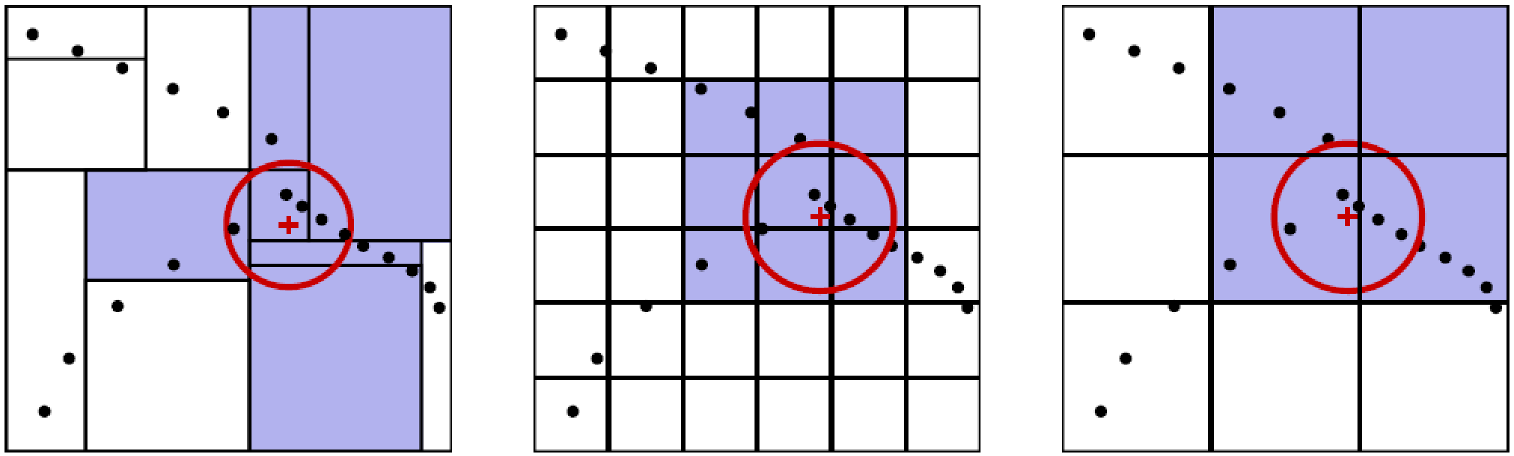
$$\langle \nabla f \rangle (\mathbf{x}) = \sum_i \nabla \Phi_i(\mathbf{x}) f_i$$

  - Divergence of a vector-valued function

$$\langle \nabla \cdot \mathbf{f} \rangle (\mathbf{x}) = \sum_i \mathbf{f}_i \cdot \nabla \Phi_i(\mathbf{x})$$

Neighbor Search Data Structures

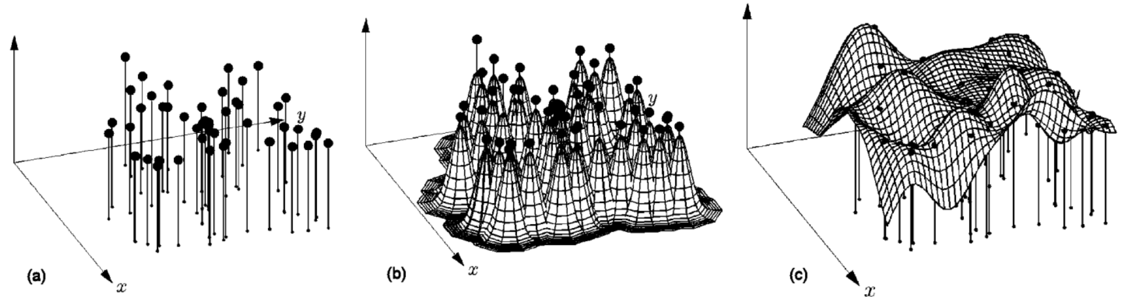
- **KD tree v.s. uniform grid**



Radial Basis Functions

- **A real-valued function**
  - Value depends only on the distance from the origin
  - Alternatively on the distance from some other point
- **Formulation**
  - RBF interpolation of f(x)

$$s(\mathbf{x}) = \sum_{k=1}^{N} \lambda_k \phi(||\mathbf{x} - \mathbf{x}_k||)$$



Radial Basis Functions

- **Solving the RBF interpolation**

$$\begin{bmatrix} \phi(||\mathbf{x}_1 - \mathbf{x}_1||) & \phi(||\mathbf{x}_1 - \mathbf{x}_2||) & \cdots & \phi(||\mathbf{x}_1 - \mathbf{x}_N||) \\ \phi(||\mathbf{x}_2 - \mathbf{x}_1||) & \phi(||\mathbf{x}_2 - \mathbf{x}_2||) & \cdots & \phi(||\mathbf{x}_2 - \mathbf{x}_N||) \\ \vdots & \vdots & & \vdots \\ \phi(||\mathbf{x}_N - \mathbf{x}_1||) & \phi(||\mathbf{x}_N - \mathbf{x}_2||) & \cdots & \phi(||\mathbf{x}_N - \mathbf{x}_N||) \end{bmatrix} \begin{bmatrix} \lambda_1 \\ \lambda_2 \\ \vdots \\ \lambda_N \end{bmatrix} = \begin{bmatrix} f_1 \\ f_2 \\ \vdots \\ f_N \end{bmatrix}$$

- **Types of RBFs**
  - Piecewise smooth RBFs

Polyharmonic spline (PHS)	$r^m,\ m = 1, 3, 5, \ldots$
	$r^m log(r),\ m = 2, 4, 6, \ldots$
Compact support ('Wendland')	$(1 - \varepsilon r)_+^m p(\varepsilon r)$, p certain polynomials

Radial Basis Functions

- **Types of RBFs**
  - Infinitely smooth RBFs

Gaussian (GA)	$e^{-(\varepsilon r)^2}$
Multiquadric (MQ)	$\sqrt{1+(\varepsilon r)^2}$
Inverse Quadratic (IQ)	$1/(1+(\varepsilon r)^2)$
Inverse Multiquadric (IMQ)	$1/\sqrt{1+(\varepsilon r)^2}$
Bessel (BE) $(d = 1, 2, \ldots)$	$J_{d/2-1}(\varepsilon r)/(\varepsilon r)^{d/2-1}$

Radial Basis Functions

- **Solving Poisson's equation**
  - Poisson problem in N-dimension

$$\begin{cases} u(\mathbf{x}) & = g(\mathbf{x}) \quad \text{on boundary } \partial\Omega \\ \Delta u(\mathbf{x}) & = f(\mathbf{x}) \quad \text{in interior of } \Omega \end{cases}$$

  - Kansa's formulation

$$u(\mathbf{x}) = \sum_{j=1}^{N} \lambda_j \phi(||\mathbf{x} - \mathbf{x}_j||)$$

$$\begin{bmatrix} \phi(||\mathbf{x} - \mathbf{x}_j||)|_{\mathbf{x}=\mathbf{x}_i} \\ ----------- \\ \triangle\phi(||\mathbf{x} - \mathbf{x}_j||)|_{\mathbf{x}=\mathbf{x}_i} \end{bmatrix} \begin{bmatrix} \underline{\lambda} \end{bmatrix} = \begin{bmatrix} \underline{g} \\ - \\ \underline{f} \end{bmatrix}$$

Radial Basis Functions

- **Solving Poisson's equation**
  - Symmetric formulation

$$u(\mathbf{x}) = \sum_{j=1}^{N_B} \lambda_j \phi(||\mathbf{x} - \mathbf{x}_j||) + \sum_{j=N_B+1}^{N} \lambda_j \triangle\phi(||\mathbf{x} - \mathbf{x}_j||)$$

$$\left[\begin{array}{c|c} \phi & \triangle\phi \\ \hline \triangle\phi & \triangle^2\phi \end{array}\right] \left[\begin{array}{c} \underline{\lambda} \end{array}\right] = \left[\begin{array}{c} \underline{g} \\ - \\ \underline{f} \end{array}\right]$$

Next Lecture: Rigid Body Dynamics I