A guide to the numerical zoo

AMCS 252, Spring 2025

David Ketcheson

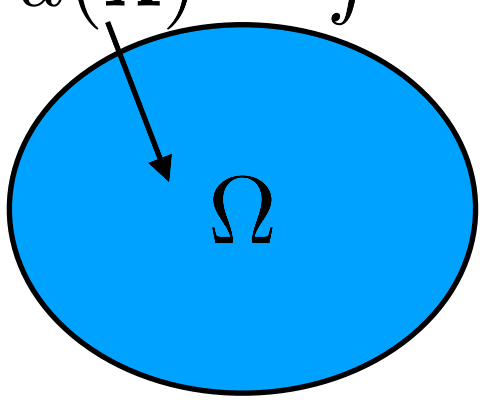
The purpose of this lecture

- Introduce some general classifications of numerical methods
- Give you a basic idea of how each kind works and what their advantages are
- We won't go into details

Three types of problems

- Steady state
    - boundary -value problem
    - Solution doesn't change in time

$$\nabla^2 u(\mathbf{x}) = f$$



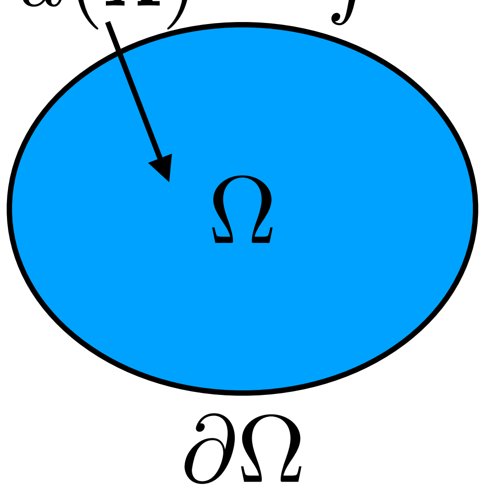
$\partial\Omega$

$$u(\mathbf{x}) = g(\mathbf{x}) \quad (\mathbf{x} \in \partial\Omega)$$

Three types of problems

- Steady state
  - boundary -value problem
  - Solution doesn't change in time

$$\nabla^2 u(\mathbf{x}) = f$$



$$u(\mathbf{x}) = g(\mathbf{x}) \quad (\mathbf{x} \in \partial\Omega)$$

- Time-dependent
  - Initial-value problem
  - Solution changes in time

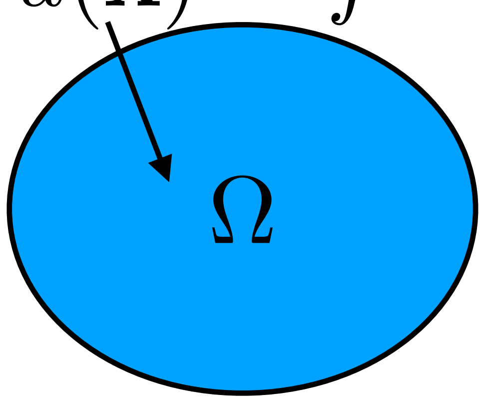
$$u'(t) = f(t, u)$$

$$u(0) = u_0$$

Three types of problems

- Steady state
  - boundary -value problem
  - Solution doesn't change in time

$$\nabla^2 u(\mathbf{x}) = f$$



$\partial\Omega$

$$u(\mathbf{x}) = g(\mathbf{x}) \quad (\mathbf{x} \in \partial\Omega)$$

- Time-dependent
  - Initial-value problem
  - Solution changes in time

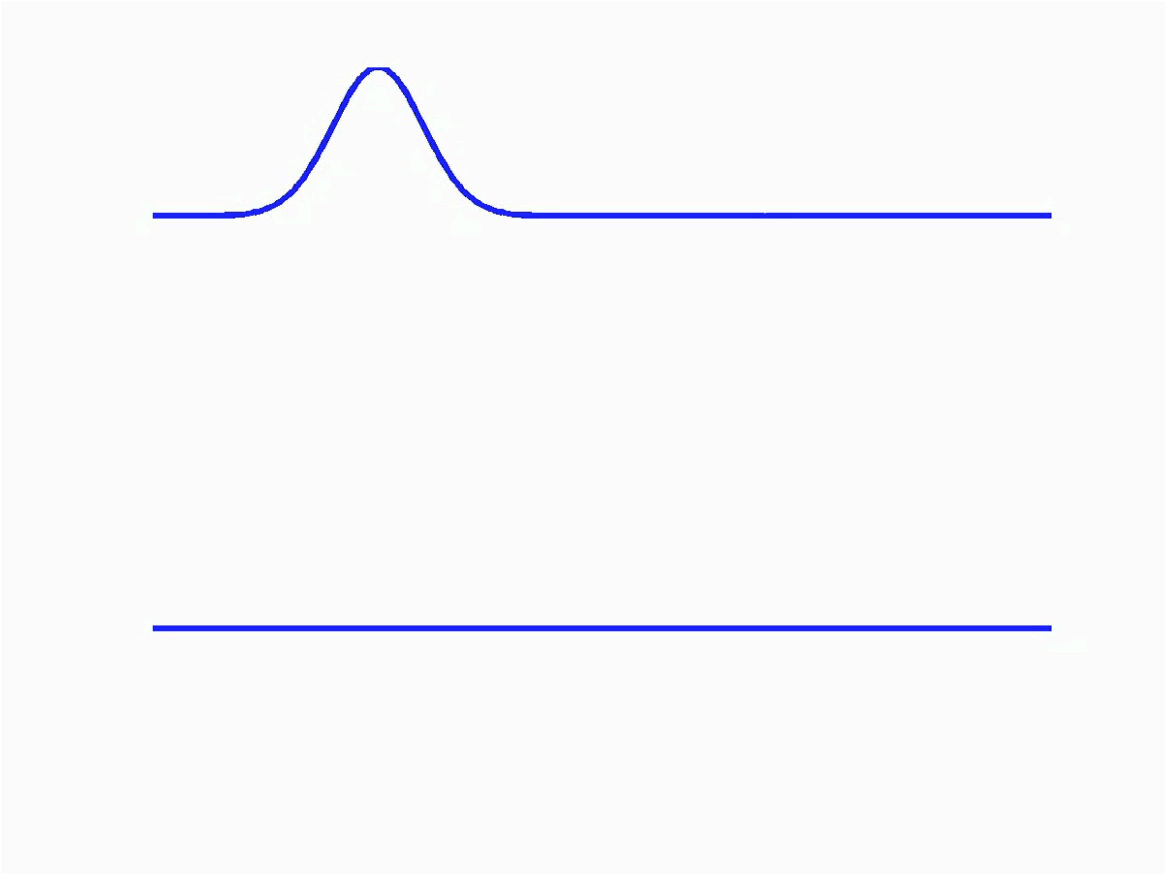
$$u'(t) = f(t, u)$$
$$u(0) = u_0$$

- Initial boundary value problem

$$u_t = \nabla^2 u(\mathbf{x}) - f$$
$$u(t, \mathbf{x} \in \partial\Omega) = g(\mathbf{x})$$
$$u(t = 0, \mathbf{x}) = u_0(\mathbf{x})$$

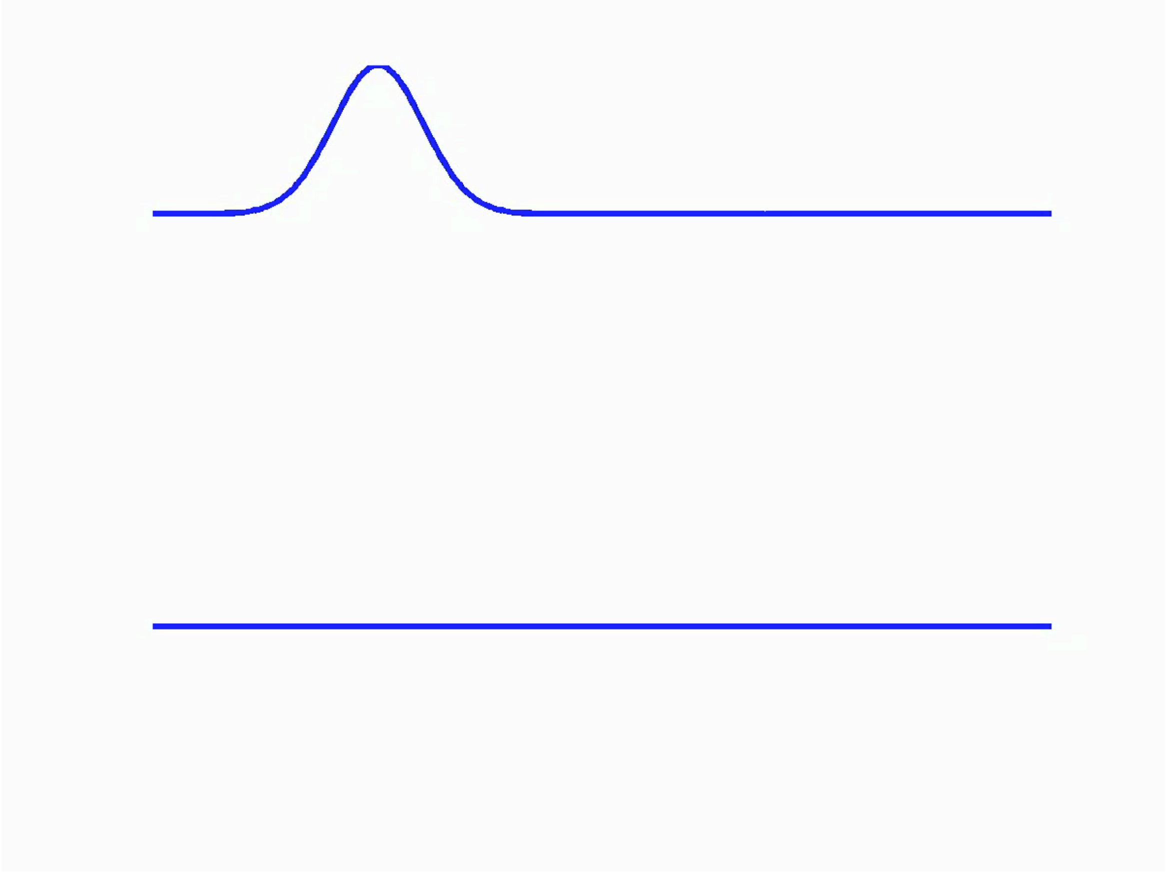
Linear vs. Nonlinear problems

- Linear problems
  - May be exactly solvable
  - Can use techniques like superposition
  - Discretizations lead to linear algebra
  - Examples:
    - diffusion of heat
    - electromagnetic waves
    - acoustic waves
    - gravitational potential



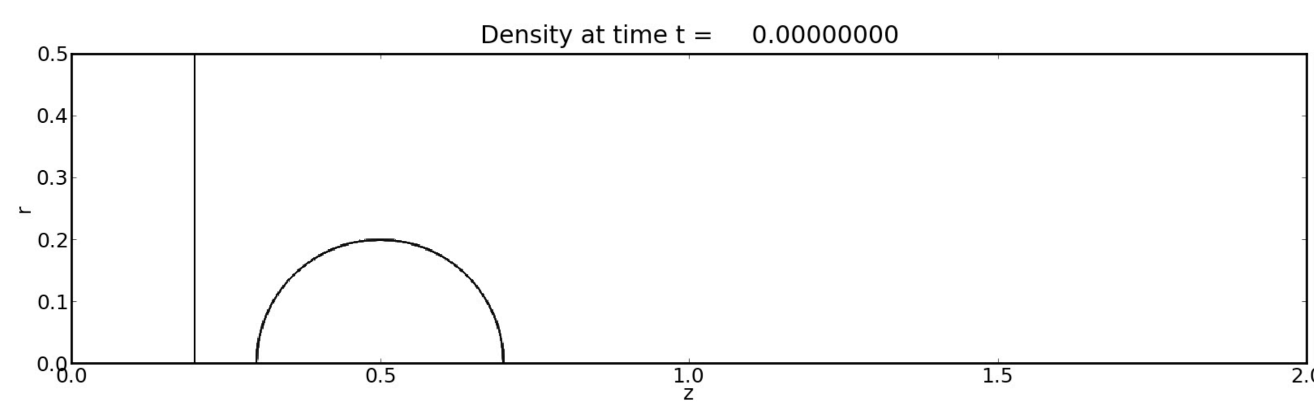
Linear vs. Nonlinear problems

- Linear problems
  - May be exactly solvable
  - Can use techniques like superposition
  - Discretizations lead to linear algebra
  - Examples:
    - diffusion of heat
    - electromagnetic waves
    - acoustic waves
    - gravitational potential



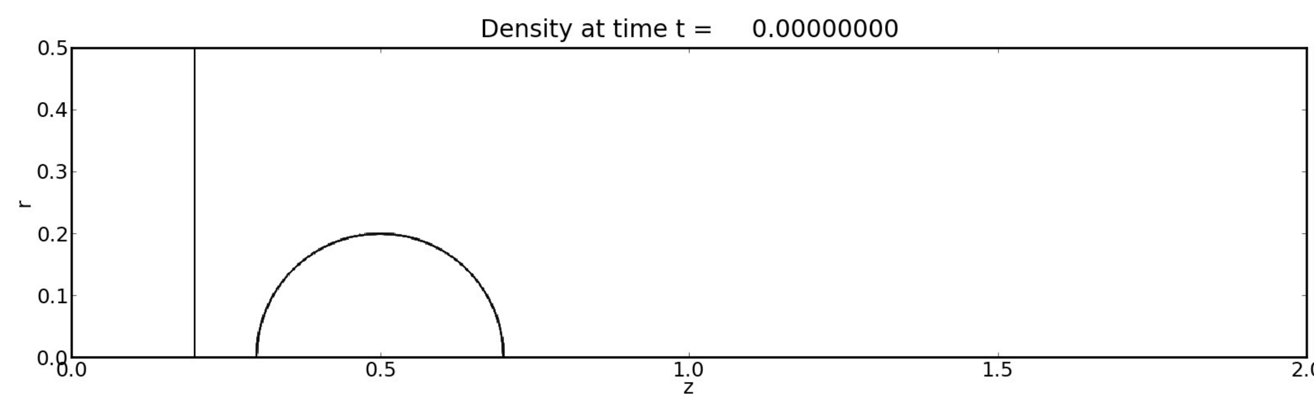
Linear vs. Nonlinear problems

- Nonlinear problems
  - Rarely have exact solutions
  - No superposition
  - Discretizations lead to nonlinear algebra
  - Solution may not be unique
  - Examples:
    - Pendulum
    - Spread of disease
    - Water waves
    - Fluid dynamics



Linear vs. Nonlinear problems

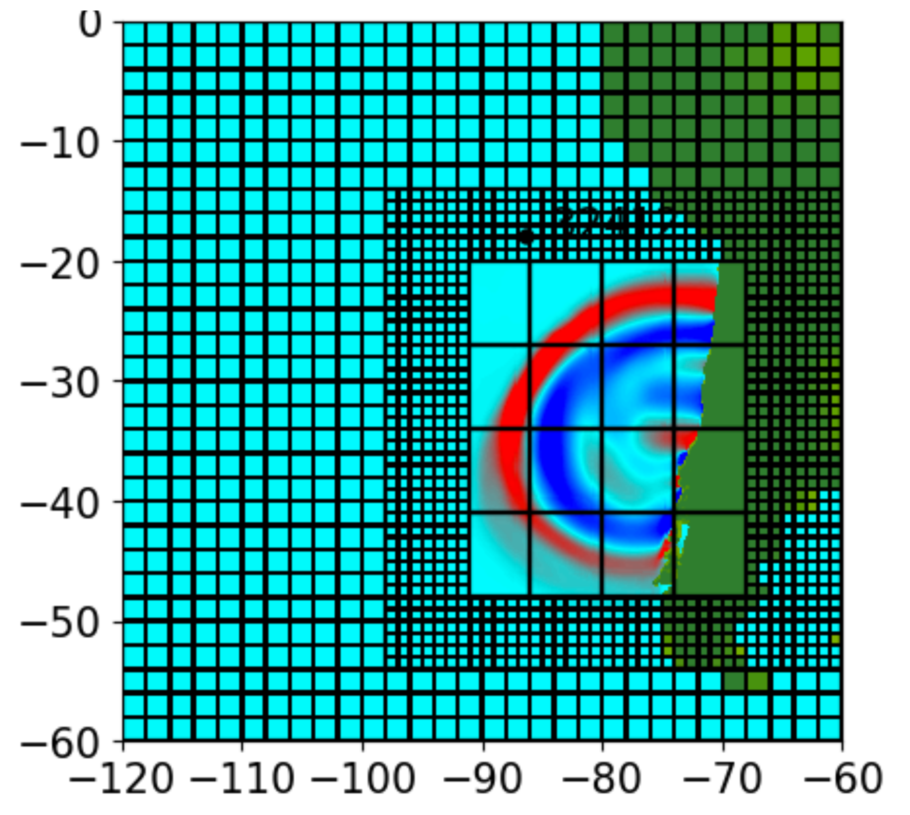
- Nonlinear problems
  - Rarely have exact solutions
  - No superposition
  - Discretizations lead to nonlinear algebra
  - Solution may not be unique
  - Examples:
    - Pendulum
    - Spread of disease
    - Water waves
    - Fluid dynamics



Discretization

- Solutions of ODEs and PDEs live in continuous, infinite-dimensional spaces!

- To compute with them, we must replace those with discrete spaces that are finite-dimensional

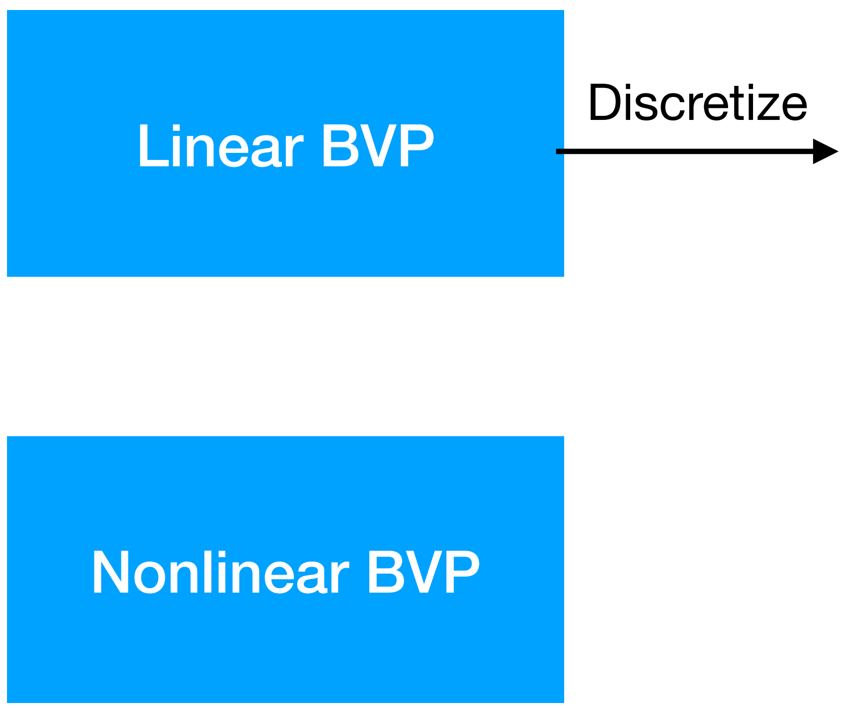


Discretization of boundary value problems

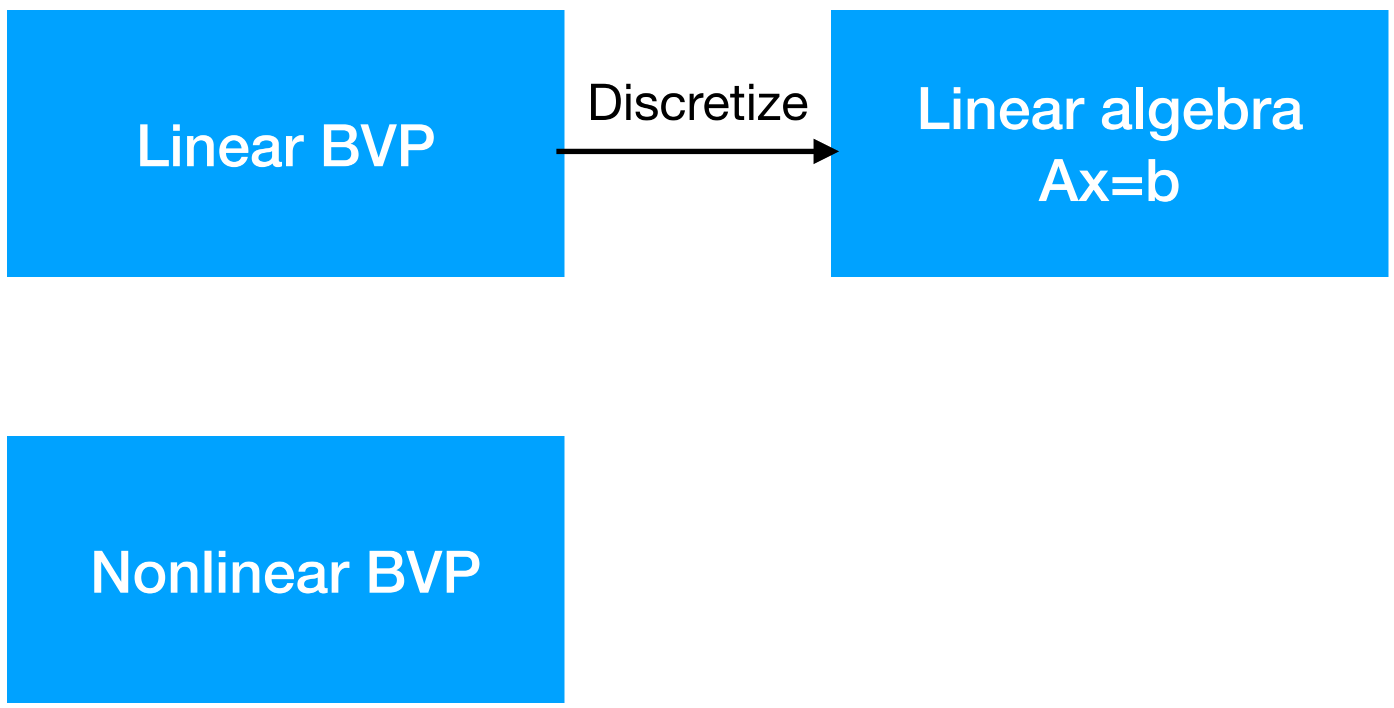
Linear BVP

Nonlinear BVP

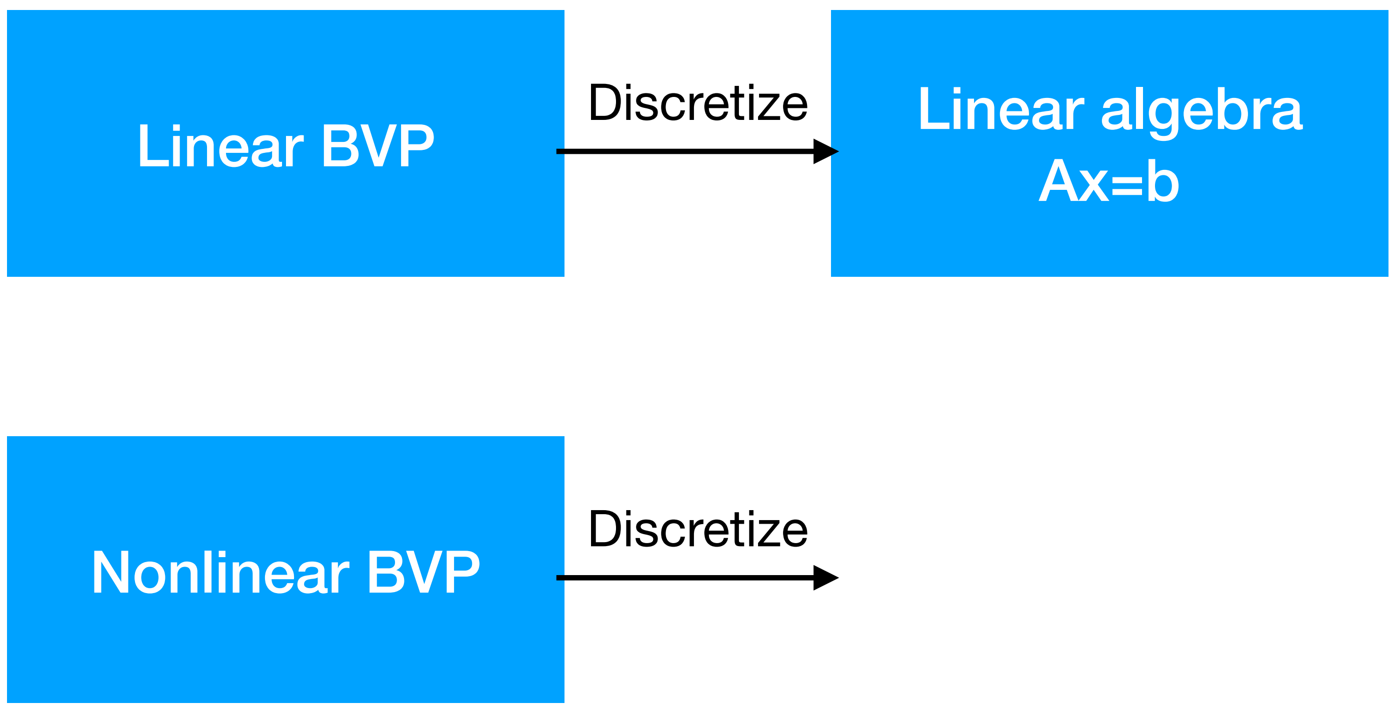
Discretization of boundary value problems



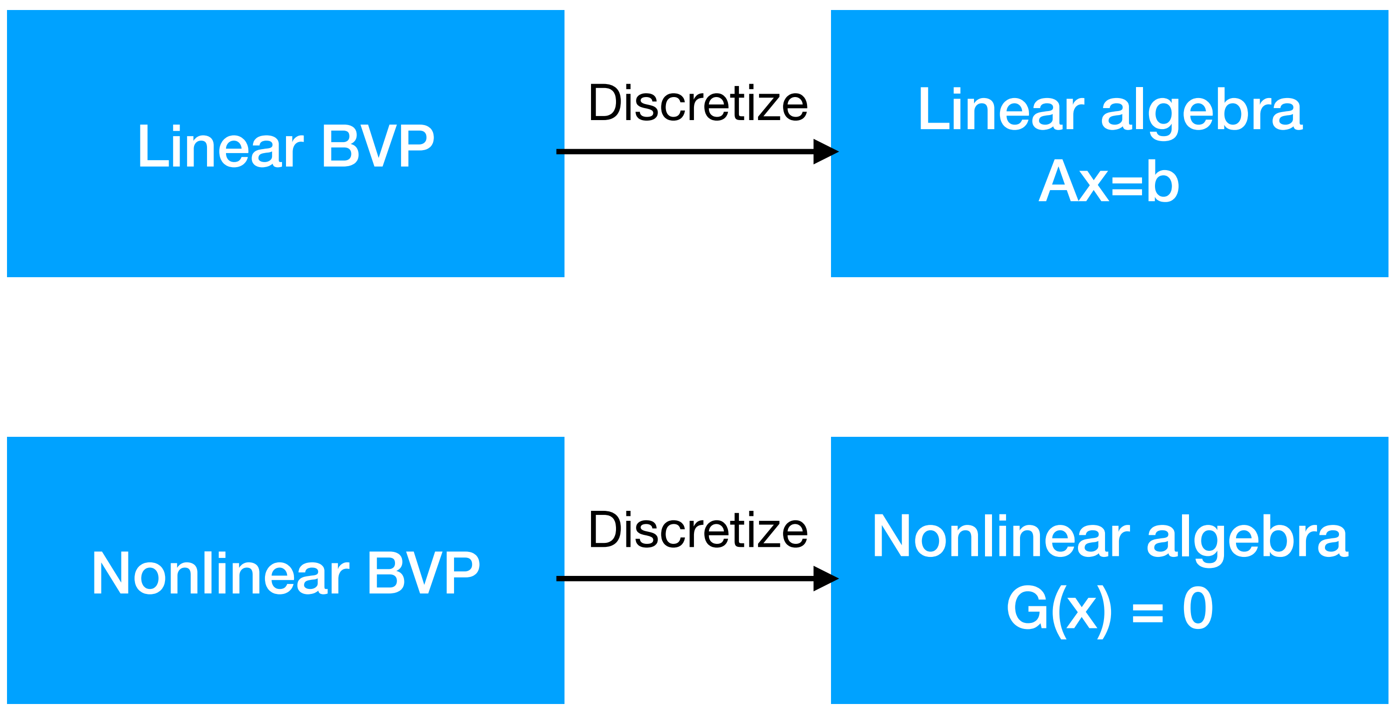
Discretization of boundary value problems



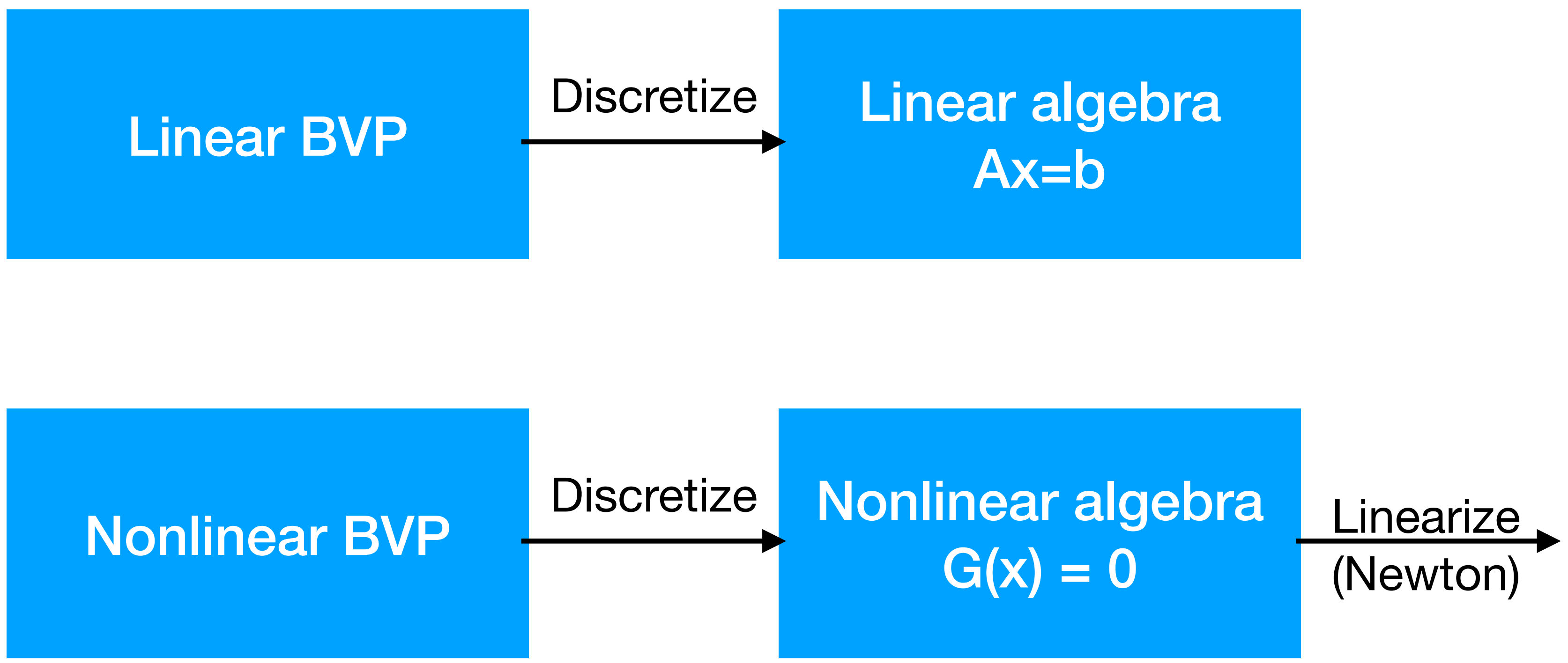
Discretization of boundary value problems



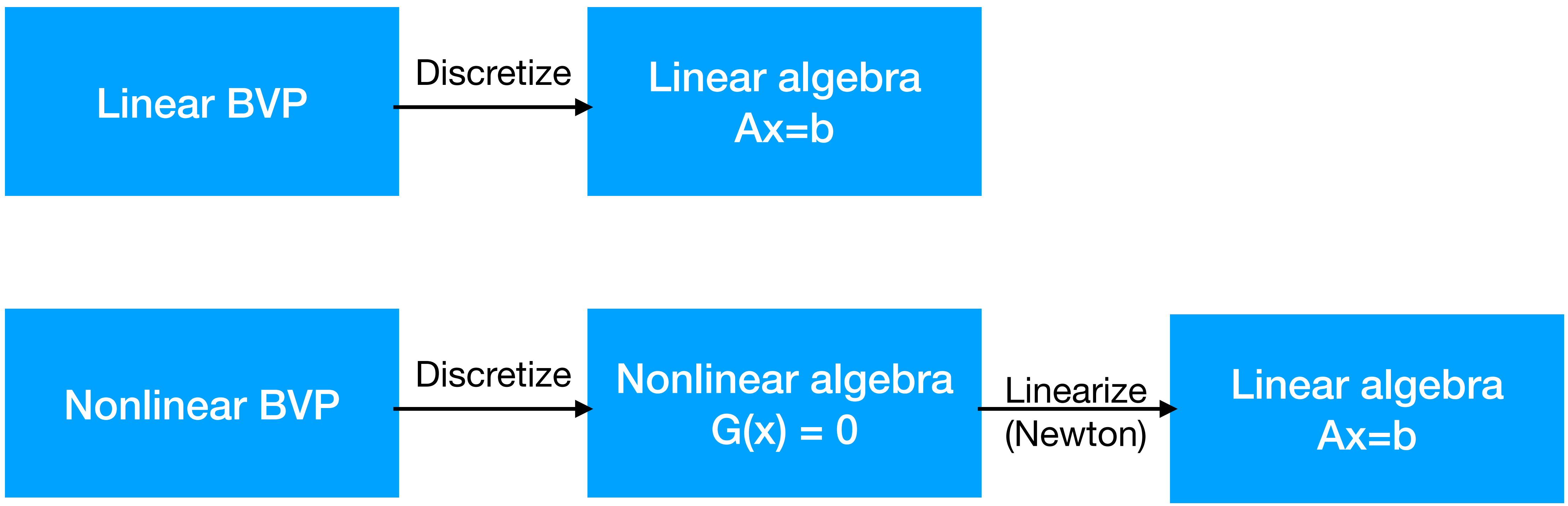
Discretization of boundary value problems



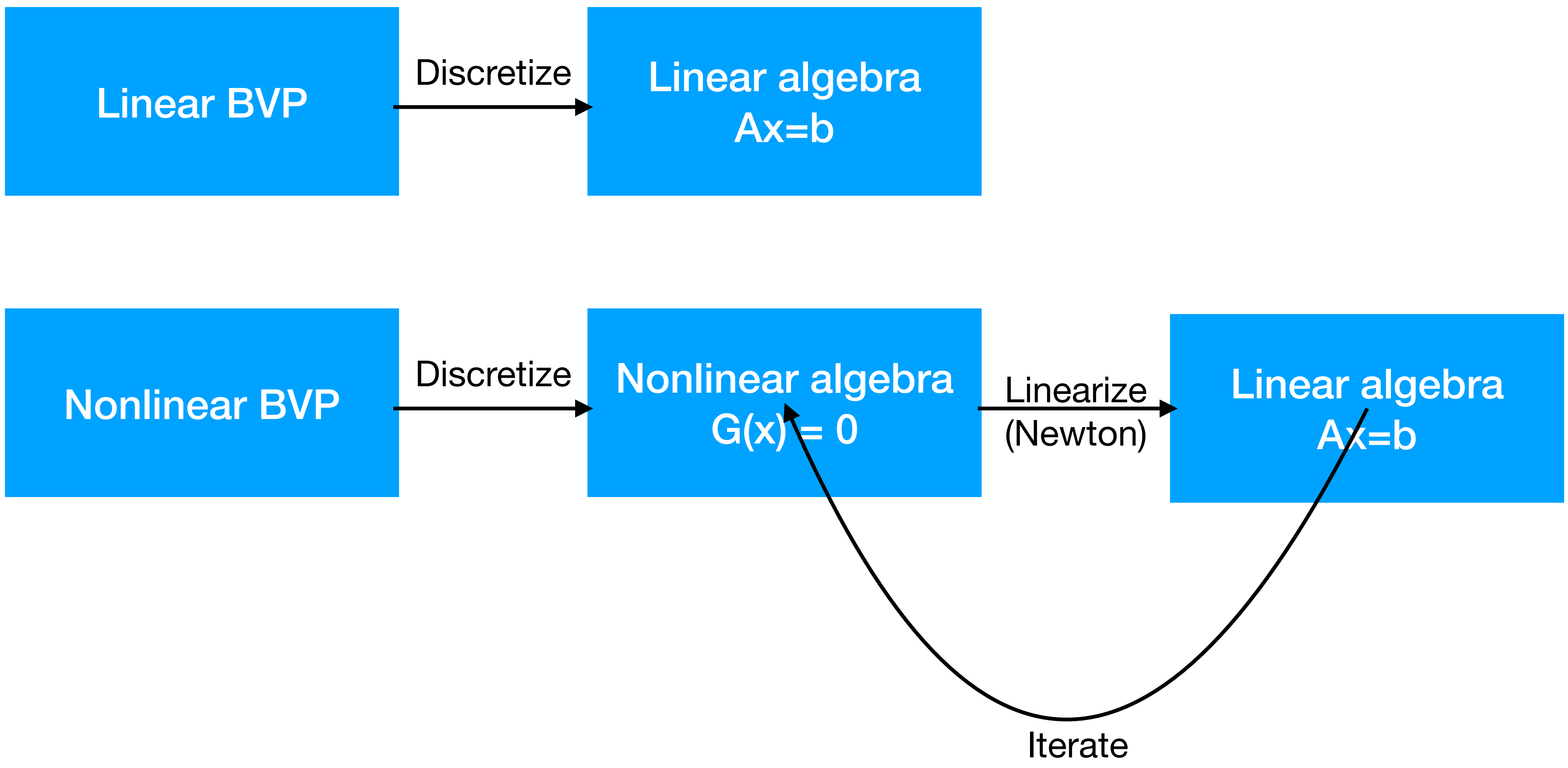
Discretization of boundary value problems



Discretization of boundary value problems



Discretization of boundary value problems



3 “finite”s

- Finite differences

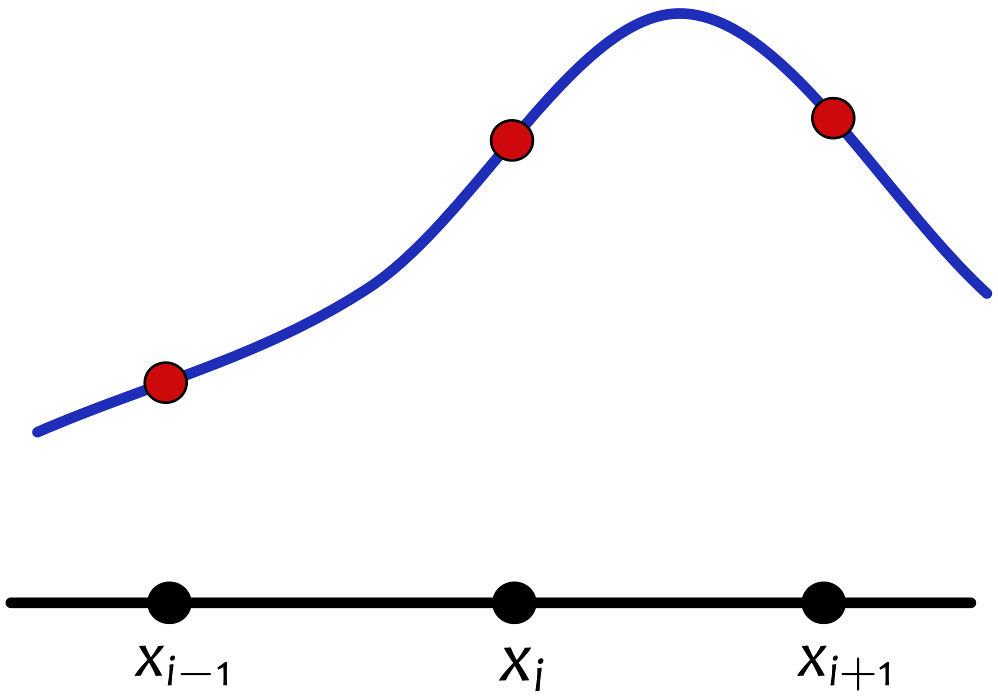
- Finite volumes

- Finite elements

3 “finite”s

- Finite differences

  $u(x_i, t)$



- Finite volumes

- Finite elements

3 “finite”s

- Finite differences

  $u(x_i, t)$

- Finite volumes

- Finite elements

3 “finite”s

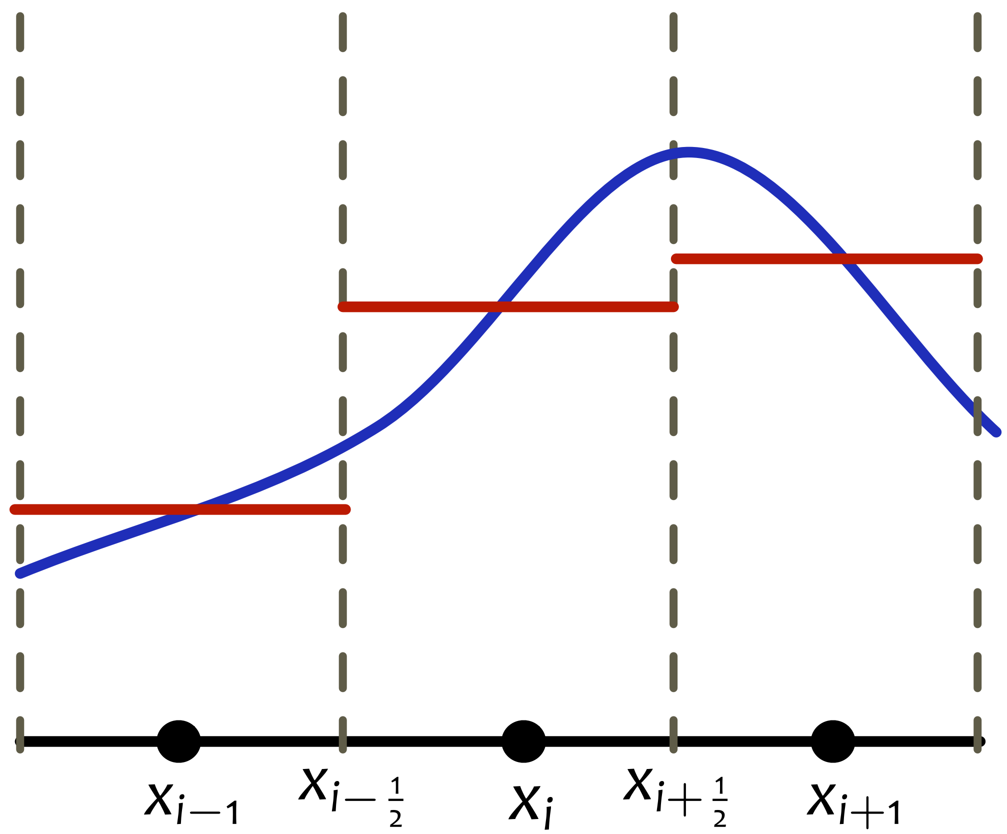
- Finite differences

$$u(x_i, t)$$

- Finite volumes

$$\int_{x_{i-1/2}}^{x_{i+1/2}} u(x,t)dx$$

- Finite elements



3 "finite"s

- Finite differences

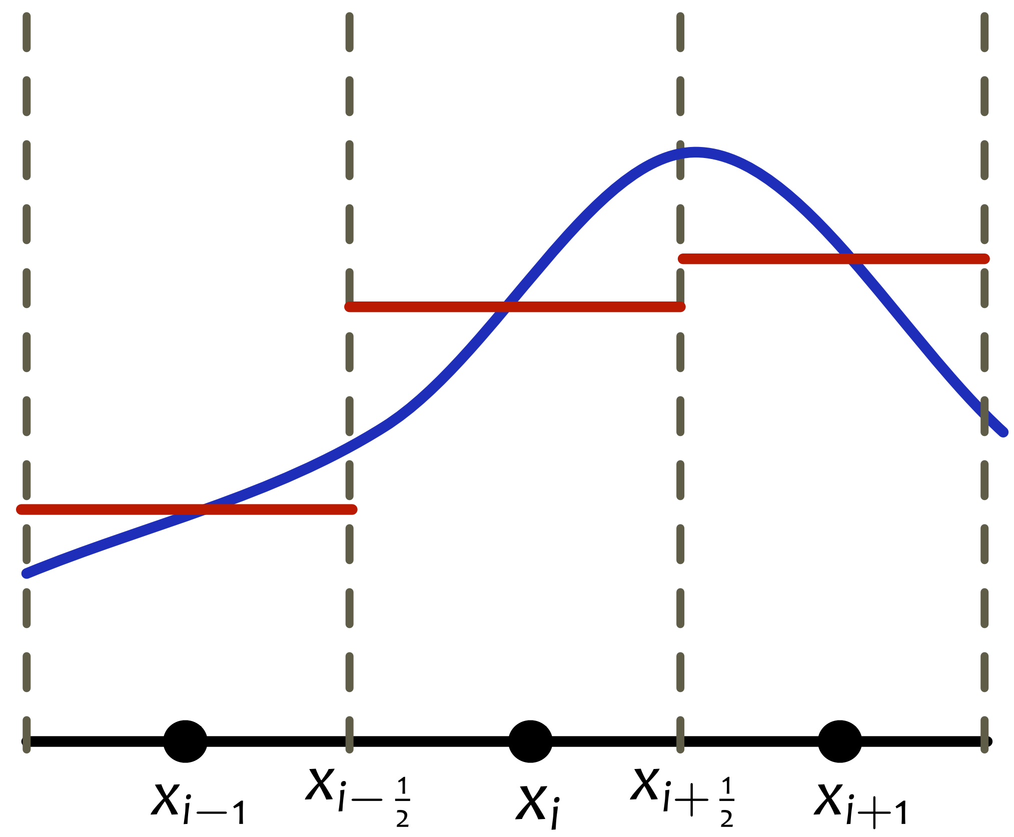
$$u(x_i, t)$$

- Finite volumes

$$\int_{x_{i-1/2}}^{x_{i+1/2}} u(x,t)dx$$

- Finite elements

$$\int_{x_{i-1/2}}^{x_{i+1/2}} u(x,t)\phi_j(x)dx$$



3 “finite”s

FDM

- Easier to code
- Lower computational cost
- Works best with simple geometries
- Simplest to understand

FVM

Falls somewhere in-between

FEM

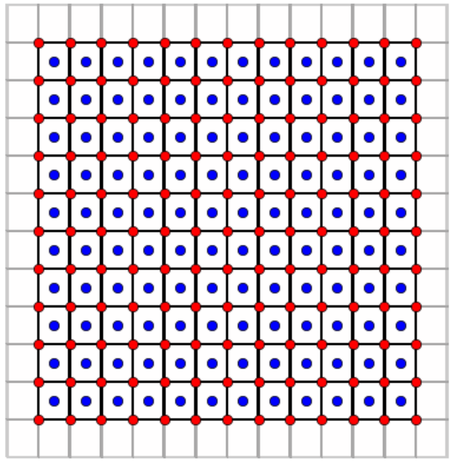
- Harder to code
- Higher computational cost
- Works easily with complex geometries
- More sophisticated mathematical foundation

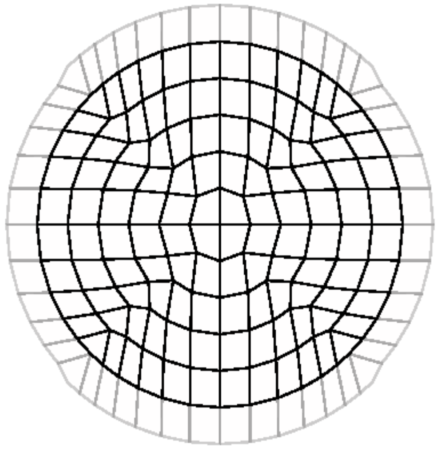
Basis functions

- Spectral methods
  - Basis functions have global support
  - More accurate for smooth solutions
  - Work best in simple geometries
  - Dense linear algebra

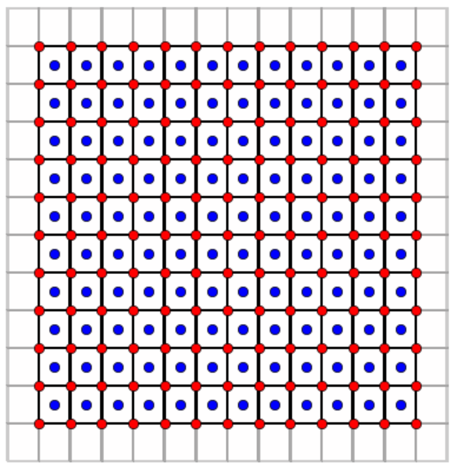
- “Local” methods
  - Basis functions have compact support
  - More suited to non-smooth solutions
  - More suited to complex geometries
  - Sparse linear algebra

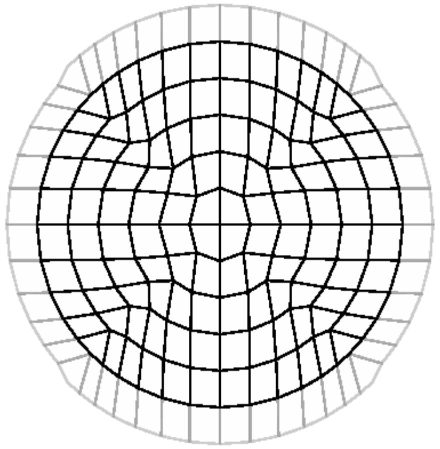
Grids





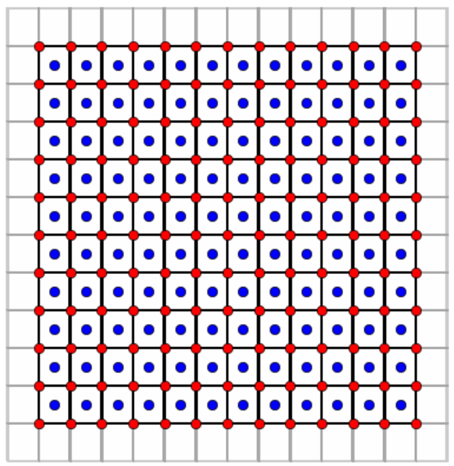
Grids

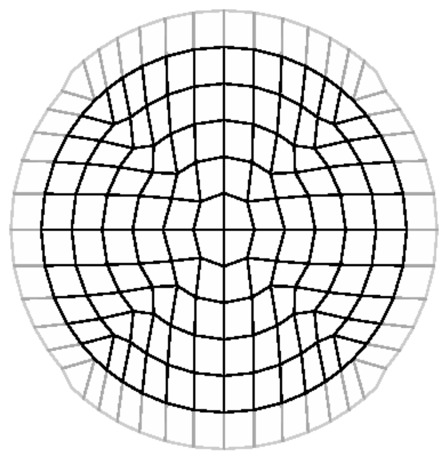




- Structured grid

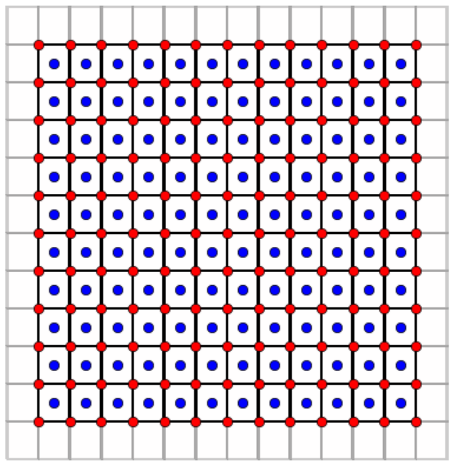
Grids

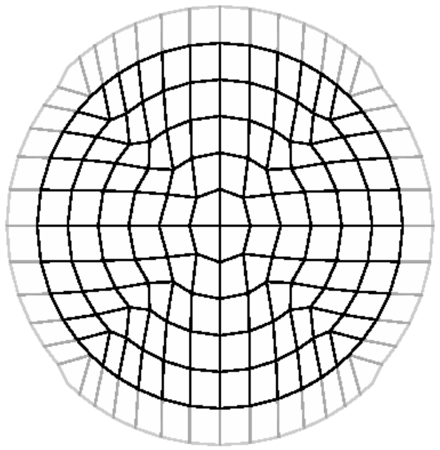




- Structured grid
  - Natural ordering

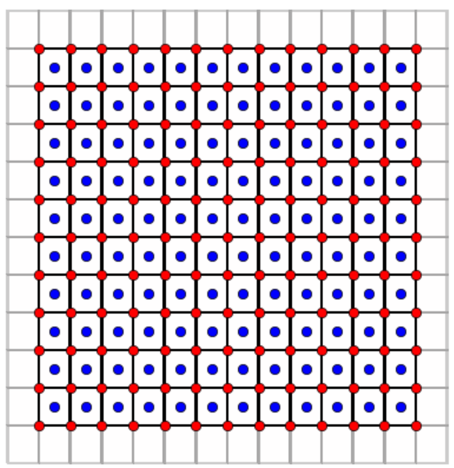
Grids

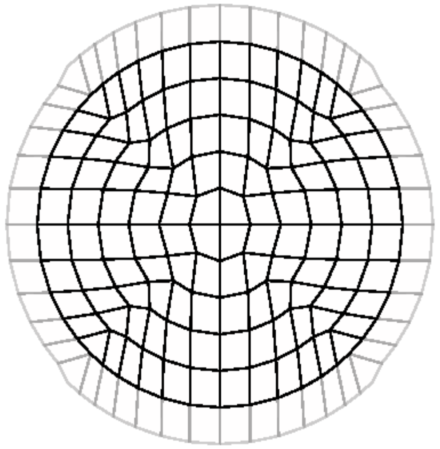




- Structured grid
  - Natural ordering
  - Neighbors are obvious

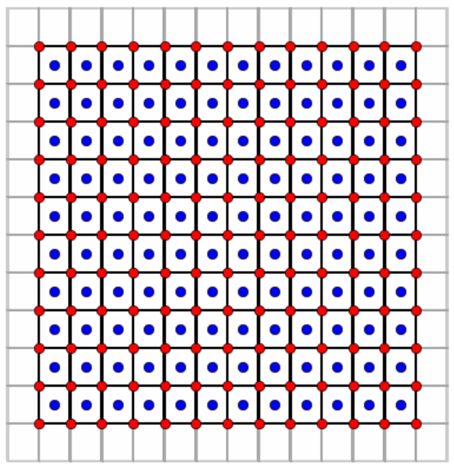
Grids

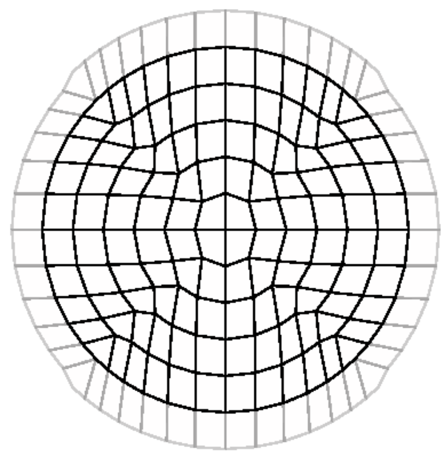


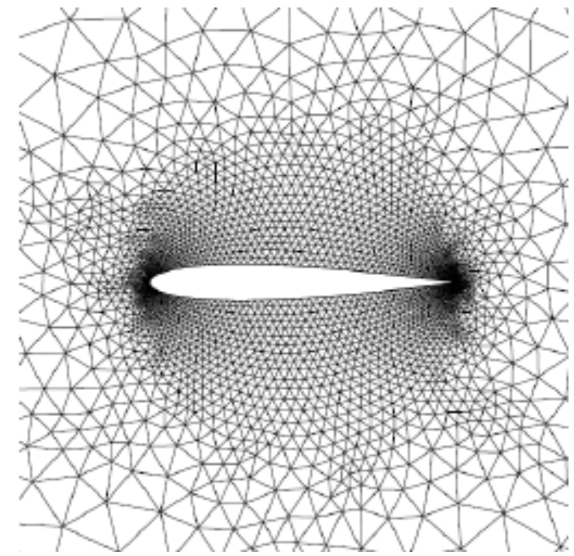


- Structured grid
  - Natural ordering
  - Neighbors are obvious
  - Computationally efficient

Grids

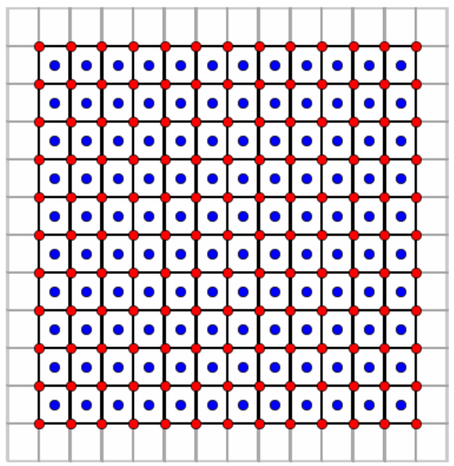


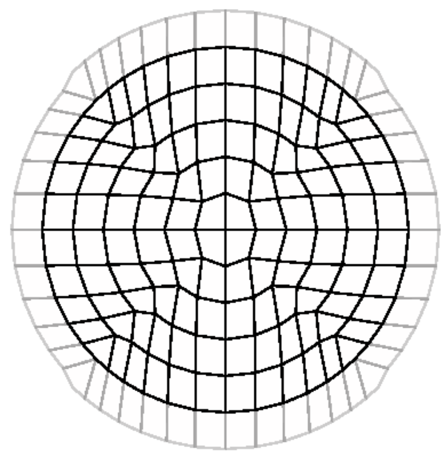


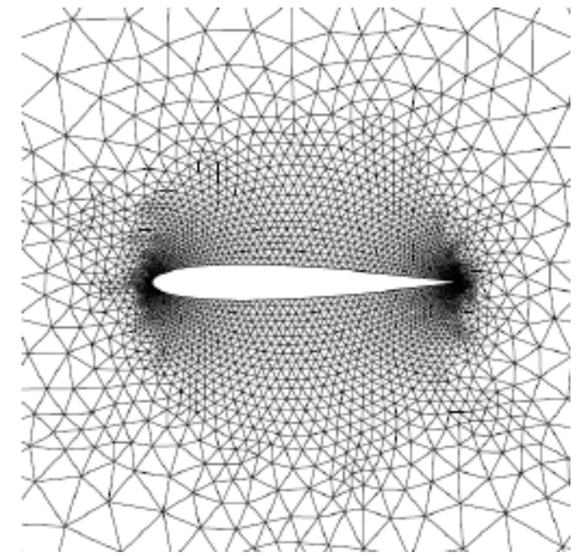


- Structured grid
    - Natural ordering
    - Neighbors are obvious
    - Computationally efficient

Grids







- Structured grid
  - Natural ordering
  - Neighbors are obvious
  - Computationally efficient

- Unstructured grids
  - Need lookup table for neighbors
  - Arbitrary geometries
  - Local refinement

Explicit vs. Implicit

$$u'(t) = f(u)$$

Explicit vs. Implicit

$$u'(t) = f(u)$$

$$u_{n+1} = u_n + \Delta t f(u_n)$$

Explicit vs. Implicit

$$u'(t) = f(u)$$

$$u_{n+1} = u_n + \Delta t f(u_n)$$

- Explicit
  - Easier to program
  - Low cost per step
  - Efficient for wave equations (hyperbolic PDEs)

Explicit vs. Implicit

$$u'(t) = f(u)$$

$$u_{n+1} = u_n + \Delta t f(u_n) \qquad\qquad u_{n+1} = u_n + \Delta t f(u_{n+1})$$

- Explicit
  - Easier to program
  - Low cost per step
  - Efficient for wave equations (hyperbolic PDEs)

Explicit vs. Implicit

$$u'(t) = f(u)$$

$$u_{n+1} = u_n + \Delta t f(u_n) \qquad\qquad u_{n+1} = u_n + \Delta t f(u_{n+1})$$

- Explicit
  - Easier to program
  - Low cost per step
  - Efficient for wave equations (hyperbolic PDEs)
- Implicit
  - Harder to program
  - Higher cost per step
  - Efficient for stiff probems (elliptic, parabolic PDEs)

Accuracy and cost

- Goal 1: Achieve a desired level of accuracy for the lowest cost
- Goal 2: Achieve the greatest possible accuracy for a given cost
- “Method x is expensive” or “method y is fast” isn’t meaningful