Computational Modeling with Partial Differential Equations (PDEs)

Chad Westphal

westphac@wabash.edu

Department of Math and Computer Science

Wabash College

Physical Phenomena described by PDEs

Porous Media	Fluid Mechanics	Elasticity
Electrostatics	Dynamical Systems	Quantum mechanics
Advection	Combustion	Atmospheric Sciences
Plasma Physics	MHD	Multiphase Flow
Coupled Systems	Seismic Modeling	Astrophysics
Finance	Ocean Modeling	Biological Modeling

Who else is interested in this?

- Industry: Engineering (Auto, Aerospace, etc.), Financial, Defense
- National Labs / Universities: Physicists, Chemists, Biologists, Engineers, etc.

Computational Modeling Process

Task	Those Involved	Issues
Observe and Measure Data	Engineers, Physicists, etc.	Simplicity
Math Model	Physicists, Mathematicians	Well-posedness
Discrete Model	Mathematicians	Accuracy, conservation
Numerical Solution	Mathematicians, Computer Scientists	Computational cost (storage, time, etc.)
Visualize Data	Computer Scientists, Engineers	Data Structures, architecture
Interpret Results	Engineers, Physicists, etc.	Calibration

Computational Modeling Process

Task	Those Involved	Issues
Observe and Measure Data	Engineers, Physicists, etc.	Simplicity
Math Model	Physicists, Mathematicians	Well-posedness
Discrete Model	Mathematicians	Accuracy, conservation
Numerical Solution	Mathematicians, Computer Scientists	Computational cost (storage, time, etc.)
Visualize Data	Computer Scientists, Engineers	Data Structures, architecture
Interpret Results	Engineers, Physicists, etc.	Calibration

What are PDEs?

Partial Differential Equations (PDEs) are equations involving partial derivatives of functions that depend on more than one variable.

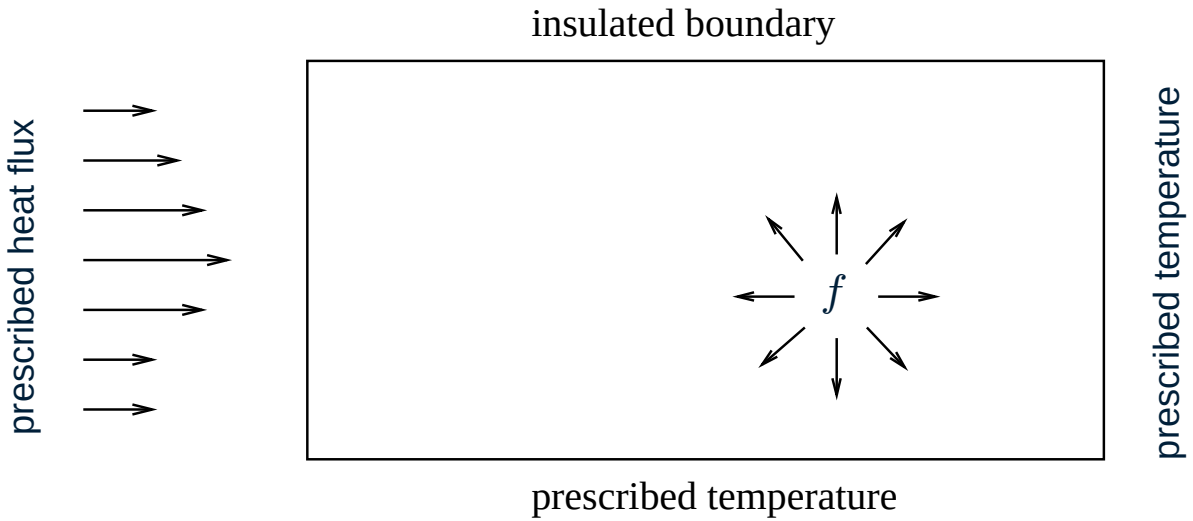
May include: systems (many unknowns), time dependent, nonlinear or stochastic behavior, higher dimensions, etc.

Example: By conservation of energy and Fourier's law of heat conduction we can derive

$$-\Delta u = -\partial_{xx}^2 u - \partial_{yy}^2 u = f$$

$u = u(x, y)$ the temperature at any point in a 2-D object

$f = f(x, y)$ internal sources of energy

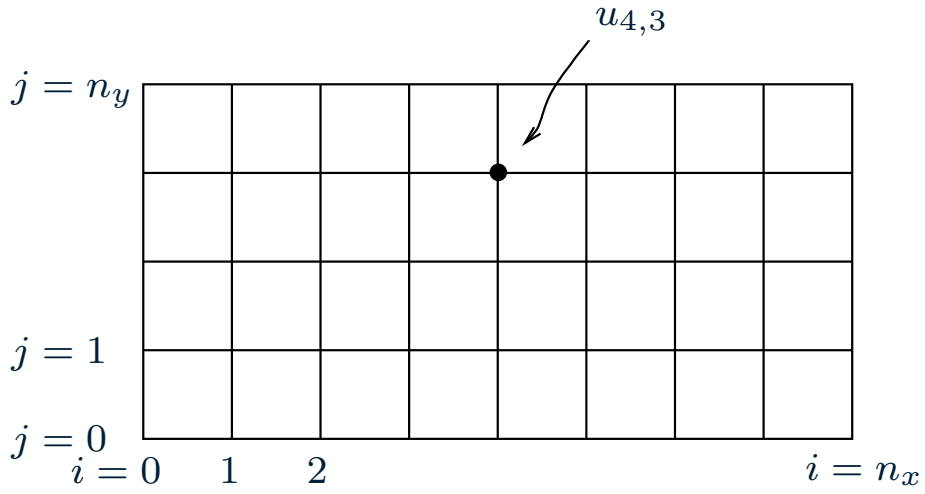


$$\begin{cases} -\Delta u = f & \Omega = [0,2] \times [0,1] \\ \partial_y u = 0 & y = 1 \\ u = 0 & x = 2, y = 0 \\ \partial_x u = g(y) & x = 0 \end{cases}$$

We want to find $u(x, y)$, the temperature in Ω, from the relation above

Sometimes it is possible to find a solution to these PDEs “by hand.” In most practical applications it becomes too difficult.

Idea: Replace the continuous domain with a discrete domain (grid):

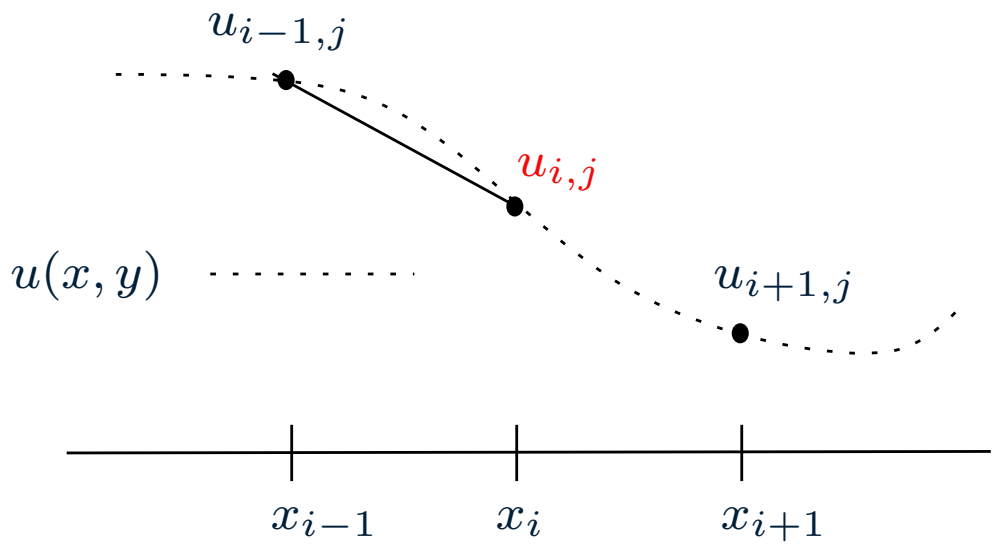


Now look for the solution only at the nodes: $u(x, y) \rightarrow u_{i,j} = u(h_x i, h_y j)$

$$i = 0, 1, 2, ..., n_x \qquad j = 0, 1, 2, ..., n_y$$

$$\triangle x = h_x = \frac{1}{n_x} \qquad \triangle y = h_y = \frac{1}{n_y}$$

Discretization: Replace the derivatives by finite differences (via truncated Taylor series)

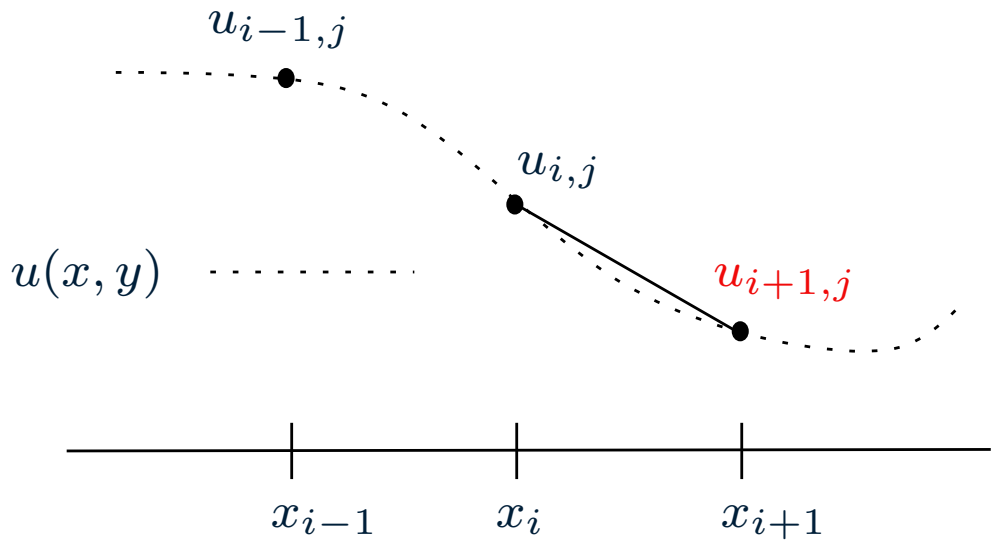


$$\partial_x u_{i,j} \approx \frac{u_{i,j} - u_{i-1,j}}{h_x}$$

$$\partial_x u_{i+1,j} \approx \frac{u_{i+1,j} - u_{i,j}}{h_x}$$

$$\partial_{xx} u_{i,j} \approx \frac{\partial_x u_{i+1,j} - \partial_x u_{i,j}}{h_x} = \frac{u_{i+1,j} - 2u_{i,j} + u_{i-1,j}}{h_x^2}$$

Discretization: Replace the derivatives by finite differences (via truncated Taylor series)

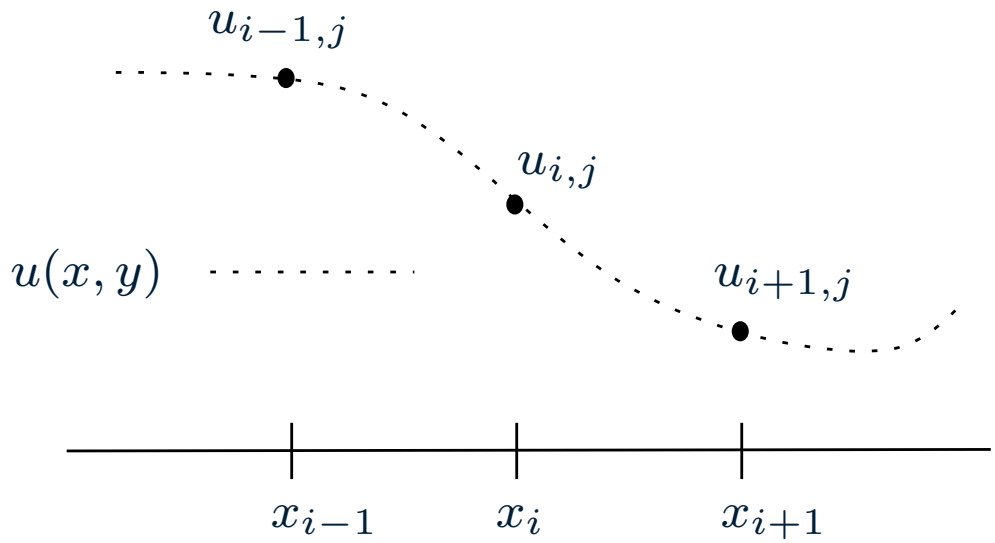


$$\partial_x u_{i,j} \approx \frac{u_{i,j} - u_{i-1,j}}{h_x}$$

$$\partial_x u_{i+1,j} \approx \frac{u_{i+1,j} - u_{i,j}}{h_x}$$

$$\partial_{xx} u_{i,j} \approx \frac{\partial_x u_{i+1,j} - \partial_x u_{i,j}}{h_x} = \frac{u_{i+1,j} - 2u_{i,j} + u_{i-1,j}}{h_x^2}$$

Discretization: Replace the derivatives by finite differences (via truncated Taylor series)



$$\partial_x u_{i,j} \approx \frac{u_{i,j} - u_{i-1,j}}{h_x}$$

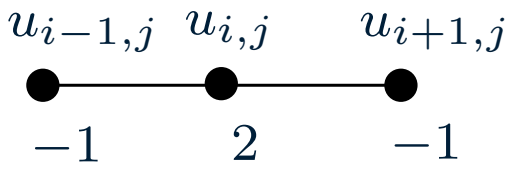
$$\partial_x u_{i+1,j} \approx \frac{u_{i+1,j} - u_{i,j}}{h_x}$$

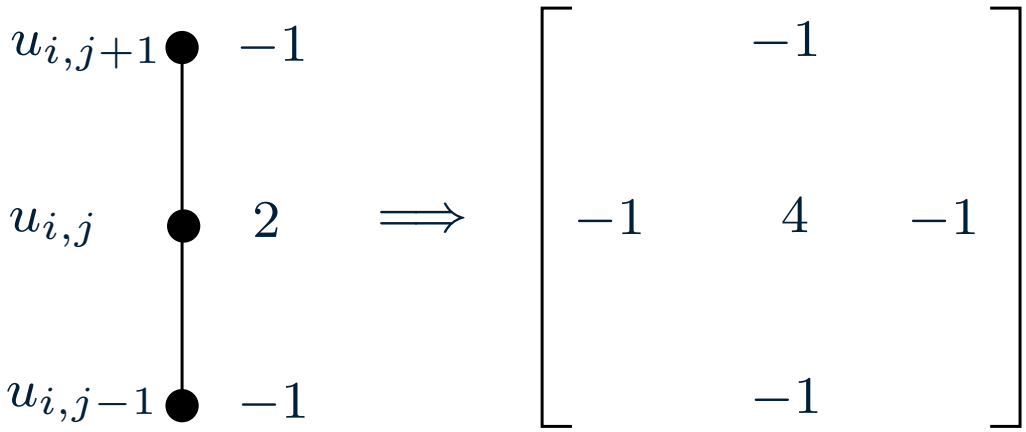
$$\partial_{xx} u_{i,j} \approx \frac{\partial_x u_{i+1,j} - \partial_x u_{i,j}}{h_x} = \frac{u_{i+1,j} - 2u_{i,j} + u_{i-1,j}}{h_x^2}$$

Our PDE now becomes a system of linear equations

$$-\left(\frac{u_{i+1,j}-2u_{i,j}+u_{i-1,j}}{h_x^2}+\frac{u_{i,j+1}-2u_{i,j}+u_{i,j-1}}{h_y^2}\right)=f(h_x i, h_y j)$$

$$i=0,1,2,...,n_x \qquad j=0,1,2,...,n_y$$





This is now a linear system of equations, $Ax = b$

- A is a matrix of size $N \times N$, $N = (n_x + 1)(n_y + 1)$
- x is an $N \times 1$ vector of unknowns (i.e., $u_{i,j}$ values)
- b is an $N \times 1$ known vector (i.e., $f(h_x i, h_y j)$ values)

We now just compute $x = A^{-1}b$, right?

A is generally large, sparse and poorly conditioned...

$$A = \frac{1}{h^2} \begin{pmatrix} 4 & -1 & \dots & -1 & & & & & \\ -1 & 4 & -1 & \dots & -1 & & & & \\ & -1 & 4 & -1 & \dots & -1 & & & \\ & & & & & & & & \\ & -1 & \dots & -1 & 4 & -1 & \dots & -1 & \\ & & & & & & & & \\ & & & -1 & \dots & -1 & 4 & -1 & \\ & & & & -1 & \dots & -1 & 4 & -1 \\ & & & & & -1 & \dots & -1 & 4 \end{pmatrix}$$

$$A = \frac{1}{h^2} \begin{pmatrix} 4 & -1 & \dots & -1 & & & & & \\ -1 & 4 & -1 & \dots & -1 & & & & \\ & -1 & 4 & -1 & \dots & -1 & & & \\ & & & & & & & & \\ & -1 & \dots & -1 & 4 & -1 & \dots & -1 & \\ & & & & & & & & \\ & & & -1 & \dots & -1 & 4 & -1 & \\ & & & & -1 & \dots & -1 & 4 & -1 \\ & & & & & -1 & \dots & -1 & 4 \end{pmatrix}$$

$$A = \frac{1}{h^2} \begin{pmatrix} 4 & -1 & \dots & -1 & & & & & \\ -1 & 4 & -1 & \dots & -1 & & & & \\ & -1 & 4 & -1 & \dots & -1 & & & \\ & & & & & & & & \\ & -1 & \dots & -1 & 4 & -1 & \dots & -1 & \\ & & & & & & & & \\ & & & -1 & \dots & -1 & 4 & -1 & \\ & & & & -1 & \dots & -1 & 4 & -1 \\ & & & & & -1 & \dots & -1 & 4 \end{pmatrix}$$

Properties of A

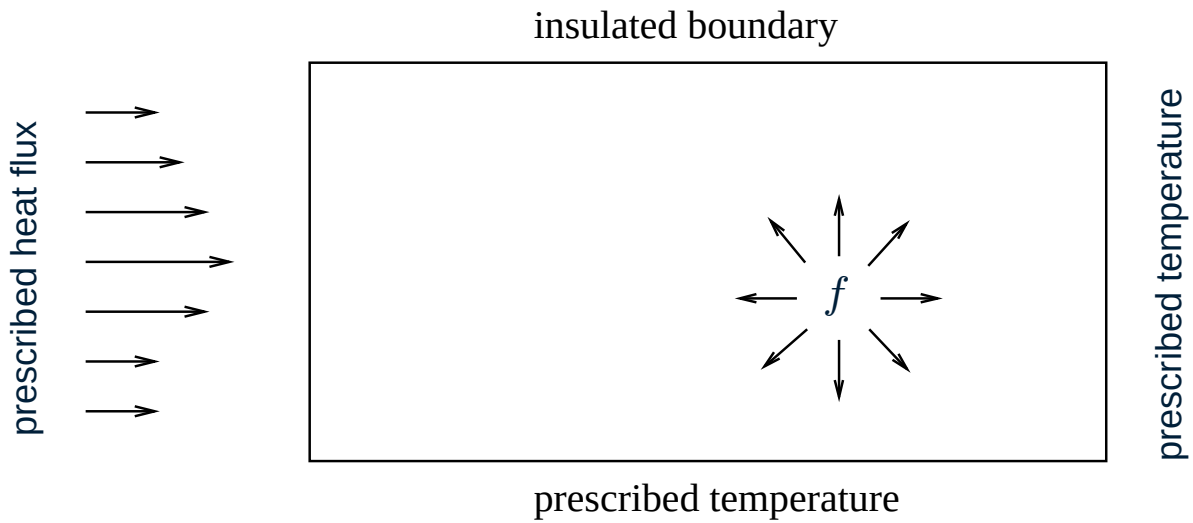
A is invertible, but has a large "near null space"...

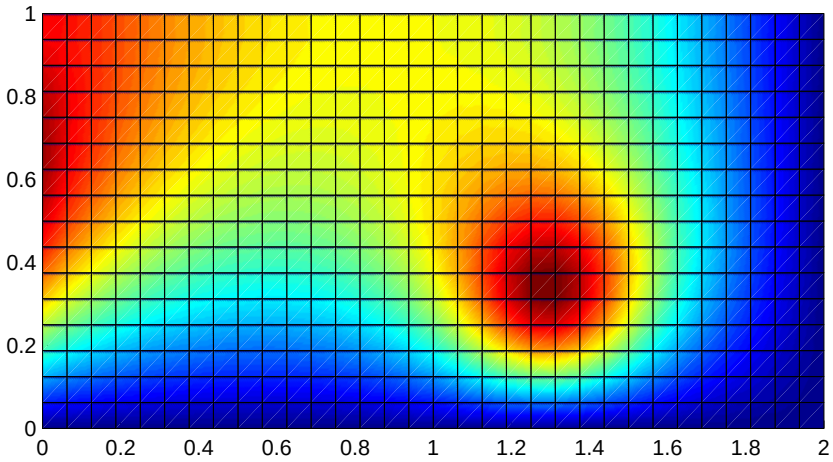
the vector

$$\boldsymbol{\xi} = \begin{pmatrix} 1 \\ 1 \\ 1 \\ : \\ 1 \\ 1 \\ 1 \end{pmatrix}$$

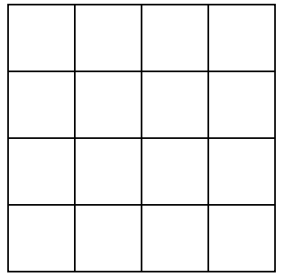
almost produces $A\boldsymbol{\xi} = \mathbf{0}$. These low energy modes are hard to resolve.

Computed Solution





Computational Challenges: 2-D



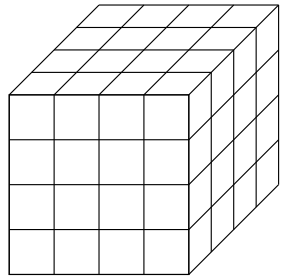
Example:

Steady state heat equation on unit square (2-D)

Examples

grid points per side	n	10	100	1000
unknowns (N)	n^2	100	10^4	10^6
Entries in Matrix	n^4	10000	10^8	10^{12}

Computational Challenges: 3-D



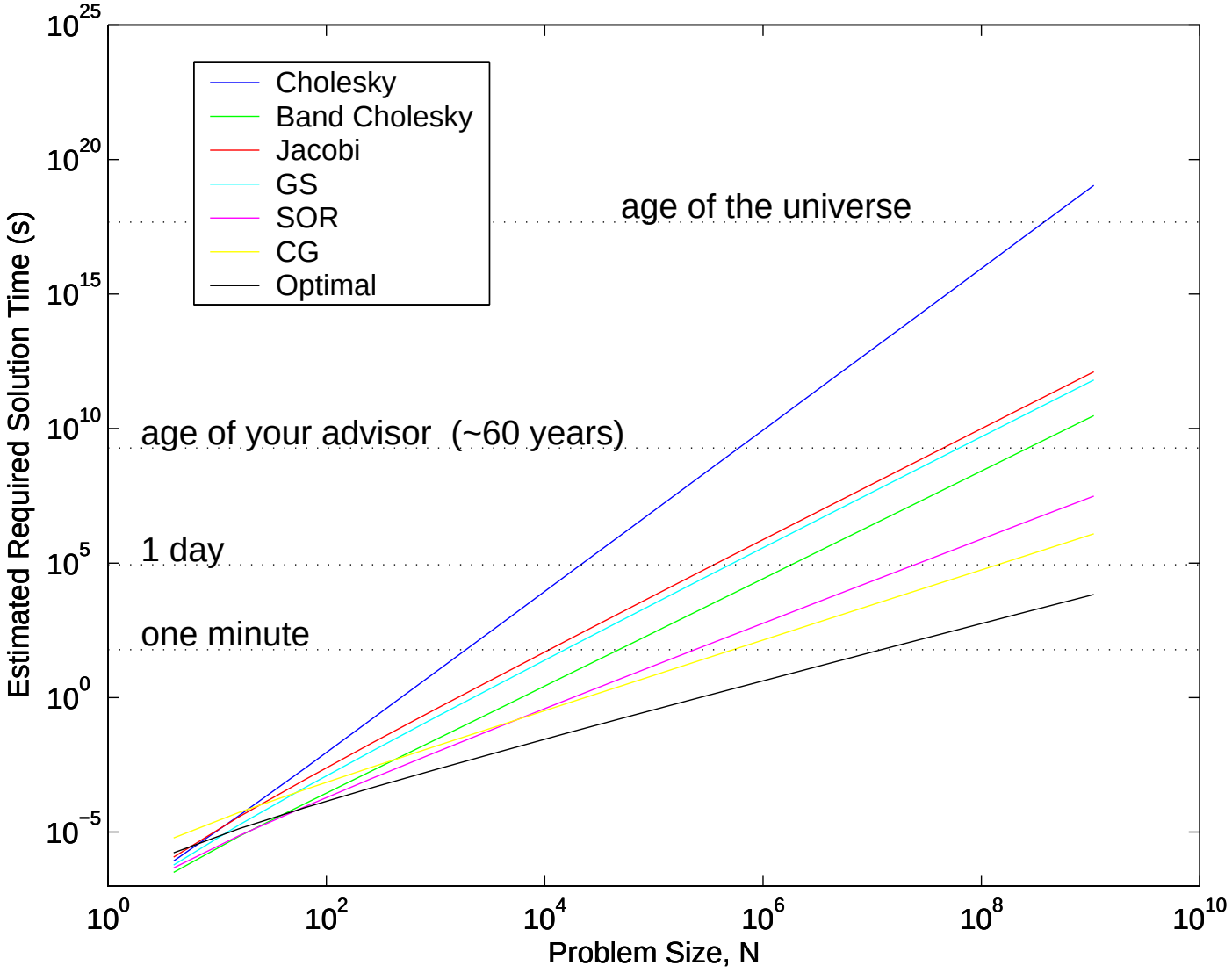
Example:

Steady state heat equation on unit cube (3-D)

Examples

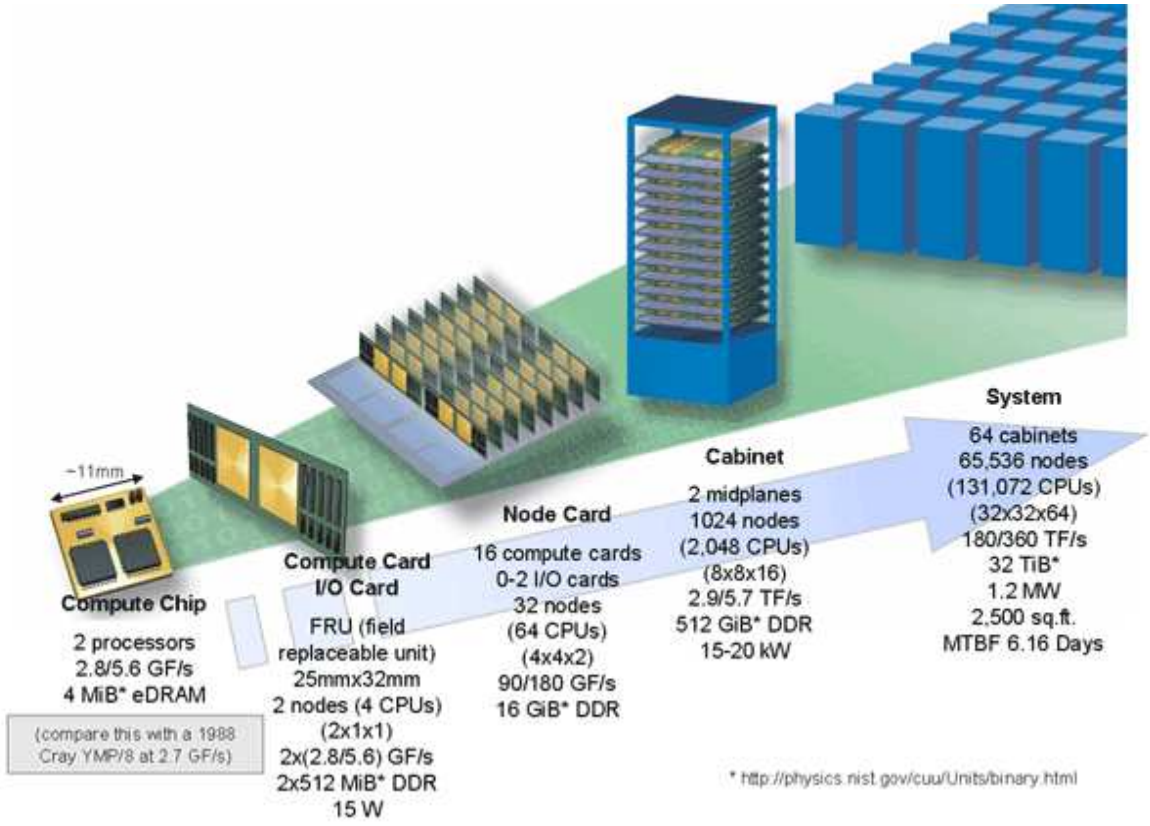
grid points per side	n	10	100	1000
unknowns (N)	n^3	1000	10^6	10^9
Entries in Matrix	n^6	10^6	10^{14}	10^{18}

Scalability of Linear Solvers



Parallel Computing

Divide and Conquor: break up the problem and send it to several different processors to compute at the same time.



Parallel Computing

Divide and Conquor: break up the problem and send it to several different processors to compute at the same time.

BlueGene/L at Lawrence Livermore National Lab recently exceeded 135 flops (trillion floating point operations per second)

Other Considerations

- Time dependent problems
- Better models often involve nonlinear phenomena
- Other discretization techniques (FE, FV, Spectral, etc.)
- Linear solvers should be scalable in the
  - number of unknowns
  - number of processors
- Linear solvers are often very problem dependent
- Visualization of data can be extremely complicated (3-D, 4-D, ...)
- Is the solution accurate, reasonable, useful?

Each step in the modeling process is a specialized process, but should consider the adjacent steps. Interdisciplinary teams often work together on this process. There is always more to be done!!