中国科学院大学
夏季强化课程
2022

Fast Solvers for Large Algebraic Systems

Lecture 4. Methods for nonlinear problems

4

非线性求解方法

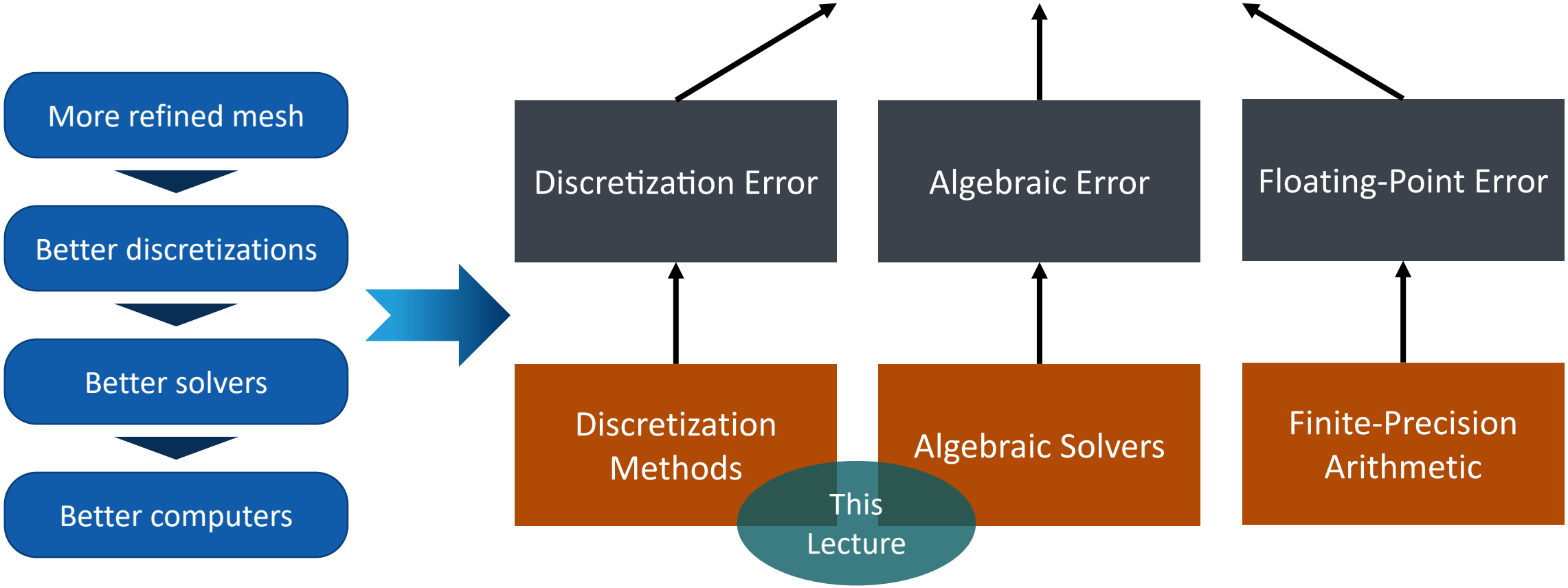
Chensong Zhang, AMSS

http://lsec.cc.ac.cn/~zhangcs

Table of Contents

Sources of Error in Simulation

Approximation: $u(x) = U_h(x) + \mathcal{E}_{\text{dis}} + \mathcal{E}_{\text{alg}} + \mathcal{E}_{\text{fp}}$



Introduction

Some examples of nonlinear differential equations

Origins of Nonlinearity

- Nonlinear partial differential equations:

  Membrane deflection, minimal surface, hyperelasticity, elastoplasticity, obstacle problem, contact problem, harmonic mapping, phase transition, plate bending, …

- Nonlinear ordinary differential equations (e.g. Chemical reactions)
- Nonlinearly coupled system of PDEs (e.g. Multiphysics problems)
- PDE-constrained optimization
- Optimal control problem
- Optimal design problem
- Geometric nonlinearity
- Modal analysis (eigenvalue problems)
- ……

Honey is a type of non-Newtonian fluid (nonlinear relation b/w shear stress and shear rate)

Example 1: Semilinear Elliptic Problem

- A model semilinear elliptic problem (SEP)

$f_u \le 0,\ f_u, f_{uu}$ are bounded

$$\begin{cases} -\Delta u = f(u), & \text{in } \Omega \\ u = 0, & \text{on } \partial\Omega \end{cases}$$

- Define bilinear forms

Linear SPD part

$$a(u,v) := \int_\Omega \nabla u \cdot \nabla v$$

$$\hat{a}(w;u,v) := a(u,v) - (f(w),v)$$

- Solve the weak problem

$$\text{Find } u \in H_0^1(\Omega):\ \hat{a}(u;u,v) = 0,\quad \forall v \in H_0^1(\Omega)$$

Two-Level Discretization for SEP

- Standard finite element formulation

$$\text{Find } u_h \in V_h : \ \hat{a}(u_h; u_h, v_h) = 0, \ \ \forall v_h \in V_h$$

- Classical two-level finite element method

Nonlinear problem

Step 1. $\text{Find } u_H \in V_H : \ \hat{a}(u_H; u_H, v_H) = 0, \ \ \forall v_H \in V_H$

Step 2. $\text{Find } u_{h,H} \in V_h : \ \hat{a}(u_H; u_{h,H}, v_h) = 0, \ \ \forall v_h \in V_h$

Only a linear problem

- Error estimate

$$\|u_h - u_{h,H}\|_1 \lesssim H^{r+1} \|u\|_{r+1}$$

J. Xu. Two-grid Discretization Techniques for Linear and Nonlinear PDEs. SIAM J. Numer. Anal. 33, 5, 1759–1777, 1996

Iterative Two-Level Methods

- Classical iterative two-level method

$$\text{Find } e_H^k \in V_H: \ a(e_H^k, v_H) = \big(f(u_h^k + e_H^k), v_H\big) - a(u_h^k, v_H), \quad \forall v_H \in V_H$$

$$\text{Find } u_{h,H}^{k+1} \in V_h: \ a(u_{h,H}^{k+1}, v_h) = \big(f(u_h^k + e_H^k), v_h\big), \quad \forall v_h \in V_h$$

- Modified iterative two-level method

$$\text{Find } e_H^k \in V_H: \ a(e_H^k, v_H) - \big(f_u(e_H^k), v_H\big) = \big(f(u_h^k), v_H\big) - a(u_h^k, v_H), \quad \forall v_H \in V_H$$

$$\text{Find } u_{h,H}^{k+1} \in V_h: \ a(u_{h,H}^{k+1}, v_h) = \big(f(u_h^k) + f_u(u_h^k)e_H^k, v_h\big), \quad \forall v_h \in V_h$$

Ref: 华南师范大学钟柳强教授课堂笔记

Example 2: Viscous Burgers Equation

- 1D nonlinear advection-diffusion equation

$$u_t + u\,u_x = \mu\,u_{xx}$$

Speed of fluid

Kinematic viscosity

Viscous Burgers Equation (VBE)

- Traveling wave solution

$$u(x,t) = w(x - x_0 - st) \quad \Rightarrow \quad u_t = -s\,w',\ \ u_x = w',\ \ u_{xx} = w''$$

VBE $\Rightarrow$ $-s\,w' + w\,w' = \mu\,w'' \Longrightarrow -s\,w + \frac{1}{2}w^2 = \mu\,w' + C$

- Enforce the boundary behavior

$$w(-\infty) = u_L,\ \ w(+\infty) = u_R,\ \ w'(\pm\infty) = 0$$

Traveling Wave Solution

- Applying the boundary conditions, we get

$$-su_L + \frac{1}{2}u_L^2 = C = -su_R + \frac{1}{2}u_R^2 \implies \boxed{s = \frac{u_L + u_R}{2}}$$

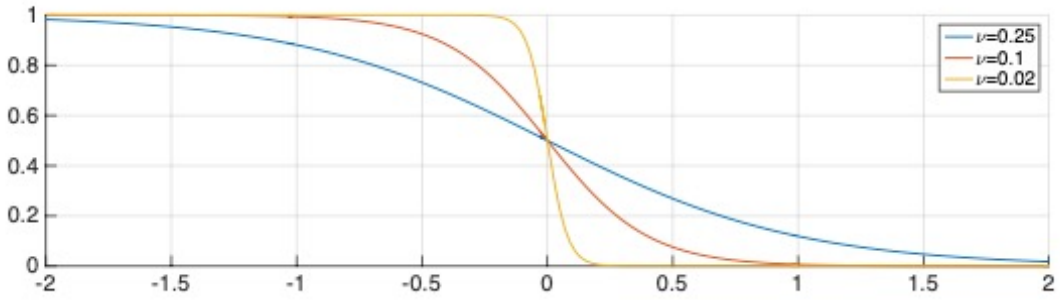
Wave propagation speed

- Obtain a special solution of VBE

$$u(x,t) = \frac{u_L + u_R}{2} - \frac{u_L - u_R}{2}\tanh\left(\frac{(x - x_0 - st)(u_L - u_R)}{4\mu}\right)$$

More difficult to solve if viscosity is small

Internal layers



Difficult to resolve

Source: Maria Cameron, https://www.math.umd.edu/~mariakc/burgers.pdf

Example 3: Burgers Equation

- Hyperbolic conservation law

$$u_t + \left[f(u)\right]_x = 0, \quad f(u) = \frac{1}{2}u^2 \quad \Longrightarrow \quad u_t + u\,u_x = 0$$

Burgers equation

- Boundary conditions

$$u(-\infty) = u_L, \quad u(+\infty) = u_R$$

- Traveling wave solution

$$u(x,t) = w(x - x_0 - st), \quad w(y) = \begin{cases} u_L, & y < 0 \\ u_R, & y > 0 \end{cases}$$

Riemann problem

- Similar to the previous example, speed of wave propagation

$$s = \frac{u_L + u_R}{2} \quad \Longleftarrow \quad s\left(u_L - u_R\right) = f(u_L) - f(u_R)$$

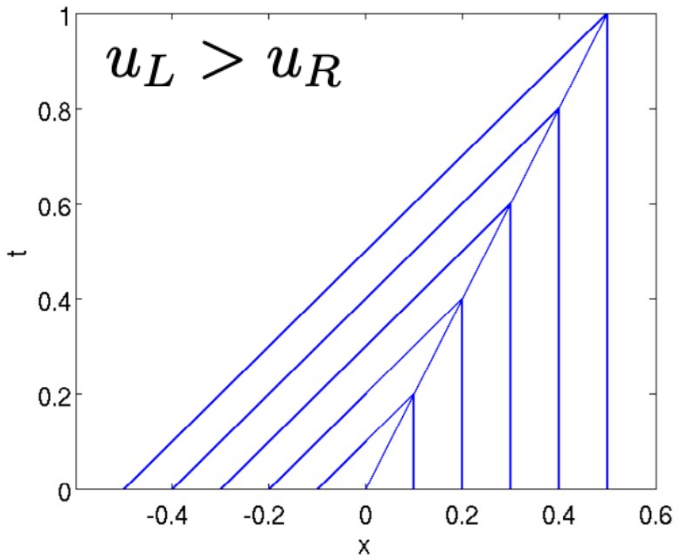
Rankine-Hugoniot Condition

Method of Characteristics

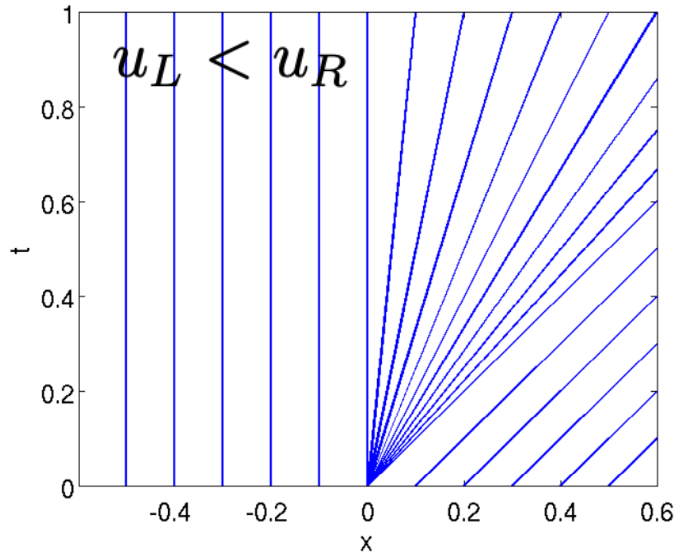
$$\dot{x} = u(x,t) \quad \Longrightarrow \quad \frac{d}{dt}u(x(t),t) = \frac{\partial u}{\partial t} + \frac{\partial u}{\partial x}\frac{\partial x}{\partial t} = u_t + u\,u_x = 0$$

The solution is a constant along the characteristics!

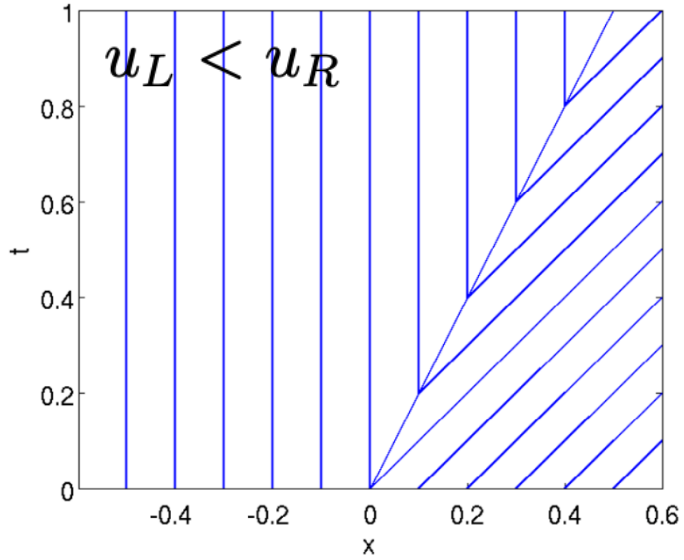
Shock wave solution



Rarefaction wave solution



Rarefaction shock solution



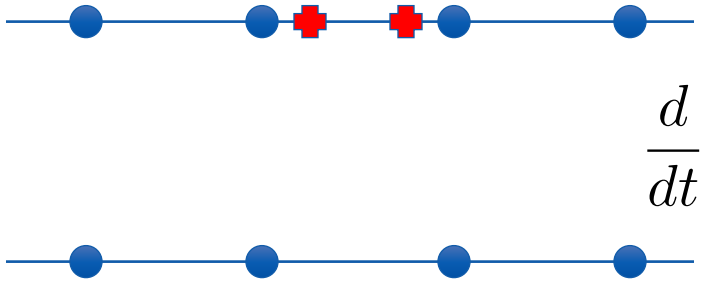
- Which one is physical? Check the viscous limit or the entropy conditions (simpler to verify).

ELLAM Method, Revisited

1D Advection-Diffusion Model Problem:

$$u_t + b \cdot \nabla u - \mu \Delta u = 0$$

Replacing the linear convection by u



$$\frac{d}{dt} y(x, s; t) = b(y(x, s; t), t), \quad y(x, s; s) = x.$$

$$\left(\frac{u^{k+1} - u_*^k}{\Delta t}, v \right) + \mu \left(\nabla u^{k+1}, \nabla v \right) = (f, v), \quad \forall v \in V$$

Nonlinear PDE ➔ Linear discrete problems

➔ Find $u_h^{k+1} \in V_h$, such that $\left(\frac{u_h^{k+1} - u_{h,*}^k}{\Delta t}, v_h \right) + \mu \left(\nabla u_h^{k+1}, \nabla v_h \right) = (f, v_h), \quad \forall v_h \in V_h$

➔ $\left(\frac{1}{\Delta t} M + \mu A \right) x = b$ Non-symmetric/nonlinear PDE ➔ SPD algebraic systems

A Naïve FD Discretization

- A simple Riemann problem

$$u_t + u\,u_x = 0 \qquad u(x,0) = u_0(x) = \begin{cases} 1, & x < 0 \\ 0, & x \geq 0 \end{cases}$$

- A first-order upwind scheme

$$U_j^{k+1} = U_j^k - \frac{\Delta t}{h} U_j^k \left(U_j^k - U_{j-1}^k \right) \qquad j \in \mathbb{Z}$$

$$U_j^k = 0, \quad j \geq 0 \quad \text{and} \quad U_j^k - U_{j-1}^k = 1 - 1 = 0, \quad j < 0$$

$$\Longrightarrow \quad U_j^{k+1} \equiv U_j^k, \quad \forall j$$

- Does not propagate at the wave propagation speed = ½ !
- Need numerical methods that are both conservative and physical!

Godunov's Method

- Step 1. Define piecewise constant initial guess

A given numerical approximation

$$\bar{u}(x,t_k) := U_j^k, \quad x_j - \frac{h}{2} < x < x_j + \frac{h}{2}$$

- Step 2. Find numerical solution of the Riemann problems $\bar{u}(x,t), \quad t \in (t_k, t_{k+1}]$

$$\frac{1}{h}\int_{x_j-\frac{h}{2}}^{x_j+\frac{h}{2}} \bar{u}(x,t_{k+1})\,dx = \frac{1}{h}\int_{x_j-\frac{h}{2}}^{x_j+\frac{h}{2}} \bar{u}(x,t_k)\,dx - \frac{1}{h}\left(\int_{t_k}^{t_{k+1}} f\left(\bar{u}(x_j+\frac{h}{2},t)\right)dt - \int_{t_k}^{t_{k+1}} f\left(\bar{u}(x_j-\frac{h}{2},t)\right)dt\right)$$

- Step 3. Find numerical solution of the Riemann problems [Godunov 1959]

$$U_j^{k+1} = \frac{1}{h}\int_{x_j-h/2}^{x_j+h/2} \bar{u}(x,t_{k+1})\,dx \Rightarrow U_j^{k+1} = U_j^k - \frac{\Delta t}{h}\left(F(U_j^k, U_{j+1}^k) - F(U_{j-1}^k, U_j^k)\right)$$

$$F(u_L,u_R) := \begin{cases} \min_{u_L \le u \le u_R} f(u), & u_L \le u_R \\ \max_{u_L \le u \le u_R} f(u), & u_L > u_R \end{cases}$$

Need a good numerical flux!

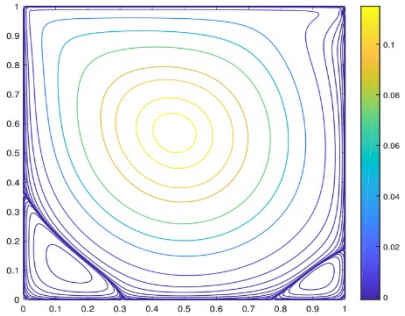
Example 4: NS Equations

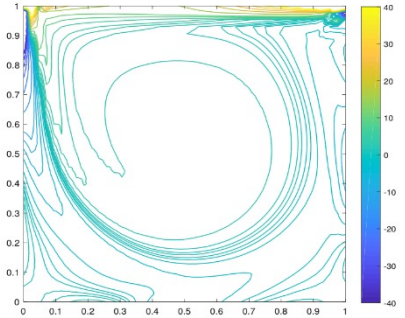
- Incompressible Navier-Stokes equations

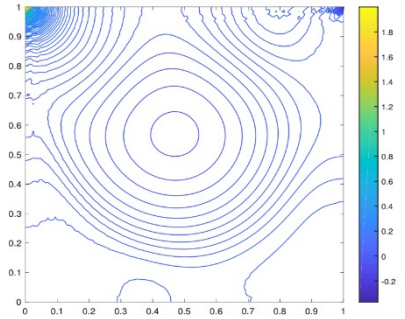
$$u_t + (u \cdot \nabla)u - \mu \Delta u + \nabla p = 0$$

$$\nabla \cdot u = 0, \quad u(x,0) = u_0(x), \quad u|_{\partial\Omega} = g$$

- Solving the NS equations
  - ✓ Method of characteristics
  - ✓ Chorin's projection method
  - ✓ Two-level discretization
  - ✓ Linearize and then discretize
  - ✓ Discretize and solve nonlinear equations







Source: Zeng, Huilan, Chen-Song Zhang, and Shuo Zhang. "A low-degree strictly conservative finite element method for incompressible flows." arXiv preprint arXiv:2103.00705 (2021).

Example 5: Reservoir Equations

$$\frac{\partial}{\partial t}\left(\phi\sum_{j=1}^{n_p} x_{ij}\xi_j S_j\right)+\nabla\cdot\mathbf{F}_i-\sum_{j=1}^{n_p} S_j r_{ij}=Q_i,\qquad i=1:n_c$$ 物质守恒

$$\mathbf{F}_i=\sum_{j=1}^{n_p}\left(x_{ij}\xi_j\mathbf{u}_j-S_j\mathbf{D}_j\nabla(\xi_j x_{ij})\right),\qquad i=1:n_c$$ 流量方程

$$\mathbf{u}_j=-\frac{\kappa\kappa_{rj}}{\mu_j}(\nabla P_j-\gamma_j\nabla z),\qquad j=1:n_p$$ Darcy定律

$$\sum_{j=1}^{n_p} S_j=1,$$ 饱和度关系

$$\sum_{i=1}^{n_c} x_{ij}=1,\qquad j=1:n_p$$ 组分比例关系

$$P_1-P_j=P_{c1j},\qquad j=2:n_p$$ 表面张力

$$f_{ij}=f_{i1},\qquad i=1:n_c,\ j=2:n_p$$ 相平衡方程

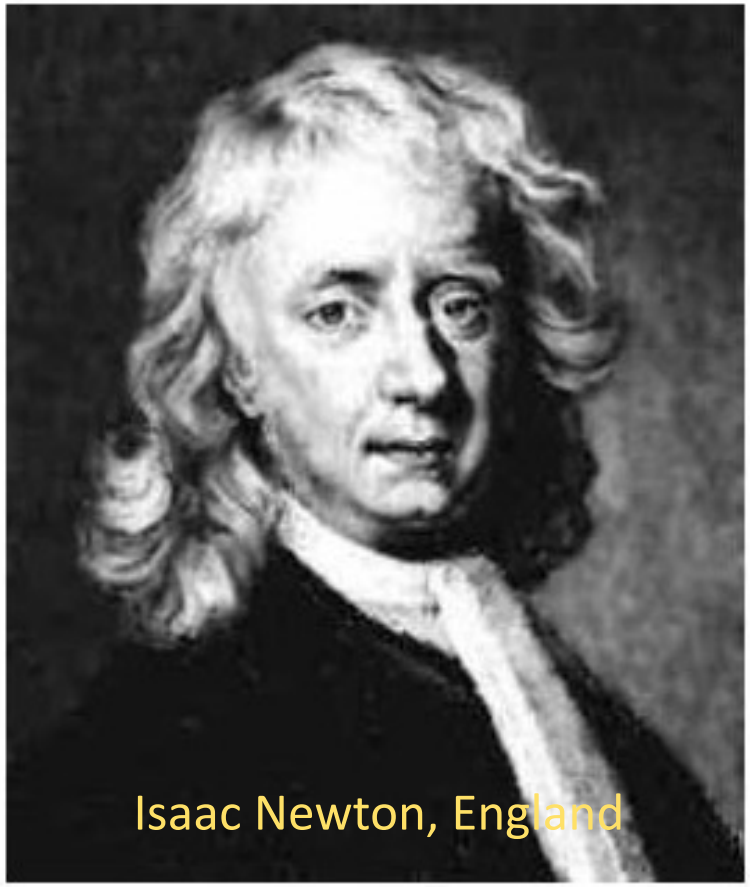
- 多重介质模型
- EOS状态方程
- Brinkman方程
- 化学反应方程
- Navier-Stokes方程
- 非等温模型
- 非Darcy模型
- 吸附解吸模型
- 岩石溶解模型
- 岩石力学方程

Nonlinear Equation Solvers

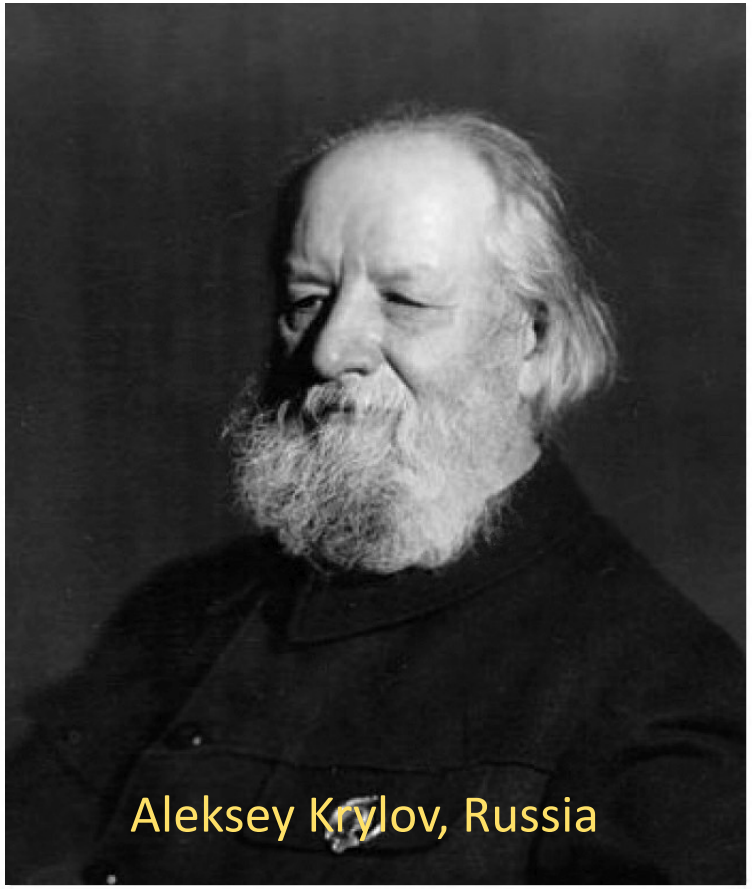
Classical nonlinear solvers that are still widely used

/02

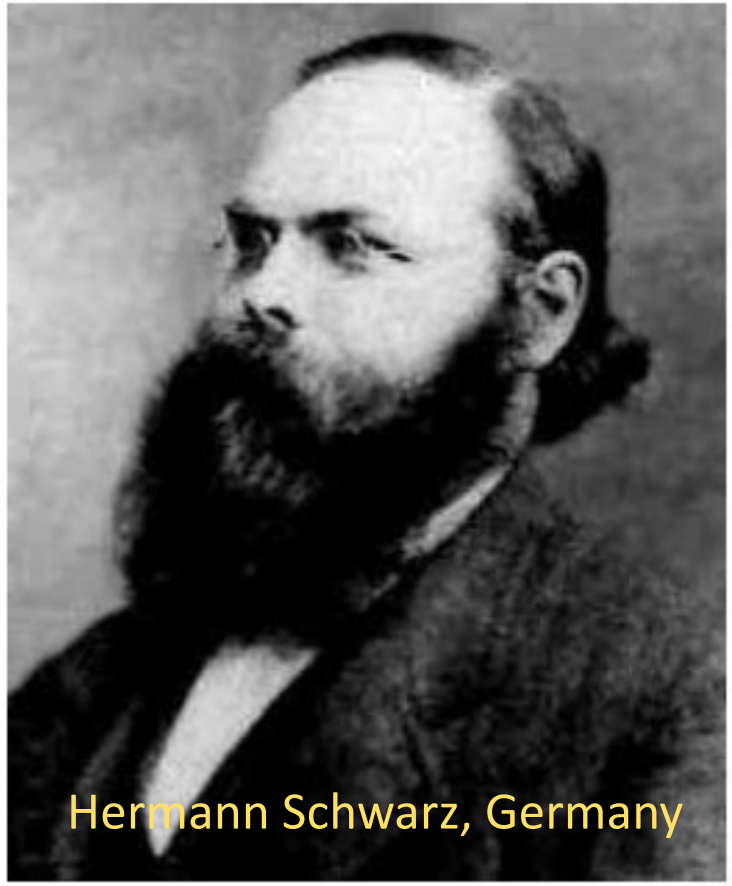
Newton-Krylov-Schwarz Methods



Newton's method for nonlinear equations



Krylov subspace iterative methods



Schwarz methods for preconditioning & parallelization

Another Choice to Make: DO or OD

$$\min_{u \in \mathcal{V}} f(u) \quad \star \quad \nabla f(u) = 0 \quad \star \quad 0 \in \nabla f(u)$$

Discretization-then-optimization (DO)

Optimization-then-discretization (OD)

Continuation Method

- Solve the nonlinear equation: $F(u) = 0$
- Construct an iteration $u(0) := u^k \implies u^{k+1}$?

How to construct such an auxiliary problem? Seems quite random …

$$\frac{du(t)}{dt} = -F(u(t))$$

➡ Steady-state solution ➔ Original solution

➔ Fixed-point method ➔ Richardson ➔ Steepest descent

$$\frac{dF(u(t))}{dt} = -F(u(t))$$

➡ $$F'(u^k)\left(u^{k+1} - u^k\right) = -\tau_k F(u^k)$$

➡ $$u^{k+1} = u^k - \tau_k\, F'(u^k)^{-1} F(u^k)$$

If $F(u)$ is differentiable in some sense

$= 1$: Newton's method

< 1: Damped Newton's method

Steepest Descent Method, Revisited

- The steepest descent method for solving linear systems (review Lecture 2)

$$\min_{x\in V} f(x) := \frac{1}{2}(Ax,x) - (b,x) \quad \Rightarrow \quad x^{\text{new}} = x^{\text{old}} + \alpha\, d$$

Optimal stepsize

Steepest descent

$$\alpha_{\text{opt}} = \frac{(b - Ax^{\text{old}}, d)}{(Ad,d)} = \frac{(r^{\text{old}}, d)}{(Ad,d)}, \quad \text{with } r^{\text{old}} := b - Ax^{\text{old}} \qquad d_{\text{steep}} := -\nabla f(x^{\text{old}}) = r^{\text{old}}$$

- Main ingredients: Finding a search direction and a stepsize
  - Step 1. Construct a search direction in a cheap way
  - Step 2. Go along the direction and update

⇒ Generalization?

Descent Direction Methods

- Suppose there is a descent direction (like the negative gradient direction)

$$x^{(k+1)} = x^{(k)} + \alpha^{(k)} d^{(k)}$$

- How far we should go along this direction? ➔ Stepsize

$$\alpha^{(k)} = \arg\min_{\alpha \in \mathbb{R}} f(x^{(k)} + \alpha d^{(k)})$$

- Backtracking linear search methods

Armijo line search (back tracking):

1. Set $c_1 \in (0,1)$, $\tau \in (0,1)$, $\alpha_0 > 0$, and $j := 0$;
2. If $f(x^{(k)}) - f(x^{(k)} + \alpha_j d^{(k)}) \geq -\alpha_j c_1 \nabla f(x^{(k)})^T d^{(k)}$ is satisfied, then return the stepsize $\alpha^{(k)} := \alpha_j$;
3. Otherwise, set $j := j + 1$, $\alpha_j := \tau \alpha_{j-1}$, and goto Step 2.

Termination of iteration

Updating Search Direction

- Usually the initial search direction is easy to obtain (residual)
- How to update the direction and continue? (Like in the Krylov subspace methods)

$$d^{(k)} := -\nabla f(x^{(k)}) + \beta^{(k)} d^{(k-1)}$$

  - Rules for updating $\beta^{(k)}$:

    Fletcher–Reeves: $\beta^{(k)} = \nabla f(x^{(k)})^T \nabla f(x^{(k)}) / \nabla f(x^{(k-1)})^T \nabla f(x^{(k-1)})$

    Polak–Ribière: $\beta^{(k)} = \nabla f(x^{(k)})^T y^{(k)} / \nabla f(x^{(k-1)})^T \nabla f(x^{(k-1)})$

    Hestenes–Stiefel: $\beta^{(k)} = \nabla f(x^{(k)})^T y^{(k)} / d^{(k-1)^T} y^{(k)}$

    Dai–Yuan: $\beta^{(k)} = \nabla f(x^{(k)})^T \nabla f(x^{(k)}) / d^{(k-1)^T} y^{(k)}$

$$y^{(k)} := \nabla f(x^{(k)}) - \nabla f(x^{(k-1)})$$

Newton's Method

$$d^{(k)} := -\nabla^2 f(x^{(k)})^{-1} \nabla f(x^{(k)})$$

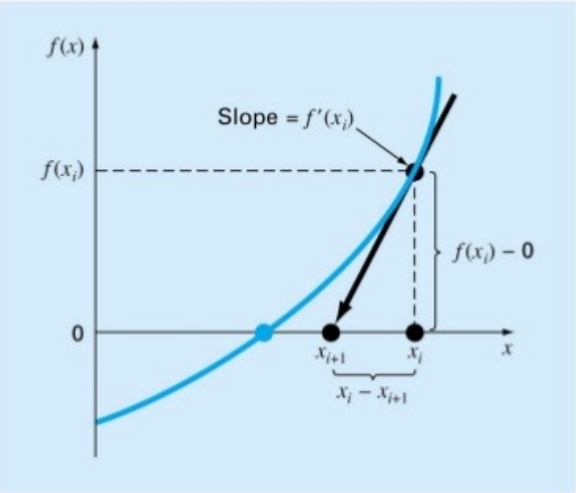
$$\arg\min_{d \in \mathbb{R}^n} f(x^{(k)}) + \nabla f(x^{(k)})^T (x^{(k)} + d) + \tfrac{1}{2}(x^{(k)} + d)^T \nabla^2 f(x^{(k)})(x^{(k)} + d)$$

- $\alpha^{(k)} = 1$ or exact line search $\Longrightarrow$ local 2nd-order convergence

Newton's Method: Line Equation

The slope of the line is given by:

$$m = \frac{y_1 - y_2}{x_1 - x_2} = f'(x_1)$$



Roots of Non-Linear Equations
Mohammad Tawfik

#WikiCourses
http://WikiCourses.WikiSpaces.com

- 算法成熟，应用广泛，工程应用中的首选
- 如果收敛，收敛速度快
- 解时间发展方程时，可与时间自适应结合
- 局部收敛，对初值要求高（不稳健）
- 需要计算目标函数的二阶导数（代价高）
- Hessian矩阵的存储和计算代价较高
- 代数方程组的求解较困难

Quasi-Newton Methods

Find an approximated Hessian, easier to compute and cheaper to store but do not harm convergence

BFGS (Broyden-Fletcher-Goldfarb-Shanno)

$$B_{\text{BFGS}}^{(k+1)} := B^{(k)} + \frac{y^{(k)^T} y^{(k)}}{y^{(k)^T} s^{(k)}} - \frac{B^{(k)} s^{(k)} s^{(k)^T} B^{(k)}}{s^{(k)^T} B^{(k)} s^{(k)}}.$$

DFP (Davidon-Fletcher-Powell)

$$B_{\text{DFP}}^{(k+1)} := \left(I - \frac{s^{(k)} y^{(k)^T}}{s^{(k)^T} y^{(k)}}\right) B^{(k)} \left(I - \frac{s^{(k)} y^{(k)^T}}{s^{(k)^T} y^{(k)}}\right) + \frac{y^{(k)} y^{(k)^T}}{s^{(k)^T} y^{(k)}}$$

The Broyden family

$$B^{(k+1)} := (1 - \phi^{(k)}) B_{\text{BFGS}}^{(k+1)} + \phi^{(k)} B_{\text{DFP}}^{(k+1)}, \qquad \phi^{(k)} \in [0, 1]$$

For $\phi^{(k)} \in [0, 1)$, it has the same convergence property as BFGS. DFP is still open.

Sometimes we need to save memory for large-scale problems and limited-memory versions of quasi-Newton methods

Trust Region

$$f(x_k + d) = f_k + g_k^T d + \frac{1}{2} d^T \nabla^2 f(x_k + \xi d) d$$

$$\approx f_k + g_k^T d + \frac{1}{2} d^T \nabla^2 f(x_k) d$$

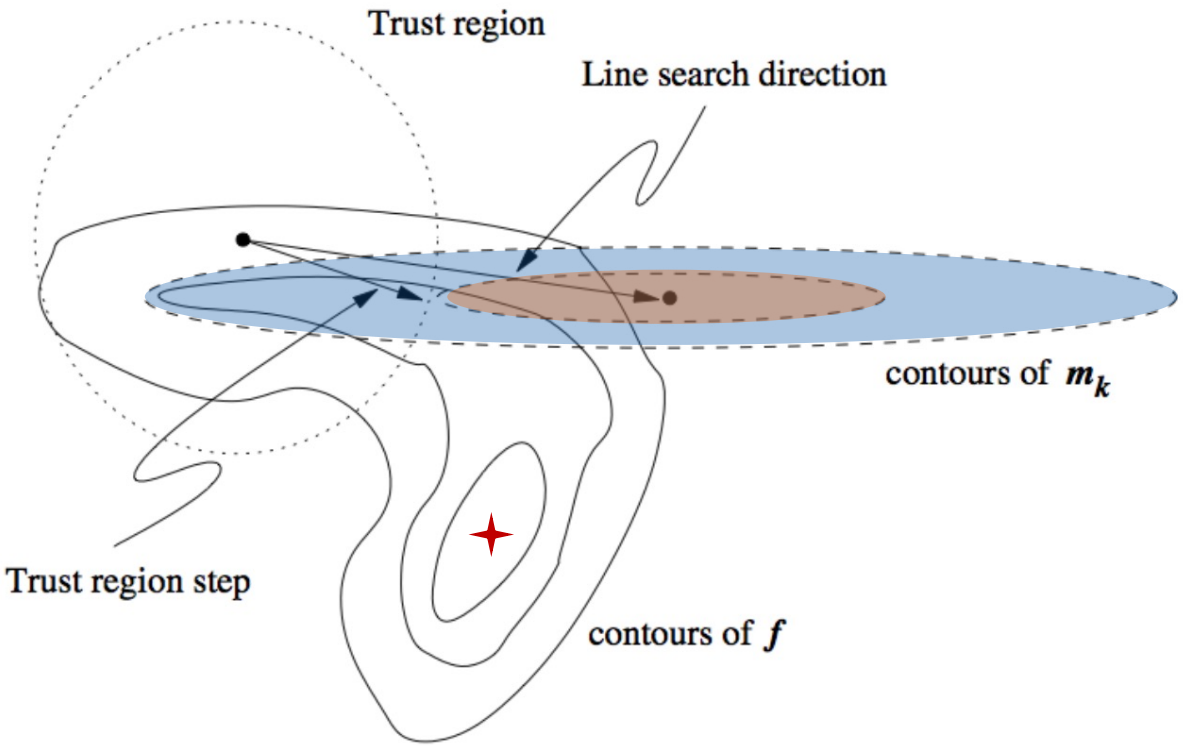
Add a trust region

$$\min_{\|d_k\| \le \delta_k} m_k(d_k)$$

Reduction prediction

$$\rho_k := \frac{f(x_k) - f(x_k + d_k)}{m_k(0) - m_k(d_k)}$$

$$m_k(d) := f_k + g_k^T d + \frac{1}{2} d^T B_k d$$



Trust Region Method

Algorithm 8: Trust region method

```
1  %% Given δ_max, δ ∈ (0, δ_max), η ∈ [0, 1/4), and an initial guess x;
2  for k = 0 : MaxIter or converged
3      Solve the trust region subproblem (approximately) to obtain d;
4      Compute reduction ratio ρ;
5      if ρ < 1/4 %% prediction is bad
6          δ ← (1/4)δ;
7      else if ρ > 3/4 && ||d|| == δ %% prediction is good
8          δ ← min(2δ, δ_max);
9      end
10     if ρ > η, x ← x + d; %% accept the not-so-good solution
11 end
```

- Line 3: The trust region subproblem is still not easy to solve numerically
- How to solve TR subproblem? Moré-Sorensen, dogleg, inexact solvers, …

Solving TR Subproblem

- In TR algorithm, Line 3 is the key step (TR subproblem—A constrained optimization problem)
- Find the Cauchy point (scaled steepest descent) in the trust region (TR linear model) 1D problem

$$\min_{\tau} m_k(\tau d_{\text{steep}}), \quad \text{s.t. } \|\tau d_{\text{steep}}\| \le \delta \quad \Longrightarrow \quad d_{\text{Cauchy}} := \alpha\, d_{\text{steep}}$$

- Construct the dogleg path

$$d_{\text{dogleg}}(\tau) := \begin{cases} \tau d_{\text{Cauchy}}, & 0 \le \tau \le 1 \\ d_{\text{Cauchy}} + (\tau - 1)(d_{\text{Newton}} - d_{\text{Cauchy}}), & 1 < \tau \le 2 \end{cases}$$

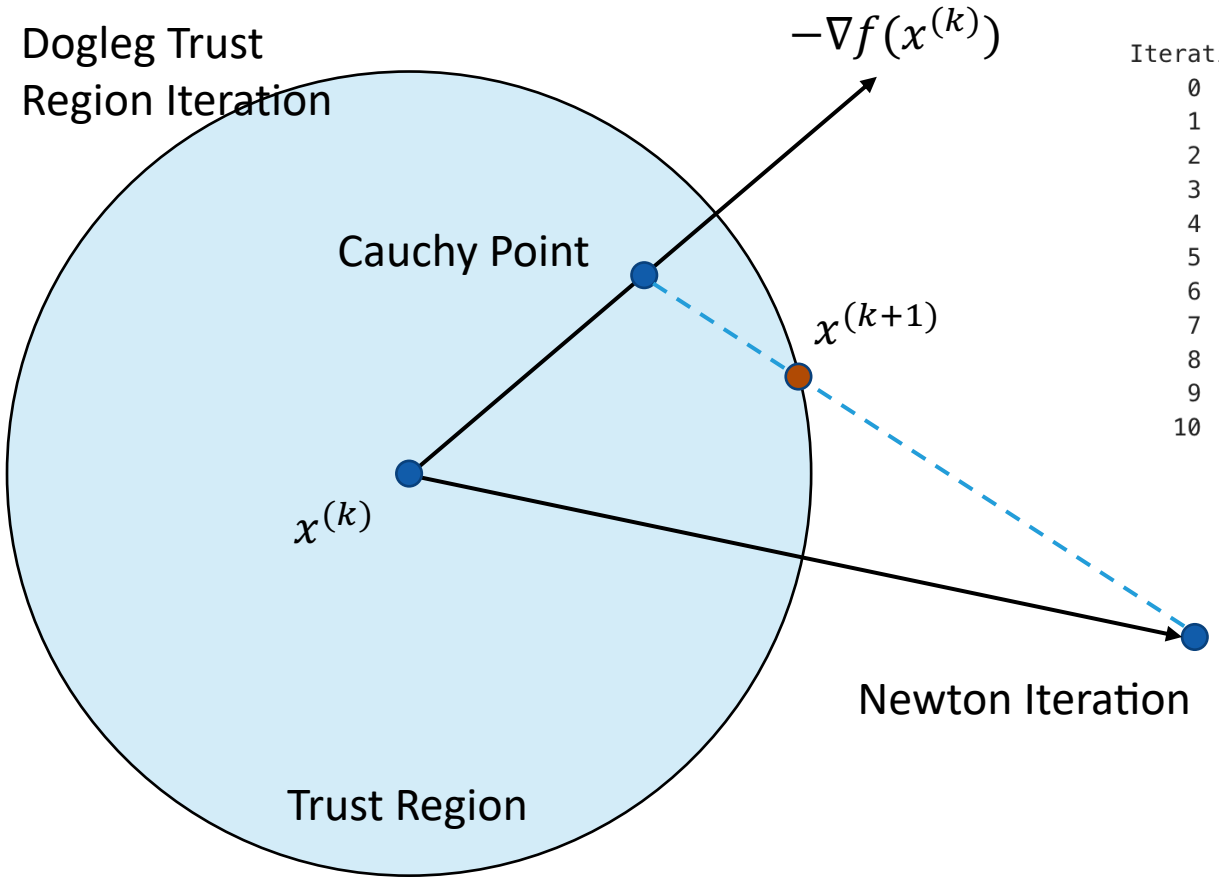
- Compute the stepsize such that

$$\min_{\|d_{\text{dogleg}}(\tau)\| \le \delta} m_k\left(d_{\text{dogleg}}(\tau)\right)$$

Also easy to solve

Ref: 袁亚湘、孙文瑜《最优化理论与方法》，科学出版社，1997

Dogleg Trust Region Method



Iteration	Func-count	f(x)	Norm of step	First-order optimality	Trust-region radius
0	3	47071.2		2.29e+04	1
1	6	12003.4	1	5.75e+03	1
2	9	3147.02	1	1.47e+03	1
3	12	854.452	1	388	1
4	15	239.527	1	107	1
5	18	67.0412	1	30.8	1
6	21	16.7042	1	9.05	1
7	24	2.42788	1	2.26	1
8	27	0.032658	0.759511	0.206	2.5
9	30	7.03149e-06	0.111927	0.00294	2.5
10	33	3.29525e-13	0.00169132	6.36e-07	2.5

Matlab fsolve function uses a variant of trust region (FD Newton) method with dogleg subsolvers for solving nonlinear equations, at each iteration mldivide is used

Ref: Powell, M. J. D., “A hybrid method for nonlinear equations”, Numerical Methods for Nonlinear Algebraic Equations, P. Rabinowitz, et al. edit, 1970.

Summary of Optimization Methods

算法名称	优点	缺点
最速下降法	算法简单，只需要使用一阶导数；全局收敛	收敛速度慢
牛顿法	如果收敛，收敛速度快	需要二阶导数，工作量大；局部二次收敛，可能不收敛
非精确牛顿法	当获得精确牛顿方向非常困难或者代价过高时，可以适用	类似于牛顿法
拟牛顿法	如果结合线搜索，全局收敛；对强凸问题，收敛快	工作量较大，收敛速度较牛顿法慢
非线性CG方法	如果结合线搜索，全局收敛；线性收敛	收敛速度可能比牛顿法慢
信赖域方法	增大牛顿法收敛域	收敛速度可能比牛顿法慢

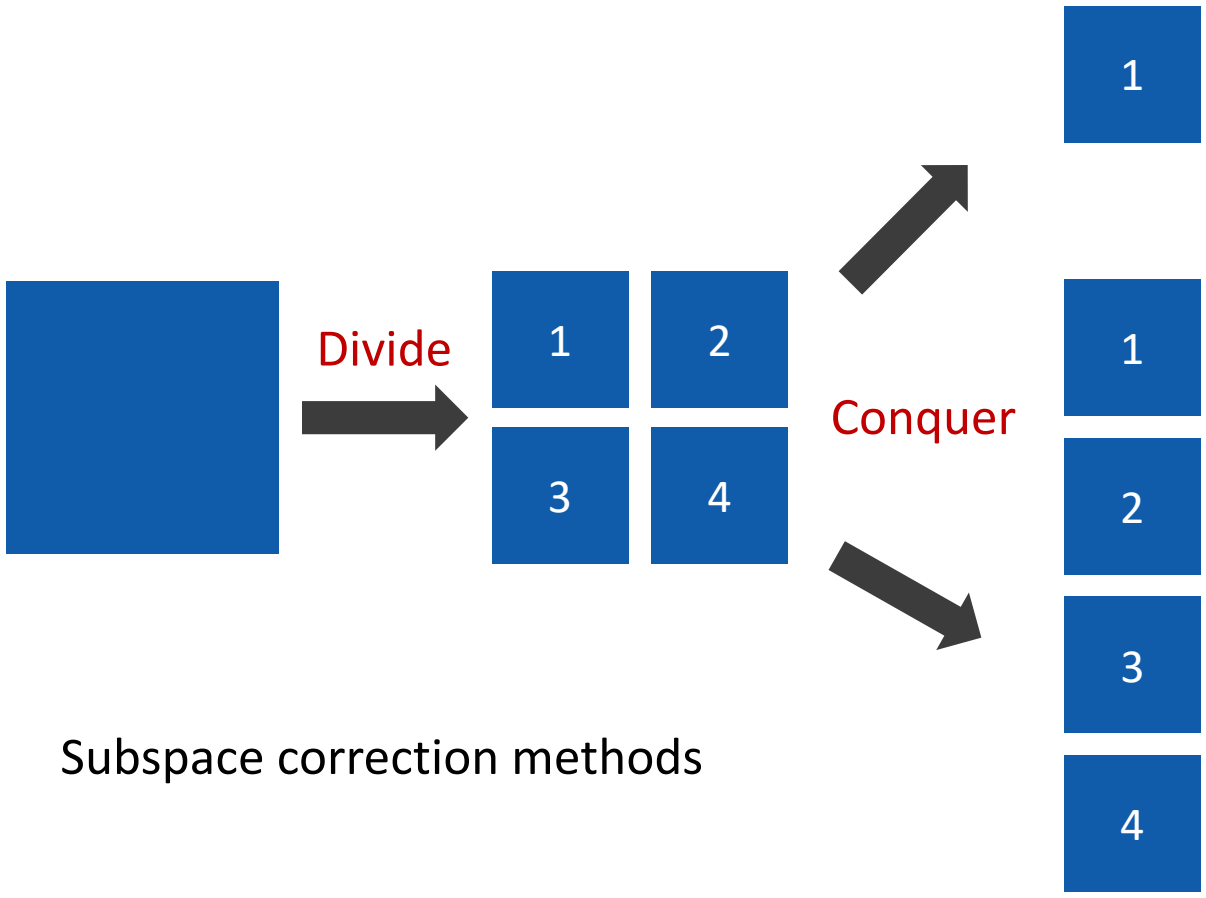
Nonlinear Schwarz Methods

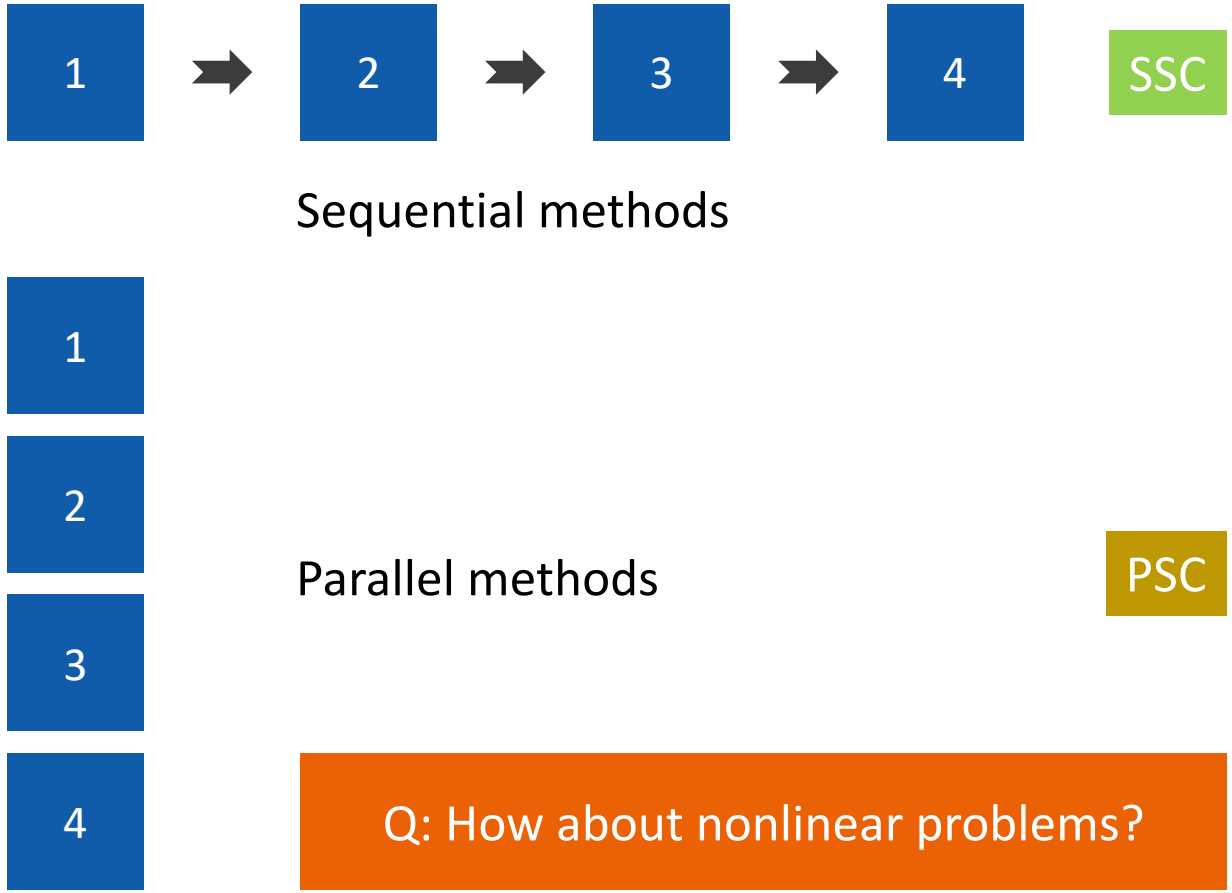
Schwarz methods for nonlinear problems

/03

Schwarz Methods, Revisited

Typical examples of Schwarz methods: Jacobi, GS, DDM , MG, BPX, …





Newton-Multigrid Method

- Consider a general set of nonlinear equations

$$F(x) := \begin{bmatrix} F_1(x_1, \ldots, x_n) \\ F_2(x_1, \ldots, x_n) \\ \vdots \\ F_n(x_1, \ldots, x_n) \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix} \Longrightarrow \nabla F(x) := \left[\frac{\partial F_i}{\partial x_j} \right]_{i,j=1}^{n}$$

- Apply a Taylor expansion and drop the high order terms (HOT)

$$F(x+d) = F(x) + \nabla F(x)\, d + \text{H.O.T.} \Longrightarrow \nabla F(x)\, d \approx F(x+d) - F(x)$$

$$\Longrightarrow \text{Newton's method} \quad x' = x - \nabla F(x)^{-1} F(x) \Longleftarrow \text{Solve it using the multigrid method}$$

- DO/OD ➔ Linear algebraic system ➔ Multigrid for linear system ➔ Is there nonlinear MG?

Twogrid Method, Revisited

- Step 1. Presmoothing

$$x \leftarrow x + M^{-1}(b - Ax)$$ → Nonlinear relaxation

- Step 2. Coarse grid correction (CGC)

$$x \leftarrow x + PA_c^{-1}P^T(b - Ax)$$ $\boxed{A_c e = r}$ Key: How to do CGC?

- Step 3. Postsmoothing (optional)

$$x \leftarrow x + M^{-T}(b - Ax)$$ → Drop it!

- Q: How to form a coarse problem? How to solve it? How to correct the fine solution?
- There is a general framework, called full approximation scheme (FAS)

Ref: Brandt, Achi. "Multi-level adaptive solutions to boundary-value problems." Mathematics of computation 31.138, 333-390, 1977.

Full Approximation Scheme

$$A_H e_H = A_H(x_H + e_H) - A_H(x_H) = r_H \quad \Rightarrow \quad x_h = x_h + P e_H$$

Linear Case

Nonlinear Case

$$F_h(x_h + e_h) = 0 \qquad x_h = x_h + P(y_H - x_H)$$

$$F_H(y_H) = F_H(R x_h) - R F_h(x_h)$$

τ-correction equation

$$F_h(x_h + e_h) - F_h(x_h) = -F_h(x_h) =: r_h$$

Solve $F_H(y_H) = r_H + F_H(x_H)$

$$x_H := R x_h \qquad y_H := x_H + e_H$$

$$F_H(x_H + e_H) - F_H(x_H) = R r_h \quad \Rightarrow \quad r_H := -R F_h(x_h)$$

Nonlinear Relaxation

So we have a multigrid framework. Q: How can we solve / approximate the problem on each level?

Divide-and-Conquer

Solve one-by-one

Nonlinear Gauss-Seidel

$$F_i(x + \alpha e_i) = 0, \quad i = 1, \ldots, n$$

Schwarz method

Solve all at once

Nonlinear Jacobi

How to solve the subspace problems?

- Scalar nonlinear equations
- Can be solved using Newton, Quasi-Newton, TR, ...

Preliminary Numerical Results

Method	No. outer iterations	No. inner iterations	Megaflops
Newton	3	–	1660.6
Newton-MG	3	20	56.4
Newton-MG	4	10	38.5
Newton-MG	5	5	25.1
Newton-MG	10	2	22.3
Newton-MG	19	1	24.6
FAS	11	–	27.1

TABLE 3. *Comparison of FAS, Newton, and Newton-multigrid methods for the problem* $-\Delta u + \gamma u e^u = f$ *on a* 127×127 *grid. In all cases, a zero initial guess is used.*

Source: Henson, V E. Multigrid Methods for Nonlinear Problems: An Overview, 2002. doi:10.1117/12.499473.

Nonlinear Preconditioning

- Nonlinear preconditioning scheme

$$F(u,v) := \begin{bmatrix} F_1(u,v) \\ F_2(u,v) \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

Precondition?

$$\mathcal{F}(u,v) := \begin{bmatrix} \mathcal{F}_1(u,v) \\ \mathcal{F}_2(u,v) \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

Decouple

$$\begin{cases} F_1(u-g,v) = 0 \implies g(u,v) \\ F_2(u,v-h) = 0 \implies h(u,v) \end{cases}$$

$$\begin{bmatrix} u \\ v \end{bmatrix} \longrightarrow \begin{bmatrix} g \\ h \end{bmatrix}$$

Solution — Zero if converged

- Nonlinear Gauss-Seidel iteration

Divide-and-Conquer

- Preconditioned system $(g,h) = 0$ is defined implicitly using the above Schwarz method
- Q: How to compute the Jacobian of the preconditioned system?

Ref: Xiao-Chuan Cai and David E. Keyes, Nonlinearly Preconditioned Inexact Newton Algorithms, SIAM Journal on Scientific Computing, 2002, 24:1, 183-200

Jacobian of Preconditioned Problem

Define $\nabla \mathcal{F} := \begin{bmatrix} \frac{\partial g}{\partial u} & \frac{\partial g}{\partial v} \\ \frac{\partial h}{\partial u} & \frac{\partial h}{\partial v} \end{bmatrix}$ $\qquad p := u - g(u,v), \quad q := v - h(u,v)$

$$\frac{\partial F_1}{\partial p}\left(I_u - \frac{\partial g}{\partial u}\right) = 0 \longrightarrow \text{Assume } \frac{\partial F_1}{\partial p} \text{ is nonsingular} \longrightarrow \frac{\partial g}{\partial u} = I_u$$

$$\frac{\partial F_1}{\partial p}\left(-\frac{\partial g}{\partial v}\right) + \frac{\partial F_1}{\partial v} = 0 \longrightarrow \frac{\partial g}{\partial v} = \left(\frac{\partial F_1}{\partial p}\right)^{-1} \frac{\partial F_1}{\partial v}$$

$$\nabla \mathcal{F} := \begin{bmatrix} \frac{\partial g}{\partial u} & \frac{\partial g}{\partial v} \\ \frac{\partial h}{\partial u} & \frac{\partial h}{\partial v} \end{bmatrix} = \begin{bmatrix} \left(\frac{\partial F_1}{\partial p}\right)^{-1} & 0 \\ 0 & \left(\frac{\partial F_2}{\partial q}\right)^{-1} \end{bmatrix} \begin{bmatrix} \frac{\partial F_1}{\partial p} & \frac{\partial F_1}{\partial v} \\ \frac{\partial F_2}{\partial u} & \frac{\partial F_2}{\partial q} \end{bmatrix}$$

ASPIN Method

$$\nabla \mathcal{F} \approx \begin{bmatrix} \left(\frac{\partial F_1}{\partial u}\right)^{-1} & 0 \\ 0 & \left(\frac{\partial F_2}{\partial v}\right)^{-1} \end{bmatrix} \begin{bmatrix} \frac{\partial F_1}{\partial u} & \frac{\partial F_1}{\partial v} \\ \frac{\partial F_2}{\partial u} & \frac{\partial F_2}{\partial v} \end{bmatrix} = \begin{bmatrix} \left(\frac{\partial F_1}{\partial u}\right)^{-1} & 0 \\ 0 & \left(\frac{\partial F_2}{\partial v}\right)^{-1} \end{bmatrix} \nabla F$$

Inexact Newton iteration: $\nabla \mathcal{F}(x)\, d = -\mathcal{F}(x)$

$$\begin{bmatrix} w_1 \\ w_2 \end{bmatrix} := \nabla F(u, v) \quad \Rightarrow \quad \text{Solve} \quad \left(\frac{\partial F_1}{\partial u}\right) y_1 = w_1, \quad \left(\frac{\partial F_2}{\partial v}\right) y_2 = w_2$$

Additive Schwarz Preconditioned Inexact Newton (ASPIN)

⇒ Multiplicative Schwarz Preconditioned Inexact Newton (MSPIN)

Ref: Liu, Lulu; Keyes, David E. “Field-Split Preconditioned Inexact Newton Algorithms”, SIAM Journal on Scientific Computing, 2015, 37 (3):A1388

Some Comments on ASPIN

- What about the function is actually a linear function

$$F(x) = Ax - b$$

$$A = \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix}, \quad b = \begin{bmatrix} b_1 \\ b_2 \end{bmatrix}, \quad \nabla \mathcal{F} \approx \begin{bmatrix} A_{11} & 0 \\ 0 & A_{22} \end{bmatrix}^{-1} \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix}$$

- ASPIN ➔ A nonlinear version of block Jacobi preconditioner

TABLE 6.4

Different subdomain partitions with the same fine mesh 128×128. *Subdomain linear systems are solved exactly.* $\epsilon_{global-linear-rtol} = 10^{-3}$ *is the stopping condition for the global GMRES iterations.* $\epsilon_{local-nonlinear-rtol} = 10^{-3}$ *is the stopping condition for the local nonlinear iterations. Overlap* $= 1$. *The finite difference step size is* 10^{-8}.

	Number of PIN iterations				
Subdomain partition	$Re = 10^0$	$Re = 10^1$	$Re = 10^2$	$Re = 10^3$	$Re = 10^4$
$N = 2 \times 2 = 4$	3	3	4	6	7
$N = 4 \times 4 = 16$	3	2	4	7	6
	Number of GMRES iterations per PIN				
$N = 2 \times 2 = 4$	22	21	23	20	15
$N = 4 \times 4 = 16$	42	37	40	31	26

Source: Liu, Lulu; Keyes, David E. “Field-Split Preconditioned Inexact Newton Algorithms”, SIAM Journal on Scientific Computing, 2015, 37 (3):A1388

Reading and Thinking

SIAM J. SCI. COMPUT.
Vol. 37, No. 3, pp. A1388–A1409

FIELD-SPLIT PRECONDITIONED INEXACT NEWTON ALGORITHMS*

LULU LIU† AND DAVID E. KEYES†

Abstract. The multiplicative Schwarz preconditioned inexact Newton (MSPIN) algorithm is presented as a complement to additive Schwarz preconditioned inexact Newton (ASPIN). At an algebraic level, ASPIN and MSPIN are variants of the same strategy to improve the convergence of systems with unbalanced nonlinearities; however, they have natural complementarity in practice. MSPIN is naturally based on partitioning of degrees of freedom in a nonlinear PDE system by field type rather than by subdomain, where a modest factor of concurrency can be sacrificed for physically motivated convergence robustness. ASPIN, originally introduced for decompositions into subdomains, is natural for high concurrency and reduction of global synchronization. We consider both types of inexact Newton algorithms in the field-split context, and we augment the classical convergence theory of ASPIN for the multiplicative case. Numerical experiments show that MSPIN can be significantly more robust than Newton methods based on global linearizations, and that MSPIN can be more robust than ASPIN and maintain fast convergence even for challenging problems, such as high Reynolds number Navier–Stokes equations.

Key words. nonlinear equations, nonlinear preconditioning, field splitting, Newton method, Navier–Stokes equations

AMS subject classifications. 65H10, 65H20, 65N22, 65N55

DOI. 10.1137/140970379

- Is your application a nonlinear problem?
- How did you handle nonlinearity?
- Do you think divide-and-conquer will work for nonlinear problems? Why?
- Do you think multilevel algorithms will work for nonlinear problems? Why?
- What is your first choice to handle nonlinear problems now?

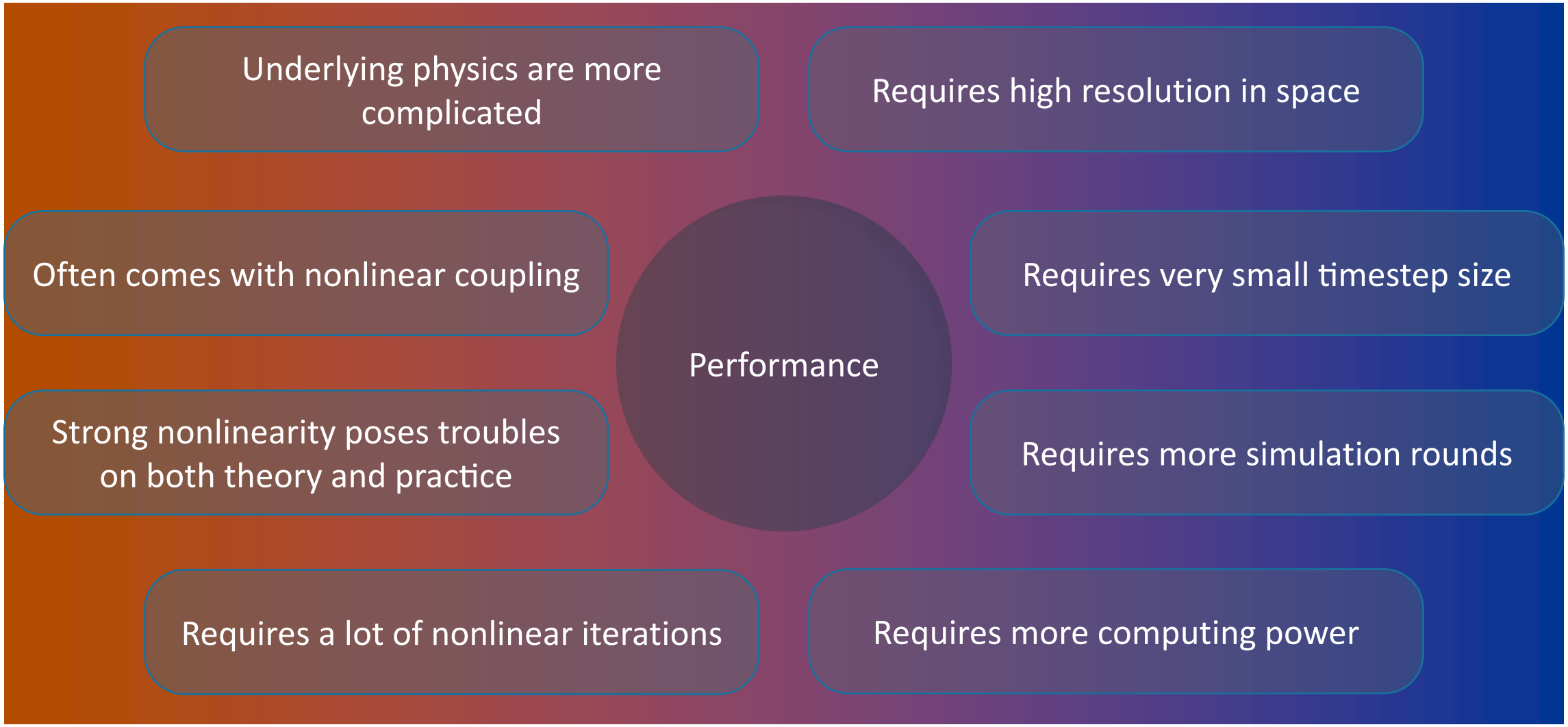
Contact Me

- Office hours: Mon 14:00—15:00
- Walk-in or online with appointment
- zhangcs@lsec.cc.ac.cn
- http://lsec.cc.ac.cn/~zhangcs

My sincere gratitude to:

Liuqiang Zhong, Xin Liu, Liang Chen, Bin Dai

Review


Underlying physics are more complicated
Requires high resolution in space
Often comes with nonlinear coupling
Performance
Requires very small timestep size
Strong nonlinearity poses troubles on both theory and practice
Requires more simulation rounds
Requires a lot of nonlinear iterations
Requires more computing power

中国科学院大学
夏季强化课程
2022

Fast Solvers for Large Algebraic Systems

Chensong Zhang, AMSS

http://lsec.cc.ac.cn/~zhangcs

Release version 2022.06.26