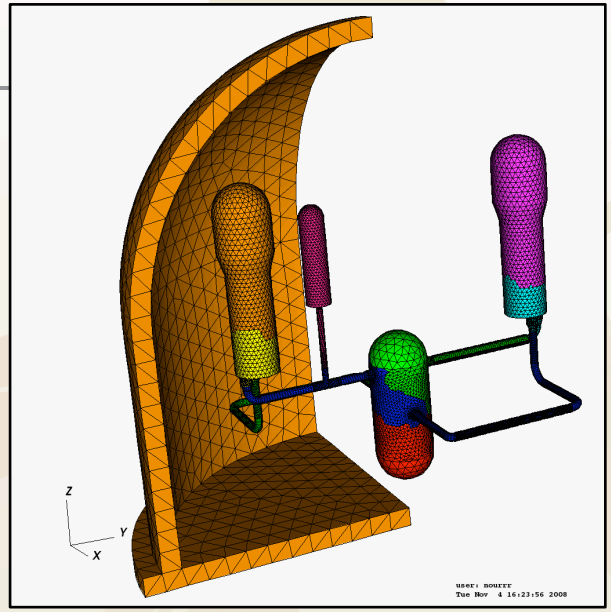
International RELAP User Group
August 11-13, 2009, Park City, UT, USA

Idaho National Laboratory

Requirements, Architecture and Computational Platform of a Next Generation System Code

Dr. Robert Nourgaliev
Dr. Vincent Mousseau
Dr. Nam Dinh
Idaho National Laboratory (INL)

Outline



I. System Analysis Codes

II. Requirements

III. Architecture

IV. Numerics

V. Coarse-grain modeling

I. System Analysis Codes

System Analysis Codes

Safety margin characterization
Balance between design, operation and safety code
Help in decision making
Licensing

Goals:
Reactor Safety and Risk Assessment and Management

Probabilistic Risk Assessment (PRA)

Simulator: Training

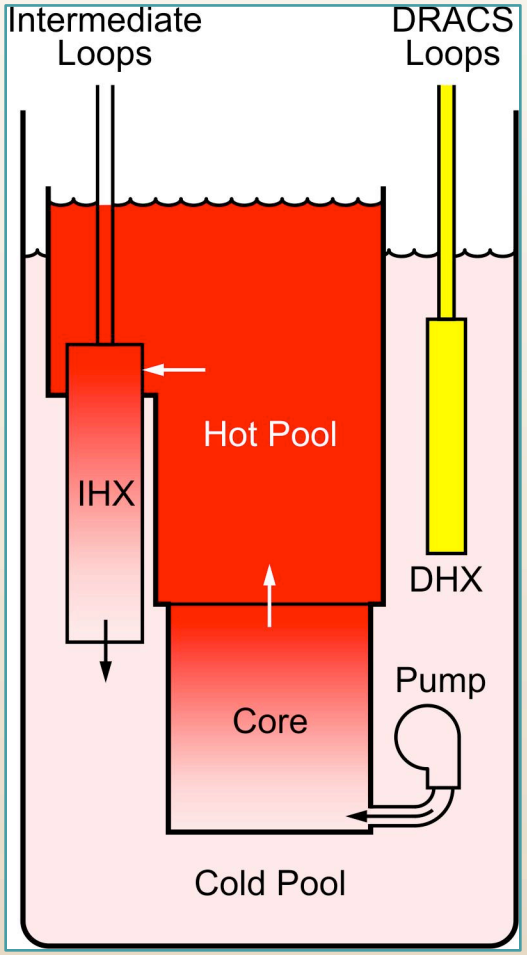
LOCA, DBA
(Legacy codes: RELAP5, TRACE, CATHARE)

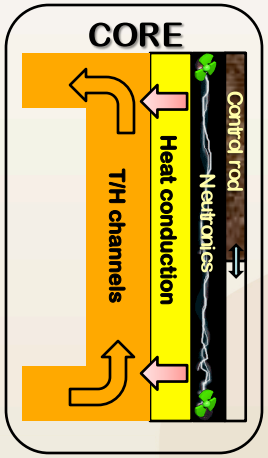
System vulnerability search engine

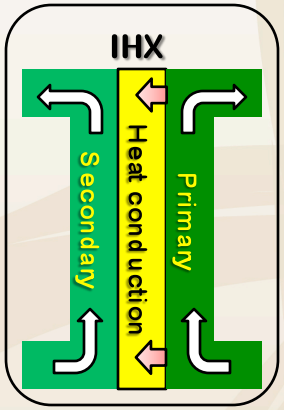
System Analysis Codes ≠ CFD!

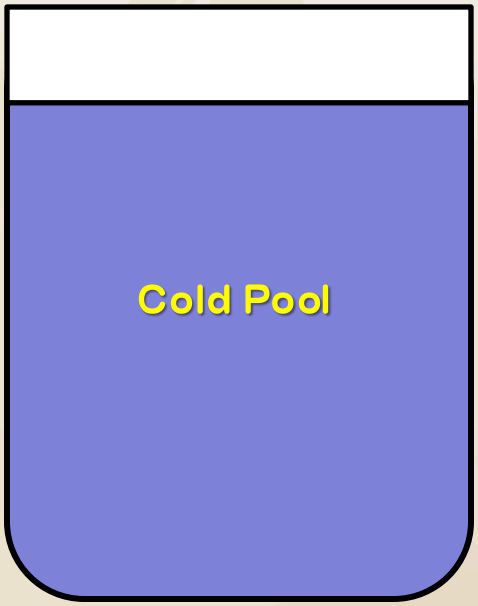
System components

ABR model





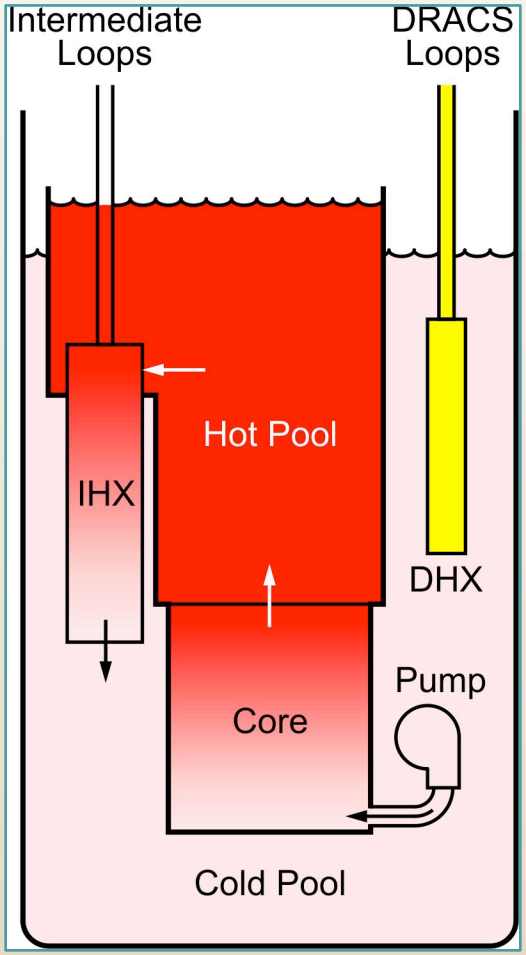




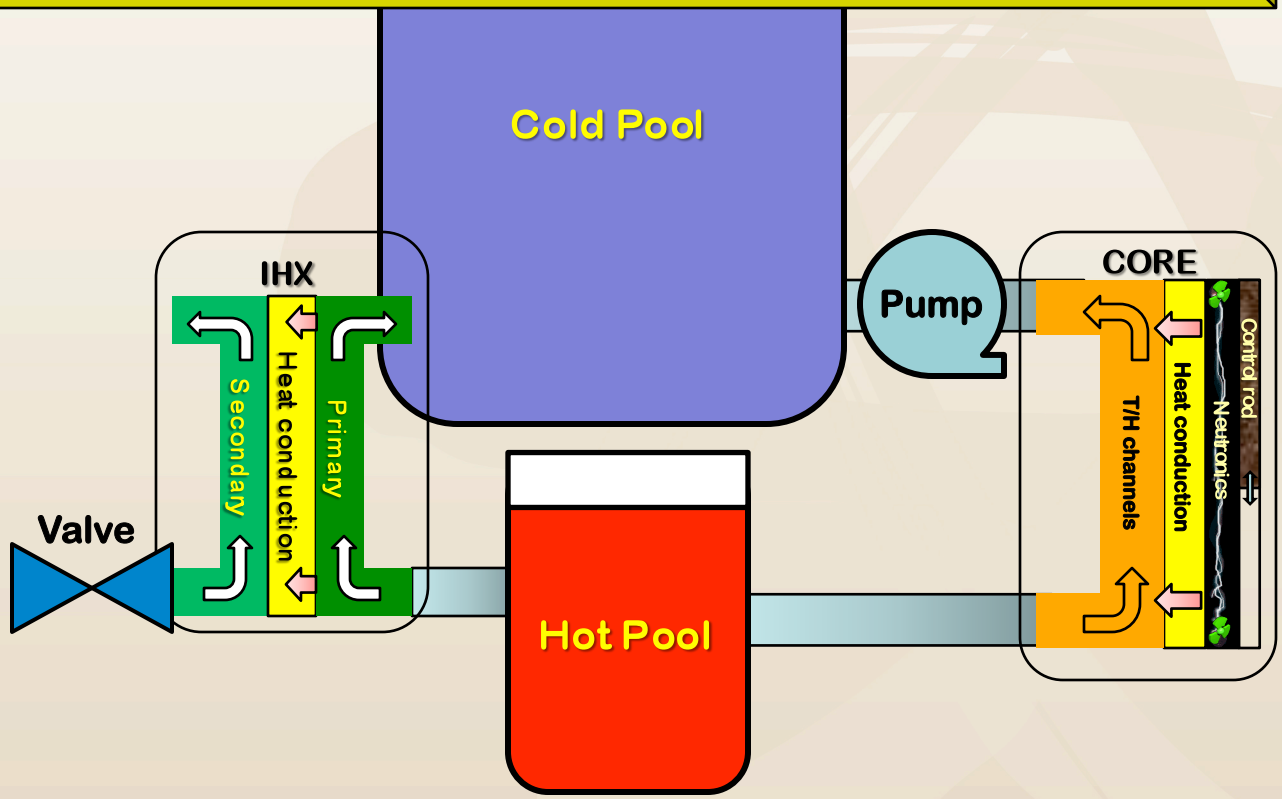
System Analysis Codes ≠ CFD!

System components

ABR model

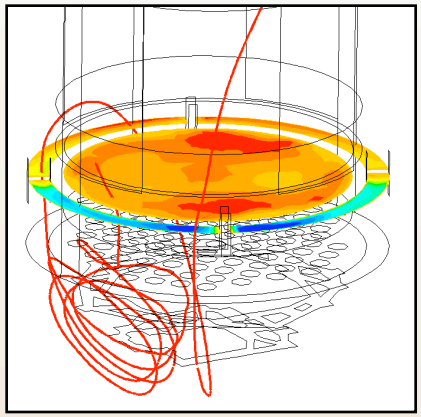


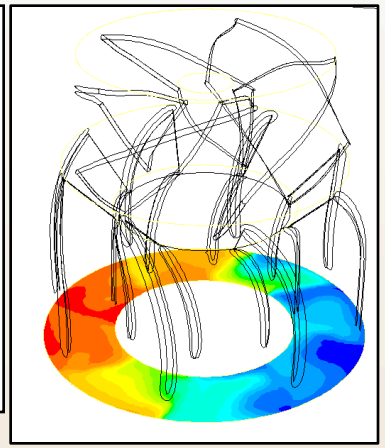
The best possible representation of the System

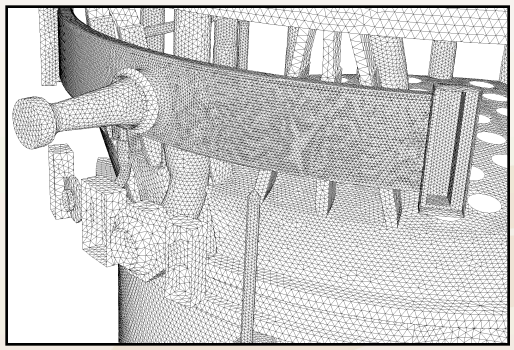


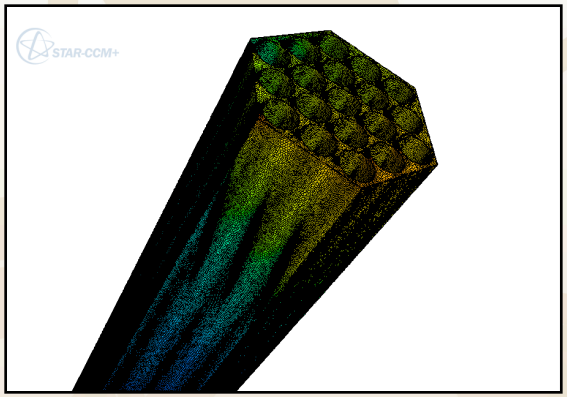
CFD:

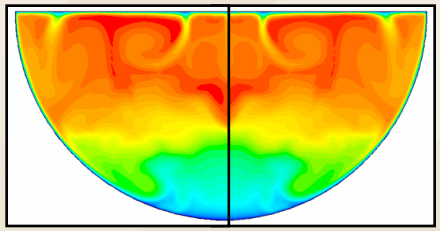
The best possible representation of the physics and geometry

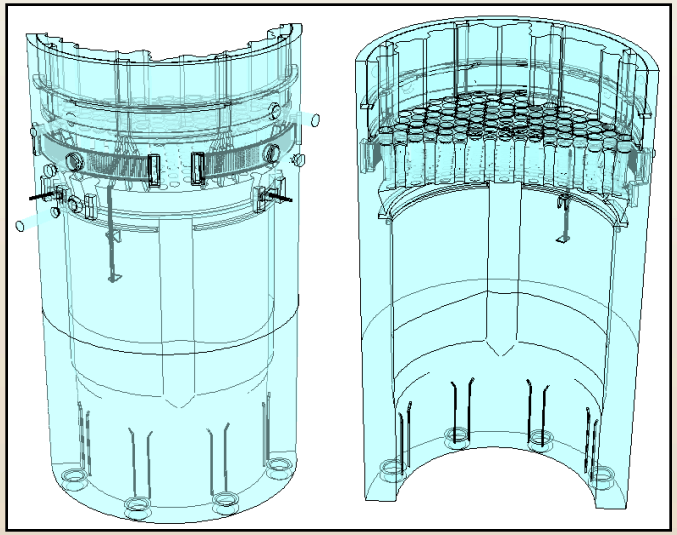


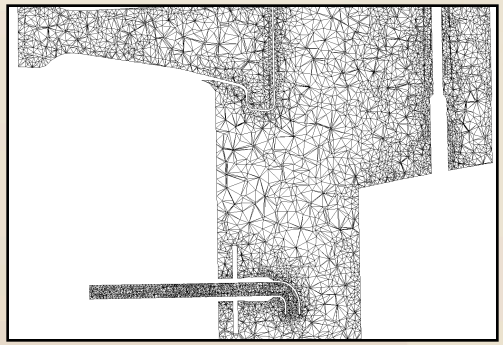


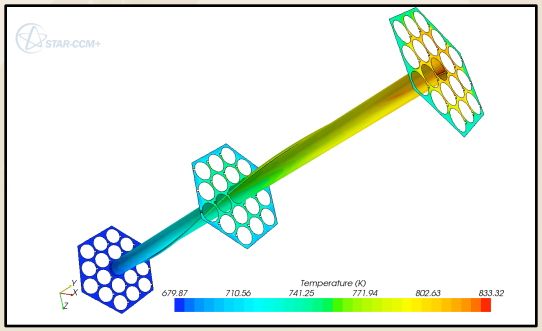


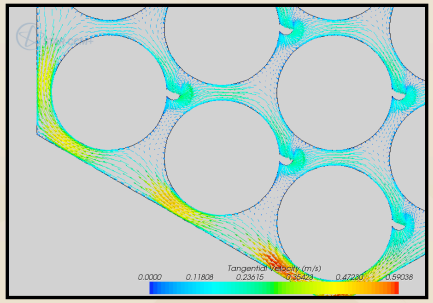






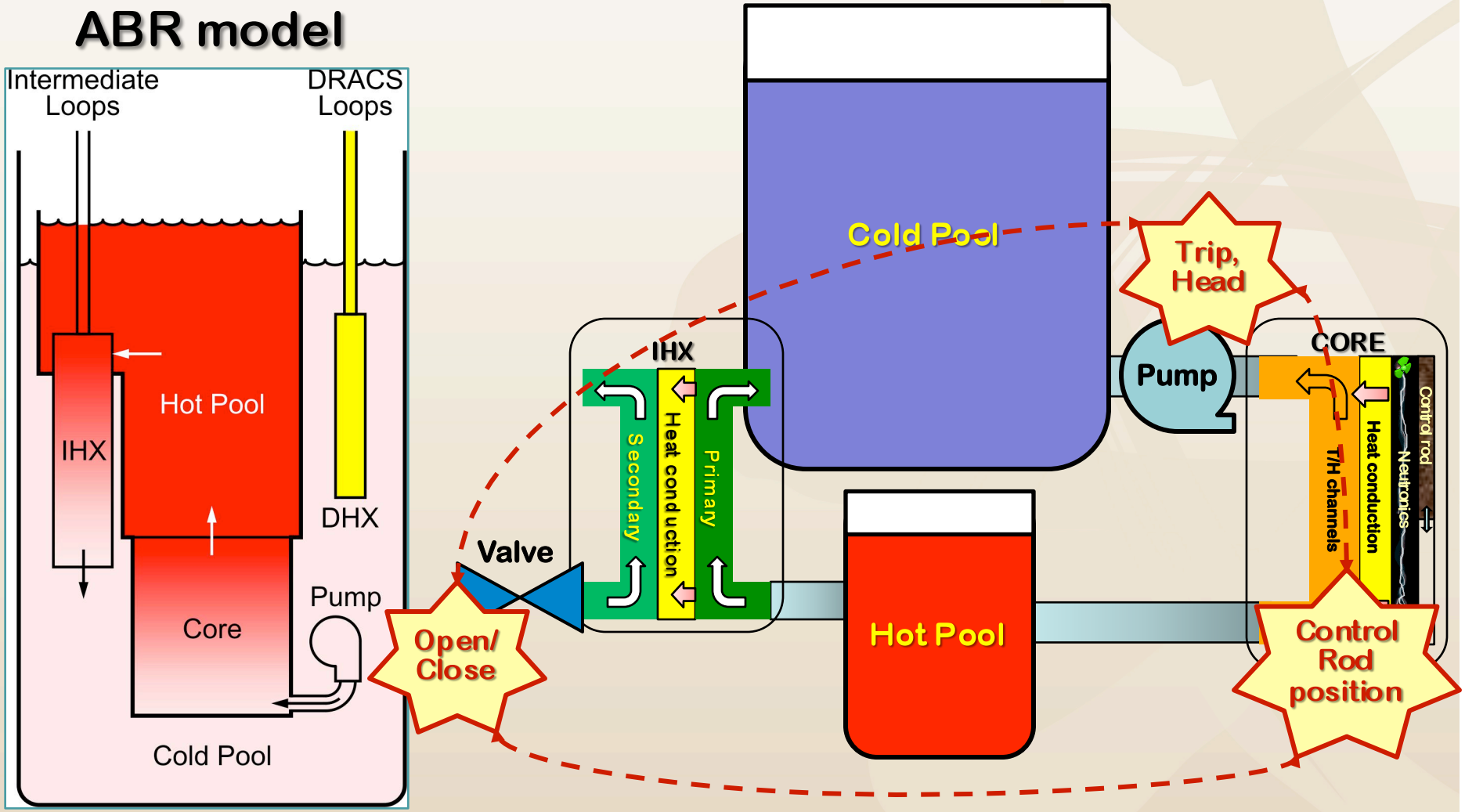






Control System

ABR model



II. Requirements

Requirements

- Multiple fidelity

7D components:
Reactor core assembly
(neutron transport)

Topological heterogenuity

0D components:
Pumps, pools,
T-junctions, turbines….

1D components:
Straight pipes, elbows,
variable-area pipes…

2D/3D components:
Pools, T-junctions,
downcomers…

Modeling

Neutronics:
Point kinetics
Quasi-Static Diffusion
Diffusion Approximation
Quasi-Diffusion
Transport

Pipe Flow:
1D: Total Mixing
"1.5D": Account for stratification

Pool Mixing:
Bulk model (total mixing, 0D)
Effective conductivity/convectivity
RANS CFD
LES CFD
DNS

Requirements

- Multiple fidelity
- Multiphysics

Tight coupling algorithms

Thermohydraulics (1-phase and multi-phase)
Thermo-structural analysis
Neutronics
Fuel behavior
Coolant chemistry
Fission Product Transport (FPT)
etc.

Requirements

- Multiple fidelity
- Multiphysics
- Uncertainty Quantification

High-order spatio-temporal discretization (local error estimates)

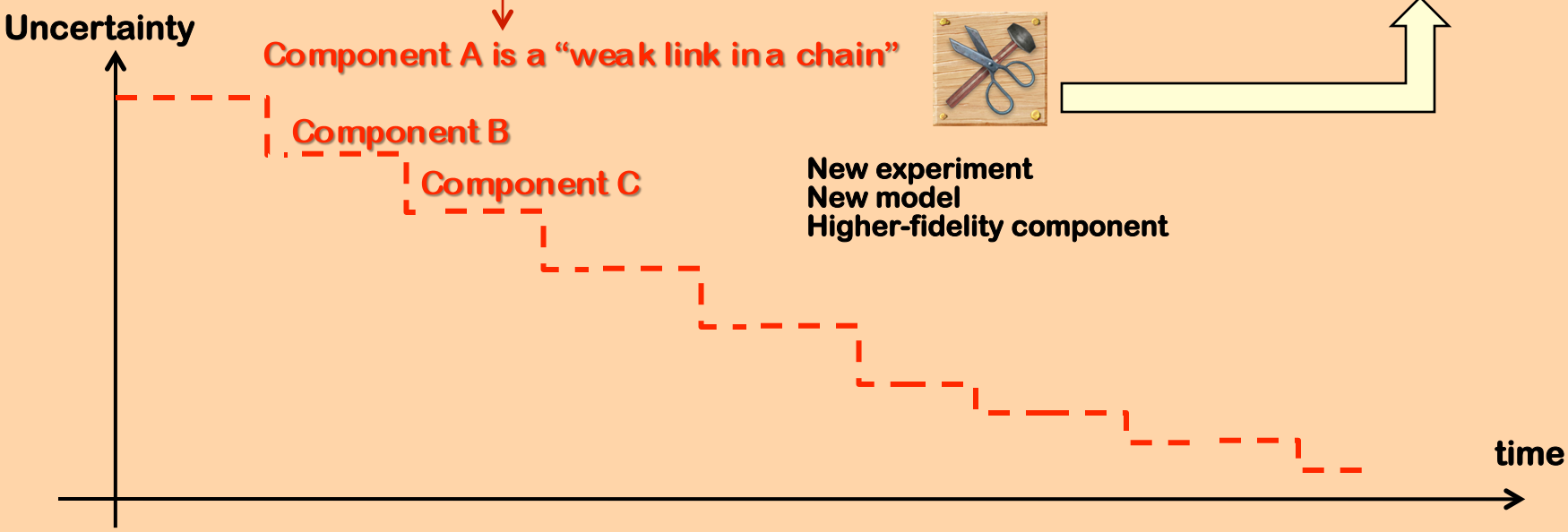
(global error estimates)

Monte-Carlo based ("black-box", DAKOTA)
Forward Sensitivity Analysis
Adjoint Analysis

Requirements

- Multiple fidelity
- Multiphysics
- Uncertainty Quantification
- Long-term code "improvability"

Industry:
Customer/Funding Organization



New experiment
New model
Higher-fidelity component

Requirements

- Multiple fidelity
- Multiphysics
- Uncertainty Quantification
- Long-term code "improvability"

"Fidelity-on-demand" (Adaptive Model Refinement, AMoR):

Individually, component-by-component, this will not necessary be the "State-of-the-Art"

It should be the "State-of-the-Art" as a system analysis tool

Requirements

- Multiple fidelity
- Multiphysics
- Uncertainty Quantification
- Long-term code "improvability"
- Compatible with risk analysis tools/ methodologies PRA, SAR, ROAAM, etc. and "system vulnerability search engine"

Smoothness of models

"Differentiability" (at least numerical) of code results

Jacobians (Hessians) are computable

⇩

Jacobian-free Newton Krylov and
High-order spatio-temporal discretization

Requirements

- Multiple fidelity
- Multiphysics
- Uncertainty Quantification
- Long-term code "improvability"
- Compatible with risk analysis tools/ methodologies and "system vulnerability search engine"
- "Scalability" of the code development environment

Community-scale effort

~50-100 code developers/contributers

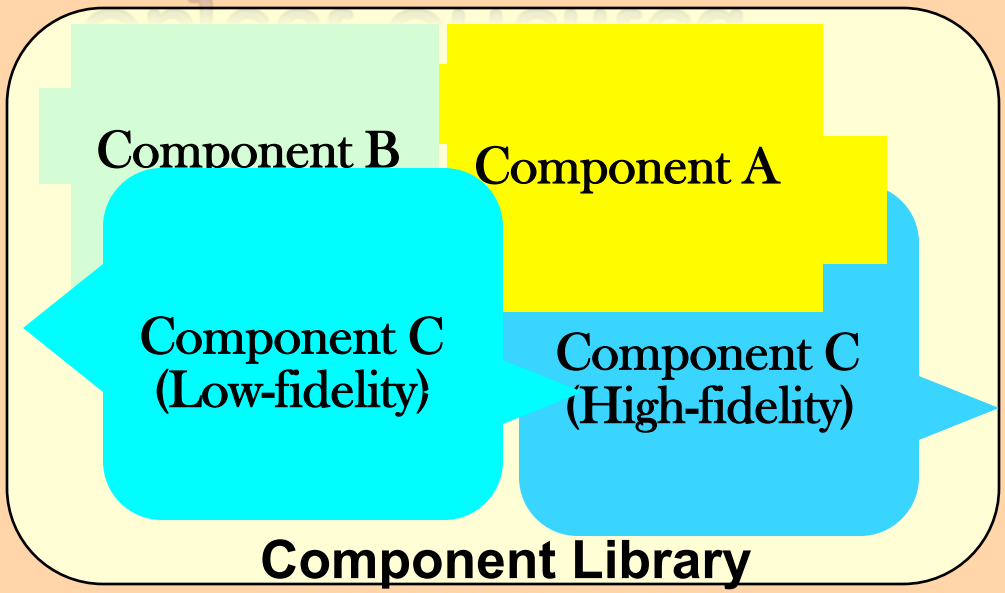
Requirements

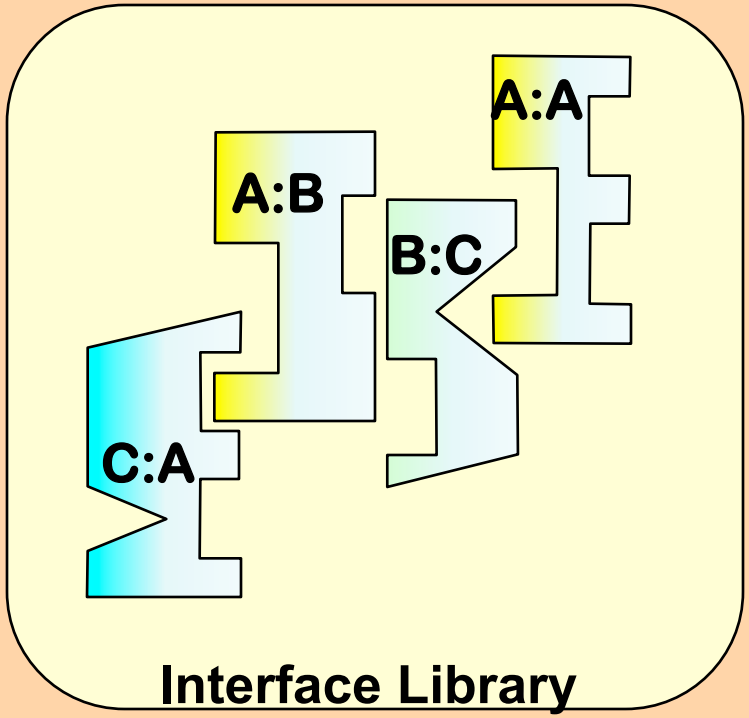
- Multiple fidelity
- Multiphysics
- Uncertainty Quantification
- Long-term code "improvability"
- Compatible with risk analysis tools/ methodologies and "system vulnerability search engine"
- "Scalability" of the code development environment

"Open-source" mentality

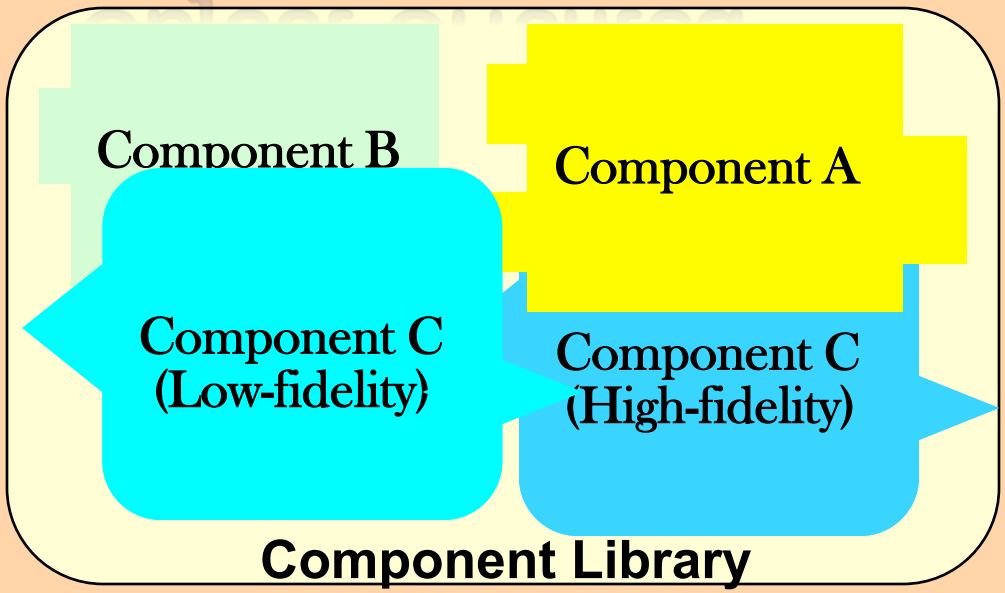
III. Architecture

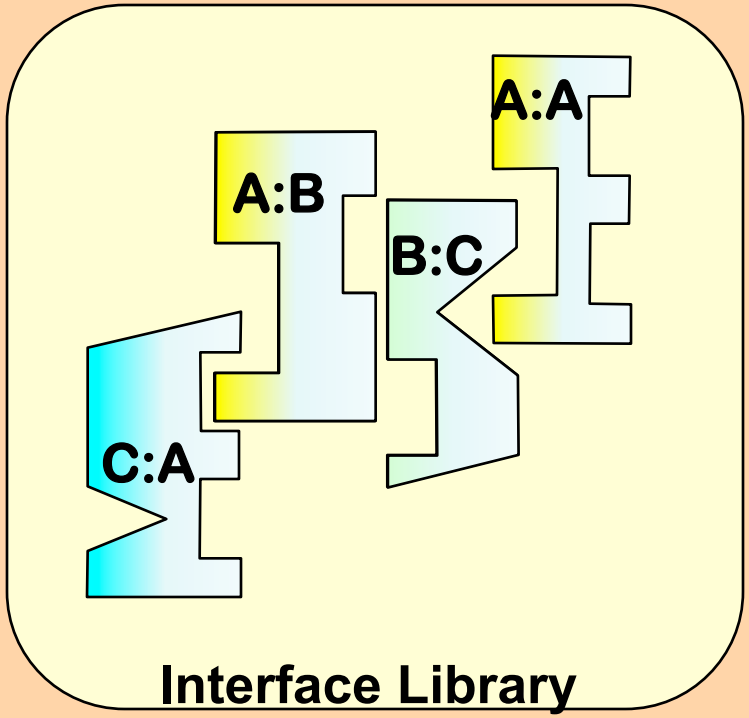
Object-Oriented





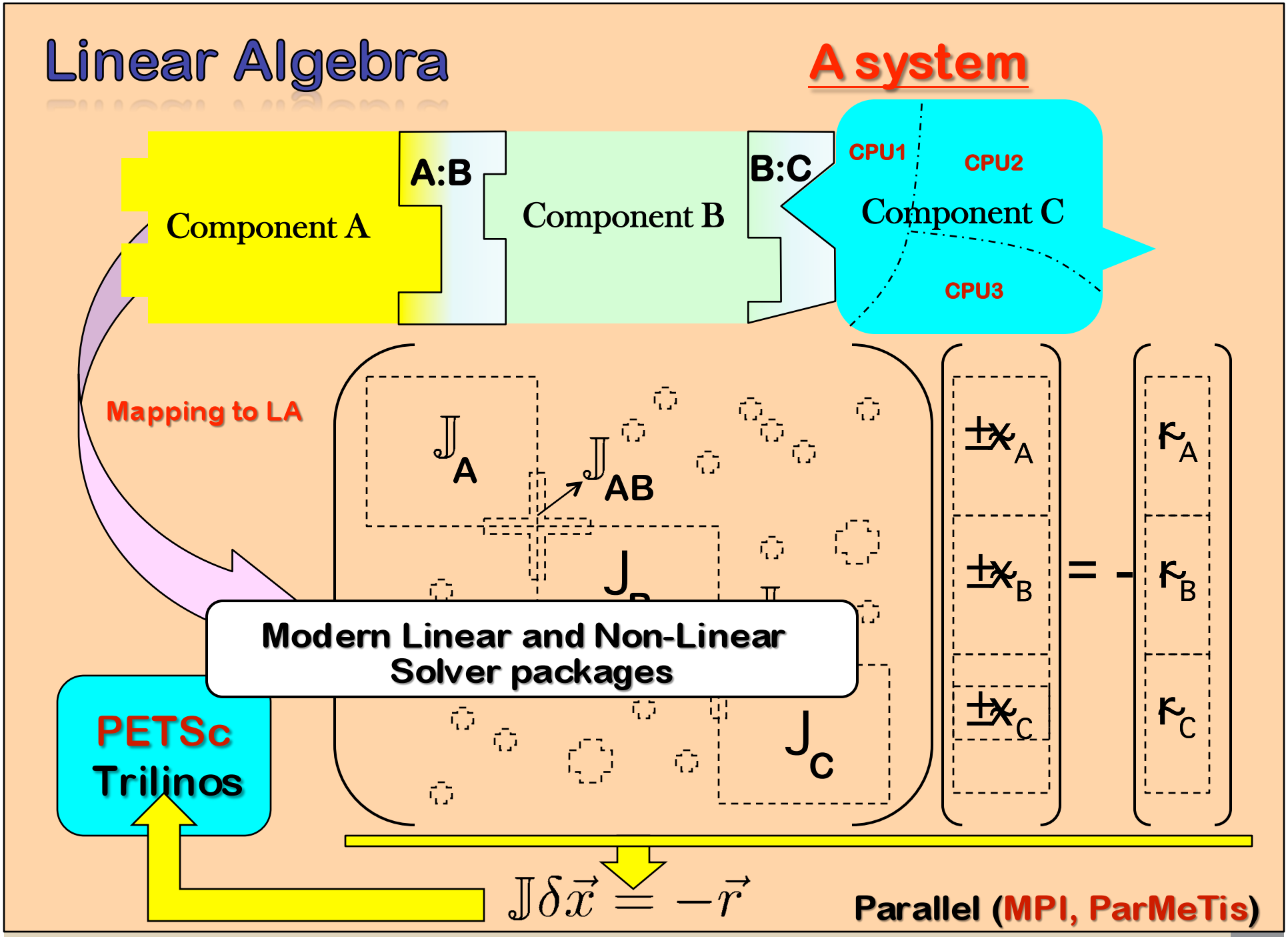
Object-Oriented





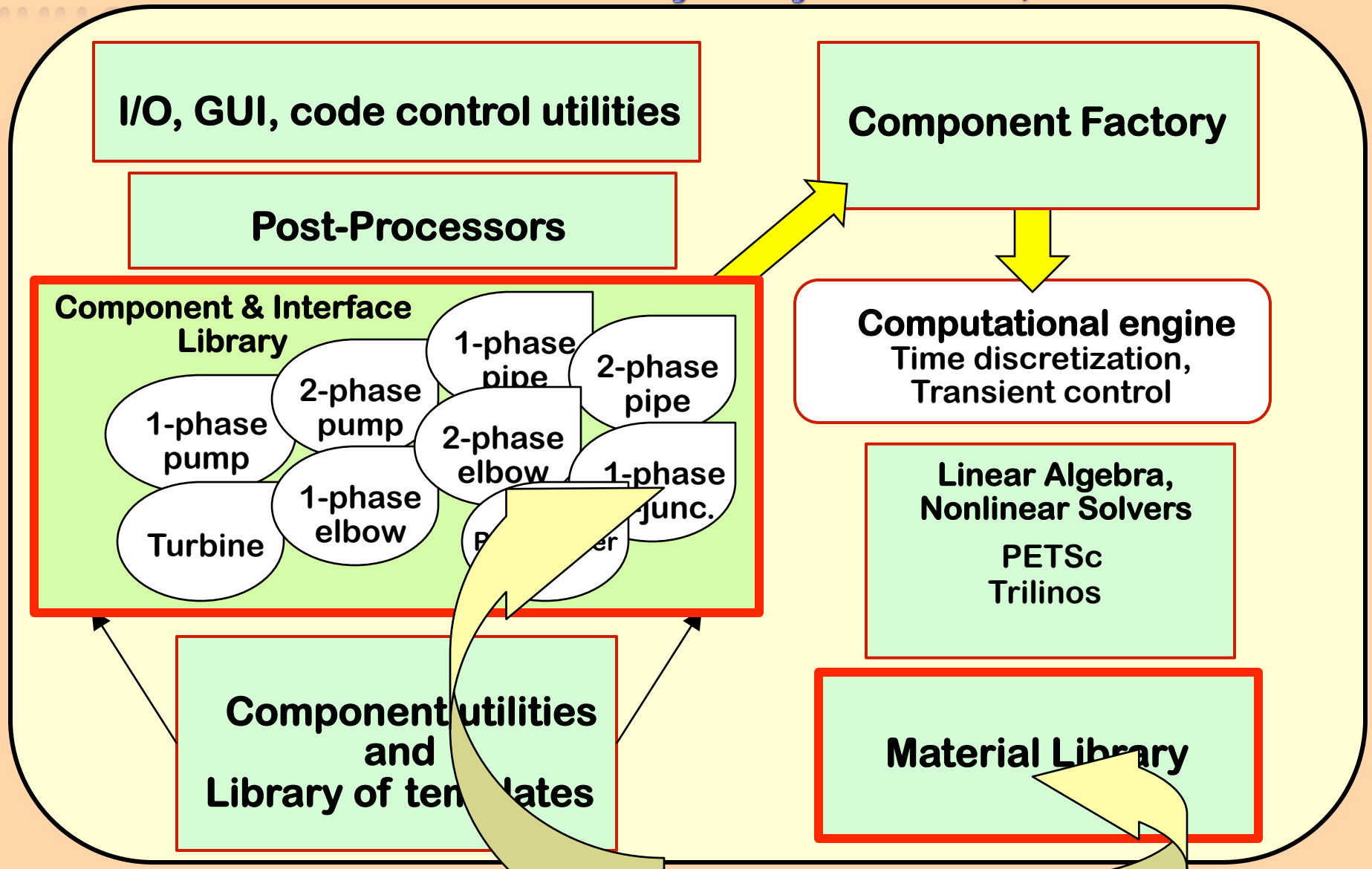
A system

Linear Algebra



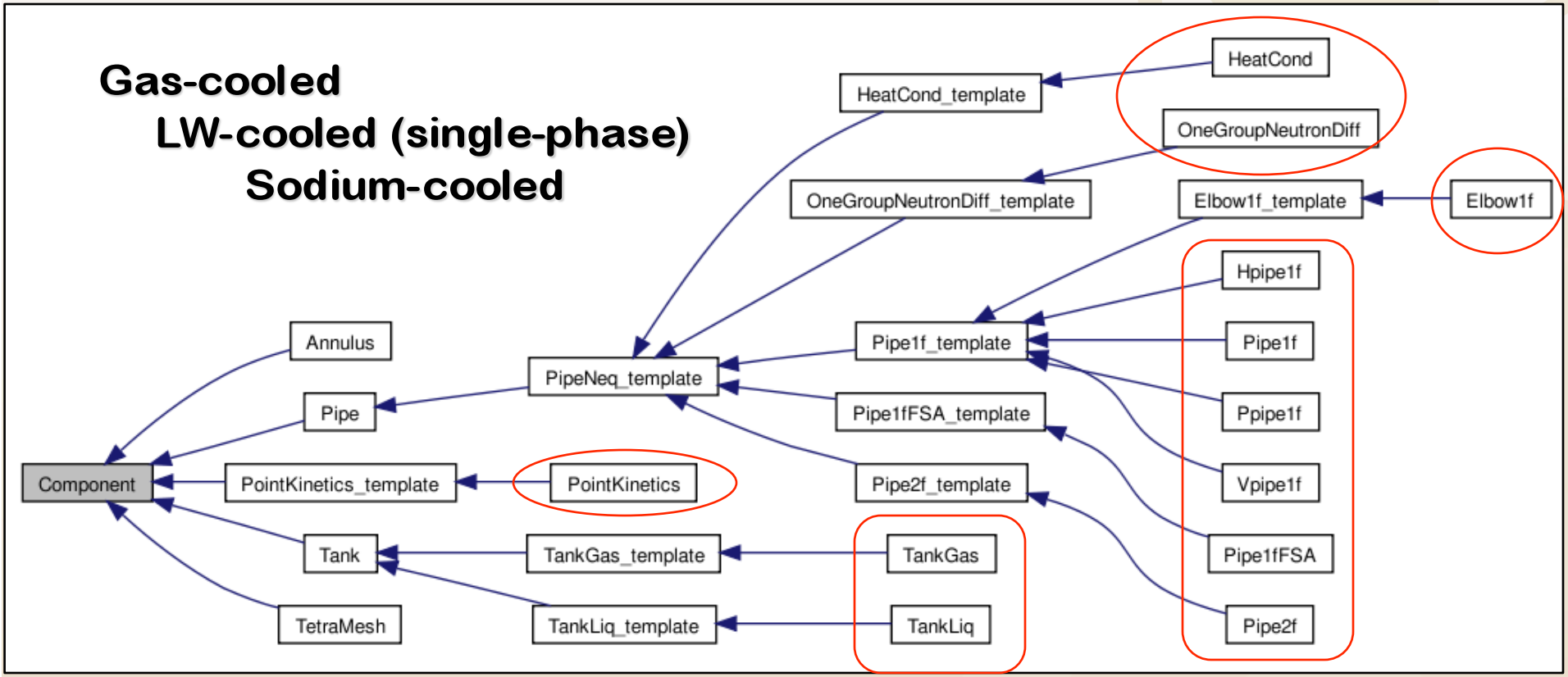
Infrastructure

Safety Analysis Vehicle, SAV

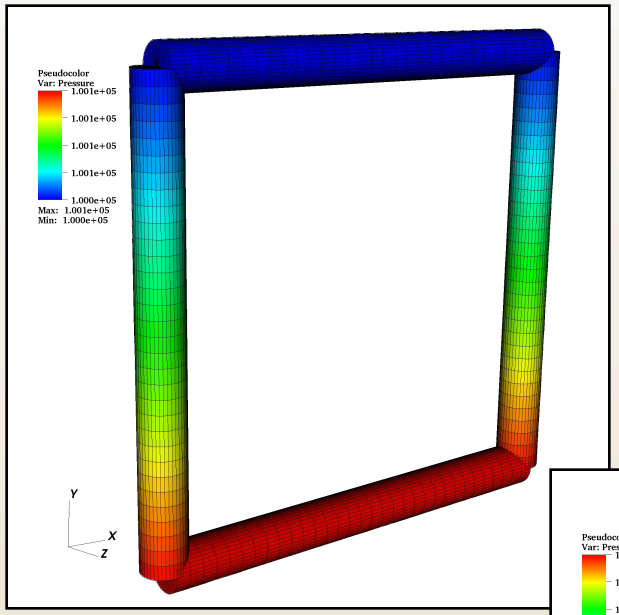


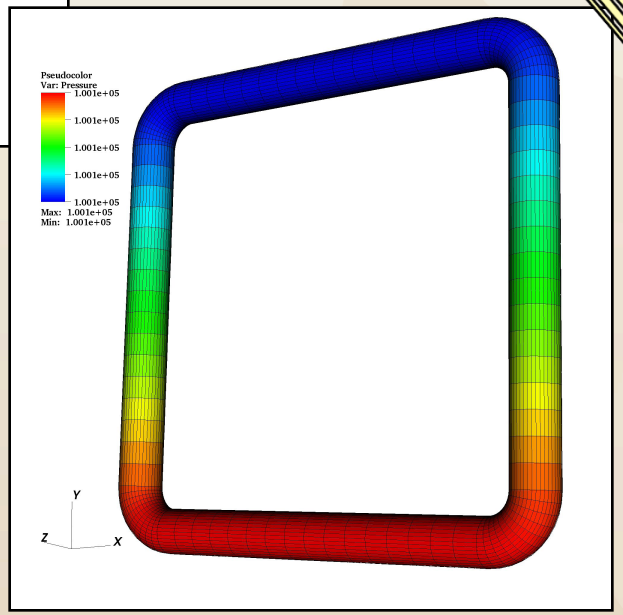
Open source: community

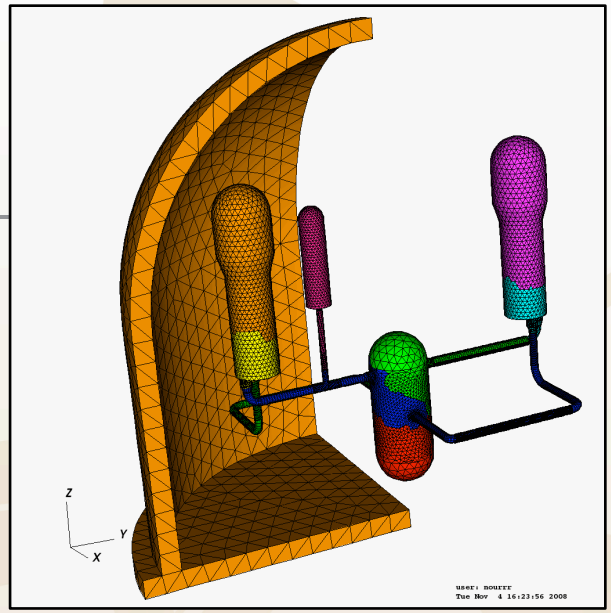
Component hierarchical tree of version β-1.0



Version β-1.0







...Future...

IV. Numerics

Numerics

State-of-the-Art for mid-1970s

Legacy codes (RELAP5, TRAC, T

Time discretization:

Semi-Implicit (ICE-based)
Nearly-implicit
SETS
Operator-Split
(1st-order)

No means to quantify numerical discretization uncertainties

Space discretization:

Staggered-grid, Donor-Cell (upwinding)
(1st-order)

Linear Algebra:

Direct Solvers
Sparse Gaussian Elimination
TDMA
(Scale as ~N^3)

Programming:

Serial, FORTRAN-IV/FORTRAN-77 → FORTRAN-90

Numerics

Tremendous advances are made in numerical discretizations/linear algebra theory and computer science

Time discretization:

Fully-Implicit, high-order, L-stable
Runge-Kutta Methods

Space discretization:

High-order finite-volume,
Finite-elements,
Discontinuous Galerkin
Godunov-based,
Collocated-grid

Linear Algebra:

Efficient Iterative Linear and
Non-linear Solvers
Krylov methods
Multigrid
(Scale as ~N log N)

Programming:

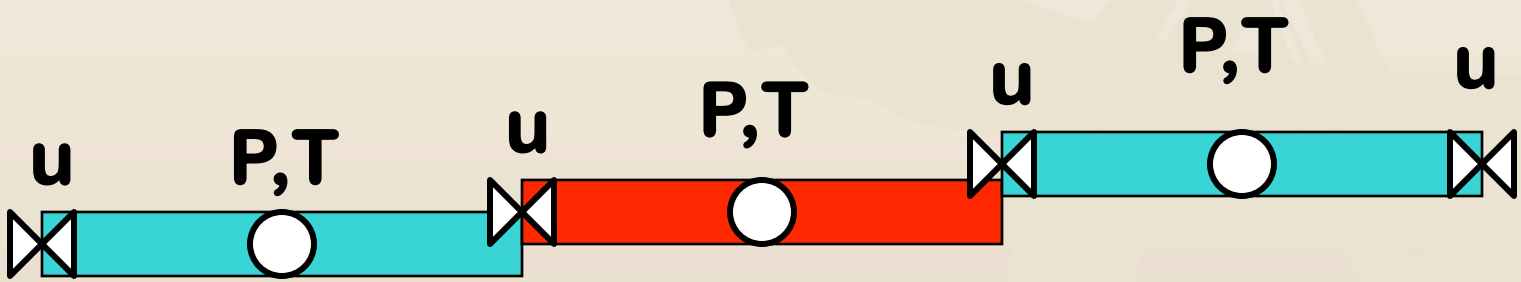
Parallel (MPI), Object-oriented (C++, Java)

Numerics

Space discretization

(RELAP5, TRAC, TRACE, CATHARE, ATHLETE)

Staggered-grid, Donor-Cell (upwinding)



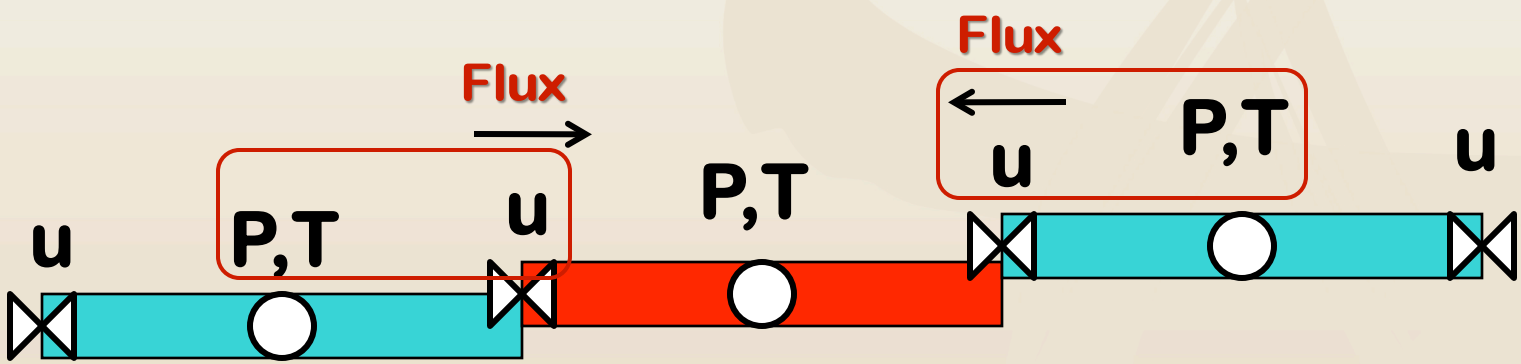
Numerics

Space discretization

(RELAP5, TRAC, TRACE, CATHARE, ATHLETE)

Staggered-grid, Donor-Cell (upwinding)

Only linear waves (contacts) are assumed
(Adequate only for incompressible flow)



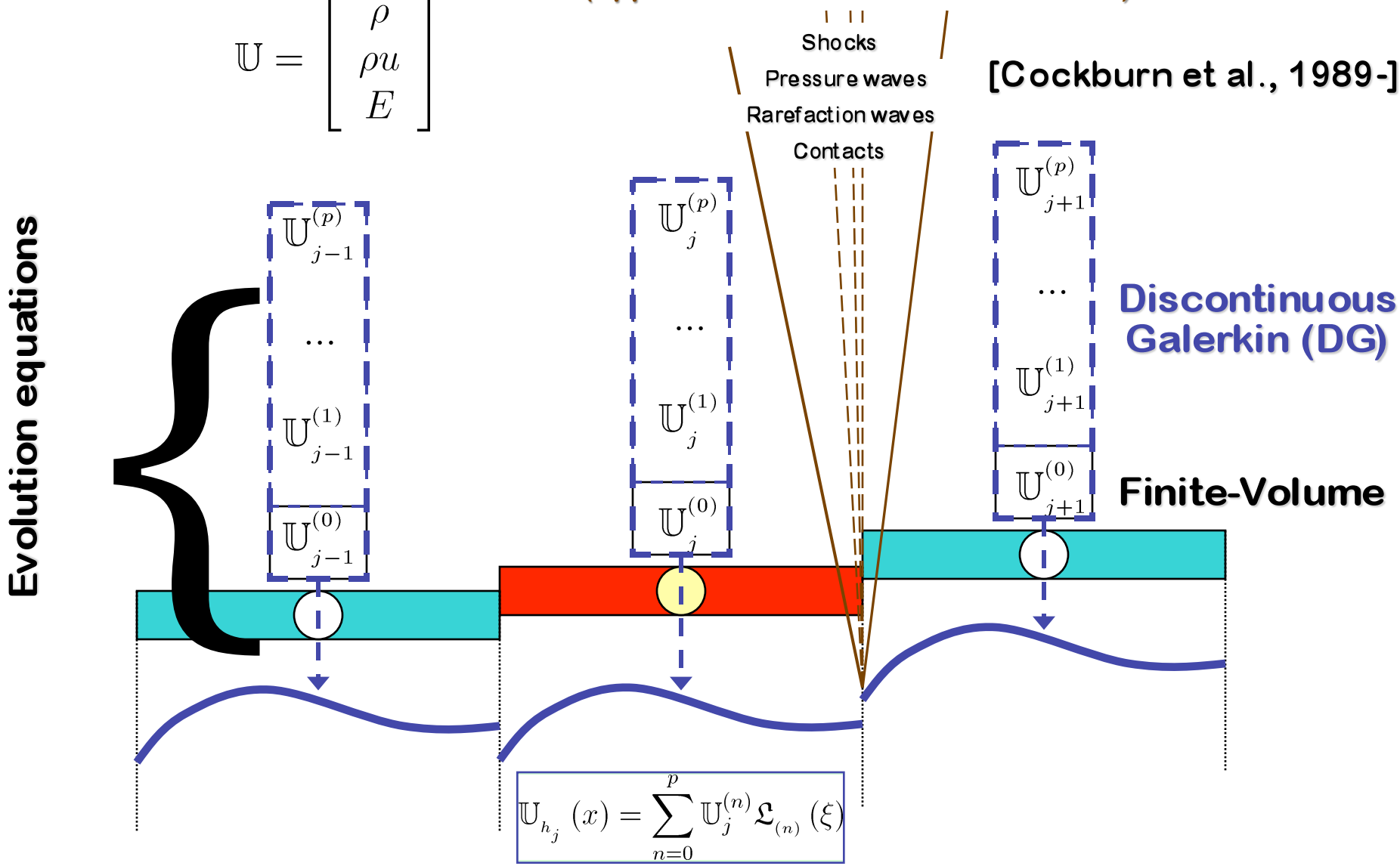
Recovery Discontinuous Galerkin (rDG)

Collocated

$$\mathbb{U} = \begin{bmatrix} \rho \\ \rho u \\ E \end{bmatrix}$$

Flux:
Upwinding based on the wave structure
(Approximate or Exact Riemann Solvers)

[Cockburn et al., 1989-]



$$\mathbb{U}_{h_j}(x) = \sum_{n=0}^{p} \mathbb{U}_j^{(n)} \mathfrak{L}_{(n)}(\xi)$$

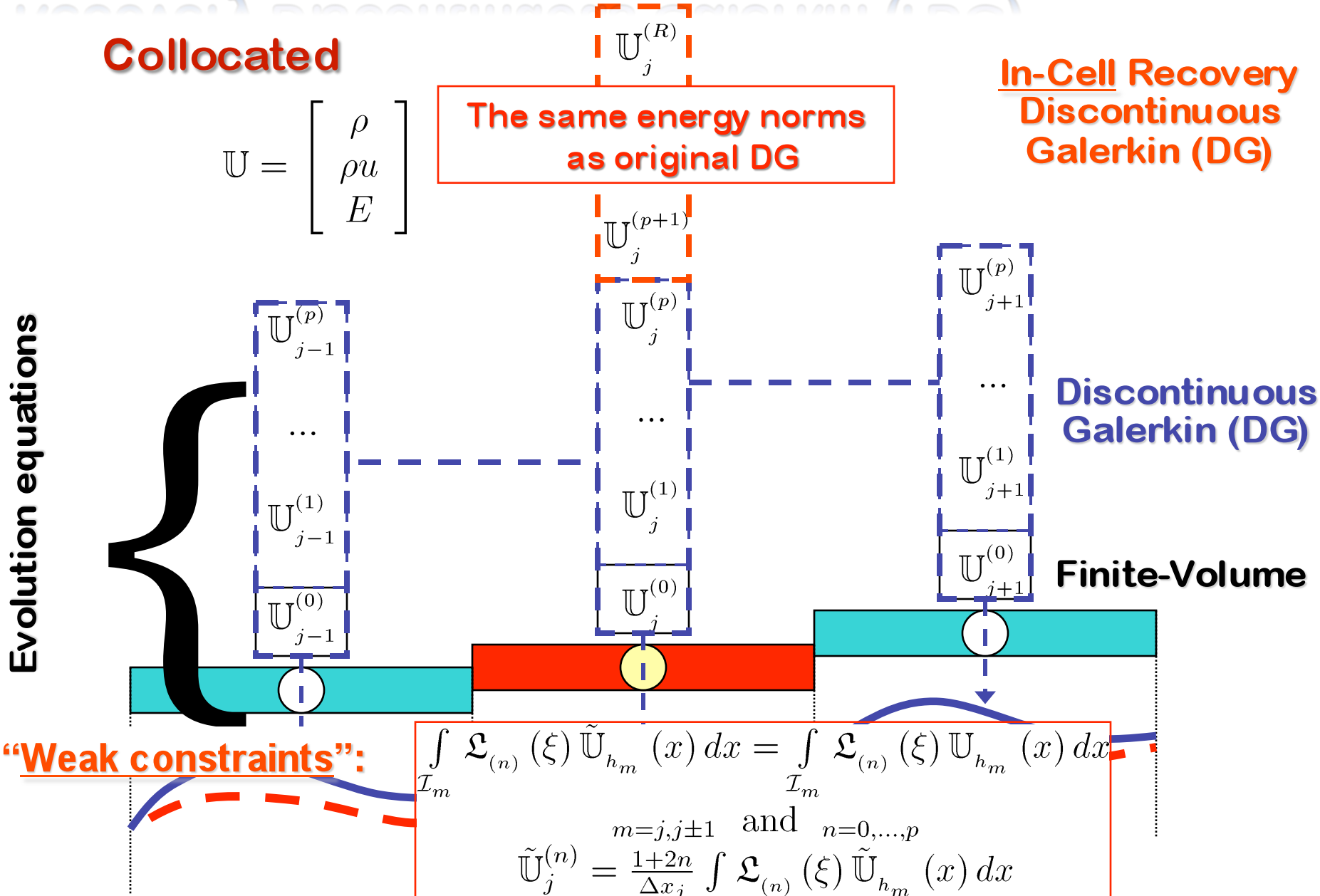
Recovery Discontinuous Galerkin (rDG)

Collocated

$$\mathbb{U} = \begin{bmatrix} \rho \\ \rho u \\ E \end{bmatrix}$$

The same energy norms as original DG

In-Cell Recovery Discontinuous Galerkin (DG)



"Weak constraints":

$$\int_{\mathcal{I}_m} \mathfrak{L}_{(n)}(\xi)\, \tilde{\mathbb{U}}_{h_m}(x)\, dx = \int_{\mathcal{I}_m} \mathfrak{L}_{(n)}(\xi)\, \mathbb{U}_{h_m}(x)\, dx$$

$$m = j, j\pm 1 \quad \text{and} \quad n = 0, \ldots, p$$

$$\tilde{\mathbb{U}}_j^{(n)} = \frac{1+2n}{\Delta x_j} \int_{\mathcal{I}_j} \mathfrak{L}_{(n)}(\xi)\, \tilde{\mathbb{U}}_{h_m}(x)\, dx$$

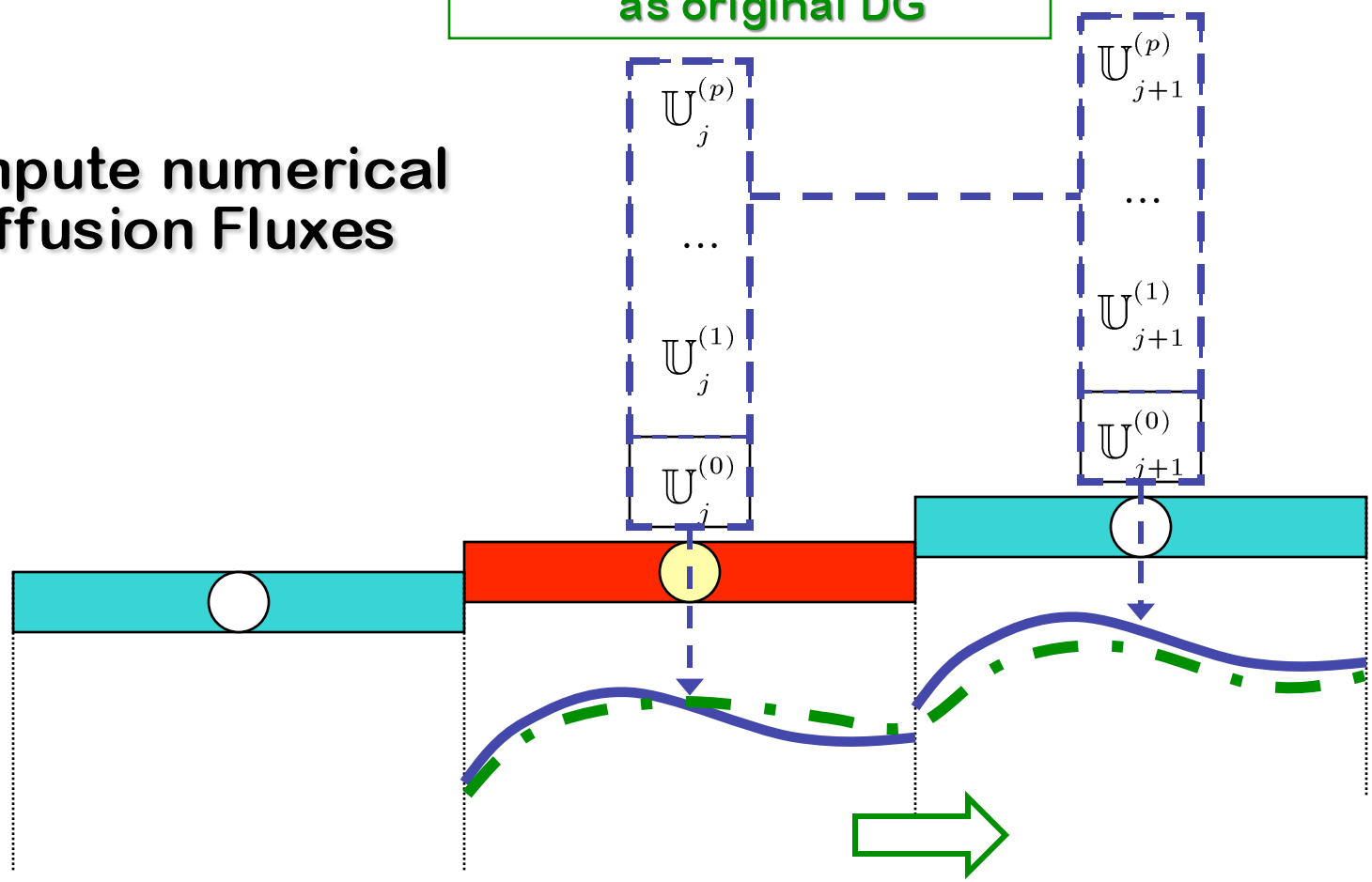
Recovery Discontinuous Galerkin (rDG)

- 0th-order DG (FV) → 3rd-order recovery (FV-PPM)
- Linear DG → 6th-order recovery
- Quadratic DG → 9th-order recovery
- Cubic DG → 12th-order recovery

Inter-Cell Recovery Discontinuous Galerkin (DG)
[Van Leer, 2005]

The same energy norms as original DG

Compute numerical Diffusion Fluxes



Method of Manufactured Solutions (MMS)

Verification

$$\partial_t \vec{\mathcal{U}} + \nabla \left(\vec{\mathcal{H}} + \vec{\mathcal{D}} \right) = \vec{\mathcal{R}}$$

"Manufacture" solution:

$$\vec{\mathcal{U}}_{\mathrm{MS}} \left(t, \mathbf{x} \right)$$

$$\vec{\mathcal{R}} = \vec{\mathcal{R}}_{\mathrm{O}} + \vec{\mathcal{R}}_{\mathrm{MS}}$$

Method of Manufactured Solutions (MMS)

Class Pipe1f_template
1-phase fluid (NS) in 1D pipe

γ-gas:

$$P = \rho i \left(\gamma - 1\right)$$

Compressible Navier-Stokes:

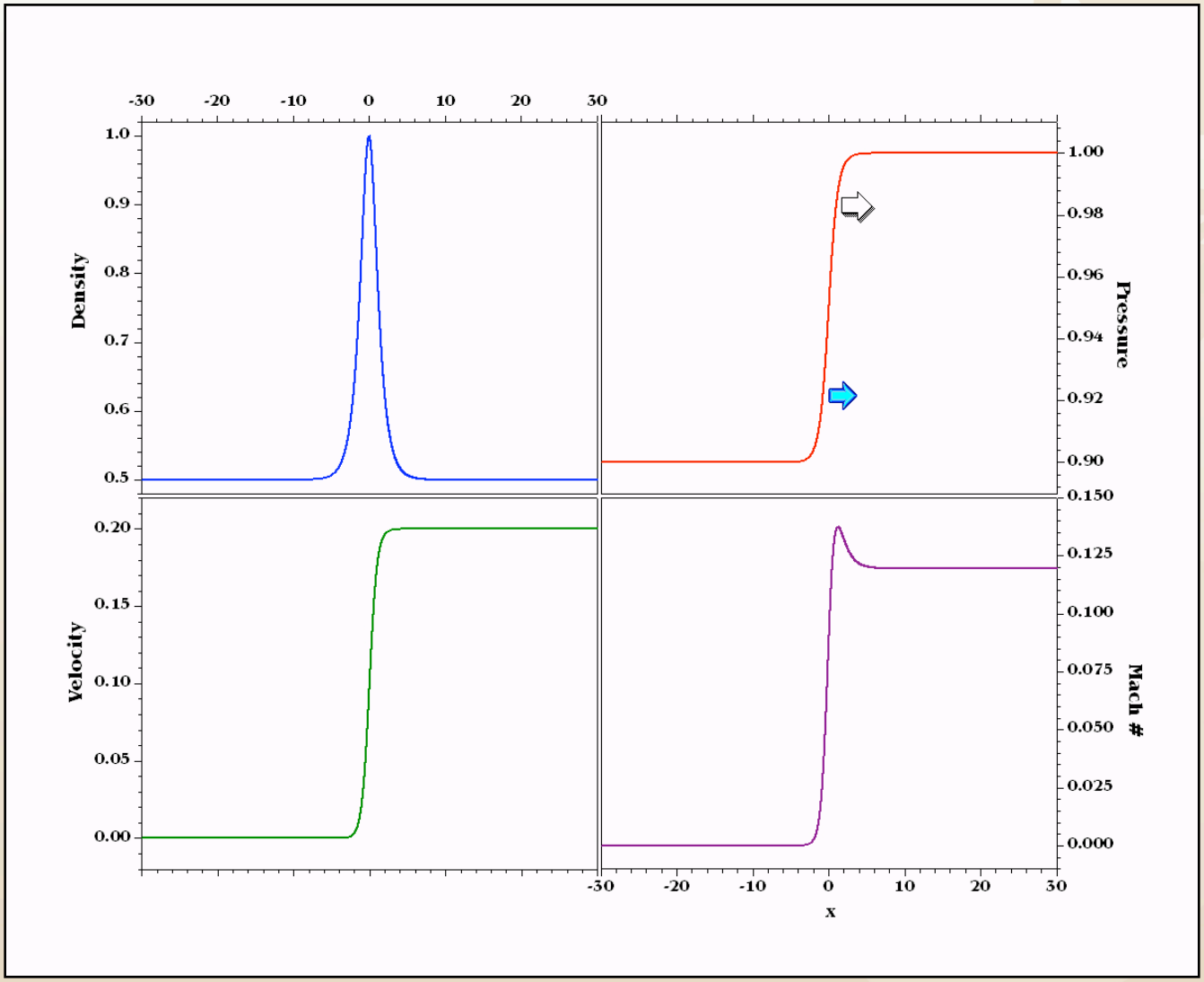
$$\vec{\mathcal{U}} = \begin{bmatrix} \rho \\ \rho u \\ \rho v \\ E \end{bmatrix}; \quad \vec{\mathcal{H}} = \begin{bmatrix} \cdot \rho \vec{u} \\ \cdot \rho \left(\vec{u} \otimes \vec{u}\right) + P \\ \cdot \vec{u} \left(E + P\right); \end{bmatrix};$$

$$\vec{\mathcal{D}} = - \begin{bmatrix} 0 \\ \cdot \underline{\tau} \\ \cdot \left(\underline{\tau} \cdot \vec{u} + \kappa \nabla T\right) \end{bmatrix}; \quad \vec{\mathcal{R}} = \begin{bmatrix} 0 \\ \rho \vec{g} \\ \rho \vec{g} \cdot \vec{u} \end{bmatrix}$$

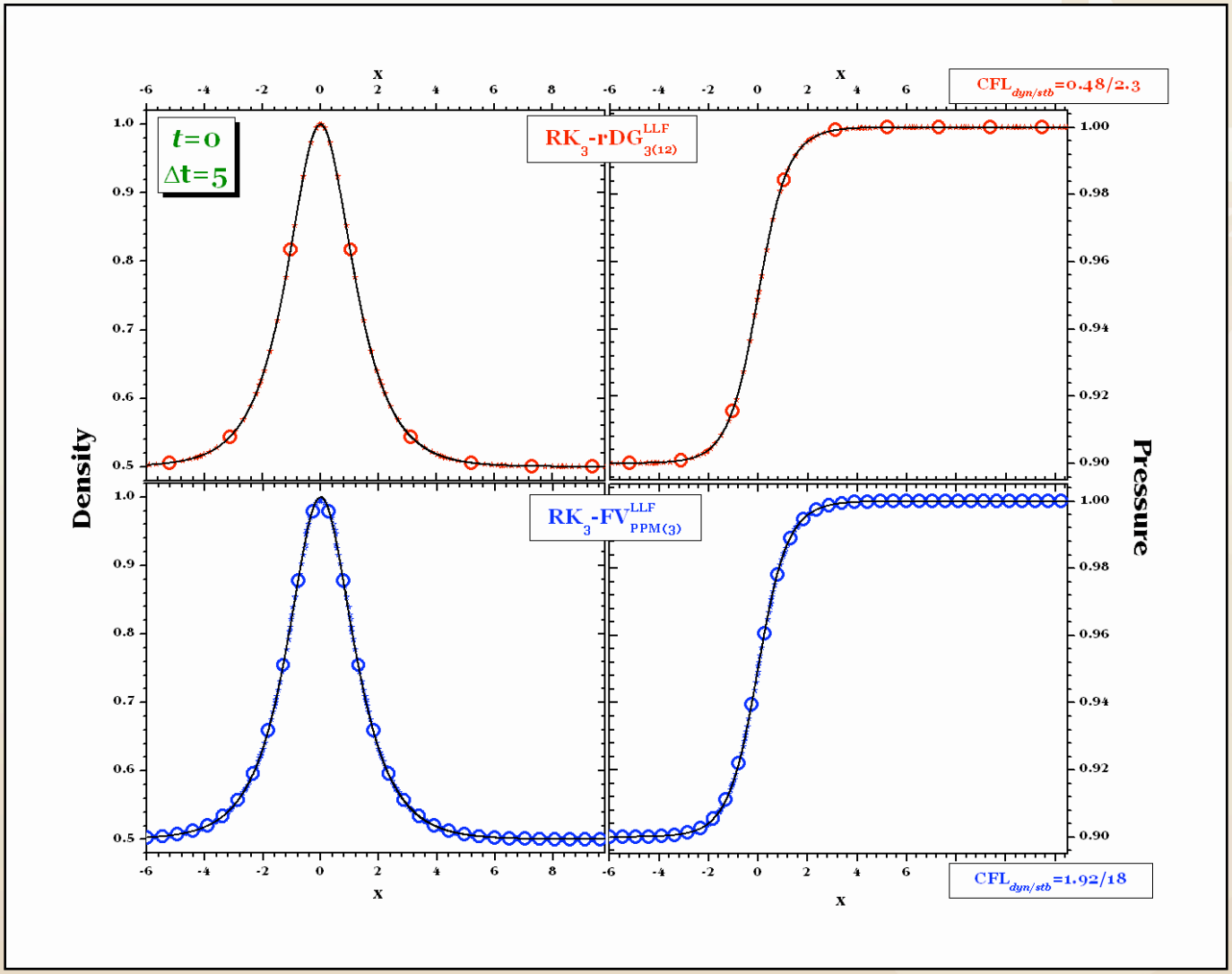
"Manufactured" solution:

$$\begin{aligned}
\rho\left(x,t\right) &= \rho_{\min} + \left(\rho_{\max} - \rho_{\min}\right) \operatorname{sech}\left(\frac{x - U_{\text{int}} t}{\delta}\right) \\
P\left(x,t\right) &= P_{\min} + \frac{1}{2}\left(\Delta P_0 \left[1 + \tanh\left(\frac{x - U_{\text{int}} t}{\delta}\right)\right] + \Delta P_1 \left[1 + \tanh\left(\frac{x - C_1 t}{\delta_1}\right)\right]\right) \\
u\left(x,t\right) &= U_{\min} + \left(U_{\text{int}} - U_{\min}\right)\left(1 + \tanh\left(\frac{x - U_{\text{int}} t}{\delta}\right)\right)
\end{aligned}$$

Manufactured solution: "Travelling Wave, γ-gas"

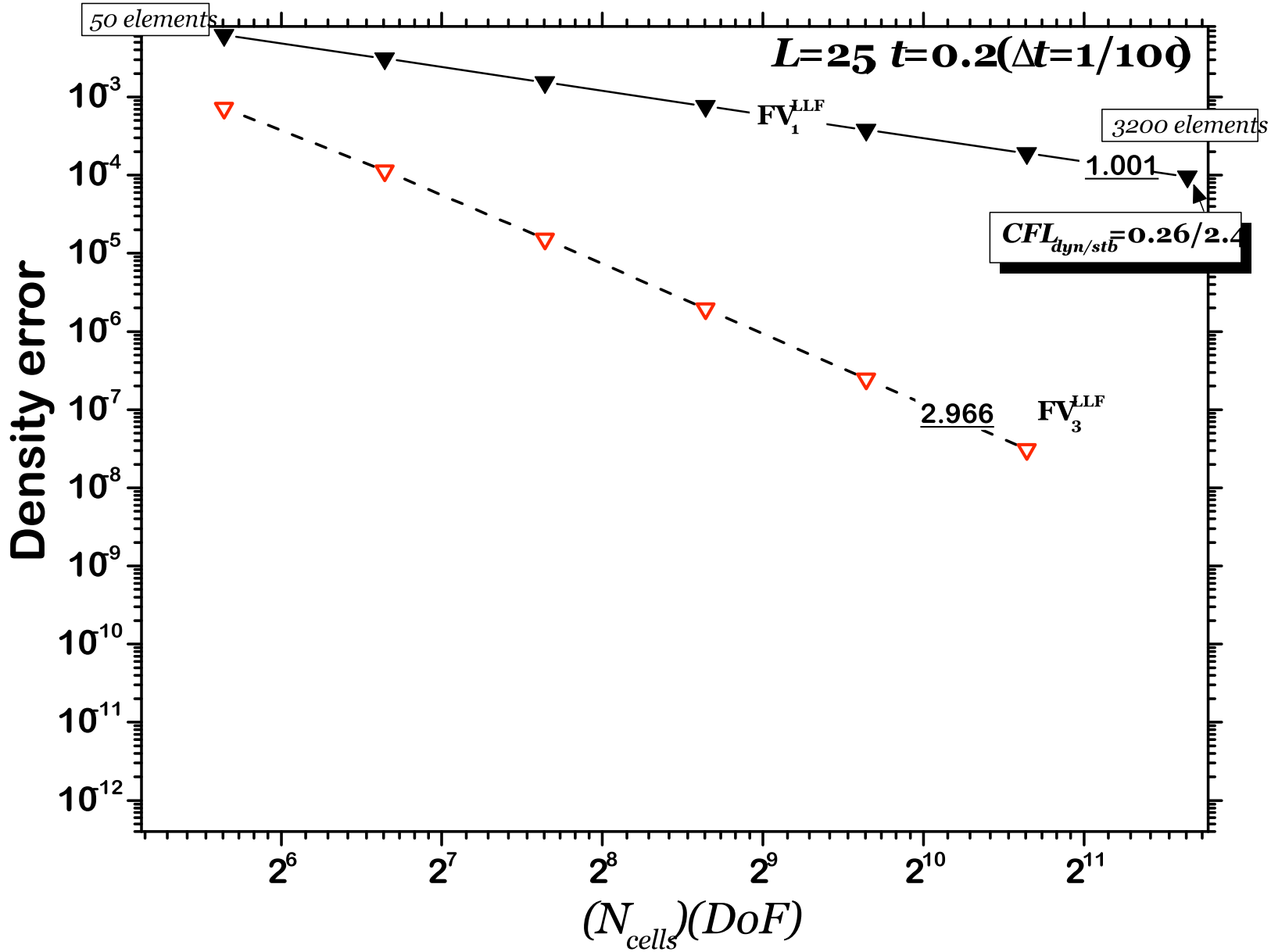


Manufactured solution: "Travelling Wave, γ-gas"



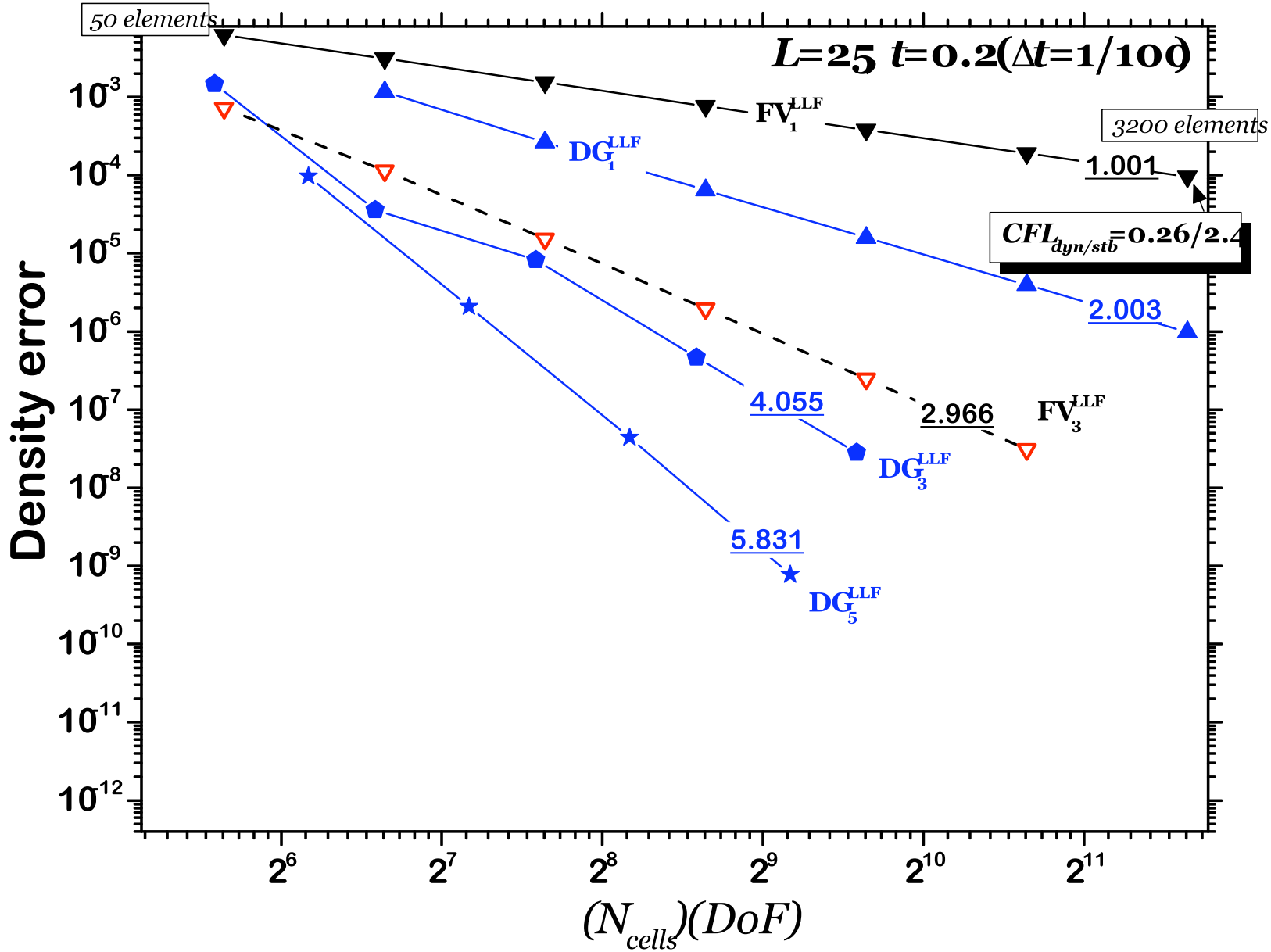
Convergence in space

Finite-Volume



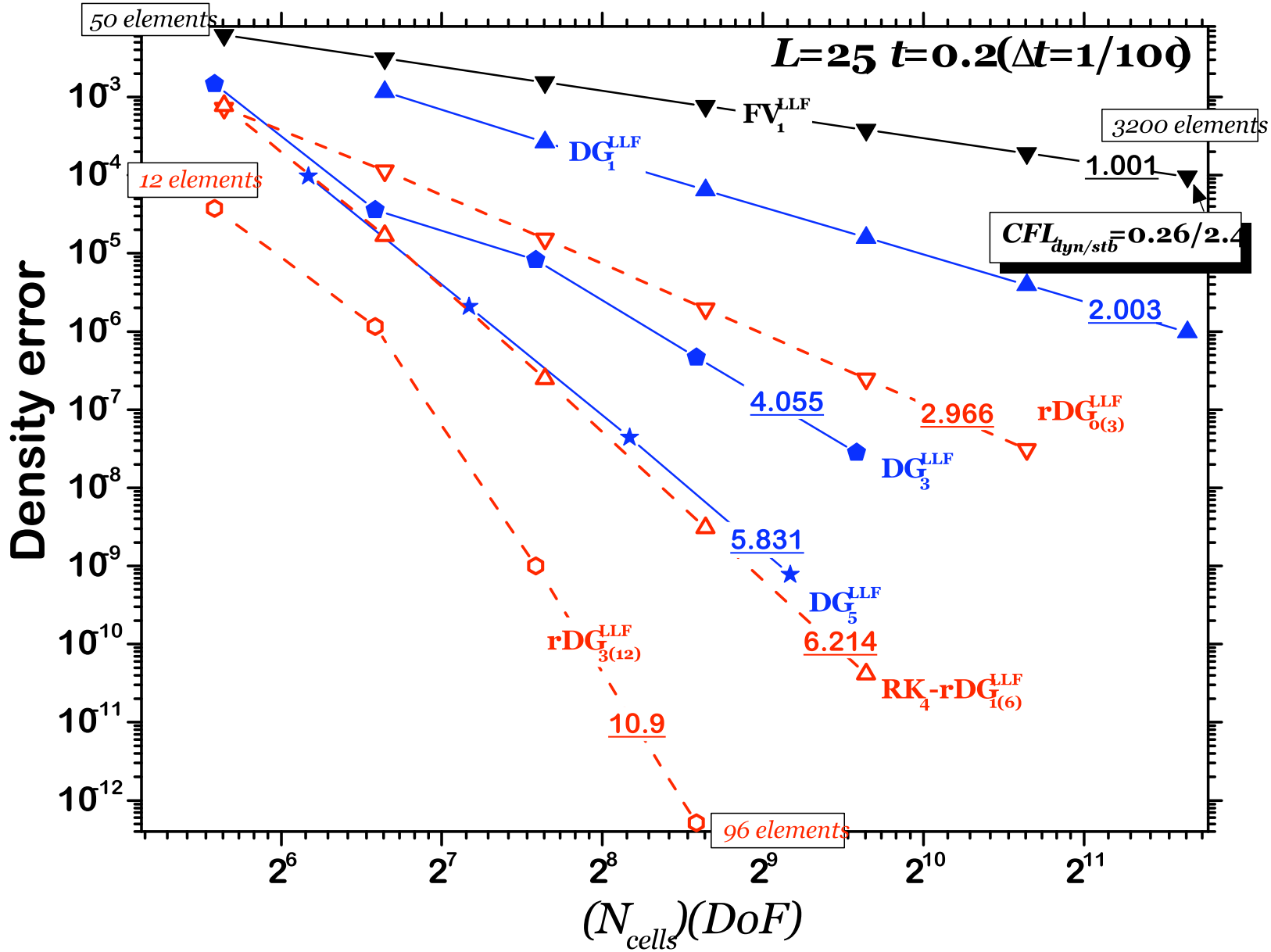
Convergence in space

Finite-Volume
Discontinuous Galerkin



Convergence in space

Finite-Volume
Discontinuous Galerkin
Recovery Discontinuous Galerkin



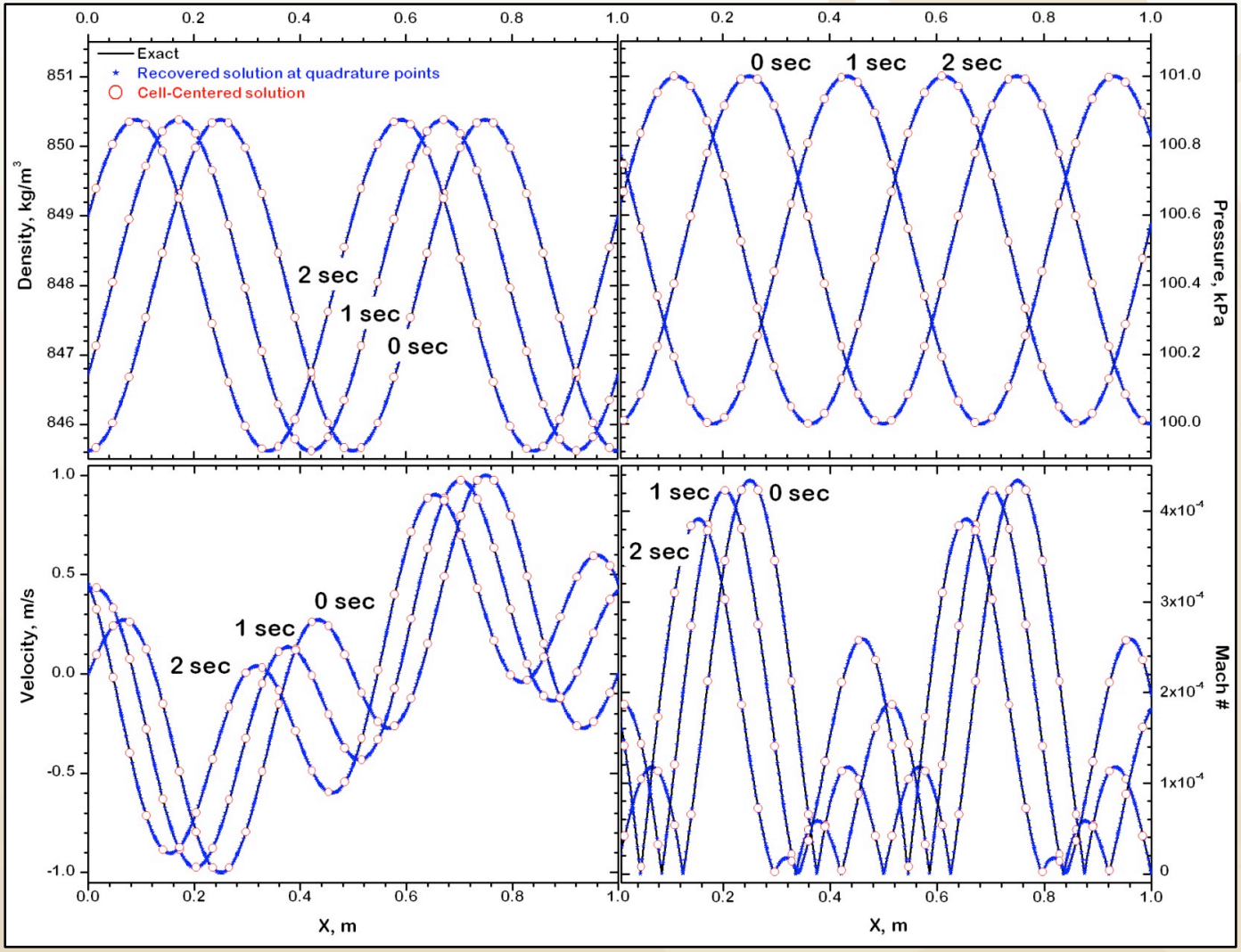
Stiff fluids (Low-Mach)

$$P = P_0 + (\rho - \rho_0)\left.\frac{\partial P}{\partial \rho}\right|_{(\rho_0, T_0)}^{\frac{Pa \cdot m^3}{kg}} + (T - T_0)\left.\frac{\partial P}{\partial T}\right|_{(\rho_0, T_0)}^{\frac{Pa}{K}}$$

Sodium
Water

$$\approx 10^6$$

Cubic rDG (12th-order)



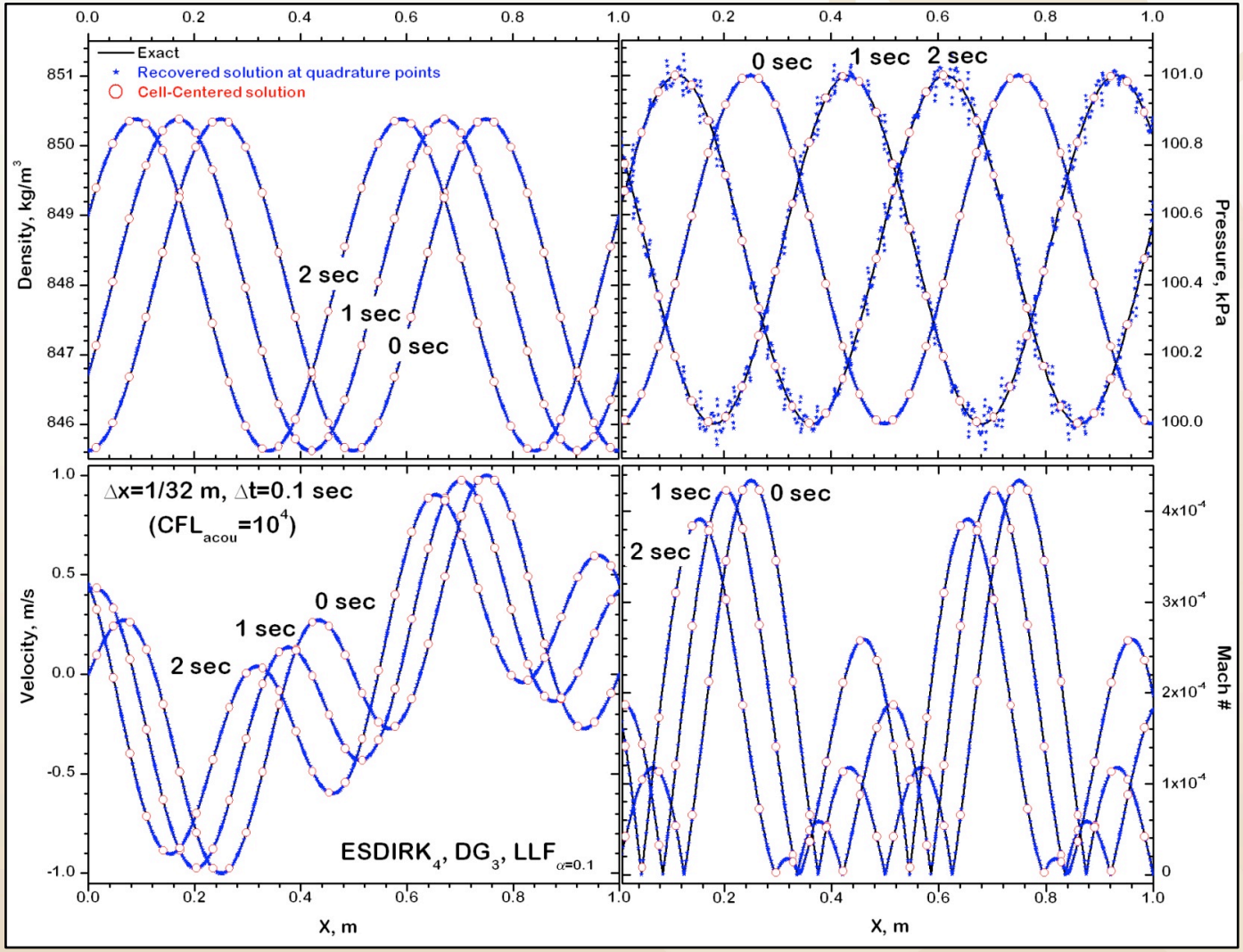
Stiff fluids
(Low-Mach)

$$P = P_0 + (\rho - \rho_0)\left.\frac{\partial P}{\partial \rho}\right|_{(\rho_0, T_0)} \boxed{\frac{Pa \cdot m^3}{kg}} + (T - T_0)\left.\frac{\partial P}{\partial T}\right|_{(\rho_0, T_0)} \boxed{\frac{Pa}{K}}$$

Sodium
Water

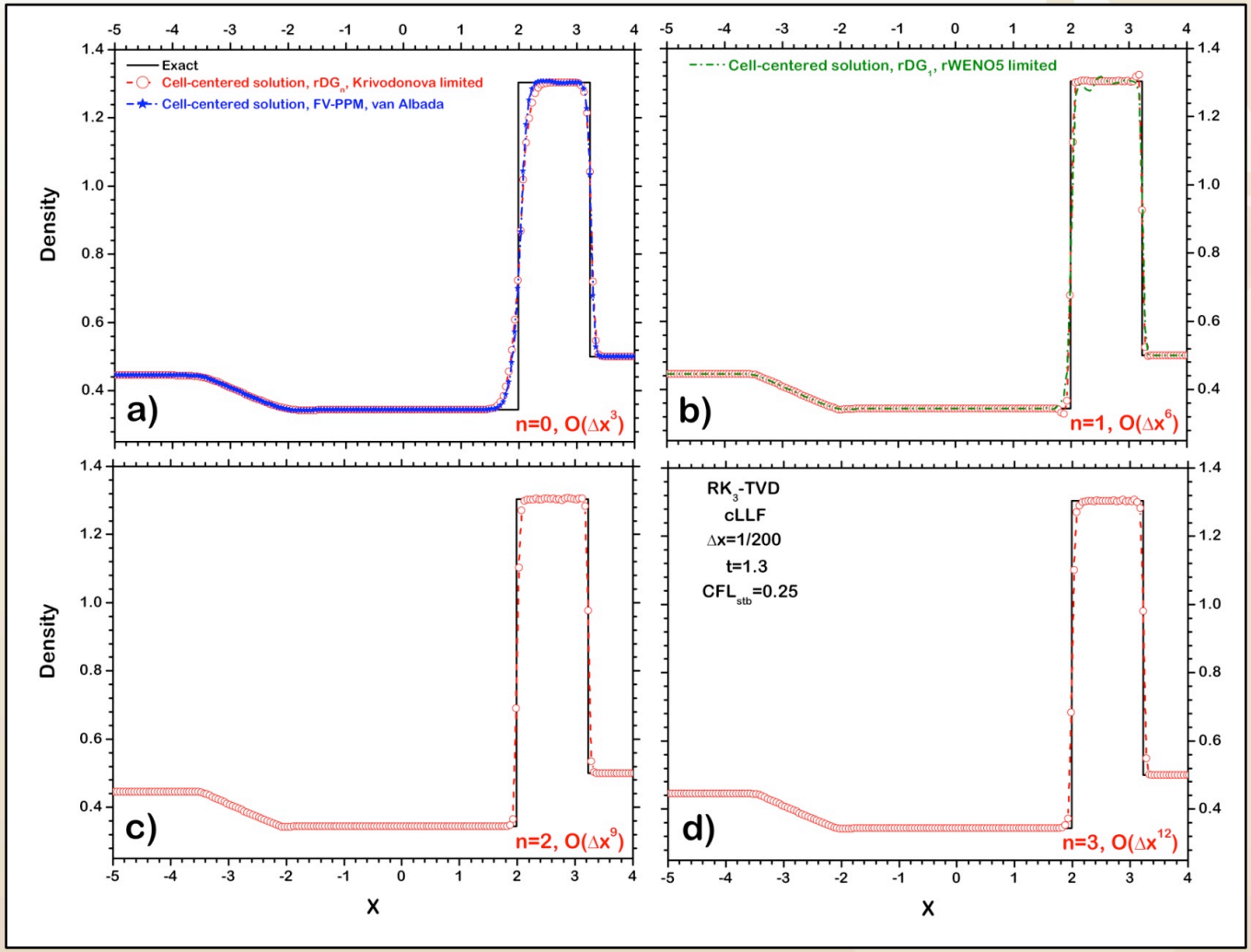
$$\approx 10^6$$

Cubic DG
(3rd-order)



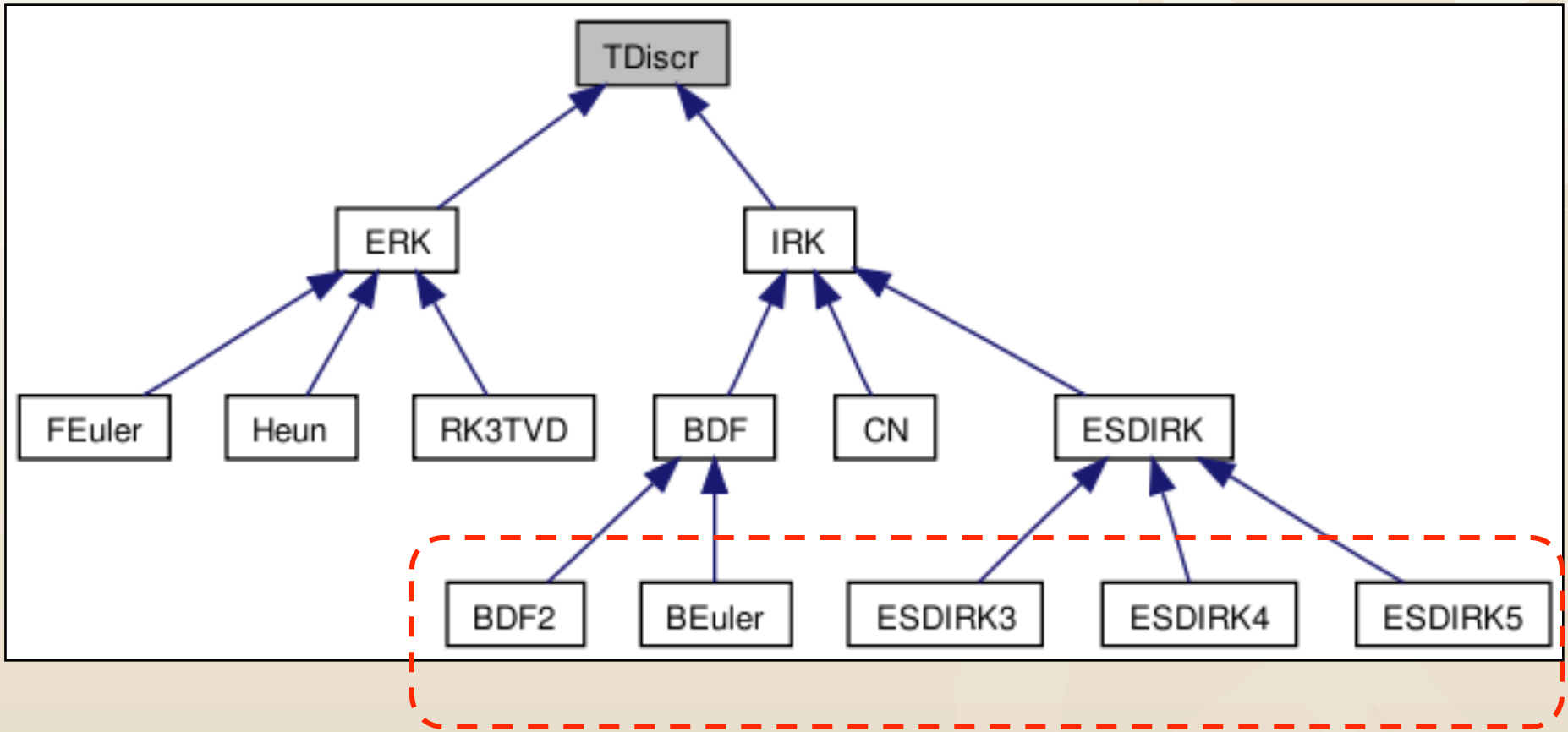
Shock Dynamics

Lax's problem (Mach ~1)



Time Discretization

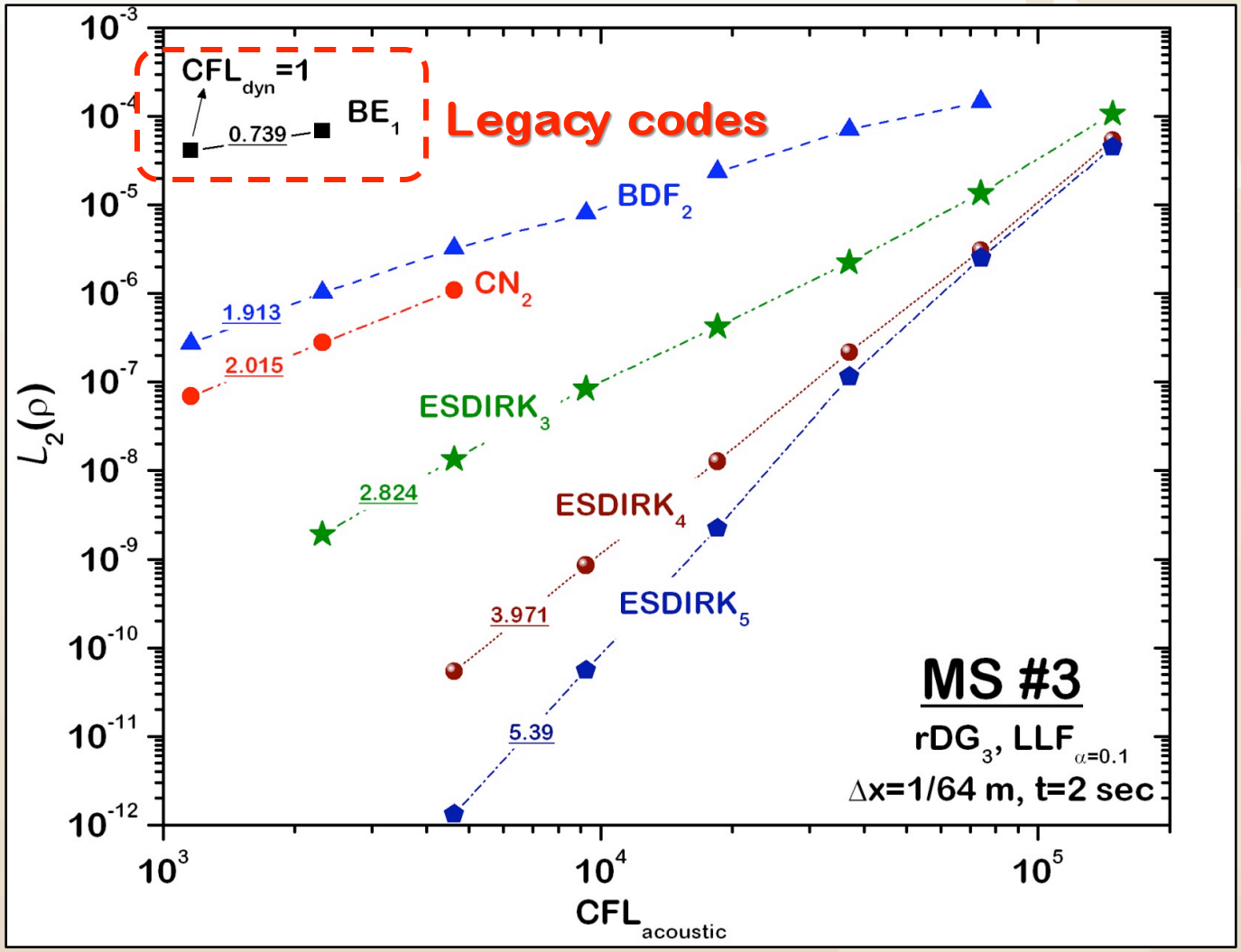
A-stability:
does not amplify any LHP-eigenvalues
L-stability:
Complete damping of all eigenvalues



L-stable

Convergence in time

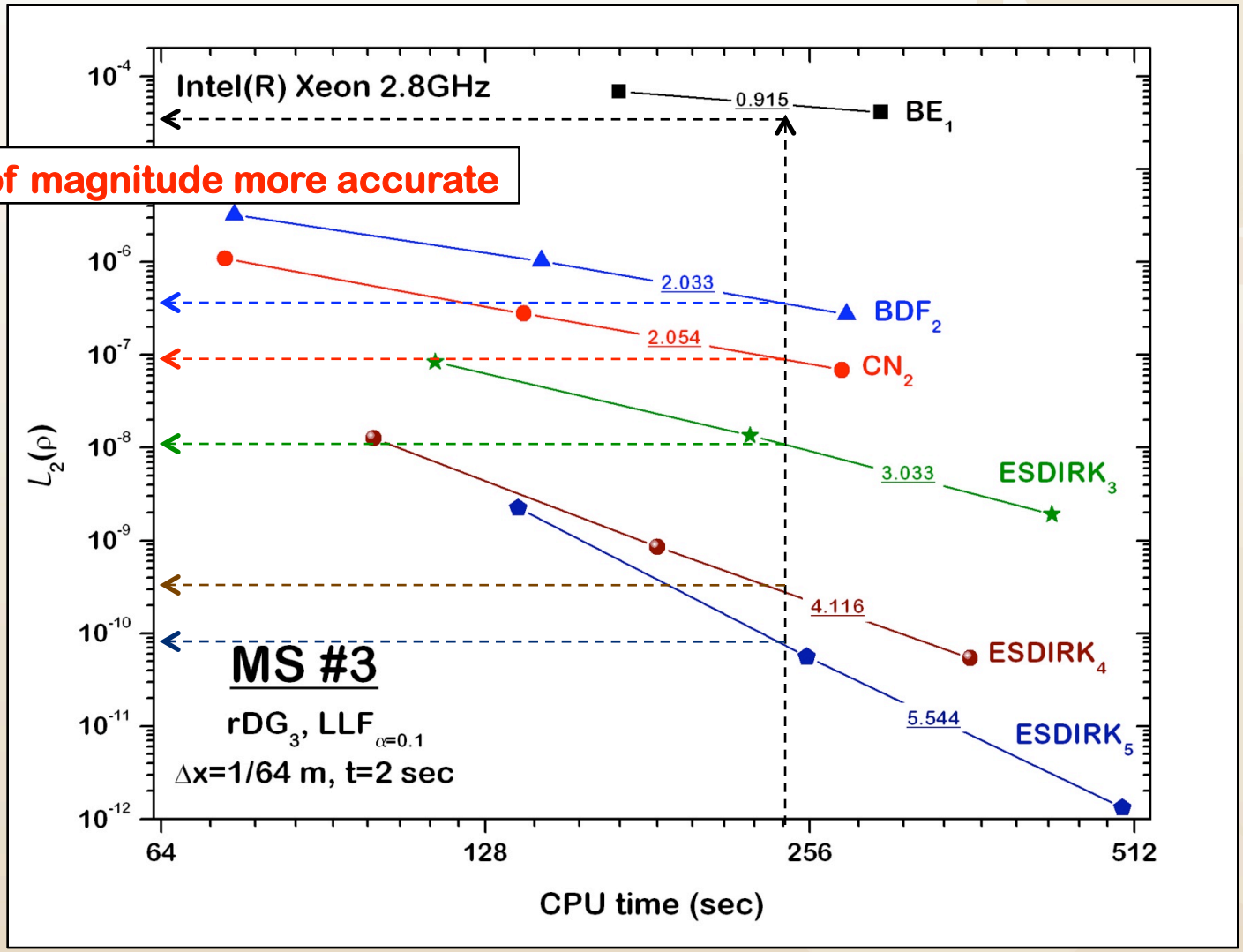
Stiff fluids (Low-Mach)



Efficacy of time discretization

Stiff fluids (Low-Mach)

Orders of magnitude more accurate



Conservative vs Non-Conservative Formulations

Conservative:

$$\partial_t \vec{\mathcal{U}} + \nabla\left(\vec{\mathcal{H}} + \vec{\mathcal{D}}\right) = \vec{\mathcal{R}}$$

$$\vec{\mathcal{U}} = \begin{bmatrix} \rho \\ \rho u \\ E \end{bmatrix}$$

$$\vec{\mathcal{U}} = \begin{bmatrix} \rho_i \alpha_i \\ \rho_i \alpha_i u_i \\ \rho_i \alpha_i e_i \end{bmatrix}, \quad i = f, g$$

Conservative variables

Non-conservative:

$$\mathbb{A}\partial_t \vec{\mathcal{V}} + \mathbb{B}\nabla\left(\vec{\mathcal{V}}\right) = \vec{\mathcal{R}}$$

$$\partial_t \vec{\mathcal{V}} + \mathbb{A}^{-1}\mathbb{B}\nabla\left(\vec{\mathcal{V}}\right) = \mathbb{A}^{-1}\vec{\mathcal{R}}$$

$$\vec{\mathcal{V}} = \begin{bmatrix} P \\ u \\ T \end{bmatrix}$$

$$\vec{\mathcal{V}} = \begin{bmatrix} P \\ \alpha_f \\ u_i \\ T_i \end{bmatrix}$$

Primitive variables

Conservative vs Non-Conservative Formulations

Conservative:

$$\partial_t \vec{\mathcal{U}} + \nabla\left(\vec{\mathcal{H}} + \vec{\mathcal{D}}\right) = \vec{\mathcal{R}}$$

Discretization is conservative to round-off

Non-conservative:

$$\mathbb{A}\partial_t \vec{\mathcal{V}} + \mathbb{B}\nabla\left(\vec{\mathcal{V}}\right) = \vec{\mathcal{R}}$$

Impossible to discretisize conserving to round-off

Bad choice for shock dynamics!

Legacy codes

Fast, short transients, LOCA-type

New designs:

Passive safety systems, long transients

(Errors accumulate → you better conserve mass, momentum and energy)

Downsides of the Conservative Formulation

- Two-phase equations are non-conservative

$$\partial_t \begin{bmatrix} \rho_i \alpha_i \\ \rho_i \alpha_i u_i \\ \rho_i \alpha_i e_i \end{bmatrix} + \partial_x \begin{bmatrix} \rho_i \alpha_i u_i \\ \rho_i \alpha_i u_i^2 + \alpha_i P \\ u_i \left(\rho_i \alpha_i e_i + \alpha_i P \right) \end{bmatrix} + \boxed{\begin{bmatrix} 0 \\ P \partial_x \alpha_i \\ u_i P \partial_x \alpha_i \end{bmatrix}} = \text{Closures}$$

Interfacial momentum/energy exchange

Downsides of the Conservative Formulation

- Two-phase equations are non-conservative
- Stiff fluids at low-Mach limit

$$P = P_0 + (\rho - \rho_0)\left.\frac{\partial P}{\partial \rho}\right|_{(\rho_0, T_0)} + (T - T_0)\left.\frac{\partial P}{\partial T}\right|_{(\rho_0, T_0)}$$

$$\approx const$$

Sodium
Water

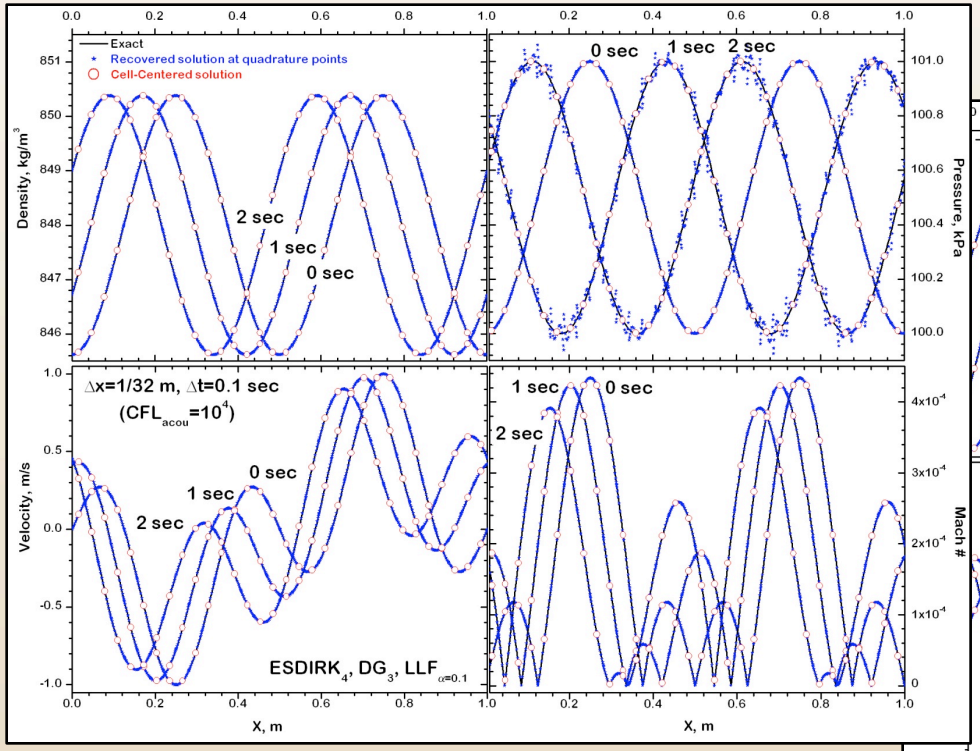
$$\approx 10^6$$

Small error in density

Huge error in pressure

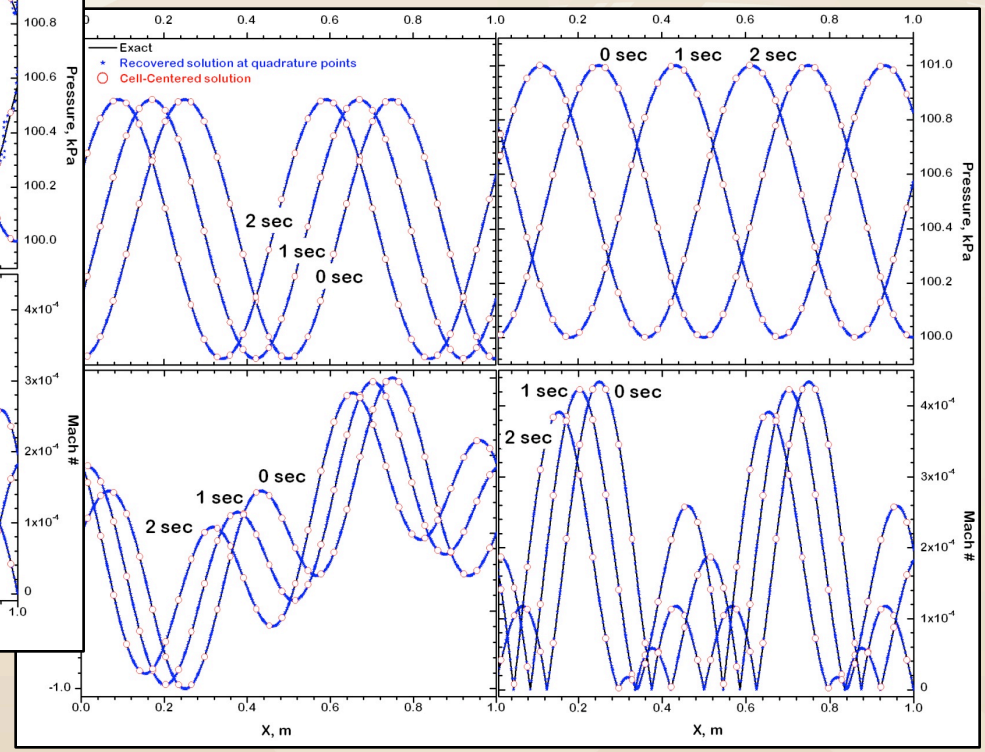
Downsides of the Conservative Formulation

- Two-phase equations are non-conservative
- Stiff fluids at low-Mach limit



3rd-order

12th-order



All-speed Solvers

Low-Mach, <10^{-1} → **High-Mach, >1**

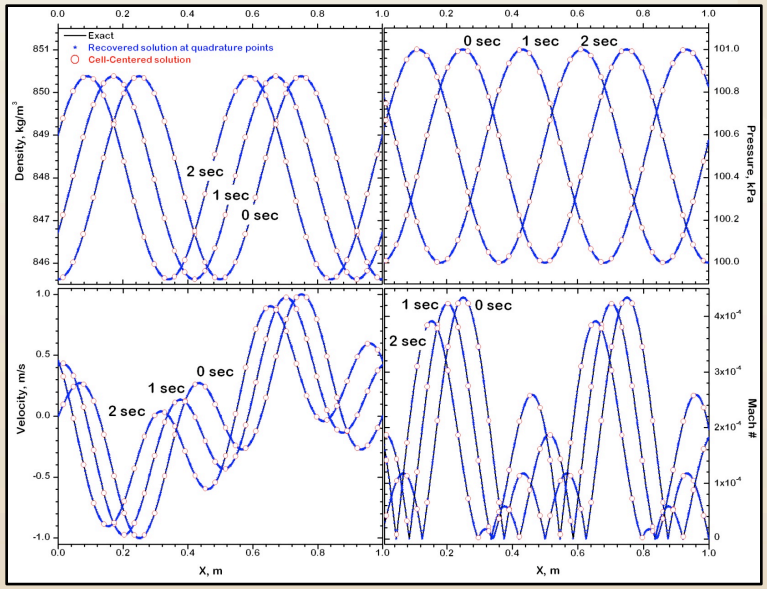
Mach #

Incompressible
ICE, Primitive-variables

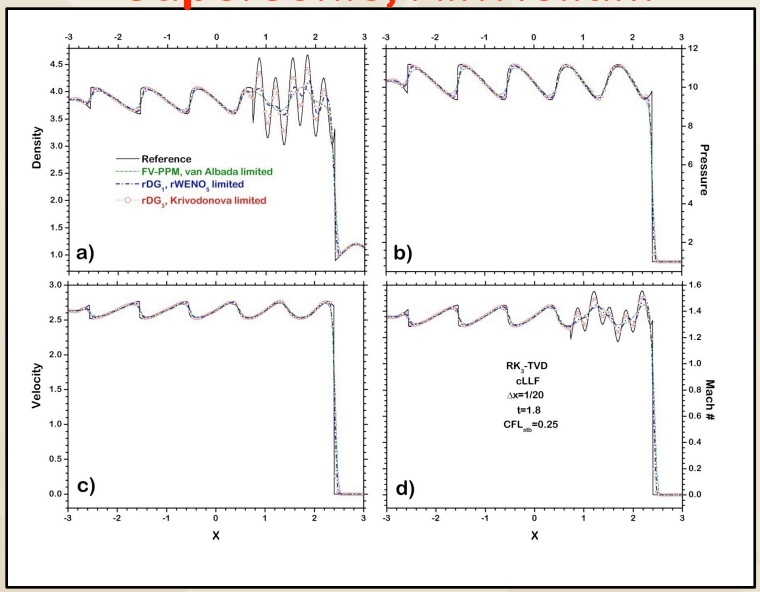
Conservative-variables

High-fidelity spatio-temporal discretization in conservative-variables

M=10^{-4}, Sodium



Supersonic, Air/Helium



Linear Algebra

Jacobian-free Newton Krylov (JFNK)

Each RK stage is a non-linear solve

$$\vec{\mathcal{U}}^{[k]} = \vec{\mathcal{U}}^{[n]} + \Delta t \sum_{r=1}^{k} \mathsf{a}_{kr} \vec{\mathcal{F}}\left(\vec{\mathcal{U}}^{[r]}\right), \quad k = 1, \ldots, s-1$$

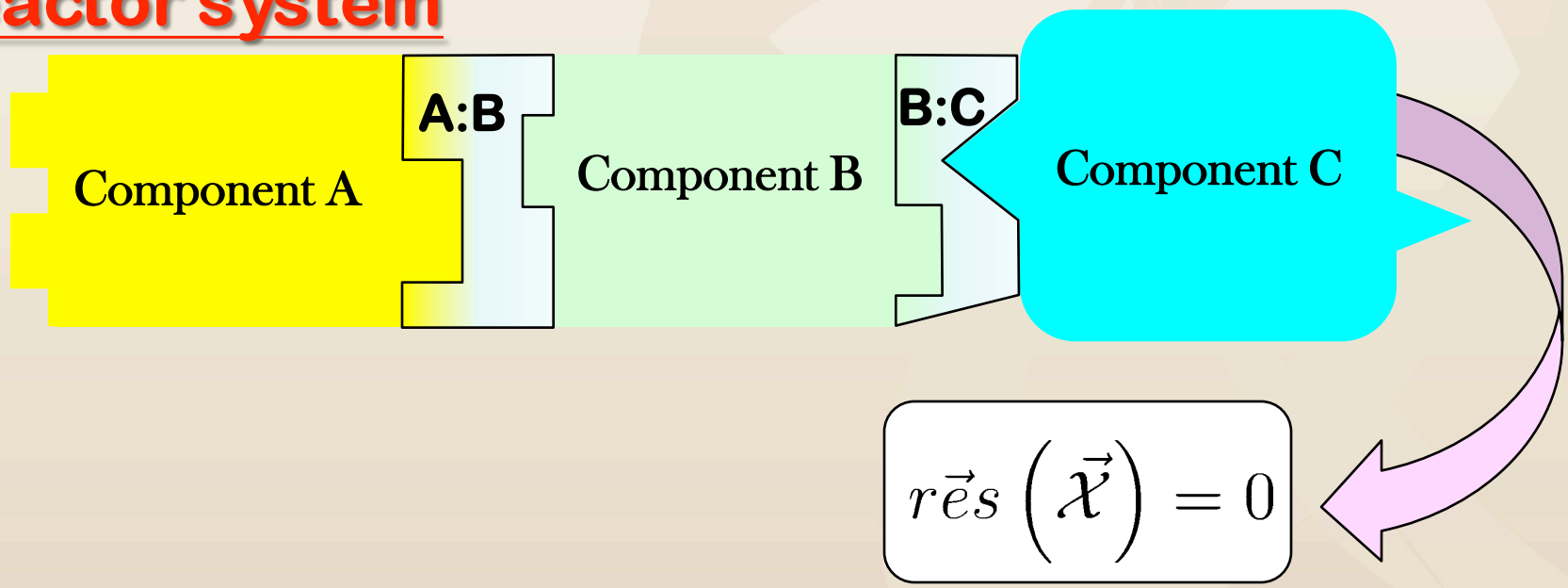
$$\vec{\mathcal{U}}^{[n+1]} = \vec{\mathcal{U}}^{[n]} + \Delta t \left(\mathsf{b}_0 \vec{\mathcal{F}}\left(\vec{\mathcal{U}}^{[n]}\right) + \sum_{r=1}^{s} \mathsf{b}_r \vec{\mathcal{F}}\left(\vec{\mathcal{U}}^{[r]}\right) \right)$$

Jacobian-free Newton Krylov (JFNK)

Each RK stage is a non-linear solve

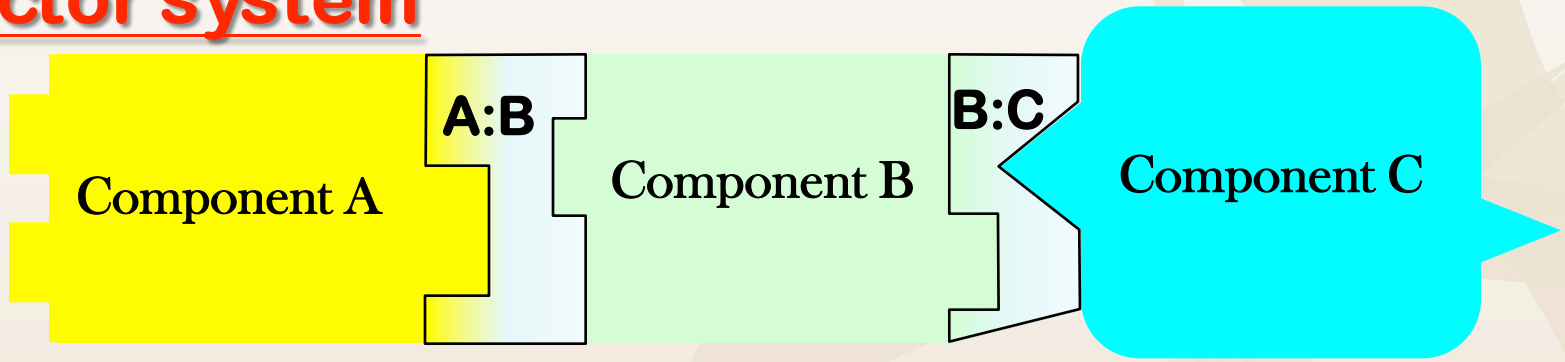
$$res_{u_{\mathrm{el}}^{(k)}} = \mathcal{U}_{\mathrm{el}}^{(k)[rk]} - \mathcal{U}_{\mathrm{el}}^{(k)[n]} - \Delta t \sum_{r=1}^{rk} \mathsf{a}_{rk,r} \mathcal{F}_{\mathrm{el}}^{(k)} \left(\vec{\mathcal{X}}^{[r]} \right)$$

Reactor system



Jacobian-free Newton Krylov (JFNK)

Reactor system



Vector of unknowns:

$$X = \begin{pmatrix} x_A \\ x_B \\ x_C \end{pmatrix}$$

Vector of residuals:

$$res = \begin{pmatrix} res_A \\ res_B \\ res_C \end{pmatrix}$$

Jacobian-free **Newton** Krylov (JFNK)

$$\mathbb{J}_{i,j} \equiv \frac{\partial res_i}{\partial \mathcal{X}_j}$$

Linear solve:

$$\mathbb{J}^{\mathrm{it}} \delta \vec{\mathcal{X}}^{\mathrm{it}} = -\vec{res}\left(\vec{\mathcal{X}}^{\mathrm{it}}\right)$$

Start Newton iteration

$$\vec{\mathcal{X}}^{\mathrm{it}+1} = \vec{\mathcal{X}}^{\mathrm{it}} + \delta \vec{\mathcal{X}}^{\mathrm{it}}$$

IF $\left\| \vec{res}\left(\vec{\mathcal{X}}^{\mathrm{it}}\right) \right\|_2 < \mathrm{tol}_{\mathrm{N}} \left\| \vec{res}\left(\vec{\mathcal{X}}^{0}\right) \right\|_2$ **ELSE**

End Newton iteration

Jacobian-free Newton Krylov (JFNK)

Linear solve

$$\mathbb{J}^{\mathrm{it}} \delta \vec{\mathcal{X}}^{\mathrm{it}} = -\vec{res}\left(\vec{\mathcal{X}}^{\mathrm{it}}\right)$$

$$\mathbb{J}_{i,j} \equiv \frac{\partial res_i}{\partial \mathcal{X}_j}$$ **(generally, non-symmetric)**

$$\delta \vec{\mathcal{X}}_m^{\mathrm{it}} = \delta \vec{\mathcal{X}}_0^{\mathrm{it}} + \mathfrak{k}_0 \mathbf{r}_0 + \mathfrak{k}_1 \mathbb{J}\mathbf{r}_0 + \mathfrak{k}_2 \mathbb{J}^2 \mathbf{r}_0 + \ldots + \mathfrak{k}_m \mathbb{J}^m \mathbf{r}_0$$

$$\mathbb{K}_j = \mathrm{span}\left(\mathbf{r}_0, \mathbb{J}\mathbf{r}_0, \mathbb{J}^2\mathbf{r}_0, \ldots, \mathbb{J}^{j-1}\mathbf{r}_0\right)$$

Jacobian-free Newton Krylov (JFNK)

GMRES

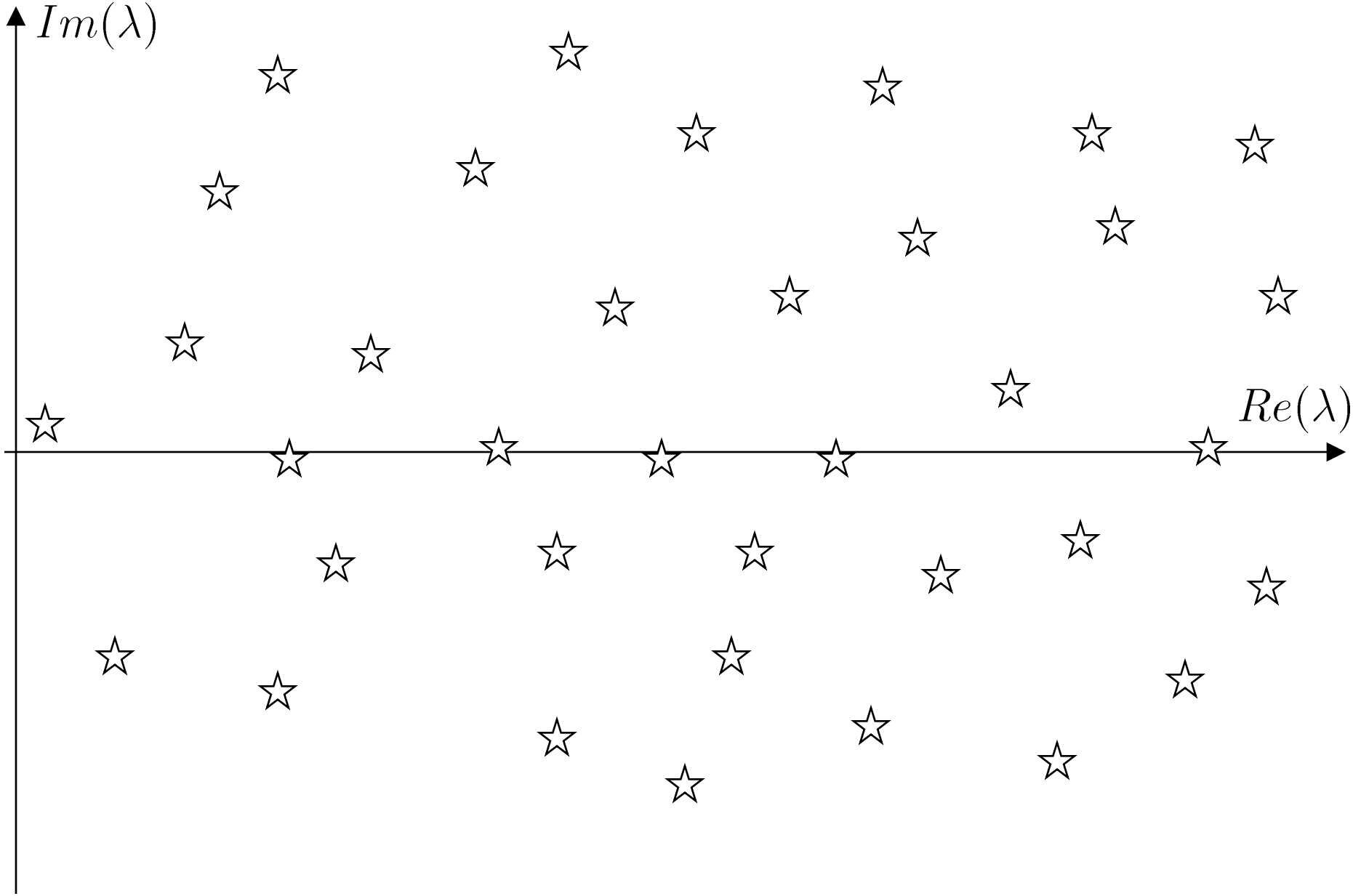
Fréchet derivatives:

$$\mathbb{J}\vec{\kappa} \approx \frac{r\vec{e}s\left(\vec{\mathcal{X}} + \varepsilon\vec{\kappa}\right) - r\vec{e}s\left(\vec{\mathcal{X}}\right)}{\varepsilon}$$

$$\varepsilon = \frac{\sum_{i=1}^{N} b\mathcal{X}_i}{N\left|\left|\vec{\kappa}\right|\right|_2}$$

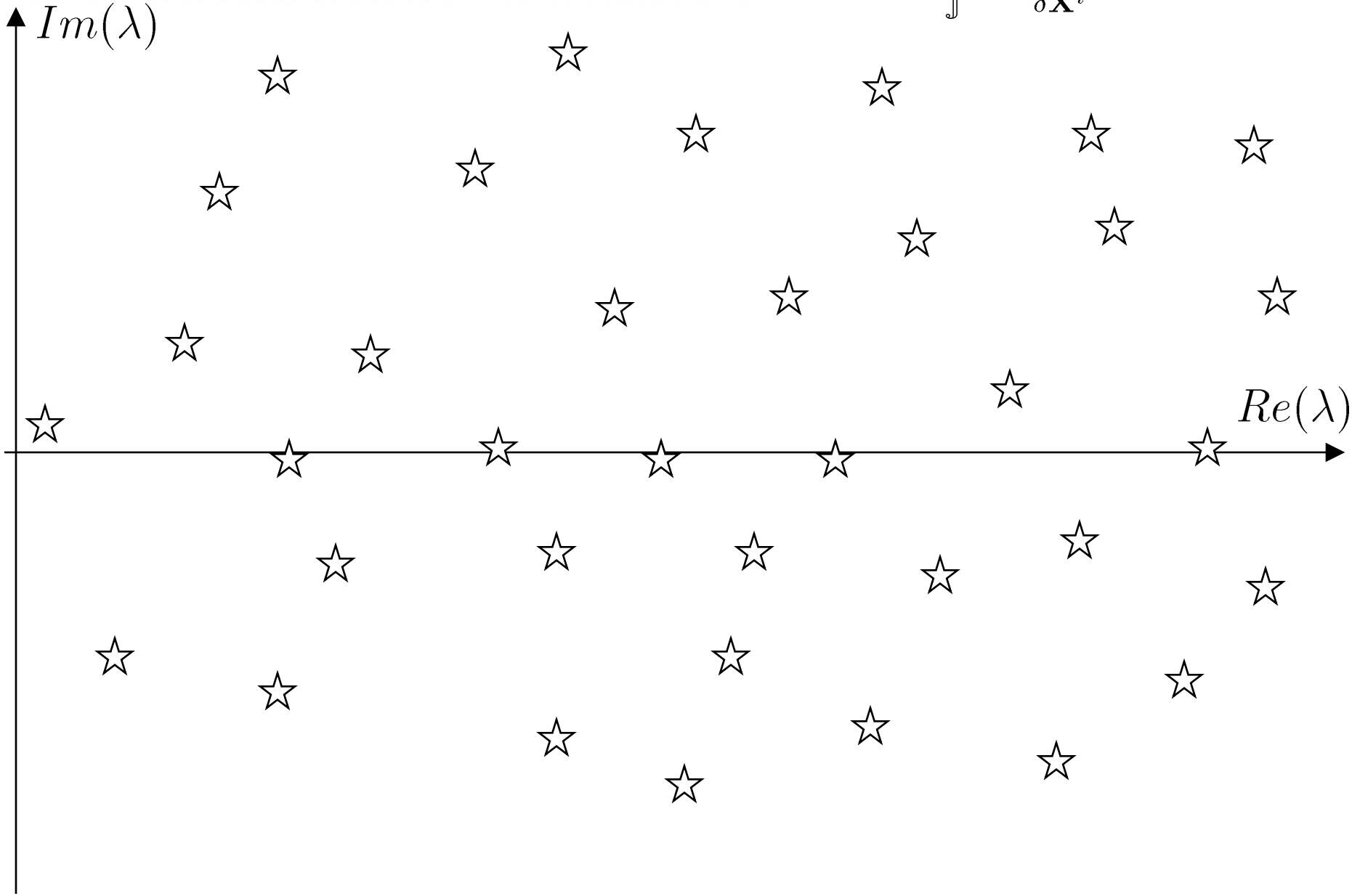
Preconditioning of GMRES

$$\mathbb{J}^i \delta \mathbf{X}^i = -\mathbf{res}\left(\mathbf{X}^i\right)$$



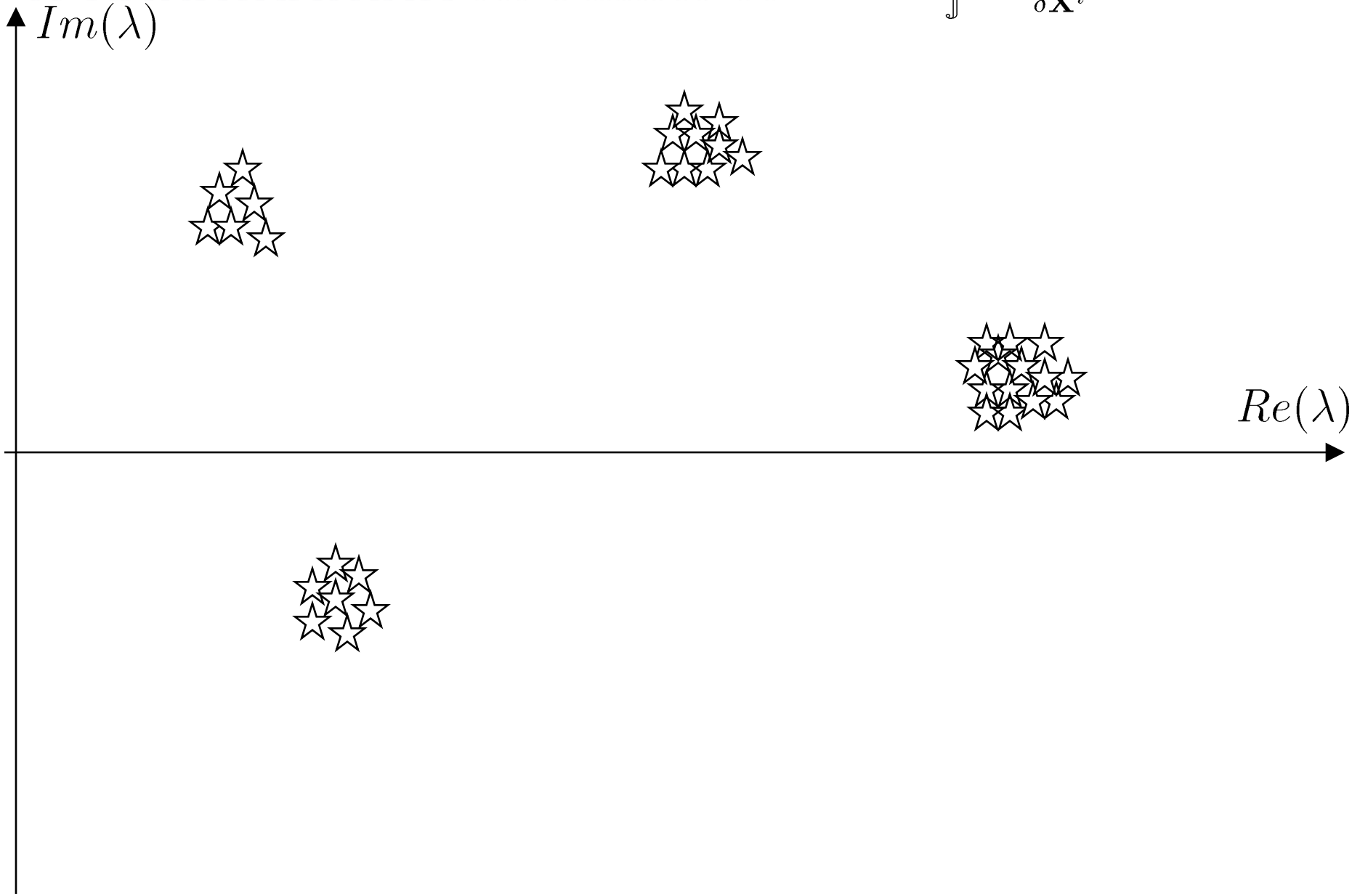
Preconditioning of GMRES

$$\underbrace{\mathbb{J}^i\mathbb{P}^{-1}}_{\hat{\mathbb{J}}}\underbrace{\mathbb{P}\delta\mathbf{X}^i}_{\delta\hat{\mathbf{X}}^i} = -\mathbf{res}\left(\mathbf{X}^i\right)$$



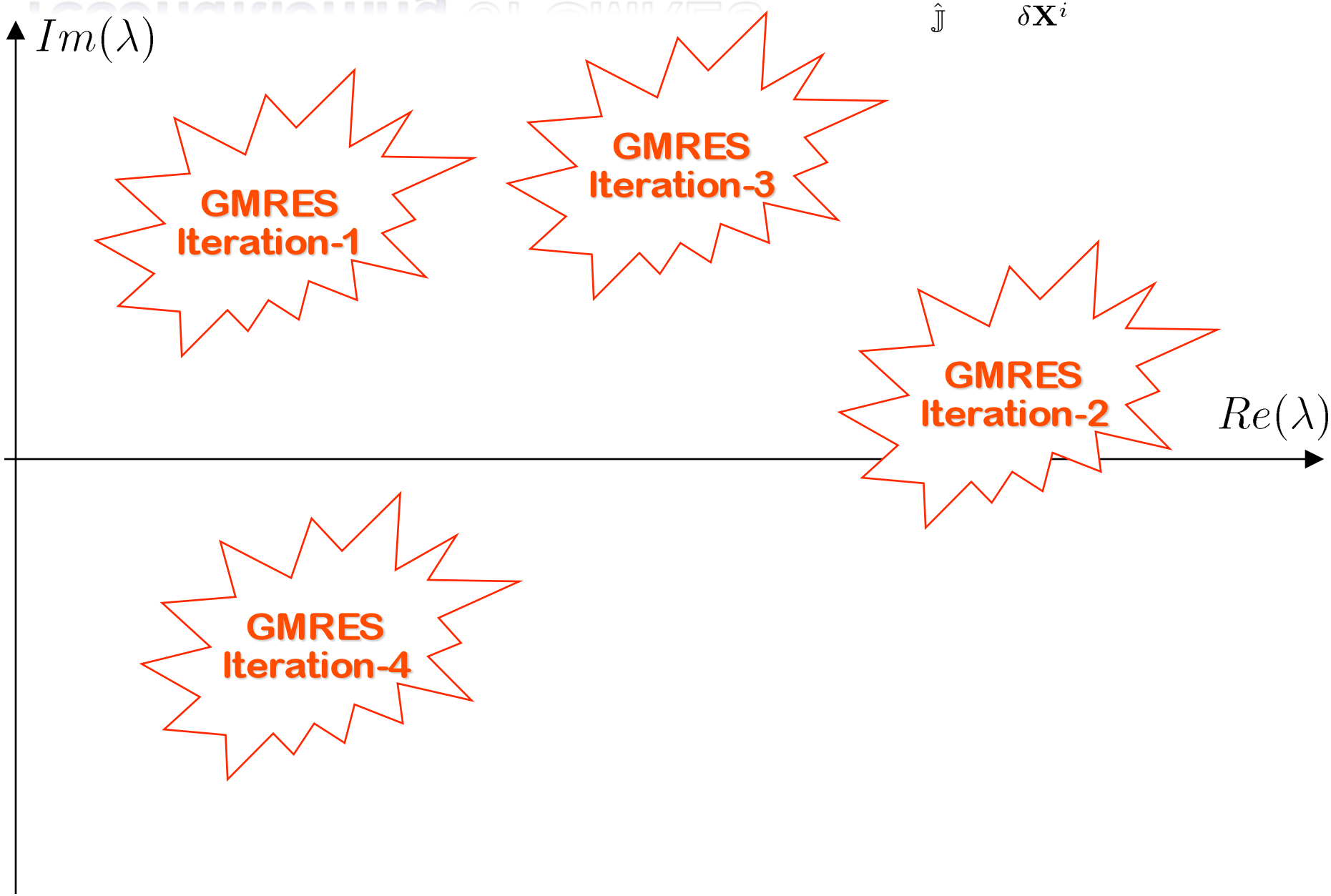
Preconditioning of GMRES

$$\underbrace{\mathbb{J}^i\mathbb{P}^{-1}}_{\hat{\mathbb{J}}}\underbrace{\mathbb{P}\delta\mathbf{X}^i}_{\delta\hat{\mathbf{X}}^i} = -\mathbf{res}\left(\mathbf{X}^i\right)$$

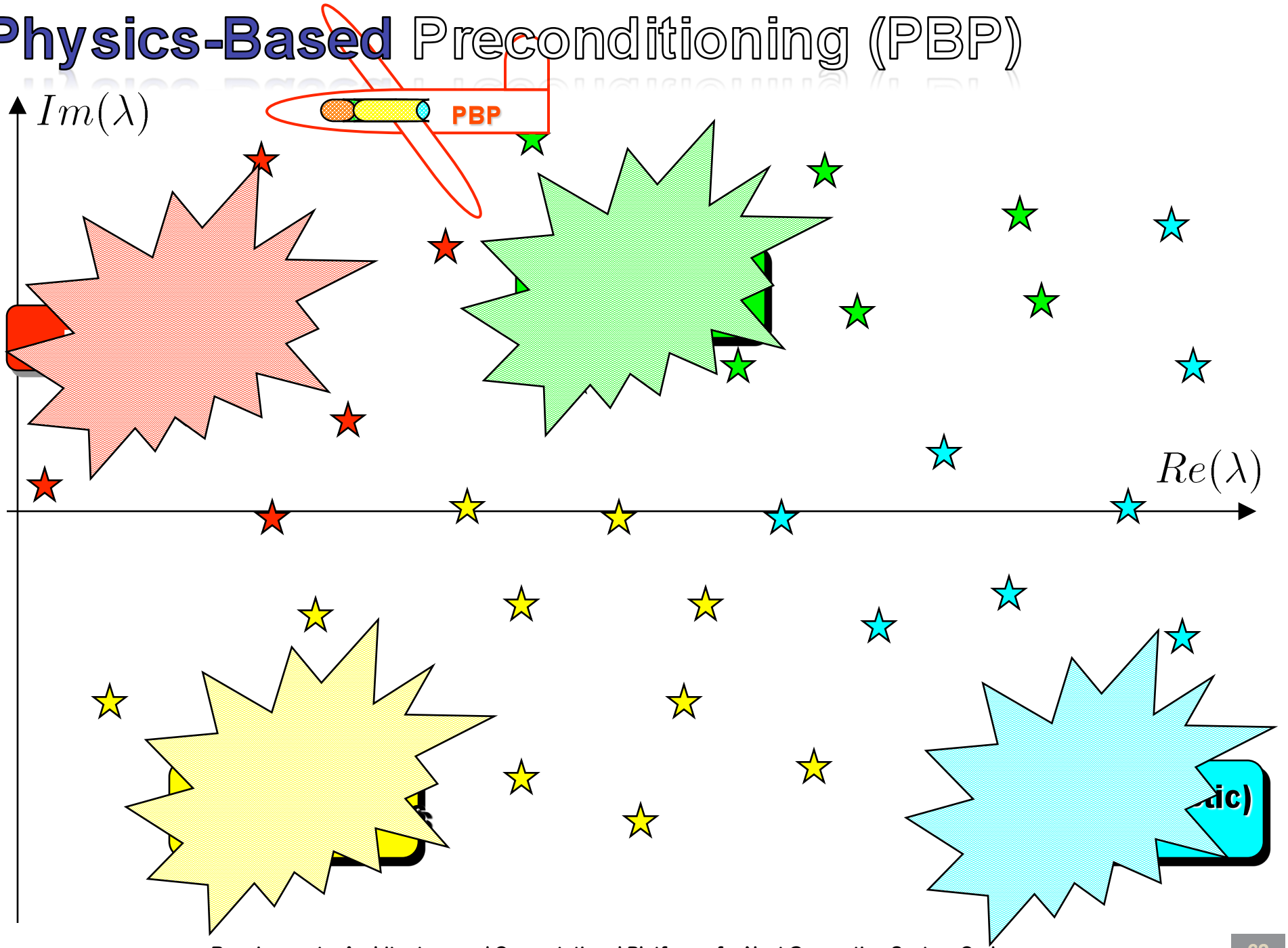


Preconditioning of GMRES

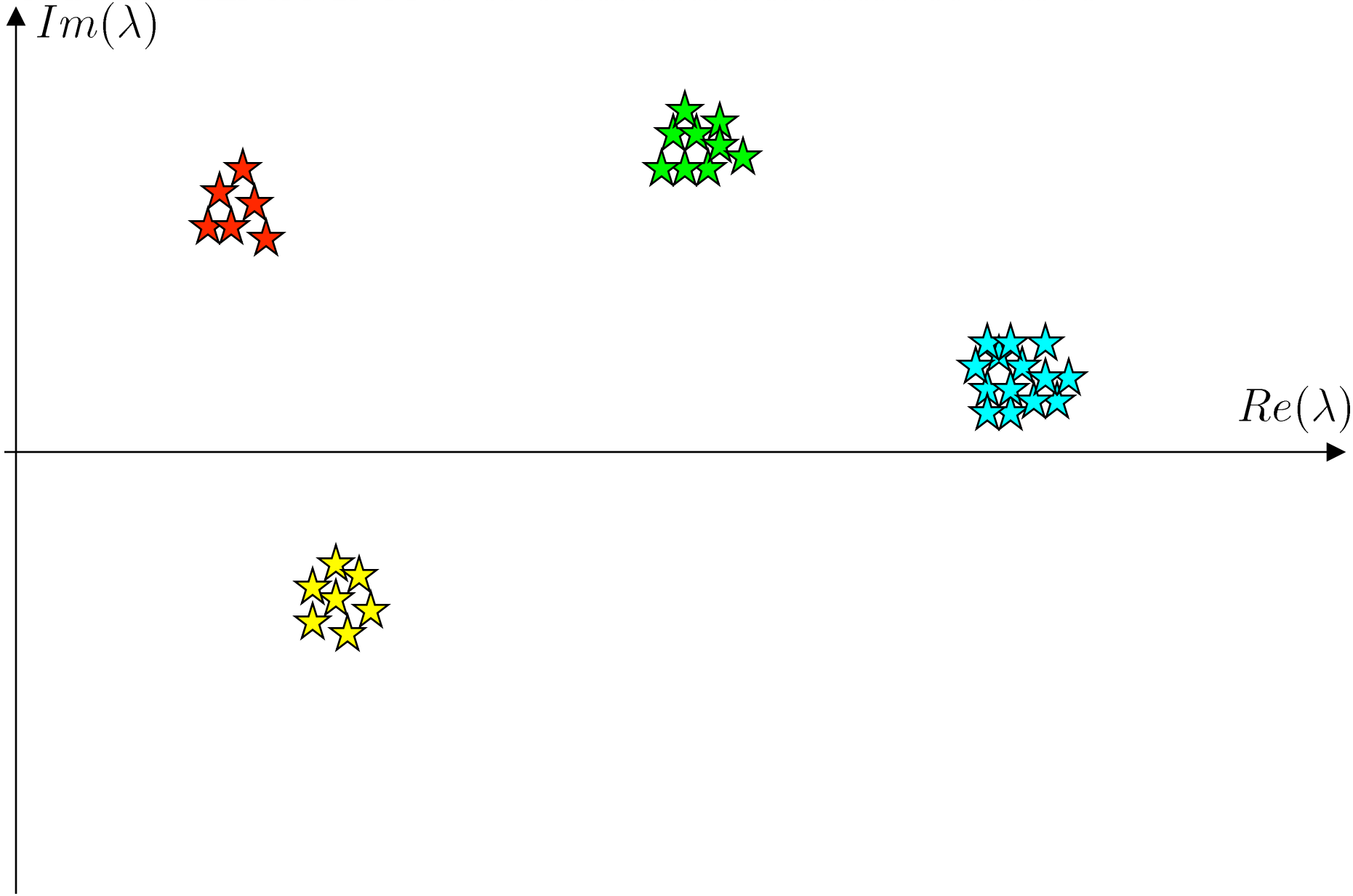
$$\underbrace{\mathbb{J}^i\mathbb{P}^{-1}}_{\hat{\mathbb{J}}}\underbrace{\mathbb{P}\delta\mathbf{X}^i}_{\delta\hat{\mathbf{X}}^i} = -\mathbf{res}\left(\mathbf{X}^i\right)$$



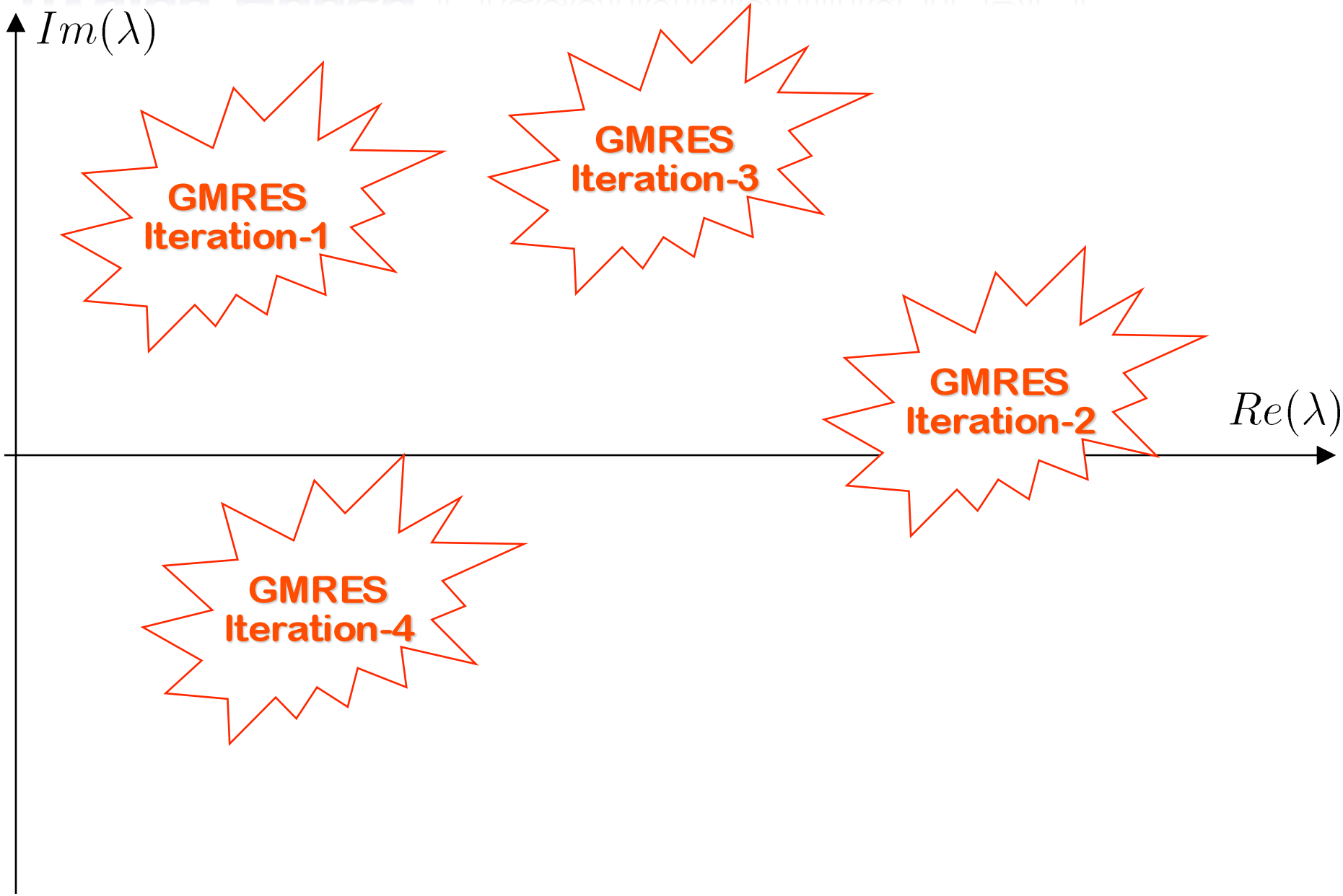
Physics-Based Preconditioning (PBP)



Physics-Based Preconditioning (PBP)



Physics-Based Preconditioning (PBP)



Block-Diagonal (BD) Preconditioning

$$\mathbb{U}_j = \left(\rho_j^{(0)}, \rho_j^{(1)}, \dots, \rho_j^{(p)}, m_j^{(0)}, m_j^{(1)}, \dots, m_j^{(p)}, E_j^{(0)}, E_j^{(1)}, \dots, E_j^{(p)}\right)^T$$

$$\underbrace{\begin{bmatrix} \tilde{\hat{\mathbb{J}}}_{11} & \tilde{\hat{\mathbb{J}}}_{12} & \tilde{\hat{\mathbb{J}}}_{13} & 0 & 0 & 0 & 0 & 0 \\ \tilde{\hat{\mathbb{J}}}_{21} & \tilde{\hat{\mathbb{J}}}_{22} & \tilde{\hat{\mathbb{J}}}_{23} & \tilde{\hat{\mathbb{J}}}_{24} & 0 & 0 & 0 & 0 \\ & & \dots\dots\dots & & & & & \\ & & \dots\dots\dots & & & & & \\ & & \dots\dots\dots & & & & & \\ & & \dots\dots\dots & & & & & \\ & & \dots\dots\dots & & & & & \tilde{\hat{\mathbb{J}}}_{N_{cells} N_{cells}} \end{bmatrix}}_{\mathbb{J}} \underbrace{\begin{pmatrix} \delta\mathbb{U}_1 \\ \delta\mathbb{U}_2 \\ \dots \\ \dots \\ \dots \\ \dots \\ \delta\mathbb{U}_{N_{cells}} \end{pmatrix}}_{\delta\hat{\mathbf{X}}} = \underbrace{\begin{pmatrix} \tilde{\hat{\mathbf{res}}}_1 \\ \tilde{\hat{\mathbf{res}}}_2 \\ \dots \\ \dots \\ \dots \\ \dots \\ \tilde{\hat{\mathbf{res}}}_{N_{cells}} \end{pmatrix}}_{\mathbf{v}}$$

$$\tilde{\hat{\mathbb{J}}}_{km} \Longrightarrow N_j \times N_j \text{ matrix}$$
$$N_j = 3\,(p+1)$$

Block-Diagonal (BD) Preconditioning

$$\mathbb{U}_j = \left(\rho_j^{(0)}, \rho_j^{(1)}, \dots, \rho_j^{(p)}, m_j^{(0)}, m_j^{(1)}, \dots, m_j^{(p)}, E_j^{(0)}, E_j^{(1)}, \dots, E_j^{(p)}\right)^T$$

$$\underbrace{\begin{bmatrix} \tilde{\hat{\mathbb{J}}}_1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & \tilde{\hat{\mathbb{J}}}_2 & 0 & 0 & 0 & 0 & 0 & 0 \\ & & & \cdots\cdots & & & & \\ & & & \cdots\cdots & & & & \\ & & & \cdots\cdots & & & & \\ & & & \cdots\cdots & & & & \\ 0 & 0 & 0 & 0 & 0 & 0 & \tilde{\hat{\mathbb{J}}}_{N_{cells}-1} & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & \tilde{\hat{\mathbb{J}}}_{N_{cells}} \end{bmatrix}}_{\mathbb{P}_{BD}} \underbrace{\begin{pmatrix} \delta\mathbb{U}_1 \\ \delta\mathbb{U}_2 \\ \dots \\ \dots \\ \dots \\ \dots \\ \delta\mathbb{U}_{N_{cells}-1} \\ \delta\mathbb{U}_{N_{cells}} \end{pmatrix}}_{\delta\hat{\mathbf{X}}} = \underbrace{\begin{pmatrix} \tilde{\hat{\mathbf{res}}}_1 \\ \tilde{\hat{\mathbf{res}}}_2 \\ \dots \\ \dots \\ \dots \\ \dots \\ \tilde{\hat{\mathbf{res}}}_{N_{cells}-1} \\ \tilde{\hat{\mathbf{res}}}_{N_{cells}} \end{pmatrix}}_{\mathbf{v}}$$

$$\boxed{\delta\mathbb{U}_j = \tilde{\hat{\mathbb{J}}}_j^{-1} \tilde{\hat{\mathbf{res}}}_j}$$

Operator-Split (PBP)

$$\underbrace{\begin{bmatrix} \mathbb{J}_{PP} & \mathbb{J}_{Pu} & \mathbb{J}_{Pi} \\ \mathbb{J}_{uP} & \mathbb{J}_{uu} & \mathbb{J}_{ui} \\ \mathbb{J}_{iP} & \mathbb{J}_{iu} & \mathbb{J}_{ii} \end{bmatrix}}_{\mathbb{J}} \underbrace{\begin{pmatrix} \delta\mathcal{P} \\ \delta\mathcal{U} \\ \delta\mathcal{I} \end{pmatrix}}_{\delta\mathbb{V}} = -\underbrace{\begin{pmatrix} \mathbf{res}_{\mathcal{P}} \\ \mathbf{res}_{\mathcal{U}} \\ \mathbf{res}_{\mathcal{I}} \end{pmatrix}}_{\mathbf{b}}$$

$$\mathbb{J}_{km} \Longrightarrow N_{\text{cells}} \times N_{\text{cells}} \text{ matrix}$$

Operator-Split (PBP)

Step I: ...Targeting heat conduction...

$$\begin{bmatrix} \cancel{0} & \cancel{0} & \mathbb{J}_{ii} \end{bmatrix} \begin{pmatrix} \delta\mathcal{P} \\ \delta\mathcal{U} \\ \delta\mathcal{I} \end{pmatrix} = -\mathbf{res}_{\mathcal{I}}$$

$\mathbb{J}_{iP}$ $\mathbb{J}_{iu}$

$$\boxed{\delta\mathcal{I} = -\mathbb{J}_{ii}^{-1}\mathbf{res}_{\mathcal{I}}}$$

$$\underbrace{\begin{bmatrix} \mathbb{J}_{PP} & \mathbb{J}_{Pu} \\ \mathbb{J}_{uP} & \mathbb{J}_{uu} \end{bmatrix}}_{Pressure\text{-}velocity\ matrix} \begin{pmatrix} \delta\mathcal{P} \\ \delta\mathcal{U} \end{pmatrix} = -\begin{pmatrix} \mathbf{res}_{\mathcal{P}} + \mathbb{J}_{Pi}\delta\mathcal{I} \\ \mathbf{res}_{\mathcal{U}} + \mathbb{J}_{ui}\delta\mathcal{I} \end{pmatrix}$$

Pressure-Poisson equation:

$$\underbrace{\left(\mathbb{J}_{PP} - \mathbb{J}_{Pu}\mathbb{J}_{uu}^{-1}\mathbb{J}_{uP}\right)}_{Laplacian,\ \mathbb{L}} \delta\mathcal{P} = \underbrace{-\left(\mathbf{res}_{\mathcal{P}} + \mathbb{J}_{Pi}\delta\mathcal{I} - \mathbb{J}_{Pu}\mathbb{J}_{uu}^{-1}\left(\mathbf{res}_{\mathcal{U}} + \mathbb{J}_{ui}\delta\mathcal{I}\right)\right)}_{\mathbf{b}}$$

Schur complement

Step II:

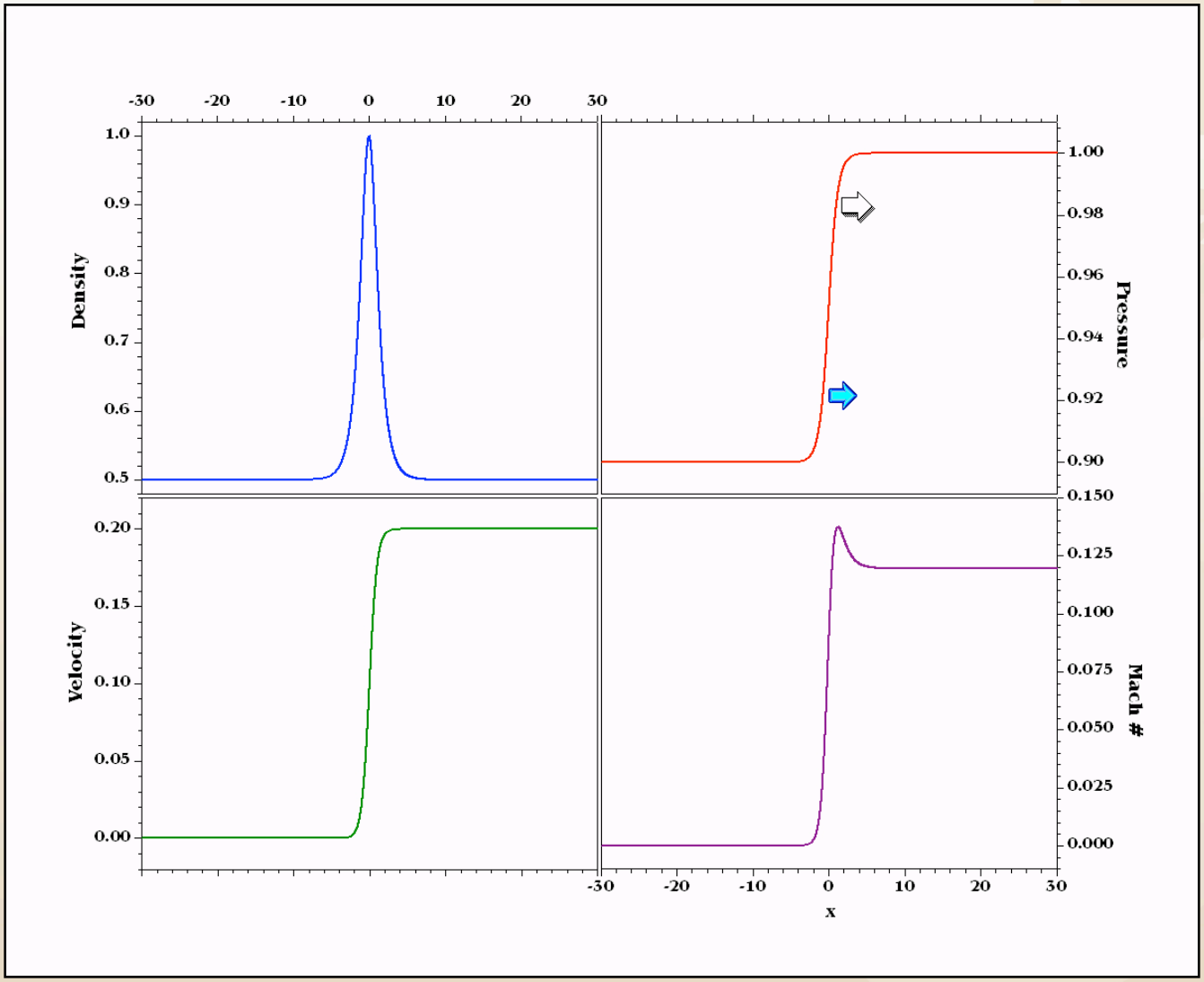
$$\boxed{\delta\mathcal{P} = \mathbb{L}^{-1}\mathbf{b}}$$

...Targeting stiff pressure waves...

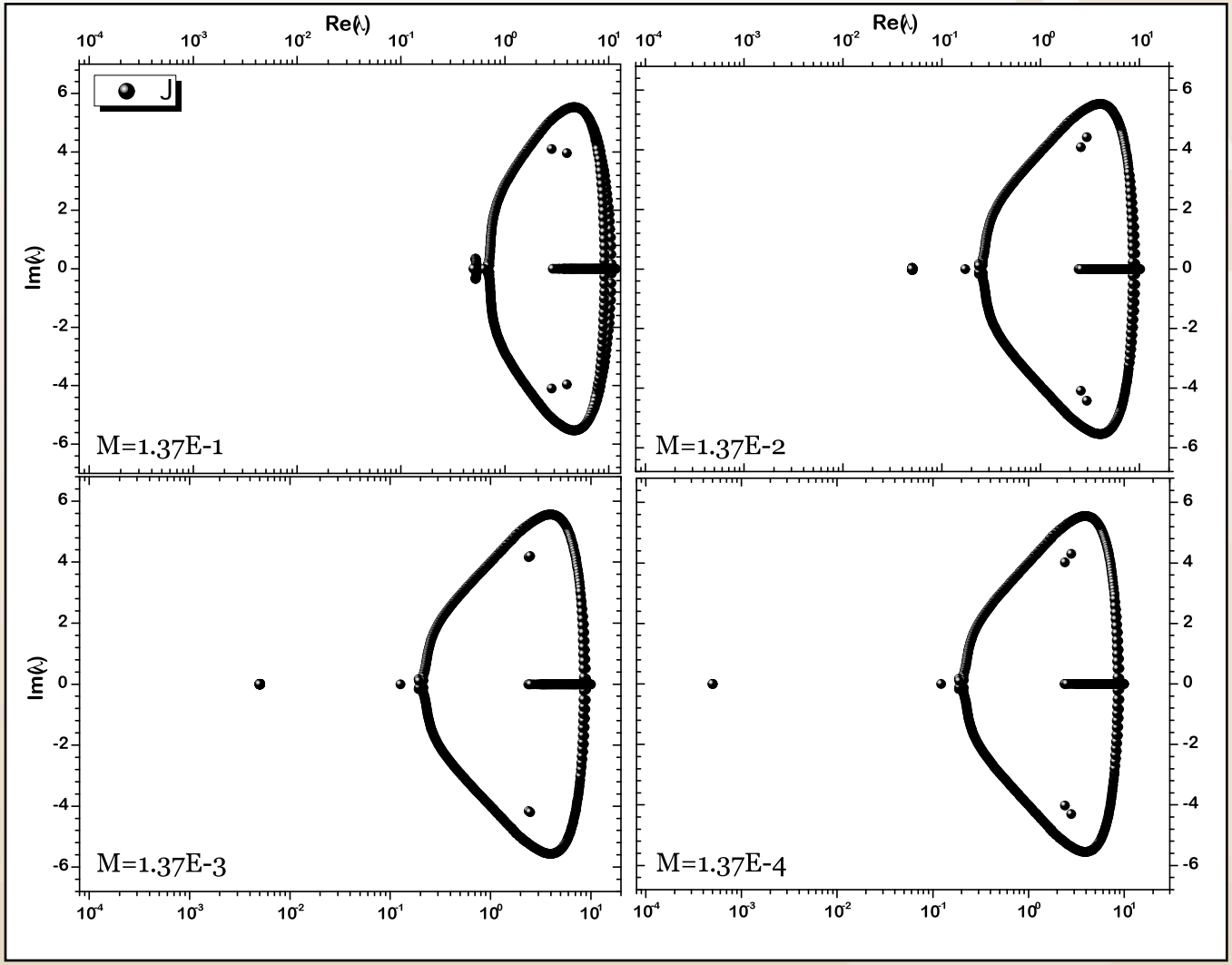
Step III: ...Targeting viscous operator...

$$\boxed{\delta\mathcal{U} = -\mathbb{J}_{uu}^{-1}\left(\mathbf{res}_{\mathcal{U}} + \mathbb{J}_{uP}\delta\mathcal{P} + \mathbb{J}_{ui}\delta\mathcal{I}\right)}$$

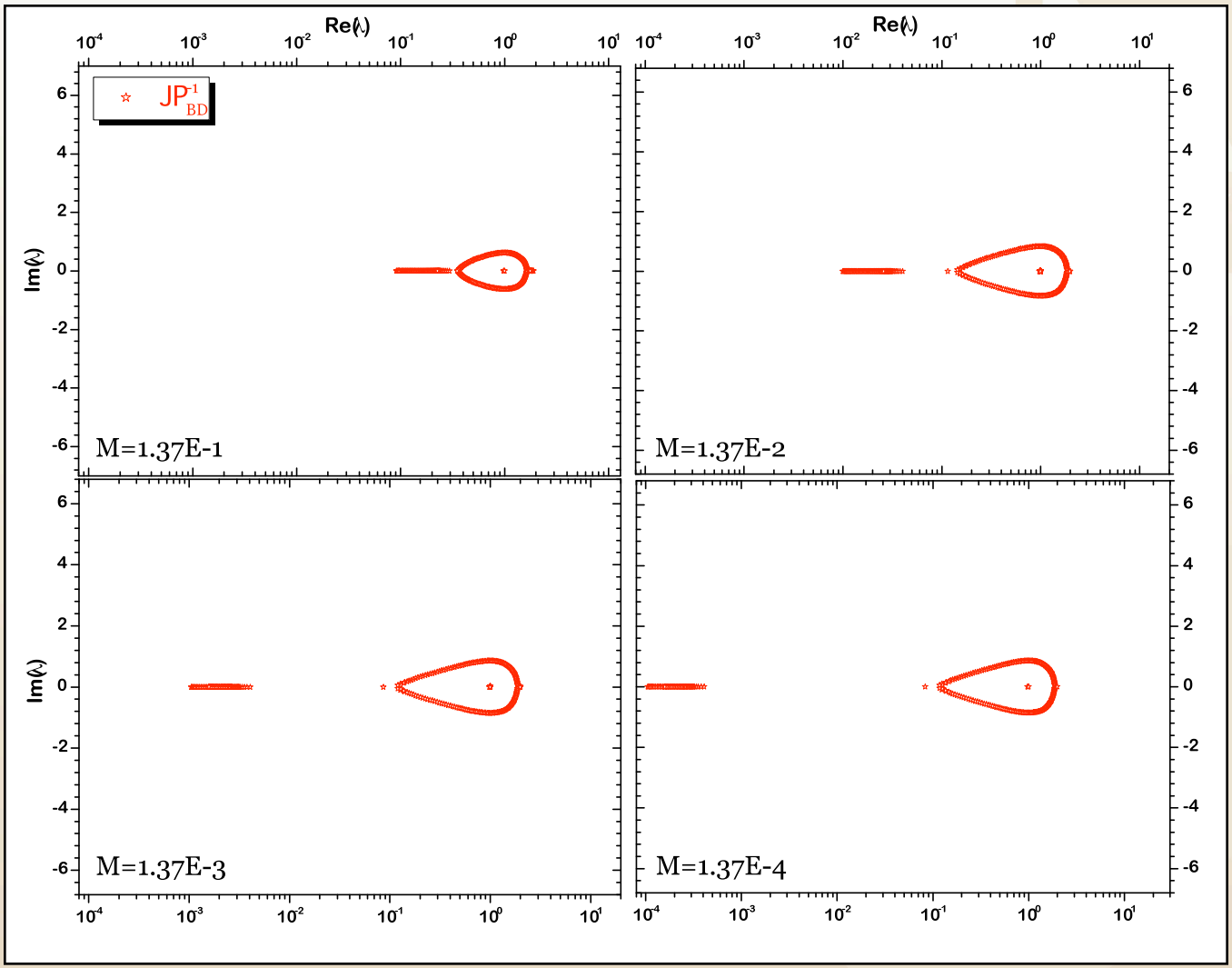
"Eigenscopy"


Density
Pressure
Velocity
Mach #
x
-30
-20
-10
0
10
20
30
1.0
0.9
0.8
0.7
0.6
0.5
1.00
0.98
0.96
0.94
0.92
0.90
0.150
0.125
0.100
0.075
0.050
0.025
0.000
0.20
0.15
0.10
0.05
0.00

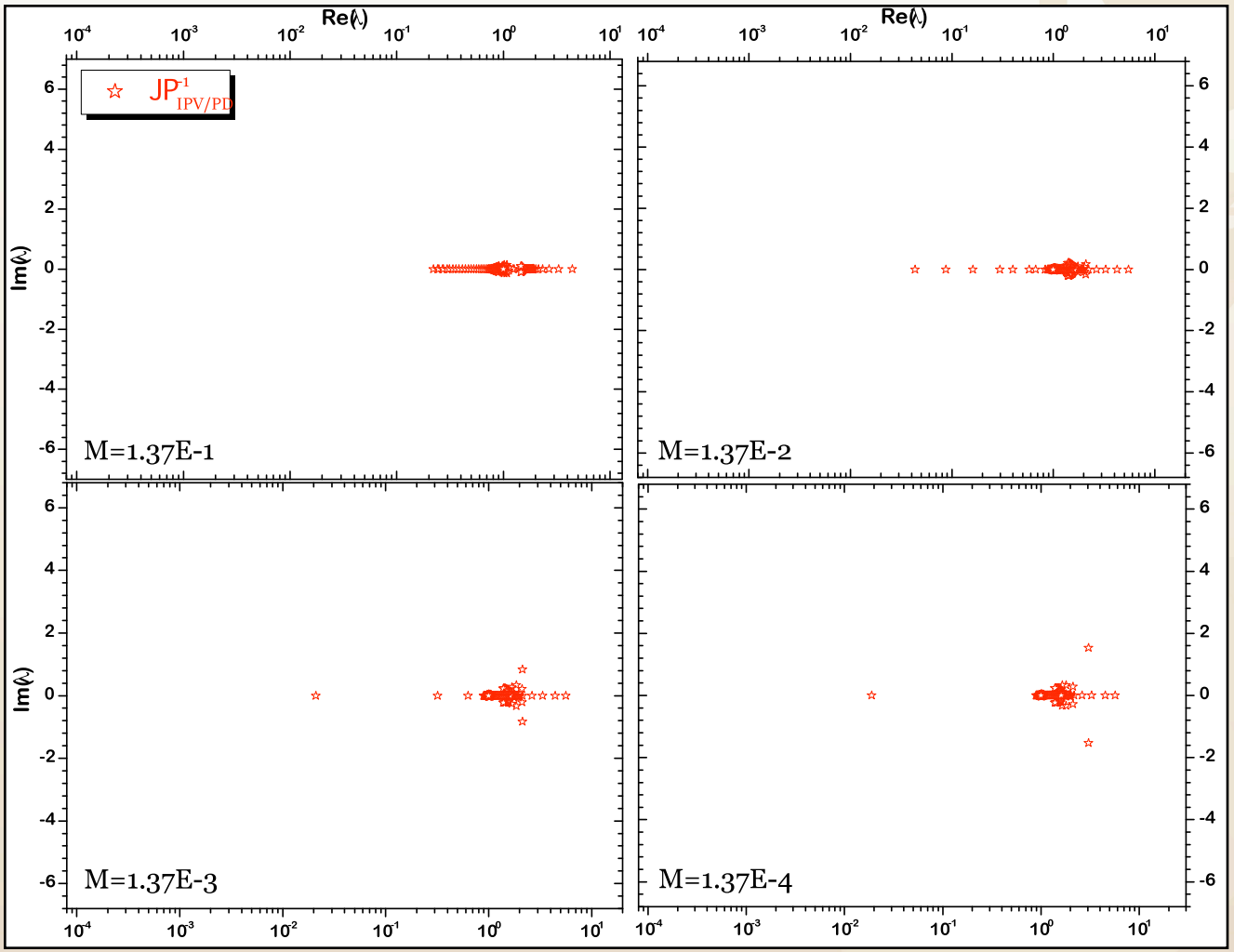
"Eigenscopy"



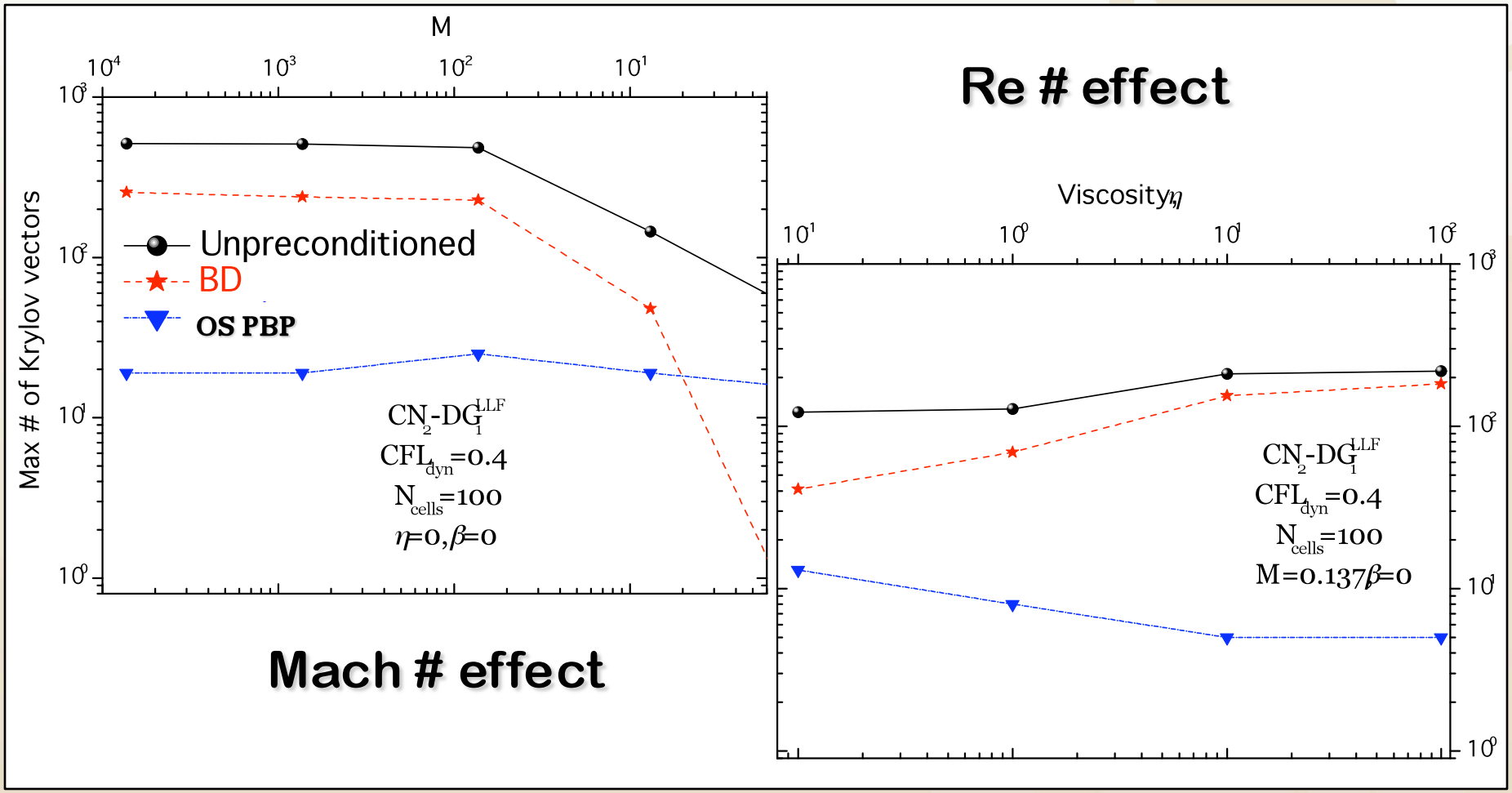
"Eigenscopy" of BD Preconditioning



"Eigenscopy" of Operator-Split PBP



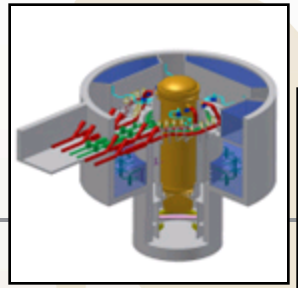
Efficacy of Preconditioning

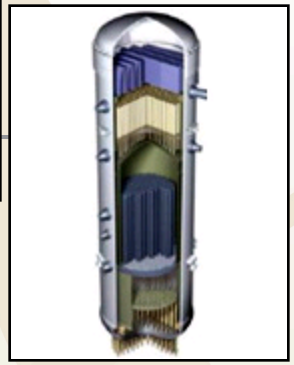


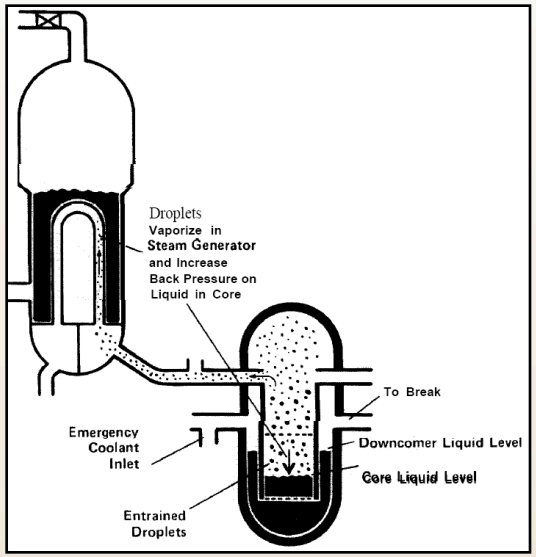
R.R.Nourgaliev, T.G.Theofanous, H.Park, V.Mousseau, and D.Knoll, "Direct Numerical Simulation of Interfacial Flows: Implicit Sharp-Interface Method (I-SIM)" (*Invited Talk*), *AIAA 2008-1453, 46th AIAA Aerospace Sciences Meeting and Exhibit*, January 7-10, 2008, Reno, NV, USA.

V. Coarse-Grain Modeling

COARSE-GRAIN MODELING





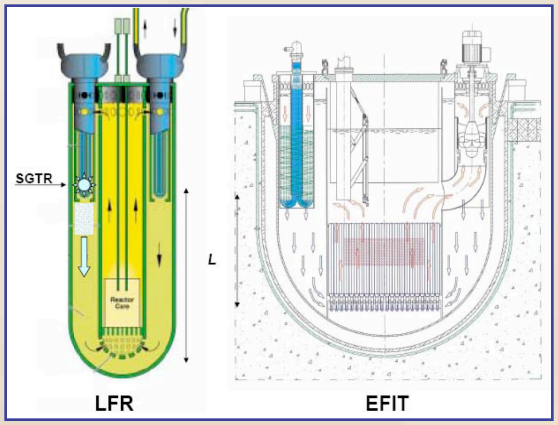


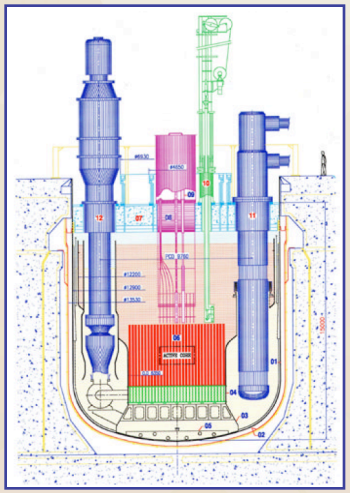
- Growing demand for (3D, system) analysis, long transients

Risk-significant (long) abnormal transients in new plant design with passive safety features

- (Practical) computing resources available for production calculations

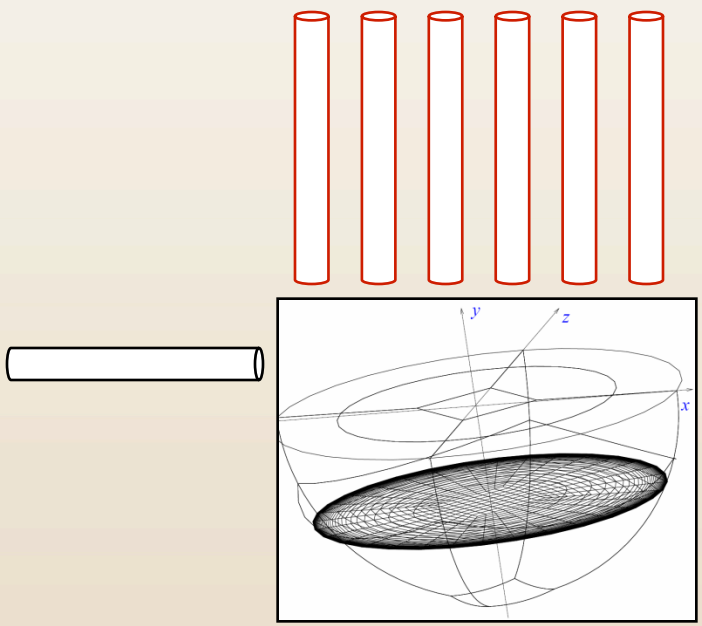
Run on parallel computers (30-100 processors)



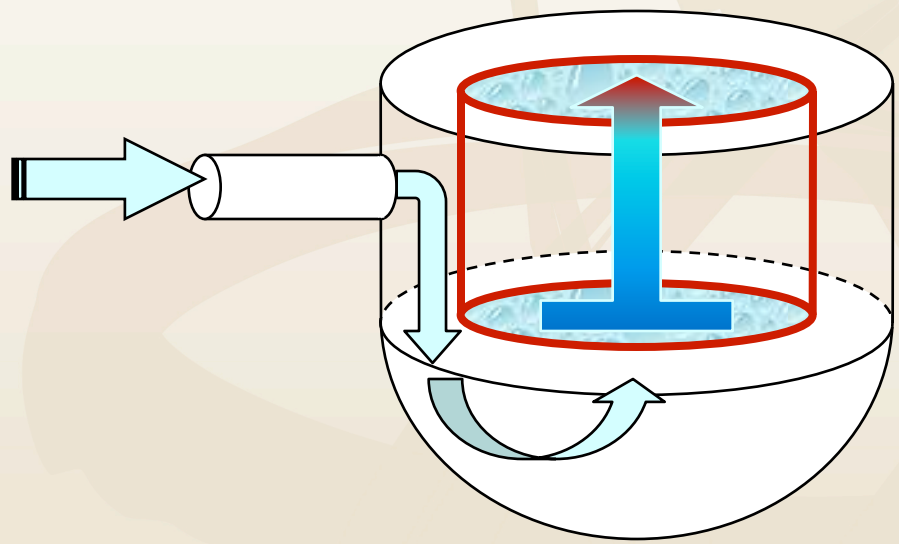


COARSE-GRAIN MODELING

System analysis



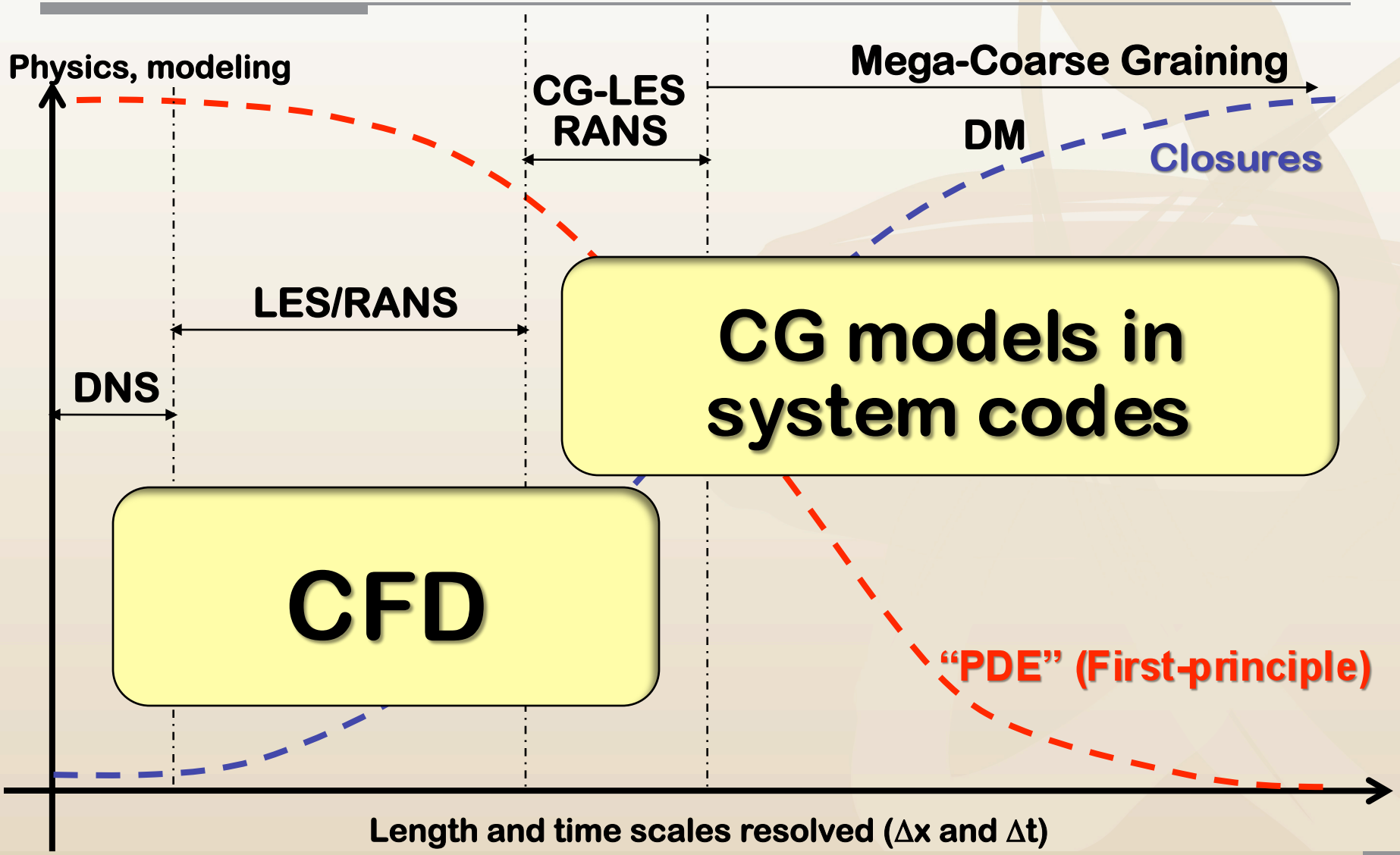
Modeled system components



3D prediction of the system dynamics and key safety parameters with reasonable accuracy by sufficiently coarse resolution

No attempt to resolve boundary layers and small-scale turbulent mixing

COARSE-GRAIN MODELING



COARSE-GRAIN MODELING

Under-resolved features are effectively modeled by coarse-grained (CG) subgrid-scale (SGS) closures

RANS

LES

Smagorinski
Dynamic
2-eqs.-Dynamic
MILES

Discrete Modeling

LANS

Questions?