HopeFOAM: High Order Parallel Extensible CFD Software

Research for Effective CFD Application Development on HPC Systems

A/Prof. Xu Xinhai

Exercise Group

National Innovation Institute of Defense Technology

June,2018@OpenFOAM Workshop, Shanghai

This report will discuss the work on OpenFOAM from a new perspective, HPC

Outline

- Part I: Who am I？

  ——Exercise Group from HPC area

- Part II: Where am I from？

  ——OpenFOAM to HopeFOAM

- Part III: Where am I going？

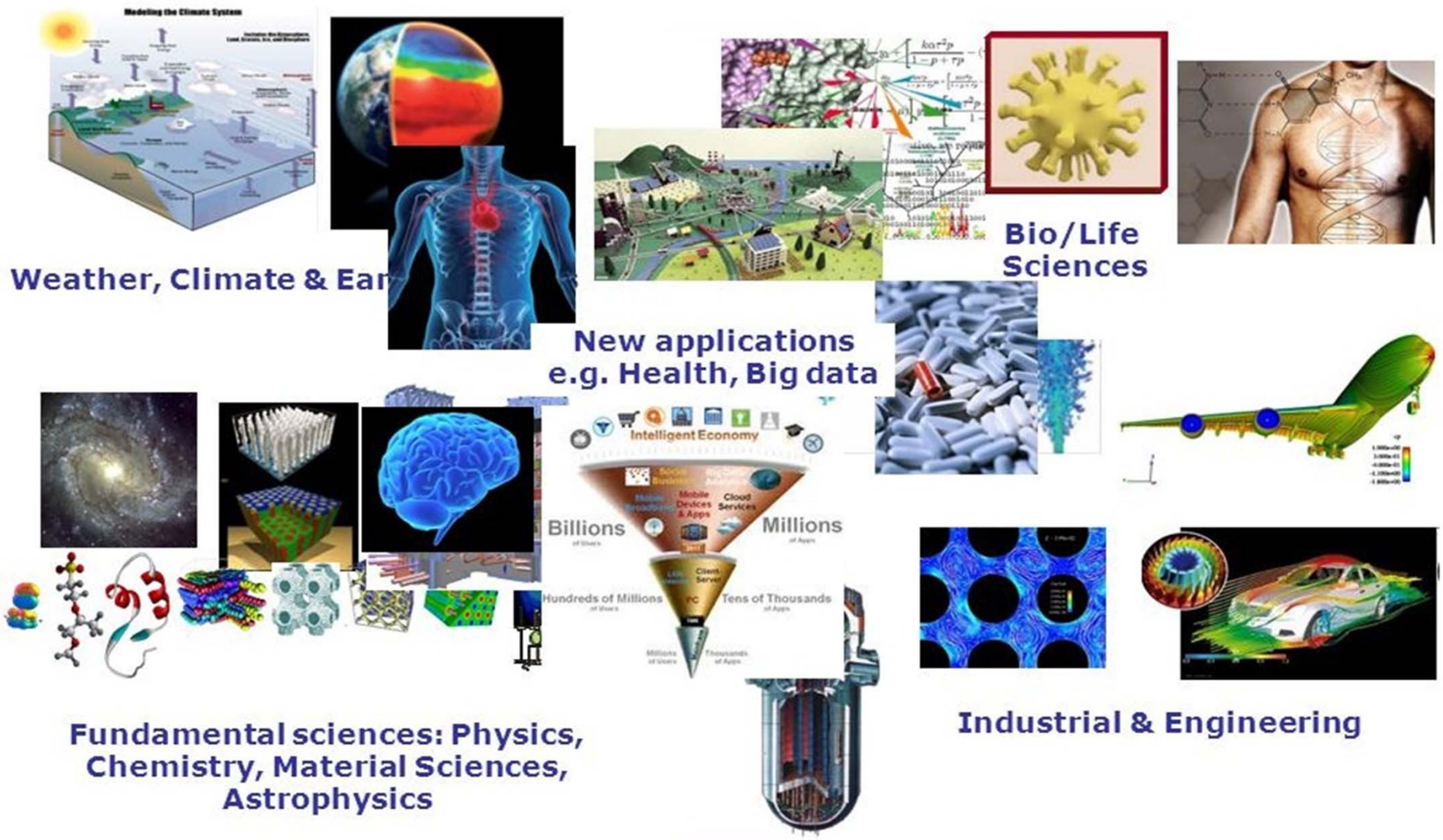
  ——HPC Service based on HopeFOAM

Part I: Who am I ?

——Exercise Group from HPC area

Development of HPC

Application areas



Development of HPC

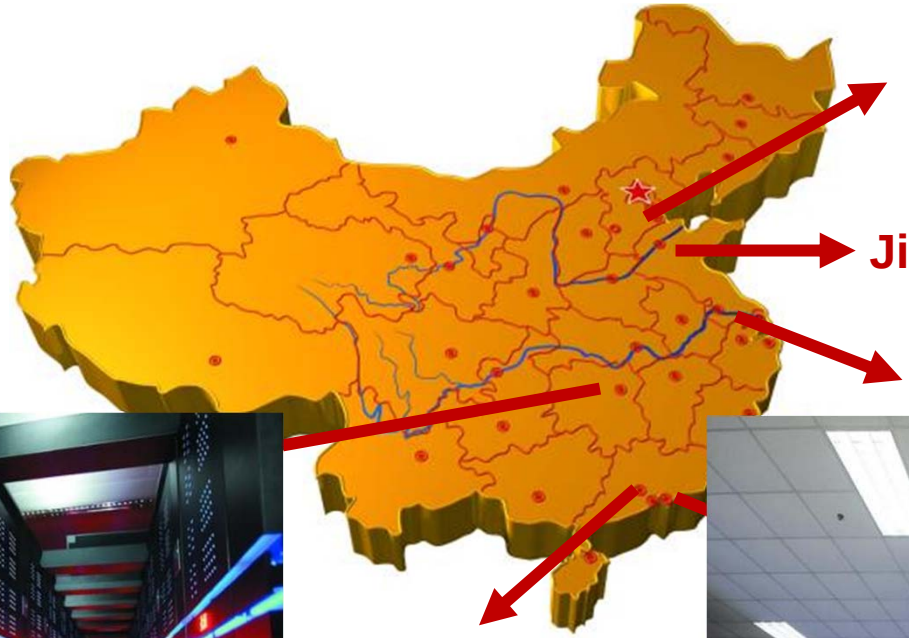
HPC Centers

U.S.

- Oak Ridge National Laboratory
- Lawrence Livermore National Laboratory
- Argonne National Laboratory
- Texas Advanced Computing Center
- San Diego Supercomputer Center

Development of HPC

- HPC Centers
  - U.S.
  - China: National SuperComputer Center in 6 cities



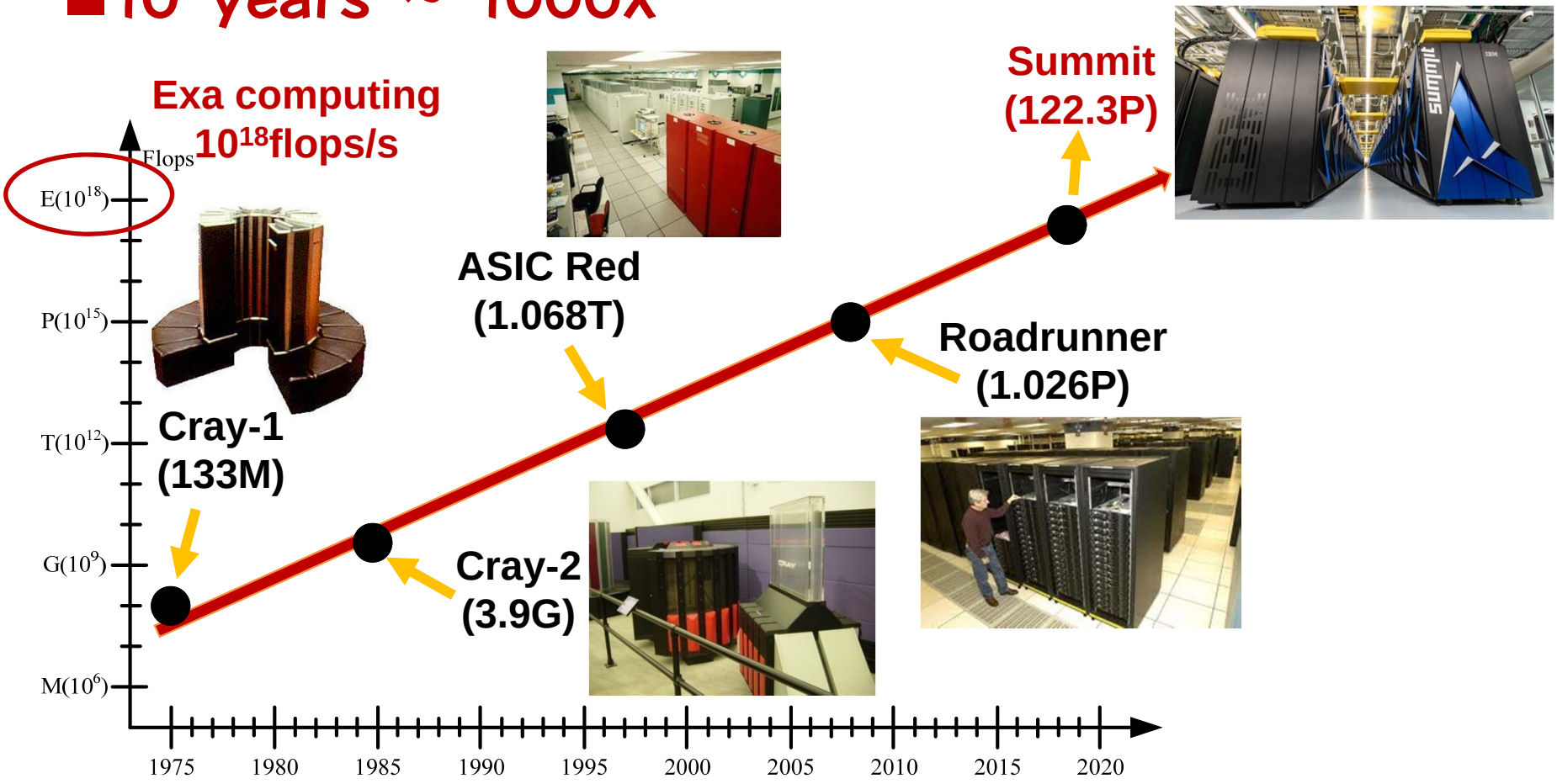
Development of HPC

- HPC Centers
  - U.S.
  - China
  - Europe
    - Germany: Juelich Supercomputing Centre
    - Spain: Barcelona Supercomputing Center
    - UK: six centers will be launched at Cambridge, Edinburgh, Exeter, Oxford, etc.
    - Italy: Cineca
    - Switzerland: Swiss National Supercomputing Centre at ETH Zurich
    - France, Russia, Sweden,…….

Development of HPC

Exponential growth of computing power

- 10 years ~ 1000X



Exa computing

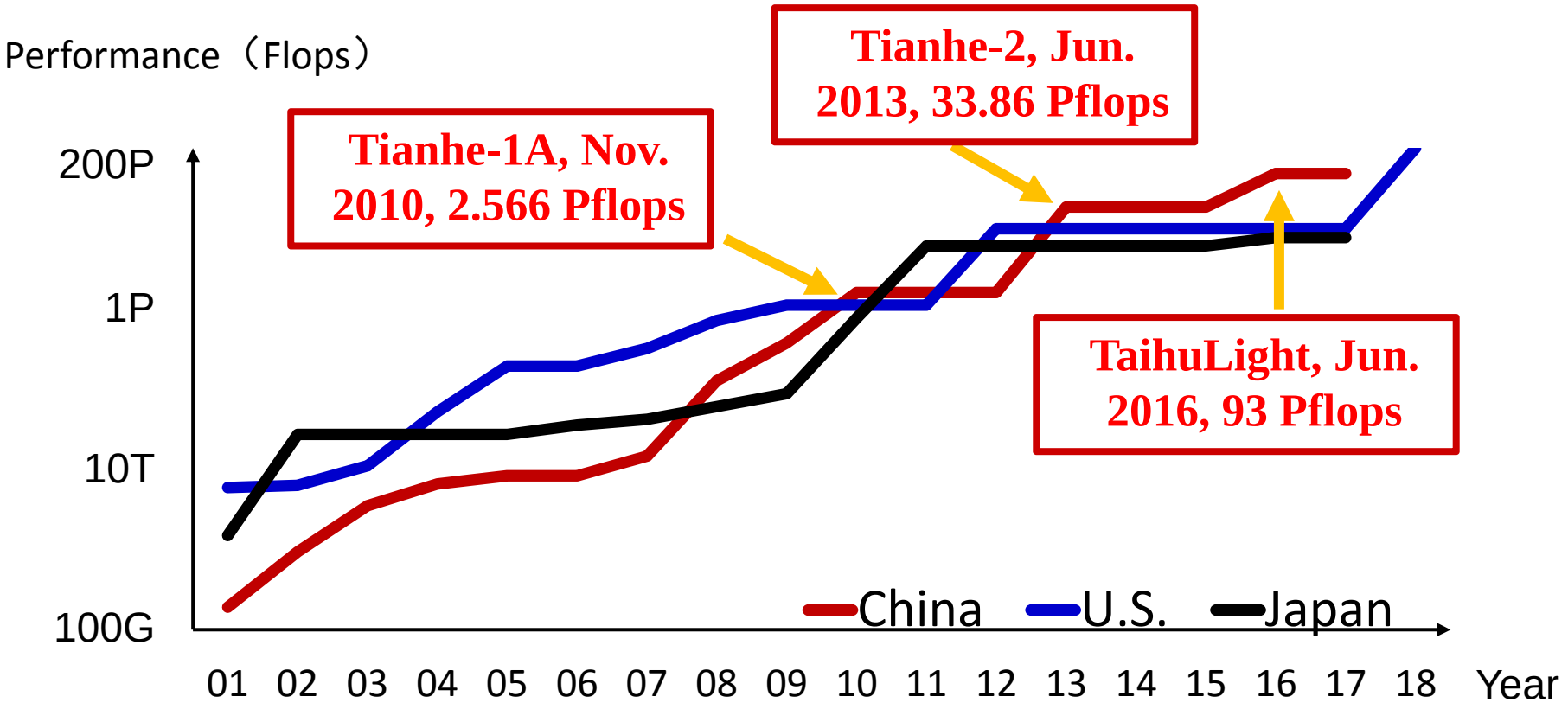
America

- In "Strategy for American Innovation", Former American president Obama ranked exa computing as the most important technical challenge for the United States in the 21st century
  - The U.S. Congress approved a funding request from U.S. Department of Energy for $126 million for exa computing research, in Dec. 2011
  - In 2016, Exascale Computing Project (ECP), led by the department of energy, was launched
- The fastest supercomputer: Summit
  - 9216 POWER9 22-core CPUs
  - 27,648 Nvidia Tesla V100 GPUs
  - Deep learning capabilities of GPUs

Exa computing

China

- Ranked 1st in Top500 list for several times



Exa computing

China

- Ranked 1st in Top500 list for several times
- Ongoing Exascale system plans
  - The special project of high performance computing in "13th five-year plan": the prototype system development project of exa supercomputing
  - Last month, the prototype system of Tianhe-3 has been announced.

中科曙光 Sugon

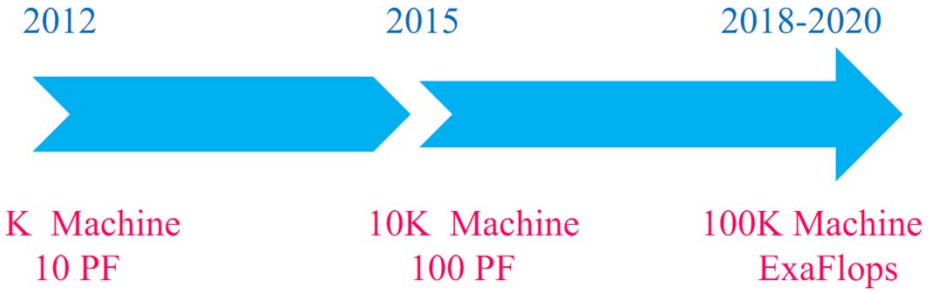
国防科技大学 NATIONAL UNIVERSITY OF DEFENSE TECHNOLOGY 1953

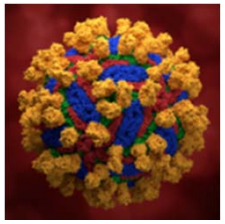
无锡江南计算技术研究所

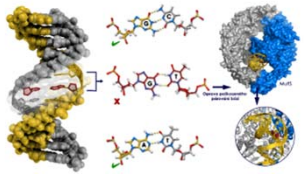
Exa computing

Japan

- The Japanese government invested $1.3 billion in exa computing research in 2011.
- FLAGSHIP2020 Project: to achieve Post K by 2020







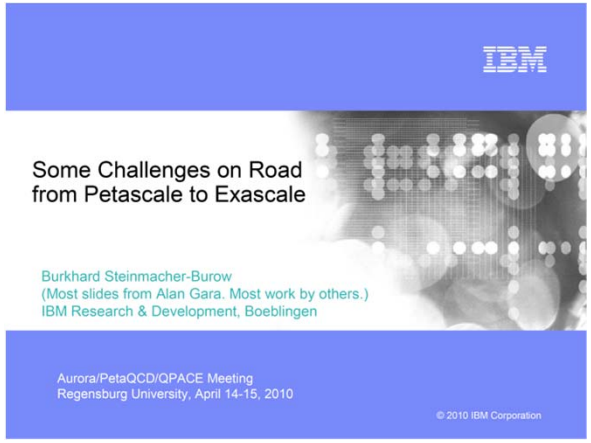
Europe

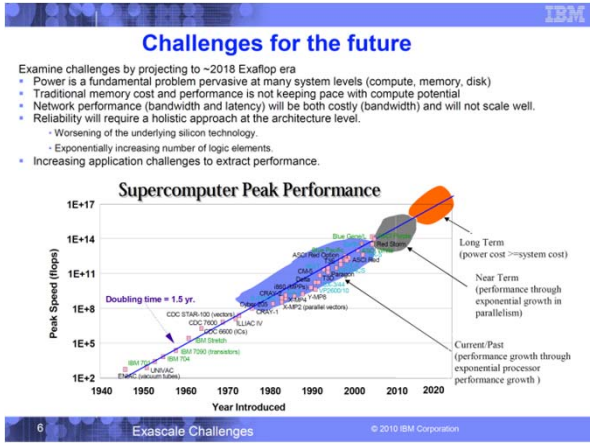
- Focus on the development of exa software

Challenges and Solution

Unfortunately…

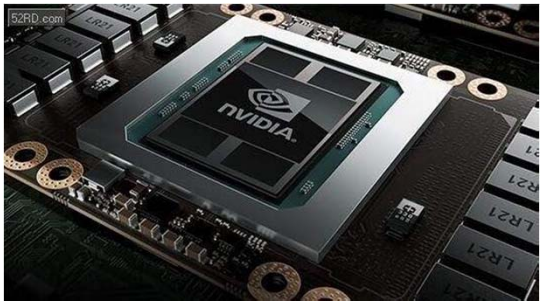
- In April 2010, IBM pointed out five challenges of exa systems: memory access, communication, reliability, energy consumption, and application in the report "Some Challenges on Road from Petascale to Exascale".





Challenges and Solution

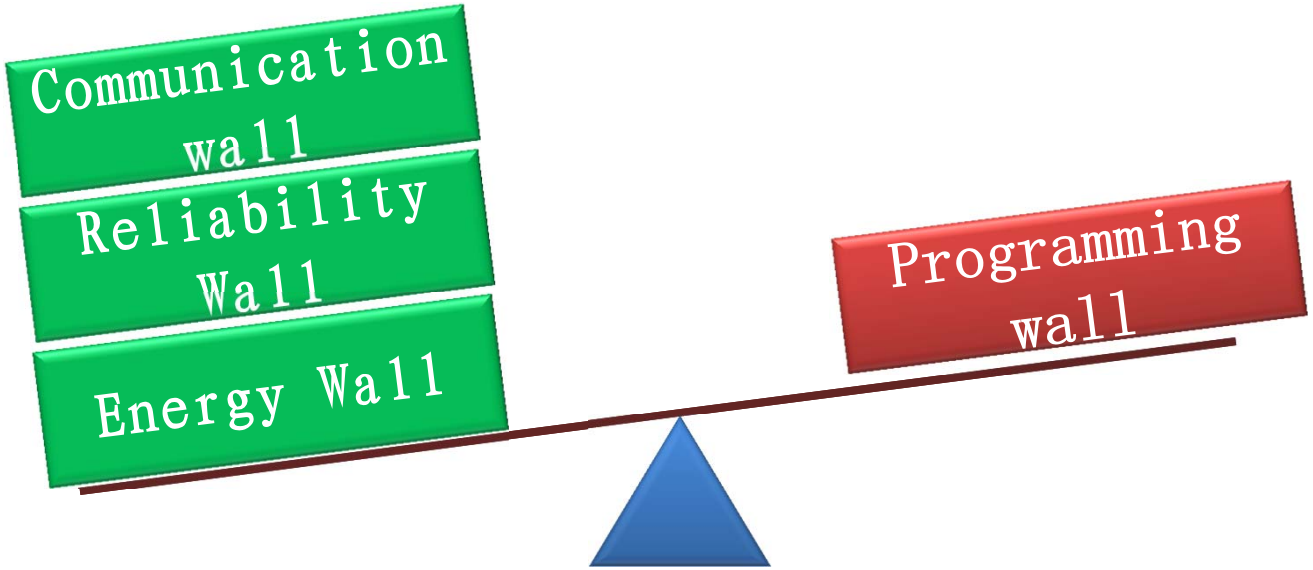
- To deal with "Five Walls" Challenges, HPC researchers keep hard working
  - Heterogeneous architecture to Reduce the system scale



  - Microprocessor: multi core→many core to reduce data movement
  - Optical interconnection technology for high bandwidth, low power consumption
  - ……

Challenges and Solution

- While alleviating the communication wall, the reliability wall and the energy wall, the development of hardware technology has aggravated the programming wall



Software technology becomes important

Exercise Group

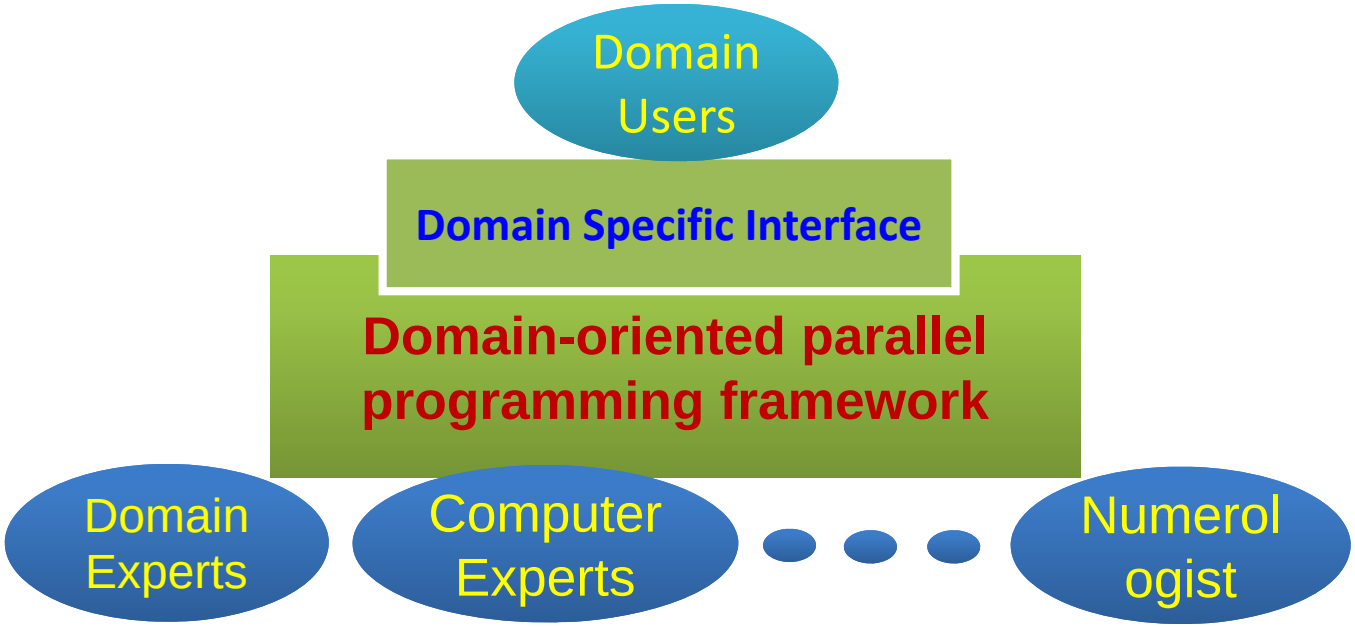
- **Programming problem**
  - Nowadays, most researchers have a common sense: *Simulation is important and necessary* for their work
  - However, lots of them (except parallel computing scientists) do not know
    - How to write programs for their research
    - How to write parallel programs
    - How to write high-efficiency parallel programs

Exercise Group

- Established in 2013, to alleviate the imbalance between hardware and software
  - *Chief scientist*: **Prof. Yang Xuejun**
    - Chief architect of Tianhe-1 and Tianhe-1A
  - *Group leader*: *A/Prof. Xu Xinhai*

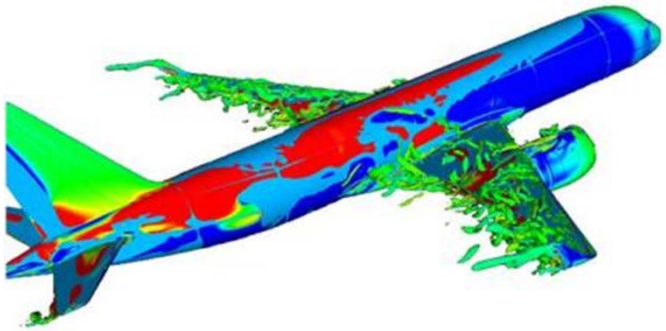
Exercise Group

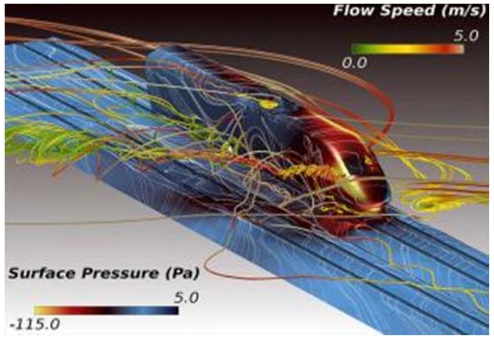
- Currently, we are researching **domain-oriented parallel programming framework**
  - ***Interdisciplinary Cooperation***

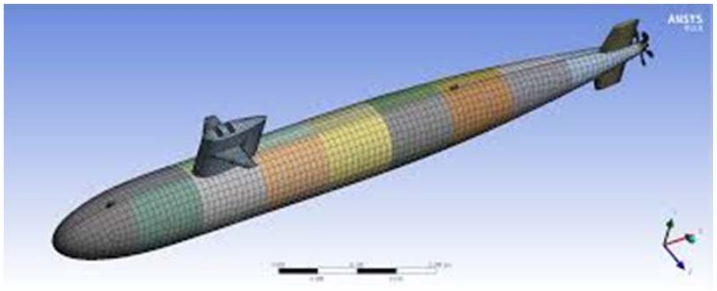


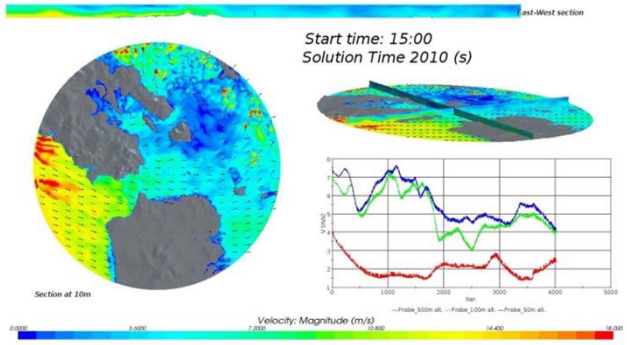
Exercise Group

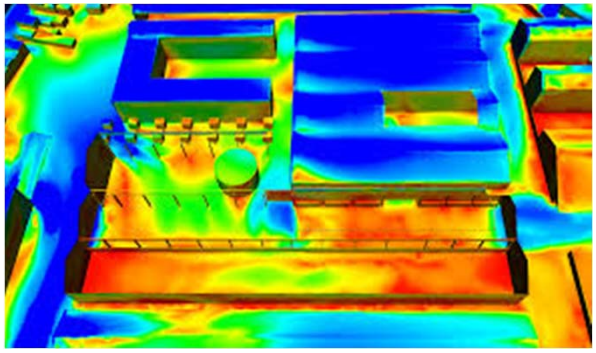
Our first domain——CFD

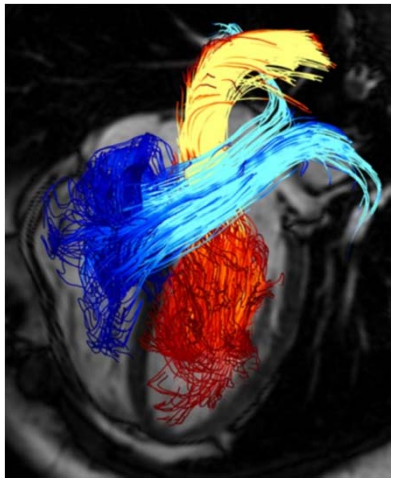












Thus, start up from OpenFOAM

Part II: Where am I from ?

——OpenFOAM to HopeFOAM

Part II

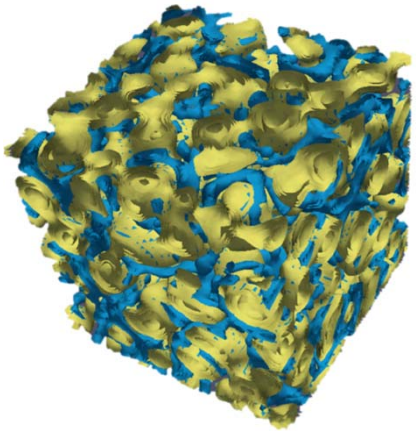
- Start with application development on OpenFOAM
  - *Some simulation cases*
  - *The FVM-LBM coupled simulation for visco-elastic fluids*
  - *Parallel RBF based mesh deformation*
- High order discretization of HopeFOAM
- HopeFOAM parallel optimization

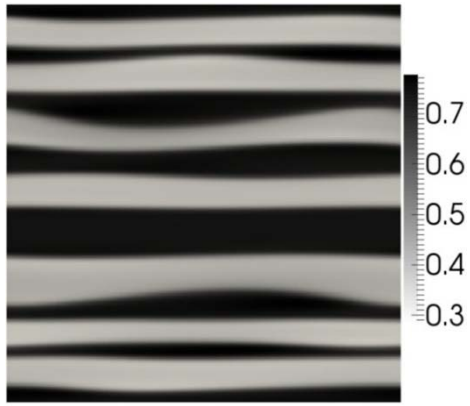
Simulation of Visco-elastic fluids

- **The wide range applications of viscoelastic fluids**
  - Typical viscoelastic fluids: Oil、polymer solutions、liquid detergent、food、blood、……
  - Applications: Polymer processing、Oil transport、Composite processing 、3D printing、Blood flow、……

Simulation of Visco-elastic fluids

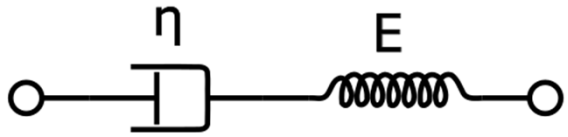
- The wide range applications of viscoelastic fluids
- We focused on the two-phase viscoelastic fluids
  - Contain two components in the fluid
  - To accurately capture the dynamics in the complex fluids, the key point is calculating the viscoelastic stress





The models for viscoelastic stress

- **The constitutive equation**: the relationship between the stress and the strain
  - ***Phenomenal constitutive equations***: *Modeling by the macroscopic phenomenon*
    - The Maxwell model
    - The Oldroyd-B model



  - ***Macroscopic constitutive model based on microscopic molecular theory***
    - The FENE model
    - The MARRUCCI model
    - The Rolie-Poly model

  - ***Calculating from the molecular theory***
    - The molecular dynamics(MD)
    - The **Brownian configuration fields(BCF)**

The macro-micro simulations

- **FH-BCF model: A macro-micro coupled two-fluid model based on BCF method for simulating the phase separation in polymer binary mixtures**
  - **Macroscopic equations for the fluid flow**

$$\vec{\nabla} \cdot \vec{v} = 0$$

$$\rho \frac{D\vec{v}}{Dt} = \eta_s \nabla^2 \vec{v} - \vec{\nabla} p - (2\phi_A - 1)\vec{\nabla}\mu + \vec{\nabla} \cdot \sigma_{\mathrm{p}}(\vec{r}, t)$$

  - **Microscopic equations for the viscoelastic stress**

$$\frac{D\overrightarrow{Q_i}(\vec{r}, t)}{Dt} = (\nabla \vec{v}(\vec{r}, t))^T \cdot \overrightarrow{Q_i}(\vec{r}, t) - \frac{F(\overrightarrow{Q_i}(\vec{r}, t))}{2\lambda} + \sqrt{\frac{1}{\lambda}} \frac{d\overrightarrow{W_i}(t)}{dt}$$

$$\sigma = \left(\frac{b+d+2}{b}\right) \frac{\eta_p}{\lambda} \left(\left\langle \vec{Q} \otimes F(\vec{Q}) \right\rangle - I \right)$$

The macro-micro simulations

- **The key issue**: the massive amount of BCFs (N_f generally 500~2000)
  - *calculate hundreds of equations*
  - *much more computational requirements*
  - *Parallel Computing is necessary*

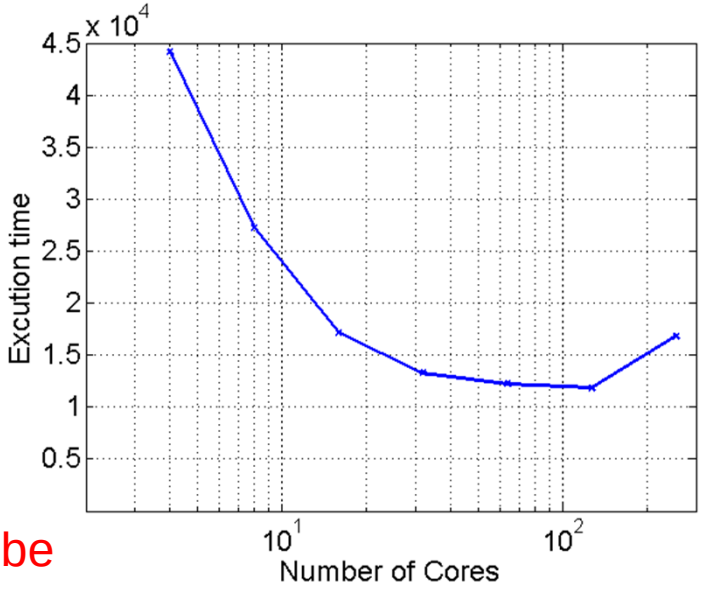
A case in literature on OpenFOAM:

Cells : 260×64×64(~1M)

N_f=800

CPU cores: 64

Time cost: 10~14 weeks



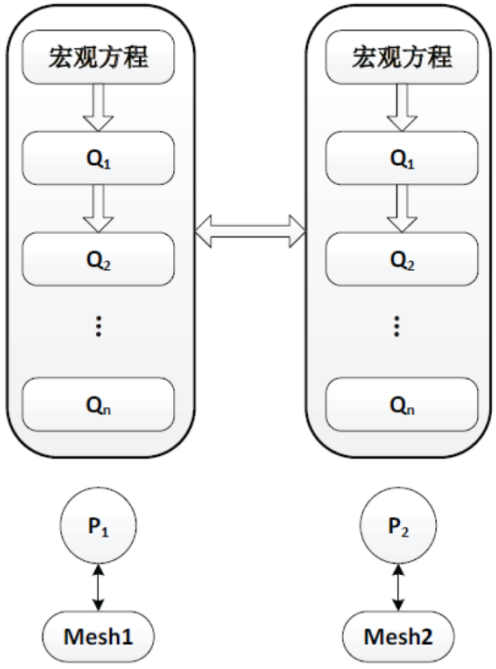
On OpenFOAM, the cases is parallelized by mesh decomposition. However, the simulation time can not be further reduced by increasing the CPU cores through mesh decomposition

hybrid decomposition algorithm

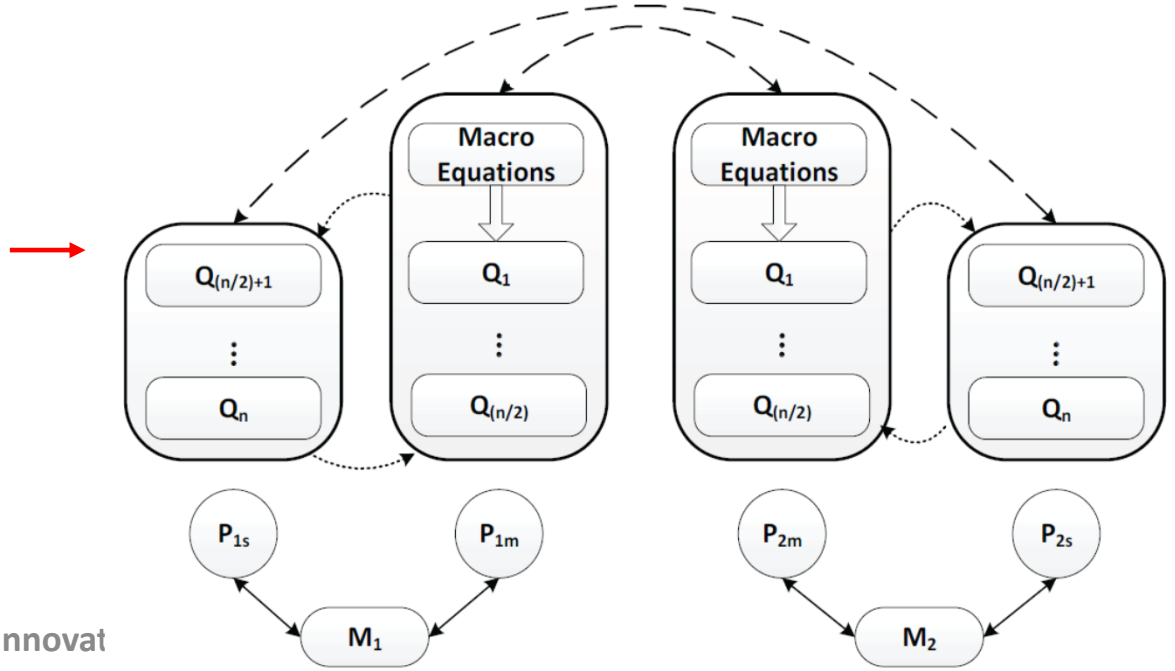
- Mesh decomposition+BCF decomposition
  - BCF equations Q_i are computationally ind… into par…

Using the hybrid decomposition algorithm, the computing task could be divided into much more CPU cores

Mesh decomposition

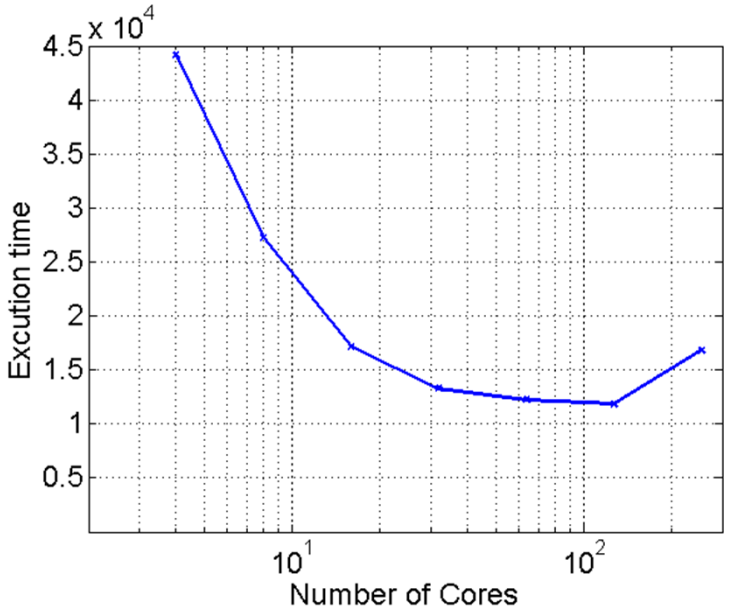


Mesh decomposition + BCF decomposition

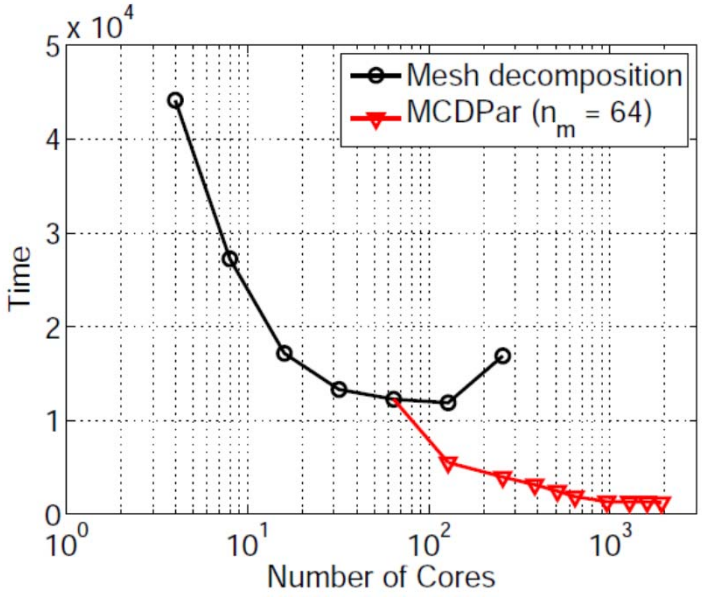


Original algorithm vs. MCDPar

- We implemented the MCDPar algorithm in a BCF numerical solver based on OpenFOAM
  - N_f = 2000 and Ncell = 262144 **[published on IPDPS 2016]**
  - Speedup = 9.23 with 7.5x cores



Mesh decomposition
The fastest run: 11854s on 128 cores
with N = $m = 128$



Mesh decomposition + BCF decompostion
The fastest run: 1284s on 960 cores
with $N = m \times c = 64 \times 15 = 960$

Some other related research for viscoelastic fluids based on OpenFOAM

- Proposed a novel two-phase fluid model based on the Rolie-Poly model： the FH-RP model
  - Simulating the instabilities in shear-banding flow without introducing extra parameters

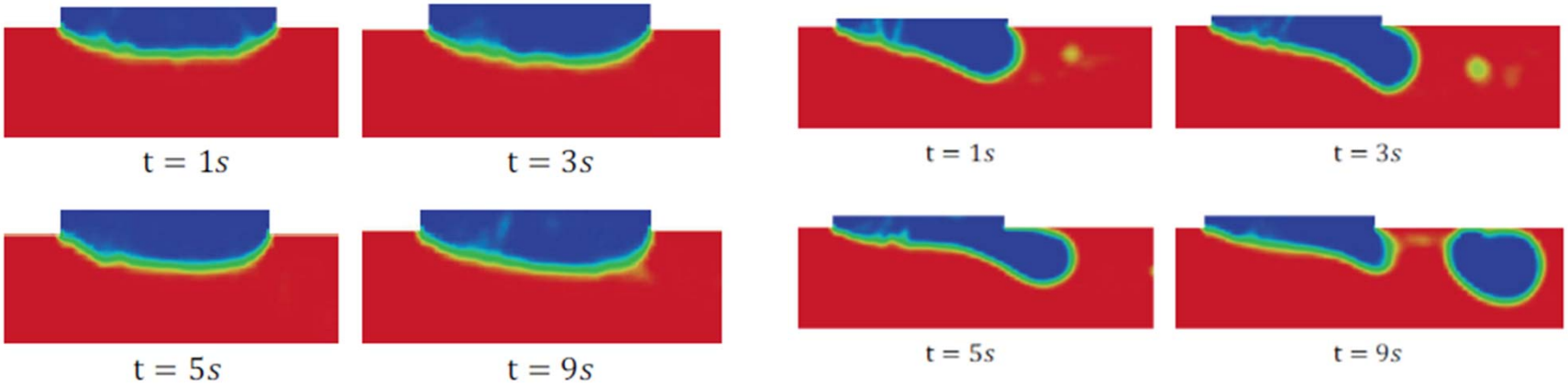
    [RSC Adv., 2014, 4, 61167.]
  - The research on non-equilibrium steady-states in viscoelastic fluids

    [Advances in Mechanical Engineering, 2015, Vol. 7(6) 1–10.]

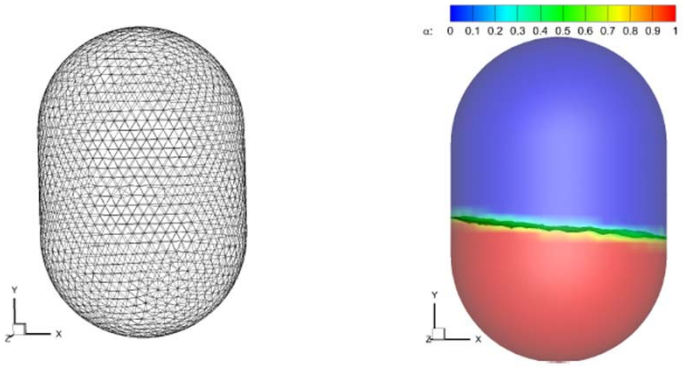
Some other simulations

☐The liquid instability in Spacecraft storage tank

■The Liquid Surface Characteristics of fuel Injection Process in a wedge flow geometry



■The simulation of Tank shaking



[Advances in Mechanical Engineering, 2016, 8(4): 1687814016645487.]

Some other simulations

Sound propagation simulation

[Shock and Vibration 2016(9):**703974**]

The wave equation

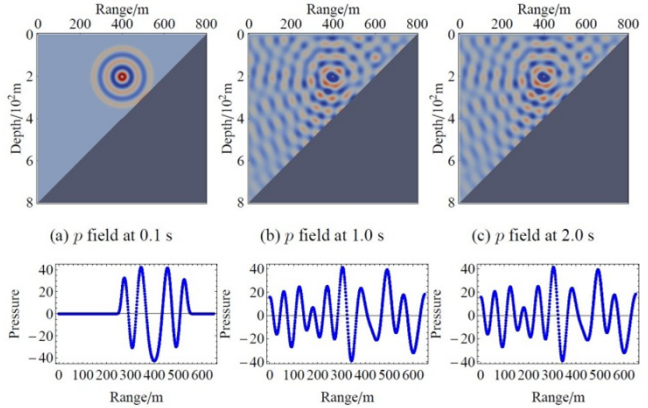
$$c(r)\rho(r)\nabla\cdot\left(\frac{1}{\rho(r)}\nabla p(t,r)\right)+\frac{\partial^2 p(t,r)}{\partial t^2}=\delta(t)S(t,r)$$

The boundary condition

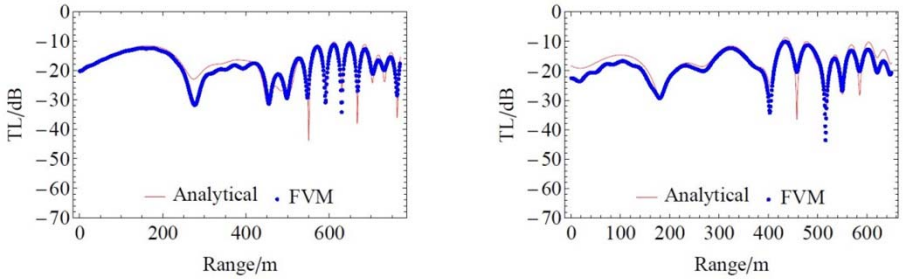
$$\frac{\partial p}{\partial t}=0$$

Results

- Time-domain



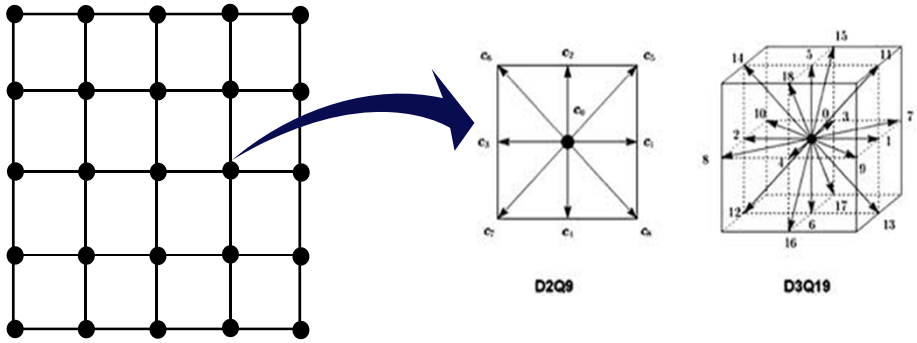
- Frenquency-domain



FVM-LBM coupled simulation

The Lattice Boltzmann method

- **A meso-scale method: To replace the N-S equation solver**
  - Discretize the space into lattices (DnQm)



  - Distribution function f could collide and migrate as particles

$$f_\alpha(\mathbf{x}+\mathbf{c}_\alpha\Delta t, t+\Delta t) - f_\alpha(\mathbf{x},t) = -\frac{1}{\tau}\left(f_\alpha(\mathbf{x},t) - f_\alpha(\mathbf{x},t)^{eq}\right) + S$$

$$\rho = \sum_\alpha f_\alpha,\ p = c_s^2\rho,\ \rho\mathbf{u} = \sum_\alpha f_\alpha \mathbf{c}_\alpha$$

  - Then, the macroscopic variables can be calculated from f.

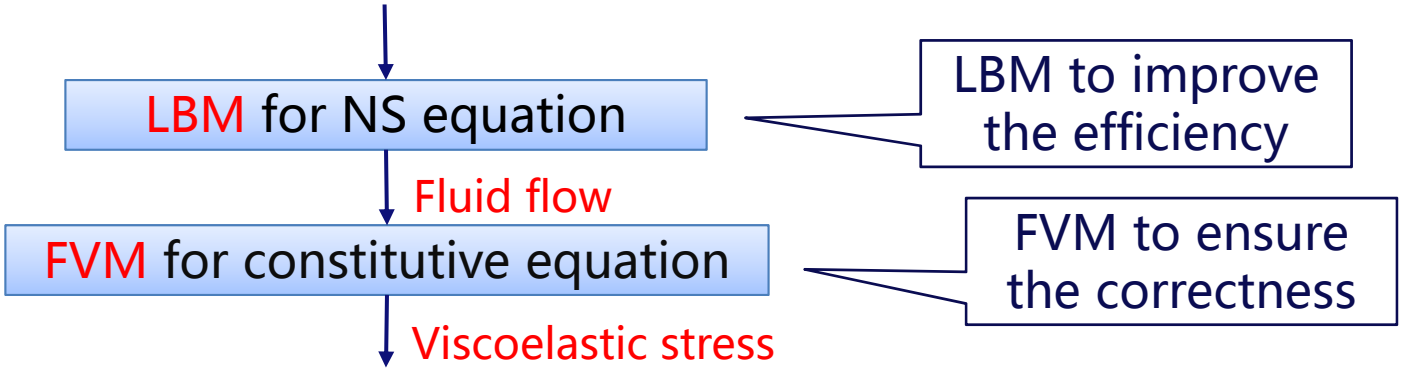
- **Compared to FVM, LBM is simple, efficient and highly scalable on high performance computers.**

FVM-LBM coupled simulation

- **Why coupled simulation?**
  - *A single method(FVM/LBM) cannot take into account the correctness and efficiency at the same time*
  - *Existed software does not support the coupled simulation*

FVM-LBM coupled simulation

- **Why coupled simulation?**
- **Developed a coupled simulation framework**
  - **Based on OpenFOAM and OpenLB**
  - **LBM for NS equation, FVM for the constitutive equation**



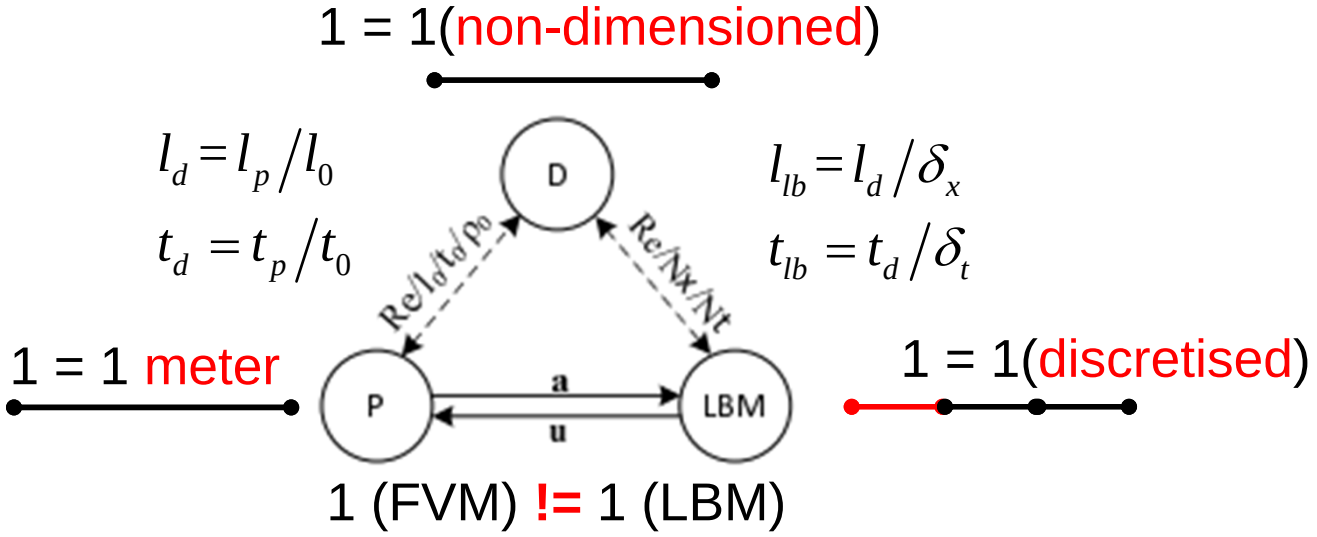
This coupled simulation framework can ensure both the efficiency and the correctness

FVM-LBM coupled simulation

Implementation

Dimension conversion is required to keep the data unified

- The basic dimension：Length、time、mass



- The general dimension conversion equations：

$$t^P = \frac{l_0^P \delta t}{u_0^P} t^{LBM} \qquad \mathbf{u}^P = \frac{u_0^P}{u_0^{LBM}} \mathbf{u}^{LBM} \qquad \mathbf{F}^{LBM} = \frac{(u_0^{LBM})^2 \delta x}{(u_0^P)^2 / l_0^P} \mathbf{F}^P$$

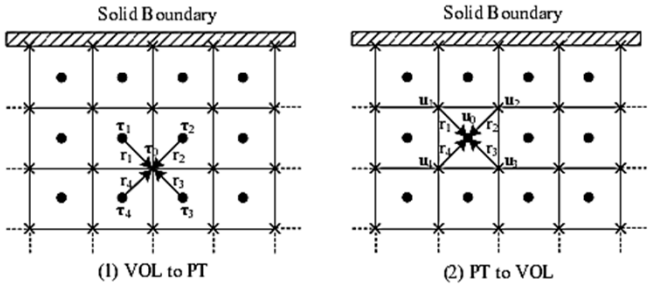
FVM-LBM coupled simulation

Implementation

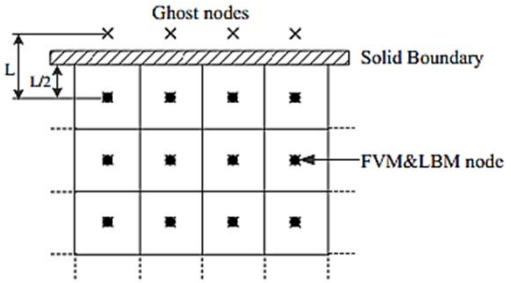
The mesh generating for coupled simulation

- LBM and FVM using **structured mesh with the same size**
- But positions storing the data are **not coincided**
- Thus we have to implement the **interpolation scheme** for the data between FVM and LBM

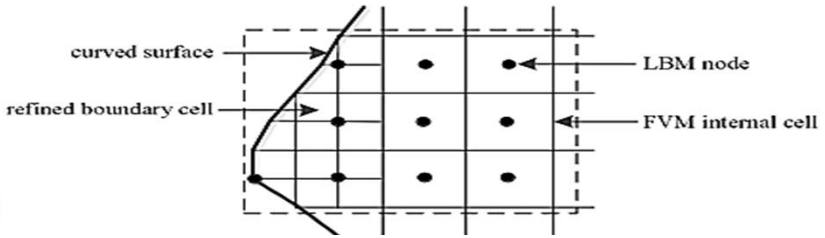
Neighbor interpolation



Processing solid nodes



The hybrid interpolation scheme



FVM-LBM coupled simulation

Implementation

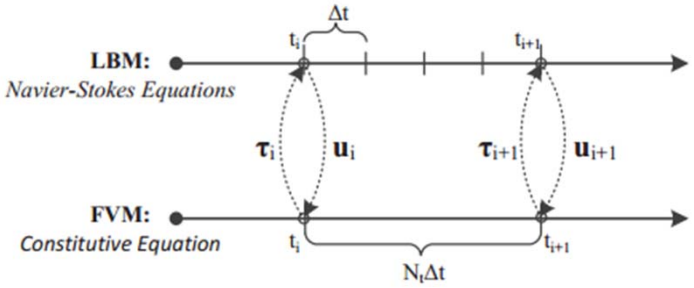
The timestep size

- Physically： LBM relaxation time 1/Re < FVM relaxation time 1/Wi
- Numerically： LBM explicit dt < FVM implicit dt

Simplest strategy: the same timestep size

- Using the LBM timestep size to ensure the time accuracy

The optimized algorithm



Based on Nt=Wi/Re
for the tradeoff between accuracy and stability.

FVM-LBM coupled simulation

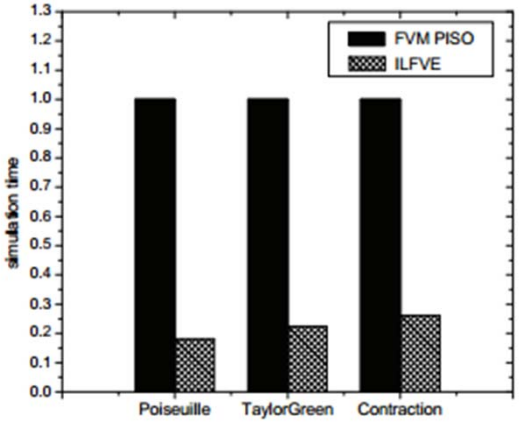
Simulation results

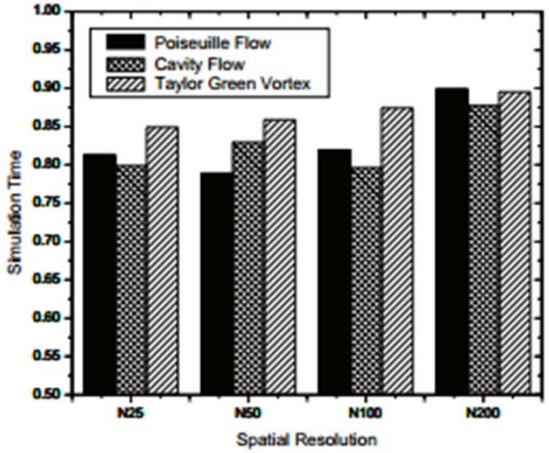
LBM–FVM vs. FVM PISO

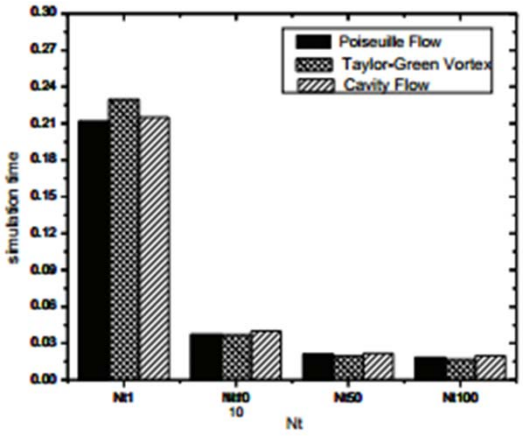
- The coupled simulation time is Only 20% of FVM

Various differencing schemes in LBM–FVM

- Space scheme： Non-interpolation Algo. Is 10% ~20% faster than the interpolation Algo.
- Time step size： while Nt= 10/50/100 , The time cost is only 17.5%/9.9%/8.6% compared with Nt=1.



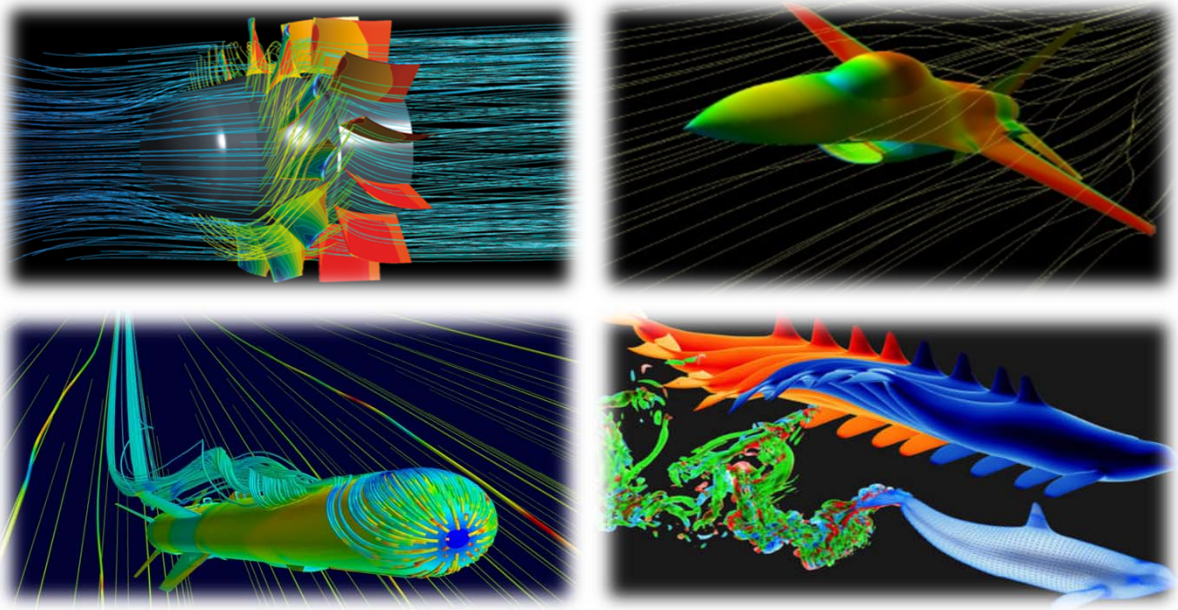




[Journal of Non-Newtonian Fluid Mechanics 211, 99-113.]

Parallel RBF based mesh deformation

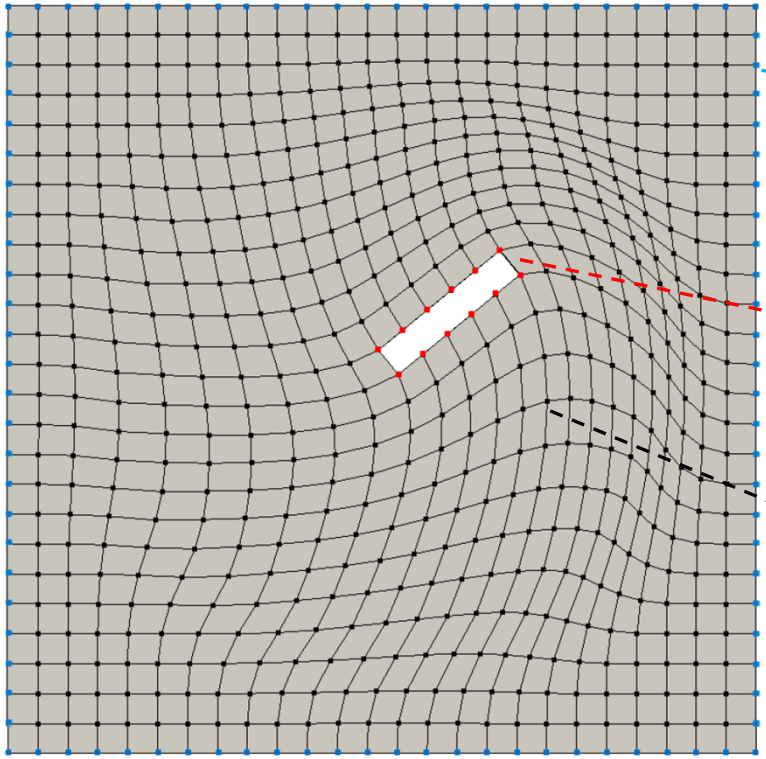
- CFD applications with **moving boundary problem**
  - To keep in accordance with the motion of the boundary, it is often necessary to *update the computational mesh* as the simulation proceeds



Parallel RBF based mesh deformation

RBF based mesh deformation

- A kind of the dynamic mesh methods
- Mesh points are grouped into three categories



Parallel RBF based mesh deformation

RBF based mesh deformation

- The basic idea of the RBF method is to establish such a polynomial fitting equation by the radial basis function and then use the boundary points to interpolate the displacements of the internal points

Radial basis function	
Volume spline(VS)	x
Thin plate spline(TPS)	$x^2 log(x)$
Multiquadratic Bi-harmonics(MQB)	$\sqrt{(a^2+x^2)}$
Inverse Multiquadratic Bi-harmonics(IMQB)	$\sqrt{(a^2+x^2)^{\frac{1}{2}}}$

$$s(x_{in_i}) = \sum_{j=1}^{N_b} \lambda_j \phi(\| x_{in_i} - x_{b_j} \|)$$

$$\lambda = \Phi_{b,b}^{-1} \Delta x_b$$

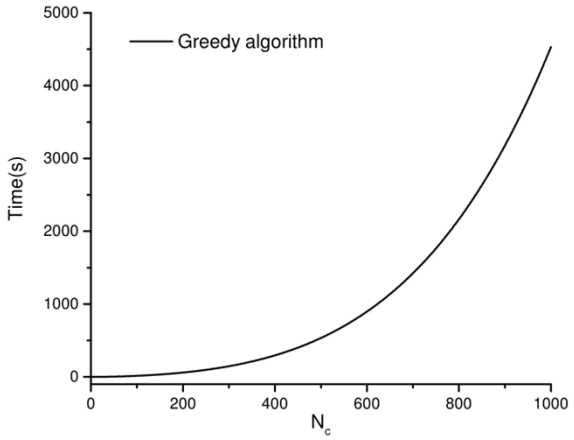
The number of boundary points(N_b) is a dominant factor and larger N_b will result in larger RBF system

Parallel RBF based mesh deformation

- To decrease the calculation amount, an effective way is to reduce N_b in the equation
  - *It is no need to use all the boundary points to interpolate the displacements of the internal points*
    - Select a subset of the boundary as representative
    - Reduced points lead to reduced calculations

Parallel RBF based mesh deformation

- To decrease the calculation amount, an effective way is to reduce N_b in the equation
  - *Original greedy method*:
    - Select one feature point at each loop step till convergence
    - Always get the optimal subset
    - However, the greedy method itself also introduces extra time consumption.



As the number of selected boundary points(N_c) increases, the time consumption will present an **exponential** growth

Parallel RBF based mesh deformation

Error based multi-selection greedy method

- Search through for the points with extreme errors and select multiple points as representative at each loop step

Algorithm 1 The multi-selection greedy algorithm

Require: $x_b, \Delta x_b, \xi_{tol}$
Ensure: $x_c, \Delta x_c$

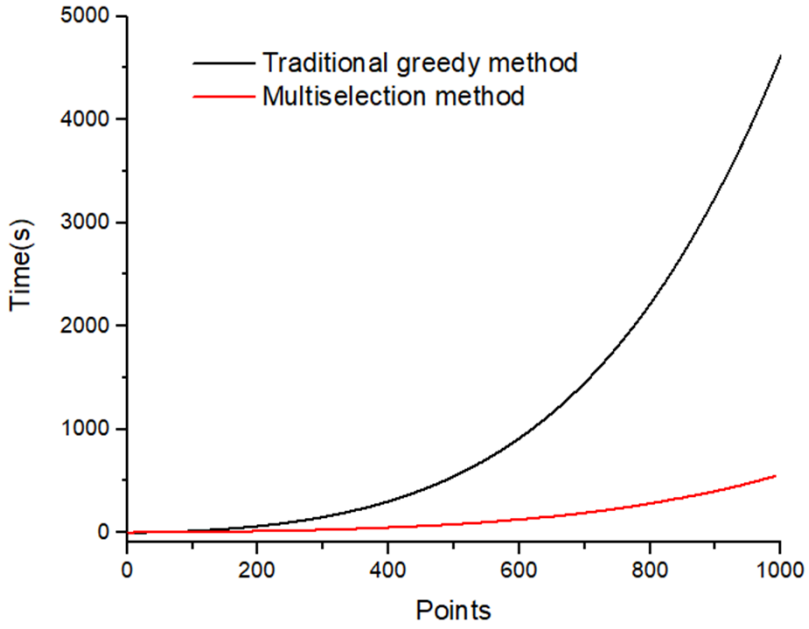
1: neighbours[][] $= Construct(x_b)$
2: $x_c = x_b[0]$ $\Delta x_c = \Delta x_b[0]$
3: **repeat**
4: $\varepsilon_{pre} = \| \varepsilon \|_2 / \| \Delta x_b \|_2$
5: $\lambda_c = \Phi_{c,c}^{-1} \Delta x_c$
6: $\Delta x_b^* = \Phi_{b,c} \lambda_c$
7: $\varepsilon = \Delta x_b^* - \Delta x_b$
8: $\varepsilon_{cur} = \| \varepsilon \|_2 / \| \Delta x_b \|_2$
9: $\varepsilon_{mxm}[] = 0$ indices$_{mxm}[] = 0$
10: **for** $i = 0$ to $N_b - 1$ **do**
11: **if** $\| \varepsilon[i] \| > \| \varepsilon$[neighbours$[i][0]] \|$ && ... && $\| \varepsilon[i] \| > \| \varepsilon$[neighbours$[i][end]] \|$ **then**
12: $\varepsilon_{mxm} = (\varepsilon_{mxm}, \varepsilon[i])$ indices$_{mxm}$ = (indices$_{mxm}, i$)
13: **end if**
14: **end for**
15: Select multiple control points based on ε_{mxm} and indices$_{mxm}$
16: **until** $\varepsilon_{cur} < \xi_{tol}$

Parallel RBF based mesh deformation

Error based multi-selection greedy method

- Search through for the points with extreme errors and select multiple points as representative at each loop step

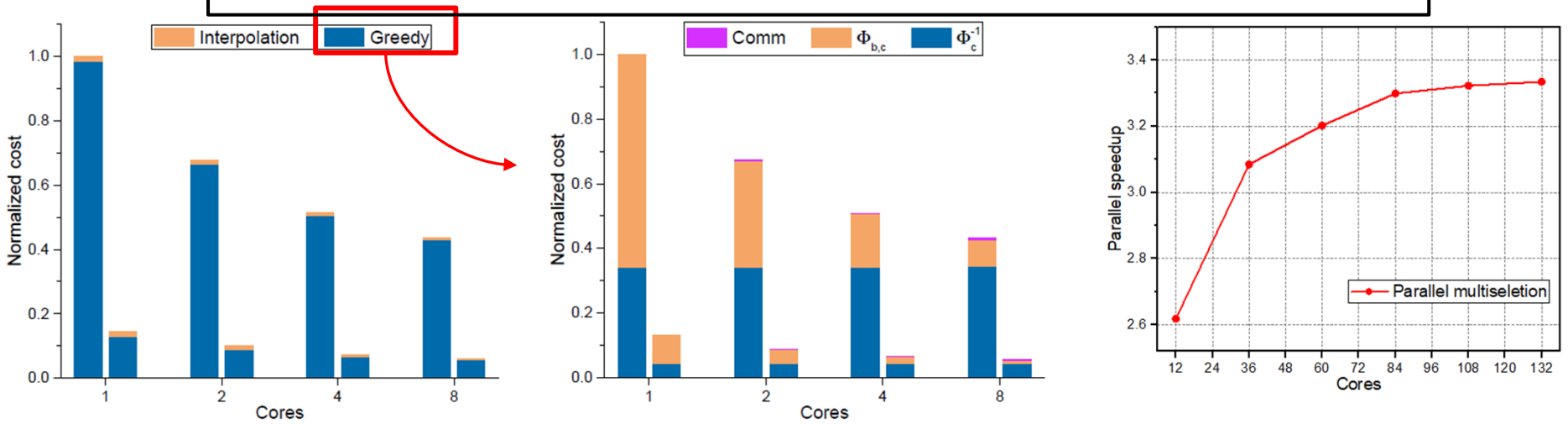
Compared to the original greedy method, The time consumption is reduced from **exponential** growth to **polynomial** growth



Parallel RBF based mesh deformation

- **Further optimization for parallel computing**
  - *Solve Wound boundary inconsistency problem*
  - *Load imbalance*

Chao Li, Xinhai Xu*, etc, A parallel multi-selection greedy method for the radial basis function based mesh deformation, INTERNATIONAL JOURNAL FOR NUMERICAL METHODS IN ENGINEERING



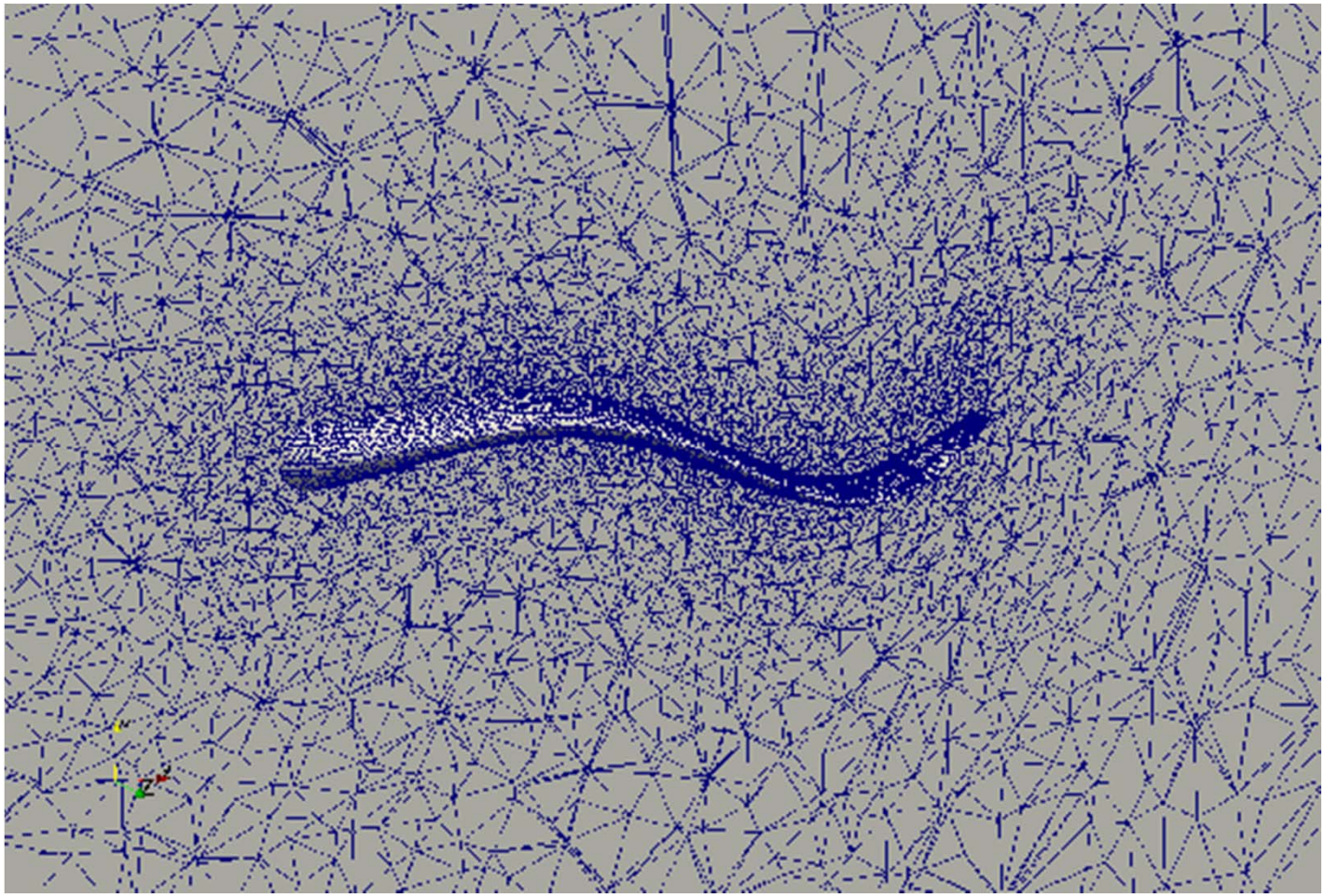
Point selection time is reduced as the parallelism increases

The parallelism could be extend to hundreds of cores

Parallel RBF based mesh deformation

☐Mesh deformation view

■A 3D undulating fish



Part II

- Start with application development on OpenFOAM
- High order discretization of HopeFOAM
  - Motivation of HopeFOAM
  - DG method and HopeFOAM-0.X
  - Some supporting techniques
  - User interfaces
- HopeFOAM parallel optimization

Motivation

Discrete method of OpenFOAM

FVM: 2nd order discretization

- much more points to achieve given accuracy

$$\text{Work} \propto (2m)^d \nu \left(\frac{\nu}{\varepsilon_p}\right)^{\frac{d+1}{2m}}$$

  - 2m = order of the scheme
  - d = dimension of the problem

- For cases with high accuracy required, the error limit is very small, and the second term is much larger than the first one.

Overall overhead will drop dramatically when using high order method

HopeFOAM

- **HopeFOAM is a major extension of OpenFOAM to provide higher order numerical methods for computational mechanics.**
  - ***Features***
    - **High Order discretization: integrates high-order discretization methods into the computational mechanics Toolbox**
    - **High Performance: integrates parallel computational toolkits/software to accelerate the discretization and computational procedures**
    - **High Extensibility: provides an extensible software framework for further development of application module and easy-to-use interfaces for developers**

DG method

Comparison of common differencing methods

DG(DG-FEM): Discontinuous Galerkin Method

	Complex geometries	High-order accuracy and *hp*-adaptivity	Explicit semi-discrete form	Conservation laws	Elliptic problems
FDM	×	✓	✓	✓	✓
FVM	✓	×	✓	✓	(✓)
FEM	✓	✓	×	(✓)	✓
DG-FEM	✓	✓	✓	✓	(✓)

× : Not suited

✓ : Suited

(✓) : Suited, possibly with modifications, but not the most natural (or efficient) choice

Taken from Hesthaven and Warburton, 2008

DG Method is the best one for High Order discretization in theory

DG method

Basic principles of DG

Nodal DG

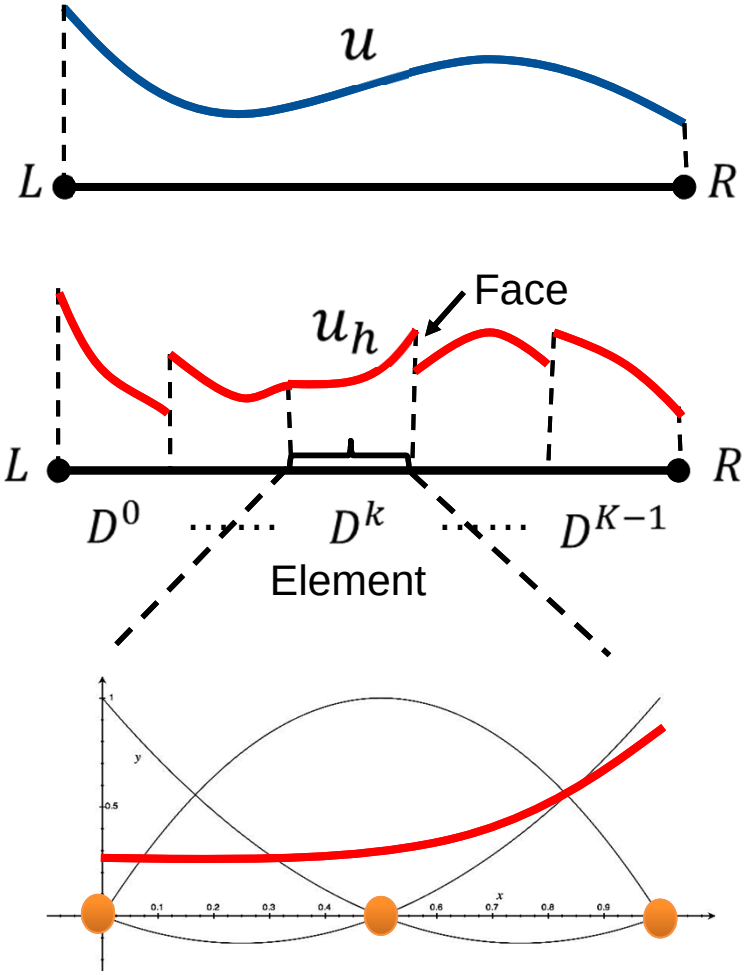
- Piece-wise polynomial
- Element k:

$$u_h^k(x_i) = \sum_{j=1}^{N_p} \hat{u}_j^k \cdot \varphi_j^k(x_i)$$

$\varphi\_j\hat{}k\ (x\_i)$—basis function

$\hat{u}\_j\hat{}k$ -- nodal value of field u

N_p -- number of basis points



DG(FVM + FEM), is the first high order scheme in HopeFOAM (HopeFOAM-0.x)

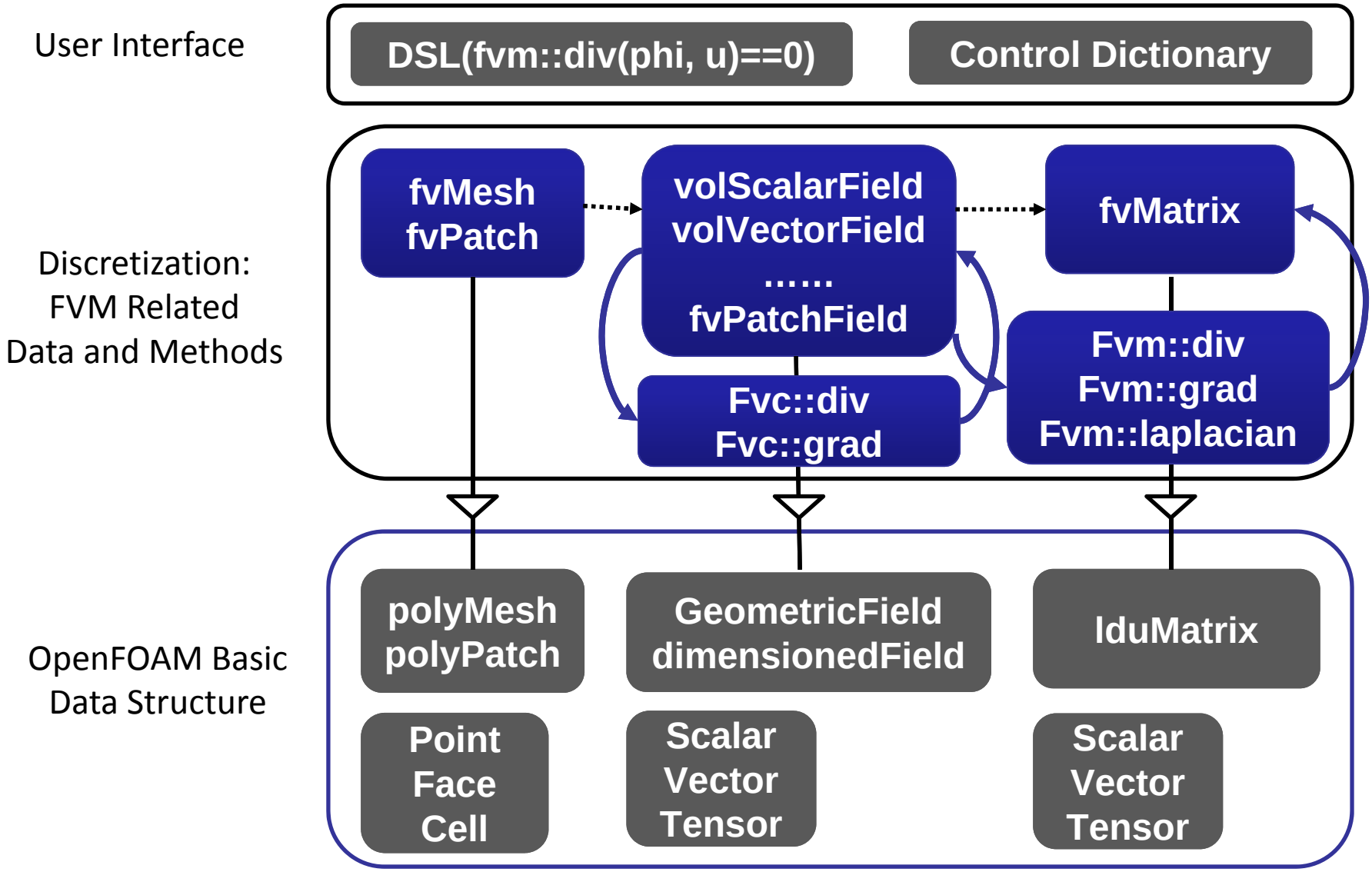
HopeFOAM-0.X

Design philosophy

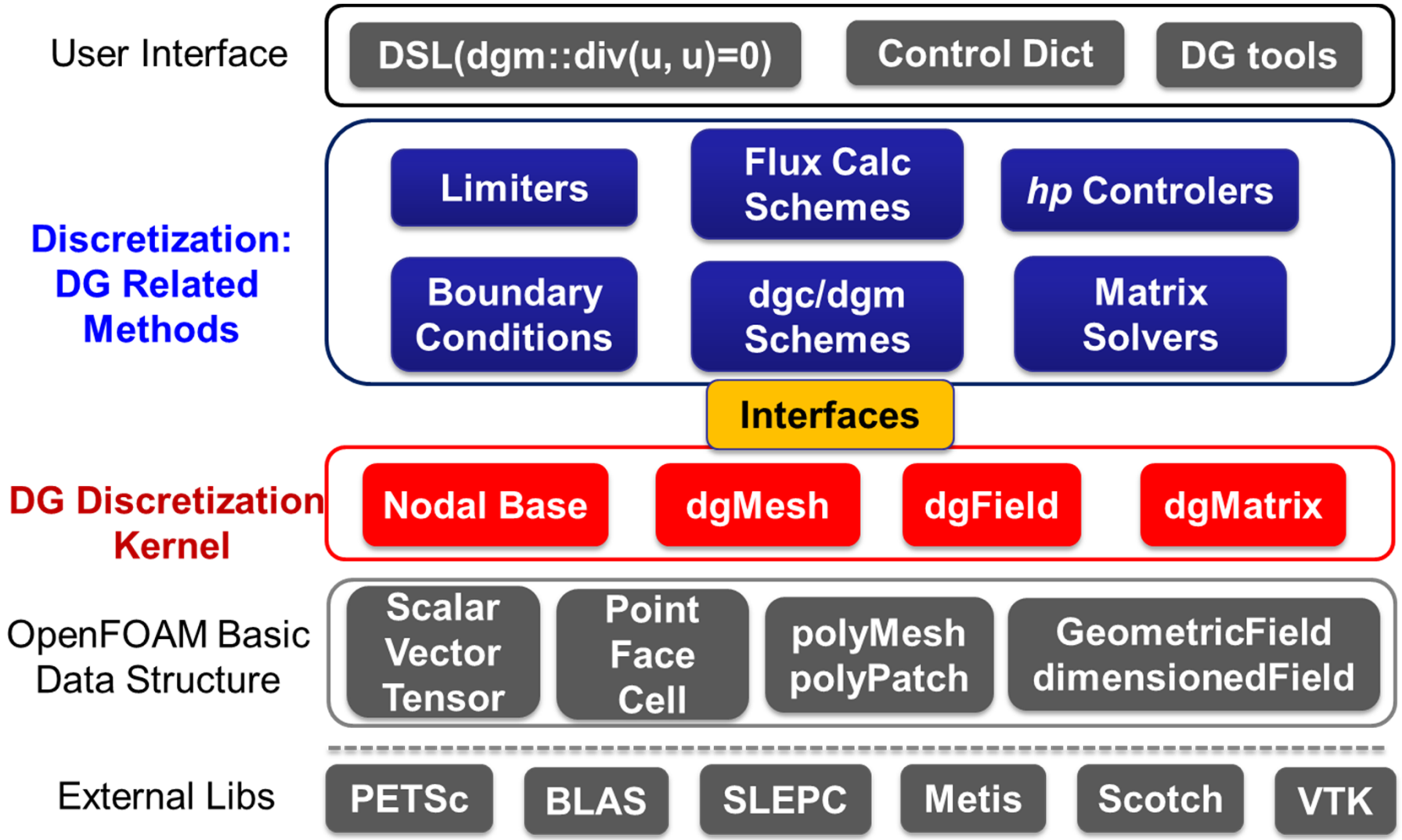
Compatible with OpenFOAM

- Reuse the framework and data structure
- Keep consistent with the user interface
- Inherit the pre and post processing tools

Framework of OpenFOAM



Framework of HopeFOAM-0.1



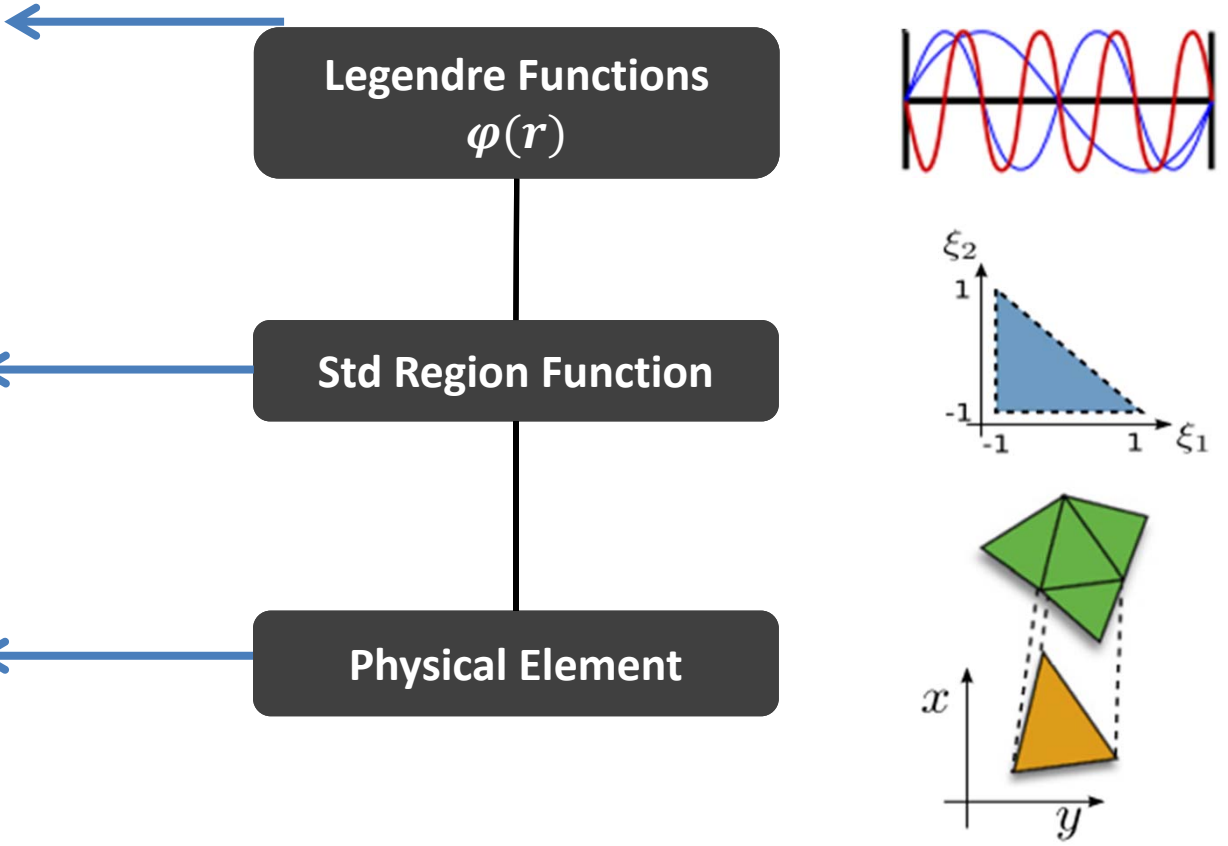
DG Discretization Kernel

Basis functions are constructed in three levels

provide high order analytical solution

provide information about element shape, polynomial order, dof locations and so on.

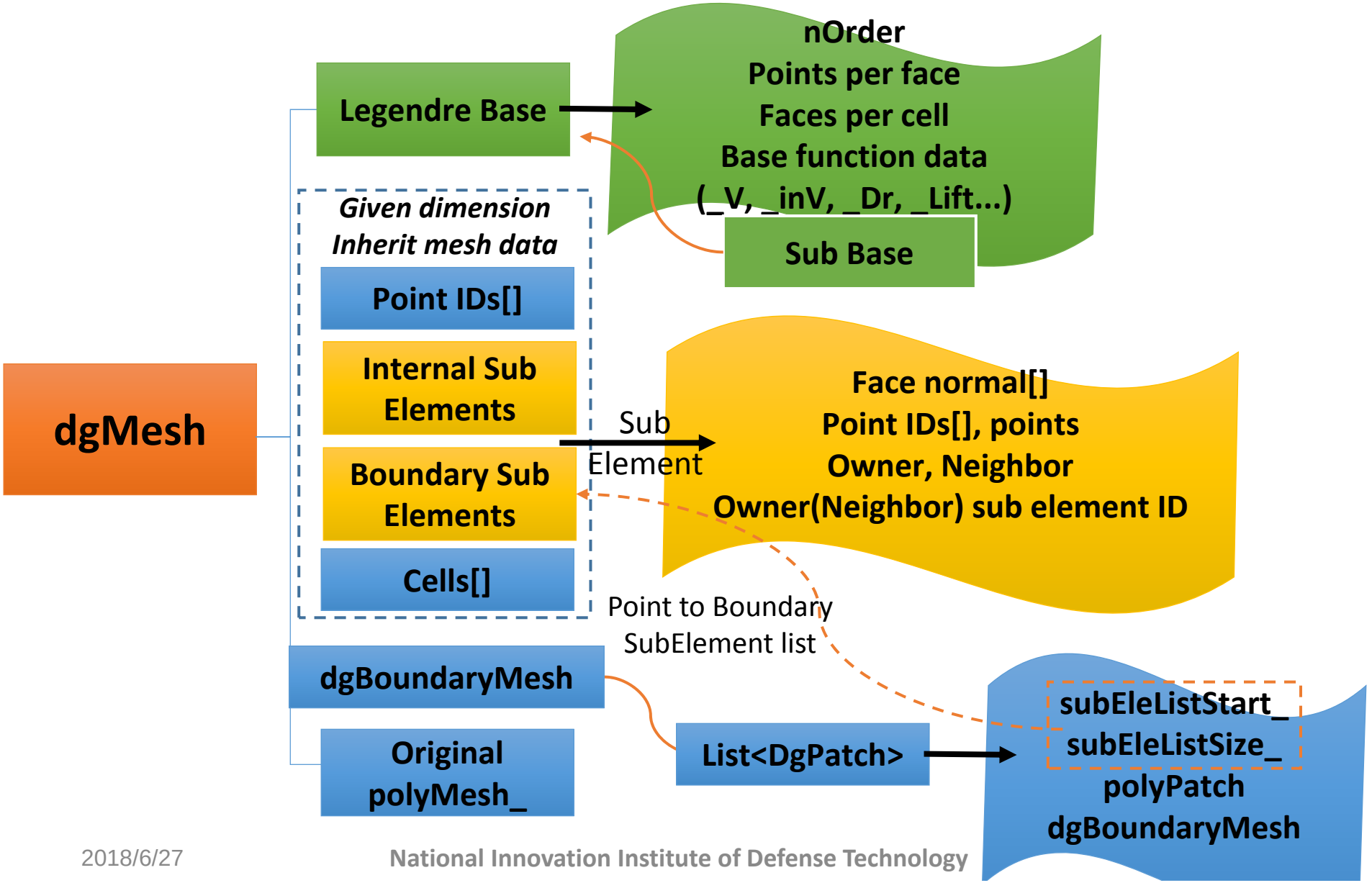
contain data of cell derivative and integration, including jacobian matrix and gauss quadrature weight coefficients



DG Discretization Kernel

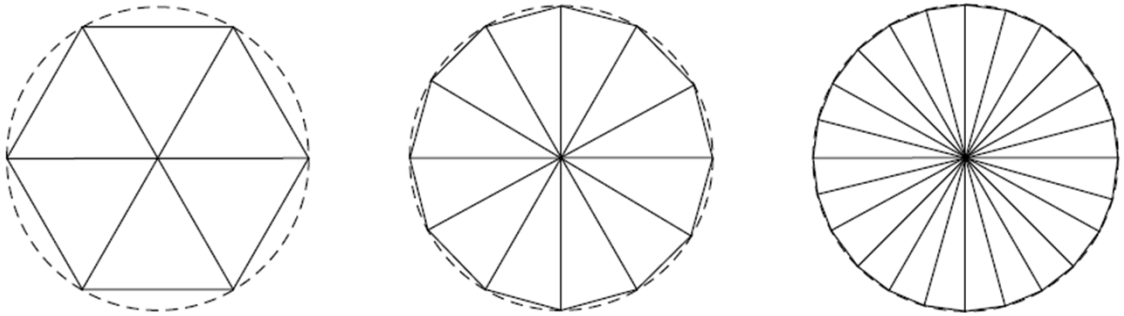
- Key data structure design
  - *dgMesh*
    - specific n-d extraction
    - DG discretization data
    - Provide Dof mapping for cell and face
  - *dgBoundaryMesh*
    - List<dgPatch>
  - *dgPatch*
    - Reuse fvPatch
    - Additional data structures: Collection of physical sub-element_ for boundary faces
  - *dgField*
  - *dgMatrix*
  - ……

DG Discretization Kernel

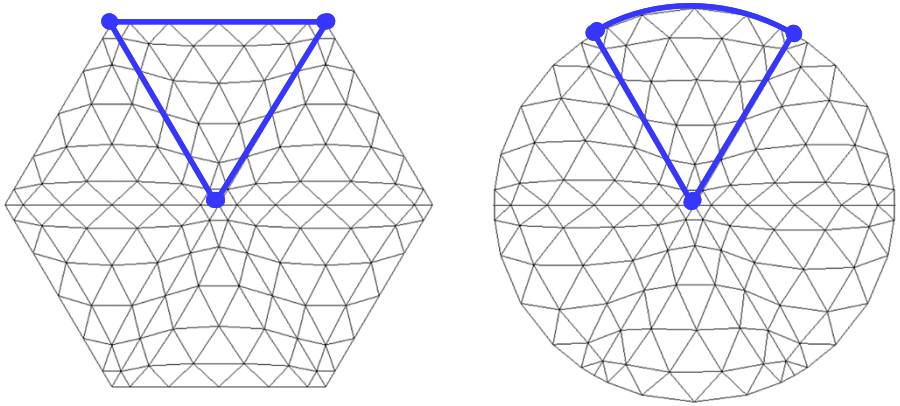


Supporting techniques

- **Curved mesh** is critical to achieve high order convergence rate
  - OpenFOAM: *refine mesh*

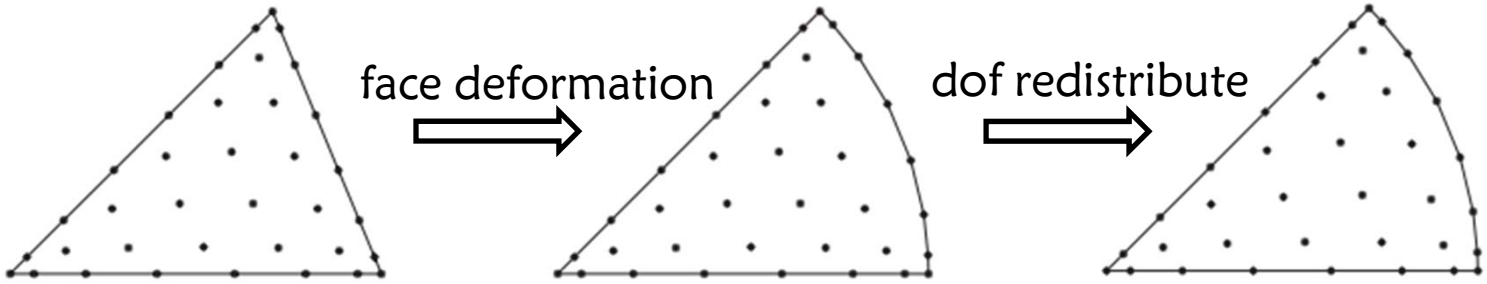


  - DGM: *insert high-order points on a grid cell*
    - 6 cells cavity with 6^{th} basis



Support of curved mesh

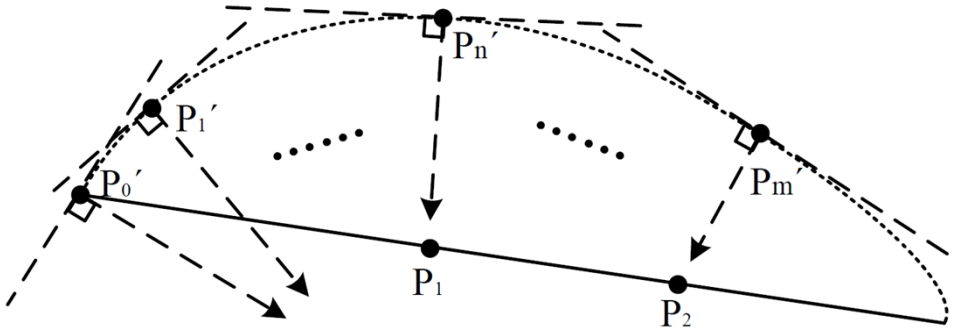
- **Moving the dof points in boundary cells**
  - First, the dof points in original straight side are shifted to fit the curve.
  - Then, all inner dof points are redistributed.



Support of curved mesh

Moving the dof points in boundary cells

- **Key Rule:** For a high-order point P, choose a point P' on the curve boundary, whose **normal** passes P, and replace P with P'



Algorithm 1 High-order Points Moving Algorithm

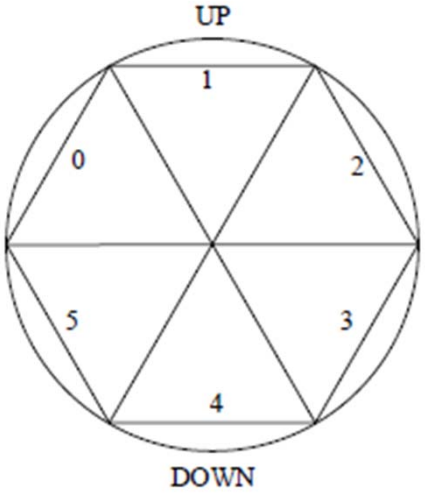
```
Get curved boundary faces faces;
Get curved boundary equations;
Set threshold threshold;
for all face in faces do
    Get initial point P';
    Get high-order points highOrderPoints of face;
    for all P in highOrderPoints do
        Initial res = 0;
        repeat
            Correct P' according to | res |;
            Calculate unit normal vector n of P';
            Calculate res= n × PP';
        until | res |< threshold
        Mapping P to P';
    end for
end for
```

Support of curved mesh

Curve Boundary Interface

- **new boundary type: arc**
- **The formulation of the curve are described by codeStream set in configuration file**
  - C++ code can be compiled in *boundary* file
  - Parameter: u, v

[published on ICA3PP 2017]




```
UP
{
    type            arc;
    nFaces          3;
    startFace       0;
    name            codeup;
    u_Range         (-0.5  0.5);
    v_Range         (0  0);
    code
    #{
        sin(pi*u),
        cos(pi*u),
        v
    #};
}
```

Range

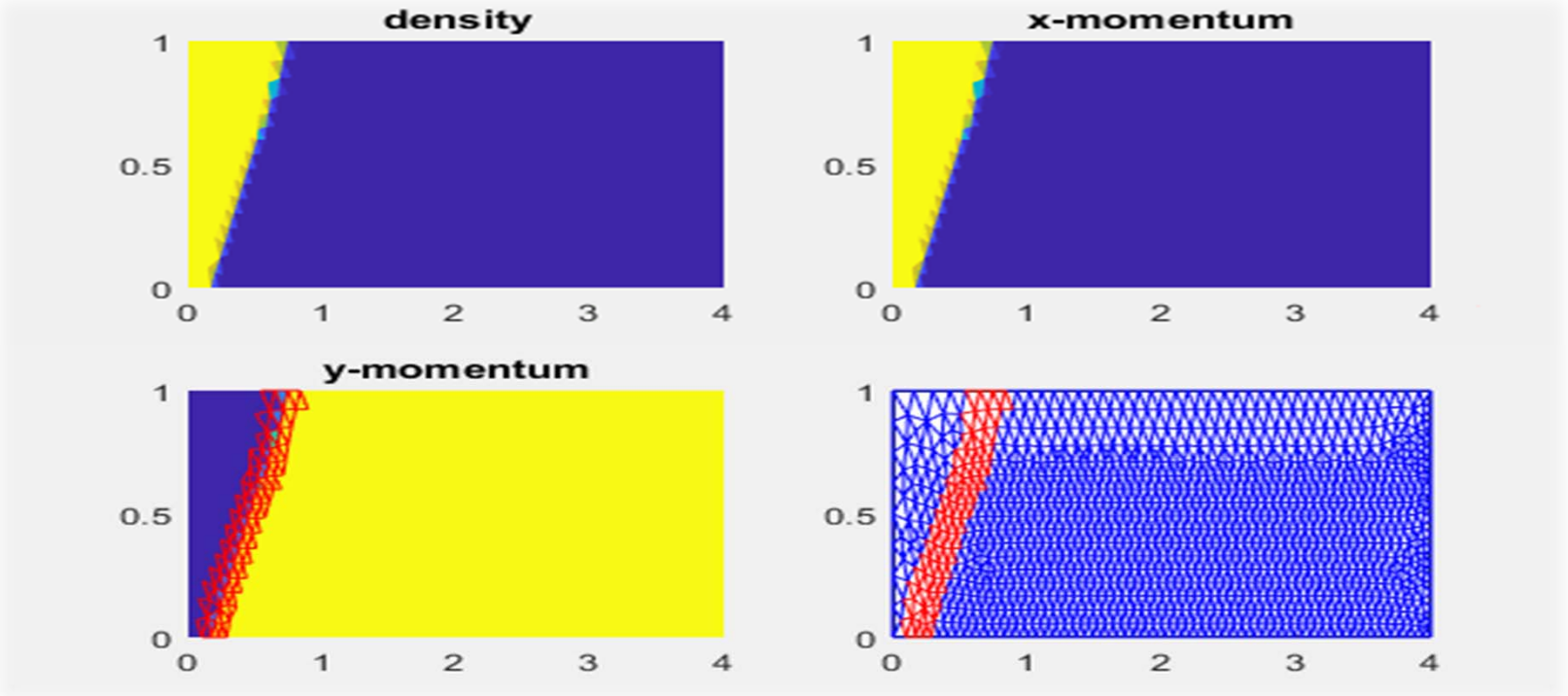
Code

Supporting techniques

- high order limiter + shock detector
  - *shock detector: find regions that need limit*
  - *high order limiter: retain high order accuracy of solution*
  - *We are working on*
    - **KXRCF detector: the quantities of density and energy flowing out the target cell decide whether it should be limited or not**
    - **WENO limiter: reconstruct the polynomial in the troubled cell using the WENO reconstruction procedure**

Support of Limiters

- **Double Mach simulation using shock detector and WENO limiter**
  - *picked cells are marked with red color.*
  - *The shock frontiers are accurately captured by our KXRCF detector.*



User interfaces

DSL in DG (Euler equation)

- We designed corresponding DSLs under the name space of **dgc and dgm** to support DG discretization

$$\frac{\partial \mathbf{q}}{\partial t} + \frac{\partial \mathbf{F}}{\partial x} + \frac{\partial \mathbf{G}}{\partial y} = 0,$$

where

$$\mathbf{q} = \begin{pmatrix} \rho \\ \rho u \\ \rho v \\ E \end{pmatrix}, \mathbf{F} = \begin{pmatrix} \rho u \\ \rho u^2 + p \\ \rho uv \\ u\left(E + p\right) \end{pmatrix}, \mathbf{G} = \begin{pmatrix} \rho v \\ \rho uv \\ \rho v^2 + p \\ v\left(E + p\right) \end{pmatrix},$$

```
dg::solve(dgm::Ddt(rho1) + dgc::Div(gther_U, rho1, Godunov.fluxRho()));

dg::solve(dgm::Ddt(rhoU1) + dgc::Div(gther_U, rhoU1, Godunov.fluxRhoU()) + dgc::Grad(gther_p));

dg::solve(dgm::Ddt(Ener1) + dgc::Div(gther_U, Ener1, Godunov.fluxEner()) + dgc::Div(gther_U, gther_p));
```

User interfaces

Control dictionary mechanics

- system/controlDict, system/decomposeParDict
  - Similar with original OpenFOAM
- system/dgSolution

```
DG
{
    meshDimension       2;
    baseOrder           3;
}
```

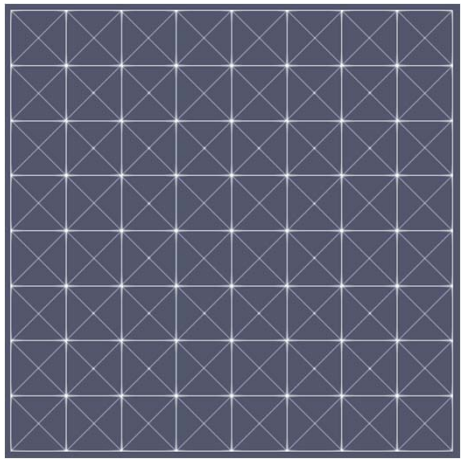
```
p
{
    kspSolver       cg;
    kspPC           hypre-boomeramg;
    tolerance       1e-12;
    relTol          0;
}
```

- system/dgSchemes

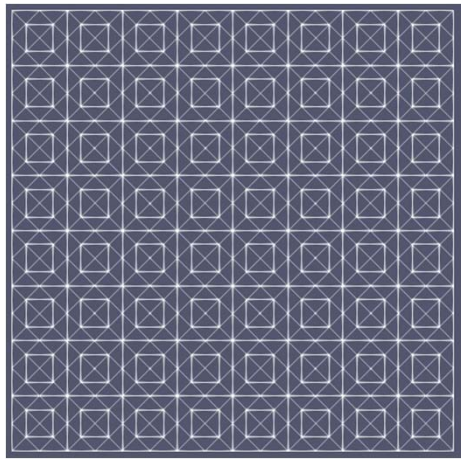
```
GodunovScheme
{
    fluxScheme    Roe;
    limiteScheme Triangle;

}
```

Preliminary results

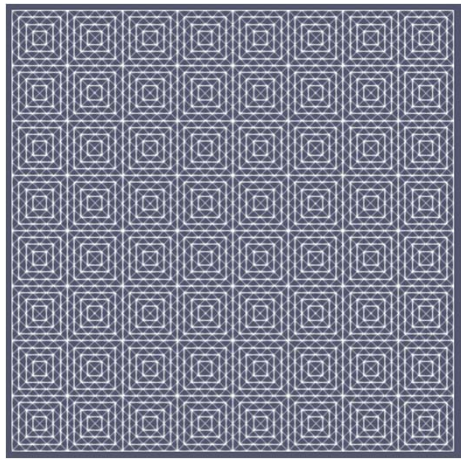
Isentropic vortex problem, inviscid



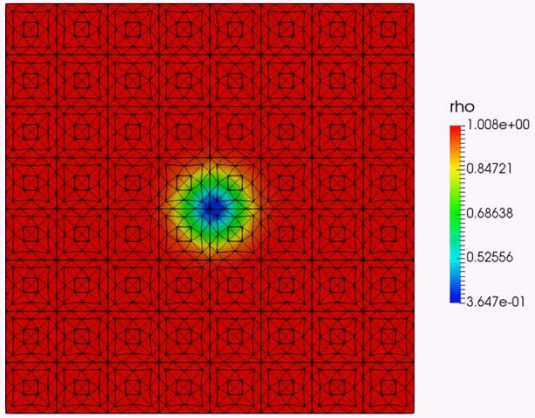
h (256)



h/2 (1024)



h/4 (2028)



Result with 256 cells and 3rd basis
High order post processing

Preliminary results

Spatial convergence rate test

Euler solver with Roe flux

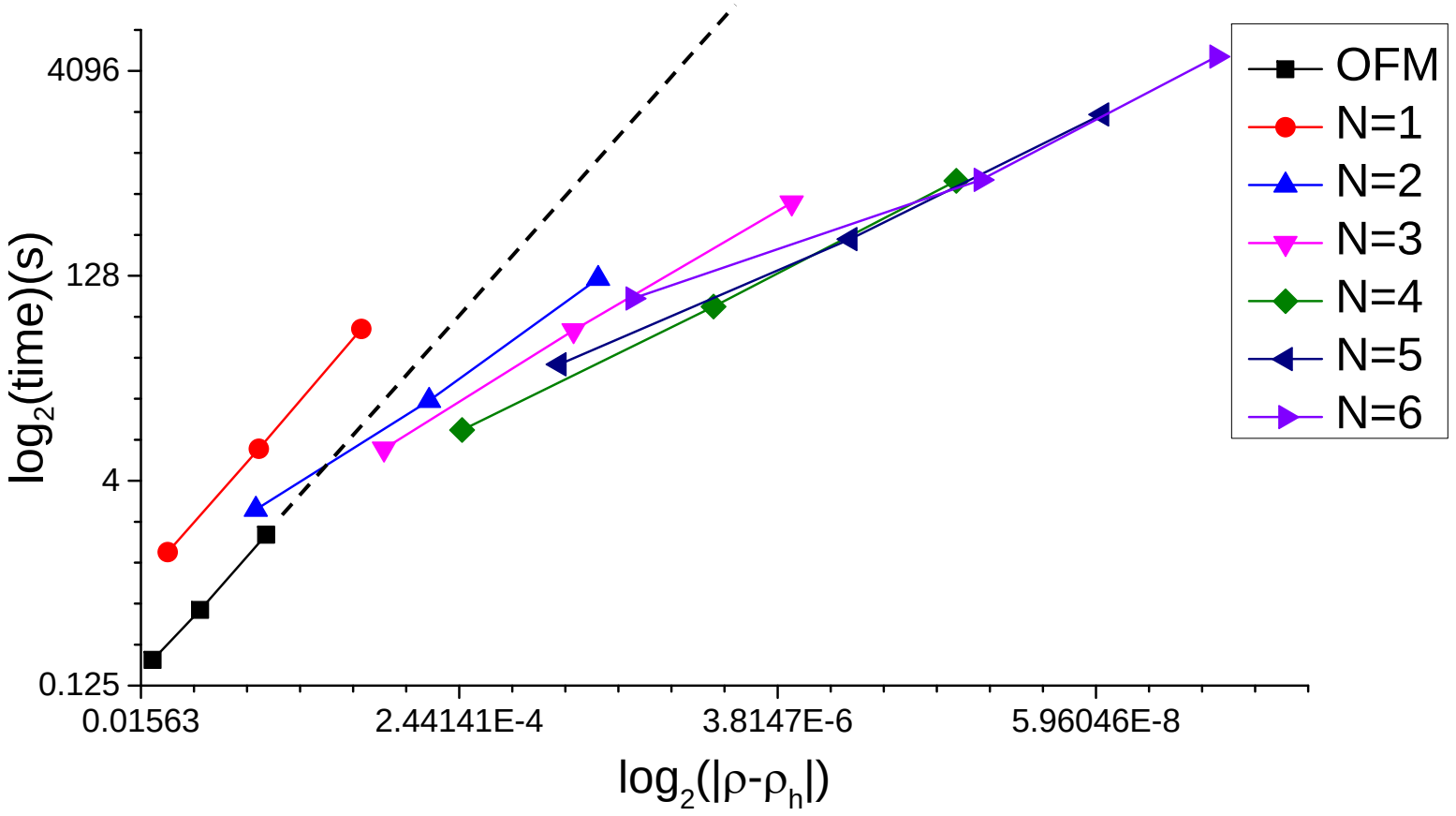
N	Error in ρ				Error in ρu			
	h	h/2	h/4	**Rate**	h	h/2	h/4	**Rate**
1	1.101E-02	3.344E-03	8.759E-04	**1.83**	2.361E-02	7.182E-03	1.793E-03	**1.86**
2	3.481E-03	3.621E-04	3.972E-05	**3.23**	6.384E-03	8.060E-04	9.689E-05	**3.02**
3	6.523E-04	5.471E-05	3.173E-06	**3.84**	1.610E-03	1.226E-04	7.141E-06	**3.91**
4	2.352E-04	8.808E-06	3.690E-07	**4.66**	5.187E-04	1.866E-05	8.533E-07	**4.62**
5	6.597E-05	1.477E-06	5.505E-08	**5.11**	1.388E-04	2.695E-06	1.591E-07	**4.88**
6	2.530E-05	2.675E-07	1.232E-08	**5.50**	4.675E-05	4.760E-07	3.837E-08	**5.13**
OFM-FVM	1.340E-02	7.209E-03	3.039E-03	**1.07**	3.466E-02	1.733E-02	6.945E-03	**1.16**

Preliminary results

Error-Cost test

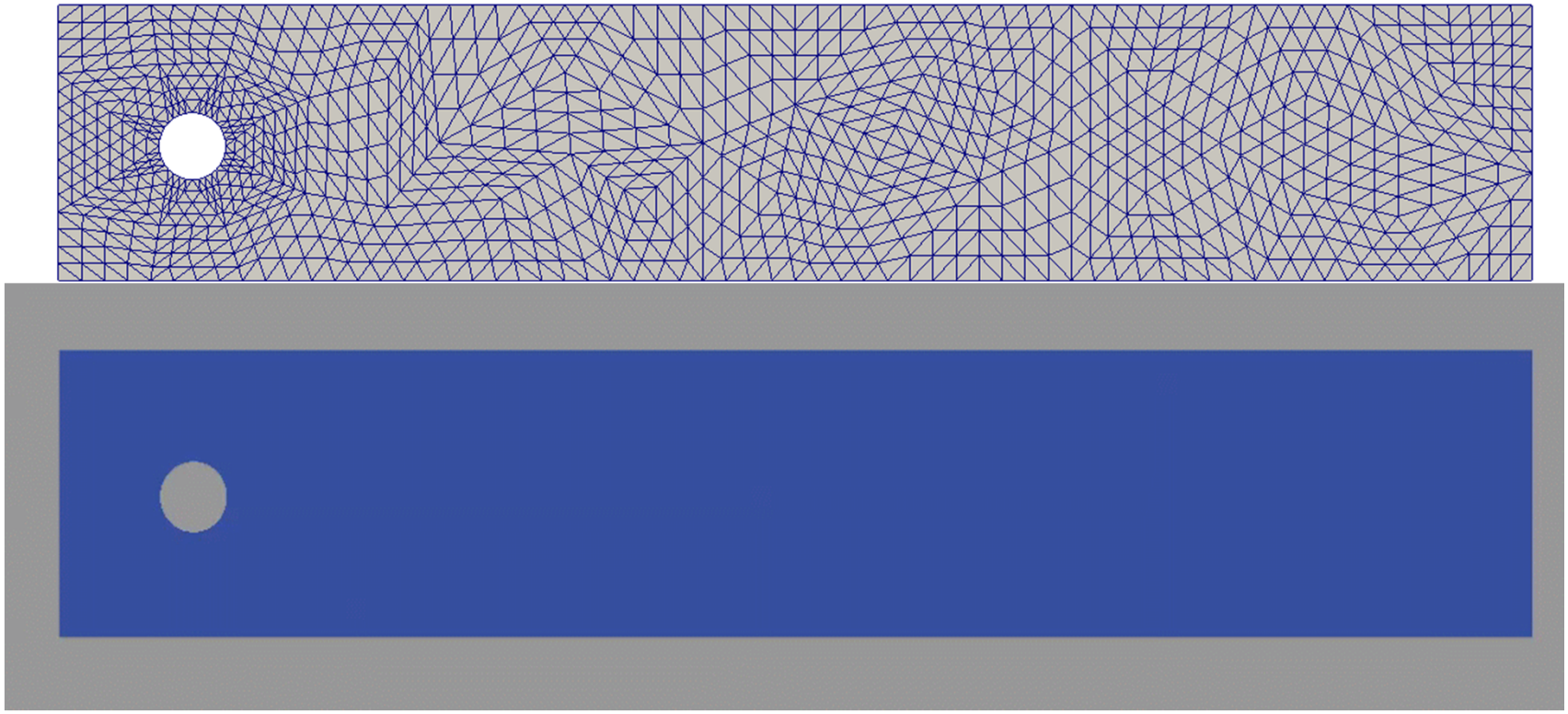
A popular misunderstanding:

High order method is computational expensive?



Preliminary results

- Flow past cylinder, Re=100
  - INS solver with pressure projection method
  - 1840 cells with 3rd basis
  - Curved mesh in cylinder patch



HopeFOAM-0.1

- More details can be found on our project on Github

HopeFOAM: High Order Parallel Extensible CFD software

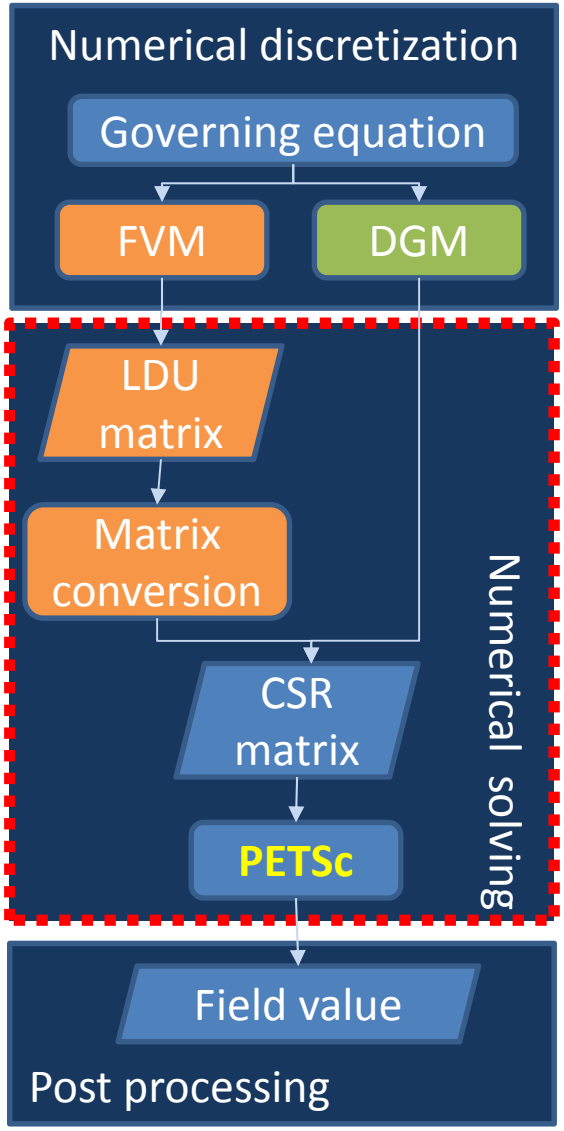
```
-------------------------------------------------------------------------------
|     __  ______  ____  ______ |                                             |
|    / / / / __ \/ __ \/ ____/ | HopeFOAM: High Order Parallel Extensible CFD Software |
|   / /_/ / / / / /_/ / __/    |                                             |
|  / __  / /_/ / ____/ /___    |                                             |
| /_/ /_/\____/_/   /_____/    | Copyright(c) 2017-2017 The Exercise Group, AMS, China.|
|                              |                                             |
-------------------------------------------------------------------------------

TITLE:     README for [[ https://github.com/HopeFOAM/HopeFOAM [HopeFOAM-0.1]]
AUTHOR:              The Exercise Group
DATE:                15th September 2017
LINK:        https://github.com/HopeFOAM/HopeFOAM
```

Part II

- Start with application development on OpenFOAM
- High order discretization of HopeFOAM
- HopeFOAM parallel optimization
  - *PETSc-based numerical module*
  - *Matrix optimization based on Matrix-Free*
  - *Communication optimization*

PETSc-based numerical module

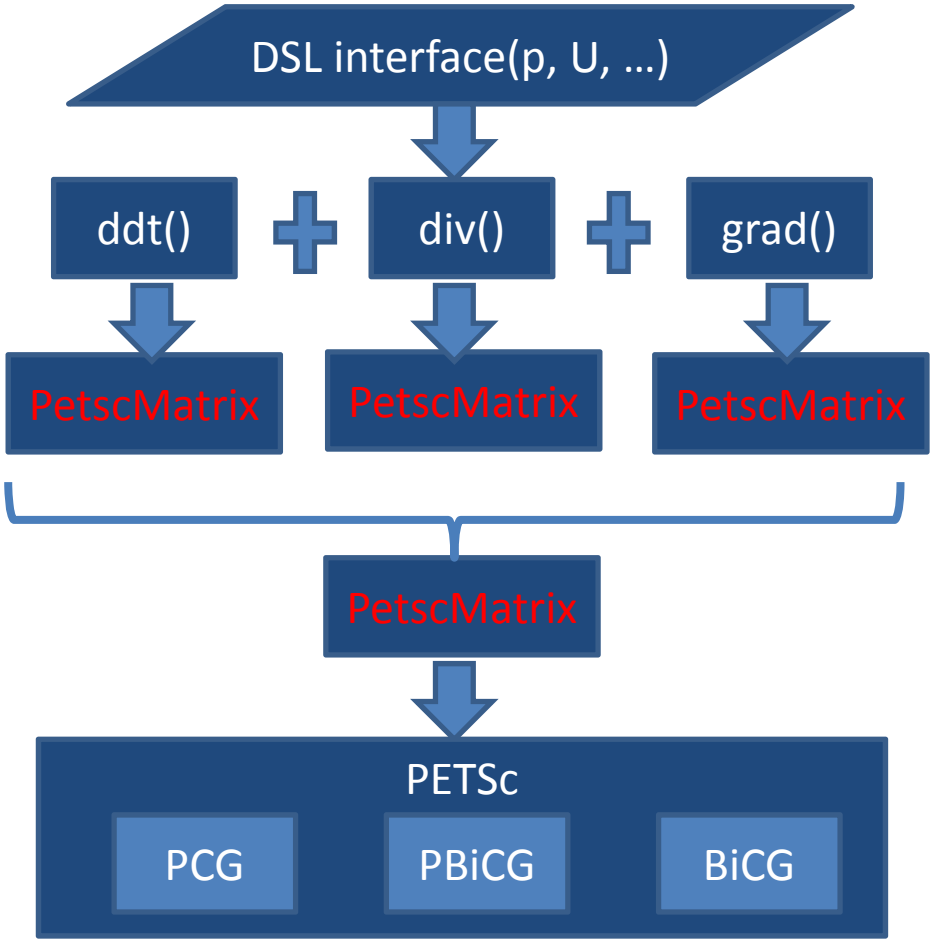
- **Numerical solving**
  - *is the core of CFD simulation(50%+ time cost)*
- **Design in HopeFOAM(PETSc)**
  - *Flexible*
    - Matrix is stored in Compressed Sparse Row(CSR) format
  - *Efficient*
    - PETSc provides a lot of numerical solution methods
  - *Common*
    - High order scheme (DGM)
    - FVM



Support for DGM

Design & Implementation

- **Main matrix class**
  - PetscMatrix
- **Member variables**
  - PETSc matrix: Mat
  - PETSc vector: Vec
- **Member functions**
  - discrete operators
    - ddt()→PetscMatrix
    - div()→PetscMatrix
    - grad()→PetscMatrix
    - and so on ……



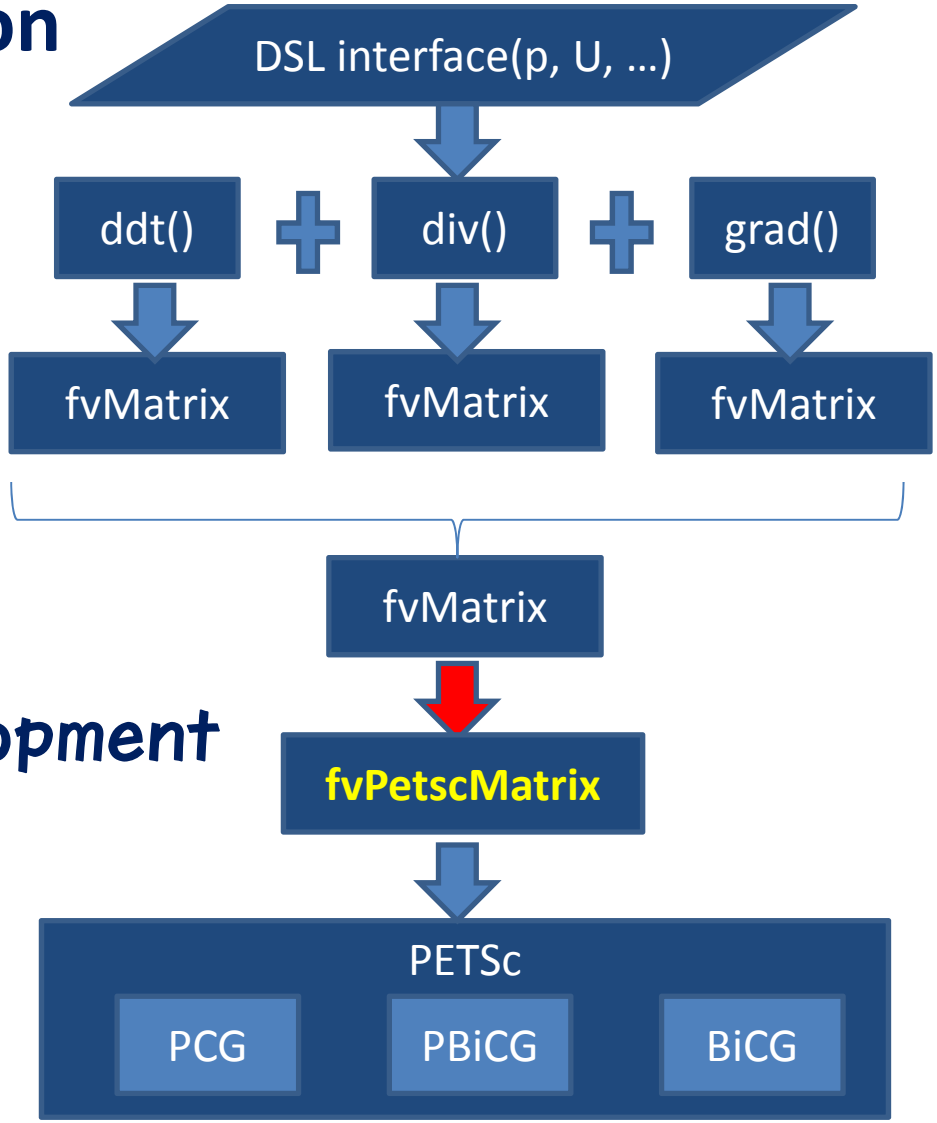
Support for FVM

Design & Implementation

- Main matrix class
  - fvPetscMatrix
- Member variables
  - PETSc matrix: Mat
  - PETSc vector: Vec
- Matrix conversion

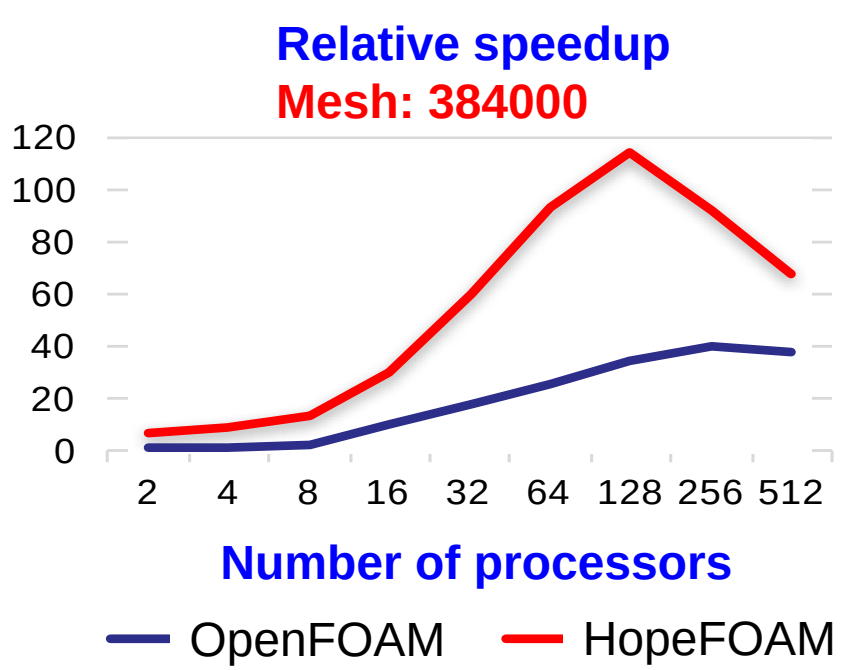
(considering the development cost and usage cost)

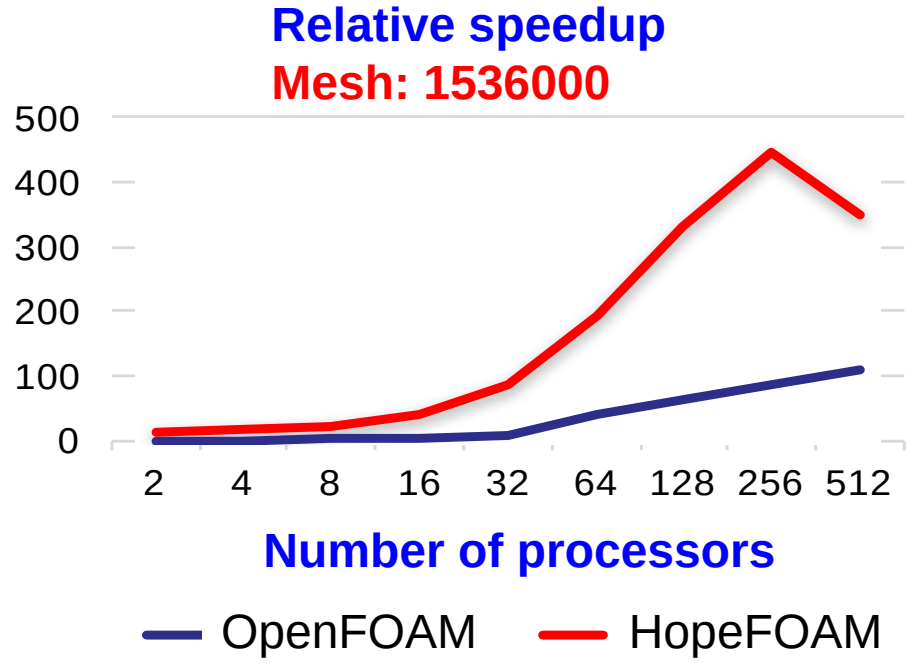
  - fvMatrix→fvPetscMatrix
  - Convert matrix format from LDU to CSR



Performance test

- Benchmark case (Flow around a cylinder) is tested with both OpenFOAM and HopeFOAM
  - *numerical solution performance of HopeFOAM is better*



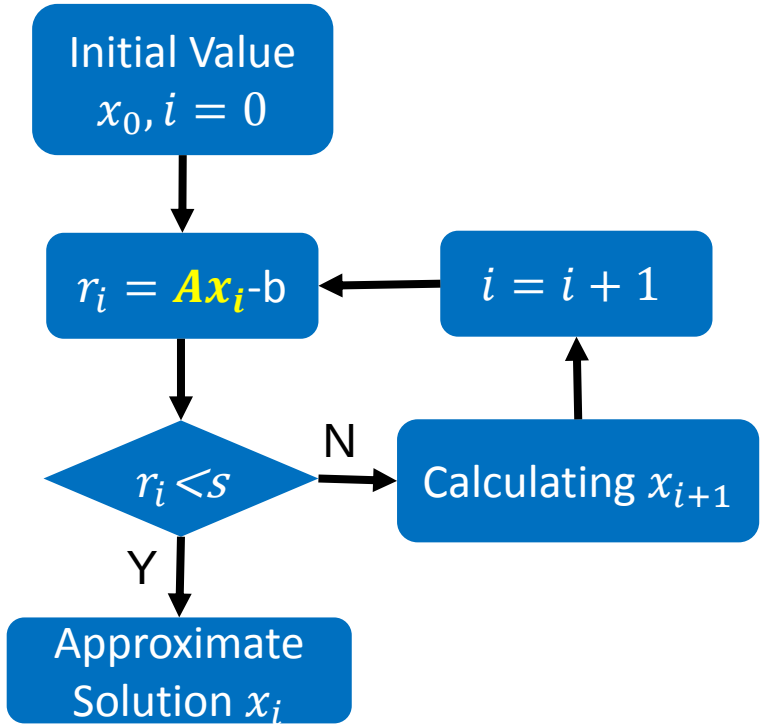


[Insertion of PETSc in the OpenFOAM Framework. ACM Transactions on Modeling and Performance Evaluation of Computing Systems, 2017]

Matrix Optimization Based on Matrix-Free

Traditional Iteration Algorithm

- lots of Matrix-Vector Multiplication operations
- Finite element software
  - Because of memory access due to these MV Multiplication
  - CPU utilization rates rarely exceed 20%



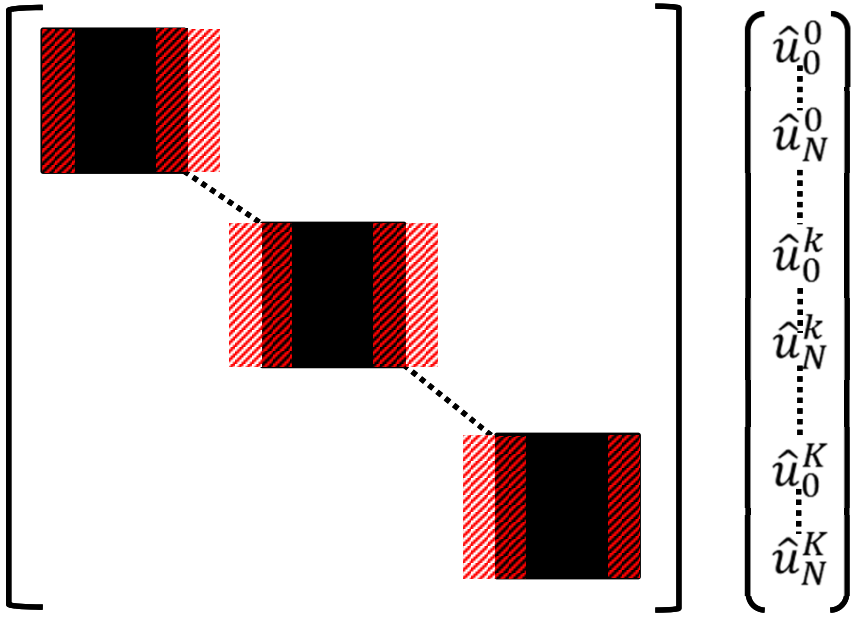
Matrix-Free：Improve CPU utilization by optimizing coefficient matrix memory overhead

Matrix Optimization Based on Matrix-Free

Discretization in DG

- Based on Cartesian Mesh
- The global coefficient matrix A consists of all cells coefficient matrix A_{cell} as below

$$A = \sum_{\text{cell}=1}^{\text{n\_cells}} P^{T}_{\text{cell,loc}-\text{glob}} A_{\text{cell}} P_{\text{cell,loc}-\text{glob}}.$$



- Each cell corresponds to a dense matrix
- The size of dense matrix is related to DoFs of cells

Matrix Optimization Based on Matrix-Free

Discretization in DG

- A_{cell} is a matrix of nDof*nDof (nDof is DoFs of a cell)

The computational complexity and storage complexity of $A_{cell} \cdot u_{cell}$ are both $O(nDOf^2)$

- A_{cell} is calculated from three parts: Basis Function Matrix(dominates the memory consumption), Jacobi Matrix and Integral Weight Vector
  - Basis Function Matrix: nDof*nDof, $O(nDof^2)$
  - Jacobi Matrix: Elements are equal in Cartesian Mesh, $O(1)$
  - Integral Weight Vector： 1*nDof, $O(nDof)$

Matrix Optimization Based on Matrix-Free

Matrix-Free Implementation*

- In the Cartesian mesh, Basis Function Matrix is obtained by the tensor product

$$V_{2D} = V_{1D} \otimes V_{1D} \qquad V_{3D} = V_{1D} \otimes V_{1D} \otimes V_{1D}$$

- Tensor Product

$$(V_{1D} \otimes V_{1D})x = (V_{1D}(V_{1D}X)^T)^T$$

$$(V_{1D} \otimes V_{1D} \otimes V_{1D})x = (V_{1D}(V_{1D}(V_{1D}X)^T)^T)^T$$

$A_{cell} \cdot u_{cell}$ **with Matrix-Free**

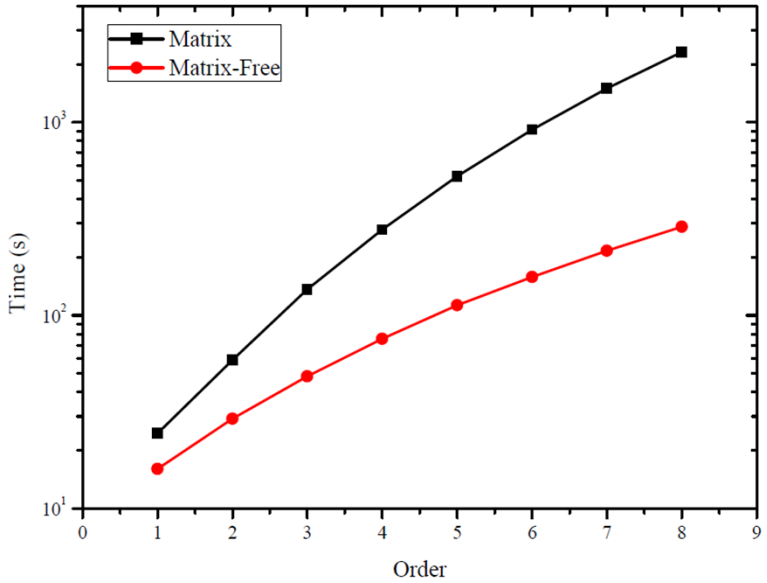
Computational Complexity 2D: $\mathbf{O}(nDOf^{3/2})$**, 3D:** $\mathbf{O}(nDOf^{4/3})$

Storage Complexity : $\mathbf{O}(nDof)$

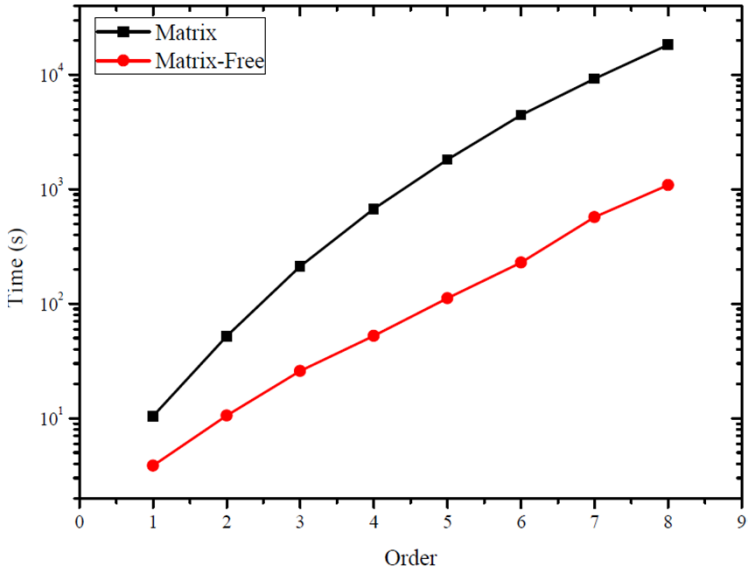
* M. Kronbichler and K. Kormann, “A generic interface for parallel cell-based finite element operator application,” *Computers & Fluids*, vol. 63, pp. 135–147, 2012.

Performance test

- We implemented matrix-free method in HopeFOAM
  - From the test on 2D and 3D isentropic vortex problem, the results show that Matrix-free has a great performance improvement.



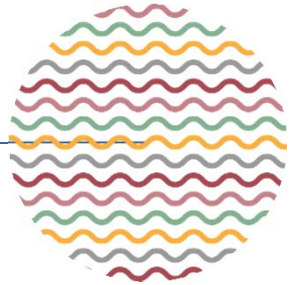
2D（50*50 cells）



3D（10*10*10 cells）

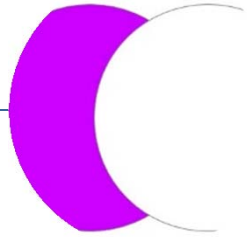
[published on HPCC 2018]

Communication optimization



Eliminate redundant communicat ions

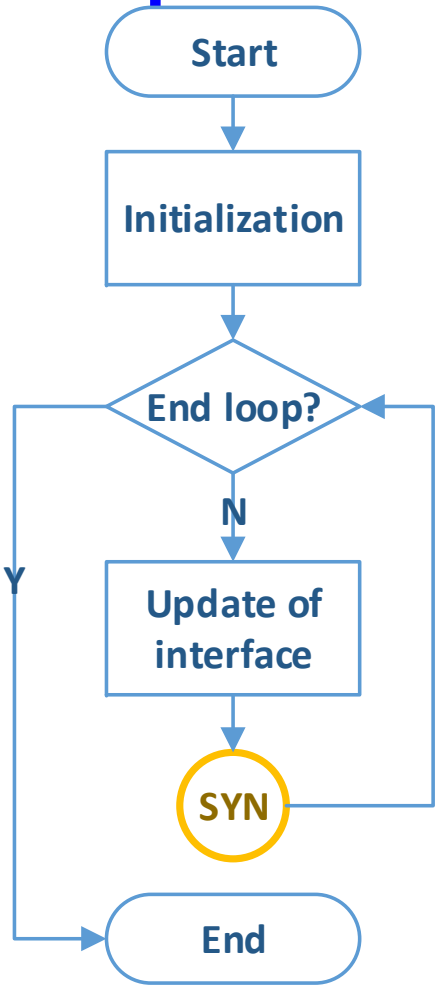
Reduce global communications



Overlap computing & communicat ion

Eliminate redundant comms

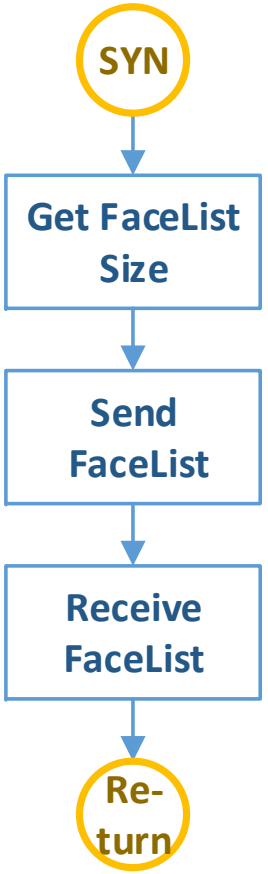
- Take Multiphase flow simulation as example
  - Interface capture uses VOF
  - Whose core step is —— MULES
    - Multi-dimensional Universal Limiter for Explicit Solution(MULES)
    - Two main operations in iteration
      - Update
      - Synchronization



Large amount of communication during synchronization becomes a major bottleneck in large-scale simulation

Eliminate redundant comms

- **Main communication content in synchronization**
  - FaceList
  - If the topology and boundary type between neighboring processes do not change, the content of the message is always the same
- **Introduce a global static variable, memArray**
  - Record the related information of the FaceList
  - Communicate facelist only once



Variables	Type	Introduction
mulesFlag	bool	Enter the correction process flag
memFlag	bool	Whether the information already record
memArray	int	Static variable, record the FaceList information

Comm optimization in PCG

PCG algorithm

Main calculations

- a matrix-vector multiply
- a preconditioner
- two dot products
- vector multiply-adds

Analysis

- Global communication caused by dot product operation is a major bottleneck in the development of parallel PCG methods

Input: A: nxn matrix, x_0: initial guess, b: rhs, M: prec
Output: x: approximate solution

1: $r_0 \leftarrow b - Ax_0 \quad z_0 \leftarrow Mr_0$
2: $\gamma_0 \leftarrow (z_0, r_0)$
3: $norm_0 \leftarrow \sqrt{(z_0, z_0)}$
4: **for** $j = 1, 2, \ldots$, **until convergence do**
5: **if** $j > 1$ **then**
6: $\beta_j \leftarrow \gamma_{j-1}/\gamma_{j-2}$
7: **else**
8: $\beta_j \leftarrow 0.0$
9: **end if**
10: $p_j \leftarrow z_{j-1} + \beta_j p_{j-1} \quad w_j \leftarrow Ap_j$
11: $\delta_j \leftarrow (p_j, w_j)$
12: $\alpha_j \leftarrow \gamma_{j-1}/\delta j \quad x_j \leftarrow x_{j-1} + \alpha_j p_j$
13: $r_j \leftarrow r_{j-1} - \alpha_j w_j \quad z_j \leftarrow Mr_j$
14: $\gamma_j \leftarrow (z_j, r_j)$
15: $norm_j \leftarrow \sqrt{(z_j, z_j)}$
16: **end for**
17: **return** x_j

Comm optimization in PCG

Algorithm optimization

- **Reduce global communications**
  - Eliminating data dependency by introducing intermediate variables
  - Two global comms are reduced to only one
- **Overlap computing & communication**
  - Increase communication hidden by algorithm reordering and non-blocking communication

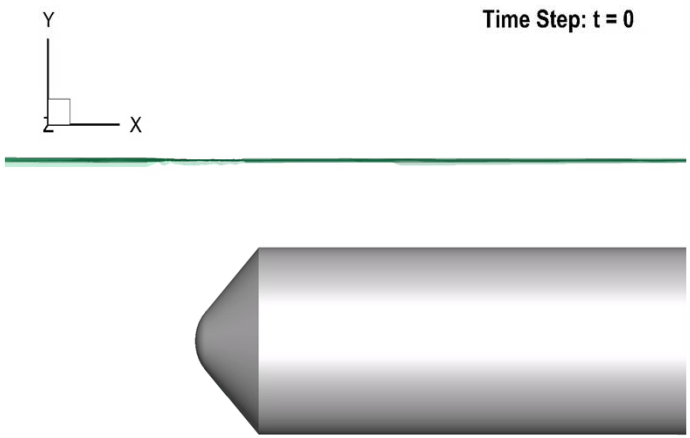
11: **for** $j = 1, residual > tolerance, j++$ **do**
12: **if** j > 1 **then**
13: $\beta_j \leftarrow \gamma_{j-1}/\gamma_{j-2}$
14: $\alpha_j \leftarrow \gamma_{j-1}/(\delta_{j-1} - \beta_j * \gamma_{j-1}/\alpha_{j-1})$
15: **else**
16: $\beta_j \leftarrow 0 \quad \alpha_j \leftarrow \gamma_{j-1}/\delta_{j-1}$
17: **end if**
18: $z_{j-1} \leftarrow n_{j-1} + \beta_j * z_{j-1} \quad q_{j-1} \leftarrow m_{j-1} + \beta_j * q_{j-1}$
19: $s_{j-1} \leftarrow w_{j-1} + \beta_j * s_{j-1} \quad p_{j-1} \leftarrow u_{j-1} + \beta_j * p_{j-1}$
20: $x_j \leftarrow x_{j-1} + \alpha_j * p_{j-1} \quad r_j \leftarrow r_{j-1} - \alpha_j * s_{j-1}$
21: $u_j \leftarrow u_{j-1} - \alpha_j * q_{j-1} \quad w_j \leftarrow w_{j-1} - \alpha_j * z_{j-1}$
22: $\delta_j \leftarrow (w_j, u_j) \quad \gamma_j \leftarrow (r_j, u_j)$
23: $R_j \leftarrow \sum |r_j^{(i)}|$
24: MPI_Iallreduce on δ_j, γ_j, R_j
25: $m_j \leftarrow M * w_j \quad n_j \leftarrow A * m_j$
26: $residual \leftarrow R_j/norm$
27: **end for**
28: **return** x_j

Performance test

Multiphase flow simulation test: Cavitation

with the communication optimization introduced before

Case	Solution	Execution Time on Different Cores(s)							
		12	24	48	96	192	384	768	1152
headform	OF-ori	7330.81	3207.62	1289	464.27	847.88	2634	11895	77486
	OF-opt	7140.80	2957	994.63	277.82	193.08	167.74	576.17	2476



Overall solution performance has been greatly improved by 63.87%

[published on HPCC 2018]

Part III:

Where am I going ?

——HPC Service based on HopeFOAM

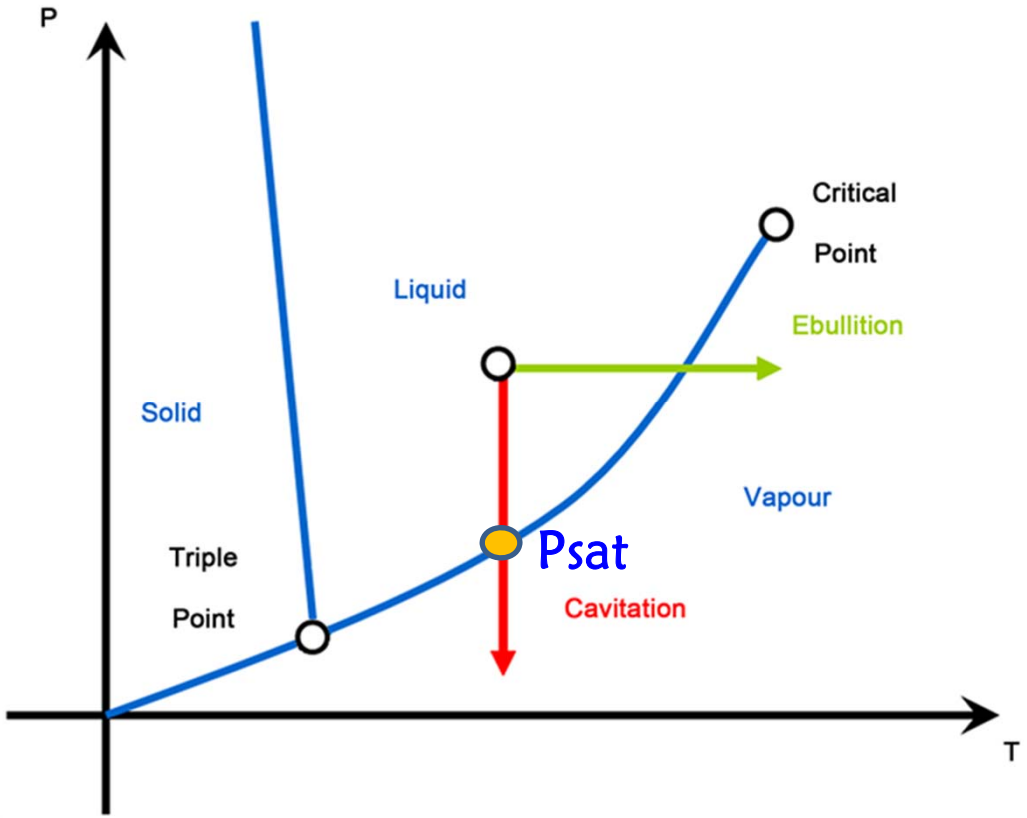
Part III

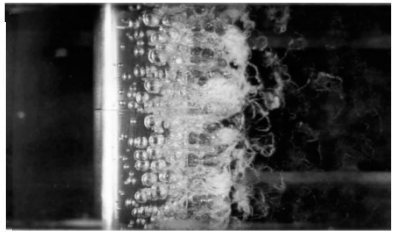
- Cooperation for Cavitation Research
- Web-based Service of HopeFOAM
- Digital Design and Optimization based on HopeFOAM

Cooperation for Cavitation Research

Cavitation

- When the liquid pressure drops below the saturated vapor pressure, cavitation happens.



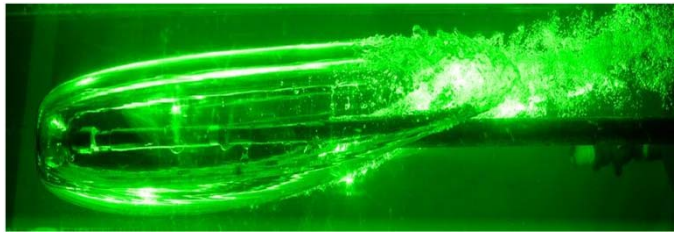


Bubble cavitation

Cloud cavitation

Sheet cavitation

Vortex cavitation



Supercavitation

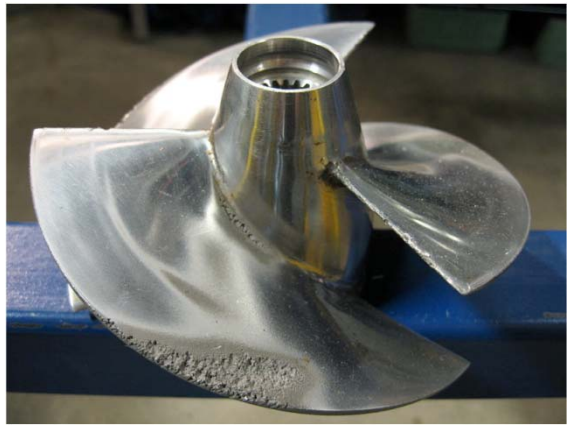
Cooperation for Cavitation Research

Cavitation

- When the liquid pressure drop below the saturated vapor pressure, cavitation happens

AD&DAD of cavitation

- negative: "Cavitation is the most hated hydrodynamic phenomenon"
  - Cavitation and cavitation erosion on propeller



Cooperation for Cavitation Research

Cavitation

- When the liquid pressure drop below the saturated vapor pressure, cavitation happens

AD&DAD of cavitation

- negative: "Cavitation is the most hated hydrodynamic phenomenon"
- positive: Supercavition drag reduction can reduce the drag up to 90%
  - In 1970s, the former Soviet Union developed a supercavitation torpedo "Shkval" (Speed 200kn, range 10km)

Cooperation for Cavitation Research

- **Collaborator**
  - Team of Prof. Zhang Weihua from NUDT
- **Cavitation Simulation based on HopeFOAM**
  - Discretization supporting
  - Model extension
  - Multiphase Cavitation Solver
  - Multi-Scale Cavitation Solver

Discretization supporting

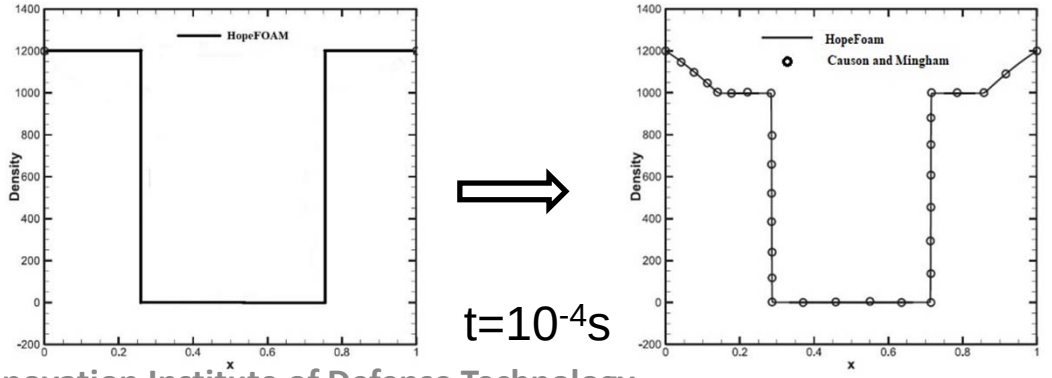
Cavitation simulation based on DG method

- For the dramatic change of field properties in cavitating flow: large gradient of density and phase fraction
- DG method could provide high accuracy solution, and have a good performance in discontinuity capture

Interface capture + LF flux + TVD limiter

```
dg::solveEquation(dgm::ddt(rho1) + dgc::div(gther_U, rho1, Godunov.fluxRho()));

dg::solveEquation(dgm::ddt(rhoU1) + dgc::div(gther_U, rhoU1, Godunov.fluxRhoU()) + dgc::grad(gther_p));

dg::solveEquation(dgm::ddt(Ener1) + dgc::div(gther_U, Ener1, Godunov.fluxEner())+ dgc::div(gther_U, gther_p));
```

Simulation of Cavity collapse (density curve)



t=0s

t=10^{-4}s

Cavitation model extension

- **Add** some common used cavitation models into HopeFOAM
  - such as the Singhal、Zwart model etc.
- **Modify** the cavitation models
  - take the second order term and viscous effect into consideration.

$$\begin{cases} \dot{m}^{+} = C_c \dfrac{3\rho_v\rho_l}{\rho}\dfrac{\alpha_v\alpha_l}{R}\sqrt{\dfrac{2|p_v-p|}{3\rho_l}\left(1-\dfrac{R_0^3}{R^3}\right)-\dfrac{2S}{\rho_l R}\left(1-\dfrac{R_0^2}{R^2}\right)} & p > p_v \\ \dot{m}^{-} = -C_v \dfrac{3\rho_v\rho_l}{\rho}\dfrac{\alpha_v\alpha_l}{R}\sqrt{\dfrac{2|p_v-p|}{3\rho_l}\left(1-\dfrac{R_0^3}{R^3}\right)-\dfrac{2S}{\rho_l R}\left(1-\dfrac{R_0^2}{R^2}\right)} & p < p_v \end{cases}$$

Second order term Viscosity term

Turbulence model extension

- Implement **Partially Averaged Navier-Stokes (PANS)** turbulence model

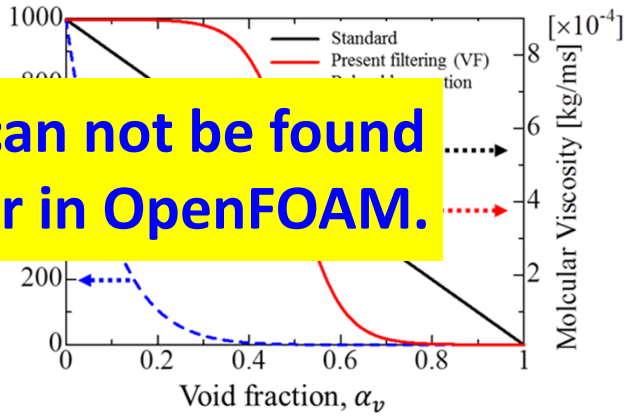
RANS → Solve Taylor Scale → Select f_k value → PANS

- Implement **viscosity modified turbulence models for cavitating flows**
  - DCM、FBM and hybrid FBDCM model

Model	Turbulence Eddy viscosity	Correct function f
Original k-ε tur		
Filter-based m		
Density correctio		
FBDCM	$C_\mu = 0.09$	$f_{hybrid} = \chi(\rho_m/\rho_l) f_{FBM} + [1-\chi(\rho_m/\rho_l)] f_{DCM}$ $\chi(\rho_m/\rho_l) = 0.5 + \frac{\tanh[\frac{4*(0.6\rho_m/\rho_l - C_2)}{0.2(1-2C_2)+C_2}]}{2\tanh(4)}$

All of these models and modifications can not be found in most of commercial software, neither in OpenFOAM.

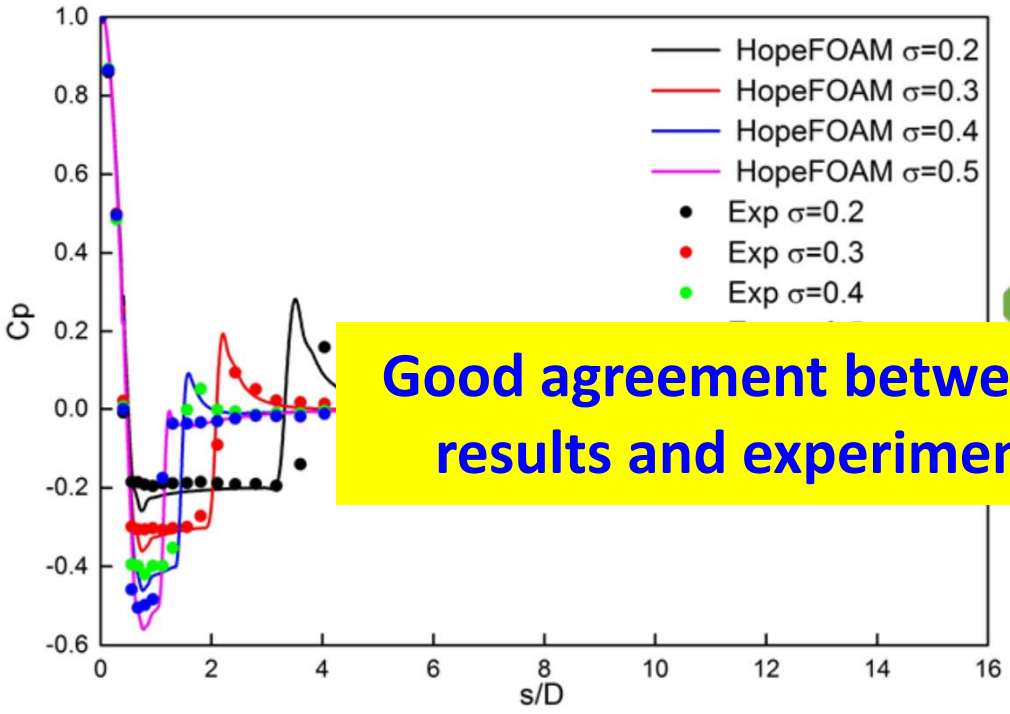
Correct function

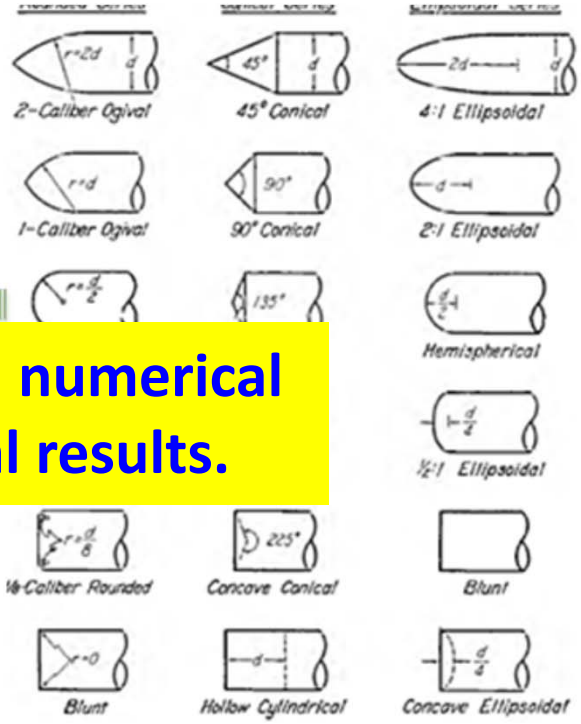


Correct effect

Simulation cases

- Based on these models and modifications, we did a lot of validations
  - 18 *different head-forms* from NHRC(Navy Hydrodynamic Research Center of America)





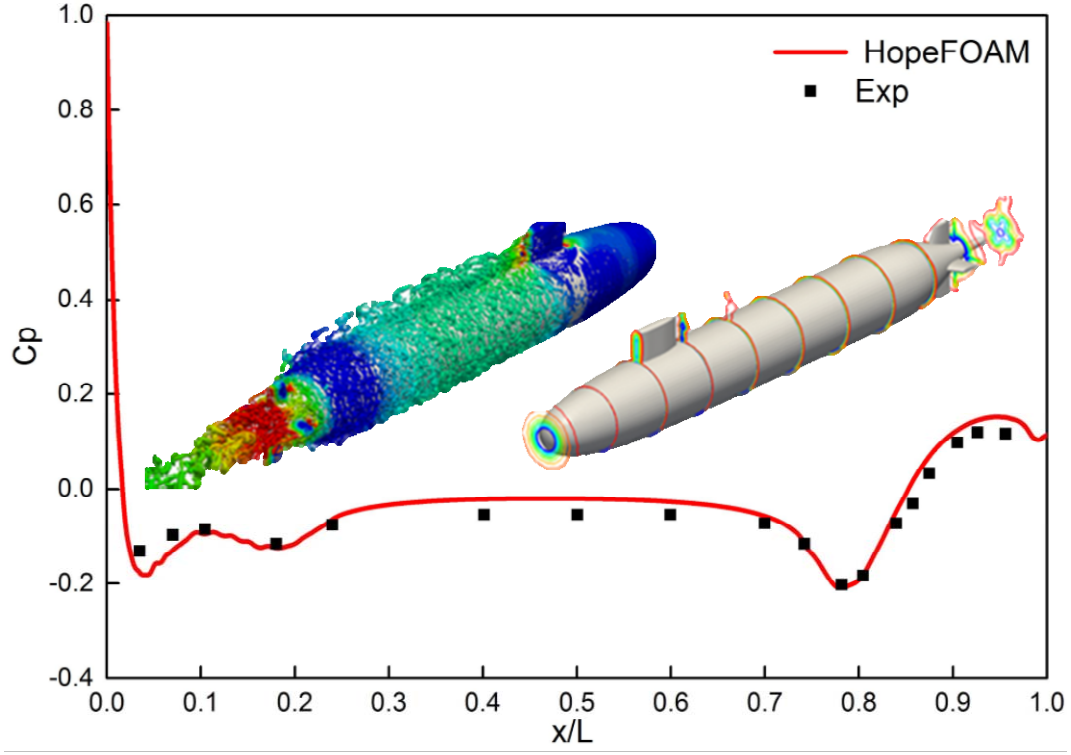
Good agreement between numerical results and experimental results.

Headform shapes and pressure coefficient along the mother line of the cylinder

Simulation cases

SUBOFF submarine from DARPA

- The results simulated with HopeFOAM is also very good



Pressure coefficient along the mother line of bare hull of the submarine

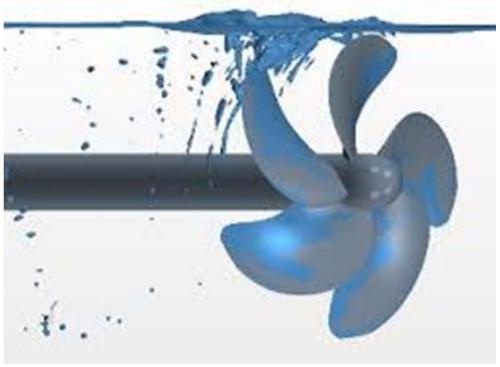
Multiphase Cavitation Solver

Background

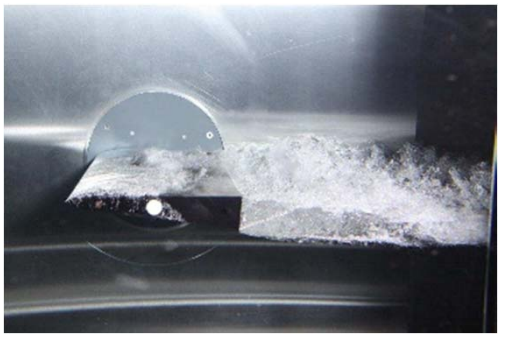
Multiphase model requirement

- Cavitation near free surface
- Natural and ventilated cavitation
- Water entry problem
- …

No such solver in the original OpenFOAM



Propeller



Natural & ventilated cavitation

High speed hydrofoil vehicle

Water entry

Multiphase Cavitation Solver

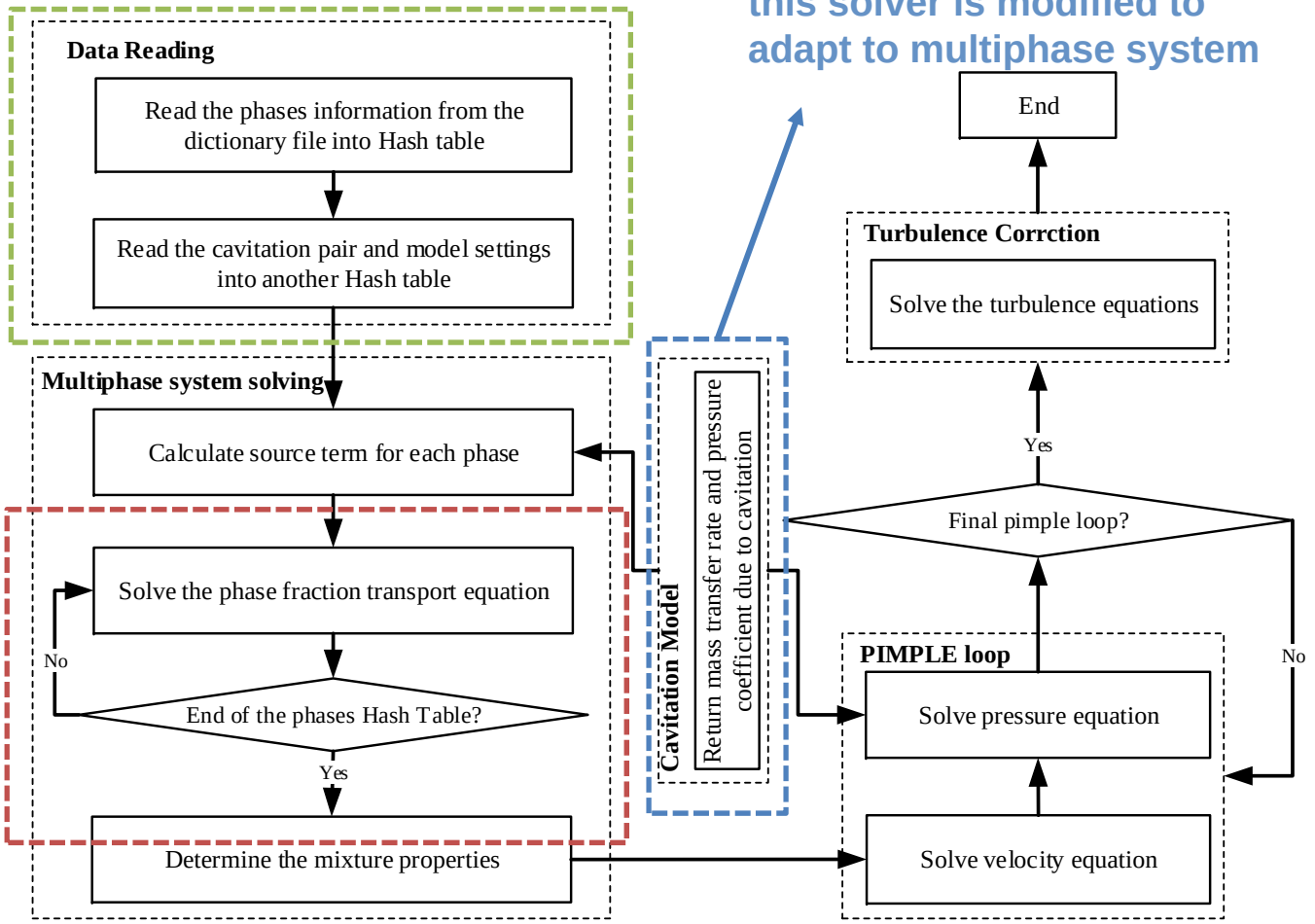
Framework of the solver

Compared with the interPhaseChangeFoam solver in OpenFOAM, this solver has several improvements:

- Two or more phases can be solved
- Multi cavitation phase pairs are allowed

- In the multiphase system part, each phase will be solved by the MULES utilities

- the cavitation model used in this solver is modified to adapt to multiphase system

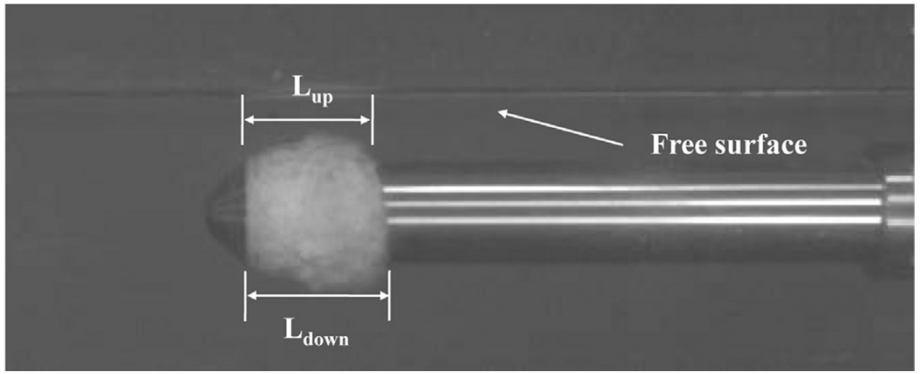


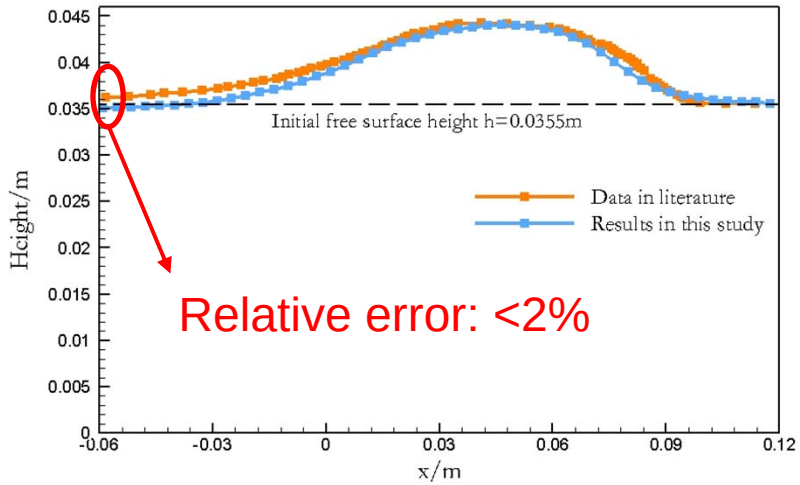
Multiphase Cavitation Solver

☐Validation case: 3D

Projectile

The cloud cavitating flows near the free surface around an launching projectile are simulated. Results were compared with the experimental data performed by Wang[1] et.al and those calculated by Fluent.





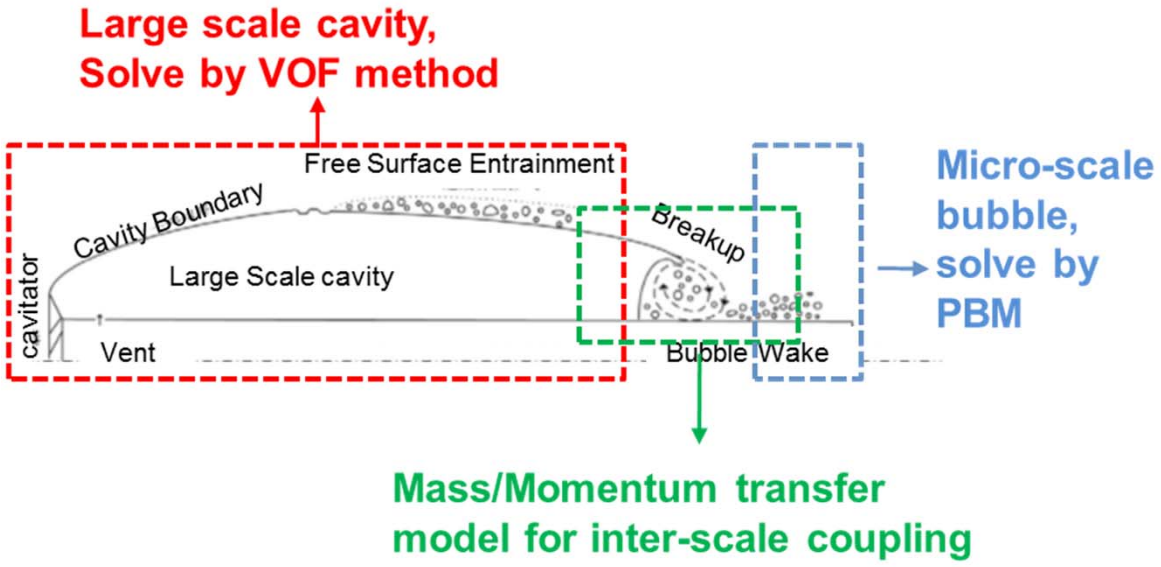
Height of the free surface

[1]Wang Y., Wu X., Huang C.,et.al. Unsteady characteristics of cloud cavitating flow near the free surface around an axisymmetric projectile. International Journal of Multiphase Flow, 85(2016), 48–56.

Multi-Scale Cavitation Solver

VOF-PBM(Population Balance Model)

- VOF+PBM to deal with different scales
- Mass/Momentum transfer model for inter-scale coupling



- Still in processing: zhouhoucun09@nudt.edu.cn

Part III

- Cooperation for Cavitation Research
- Web-based Service of HopeFOAM
- Digital Design and Optimization based on HopeFOAM

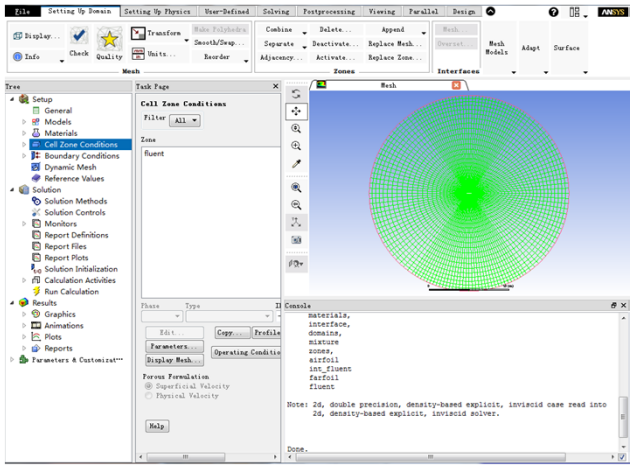
After cooperation with other teams, we found that although OpenFOAM is an outstanding open source CFD framework, there are still services can be improved.

Web-based Service of HopeFOAM

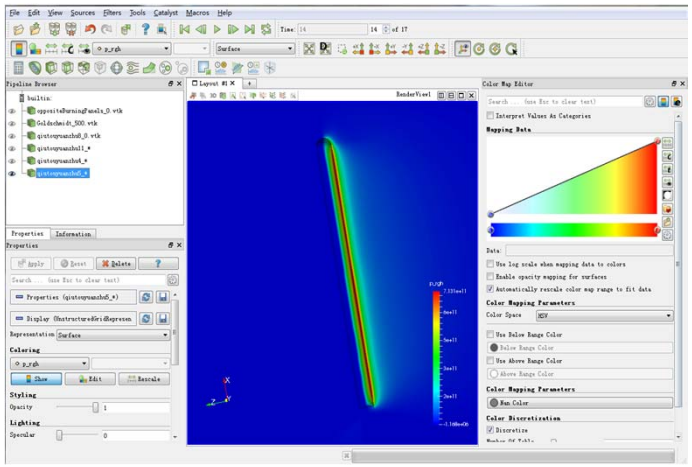
Functional requirements

Front-end Operating Platform With Graphical Interface

- Inconvenient to use command line
- Build visualization software framework system
  - Simulation task monitor and management
  - Pre-process configure cases&Post-processing tool integration



+



Web-based Service of HopeFOAM

Functional requirements

Front-end Operating Platform With Graphical Interface

Data Management Platform

- Data stored with file systems may be lost and hard for preservation
- Databases system can maintain data easily
  - Data insert/modify/query
  - Comparison of simulation results for error analysis
  - BIG-DATA expert system for configuration reference

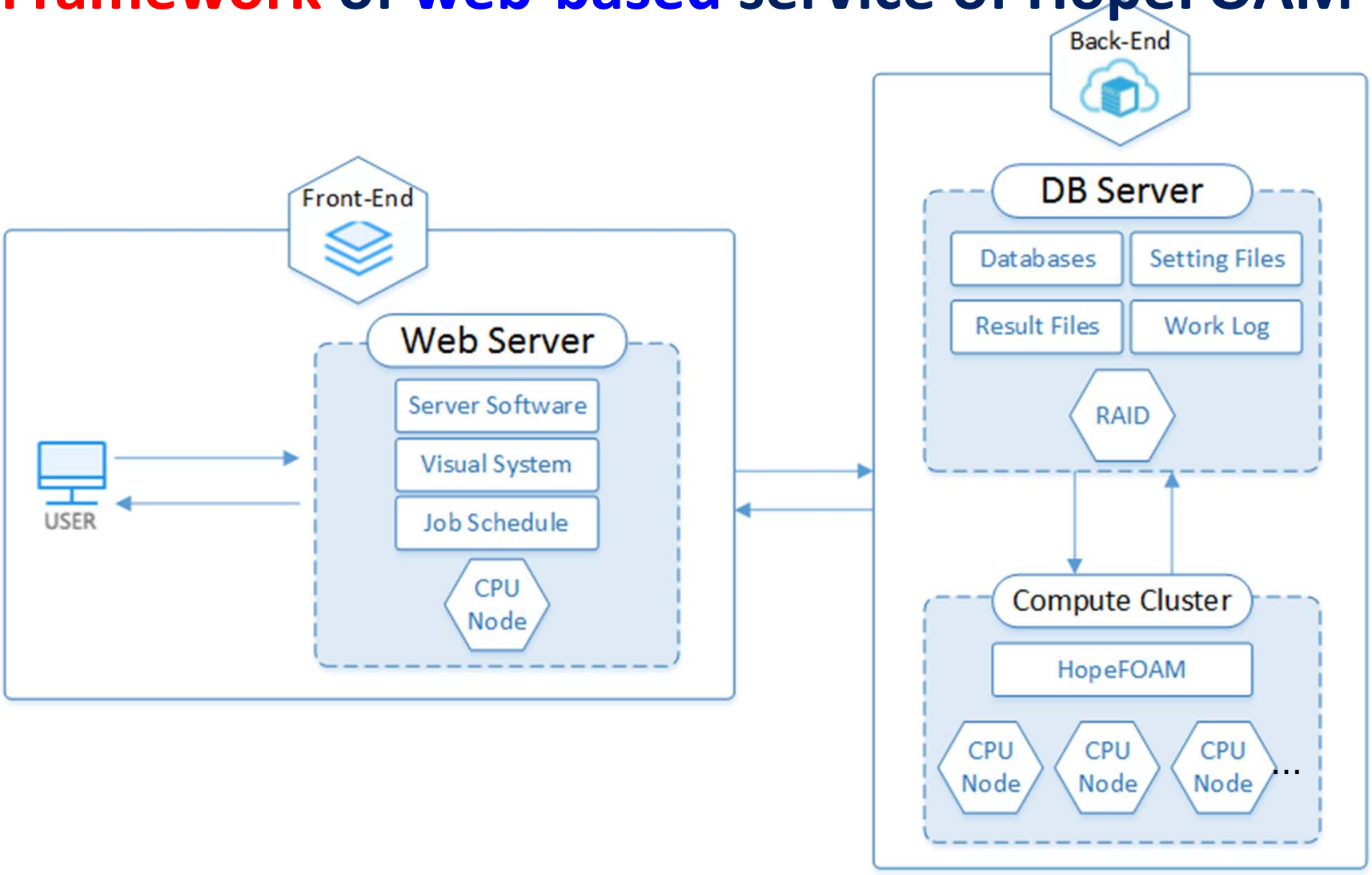
System design

- In order to implement the functions, the first thing is to do the system design
  - **Web-besed OR Software Client ?**

	Web	Software Client
Install	No need	Need to install
Maintain	low maintenance,server only	High maintenance
Cross-platform	Allow	Need multiple versions
Reference	PDM/SDM,HPC,simulation platform	Fluent,CFX,other CAE softwares for PC
Responding Speed	Depends on local Telecommunication, Generally slower than the client	Depends on local device
Development Difficulties	High difficulty, developer need master multiple language frameworks	Low difficuty with various development tools

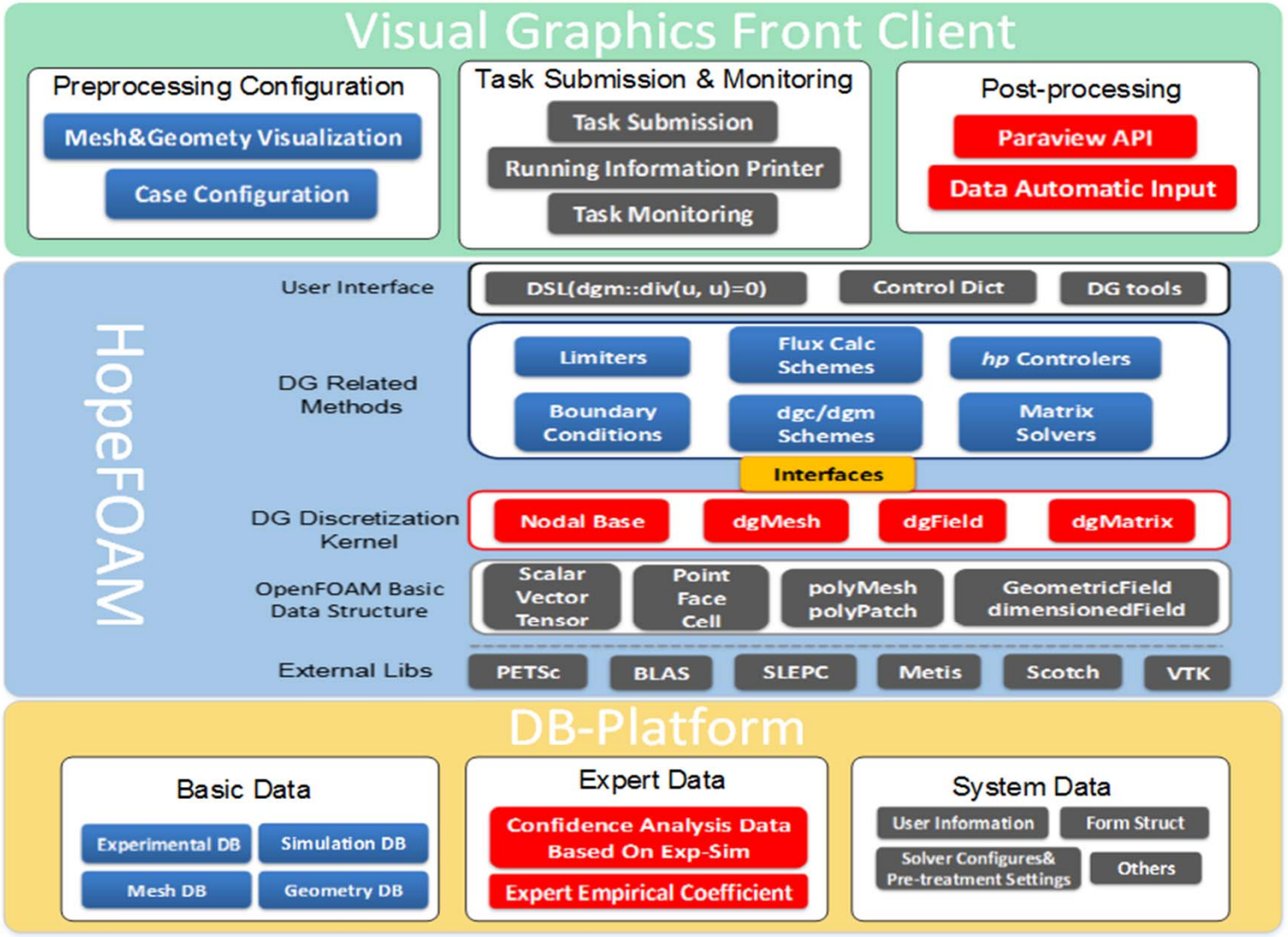
System design

Framework of web-based service of HopeFOAM



System design: Web-besed

Software Architecture



Already done

Numerical simulation graphical interface

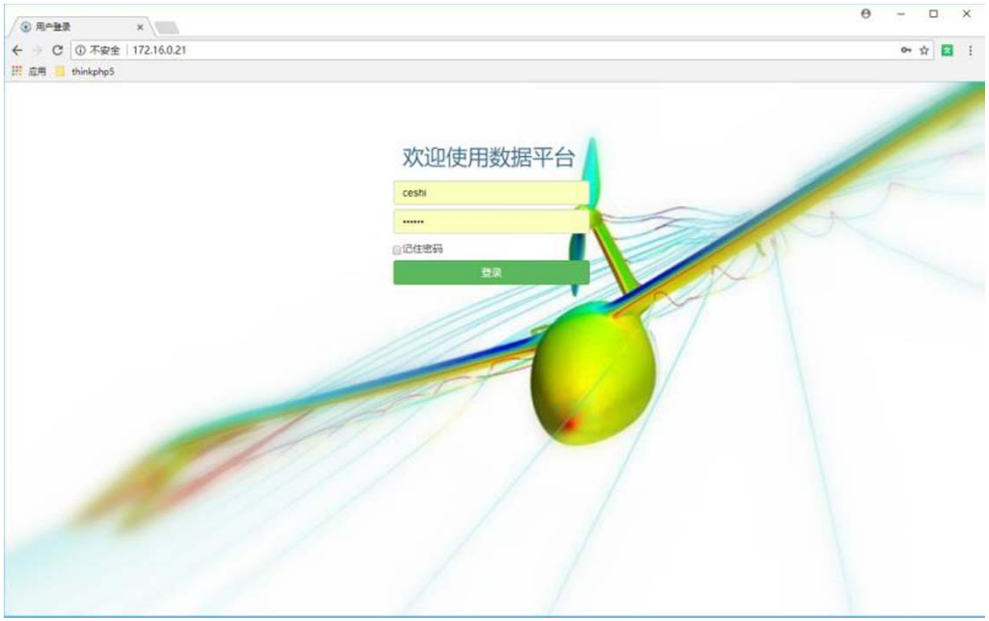
Web front-end visualization interface

- Mesh → solver → parameter → run → download output

OpenFOAM

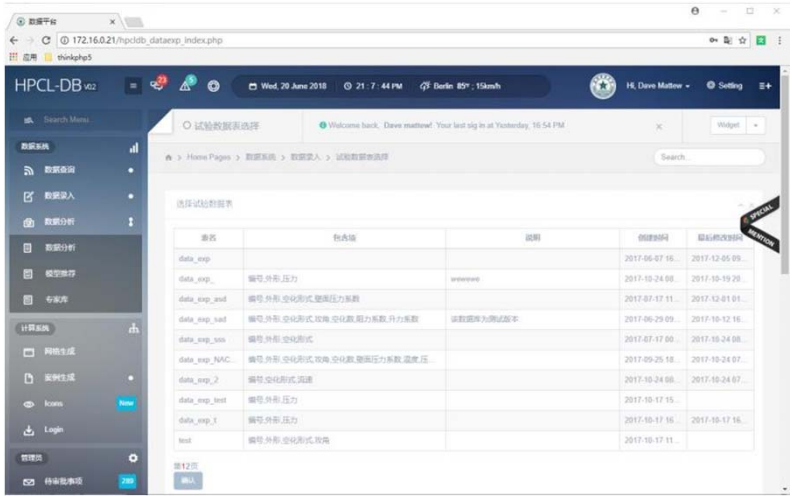
Already done

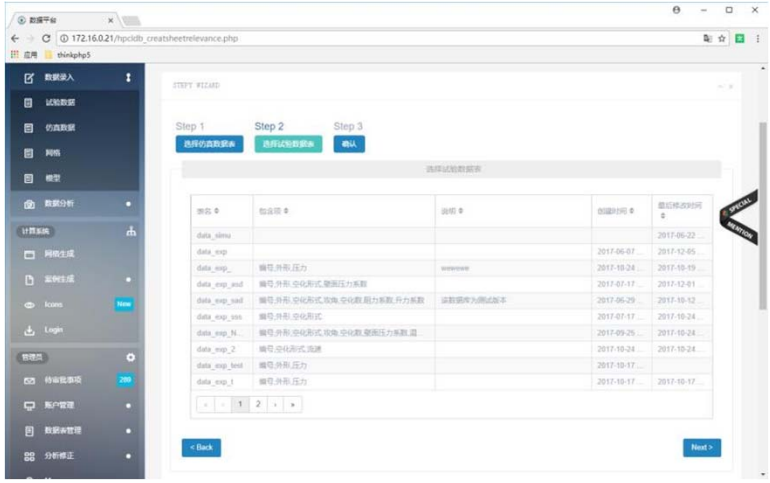
- Numerical simulation graphical interface
- Data Platform
  - *Mesh database*



Already done

- Numerical simulation graphical interface
- Data Platform
  - *Case database*
    - Experimental data
    - Simulation results
    - Preliminary error analysis





Already done

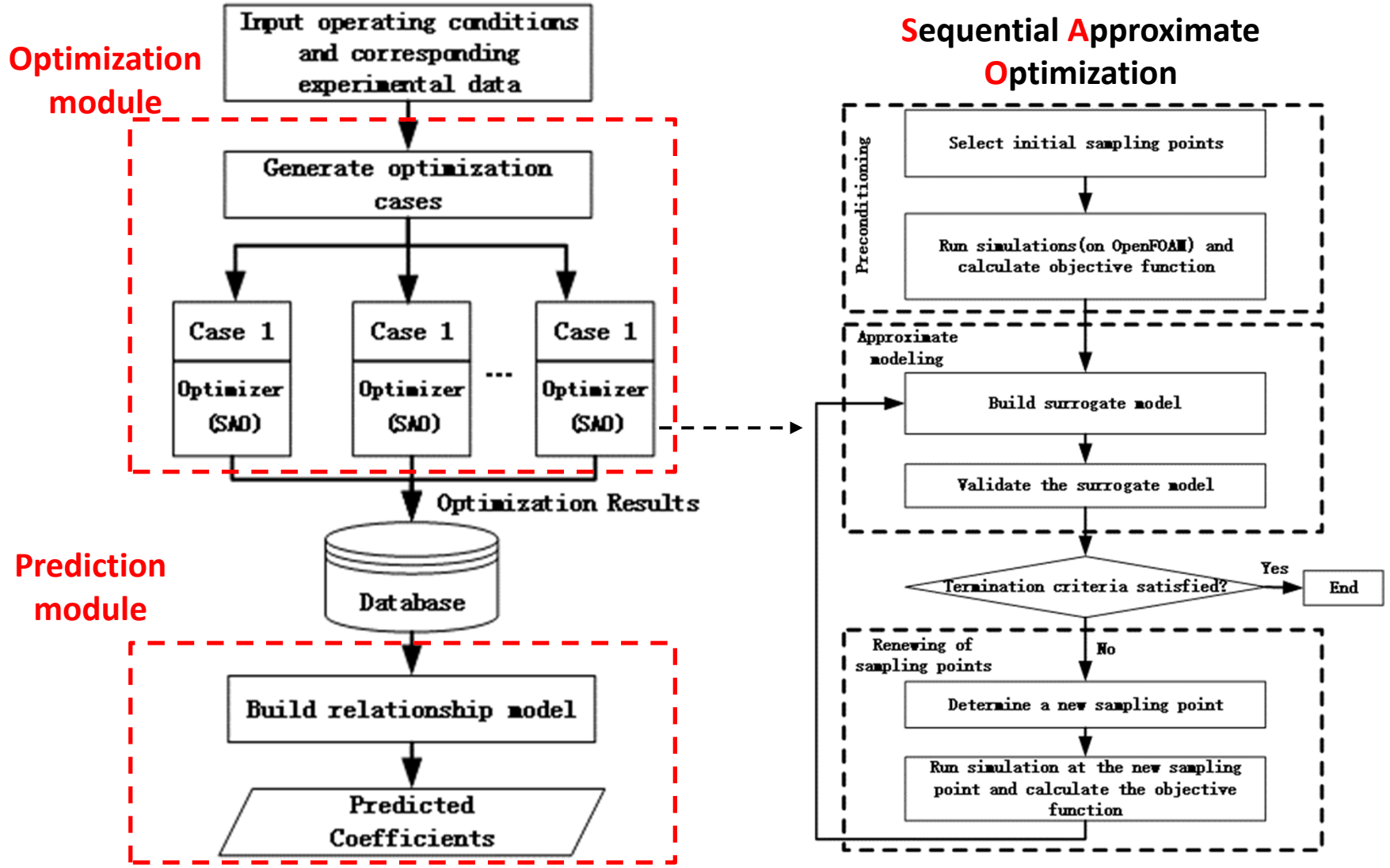
- Numerical simulation graphical interface
- Data Platform
  - Solver database
    - Maintain kinds of solvers
    - Model coefficient adaptation

Zwart model

$$\begin{cases} \dot{m}^{+} = F_c \dfrac{3\alpha_v \rho_v}{R_B} \sqrt{\dfrac{2}{3}\dfrac{P - P_v}{\rho_l}} \\ \dot{m}^{-} = -F_e \dfrac{3\alpha_{nuc}(1-\alpha_v)\rho_v}{R_B} \sqrt{\dfrac{2}{3}\dfrac{P_v - P}{\rho_l}} \end{cases}$$

- How to determine the optimal coefficients for different cases(flow condition) to guarantee the accurate results?
- By experience?
  - Inaccurate and inefficient

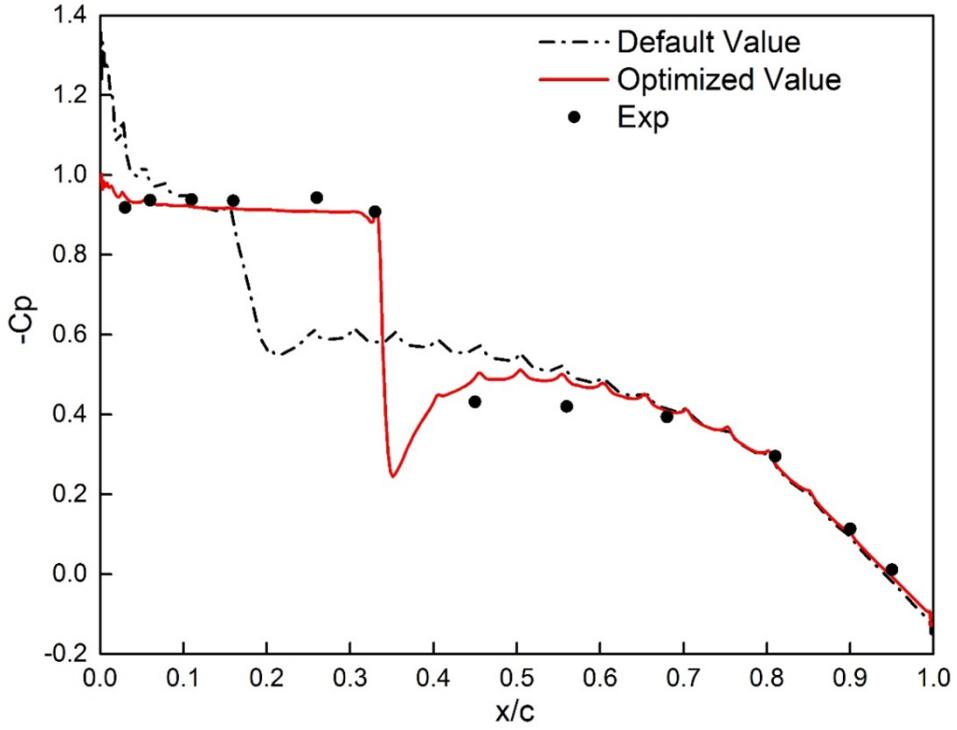
Model coefficient adaptation



Model coefficient adaptation

Zwart model optimized results

Operating condition: cavitation number=0.91



	Default	Optimal
Fc	0.01	0.015
Fe	50	483
Normalized Error	1.125	0.186

Compared with default coefficients, our method improves the accuracy of simulation.

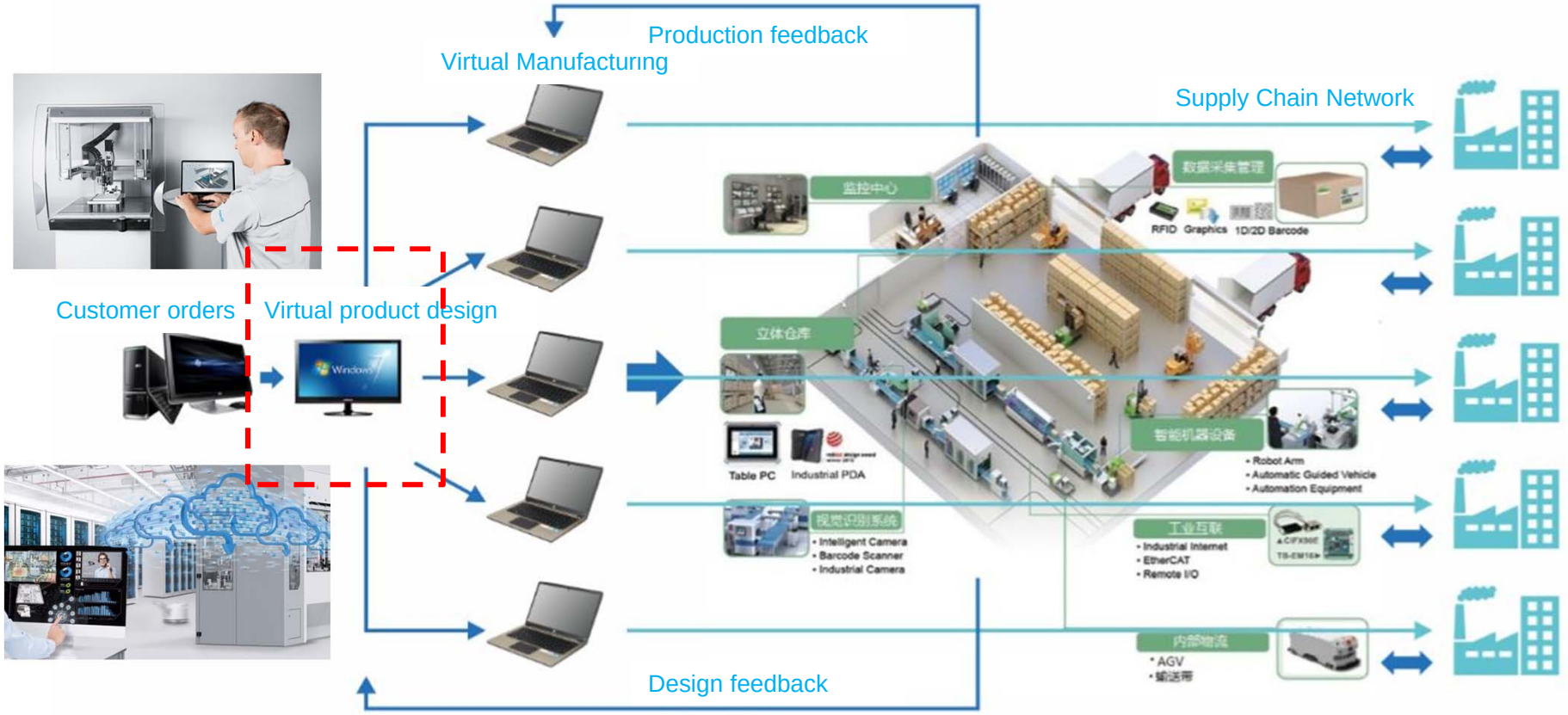
Part III

- Cooperation for Cavitation Research
- Web-based Service of HopeFOAM
- Digital Design and Optimization based on HopeFOAM

Digital Design and Optimization

Industrial manufacturing

- *Optimization* **design accounts 5-15% life cycle cost**
- **Affecting 70-75% product cost and 80% quality**

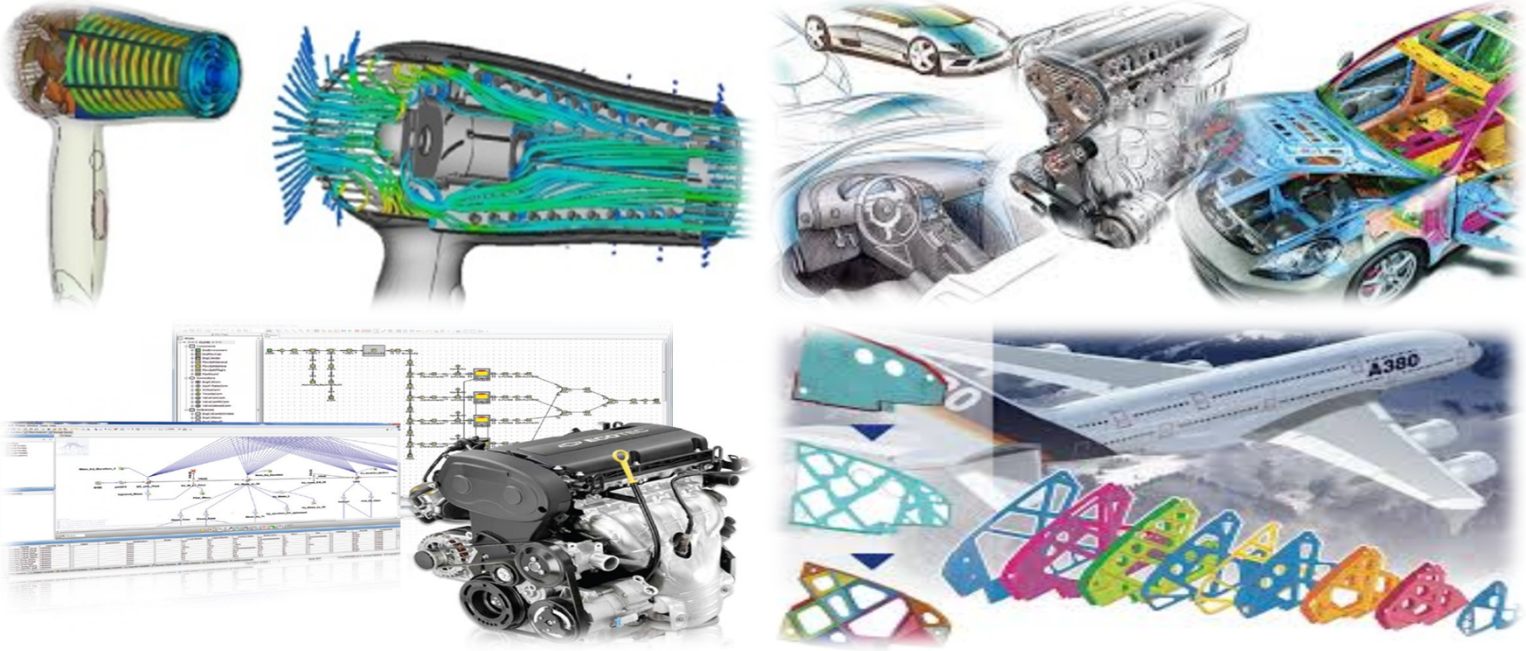


Digital Design and Optimization

Digital Design and Optimization

- Improve product quali
- Shorten design cycle
- Save product costs

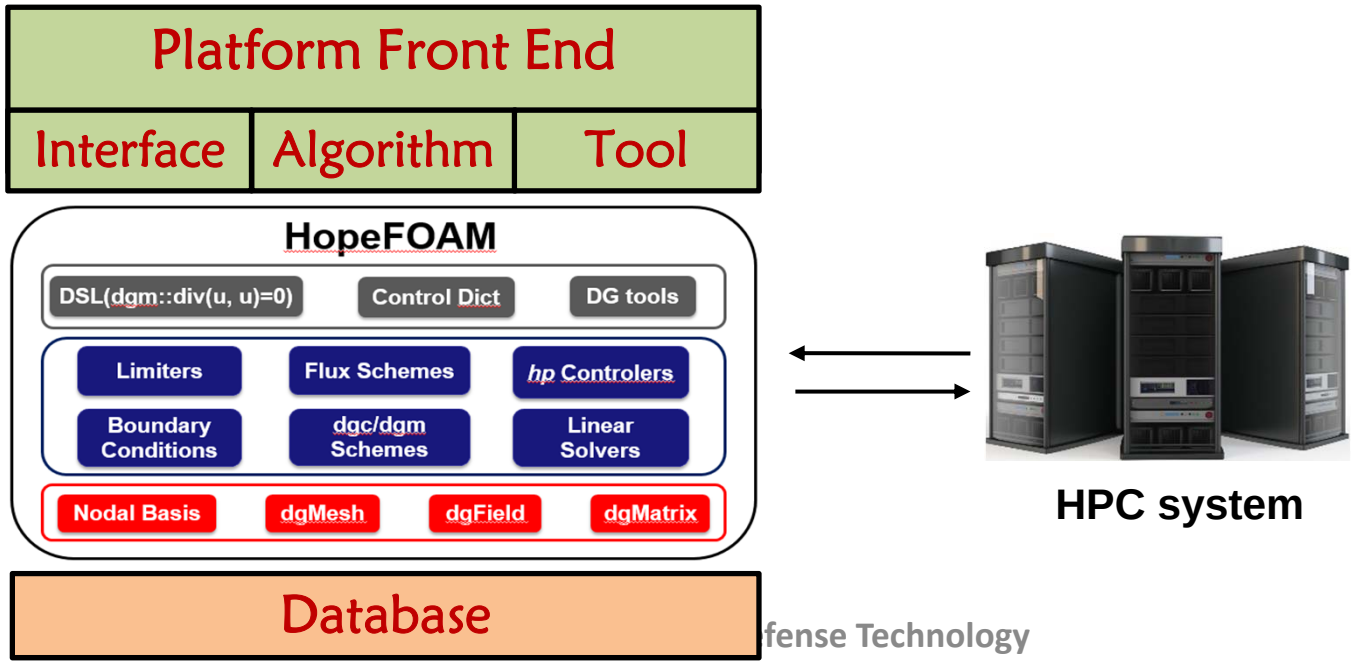
Find "best possible" solutions from a number of possible solutions and select "optimal" parameters from multiple parameters.



Digital Design and Optimization

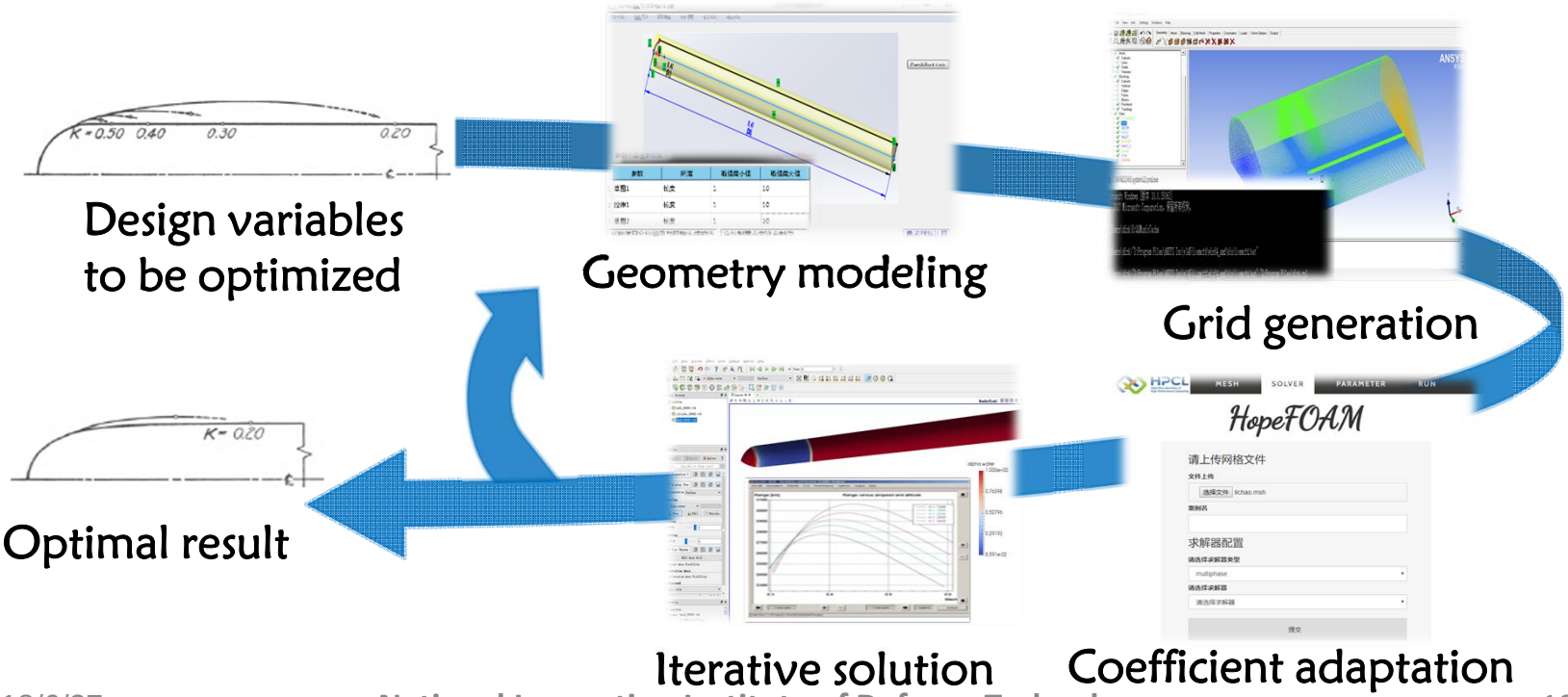
Optimization Platform Based on HopeFOAM

- *Trusted results*: HopeFOAM as simulation kernel
- *Engineering oriented*: graphic interface and tools
- *Reliable data*: build database for data mining



Digital Design and Optimization

- Based on the platform, we **Integrated automatic optimization process**
  - *Design variables to be optimized*
  - *Interactive geometry modeling*
  - *Automatic grid generation*



Welcome to join us!

Currently, working at Guangzhou

(xuxinhai@nudt.edu.cn)

Thank You!