The new OpenFOAM computational framework for fluid flows based on regularized equations

Matvey V. Kraposhin

20 May, 2020

Table of Contents

Talk outline

- The approach to solve various fluid flow problems with regularized (or QGD) equations and unstructured finite volume method is presented
- Brief history and essentials of the QGD equations and corresponding QGD algorithms are set out
- The QGD-based approach is implemented as framework "QGDsolver" on the basis of OpenFOAM platform
- The overview of "QGDsolver" framework capabilities is given
- The source of "QGDsolver" is published on GitHub: `https://github.com/unicfdlab/QGDsolver`
- The presentation could be downloaded from: `PDFURL`

History

1982 – QGD system derived from Boltzmann equation

Prof. Boris N. Chetverushkin

Prof. Tatiana G. Elizarova

1997 – QGD system formulated as conservation laws

Prof. Yu. V. Sheretov

From then to now

regularized or sometime Quasi Gas Dynamic (QGD) and Quasi Hydro Dynamic (QHD) equations are extensively used for various flows simulations – incompressible, compressible, multicomponent, magnetohydrodynamic, porous flows, two-phase flows – in Russia, Europe and in Keldysh Institute of Applied Mathematics of the RAS `https://keldysh.ru/`

The brief introduction to regularized equations algorithms

- For brevity we use “regularized” and “QGD” as synonyms
- We introduce QGD or regularized algorithms as an approximation of QGD or regularized equations
- Originally QGD equations were derived before other types of systems
- But for the sake of visibility, the explanation in the presentation doesn't follow the historical order:
  1. the procedures to regularize 1D transport equation and 1D barotropic gas equations are presented
  2. the procedure is extended by analogy to 3D
  3. some issues of QGD algorithms are discussed
  4. incompressible viscous fluid regularized equations (QHD) are presented
  5. compressible viscous perfect gas regularized equations (QGD) are presented
  6. finally, the application of QGD approach to other flow types and it's combination with standard OpenFOAM libraries is presented

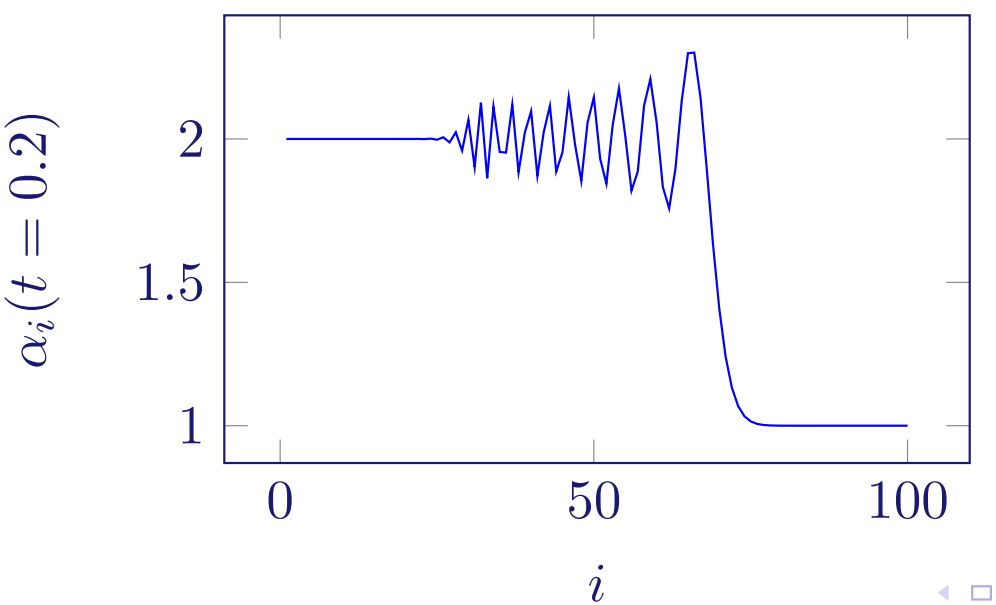
Scalar transport equation

Consider FVM approximation of scalar transport equation:

$$\frac{\partial \alpha}{\partial t} + \nabla \cdot \left(\vec{U} \alpha \right) = 0$$

on the 1D interval [0, 1] with central differencig spatial scheme and explicit Euler 1^{st} order scheme:

$$\Delta x \frac{\alpha_i^n - \alpha_i^o}{\Delta t} + \left(U_{i+1/2} \alpha_{i+1/2}^o \right) - \left(U_{i-1/2} \alpha_{i-1/2}^o \right) = 0$$



Without the artificial numerical diffusion the numerical solution is known to be oscillating. To mitigate this issue TVD schemes are usually used.

Regularization procedure

If we regularize the previous equation by following steps:

1. integration and averaging of equation over small time interval τ

$$\langle \alpha \rangle = \frac{1}{\tau} \int_0^\tau \alpha dt$$

2. substitution of avarage values with instant values

$$\langle \alpha \rangle = \alpha + \tau \frac{\partial \alpha}{\partial t}$$

3. subsitution of first time derivative terms using **original equation**

$$\frac{\partial \alpha}{\partial t} = -\nabla \cdot \left(\vec{U} \alpha \right)$$

4. neglection of second time derivatives

$$\tau \frac{\partial^2 \alpha}{\partial t^2} = 0, \tau^2 = 0$$

we can get new smoothed equation with controllable introduced diffusion.

Regularized scalar transport equation

By appying regularization procedure to transport equation we get

- At the step 1:

$$\left\langle \frac{\partial \alpha}{\partial t} \right\rangle + \nabla \cdot \left(\vec{U} \left\langle \alpha \right\rangle \right) = 0$$

- At the step 2:

$$\frac{\partial \alpha}{\partial t} + \tau \frac{\partial^2 \alpha}{\partial t^2} + \nabla \cdot \left(\vec{U} \alpha \right) + \nabla \cdot \left(\vec{U} \tau \frac{\partial \alpha}{\partial t} \right) = 0$$

- At the step 3:

$$\frac{\partial \alpha}{\partial t} + \tau \frac{\partial^2 \alpha}{\partial t^2} + \nabla \cdot \left(\vec{U} \alpha \right) - \nabla \cdot \left(\vec{U} \tau \nabla \cdot \left(\vec{U} \alpha \right) \right) = 0$$

- At the step 4:

$$\frac{\partial \alpha}{\partial t} + \nabla \cdot \left(\vec{U} \alpha \right) - \nabla \cdot \left(\vec{U} \tau \nabla \cdot \left(\vec{U} \alpha \right) \right) = 0$$

Application to system of equations

The approach could be applied to a system of equations. For example, one considers inviscid 1D flow of compressible barotropic perfect gas

$$\frac{\partial \rho}{\partial t} + \frac{\partial \rho U}{\partial x} = 0$$

$$\frac{\partial \rho U}{\partial t} + \frac{\partial \rho U U}{\partial x} + \frac{\partial p}{\partial x} = 0$$

$$\rho = \rho(p), \quad c^2 = \frac{\partial p}{\partial \rho}$$

By averaging equations, substituting avarage values with instant values $\langle \rho \rangle = \rho + \tau \frac{\partial \rho}{\partial t}$, $\langle U \rangle = U + \tau \frac{\partial U}{\partial t}$, $\langle p \rangle = p + \tau \frac{\partial p}{\partial t}$ we get:

$$\frac{\partial \rho}{\partial t} + \tau \frac{\partial^2 \rho}{\partial t^2} + \frac{\partial \rho U}{\partial x} + \frac{\partial}{\partial x} \tau \frac{\partial \rho U}{\partial t} = 0$$

$$\frac{\partial \rho U}{\partial t} + \tau \frac{\partial^2 \rho U}{\partial t^2} + \frac{\partial \rho U U}{\partial x} + \frac{\partial}{\partial x} \tau \frac{\partial \rho U U}{\partial t} + \frac{\partial p}{\partial x} + \frac{\partial}{\partial x} \tau \frac{\partial p}{\partial t} = 0$$

Regularized system of equations for 1D barotropic inviscid perfect gas

Now according to step (3) of the abovementined procedure
we can cast time derivatives into spatial derivatives using the original system:

$$\frac{\partial \rho}{\partial t} + \tau \frac{\partial^2 \rho}{\partial t^2} + \frac{\partial \rho U}{\partial x} + \frac{\partial}{\partial x}\tau(-\frac{\partial \rho U U}{\partial x} - \frac{\partial p}{\partial x}) = 0$$

$$\frac{\partial \rho U}{\partial t} + \tau \frac{\partial^2 \rho U}{\partial t^2} + \frac{\partial \rho U U}{\partial x} +$$

$$\frac{\partial}{\partial x}(\tau(-\frac{\partial \rho U U}{\partial x} - \frac{\partial p}{\partial x})U + \tau \rho U(-U\frac{\partial U}{\partial x} - \frac{1}{\rho}\frac{\partial p}{\partial x})) + \frac{\partial p}{\partial x} + \frac{\partial}{\partial x}\tau c^2(-\frac{\partial \rho U}{\partial x}) = 0$$

Finally, after step (4) and letting $j_m = \rho U - \tau(\frac{\partial \rho U U}{\partial x} + \frac{\partial p}{\partial x})$ we have regularized system:

$$\frac{\partial \rho}{\partial t} + \frac{\partial j_m}{\partial x} = 0$$

$$\frac{\partial \rho U}{\partial t} + \frac{\partial}{\partial x}\left(j_m U + \tau \rho U(-U\frac{\partial U}{\partial x} - \frac{1}{\rho}\frac{\partial p}{\partial x})\right) + \frac{\partial p}{\partial x} + \frac{\partial}{\partial x}\tau c^2(-\frac{\partial \rho U}{\partial x}) = 0$$

General procedure to derive QGD algorithm

The mentioned above simple procedure of regularization could be extended to 3D compressible and incompressible flows:

1. write down original system of equations
2. average and regularize original system using aforementioned rules
3. drop-out small terms: $O(\tau^2)$, $\frac{\partial^2}{\partial t^2}$, terms that are proportional to product of τ and viscosity: $O(\tau\mu)$
4. approximate equations using "naive"or entropy-preserving schemes
5. derive an algorithm to solve discretized system of equations sequentially

Main properties of regularized fluid flows equations

Application of the presented procedure to

compressible gas flow equations and to the incompressible fluid equations produces equations which are consistent with original governing equations. It was proven (by prof. Yu. V. Sheretov) that:

1. regularized and original NS systems have a number of common exact solutions;
2. regularized gas dynamic and hydrodynamic systems obey laws of:
   - mass conservation
   - momentum conservation
   - total energy conservation
   - angular momentum conservation
3. the dissipative functions of both systems are positive, ensuring fulfillment of second law of thermodynamic

Numerical algorithms for regularized equations

are much simpler than for the original NS, especially for the incompressible flows

Numerical approximation of 1D regularized scalar transport equation

One of the key benefit of QGD algorithms is their ability to work with very simple approximation expressions – in most cases naive approximations of $\tau-$terms works well:

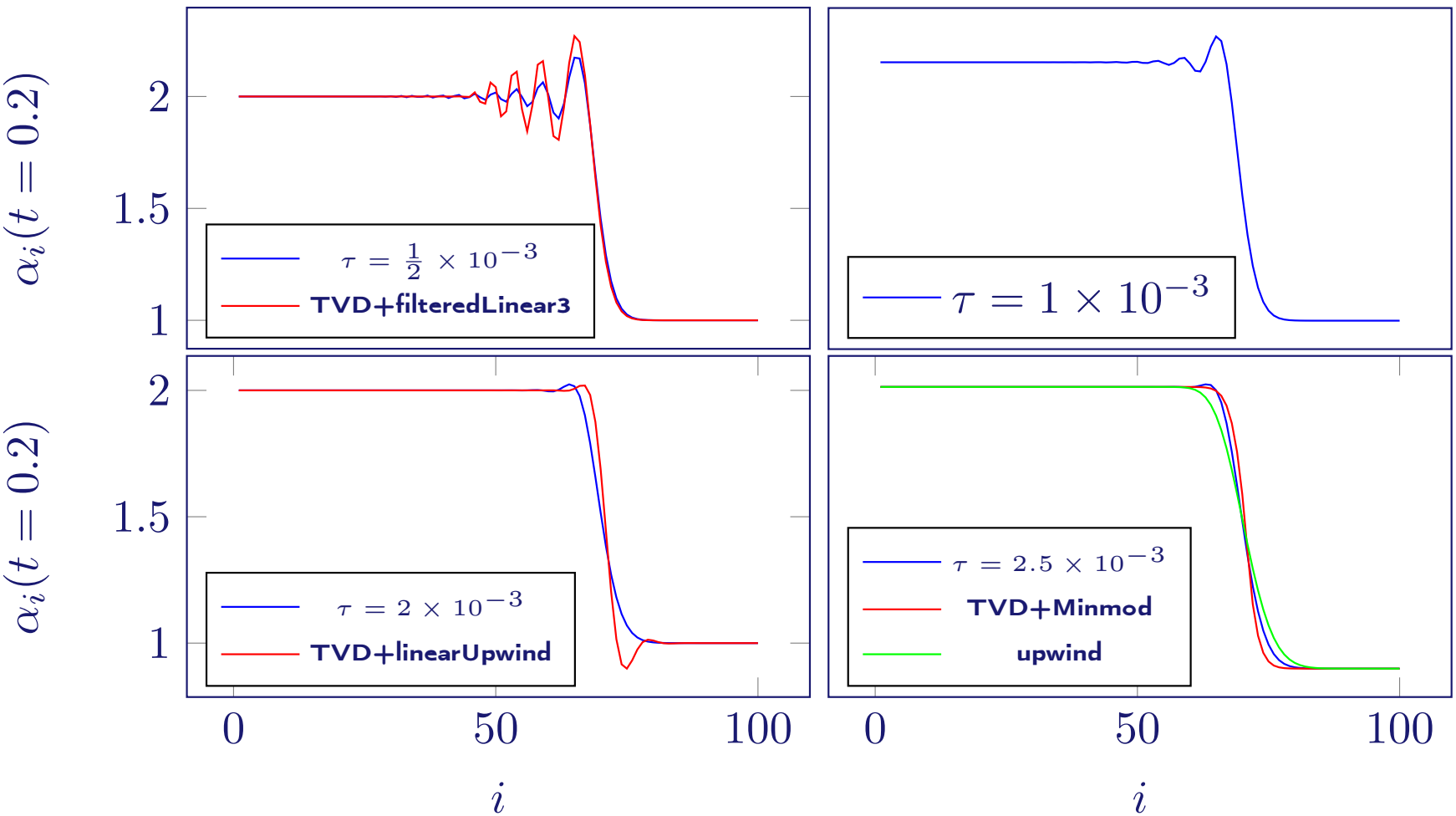
$$\Delta x \frac{\alpha_i^n - \alpha_i^o}{\Delta t} + \left(U_{i+1/2}\alpha^o_{i+1/2}\right) - \left(U_{i-1/2}\alpha^o_{i-1/2}\right) - (\tau\vec{U})_{i+1/2}\left(\frac{\delta\vec{U}\alpha}{\delta x}\right)_{i+1/2} + (\tau\vec{U})_{i-1/2}\left(\frac{\delta\vec{U}\alpha}{\delta x}\right)_{i-1/2} = 0$$

Here

$$\left(\frac{\delta\vec{U}\alpha}{\delta x}\right)_{i-1/2} = \frac{(\vec{U}\alpha)_i - (\vec{U}\alpha)_{i-1}}{x_i - x_{i-1}}$$

Numerical experiment

Consider previous example of the wave transport, but using regularized equation with different values of $\tau-$ parameter. It is seen from the figure below that growth in τ reduces oscillations



Notes on approximation

1. Regularization term $\left(\vec{U}\tau\nabla\cdot\left(\vec{U}\alpha\right)\right)$ should be calculated in half-points (face-centers of the unstructured mesh) and it could be evaluated in several ways, for example:
   - by approximating flux $\left(\vec{U}\alpha\right)$ at face centers with next evaluation of flux divergence at cell centers and it's interpolation to face centers
   - by using finite-difference methods on unstructured and structured grids to calculate partial derivatives
2. three ways to calculate partial derivatives on 3D unstructured grids are considered in "QGDsolver" library:
   - least-squares method
   - Ostrogradsky-Gauss method
   - truncation of tangential direction derivatives
3. despite the claimed simplicity of approximation, the choice of ways to group terms (e.g. $(\tau\vec{U})_{i-1/2}$ or $\tau_{i-1/2}\vec{U}_{i-1/2}$) sometime could significantly impact an algorithm's stability
4. to find the best approximation for regularized fluid flow equations, specialists use entropy-conservative approximation (Yu.V. Sheretov "Regularized hydrodynamic equations" , A. A. Zlotnik, `https://doi.org/10.1134/S0965542517020166`)

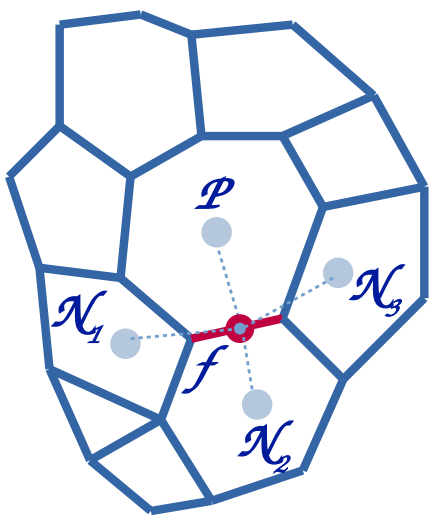
Least squares method

By considering Taylor's expansion of α near the face center f

$$\alpha_i = \alpha_f + (\vec{x}_f - \vec{x}_i) \cdot \nabla\alpha_f + e_i$$

and minimizing

$$e_f^2 = \sum_i w_i^2 e_i^2$$

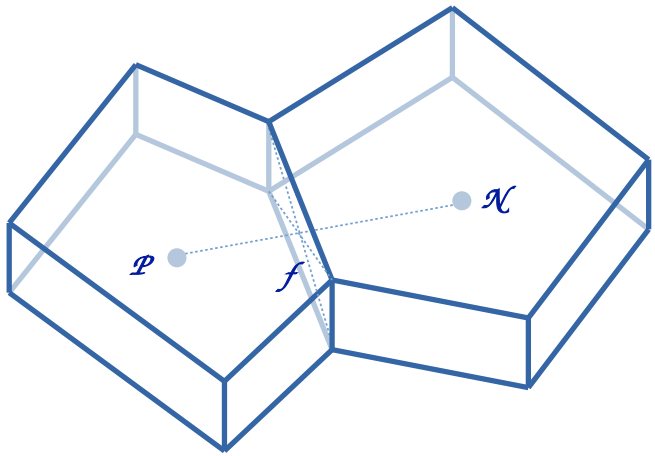


We get the approximate expression for α field gradient:

$$(\nabla\alpha)_f \approx \hat{G}^{-1} \sum_i w_i^2 \vec{d}_i \otimes \vec{d}_i \, (\alpha_i - \alpha_f)$$

$$\vec{d}_i = \vec{x}_f - \vec{x}_i$$
$$w_i = |\vec{d}_i|^{-1}$$

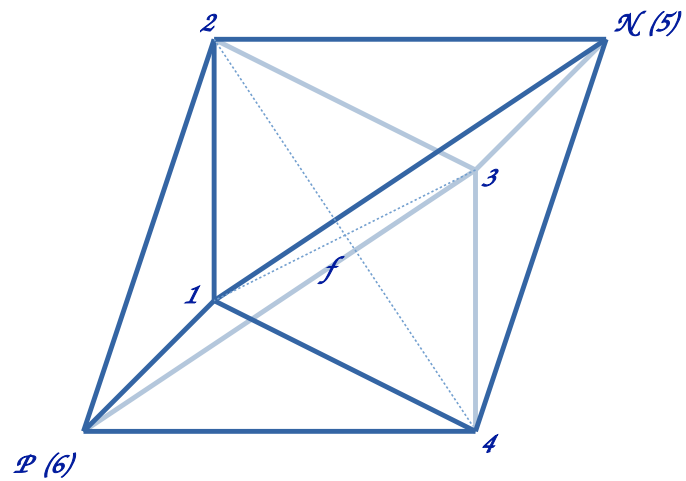
Ostrogradsky-Gauss method



The partial derivative on the face f centroid can be calculated as:

$$\frac{\partial \alpha}{\partial x} = \frac{1}{V_f} \sum_{m=1}^{6} n_{m,x} \alpha_m$$

- The formula can be extended to face with arbitrary number of vertices.
- Thanks to *snappyHexMesh*, most faces are quadrangle
- For other faces we can use *reduced* approach (next slide)



All formulae could be found in M. Istomina, E. Shilnikov 2019, KIAM Preprint No. 86 (In Russian)

Reduced method

Within the so-called "reduced" method of τ-terms approximation the derivatives in direction tangential to mesh faces are neglected:

- Gradient of scalar α

$$(\nabla\alpha)_f \approx \frac{\partial\alpha}{\partial\vec{n}_f}\vec{n}_f$$

- Divergence of vector $\vec{U}$

$$(\nabla\cdot\vec{U})_f \approx \vec{n}_f \cdot \frac{\partial\vec{U}}{\partial\vec{n}_f}$$

- Gradient of vector $\vec{U}$

$$(\nabla\vec{U})_f \approx \vec{n}_f \otimes \frac{\partial\vec{U}}{\partial\vec{n}_f}$$

- Divergence of tensor $\hat{\Pi}$

$$(\nabla\cdot\hat{\Pi})_f \approx \vec{n}_f \cdot \frac{\partial\hat{\Pi}}{\partial\vec{n}_f}$$

The method doesn't endorsed by developers of QGD/QHD algorithms, but simple and efficient

Pro's and Con's of QGD

Advantages of QGD algorithms

- they can work without flux limiters
- they converge monotonically to real solution
- they do not involve Rieman-solvers
- the procedure of approximation is universal for all types of flows
- they can be integrated with other OpenFOAM models
- by contrast to PISO/SIMPLE they don't involve non-orthogonal or pressure-velocity correctors
- all abovementined features make QGD algorithms a useful tool for studying transient flows phenomena

Drawbacks of QGD algorithms

- they are usually slower (3-4 times) than conventional PISO or Godunov-type methods
- additional conditions are imposed for stability criteria
- they require finer grids and smaller time steps in comparison with PISO algorithm for advection-dominated flows

QGD Target audience

According to stated advantages and drawbacks of QGD algorithms, they could be useful to:

- scientists, who want to solve complex set of equations, but still haven't elaborated PISO/SIMPLE or Godunov-type procedure
- researches or engineers who want to validate other methods and programs and numerical models, but they don't have analytic solution
- engineers, who want to simulate complex transient flows which could not be reproduced by PISO/SIMPLE algorithms

QGDsolver framework

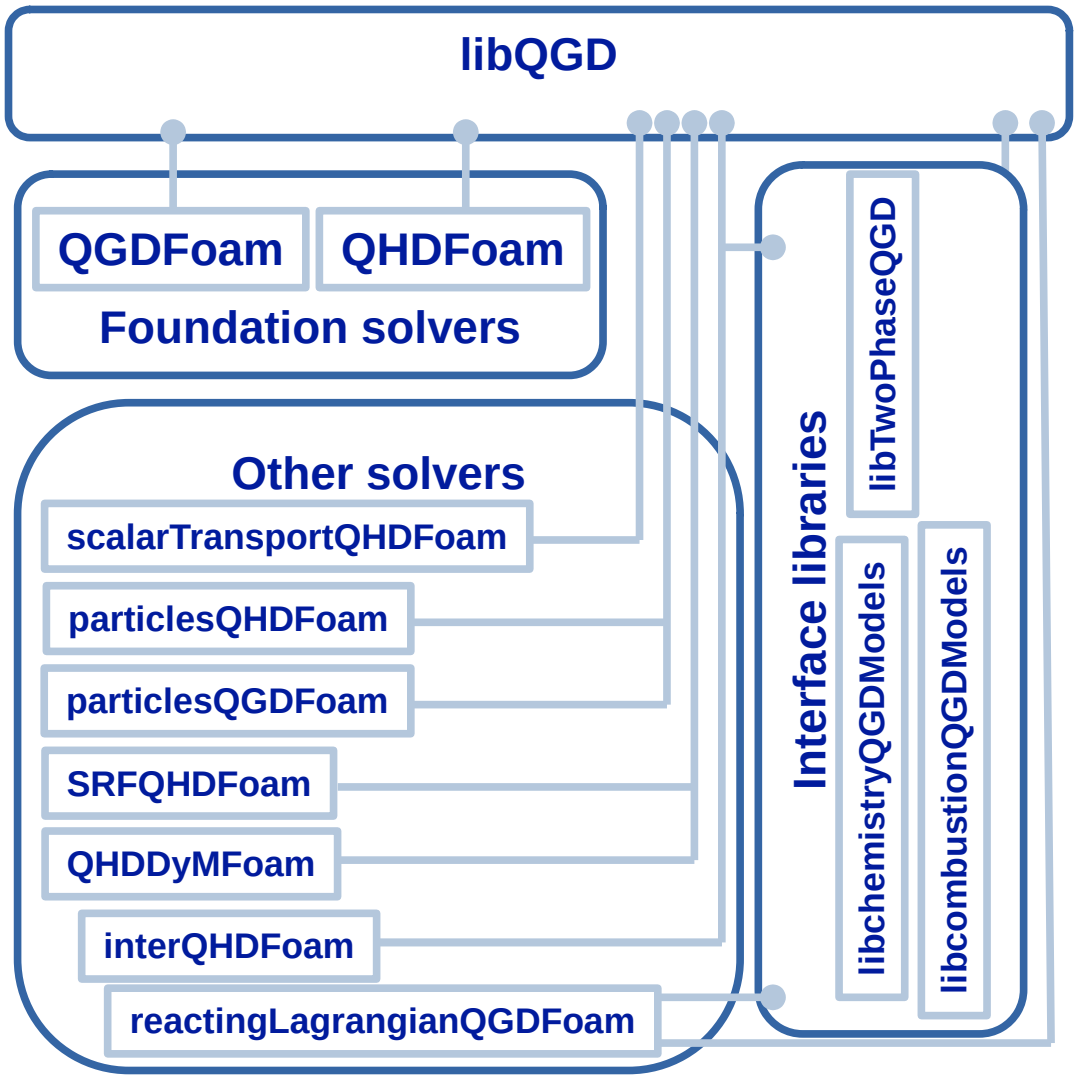
The QGDsolver framework is based on OpenFOAM+ technology and includes:

- Numerical simulation solvers for various types of flows based on the regularized equations
- τ-terms approximation library for calculation of partial derivatives on mesh faces
- Boundary conditions, QGD coefficients library
- Interfaces to some standard OpenFOAM libraries
- Tutorials

Source code stores at GitHub:
`https://github.com/unicfdlab/QGDsolver`

- Branches "digitef-dev-ABCD" are for OpenFOAM+ ver. ABCD
- Latest OpenFOAM version is OpenFOAM+ ver. 1912
- Releases are also available

QGDsolver framework structure



- Each solver implementing QGD algorithm must use libQGD library
- Two foundation solvers *QGDFoam* and *QHDFoam* show essential principles of QGD-algorithms
- Other solver could be regared as combination of foundation algorithms and OpenFOAM models
- Interface libraries are used to connect QGD solver to OpenFOAM models when interfaces have changed

OpenFOAM QGDsolver collaborators

People

- Prof. Tatiana G. Elizarova
- Maria A. Istomina
- Tatiana V. Stenina
- Daniil A. Ryazanov
- Andrey S. Epikhin
- Alexander V. Ivanov
- Kirill A. Vatutin
- Matvey V. Kraposhin

Institutions

- Keldysh institute for Applied Mathematics of the RAS
- Ivannikov Institute for System Programming of the RAS

Solvers groups

Foundation solvers

- QGDFoam – all Mach numbers compressible viscous flow of perfect gas
- QHDFoam – incompressible viscous flow with buoyancy

Miscellaneous solvers

- Mesh motion and reference frame rotation
  - QHDDyMFoam
  - SRFQHDFoam
- Transport phenomena
  - scalarTransportQHDFOam
  - mulesQHDFoam
- Flow with particles
  - particlesQGDFoam
  - particlesQHDFoam
- Multicomponent and multiphase flows
  - reactingLagrangianQGDFoam
  - interQHDFoam

General notes on case setup

0/ folder

- Since all solvers use thermophysical libraries, at least 3 fields are needed to be specified in "0" folder: "T", "U", "p"

constant/ folder

- Thermophysical libraries are used through interfaces: hePsiQGDThermo instead of hePsiThermo, heRhoQGDThermo instead of heRhoThermo
- QGD regularization parameters are set in subdictionary "QGD" of thermophysicalProperties dictionary

system/ folder

- Approximation scheme for $\tau-$terms is set in the "fvsc" subdictionary of fvSchemes dictionary
- Proportionality between maximum time step and τ is controlled via $Ctau$ keyword of controlDict dictionary

QHDFoam

- Application of regularization procedure to incompressible Navier-Stokes equations with buoyancy force yields regularized hydrodynamic equations
- The approach was originally developed by prof. Yu. V. Sheretov
- First approximation and testing of equations were made in close cooperation with prof. T.G. Elizarova
- The implementation of the numerical algorithm, based on the QHD equations is "QHDFoam". It's closes rivals amongst standard OpenFOAM solvers are:
  - "pisoFoam"
  - "buoyantBoussinesqPimpleFoam"
- It's main differences from standard OpenFOAM solvers – ability to work at high Re numbers without TVD schemes, absence of non-orthogonal and pressure-velocity correctors, pressure equation with boundary conditions are direct consequence of continuity equation

QHDFoam equations

The regularized system of hydrodynamic equations comprises of:

- continuity equation:

$$\nabla \cdot \left(\vec{U} - \vec{W} \right) = 0, \quad \vec{W} = \tau \left((\vec{U} \cdot \nabla)\vec{U} + \frac{1}{\rho} \nabla \tilde{p} - \beta \vec{g} \tilde{T}) \right)$$

- momentum equation:

$$\frac{\partial \vec{U}}{\partial t} + \nabla \cdot \left(\left(\vec{U} - \vec{W} \right) \otimes \vec{U} \right) + \frac{1}{\rho} \nabla \tilde{p} = \frac{1}{\rho} \nabla \cdot \hat{\Pi} + \beta \vec{g} \tilde{T}$$

- scalar (temperature) transport equation:

$$\frac{\partial T}{\partial t} + \nabla \cdot \left(\left(\vec{U} - \vec{W} \right) T \right) - \nabla \cdot \left(\tau \vec{U} \left(\vec{U} \cdot \nabla \right) T \right) - \nabla \cdot \left(\frac{\mu}{\rho Pr} \nabla T \right) = 0$$

- incompressible EoS and regularized stress tensor:

$$\rho = \rho_0 \left(1 + \beta \tilde{T}) \right), \quad \hat{\Pi} = \vec{U} \otimes \vec{W} + \hat{\Pi}_{NS}$$

QHDFoam approximation

The integration algorithm in "QHDFoam" is similar to "pisoFoam", but without pressure correction and non-orthogonal iterations

- First, the continuity equation is solved to get new pressure:

$$\sum_f \vec{U}_f^o \cdot \vec{S}_f = \sum_f \tau \left(\vec{U}^o \cdot \nabla \vec{U}^o + \frac{1}{\rho} \nabla \tilde{p}^n - \beta \vec{g} \tilde{T}^o) \right)_f \cdot \vec{S}_f$$

- Then, the momentum balance equation without viscous terms is solved, followed by viscous correction:

$$\left(\vec{U}\right)^p = \left(\vec{U}\right)^o - \frac{\Delta t}{V} \sum_f \Phi_f(\left(\vec{U}\right)^o) \qquad \frac{\vec{U}^n - \vec{U}^o}{\Delta t} - \frac{\vec{U}^p - \vec{U}^o}{\Delta t} = \frac{1}{V} \sum_f \vec{S}_f \cdot \hat{\Pi}_f^{NS}$$

- Then, the temperature without viscous terms is solved, followed by viscous correction:

$$T^p = T^o - \frac{\Delta t}{V} \sum_f \Phi(T^o) \qquad \frac{T^n - T^o}{\Delta t} - \frac{T^p - T^o}{\Delta t} = \frac{1}{V} \sum_f \left(\frac{\mu}{\rho Pr} \right)_f \frac{\delta T^n}{\delta \vec{n}_f} |\vec{S}_f|$$

- Finally, the viscosity μ is corrected by new temperature field

QHDFoam Boundary conditions

Inflow boundary conditions

- Fixed pressure and zero velocity gradient, if the flow is driven by pressure difference
- Fixed velocity and fixed pressure gradient, if the flow is driven by prescribed inlet flux rate

Outflow boundary conditions

- fixed total pressure and zero velocity gradient for flows driven by pressure difference and by prescribed inlet flux rate

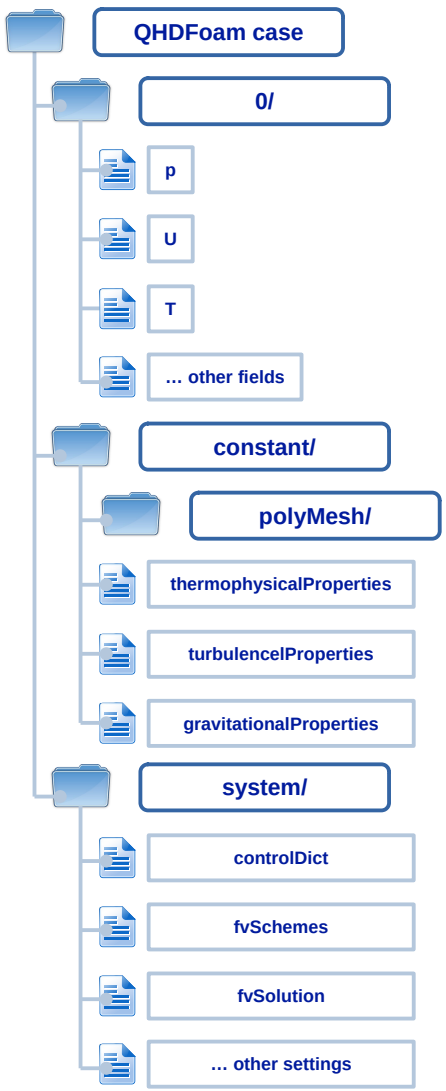
Impermeable wall boundary conditions

- Fixed value for velocity vector: $(0, 0, 0)$
- Fixed pressure gradient to balance τ-sterms of mass flux

QHDFoam equations coefficients

- Navier-Stokes stress-tensor and heat flux vector are approximated using Newtonian fluid assumption and Fourier law
- Value of τ coefficient is selected to be equal or less than some characteristic hydrodynamic time using characteristic velocity magnitude U, kinematic viscosity ν grid step Δx or other parameters:
  - $\tau \sim \nu / U^2$, $\tau \sim \Delta x / U$
  - For simple cases, τ could be estimated from dimensionless numbers (like Re, Gr or others): $\tau \approx \tau_0 Re^{-1}$, $\tau \approx \tau_0 Gr^{-1}$;
  - As inverse relation between viscosity and grid step: $\tau \approx \frac{\Delta x^2}{\mu / \rho}$
  - Through the max CFL $Co^{max} = |\vec{U}|^{max} \Delta t / \Delta x$ number: $\tau \approx \frac{Co^{max} \Delta x}{|\vec{U}|^{max}} C_\tau$, where C_τ is a constant less than 1
- Two stability criteria are used:
  1. $\Delta t < C_\tau \tau$, where $C_\tau \leq \frac{1}{2}$. In some cases C_τ could be set to 0.75
  2. $Co = \frac{U \Delta t}{\Delta x} < Co^{max}$. The Co^{max} is usually about $0.1 - 0.2$ in most cases

QHDFoam case structure



It is similar to *rhoPimpleFoam* case structure

Initial and boundary conditions

Initial conditions are set in the folder "0". Three fields are mandatory to start a simulation: pressure "p", velocity "U" and temperature "T"

Fluid properties

Thermophysical fluid properties (density, heat capacity coefficients, viscosity and heat conductivity coefficients) are set in "thermophysicalPropertis" dictionary. By default the turbulence modelling is turned off in the "turbulenceProperties" dictionary. Value and direction of gravity bulk field is set in "gravitationalProperties"

Numerical schemes

Numerical schemes settings are stored in "fvSchmes" and "fvSolution", time advancement control is in "controlDict"

QHDFoam QGD's specific settings

Choice of QGD coefficients model

By default a user must specify τ value. To change this model, use keyword "QGDCoeffs" in the subdictionary "QGD" of "thermophysicalProperties" dictionary. The name for default coefficients is "constTau".

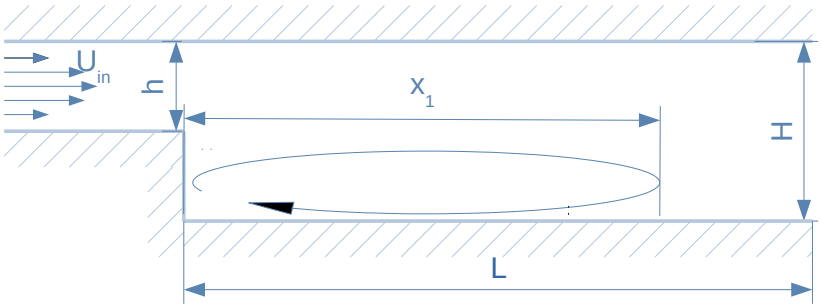
Choice of $\tau-$terms appoximation

- Usually QGD algorithms use the same technique (Gauss with linear interpolation) for all except $\tau-$ terms. One can set approximation method for $\tau-$ terms in the "fvsc" subdictionary of "fvSchemes" dictionary.
- Diffusion terms could be approximated with implicit or explicit scheme – the trigger "implicitDiffusion" is located in "thermophysicalProperties" dictionary.

Time step restriction

The value for C_τ : $\Delta t \leq C_\tau \tau$ is stored in "controlDict" dictionary. Default value $C_\tau = 0.75$

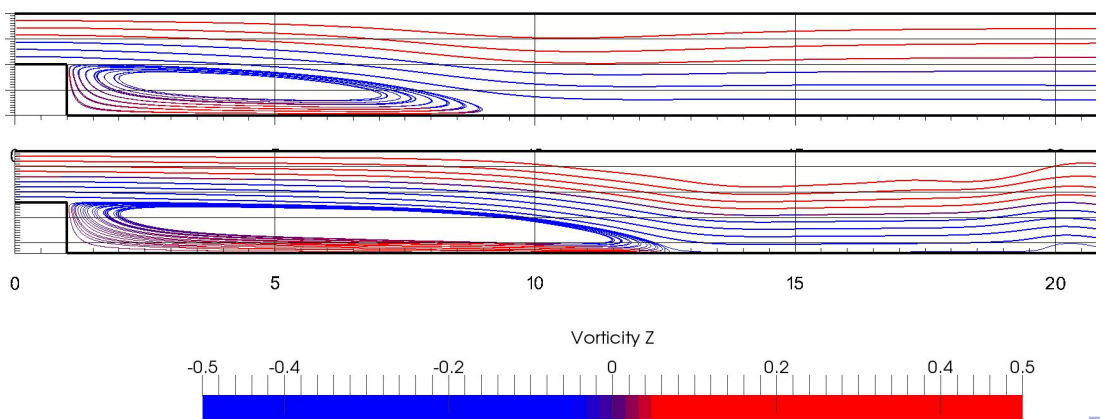
QHDFoam test case: backward step



Viscous flow of initially resting fluid with different $Re = hU_{in}/\nu$ numbers,gravity is zero. The length of separation zone is compared with experiment

QHD settings

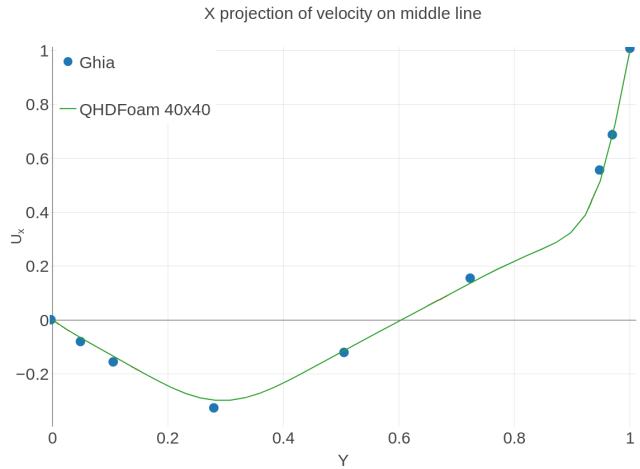
- $\tau = \frac{h}{U_{in}}$, $\Delta t = \frac{1}{2}\tau$
- Implicit diffusion approximation is "on"
- Least Squares Method is used for $\tau-$terms approximation

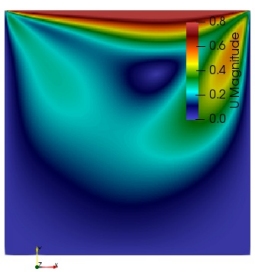


Re	Exp.	QHDFoam
50	3.0	2.9
100	5.0	4.9
200	8.5	8.2
300	11.3	10.2
400	14.2	11.6

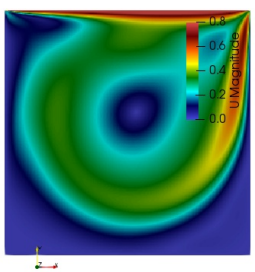
QHDFoam test case: cavity

The renowned lid-driven cavity flow case is used to demonstrate peculiarities of the QHD approach. The simulation conducted for different Re numbers from 10^2 to 10^5 but without applying any flux limiters (TVD schemes) – see below instant velocity magnitude after simulation of 10 seconds.

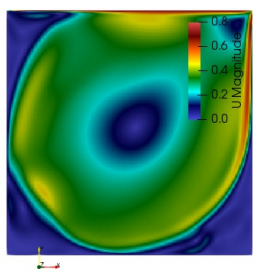




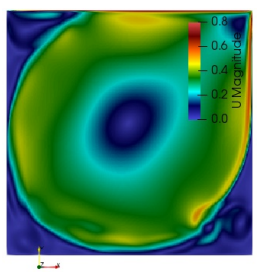
$Re = 10^2$



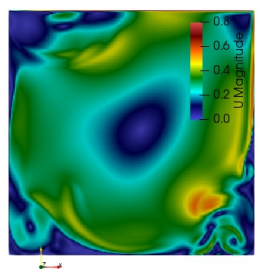
$Re = 10^3$



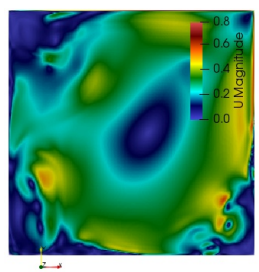
$Re = 10^4$



$Re = 2 \times 10^4$



$Re = 5 \times 10^4$



$Re = 10^5$

Case parameters

- $Re = U_{lid} H_{lid} / \nu$, $U_{lid} = 1$ m/s, $H_{lid} = 1$ m
- $\tau = \Delta x / U_{lid}$
- Grid size: 250×250
- Spatial approximation: Gauss + linear, Least Squares Methods for $\tau-$ terms

QGDFoam

- Application of regularization procedure to compressible perfect gas equations yields regularized gasdynamic equations or QGD equations
- The approach was developed by russian academician prof. B.N. Chetverushkin and prof. T.G. Elizarova
- The implementation of the numerical algorithm, based on the QGD equations is "QGDFoam". It's closes rivals amongst standard OpenFOAM solvers are:
  - "rhoCentralFoam", but QGDFoam's lower Mach boundary is 0.01 or even less
  - "rhoPimpleFoam", but QGDFoam's upper Mach boundary is limited only by computational resources
  - "pimpleCentralFoam", but QGDFoam uses homogeneous approximation for all Mach numbers
- It's main differences from standard OpenFOAM solvers – ability to work at high Re numbers without TVD schemes, without non-orthogonal and pressure-velocity correctors

QGDFoam equations

The regularized system of gad dynamic equations comprises of:

- continuity equation:

$$\frac{\partial \rho}{\partial t} + \nabla \cdot \vec{j}_m = 0$$

- momentum equation:

$$\frac{\partial \rho \vec{U}}{\partial t} + \nabla \cdot (\vec{j}_m \otimes \vec{U}) + \nabla p = \nabla \cdot \hat{\Pi}$$

- energy equation:

$$\frac{\partial \rho e}{\partial t} + \nabla \cdot (\vec{j}_m (e + p/\rho)) = \nabla \cdot \left(\hat{\Pi} \cdot \vec{U} \right) - \nabla \cdot \vec{q}$$

- perfect gas EoS:

$$p = \rho \tilde{R} T, \quad u = e - 1/2 \vec{U} \cdot \vec{U}$$

- regularized mass flux:

$$\vec{j}_m = \rho(\vec{U} - \vec{W}), \vec{W} = \frac{\tau}{\rho}(\nabla \cdot (\rho \vec{U} \otimes \vec{U}) + \nabla p)$$

- regularized stress tensor:

$$\hat{\Pi} = \Pi_{NS} + \tau \vec{U} \otimes \left(\rho(\vec{U} \cdot \nabla) U + \nabla p \right) + \tau \hat{I} \left((\vec{U} \cdot \nabla) p + \gamma p \nabla \cdot \vec{U} \right)$$

- regularized heat flux:

$$\vec{q} = \vec{q}_{NS} - \tau \rho \vec{U} \left((\vec{U} \cdot \nabla) u + p (\vec{U} \cdot \nabla) \frac{1}{\rho} \right)$$

QGDFoam approximation and algorithm

The segregated semi-implicit finite-volume approach is implemented in "QGDFoam"

- First, the mass or continuity equation is solved:

$$\rho^n = \rho^o - \frac{\Delta t}{V}\sum_f \Phi_f(\rho^o)$$

- Then, the momentum balance equation without viscous terms is solved, followed by viscous correction:

$$\left(\rho\vec{U}\right)^p = \left(\rho\vec{U}\right)^o - \frac{\Delta t}{V}\sum_f \Phi_f(\left(\rho\vec{U}\right)^o) \qquad \frac{\rho^n\vec{U}^n - \rho^o\vec{U}^o}{\Delta t} - \frac{\rho^n\vec{U}^p - \rho^o\vec{U}^o}{\Delta t} = \frac{1}{V}\sum_f \vec{S}_f \cdot \hat{\Pi}_f^{NS}$$

- Then, the energy equation without viscous terms is solved, followed by viscous correction:

$$(\rho e)^p = (\rho e)^o - \frac{\Delta t}{V}\sum_f \Phi((\rho e)^o)$$

$$\frac{\rho^n u^n - \rho^o u^o}{\Delta t} - \frac{\rho^n u^p - \rho^o u^o}{\Delta t} = \frac{1}{V}\sum_f \left(\frac{\kappa}{C_v}\right)_f \frac{\delta u^n}{\delta \vec{n}_f}|\vec{S}_f| + \frac{1}{V}\sum_f \vec{S}_f \cdot \left(\hat{\Pi}_f^{NS} \cdot \vec{U}_f^o\right)$$

- Finally, the pressure and temperature are corrected through EoS

QGDFoam Boundary conditions

Inflow boundary conditions

- For supersonic inflow – fixed pressure, temperature and velocity
- For subsonic inflow – fixed temperature + incompressible inflow BCs (see QHDFoam BCs)

Outflow boundary conditions

- For supersonic outflow — zero derivative in normal direction for pressure, temperature and velocity
- For subsonic outflow — zero gradient for temperature + incompressible outflow BCs (see QHDFoam BCs)

Wall boundary conditions

- Zero fixed value velocity for no-slip walls
- Zero normal component of velocity for walls with slip
- Zero pressure gradient

QGDFoam equations coefficients

- Navier-Stokes stress-tensor and heat flux vector are approximated using Newtonian fluid assumption and Fourier law
- Values of τ coefficients:
  - For rarified flows τ is a mean time between collisions: $\tau = \frac{\mu}{pSc}$, where Sc – gas Schmidt number
  - For dense flows additional numerical dissipation is needed:
    - $\tau = \alpha^{QGD} \frac{\Delta x}{c}$, where Δx – computational grid step
    - $\tau = \alpha^{QGD} \frac{\Delta x}{c} + \frac{\mu}{pSc}$, where α^{QGD} – tuning parameter in the range $[0, 5]$
  - For flows with shocks even more dissipation and/or artificial velocity diffusion are needed $\mu^{Eff} = \mu^{QGD} + \mu$:
  - The value of μ^{QGD} is derived from τ by analogy to rarified flows $\mu^{QGD} = \tau p Sc^{QGD}$, where Sc^{QGD} is a:
    - constant in a range from 0 to 1
    - function depending on flow regime (Mach number)
    - or Sc^{QGD} could be regarded as shock indicator, for example:

$$Sc_i^{QGD} = \frac{p_{i+1} - 2p_i + p_{i-1}}{p_{i+1} + 2p_i + p_{i-1}}$$

QGDFoam stability region

Two criteria are used

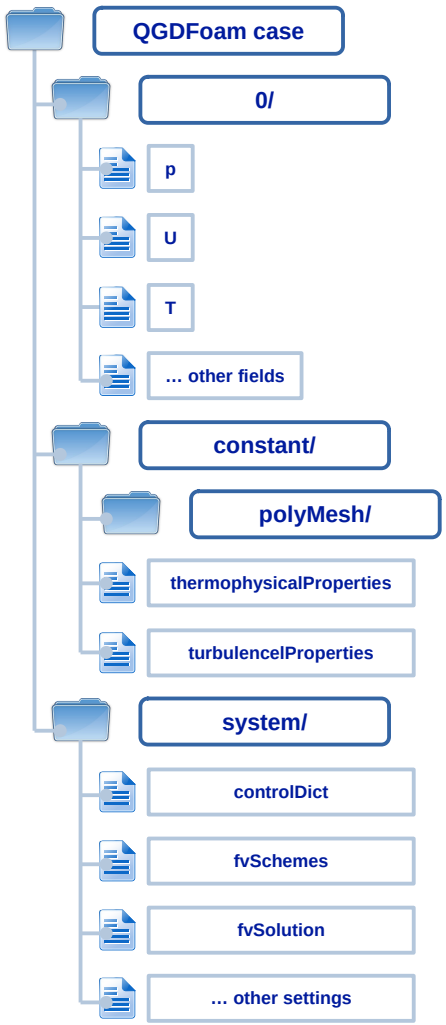
1. $\Delta t < C_\tau \tau$, where $C_\tau \leq \frac{1}{2}$. In some cases C_τ could be set to 0.75
2. $Co = \frac{max(U+c, -U+c)\Delta t}{\Delta x} < Co^{max}$

Co^{max} value

- Co^{max} depends on Mach number – the larger Mach number the less Co^{max} ought to be
- For Mach numbers equal to 1 or less, the Co^{max} is about 1
- For Mach numbers equal to 10-20, the Co^{max} is about 0.1

QGDFoam case structure

It is similar to *rhoCentralFoam* or *pimpleCentralFoam*



Initial and boundary conditions

Initial conditions are set in the folder "0". Three fields are mandatory to start a simulation: pressure "p", velocity "U" and temperature "T"

Gas properties

Thermophysical gas properties (molar weight, heat capacity coefficients, viscosity and heat conductivity coefficients) are set in "thermophysicalPropertis" dictionary. By default the turbulence modelling is turned off in the "turbulenceProperties" dictionary

Numerical schemes

Numerical schemes settings are stored in "fvSchmes" and "fvSolution", time advancement control is in "controlDict"

QGDFoam QGD's specific settings

Custom ICs and BCs for α^{QGD} and Sc^{QGD} fields

The default value for α^{QGD} is 0.5, Sc^{QGD} is 1. To set custom values, create files "alphaQGD" and "ScGQD" in the folder "0/"

Choice of QGD coefficients model

By default the model with constant $Sc = 1$ and $\tau = \alpha^{QGD} \frac{\Delta x}{c}$ is used. To change this model, use keyword "QGDCoeffs" in the subdictionary "QGD" of "thermophysicalProperties" dictionary. The name for default coefficients is "constScPrModel1".

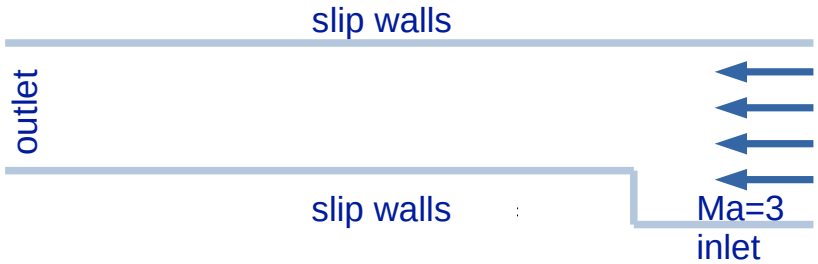
Choice of $\tau-$terms appoximation

- Usually QGD algorithms use the same technique (Gauss with linear interpolation) for all except $\tau-$ terms. One can set approximation method for $\tau-$ terms in the "fvsc" subdictionary of "fvSchemes" dictionary.
- Diffusion terms could be approximated with implicit or explicit scheme – the trigger "implicitDiffusion" is located in "thermophysicalProperties" dictionary.

Time step restriction

The value for C_τ : $\Delta t \leq C_\tau \tau$ is stored in "controlDict" dictionary. Default value $C_\tau = 0.75$

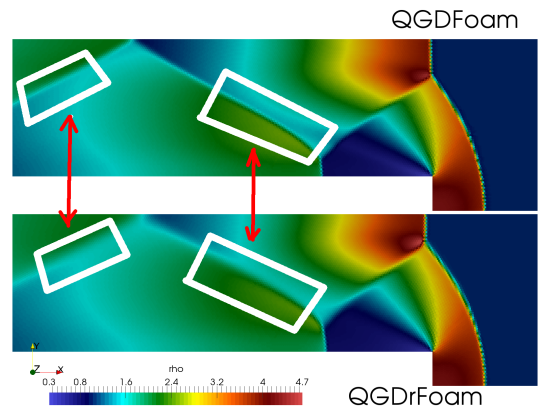
QGDFoam test case: forward step

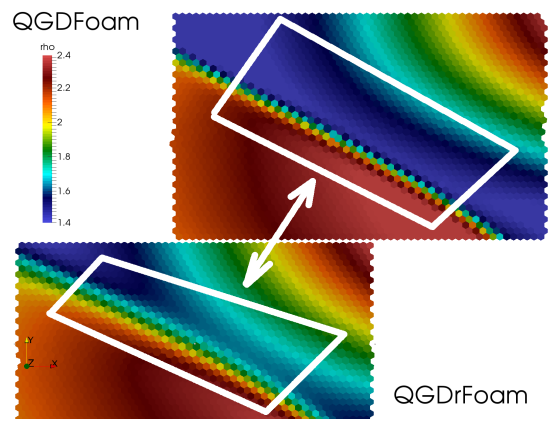


Inviscid flow of initially moving with uniform $Ma = 3$ gas with $\gamma = 1.4$ near the obstacle is considered.

QGD settings

- $\alpha^{QGD} = 0.5$
- $Sc^{QGD} = 1$
- Implicit diffusion approximation is "on"
- $Co^{max} = 0.2$
- Two settings for $\tau-$terms approximation considered:
  - Least Squares Method, denoted as "QGDFoam" **behaves like Minmod**
  - Reduced Method, denoted as "QGDrFoam" **behaves like upwind**





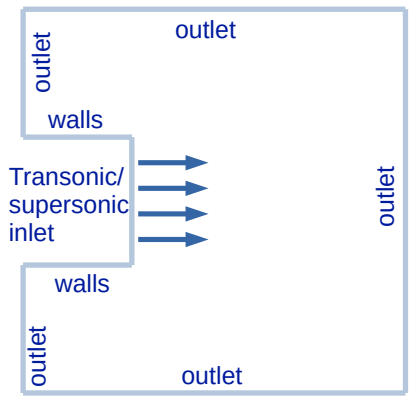
QGDFoam test case: jet flow

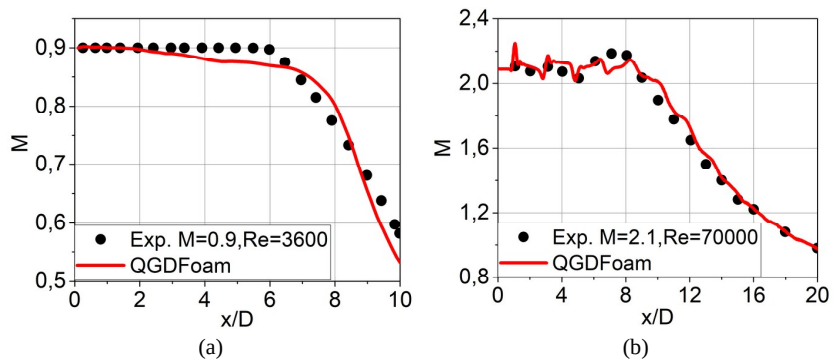
Two free jets of perfect gas with $\gamma = 1.4$ were considered:

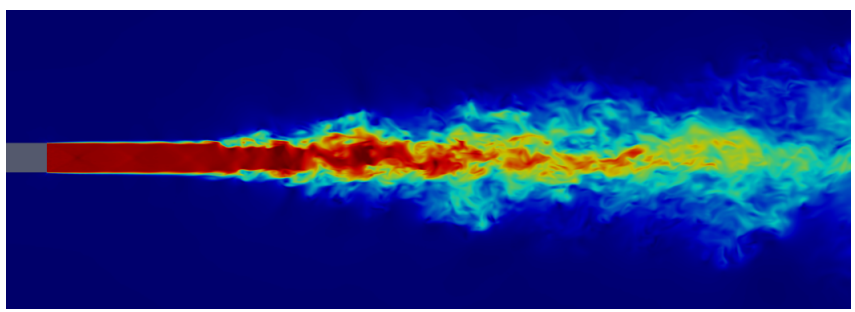
1. $Re = 3600$
2. $Re = 70000$

QGD settings was:

- $\alpha^{QGD} = 0.15$, $Co^{max} = 0.15$
- Sc^{QGD} was gradually reduced from 1 at the start to 0 at the end of calculation
- Ostrogradsky-Gauss appoximation of $\tau-$terms
- laminar model for $Re = 3600$ was used
- Smagorinsky model for $Re = 70000$ was used
- Computational mesh with 33 mln cells was used in both cases







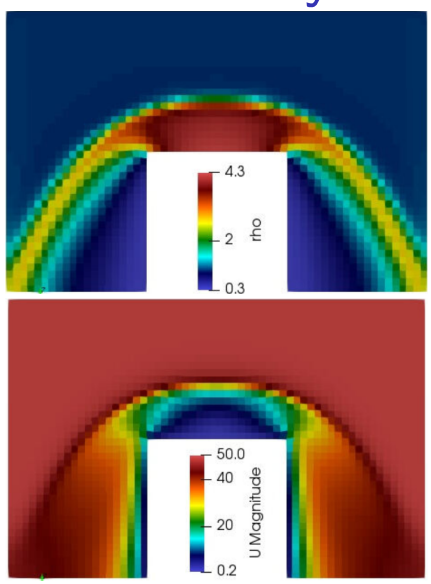
QGDFoam test case: Mach 50 flow

Inviscid Mach 50 flow of perfect gas $\gamma = 1.667$ around cylinder is considered.

QGD settings

- $Co^{max} = 0.015$
- $\tau = \alpha^{QGD} \frac{\Delta x}{c}$
- $\alpha^{QGD} = 0.5$
- $Sc^{QGD} = 0.5$ in internal field
- $Sc^{QGD} = 0$ on the cylinder
- The mesh counts 137k cells
- Ostrogradsky-Gauss appoximation of $\tau-$terms

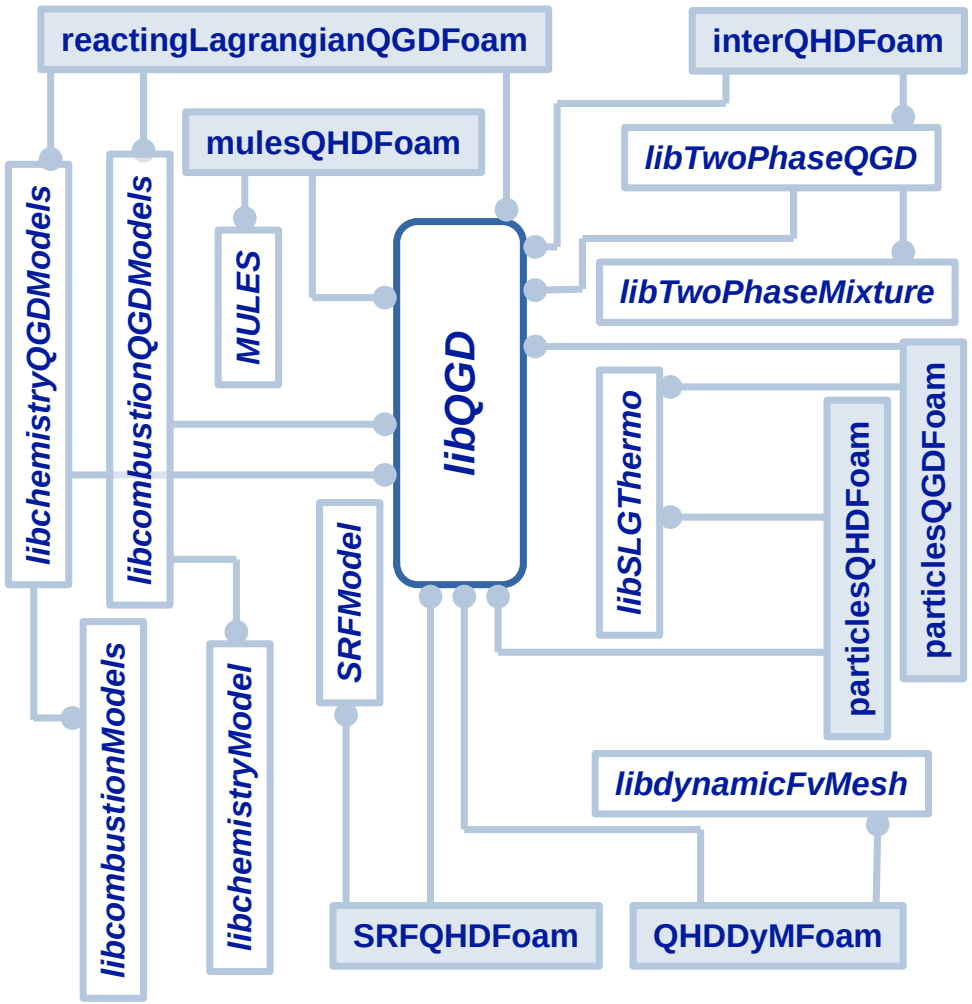
Velocity and density in slice



QGDCoeffs types

QGDCoeffs	Formula
constScPrModel1	$\tau = \alpha^{QGD}\frac{\Delta x}{c}$ $\mu^{QGD} = \tau p Sc^{QGD}$
constScPrModel1n	$\tau = \alpha^{QGD}\frac{\Delta x}{\vert\vec{U}\vert+c}$ $\mu^{QGD} = \tau p Sc^{QGD}$
constScPrModel2	$\tau = \alpha^{QGD}\frac{\Delta x}{c} + \frac{\mu}{pSc}$ $\mu^{QGD} = \alpha^{QGD}\frac{\Delta x}{c} p Sc^{QGD}$
varScModel6	$\tau = \alpha^{QGD}\frac{\Delta x}{c}$ $Sc^{QGD} = \frac{p_{i-1}+2p_i-p_{i+1}}{p_{i-1}+2p_i+p_{i+1}}$ $\mu^{QGD} = \tau p Sc^{QGD}$

Connections of QGD/QHD solvers to OpenFOAM models



- Some solvers are connected to OpenFOAM libraries through standard API
- Some – through interface classes, which which link regularization parameters with the solution vector and thermophysical variables

QHDDyMFoam: deformed mesh

Purpose of the solver

Simulation of flows around bodies with flexible or moving boundaries.

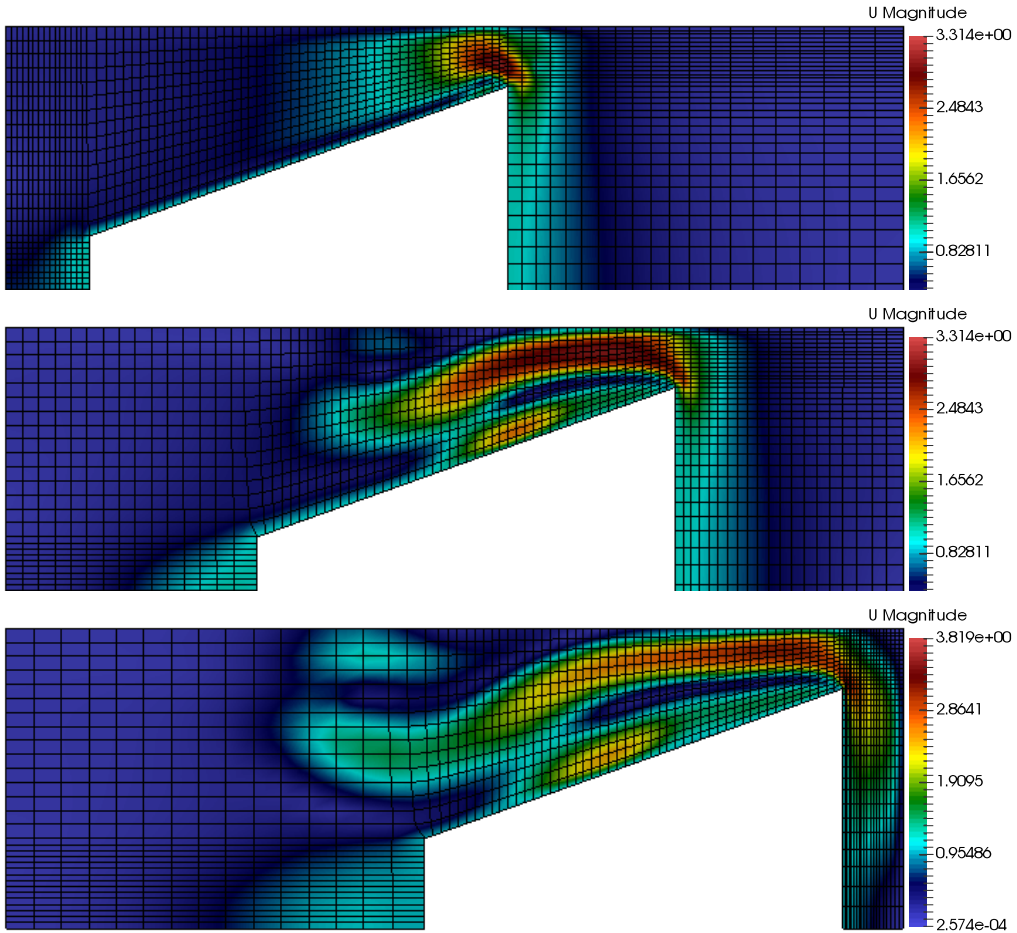
Changes in the foundation solver QHDFoam

- The original term in transport equations $(\vec{U} - \vec{W})$ is replaced with $\vec{U} - \vec{U}_b - \vec{W}$ to obey space continuity condition, where $\vec{U}_b$ is a field of body boundaries motion
- The "fvMesh" class is replaced with "dynamicFvMesh"
- Advection fluxes are updated with mesh motion after pressure correction:

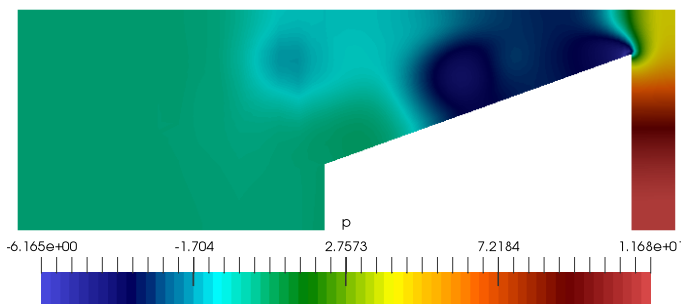
```
if (mesh.changing())
{
    fvc::makeRelative(phi, U);

    if (checkMeshCourantNo)
    {
        #include "meshCourantNo.H"
    }
}
```

QHDDyMFoam: moving cone



- Cone moves at constant velocity of 1 m/s
- liquid kinematic viscosity 10^{-5} m^2/s
- $\Delta t = 5 \times 10^{-6} s$
- $\tau = 10^{-6} s$
- Least Squares Method is used for τ-terms approximation



mulesQHDFoam: MULES algorithm

Purpose of the solver

Simulation of flows with zero temperature (infinity Pr) or zero mass diffusion using MULES technique implemented in OpenFOAM for advection-only transport of bounded scalars

Changes to the foundation solver QHDFoam

The scalar T transport equation reads:

$$\frac{\partial T}{\partial t} + \nabla \cdot \left((\vec{U} - \vec{W}) T \right) - \nabla \cdot \frac{\mu}{\rho Pr} \nabla T = 0$$

The source code to solve the equation is:

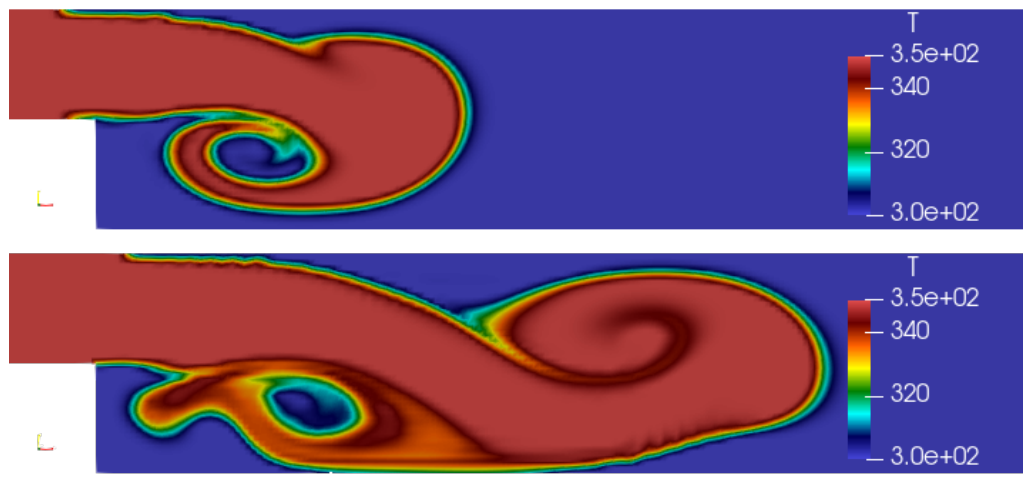
```
MULES::explicitSolve
(
    geometricOneField(),
    T,
    phi,
    phiTf,
    zeroField(),
    zeroField(),
    maxT,
    minT
);

solve(fvm::ddt(T)-fvc::ddt(T)
    - fvm::laplacian(Hif, T));
```

mulesQHDFoam: scalar transport w/o diffusion

- scalar transport w/o diffusion
- case: backward step, described above
- kinematic viscosity 2.5×10^{-4} m^2/s
- Least Squares Method is used for τ-terms approximation
- Inlet temperature $T_{in} = 350$ K
- $T|_{t=0} = 300$ K
- $\Delta t = 10^{-3}s$, $\tau = 0.1s$
- implicit diffusion approximation is “on”

- $\mathrm{Pr} = 10^5$
- $\rho = 1$



SRFQHDFoam: flow in rotational reference frame

Purpose of the solver

To solve equation of incompressible fluid flow with buoyancy in the rotating reference frame

Changes to the foundation solver QHDFoam

The bulk force expression changes to:

$$\vec{F_b} = \beta \tilde{T} \vec{g} - 2\vec{\Omega} \times \vec{U}$$

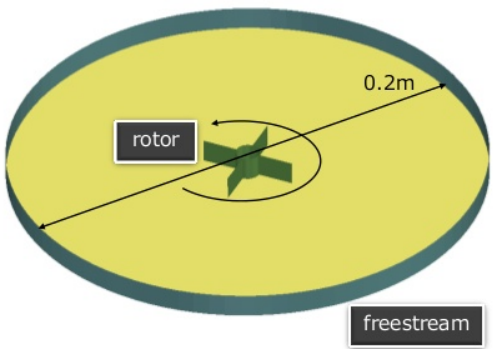
The source code to solve the equation is:

```
BdFrc = -beta*T*g - 2.0*(
    SRF->omega()^U);
```

SRFQHDFoam: 2D rotor

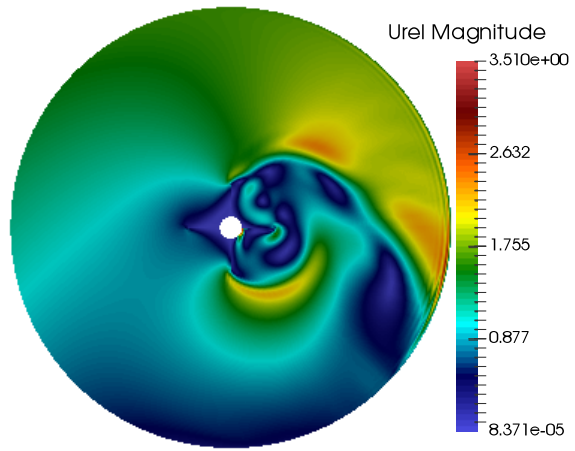
See: *tutorials/incompressible/SRFPimpleFoam/rotor2D*.

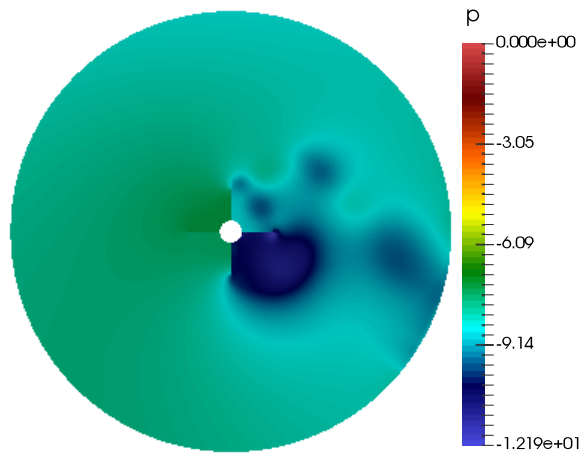
A wall named "rotor" is rotating counterclockwise at 60 rpm.



- liquid kinematic viscosity $10^{-4} m^2/s$
- $\tau = 10^{-4} s$
- $\Delta t = 10^{-6} s$
- Gauss Method is used for τ-terms approximation

results at time 0.5 s:





particlesQHDFoam, particlesQGDFoam: flow with particles

Purpose of the solver

To solve equation of incompressible (particlesQHDFoam) fluid or compressible gas (particlesQGDFoam) flow together with equations of particles clouds motion

Changes to the foundation solvers

The motion of particles clouds is described by equations:

$$\frac{d\vec{x}}{dt} = \vec{U}_p$$

$$\frac{d\vec{U}_p}{dt} = \frac{1}{\rho_p}\sum_i \vec{F}_i$$

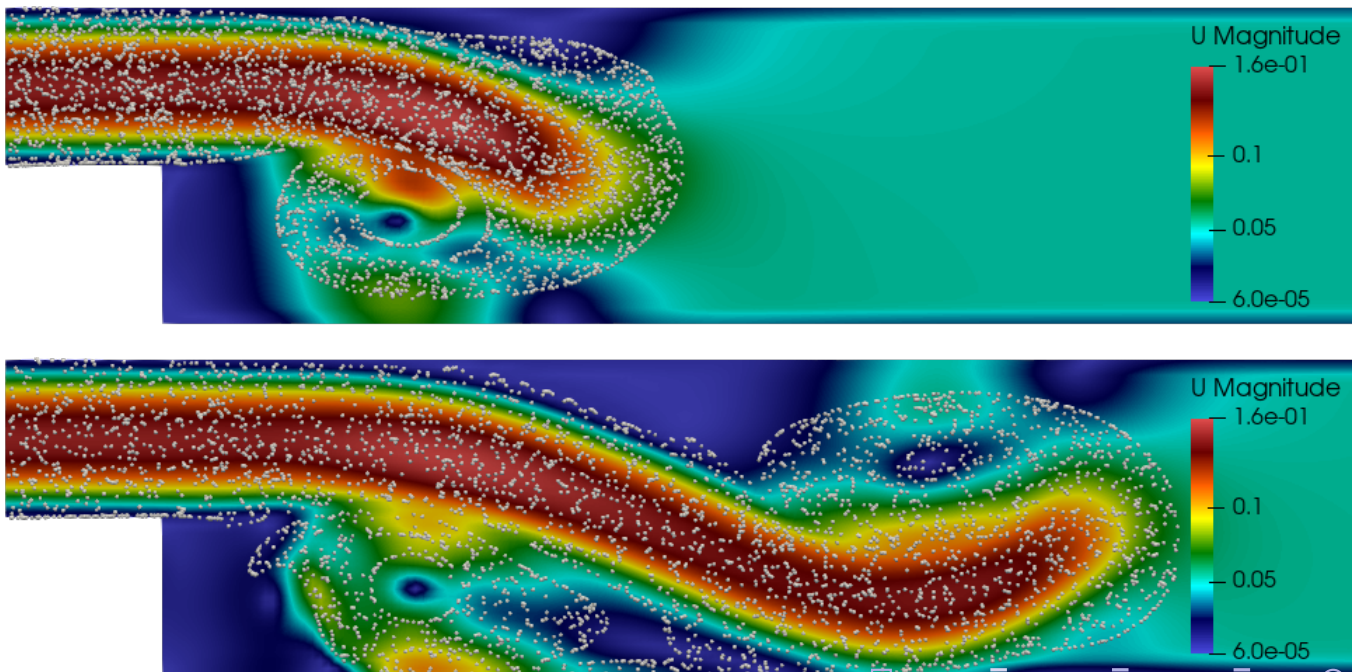
The source code to solve the equation is:

```
parcels.evolve();
```

particlesQHDFoam (particlesQGDFoam): backward step

Backward facing step considered in mulesQHDFoam case.

- particle type: copper $\rightarrow D_p = 70\mu m, \rho_p = 8800 kg/m^3$
- particles per second: 10^3, $U_0 = 9.39 m/s$
- liquid density $\rho_l = 1000$, liquid kinematic viscosity $1.5 \times 10^{-2} m^2/s$
- $\Delta t = 10^{-3} s$, $\tau = 0.1 s$
- Least Squares Method is used for τ-terms approximation



reactingLagrangianQGDFoam: multicomponent flow w/ reactions

Purpose of the solver

To solve regularized equations for compressible multicomponent all-Mach numbers gas flows with chemical reactions and particles motion, as well as mass, momentum and energy exchange between dispersed and continuous phases

Changes to the foundation solver

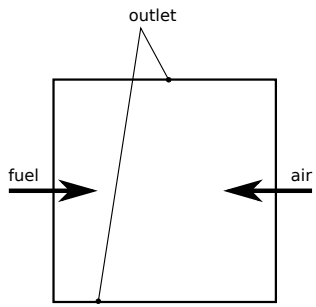
The system of equations is supplemented by the equation for the mass fraction Y_i balance of the specie No. i

$$\frac{\partial \rho Y_i}{\partial t} + \nabla \cdot \left(\vec{j}_m Y_i \right) - \nabla \cdot \left(\frac{\mu}{Sc_i} \nabla Y_i \right) - \nabla \cdot \left(\tau \rho \vec{U} \left(\vec{U} \cdot \nabla Y_i \right) \right) = S_i$$

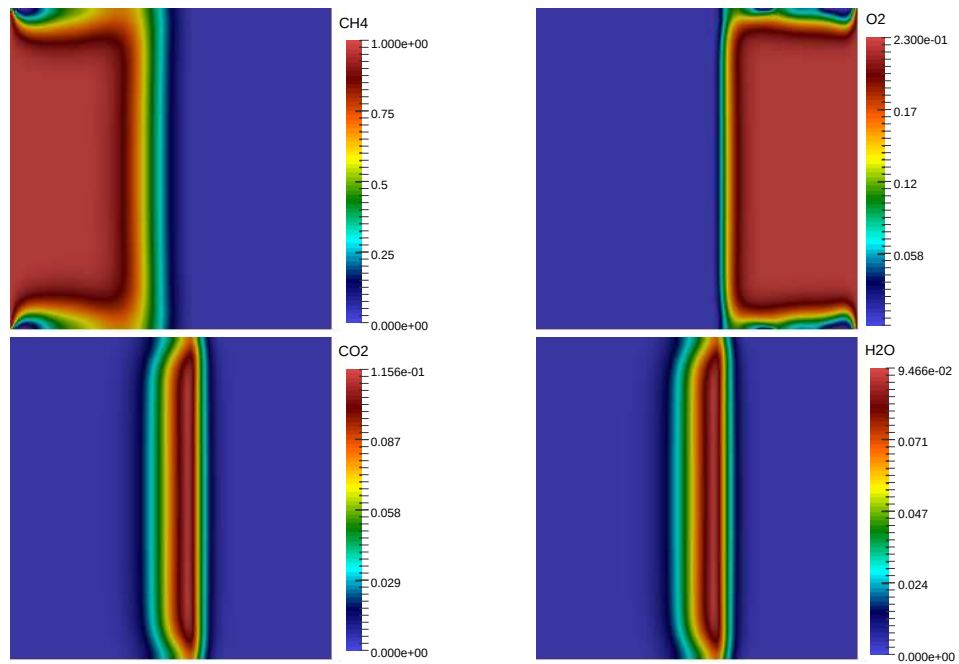
reactingLagrangianQGDFoam: basic reaction tutorial

Mixed combustion of methane and air (two-dimensional).

- square domain $\rightarrow$
- $\Delta t = 10^{-6} s$
- $\tau = \alpha^{QGD} \frac{\Delta x}{c}$
- $\mu^{QGD} = \tau p Sc^{QGD}$
- $Sc^{QGD} = 0$
- $Pr^{QGD} = 1$
  - methane (CH_4) from the region "fuel" and air (N_2 77%, O_2 23%) from the region "air" are injected at a flow rate of $0.1 m/s$
  - domain initially filled with N_2
  - the gas after combustion is discharged from the "outlet"



- reaction: $CH_4 + 2O_2 = CO_2 + 2H_2O$
- results at time 0.5 s:



interQHDFoam: Incompressible multiphase flow w/ surface tension

Purpose of the solver

To solve two-phase incompressible mixture equations with gravity and surface tension forces using regularized VoF approach

Changes to the foundation solver

Within the regularized VoF approach the equations of two incompressible fluids with diffrent densities are considered. To derive regularized 2-phase equation the previous procedure is used:

1. equations for each phase are regularized and summed to produce mixture equations
2. surface tension and buoyance forces are implemented in equations
3. resulting equations are approximated and solved using sequential solution procedure similar to “QHDFoam”

interQHDFoam: governing equations

Mixture volume continuity equation:

$$\nabla \cdot \left(\vec{U} - \vec{W} + \vec{U}\Delta\tau \left(\frac{\partial \alpha_1}{\partial t} \right)^* \right) = 0, \quad \vec{W} = \alpha_1 \vec{W}_1 + \alpha_2 \vec{W}_2, \quad \left(\frac{\partial \alpha_i}{\partial t} \right)^* = -\vec{U} \cdot \nabla \alpha_i$$

i-th phase volume fraction conservation

$$\frac{\partial \alpha_i}{\partial t} + \nabla \cdot \left(\left(\vec{U} - \vec{W}_i \right) \alpha_i \right) + \nabla \cdot \tau_i \vec{U} \left(\frac{\partial \alpha_i}{\partial t} \right)^* = 0, \ \vec{W}_i = \tau \left(\vec{U} \cdot \nabla \vec{U} + \frac{1}{\rho_i} \nabla p - \vec{g} - \frac{1}{\rho_i} \vec{F}_i \right)$$

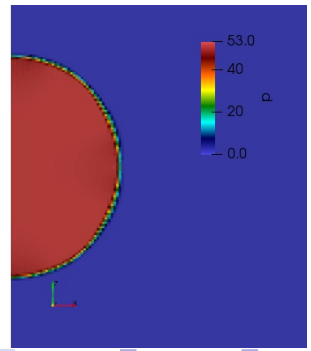
Mixture momentum balance

$$\frac{\partial \rho \vec{U}}{\partial t} + \nabla \cdot \left(\sum_i \rho_i (\vec{U} - \vec{W}_i + \vec{U}\tau_i \left(\frac{\partial \alpha_i}{\partial t} \right)^*) \otimes \vec{U} \right) - \nabla \cdot \vec{U} \otimes \sum_i \alpha_i \rho_i \vec{W}_i =$$
$$- \nabla p (1 + \sum_i \tau_i \left(\frac{\partial \alpha_i}{\partial t} \right)^*) + \sum_i \vec{F}_i (1 + \tau_i \left(\frac{\partial \alpha_i}{\partial t} \right)^*)$$

interQHDFoam: droplet equilibrium

The equilibrium of spherical droplet in zero gravity field is studied. Surface tension force approximation is tested.

- liquid density is $953kg/m^3$
- gas density is $1.204kg/m^3$
- liquid kinematic viscosity is $2^{-5}m^2/s$
- gas kinematic viscosity is $1.58^{-5}m^2/s$
- liquid τ $is5 \times 10^{-8}s$
- gas τ is $5 \times 10^{-8}s$
- $Deltat = \frac{1}{2}\tau = 2.5 \times 10^{-8}s$
- surface tension coefficient is $0.0254Pa/m$
- mesh resolution is 300 cells per diameter
- Radius is $0.9595 \times 10^{-3}m$
- pressure drop is $\Delta P = \frac{2\sigma}{R} = 53Pa$
- gravity vector is $\vec{g} = (0, 0, 0)^T$
- calculated pressure drop: $52.1Pa$
- maximum parasitic current magnitude: $< 0.02m/s$
- 3D simulation can be viewed on YouTube `https://youtu.be/fqqSXh5t_38`
- pressure field distribution:

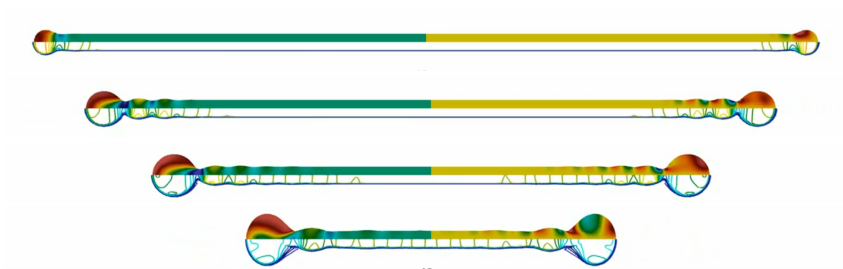


interQHDFoam: filament collapse

Long axisymmetric filament with rounded ends starts to collapse under the surface tension force (See "Drop Formation from axi-symmetric fluid jets", PhD Thesis by Theo Driessen).

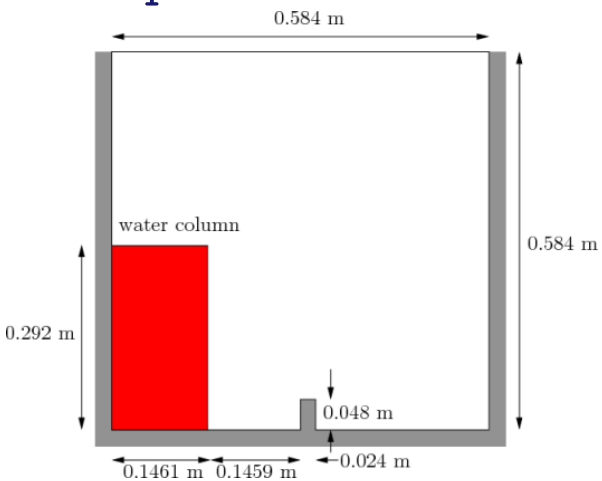
- liquid density is 1000
- gas density is 10
- liquid kinematic viscosity is 0.04
- gas kinematic viscosity is 0.04
- liquid is τ 0.001
- gas is τ 0.001
- surface tension coefficient is 1000
- mesh resolution is 20 cells per diameter
- Radius is 1
- Length is 96
- Tips radius is 1
- gravity vector is $\vec{g} = (0, 0, 0)^T$

- 3D simulation can be viewed on YouTube `https://youtu.be/8bYMBWOlkTw`



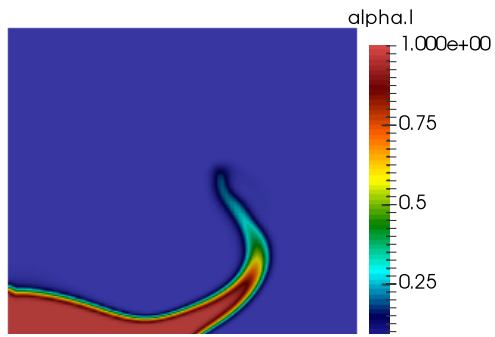
interQHDFoam: damBreak

Case demonstrates water column collapse. See:
`https://cfd.direct/openfoam/user-guide/v6-dambreak/`

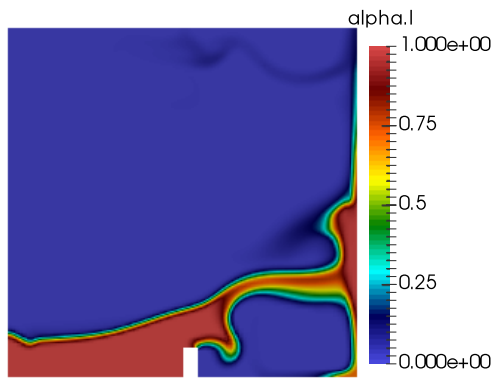


- liquid density $1000kg/m^3$
- gas density $1kg/m^3$
- liquid kinematic viscosity $10^{-6}m^2/s$
- gas kinematic viscosity $10^{-5}m^2/s$
- liquid τ $0.25 \times 10^{-4}s$
- gas τ $0.5 \times 10^{-4}s$
- surface tension coefficient $0Pa/m$

- gravity vector $\vec{g} = (0, -9.81, 0)^T$
- $\Delta t = 0.001s$
- Phase fraction:



Time $= 0.25s$



Time $= 0.5s$

Increasing time-step value

- Sometime QGD algorithms require relatively small time steps (corresponding to $Co \approx 0.01$ or less) to integrate advection equations with low diffuson
- There are several tricks to increase time step possibly at the cost of accuracy and simplicity of algorithm:
  1. the first trick consists in using of solution with decreased Re as initial distribution – the Re number could be later increased, but the initial smoothing should be enough to stabilize simulation;
  2. use MULES schemes for convection part of the equation (see mulesQHDFoam);
  3. use standard OpenFOAM flux limiters – in fact, limiters will work only for regions with abrupt change of properties, in smooth regions the algorithm should switch to central differencing with $\tau-$terms, for example for "interQHDFoam" best settings are:

```
div((nuEff*dev2(T(grad(U))))) Gauss linear;
div(phi,alpha1) Gauss vanLeer;
div(phir,alphar) Gauss linear;
div(rhoPhi,U) Gauss linearUpwind grad(U);
```

Wiki at GitHub

- The "QGDsolver" framework will continue to evolve
- The latest information about recent publications, tutorials and best practices will be located at the project Wiki:
  `https://github.com/unicfdlab/QGDsolver/wiki`

Issues at GitHub

- In case of bugs or requests for new features, please add issues at the project Issues:
  `https://github.com/unicfdlab/QGDsolver/issues`

Threads at cfd-online

- Announcement thread at cfd-online: `https://www.cfd-online.com/Forums/openfoam-news-announcements-other/227336-qgdsolver-openfoam-computational-framework-fluid-flow.html`
- To have a talk with developers, please use dedicated threads at cfd-online:
  1. Go to "OpenFOAM community contributions section": `https://www.cfd-online.com/Forums/openfoam-community-contributions/`
  2. Look for appropriate thread with title starting "[QGDsolver]"
  3. If the thread is not found, create your own with title prefix "[QGDsolver]"
  4. Ask questions

Articles

1. Kraposhin M.V., Smirnova E.V., Elizarova T.G., Istomina M.A. (2018) Development of a new OpenFOAM solver using regularized gas dynamic equations. Computers & Fluids, Volume 166, 30 April 2018, Pages 163–175. DOI: https://doi.org/10.1016/j.compfluid.2018.02.010

Books

1. Boris N. Chetverushkin. Kinetic Schemes and Quasi-Gas Dynamic System of Equations. CIMNE Barcelona, Spain, 2008 ISBN 978-84-96736-46-7
2. Yurii V. Sheretov. Regularized Hydrodynamic Equations (in Russian). Tver State University, 2016. ISBN: 978-5-7609-1076-9
3. Yurii V. Sheretov, Dynamics of Continuous Media with Spatial and Temporal Averagin (Regulyarn. Khaotich. Dinamika, Moscow, Izhevsk, 2009) [in Russian]
4. Tatiana G. Elizarova. Quasi-Gas Dynamic Equations. Springer, 2009
5. Tatiana G. Elizarova. Квазигазодинамические уравнения и методы расчета вязких течений. `http://elizarova.imamod.ru/_media/course-book.pdf`

QGD/QHD Theory Web-resources

- ResearchGate profile of Prof. Yu. V. Sheretov: `https://www.researchgate.net/profile/Yurii_Sheretov`
- ResearchGate profile of Prof. T.G. Elizarova: `https://www.researchgate.net/profile/Tg_Elizarova`
- Personal web page of Prof. T.G. Elizarova: `http://elizarova.imamod.ru/english-main-page.html`
- ResearchGate profile of Prof. A.A. Zlotnik: `https://www.researchgate.net/profile/Alexander_Zlotnik2`

Thank you for attention

And don't forget to follow `https://github.com/unicfdlab/` if you have found this project useful