XinFOAM

Shaohui Chen

January 6, 2020

Contents

Part I

Theory background

1 Theory

1.1 Navier Stokes Equations

The Navier Stokes describe the motion of viscous fluid substances. These balance equations are derived by applying the mass conservation and Newton's second law (momentum conservation principles) to a control volume. Additionally, energy or other interested fields.

1.1.1 Mass Conservation

The mass conservation in a 2D controlled volumes (CV)is readily derived as:

[Rate of mass change in CV] = [Rate of mass flow into CV] - [Rate of mass flow out of CV]

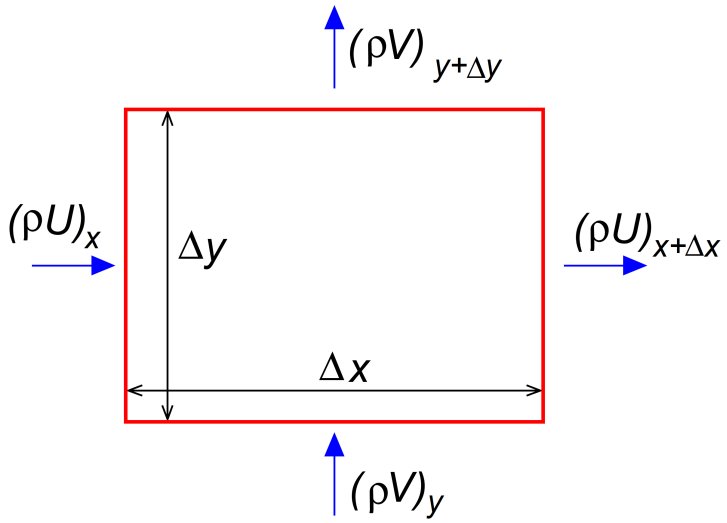


Figure 1: The conservation of mass in a controlled volume

$$\frac{\partial(\rho\Delta x\Delta y)}{\partial t} = (\rho U)_x\Delta y + (\rho V)_y\Delta x - [(\rho U)_{x+\Delta x}\Delta y + (\rho V)_{y+\Delta y}\Delta x] \tag{1}$$

Division with $\Delta x\Delta y$ and rearrange the equation leads to:

$$\frac{\partial\rho}{\partial t} = \frac{(\rho U)_x - (\rho U)_{x+\Delta x}}{\Delta x} + \frac{\Delta y - (\rho V)_{y+\Delta y}}{\Delta y} \tag{2}$$

At the limit $\Delta t \to 0$, control volume shrink to infinitesimally small, using Taylor expansion:

$$(\rho U)_{x+\Delta x} = (\rho U)_x + \Delta x \frac{\partial (\rho U)_x}{\partial x}$$

. Devision with $\Delta x \Delta y$ and rearrange the equation leads to:

$$\frac{\partial \rho}{\partial t} + \frac{\partial (\rho U)}{\partial x} + \frac{\partial (\rho V)}{\partial y} + \frac{\partial (\rho W)}{\partial z} = 0 \text{ or } \frac{\partial \rho}{\partial t} + \frac{\partial (\rho u_i)}{\partial x_i} = 0 \tag{3}$$

1.1.2 Momentum Conservation

The momentum conservation principle is:

[Rate of momentum change in CV] =

[Rate of momentum flow into CV] - [Rate of momentum flow out of CV]

+ [Force acting on CV faces] + [Body forces within CV]

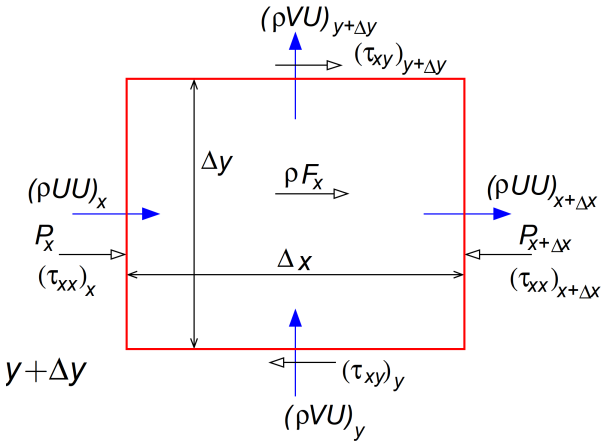


Figure 2: momentum on a CV

In x direction the momentum conservation is

$$\underbrace{\Delta x \Delta y \left[\partial (\rho U) / \partial t\right]}_{\text{Momentum change Rate}} = \underbrace{\left[((\rho UU)_x - \rho UU)_{x+\Delta x}\right] \Delta y + \left[(\rho VV)_y - (\rho VV)_{y+\Delta y}\right] \Delta x}_{\text{Momentum net flow}}$$
$$+ \underbrace{\left[(p + \tau_{xx})_x - (p + \tau_{xx})_{x+\Delta x})\right] \Delta y + \left[\tau_{xy})_y - ((\tau_{xy})_{y+\Delta y})\right] \Delta x}_{\text{Net surface force}}$$
$$+ \underbrace{\rho F_x \Delta x \Delta y}_{\text{Body force}}$$

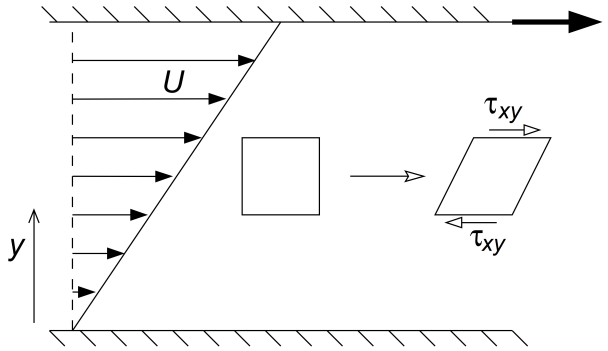
Divide by $\Delta x \Delta y$ and at limit $\Delta x \to 0$ and $\Delta y \to 0$. Expand it to 3-D.

$$\frac{\partial(\rho U)}{\partial t} + \frac{\partial(\rho U^2)}{\partial x} + \frac{\partial(\rho UV)}{\partial y} + \frac{\partial(\rho UW)}{\partial z} = -\frac{\partial p}{\partial x} - \frac{\partial \tau_{xx}}{\partial x} - \frac{\partial \tau_{xy}}{\partial y} - \frac{\partial \tau_{xz}}{\partial z} + \rho F_x \tag{4}$$

Similarly applies to y and z directions. And put it in index form:

$$\frac{\partial(\rho u_i)}{\partial t} + \frac{\partial(\rho u_i u_j)}{\partial x_j} = -\frac{\partial p}{\partial x_i} - \frac{\partial \tau_{ij}}{\partial x_j} + \rho F_i \tag{5}$$

The shear stress in the above equation is also related to the velocity. Consider a flat plate on the top surface moves at constant velocity U while the bottom wall is fixed.



The viscous shear stress is related to the velocity gradient along the perpendicular direction. Fluids follow this behaviour are referred as Newtonian fluids.

$$\tau_{xy} = -\mu \frac{\partial U}{\partial y} \tag{6}$$

For Newtonian fluids these general stress-strain relations can be expressed as the viscous stresses being linearly related to the strain rates, with the constant of proportionality being the viscosity μ.

$$\tau_{xx} = -2\mu \frac{\partial U}{\partial x} \quad \tau_{yy} = -2\mu \frac{\partial V}{\partial y} \quad \tau_{xy} = -\mu(\frac{\partial U}{\partial y} + \frac{\partial V}{\partial x}) \tag{7}$$

Substitute this relation to the 4.

$$\begin{aligned} \frac{\partial(\rho U)}{\partial t} + \frac{\partial(\rho U^2)}{\partial x} + \frac{\partial(\rho UV)}{\partial y} + \frac{\partial(\rho UW)}{\partial z} &= -\frac{\partial p}{\partial x} + \rho F_x \\ + 2\frac{\partial}{\partial x}\left[\mu \frac{\partial U}{\partial x}\right] + \frac{\partial}{\partial y}\left[\mu\left(\frac{\partial U}{\partial y} + \frac{\partial V}{\partial x}\right)\right] &+ \frac{\partial}{\partial z}\left[\mu\left(\frac{\partial U}{\partial z} + \frac{\partial W}{\partial x}\right)\right] \end{aligned}$$

Similar equations exist for y, z directions and put them in index form:

$$\frac{\partial \rho u_i}{\partial t} + \frac{\partial \rho u_i u_j}{\partial x_j} = -\frac{\partial p}{\partial x_i} + \frac{\partial}{\partial x_j}\left(\mu \frac{\partial u_i}{\partial x_j}\right) + \rho F_i \tag{8}$$

Combine the mass and momentum conservation equations together are the Navier-Stokes equations.

$$\begin{cases} \dfrac{\partial \rho}{\partial t} + \dfrac{\partial (\rho u_i)}{\partial x_i} = 0 \\ \dfrac{\partial \rho u_i}{\partial t} + \dfrac{\partial \rho u_i u_j}{\partial x_j} = -\dfrac{\partial p}{\partial x_i} + \dfrac{\partial}{\partial x_j}\left(\mu \dfrac{\partial u_i}{\partial x_j}\right) + \rho F_i \end{cases} \tag{9}$$

Instead of the above simplification, other way to proceed is decomposing the stress to normal and shear parts:

$$\boldsymbol{\tau} = \zeta(\nabla \cdot \mathbf{u})\mathbf{I} + \mu\left(\nabla \mathbf{u} + (\nabla \mathbf{u})^{\mathrm{T}} - \tfrac{2}{3}(\nabla \cdot \mathbf{u})\mathbf{I}\right) \tag{10}$$

For a incompressible case, the density ρ is a constant. The mass continuity equation can be simplified to:

$$\frac{\partial u_i}{\partial x_i} = 0 \tag{11}$$

Further, if the viscosity is a constant, the diffusion can be simplified by taking μ outside the derivatives and combine the mass conservation condition. This form is quite commonly seen in texts.

$$\frac{\partial u_i}{\partial t} + u_i \frac{\partial u_j}{\partial x_j} = -\frac{1}{\rho}\frac{\partial p}{\partial x_i} + \mu\left(\frac{\partial^2 u_i}{\partial x_i x_j}\right) + \rho F_i \tag{12}$$

The first term usually is referred as the time derivative term, the second term is the convection term, the laplacian term of the velocity is the diffusion term. The body force neglected.

$$\begin{cases} \dfrac{\partial \mathbf{u}}{\partial t} + (\mathbf{u} \cdot \nabla)\,\mathbf{u} = -\dfrac{1}{\rho}\nabla p + \nu \nabla^2 \mathbf{u} \\ \nabla \cdot \mathbf{u} = 0 \end{cases} \tag{13}$$

For incompressible case, the viscosity usually is replaced by kinematic viscosity as the above equations indicated.

$$\nu = \frac{\mu}{\rho}$$

The components of viscous stress tensor τ_{ij} in Navier-Stokes equations is defined as :

$$\tau_{ij} = 2\mu S_{ij} + \lambda \frac{\partial u_i}{\partial x_i}\delta_{ij} \tag{14}$$

For the rotating frame reference introduced some interesting pseudo-forces into the equations through the material derivative term. Consider a stationary inertial frame of reference K, and a non-inertial frame of reference K', which is translating with velocity $\mathbf{U}(t)$ and rotating with angular velocity $\boldsymbol{\Omega}(t)$ with respect to the stationary frame. The Navier–Stokes equation observed from the non-inertial frame then becomes

$$\rho \frac{D\mathbf{u}}{Dt} = -\nabla \bar{p} + \mu \nabla^2 \mathbf{u} + \tfrac{1}{3}\mu \nabla(\nabla \cdot \mathbf{u}) + \rho \mathbf{g} - \rho\left(2\boldsymbol{\Omega} \times \mathbf{u} + \boldsymbol{\Omega} \times (\boldsymbol{\Omega} \times \mathbf{x}) + \frac{d\mathbf{U}}{dt} + \frac{d\boldsymbol{\Omega}}{dt} \times \mathbf{x}\right) \tag{15}$$

Here **x**, **U** are measured in the non-inertial frame. The first term in the parenthesis represents Coriolis acceleration, the second term is due to centrifugal acceleration, the third is due to the linear acceleration of K' with respect to K and the fourth term is due to the angular acceleration of K' with respect to K.

1.1.3 Energy Conservation

First law of thermodynamics, the rate of change of energy (left hand side) is equal to the sum of rate of added heat (negative part of the right-hand side) and the rate of work done (positive part of right-hand side).

$$\rho \frac{D}{Dt}\left(e + \frac{1}{2}u_i u_i\right) = \rho F_i u_i - \frac{\partial}{\partial x_i}(p u_i) + \frac{\partial}{\partial x_i}(\tau_{ij} u_j) - \frac{\partial q}{\partial x_i} \tag{16}$$

The internal energy change rate, equals the work rate contributed by the body force, the pressure surface work, viscous surface stress, heat flux.

The heat flux need to be related to the temperature gradients with Fouriers law.

$$q_i = -\kappa \frac{\partial T}{\partial x_i} \tag{17}$$

where $\kappa = \kappa(T)$ is the thermal conductivity. This allows us to write the thermal energy equation as Alternative form of the thermal energy equation can be derived using the definition of the enthalpy

$$h = e + \frac{p}{\rho} \tag{18}$$

[Rate of energy change in CV] = [Rate of net energy change in CV] +

[Work done by surface force acting on CV faces] + [Work done by sbody forces within CV]

$$\begin{aligned}
\frac{\partial}{\partial t}\left[\rho\left(h + \frac{u_i u_i}{2}\right)\right] + \nabla \cdot \frac{\partial}{\partial t}\left[\rho\left(e + \frac{u_i u_i}{2}\right)\right] & \\
&= \rho\dot{q} + \frac{\partial}{\partial x}\left(k\frac{\partial T}{\partial x}\right) + \frac{\partial}{\partial y}\left(k\frac{\partial T}{\partial y}\right) + \frac{\partial}{\partial z}\left(k\frac{\partial T}{\partial z}\right) \\
&\quad - \frac{\partial(u_x p)}{\partial x} - \frac{\partial(u_x p)}{\partial y} - \frac{\partial(u_z p)}{\partial z} \\
&\quad + \frac{\partial(u_x \tau_{xx})}{\partial x} + \frac{\partial(u_x \tau_{yx})}{\partial y} + \frac{\partial(u_x \tau_{zx})}{\partial z} \\
&\quad + \frac{\partial(u_y \tau_{xy})}{\partial x} + \frac{\partial(u_y \tau_{yy})}{\partial y} + \frac{\partial(u_y \tau_{zy})}{\partial z} \\
&\quad + \frac{\partial(u_z \tau_{xz})}{\partial x} + \frac{\partial(u_z \tau_{yz})}{\partial y} + \frac{\partial(u_z \tau_{zz})}{\partial z} \\
&\quad + \rho F_i u_i
\end{aligned}$$

1.1.4 Transport equations

The transport equation describes how a scalar quantity is transported in a space. Usually, it is applied to the transport of a scalar field (e.g. chemical concentration, material properties or temperature) inside an incompressible flow. From the mathematical point of view, the transport equation is also called the convection-diffusion equation, which is a first order PDE (partial differential equation). The convection-diffusion equation is the basis for the most common transportation models. The transport equation (or convection-diffusion equation) can be seen as the generalization of the continuity equation. While the continuity equation (extensively described in the article about incompressible flow) usually describes the conservation of mass, the convection-diffusion equation describes the continuity/conservation of any scalar field in any space. Let's consider an infinitesimal portion of space Ω and its boundaries Γ. A conservation field around the volume should fulfil the following equations.

$$\frac{\partial}{\partial t}\int_{\Omega}\phi d\Omega = -\int_{\Gamma} F d\Gamma + \int_{\Omega} Q d\Omega \tag{19}$$

A continuity equation (or conservation law) is an integral relation stating that the rate of change of some integrated property ϕ defined over a control volume Ω must be equal to what amount is lost or gained through the boundaries Γ of the volume plus what is created or consumed by sources and sinks inside the volume. The numerical integration is then easily done by evaluate the volume integral and the surface integral at the finite cell center. The average value of the cell is stored at the centroid of the cell, the volume of the cell $\times$ the average value is then the integral. The surface integral follows the same way, surface area $\times$ the average surface value.

A general form of this is: Time derivative $(d\phi/dt)$ + Convection terms $(du_i\phi/dx_i)$ = Diffusion terms $(d\phi^2/dx_i)$ + Source terms(S_i). This can be u_i, T. Later the transport equation for turbulence related quantities, chemical concentration....

This is expressed by the following integral continuity equation:

$$\frac{\partial \phi}{\partial t} + \frac{\partial (u_i \phi)}{\partial x_i} = \frac{\partial}{\partial x_i}(\frac{\partial \phi}{\partial x_i}) + S_i \tag{20}$$

where u_i is the flow velocity of the fluid and s represents the sources and sinks in the flow, taking the sinks as positive. These equations can be directly solved by numerical method. This way is called DNS, direct numerical simulation. It is very accurate. However, since the turbulence is usually at very small scale, not practical for engineering application.

1.2 Turbulence

Swirls and circulations, randomness makes it very hard to predict.

The average based method is the most commonly used. The Reynolds-Averaged Navier-Stokes (RANS) technique. Every parameter u is divided to mean part $\bar{u}$ and fluctuating part u', as $u = \bar{u}+u'$ All other parameters p, T can be divided like this to mean and fluctuating. Then Time averaging appropriate for stationary

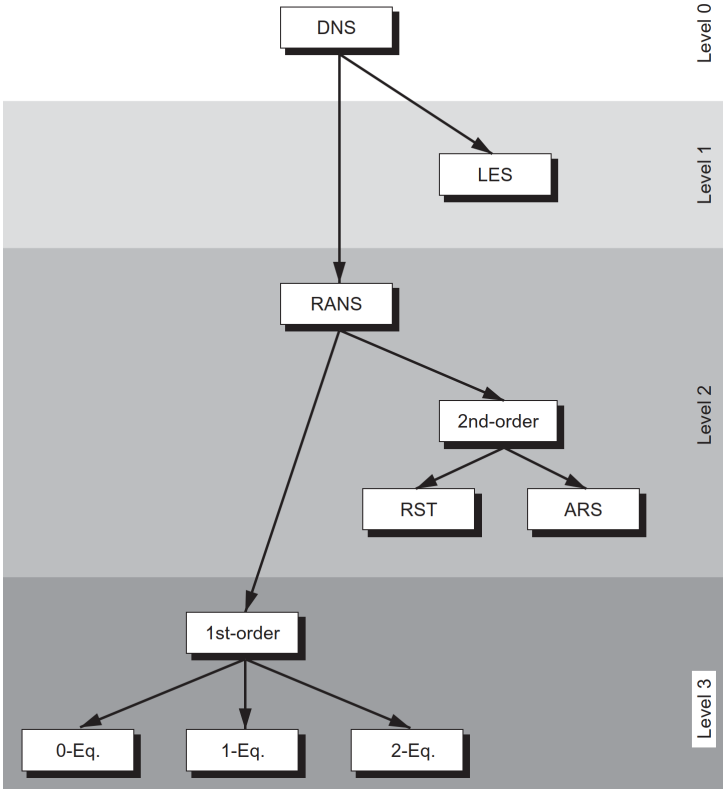


Figure 3: The hierarchy of turbulence models. DNS: direct numerical simulation, LES: large-eddy simulation, ARS: algebraic Reynolds-stress models, RANS: Reynolds-averaged Navier-Stokes equations, 1st-order, first-order closure, RST: Reynolds.stress transport models

turbulence (statistically steady turbulence), Spatial averaging appropriate for homogeneous turbulence and Ensemble averaging—appropriate for general turbulence.

For all three approaches, the average of the fluctuating part is zero, that is, $\overline{v_i'} = 0$. However, it can be easily seen that $\overline{v_i' v_i'} \neq 0$. The same is true for $\overline{v_i' v_j'} \neq 0$, if both turbulent velocity components are correlated.

The model is coming from the averaged Reynolds stress tensor, $\bar{u_l' u_j'}$ is a symmetric tensor and contains 6 new unknown variables without adding new equations. How to solve the unconstrained equations is the main different between the models. The mostly used $k-\epsilon$ and $k-\omega$ models use the Boussinesq assumption to model the Reynolds stress tensor. The boussinesq assumption relates the Reynolds stress tensor to the velocity gradients and turbulent viscosity as the following.

There are many variants such as one equation model and two equations models. Of the two equations model variants of the $k-\epsilon$ and the $k-\omega$ models are among others the most used. The k is stand for turbulent kinetic energy, ϵ is the turbulent dissipation, ω is the specific dissipation. The RANS models can be divides into different groups: Algebraic models; One equation model; Two equations models; Reynolds stress models.

$$\begin{cases} \frac{\partial \overline{u}_i}{\partial t} + \overline{u}_j \frac{\partial \overline{u}_i}{\partial x_j} = -\frac{1}{\rho}\frac{\partial \overline{p}}{\partial x_i} + \mu \nabla^2 \overline{u}_i - \frac{\partial}{\partial x_j}(\overline{u_i' u_j'}) \\ \frac{\partial \overline{u}_i}{\partial x_i} = 0 \end{cases} \tag{21}$$

where the $u_i = \bar{u_i}$ is the average of the continuity equation also has been added. We find the Reynolds average equations

$$\overline{u_l' u_j'} = -\nu_T \left(\frac{\partial \bar{u}_i}{\partial x_j} \frac{\partial \bar{u}_j}{\partial x_i}\right) + \frac{2}{3} k \delta_{ij} = -2\nu_t s_{ij} + \frac{2}{3} k \delta_{ij} \tag{22}$$

$$\frac{\partial}{\partial x_j}(\overline{u_i' u_j'}) = \frac{\partial}{\partial x_j} \begin{bmatrix} \overline{(v_1')^2} & \overline{v_1' v_2'} & \overline{v_1' v_3'} \\ \overline{v_2' v_1'} & \overline{(v_2')^2} & \overline{v_2' v_3'} \\ \overline{v_3' v_1'} & \overline{v_3' v_2'} & \overline{(v_3')^2} \end{bmatrix} \tag{23}$$

However, since v_i' and v_j' in the correlations can be interchanged, the Reynolds-stress tensor contains only six independent components. The sum of the normal stresses divided by density defines the turbulent kinetic energy, that is,

$$K = \frac{1}{2}\overline{v_i' v_i'} = \frac{1}{2}\left[\overline{(v_1')^2} + \overline{(v_2')^2} + \overline{(v_3')^2}\right] \tag{24}$$

As we can see here, the fundamental problem of turbulence modelling based on the Reynolds-averaged Navier-Stokes equations is to find six additional relations in order to close the NS.

The

Part II

OpenFOAM case setup

2 Numerical

OpenFOAM applications are designed for use with unstructured meshes, offering up to second order accuracy, predominately using collocated variable arrangements. Most focus on the Finite Volume Method, for which the conservative form of the general scalar transport equation for the property ϕ takes the form:

$$\underbrace{\frac{\partial}{\partial t}(\rho\phi)}_{\text{unsteady}} + \underbrace{\nabla \cdot (\rho\phi\mathbf{u})}_{\text{convection}} = \underbrace{\nabla \cdot (\Gamma\nabla\phi)}_{\text{diffusion}} + \underbrace{S_\phi}_{\text{source}} \tag{25}$$

unsteady:	time dependent
convection	velocity dependent
diffusion	viscosity dependent
source	heat, force...

The finite volume method requires the integration over a 3D-control volume, such that:

$$\int_V \frac{\partial}{\partial t}(\rho\phi)dV + \int_V \nabla \cdot (\rho\phi\mathbf{u})\, dV = \int_V \nabla \cdot (\Gamma\nabla\phi)\, dV + \int_V S_\phi dV \tag{26}$$

or in a more compact way:

$$\boldsymbol{A}\mathbf{x} = \mathbf{b} \tag{27}$$

where

A:	coefficient matrix
x	vector of unknows dependent
b	source vector

The discretisation process employs user selected schemes to build the A matrix and b vector, described in the following sections. Choice of schemes are set in the *fvSchemes* dictionary.

OpenFOAM includes a wide range of solution and scheme controls, specified via dictionary files in the case system sub-directory. These are described by:

Numerical schemes: The treatment of each term in the system of equations is specified in the fvSchemes dictionary. This enables fine-grain control of e.g. temporal, gradient, divergence and interpolation schemes.

Linear equation solvers Solution methods: Case solution parameters are specified in the fvSolution dictionary. These include choice of linear equation solver per field variable, algorithm controls e.g. number of inner and outer iterations and under-relaxation.

Finite volume options: Additional run-time selectable physical modelling and general finite terms are prescribed in the fvOptions dictionary, targeting e.g. acoustics, heat transfer, momentum sources, multi-region coupling, linearised sources/sinks and many more.

2.1 Schemes

The fvSchemes dictionary in the system directory sets the numerical schemes for terms, such as derivatives in equations, that appear in applications being run. This section describes how to specify the schemes in the fvSchemes dictionary.

The terms that must typically be assigned a numerical scheme in fvSchemes range from derivatives, e.g. gradient ∇, and interpolations of values from one set of points to another. The aim in OpenFOAM is to offer an unrestricted choice to the user. For example, while linear interpolation is effective in many cases, OpenFOAM offers complete freedom to choose from a wide selection of interpolation schemes for all interpolation terms.

The derivative terms further exemplify this freedom of choice. The user first has a choice of discretisation practice where standard Gaussian finite volume integration is the common choice. Gaussian integration is based on summing values on cell faces, which must be interpolated from cell centres. The user again has a completely free choice of interpolation scheme, with certain schemes being specifically designed for particular derivative terms, especially the convection divergence ∇ terms.

The set of terms, for which numerical schemes must be specified, are subdivided within the fvSchemes dictionary into the categories listed in Table 6.2. Each keyword in Table 6.2 is the name of a sub-dictionary which contains terms of a particular type, e.g.gradSchemes contains all the gradient derivative terms such as grad(p) (which represents $\nabla(p)$. Further examples can be seen in the extract from an fvSchemes dictionary below:

Keyword	Category of mathematical terms
ddtSchemes	first and second time derivatives
gradSchemes	Gradient
divSchemes	Divergence
laplacianSchemes	lapalcian
interpolationSchemes	point-to-point interpolation of values
snGradSchemes	Component of gradient normal to a cell face

2.1.1 ddtSchemes

OpenFOAM includes a variety of schemes to integrate fields with respect to time. Time schemes define how a property is integrated as a function of time, i.e.

$$\frac{\partial}{\partial t}(\phi) \tag{28}$$

Time schemes properties are input in the *fvSchemes* file under the *ddtSchemes* sub-directory using the syntax. The first time derivative terms are specified in the ddtSchemes sub-dictionary. When specifying a

time scheme it must be noted that an application designed for transient problems will not necessarily run as steady-state and visa versa. For example the solution will not converge if steadyState is specified when running icoFoam, the transient, laminar incompressible flow code; rather, simpleFoam should be used for steady-state, incompressible flow. The discrete scheme for each term can be selected from those listed in following table.

Schemes	Description
steadyState	Do not solve for time derivatives.
Euler	First order, bounded, implicit
localEuler	Local-time step, first order, bounded, implicit
backward	Second order, implicit
CrankNicholson ψ	Second order, bounded, implicit

Depending on the choice of schemes, field values at previous time steps are required, represented in the following as ϕ^0 and ϕ^{00} for the old and older time levels.

Available time schemes include

- Backward time scheme
$$\frac{\partial}{\partial t}(\phi) = \frac{1}{\Delta t}(\frac{3}{1}\phi - 2\phi^0 + \frac{1}{2}\phi^{00}) \tag{29}$$
Using implicit, second order accuracy, transient, boundedness not guaranteed, conditionally stable.

```
ddtSchemes
{
    default         backward;
}
```

- Crank-Nicolson time scheme Second order, Transient, Bounded, when using uniform time steps the schemes equate to:
$$\frac{\partial}{\partial t} = \frac{\phi - \phi^{00}}{2\Delta t} \tag{30}$$
The schemes is specified using:

```
ddtSchemes
{
    default         CrankNicolson <coeff>;
}
```

The coefficient provides a blending between Euler and Crank-Nicolson schemes:

0: Euler 1: Crank-Nicolson A value of 0.9 is a good compromise between accuracy and robustness

- Euler implicit time scheme

Implicit, First order, Transient
$$\frac{\partial}{\partial t} = \frac{\phi - \phi^0}{\Delta t} \tag{31}$$

```
ddtSchemes
{
    default         Euler;
}
```

- Local Euler implicit/explicit time scheme First order Pseudo transient, designed for steady cases Spatially varying, cell-based time scale set by specific Local Time Stepping (LTS) solvers

```
ddtSchemes
{
    default         localEuler;
}
```

- Steady state time scheme Steady state, set temporal derivative contributions to zero,

$$\frac{\partial}{\partial t} = 0 \tag{32}$$

2.1.2 gradSchemes

At their core, spatial schemes rely heavily on *interpolation schemes* to transform cell-bases qualities to cell faces, in combination with **Gauss Theorem** to convert volume integrals to surface integrals. The gradSchemes sub-dictionary contains gradient terms. The discretisation scheme for each term can be selected from those listed in Table.

Discretisation scheme	Description
Gauss ¡interpolationSchemes¿	Second order, Gaussian integration
leastSquares	Second order, least squares
fourth	Fourth order, least squares
cellLimited ¡gradScheme¿	Cell limited version of one of the above schemes
faceLimited ¡gradScheme¿	Face limited version of one of the above schemes

The Gauss keyword specifies the standard finite volume discretisation of Gaussian integration which requires the interpolation of values from cell centres to face centres. Therefore, the Gauss entry must be followed by the choice of interpolation scheme. It would be extremely unusual to select anything other than general interpolation schemes and in most cases the linear scheme is an effective choice, e.g.

Gauss linear;

Limited versions of any of the 3 base gradient schemes — Gauss, leastSquares and fourth — can be selected by preceding the discretisation scheme by cellLimited (or faceLimited), e.g. a cell limited Gauss scheme:

cellLimited Gauss linear 1;

2.1.3 divSchemes

Scheme Numerical behaviour	
linear	Second order, unbounded
skewLinear	Second order, (more) unbounded, skewness correction
cubicCorrected	Fourth order, unbounded
upwind	First order, bounded
linearUpwind	First/second order, bounded
QUICK	First/second order, bounded
TVD schemes	First/second order, bounded
SFCD	Second order, bounded
NVD schemes	First/second order, bounded

- Gradient Schemes
- Divergence schemes
- Laplacian schemes
- Surface-normal gradient schemes

2.2 Wall distance calculation method

Distance to the nearest wall is required, e.g. for a number of turbulence models. Several calculation methods are available:

- Mesh-wave wall distance
- Poisson wall distance

2.3 Solutions

After using different schemes to build the equation systems. The next step is to chose a technic to solve the linear equations, the matrix sytem.

$$x = A^{-1}b \tag{33}$$

where the inverse of the diagonal matrix is simply:

$$A^{-1} = \frac{1}{diag(A)} \tag{34}$$

This is available as the diagonalSolver. More tyically the matrix cannot be inverted easily and the system is solved using iterative methods, as discribed in the following sections.

2.4 Solvers

- Smooth solvers
- Conjugate gradient solvers
- Multigrid solvers

2.5 Solver control

- Under relaxation
- Residuals
- Case termination

Common usage minIter: minimum number of solver iterations maxIter: maximum number of solver iterations nSweeps: number of solver iterations between checks for solver convergence Implementation details Matrix structure Matrix coefficients are stored in upper-triangular order

neighbour cell index always higher than owner cell index across a face when looping over cell faces, the face index increases with increasing cell index for the 1-D case, if cell index 0 is at the boundary, this equates to a monotonic increase in cell numbers, i.e. defines a continuous sweep across the 1-D region used in Gauss-Seidel method

3 boundary condition

Boundary conditions are extremely important. Indeed, they cause the most common errors. OpenFOAM uses similar set-up as other solvers.

- At the inlet (flow inlet) to the computational domain total pressure and total temperature are set. The rest of variables are reconstructed.
- At the outlet the static pressure is set.
- At the rigid walls velocity is set to zero.
- Multiple Reference Frame (MRF) is used for rotational components.
- For steady-state computations the initial conditions have no influence to the results.
- For steady-state computations the initial conditions just help to make the case run.

The main variables to be set are following:

p: static pressure p

U: velocity vector U

T: static temperature

k: turbulent kinetic energy k

ω: specific turbulence dissipation

ϵ specific turbulence dissipation

The initial and boundary conditions for all variables are set in files located in directories named by numbers. Typically directory 0 is recommended to start a simulation from.

Initial conditions are set in parameter internalField putting the values into the cell centres.

At boundaries, initial conditions are set individually by parameter value.

Following table shows recommended model of boundary conditions for computed variables:

boundaty type	p	U	T	k	ω / ϵ
STATOR:					
inlet	totalPressure	pressureDirected InletVelocity	totalTemperature	turbulentIntensity KineticEnergyInlet	fixedValue
volute_wall	zeroGradient	fixexValue	compressible:: turbulentTemperature CoupledBaffeMixed	compressible:: kqRWallFunction	compressible:: omegaWallFunction
ring*	mixingPlane	inletOutlet	inletOutlet	inletOutlet	inletOutlet
ROTOR					
outlet	fixedMeanValue	inletOutlet	inletOutlet	inletOutlet	inletOutlet
outlet wall	zeroGradient	fixedValue	compressible:: turbulentTemperature CoupledBaffeMixed	compressible:: kqRWallFunction	compressible:: omegaWallFunction
wheel wall	zeroGradient	fixexValue	zeroGradient	compressible:: kqRWallFunction	compressible:: omegaWallFunction
ring*	zeroGradient	mixingPlaneVelocity	mixingPlane	mixingPlane	mixingPlane
SOLID:					
rotor inner wall	x	x	compressible:: turbulentTemperature CoupledBaffeMixed	x	x
stator inner wall	x	x	compressible:: turbulentTemperature CoupledBaffeMixed	x	x
outer wall	x	x	zeroGradient	x	x

Table 1: Boundary condition recommondation*

*: taken from: https://www.cfdsupport.com/Turbomachinery-CFD-manual/node283.html#13201

The shortcuts from the above table have following meaning:

- tP - totalPressure constant, e.g. 250 000 Pa, gamma = 1.2853 [-] is specific heat ratio
- pDIV - pressureDirectedInletVelocity, velocity is computed from difference between total and static pressure, inletDirection is velocity vector to be specified
- tT - totalTemperature, constant, e.g. 1050 K, gamma= 1.2853 [-] is specific heat ratio
- tIKEI - turbulentIntensityKineticEnergyInlet, intensity = 0.02, corresponds to turbulence intensity 2
- fV - fixedValue, e.g. velocity at the wall (0 0 0), or omega at inlet
- fMV - fixedMeanValue, is the same as fixed value, e.g. for pressure, but certain freedom is allowed to keep the variable average equal to meanValue
- zG - zeroGradient, the flux of the variable is zero in direction perpendicular to the surface
- TWF - compressible::turbulentTemperatureCoupledBaffleMixed special boundary condition for temperature enabling heat transfer from other regions
- kWF - compressible::kqRWallFunction is a standard wall function for k for compressible flow
- oWF - compressible::omegaWallFunction is a standard wall function for omega for compressible flow
- mPV - mixingPlaneVelocity, averaged velocity is mapped from neighbour patch
- mP - mixingPlane, averaged variable is mapped from neighbour patch
- iO - inletOutlet is by default zeroGradient, but changes to fixedValue when velocity vector direction points inside the computational domain (backward flow)

Another recommended boundary conditions for each vairable (depending on the selected turbulence model) at each boundary type. Note the variations for Wall BCs for a High Reynolds model(y+ 30-100), and a Low Reynolds model(y+ 1).

	inlet	outlet	farfield	stationary walls	moving walls	rotating walls
p	**zeroGradient**	**fixedValue 0**	**outletInlet 0**	**zeroGradient**	**zeroGradient**	**zeroGradient**
U	**surfaceNormalFV** negative value for entry velocity	**zeroGradient** **inletOutlet**	**inletOutlet** free stream velocity	**fixedValue 0**	**fixedValue** velocity of the wall	**rotatingWallVelocity** rotational velocity of the wall
k	**fixedValue** k = $(3/2) * (UI)^2$ U:free stream velocity I: Turbulent intensity	**zeroGradient** **inletOutlet**	**inletOutlet** k = $(3/2) * (UI)^2$ U:free stream velocity I: Turbulent intensity	**kqRWallFunction** (for y+ 30 to 100) **fixedValue 0** (for y+ 1)	**kqRWallFunction** (for y+ 30 to 100) **fixedValue 0** (for y+ 1)	**kqRWallFunction** (for y+ 30 to 100) **fixedValue 0** (for y+ 1)
epsilon	**fixedValue** $\epsilon = 0.09 * k * \omega$	**zeroGradient** **inletOutlet**	**inletOutlet** $\epsilon = 0.09 * k * \omega$	**epsilonWallFunction** (for y+ 30 to 100) fixedValue 10^{-12} (for y+ 1)	**epsilonWallFunction** (for y+ 30 to 100) fixedValue 10^{-12} (for y+ 1)	**epsilonWallFunction** (for y+ 30 to 100) fixedValue 10^{-12} (for y+ 1)
omega	**fixedValue** $\omega = \rho * \frac{k}{\mu} * (\frac{\mu_t}{\mu})^{-1}$ ρ:Density k:Turbulence kinetic energy μ: viscosity μ_t:turbulent viscosity	**zeroGradient** **inletOutlet**	**inletOutlet** $\omega = \rho * \frac{k}{\mu} * (\frac{\mu_t}{\mu})^{-1}$ ρ:Density k:Turbulence kinetic energy μ: viscosity μ_t:turbulent viscosity	**omegaWallFunction**	**omegaWallFunction**	**omegaWallFunction**
nutTilda	**fixedValue** nuTilda = $\sqrt{3/2} * U * I * L$ U:free stream velocity I:turbulent intensity L:length scale	**zeroGradient** **inletOutlet**	**inletOutlet** nuTilda = $\sqrt{3/2} * U * I * L$ U:free stream velocity I:turbulent intensity L:length scale	**zeroGradient** (for y+ 30 to 100) **fixexValue 0** (for y+ 1)	**zeroGradient** (for y+ 30 to 100) **fixexValue 0** (for y+ 1)	**zeroGradient** (for y+ 30 to 100) **fixexValue 0** (for y+ 1)
nut	**calculated**	**calculated**	**calculated**	**nutkWallFunction** (for y+ 30 to 100) **nutUSpalding** **WallFunction** (for y+ 1)	**nutkWallFunction** (for y+ 30 to 100) **nutUSpalding** **WallFunction** (for y+ 1)	**nutkWallFunction** (for y+ 30 to 100) **nutUSpalding** **WallFunction** (for y+ 1)

Table 2: Boundary condition recommondation*

*: taken from: ANSA for CFD Brief User's Guide. v19.1.2

solver	transient	compressible	turbulence	heat transfer	buoyancy	combustion	multi phase	particles	dynamicMesh	multi region	fvOptions
boundaryFoam											
buoyantPimpleFoam	√	√	√	√	√						√
buoyantSImpleFoam			√	√	√	√					√
chemFoam	√			√		√					
chtMultiRegionFoam	√	√	√	√	√					√	√
coldEngineFoam	√	√	√	√		√			√		√
engineFoam	√	√	√	√		√			√		√
fireFoam	√	√	√	√	√	√				√	√
icoFoam	√										
interFoam	√		√				√		√		√
laplacianFoam	√										√
pimpleFoam	√		√						√		√
pisoFoam	√		√								√
potensialFoam											
reactingFoam	√	√	√	√		√					√
reactingParcelFoam	√	√	√	√	√	√		√			√
rhoCentralFoam	√	√	√	√							
rhoPimpleFoam	√	√	√	√					√		√
rhoSimpleFoam		√	√	√							√
scalarTransportFoam	√										
simpleFoam											√
spratFoam	√	√	√	√	√	√		√			√
XiFoam	√	√	√	√		√					√

Table 3: application solver capability matrix*

*: taken from: https://www.openfoam.com/documentation/guides/latest/doc/openfoam-guide-applications-solvers.html#sec-applications-solve

4 Conclusion

"I always thought something was fundamentally wrong with the universe" [1]

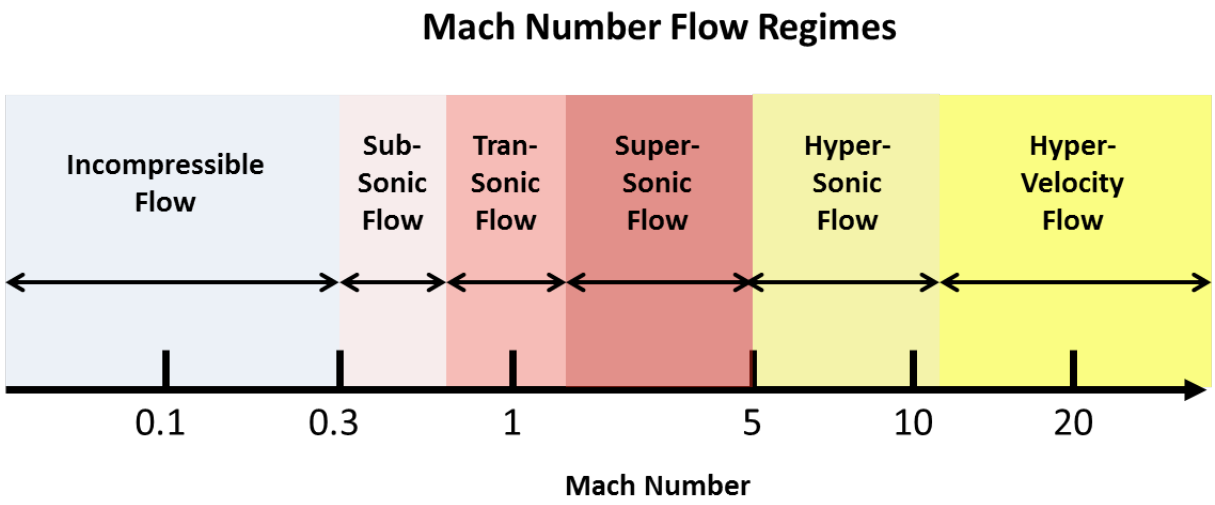


Figure 4: compressible or incompressible dependence of Mach number.

A Appendix

Some important mathematics refreshments:

A.1 Gradient

∇, For 1-dimensional domain, it is a standard derivative defined in calculus. For multi-dimensional domain, it may denote gradient, divergence, curl field. If ∇ is applied to a scalar function, $f(x, y, z)$, it is the direction vector of the scalar function, stand for the local steepest slope.

$$grad\, f = \nabla f = \frac{\partial f}{\partial x}\vec{e}_1 + \frac{\partial f}{\partial y}\vec{e}_2 + \frac{\partial f}{\partial z}\vec{e}_3 \quad or \quad \left(\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}, \frac{\partial f}{\partial z}\right) \tag{35}$$

A.2 Divergence

if the ∇ is applied to a vector, it is the dot product of ∇ with the vector $\nabla \cdot \vec{v}$ defined as:

$$Div\, \vec{v} = \nabla \cdot \vec{v} = \frac{\partial v_x}{\partial x} + \frac{\partial v_y}{\partial y} + \frac{\partial v_z}{\partial z} \tag{36}$$

Divergence is a scalar function. It is the intensity of the converge flux toward or repel from a point.

A.3 Curl

$\nabla \times \vec{v}$ is the curl of a vector. It is a measure of the rotation of the vector field at the point.

$$curl\, \vec{v} = \nabla \times \vec{v} = (\frac{\partial \vec{v}_z}{\partial y} - \frac{\partial \vec{v}_y}{\partial z})\vec{e}_x + (\frac{\partial \vec{v}_x}{\partial z} - \frac{\partial \vec{v}_z}{\partial x})\vec{e}_y + (\frac{\partial \vec{v}_y}{\partial x} - \frac{\partial \vec{v}_x}{\partial y})\vec{e}_z \tag{37}$$

The result is refered as vorticity, If a cubic is thrown into a river, it rotates while if it flows, then there is vorticity in the river.

A.4 Laplacian

Δ. It is a measure of the source. If there is no heat source inside a closed smooth surface, the Laplacian of heat flux on the closed surface should be zero. where: $\Delta q = 0$.

$$\Delta f = \nabla \cdot \nabla f = \nabla^2 f = \sum \frac{\partial^2 f}{\partial x_i^2} = \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2} + \frac{\partial^2 f}{\partial z^2} \tag{38}$$

Description the diffusion: if there is no source in closed smooth surface, the Laplacian is zero, such as steady state heat transfer equation.

A.5 Gauss Theorem

Gauss Theorem is also refereed as divergence theorem. It connects the volume integral to the closed surface integral. If $\mathbf{u}$ is a continuously differentiable vector field defined on a neighborhood of V, then we have:

$$\int_V (\nabla \mathbf{u}) dV = \int_S (\mathbf{n} \cdot \mathbf{u}) dS \tag{39}$$

The left side is a volume integral over the volume V, the right side is the surface integral over the boundary of the volume V. The closed manifold ∂V is quite generally the boundary of V oriented by outward-pointing normals, and n is the outward pointing unit normal field of the boundary ∂V. (dS may be used as a shorthand for ndS.) The symbol within the two integrals stresses once more that ∂V is a closed surface. In terms of the intuitive description above, the left-hand side of the equation represents the total of the sources in the volume V, and the right-hand side represents the total flow across the boundary S.

In the physical theory of diffusion, the Laplace operator (via Laplace's equation) arises naturally in the mathematical description of equilibrium. Specifically, if u is the density at equilibrium of some quantity such as a chemical concentration, then the net flux of u through the boundary of any smooth region V is zero, provided there is no source or sink within V:

$$\int_{\partial V} \nabla u \cdot \mathbf{n} dS = 0 \tag{40}$$

where **u** is the outward unit normal to the boundary of V. By divergence theorem.

$$\int_V \text{div}\nabla u dV = \int_{\partial V} \nabla \cdot \mathbf{u} dS = 0 \tag{41}$$

Since this holds for all smooth regions V, it can be shown that this implies

$$\text{div}\nabla u = \Delta u = 0 \tag{42}$$

The left-hand side of this equation is the Laplace operator. The Laplace operator itself has a physical interpretation for non-equilibrium diffusion as the extent to which a point represents a source or sink of chemical concentration, in a sense made precise by the diffusion equation.

Also known as diffusion, the Laplace operator (via Laplace's equation) arises naturally in the mathematical description of equilibrium.Specifically, if u is the density at equilibrium of some quantity such as a chemical concentration, then the net flux of u through the boundary of any smooth region V is zero, provided there is no source or sink within V.

A.6 Material derivative

In continuum mechanics, the material derivative describes the time rate of change of some physical quantity (like heat or momentum) of a material element that is subjected to a space-and-time-dependent macroscopic velocity field. The material derivative can serve as a link between Eulerian and Lagrangian descriptions of continuum deformation.

For example, in fluid dynamics, the velocity field is the flow velocity, and the quantity of interest might be the temperature of the fluid. In which case, the material derivative then describes the temperature change of a certain fluid parcel with time, as it flows along its pathline (trajectory).

$$\frac{D\phi}{Dt} = \frac{\partial \phi}{\partial t} + \frac{\partial(\phi u_i)}{\partial x_i} \tag{43}$$

The first term is the change of ϕ over time, local part. The second term is the change due to the space variation. convection part.

A.7 Reynold transportation theorem

In fact, it is a material derivative to a integral. Enable to evaluate the integral derivation of a time dependent function over a time with varies controlled volume.

$$\frac{D}{Dt}\int_{V(t)} T_{ij} dV = \int_{V(t)} \left[\frac{\partial T_{ij}}{\partial t} + \frac{\partial}{\partial x_k}\left(u_k T_{ij}\right)\right] dV \tag{44}$$

Part III

Visualization

B VTK

The VTK pipeline The key to using VTK is to construct a pipeline. Most applications will go through similar stages when creating their pipeline. In approximately 4 general steps these are:

- Reading data: A visualisation needs data to visualise!
- Filtering and processing data: Once data is read into the pipeline it often needs further processing and manipulation before it can be packaged into an actor.
- Creating actors from that data and assembling them in a scene: Data and geometry are packaged into an actor (don't worry, more on this will follow below).
- Assembling pieces of geometry into a final render: The actors described above are positioned and an interactive 3D environment is rendered.

You can probably tell by now that VTK uses the metaphor of a theatre/film set when describing the rendering environment. The goal of the early stages of a VTK pipeline is to read data including information about the geometry of a model and package it into an actor. An actor is an instance of an object that includes all of the information necessary for rendering (how to do this will be explained below). These actors will be arranged in a scene by you and rendered. The scene created by VTK is is interactive. The user will be able to move around and rotate the objects. It is possible, if programmed into the visualisation, to cut, move, and otherwise manipulate these actors to get at precisely the part of the geometry that is of most interest to the user.

Creating a mapper is an intermediate step in the visualisation pipeline. It takes data and eeds into an actor. Its role is to map data into graphics primitives. For polygon meshes this is more straightforward than other types of data. It also includes making settings about how data is converted to colours (see lookup tables below), or whether to use cell data or node data to colour a model. The actor is concerned with how the model fits into the scene. As a result parameters that can be set from within the actor often concern texturing, lighting, positioning etc. of the object within the scene.

The final steps to complete the visualisation are the following.

```
window = vtk.vtkRenderWindow()
# Sets the pixel width, length of the window.
window.SetSize(500, 500)

interactor = vtk.vtkRenderWindowInteractor()
interactor.SetRenderWindow(window)
```

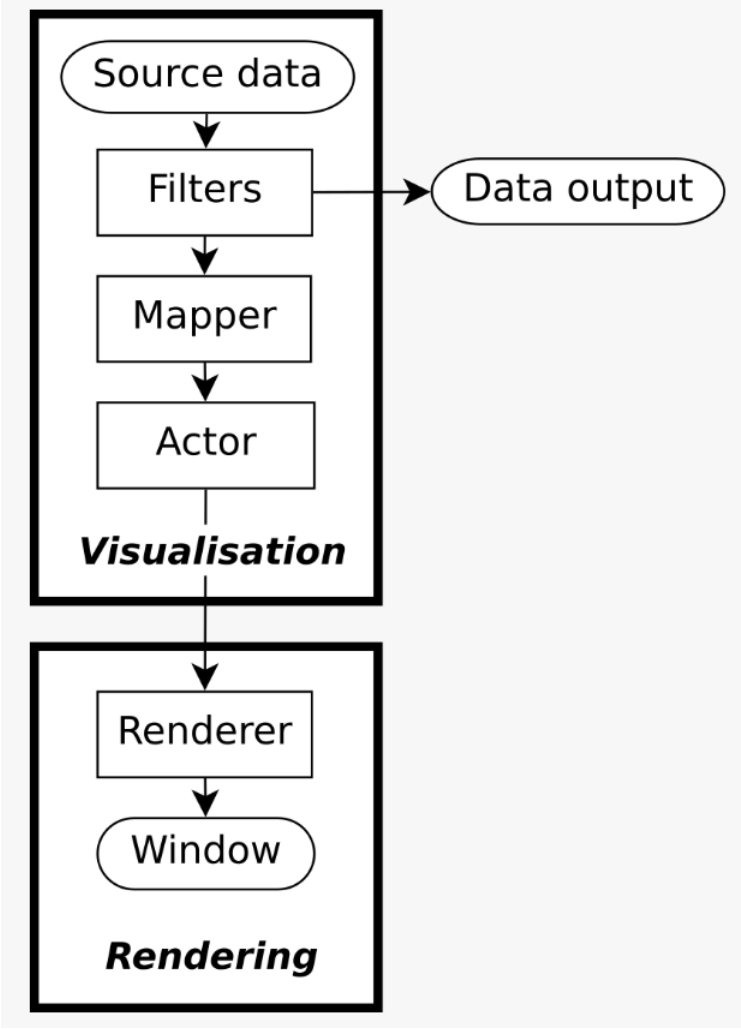


Figure 5: The pipe line of the VTK visualization

```
renderer = vtk.vtkRenderer()
window.AddRenderer(renderer)

renderer.AddActor(actor)
# Setting the background to blue.
renderer.SetBackground(0.1, 0.1, 0.4)

window.Render()
interactor.Start()
```

A lookup table for polygon data is created using lut = vtk.vtkPolyDataMapper(). The lookup table is associated with a model through the mapper object. The method .SetLookupTable(lut) connects the lookup table lut to the mapper. Further, the method .SetUseLookupTableScalarRange(True) needs to be called on the mapper. This causes the mapper to use the same range as that used by the lookup table. Not using this method will cause the mapper to override the range set in the lookup table which is probably not what you want.

References

[1] D. Adams. *The Hitchhiker's Guide to the Galaxy.* San Val, 1995.