Flowsquare 4.0: Theory and Computation

Yuki Minamoto (http://flowsquare.com)

12th December 2013

Flowsquare is a two-dimensional Computational Fluid Dynamics (CFD) software for unsteady, non-reactive, reactive and subsonic/supersonic flows. The aim of this software is to provide a handy CFD environment so that more people can get to know what CFD is like and simulate flows for their educational and/or academic interests. This documentation includes theoretical and numerical aspects of the software, which are basis of the parameters in *grid.txt*. Although flow simulations can be carried out without understanding them, the users are recommended to read this documentation.

1. Non-reactive flows

First, numerical method for non-reactive flow simulation is explained.

1.1. Governing equations

In Flowsquare, the incompressible governing equations are solved. They consist of the continuity equation (mass conservation, and also Einstein notation is used):

$$\frac{\partial \rho}{\partial t} + \frac{\partial}{\partial x_i}(\rho u_i) = 0, \tag{1}$$

and the equation for momentum:

$$\frac{\partial}{\partial t}(\rho u_j) + \frac{\partial}{\partial x_i}(\rho u_i u_j) = -\frac{\partial p}{\partial x_j} + \frac{\partial \tau_{ij}}{\partial x_i} + (\rho - \rho_\infty) g_j, \tag{2}$$

where u_i is the velocity component in *i*-direction (m/s), ρ is the mixture density (kg/m^3), p is the pressure (Pa), g_i is the external force in *i*-direction due to buoyancy (m/s^2). The viscous term τ_{ij} is written as:

$$\tau_{ij} = -\frac{2}{3}\mu \frac{\partial u_k}{\partial x_k}\delta_{ij} + \mu\left(\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i}\right), \tag{3}$$

where μ is the dynamic viscosity (kg/(m s)) and δ_{ij} is the Kronecker delta. Note that the equation is solved assuming a constant density, although Eqs. (1) and (2) are written in conservative form. Equation (2) is solved (integrated in time) in two steps. In the first step, the equation is solved for $(\rho u_j)^*$ without the pressure term as:

$$\frac{\partial}{\partial t}(\rho u_j)^* = -\frac{\partial}{\partial x_i}(\rho u_i u_j) + \frac{\partial \tau_{ij}}{\partial x_i} + (\rho - \rho_\infty) g_j, \tag{4}$$

In the second step (time integration), the pressure term is included so that the mass conservation is taken into account in the second time integration:

$$\frac{\partial}{\partial t}(\rho u_j) = -\frac{\partial p^*}{\partial x_j}. \tag{5}$$

By calculating divergence of Eq. (5), the Poisson's equation for the correction pressure p^* is as:

$$\frac{\partial^2 p}{\partial x_i^2} = \left(\frac{\partial}{\partial x_j}(\rho u_j)^* + \frac{\partial \rho}{\partial t}\cdot \omega_d\right)\cdot\frac{1}{\Delta t}, \tag{6}$$

where Δt is the time step and ω_d, a user defined parameter, is typically unity. Once the corrected pressure field in obtained by solving Eq. (6), the corrected velocity in the next time step is computed from Eq. (5) as:

$$\rho u_j = (\rho u_j)^* - \frac{\partial p^*}{\partial x_j} \Delta t. \qquad (7)$$

The corrected velocity field satisfying the continuity equation (Eq. 1) is obtained. Note that in this numerical method, the continuity equation is not explicitly solved.

2. Reactive flows

In the simulation of reactive flows, Eqs. (1)—(7) are solved to obtain a velocity field. This means the flow field is solved based on the low-Mach number assumption (the flow has to be "slow" compared to the speed of sound). Also, changes of species and internal energy per unit mass need to be considered by solving the transport equations, due to the chemical reactions and heat release. The transport equations for mass fraction species *i*, Y_i and temperature T maybe written under appropriate assumptions as:

$$\frac{\partial(\rho Y_i)}{\partial t} + \frac{\partial}{\partial x_j}(\rho u_j Y_i) = \frac{\partial}{\partial x_j}\left(\rho D_i \frac{\partial Y_i}{\partial x_j}\right) + \omega_i, \qquad (8)$$

$$\frac{\partial(\rho T)}{\partial t} + \frac{\partial}{\partial x_j}(\rho u_j T) = \frac{\partial}{\partial x_j}\left(\frac{\lambda}{c_p}\frac{\partial T}{\partial x_j}\right) + \frac{\omega_T}{c_p}, \qquad (9)$$

where the energy conservation is written in terms of temperature. The molecular diffusivity of species *i*, thermal conductivity and specific heat capacity at constant pressure are respectively denoted by D_i, λ and c_p. Species and heat production rate per unit volume are denoted by ω_i and ω_T. For the simplest reaction system, where the fuel F and oxidiser O react to produce products P as:

$$F + O \rightarrow P, \qquad (10)$$

the total number of equations for Eqs. (8) and (9) is four.

In Flowsquare, in order to reduce the number of equations to be solved, a progress variable and a mixture fraction are introduced. The progress variable is used for reactive flows with premixed mixtures, while the mixture fraction is used for reactive systems with non-premixed mixtures. These two modes of reactions and the reduced governing equation are explained in the next subsections.

2.1 Premixed reacting flows

A chemical reaction is called premixed reaction when a fuel and oxidiser are fully mixed before the reaction take place. For example, premixed reaction (combustion) can be observed in traditional gas stove and heater burners.

One of the important features of this reaction mode is that a reaction front (a flame in combustion) can freely propagate toward unburnt mixture (see Fig. 1a), or mixture can ignite at any locations if there is enough activation energy. This is not the case in non-premixed systems as explained in the next subsection.

For premixed mixture, reaction can be expressed by using reaction progress variables. A progress variable, often denoted by c, is a normalised scalar which shows the extent of reaction progress in premixed reactive systems. Although there are several definitions for progress variables, following definitions are frequently used:

$$c_{Y_{F/O}} = 1 - \frac{Y_{F/O}}{Y_{F/O,u}}, \quad c_{Y_P} = \frac{Y_P}{Y_{P,b}}, \text{ and}$$

$$c_T = \frac{(T - T_u)}{(T_b - T_u)}, \qquad (11)$$

where the subscripts, u and b denote unburnt and burnt mixtures, and F, O and

P denote fuel, oxidiser and product, respectively. As clearly seen from Eq. (11), a progress variable is 0 in the unburnt mixture and 1 in the fully burnt mixture. Figure 1b shows a typical progress variable variation for one-dimensional premixed flame. Flame front (reaction front) is often defined using a iso-contour of $c = 0.3 \sim 0.6$.

Lewis number Le is defined as the ratio of thermal diffusivity to mass diffusivity. If unity Lewis number, constant pressure, adiabatic conditions and $c = c_{Y_{F/O}} = c_{Y_P} = c_T$ are assumed, Eqs (8) and (9) are reduced to one transport equation of progress variable as:

$$\frac{\partial(\rho c)}{\partial t} + \frac{\partial}{\partial x_j}(\rho u_j c) = \frac{\partial}{\partial x_j}\left(\rho D \frac{\partial c}{\partial x_j}\right) + \omega_c, \quad (12)$$

where ω_c is the reaction rate of c, and D is the diffusivity of the progress variable (under the above assumptions, it can be either thermal diffusivity or mass diffusivity of species). Therefore, once a progress variable field is given, temperature and species mass fraction field can be obtained using Eq. (11). The reaction rate ω_c is modelled in the software as follows:

$$\omega_c = \rho k \exp\left(-\frac{T_a}{T}\right) T^n (1-c)\,\xi_c, \quad (13)$$

where k, T_a and n are mixture specific and constant for the single step reaction explained in Eq. (10). In Eq. (13), ξ_c is the mixture fraction between a fuel-oxidiser mixture and air (inert gas). Although the mixture fraction is explained in the next subsection, this may be used to represent surrounding air in premixed combustion. In $\xi_c = 1$ regions, the local mixture consists of only the fuel-oxidiser mixture, while only the air exists in $\xi_c = 0$. By multiplying ξ_c to the reaction rate as in Eq. (13), partially premixed reaction such as premixed combustion in surrounding air (pure-air stream) can be considered. When no pure-air stream is set in the computational domain, ξ_c is automatically set as unity. The scalar transport equation, Eq. (12) may be solved for non-reactive cases without source

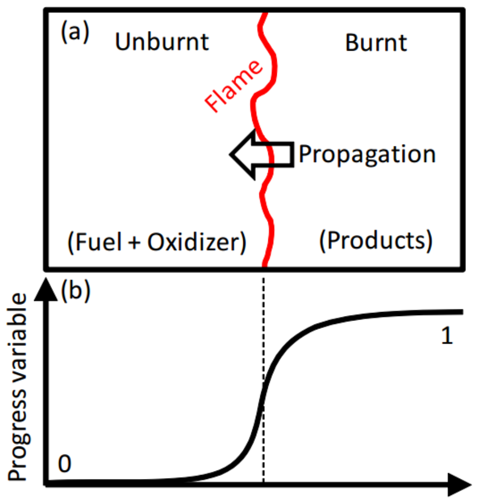


Figure 1: Schematic illustrations of (a) a premixed flame and (b) one-dimensional variation of progress variable.

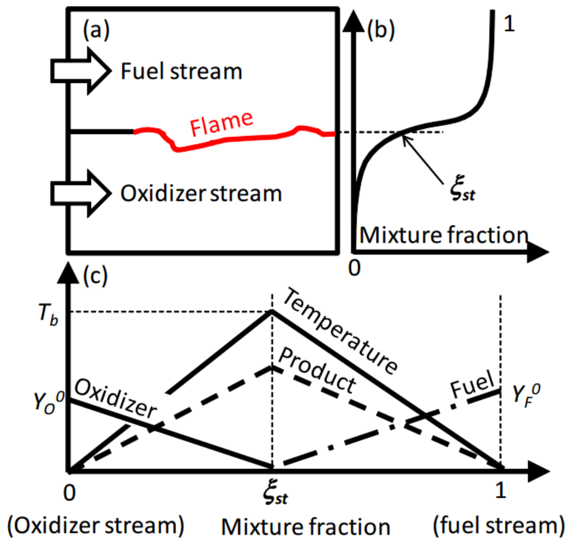


Figure 2: Schematic illustrations of (a) a non-premixed flame, (b) one-dimensional variations of mixture fraction, and (c) temperature and mass fraction variations in a mixture fraction space.

term ω_c when local scalar boundary condition is used (see Sec. 4.5.4).

2.2 Non-premixed reacting flows

In contrast to premixed reacting flows, non-premixed reactive systems do not require premixing of fuel and oxidiser before reactions. A schematic illustration of a typical non-premixed flame is shown in Fig. 2a. There are fuel and oxidiser streams and reaction takes place only at the location where the fuel and oxidiser meet.

Using the given chemical reaction model in Eq. (10), the mass production rate of fuel, oxidiser, product and heat release are related as:

$$-\omega_F = -\frac{\omega_O}{s} = \frac{\omega_P}{1+s} = \frac{\omega_T}{Q}, \qquad (14)$$

where Q is the heating value of fuel and s is the mass of oxidiser required per unit mass of fuel, defined as $s = W_O / W_F$. Here, W_i is the molar mass of species *i*. Under the unity Lewis number assumption, the conserved scalar β is defined from any pair of variables as:

$$\beta_{F,O} = sY_F - Y_O, \qquad (15)$$

$$\beta_{F,P} = Y_F + \frac{Y_P}{1+s}, \qquad (16)$$

$$\beta_{O,P} = Y_O + \frac{Y_P}{1+s}, \qquad (17)$$

$$\beta_{F,T} = Y_F + \frac{c_p T}{Q}, \qquad (18)$$

$$\beta_{F,O} = Y_O - \frac{sc_p T}{Q}. \qquad (19)$$

β is called conserved scalar because no source term appears in the transport equations of these conserved scalars (such transport equation can be obtained from Eqs. 8 and 9); for above $\beta_{F,O}$, the transport equations of sY_F and Y_O are written as:

$$\frac{\partial(\rho s Y_F)}{\partial t} + \frac{\partial}{\partial x_j}(\rho u_j s Y_F) = \frac{\partial}{\partial x_j}\left(\rho D \frac{\partial s Y_F}{\partial x_j}\right) + s\omega_F, \qquad (20)$$

$$\frac{\partial(\rho Y_O)}{\partial t} + \frac{\partial}{\partial x_j}(\rho u_j Y_O) = \frac{\partial}{\partial x_j}\left(\rho D \frac{\partial Y_O}{\partial x_j}\right) + \omega_O. \qquad (21)$$

By combining Eqs. (20) and (21) using the relation in Eq. (14), it is clear that the source terms are cancelled out and following is obtained:

$$\frac{\partial(\rho \beta_{F,O})}{\partial t} + \frac{\partial}{\partial x_j}(\rho u_j \beta_{F,O}) = \frac{\partial}{\partial x_j}\left(\rho D \frac{\partial \beta_{F,O}}{\partial x_j}\right) \qquad (22)$$

Once a conserved scalar is defined, a normalised conserved scalar ξ is obtained. This normalised conserved scalar is generally called mixture fraction, and written as:

$$\xi = \frac{\beta - \beta_{OX}}{\beta_{FU} - \beta_{OX}}, \qquad (23)$$

where the subscripts FU and OX denote fuel and oxidiser streams, respectively. Figure 2b shows a one-dimensional mixture fraction variation for a non-premixed flame shown in Fig. 2a. With the above definition of mixture fraction, $\xi = 1$ corresponds to the fuel stream and $\xi = 0$ corresponds to the oxidiser stream. As shown in Fig. 2b, the flame location in non-premixed systems is often assumed as the location where $\xi = \xi_{st}$. Here, ξ_{st} is the stoichiometric mixture fraction, at which there is the exact amount of oxidiser to convert unit quantity of fuel into product. The stoichiometric mixture fraction is written as:

$$\xi_{st} = \frac{-\beta_{OX}}{\beta_{FU} - \beta_{OX}}. \tag{24}$$

The mixture fraction also satisfies Eq. (22). Therefore, for the simulations of reactive non-premixed systems, the equations of energy and species can be reduced to one equation using mixture fraction as:

$$\frac{\partial(\rho\xi)}{\partial t} + \frac{\partial}{\partial x_j}(\rho u_j \xi) = \frac{\partial}{\partial x_j}\left(\rho D \frac{\partial \xi}{\partial x_j}\right). \tag{25}$$

Once the mixture fraction field is obtained, temperature and species mass fraction fields can be reconstructed using the relations in Fig. 2c. These relations are directly obtained from Eq. (25) by assuming infinite chemistry, but only results are shown here.

For $\xi < \xi_{st}$

$$Y_F = 0,$$

$$Y_O = Y_O^0\left(1 - \frac{\xi}{\xi_{st}}\right),$$

$$Y_P = (1-s)Y_F^0 \xi,$$

$$T = (T_b - T_O^0)\frac{\xi}{\xi_{st}} + T_O^0.$$

For $\xi \geq \xi_{st}$ (26)

$$Y_F = Y_F^0 \frac{\xi - \xi_{st}}{1 - \xi_{st}},$$

$$Y_O = 0,$$

$$Y_P = (1-s)Y_F^0 \xi \frac{1-\xi}{1-\xi_{st}}$$

$$T = \frac{T_b - T_F^0}{\xi_{st} - 1}\xi + \frac{T_F^0 \xi_{st} - T_b}{\xi_{st} - 1},$$

where T_b is the adiabatic flame temperature, Y_F^0 and Y_O^0 are respectively the fuel and oxidiser mass fraction at each stream, and T_F^0 and T_O^0 are temperatures at fuel and oxidiser stream, respectively.

2.3 Equation of state and density

In premixed reaction mode, Flowsquare solves Eqs. (1), (2) and (12), and Eq. (25) is solved for $\beta_{c,air}$ only if pure-air streams are set. In non-premixed mode, Eqs. (1), (2) and (25) are solved. Temperature is calculated from either the progress variable or mixture fraction field. Density field is obtained based on the standard thermal equation of state for the mixture:

$$p = \rho R T. \tag{27}$$

Here, R is the specific gas constant (J/kg K) calculated as $R = 8.31/W$, where W is the molar mass of the mixture. For instance, $R = 287.058$ (J/kg K) for dry air. In Flowsquare, all thermodynamic and transport properties are assumed as constant for reacting flow simulations.

For non-premixed mode, temperature is just mapped onto computational domain based on ξ. Therefore, the change of density (Eq. 27) can be much larger than that captured with Δt initially set. In order to avoid such situations and keep simulation stable, following relaxation operation can be used for density:

$$\rho_{n+1} = \sigma\rho'_{n+1} + (1-\sigma)\rho_n, \tag{28}$$

where σ is the relaxation parameter for density, and ρ'_{n+1} .is the density calculated using Eq. (27). Clearly, $\sigma = 1$ yields right solution (but tends to be unstable under low resolution conditions).

3. Subsonic/supersonic flows

In high speed flows, the viscous term in the momentum transport equation (Eq. 2) may be negligible:

$$\frac{\partial}{\partial t}\left(\rho u_j\right)+\frac{\partial}{\partial x_i}\left(\rho u_i u_j\right)=-\frac{\partial p}{\partial x_j}. \qquad (29)$$

This equation is a nonlinear hyperbolic equation, which is often used for simulations of waves. Also, since a fluid velocity is so large, incompressible flow assumption is no longer valid; energy conservation needs to be taken into account. Assuming inviscid fluids (or conditions), the energy conservation equation is written as:

$$\frac{\partial E}{\partial t}+\frac{\partial}{\partial x_i}\left((E+p)u_i\right)=0, \qquad (30)$$

where E is the total energy per unit volume (J/m^3). The total energy is related with the kinetic energy as:

$$E=\rho e+0.5\rho u_i u_i. \qquad (31)$$

Here, e is the internal energy per unit mass (J/kg) expressed as $\rho e=p/(\gamma-1)$ for ideal gas, where $\gamma(=c_p/c_v)$ is the adiabatic index (heat capacity ratio; approx. 1.4 for most of gases, and this value is used in the software). The static pressure is calculated in every time step as: $p=\frac{R}{c_v}\left(E+0.5\rho u_i u_i\right)$. The set of Eqs. (1), (2) and (3) is called Euler equations and often written in vector form as:

$$\frac{\partial q}{\partial t}+\frac{\partial F}{\partial x}+\frac{\partial G}{\partial y}=0, \qquad (32)$$

where

$$q=\begin{pmatrix}\rho\\ \rho u\\ \rho v\\ E\end{pmatrix}, F=\begin{pmatrix}\rho u\\ \rho u^2+p\\ \rho uv\\ (E+p)u\end{pmatrix}, G=\begin{pmatrix}\rho v\\ \rho uv\\ \rho v^2+p\\ (E+p)v\end{pmatrix}. \qquad (33)$$

They are solved (also with Eq 31) for fully-compressible subsonic/ supersonic (inviscid) flows.

4. Numerical methods

In Flowsquare ver 4, there are a variety of choices of numerical methods. Each method has pros and cons, and they need to be considered for the choice. In this section, these methods are summarised.

4.1. Differentiations

Spatial differentiations are computed on a uniform mesh (cell). The 1st derivative of f in *i* direction at a location *j* is computed using one of following methods ($1\le i\le N$, N is the number of grid points in *i* direction).

4.1.1. 2nd order central finite difference

$$\left(\frac{\partial f}{\partial x_i}\right)_j=(f_{j+1}-f_{j-1})/(2\Delta x_i), \qquad (34)$$

$$\left(\frac{\partial f}{\partial x_i}\right)_1=-(f_1-f_2)/\Delta x_i, \qquad (35)$$

$$\left(\frac{\partial f}{\partial x_i}\right)_N=(f_N-f_{N-1})/\Delta x_i. \qquad (36)$$

4.1.2. 4th order central finite difference

$$\left(\frac{\partial f}{\partial x_i}\right)_j=(-f_{j+2}+8f_{j+1}-8f_{j-1}+f_{i-2})/(12\Delta x_i) \qquad (36)$$

$$\left(\frac{\partial f}{\partial x_i}\right)_1=\frac{-25f_1+48f_2-36f_3+16f_4-3f_5}{12\Delta x_i} \qquad (37)$$

$$\left(\frac{\partial f}{\partial x_i}\right)_2 = \frac{-3f_1 - 10f_2 + 18f_3 - 6f_4 + f_5}{12\Delta x_i} \quad (38)$$

$$\left(\frac{\partial f}{\partial x_i}\right)_{N-1} = \frac{3f_N + 10f_{N-1} - 18f_{N-2} + 6f_{N-3} - f_{N-4}}{12\Delta x_i} \quad (39)$$

$$\left(\frac{\partial f}{\partial x_i}\right)_N = \frac{25f_N - 48f_{N-1} + 36f_{N-2} - 16f_{N-3} + 3f_{N-4}}{12\Delta x_i} \quad (40)$$

4.2. Time integration

In this section, formulations to obtain a solution in the next time step (n+1) from the current time step (n) are explained.

4.2.1. Euler method (1st order)

The easiest way to integrate the governing equations in time is to use Euler method which is explicit 1st order scheme:

$$f_{n+1} = f_n + \frac{\partial f_n}{\partial t} \cdot \Delta t . \quad (41)$$

This is first order forward method and is numerically unstable.

4.2.2. Lax-Wendroff method (=2nd order midpoint method)

Lax-Wendroff method is a numerical solution often used for hyperbolic partial differential equation. In this software, the midpoint method similar to Lax-Wendroff's time integration part can be used. This method compensates the lack of the Euler method by integrating in two steps.

$$f_n' = f_n + \frac{\partial f_n}{\partial t}\frac{\Delta t}{2}, \quad (42)$$

$$f_{n+1} = f_n' + \frac{\partial f_n'}{\partial t}\Delta t . \quad (43)$$

4.2.3. 3rd order Runge-Kutta (low-storage)

In 3rd order Runge-Kutta method, each time integration is achieved in following three sub steps.

$$\Delta f_n = \frac{1}{3}\Delta t \frac{\partial f_n}{\partial t}, \quad (44)$$

$$f_n' = f_n + \Delta f_n, \quad (45)$$

$$\Delta f_n' = \frac{15}{16}\Delta t\left(-\frac{5}{9}\Delta f_n + \frac{\partial f_n'}{\partial t}\right), \quad (46)$$

$$f_n'' = f_n' + \Delta f_n', \quad (47)$$

$$\Delta f_n'' = \frac{8}{15}\Delta t\left(-\frac{153}{128}\Delta f_n' + \frac{\partial f_n''}{\partial t}\right), \quad (48)$$

$$f_{n+1} = f_n'' + \Delta f_n'' . \quad (49)$$

4.3. Poisson's equation

Poisson's equation is solved iteratively to obtain pressure variation in Eq. (6). Poisson's equation is written as

$$\nabla^2 \phi = f , \quad (50)$$

where ∇^2 denotes the Laplace operator. If second order central differentiation is applied to above equation, the discrete form is:

$$\frac{\phi_{i+1,j} - 2\phi_{i,j} + \phi_{i-1,j}}{\Delta x^2} + \frac{\phi_{i,j+1} - 2\phi_{i,j} + \phi_{i,j-1}}{\Delta y^2} = f_{i,j} . \quad (51)$$

Therefore, the Poisson's equation can be reduced to *n* linear equations with unknown ϕ. In Flowsquare, the Poisson's equation is iteratively solved using Successive Over-Relaxation (SOR) method. The (*n+1*)-th iterative solution is calculated from *n*-th iterative solution as:

$$\phi_{i,j}^{n+1} = (1-\omega)\phi_{i,j}^{n} + \omega \times \left(\frac{\phi_{i+1,j}^{n} + \phi_{i-1,j}^{n+1}}{\Delta x^2} + \frac{\phi_{i,j+1}^{n} + \phi_{i,j-1}^{n+1}}{\Delta y^2} - f_{ij} \right) \times \frac{\Delta x^2 \Delta y^2}{2(\Delta x^2 + \Delta y^2)}, \quad (52)$$

where ω is the relaxation parameter. Values of ω>1 are used to accelerate convergence speed, while ω <1 are used to help convergence solutions of an unstable iterative process. Typically, ω =1.6—1.8 is used for both better results and fast convergence. The iteration calculation is continued until $\phi^{n+1} - \phi^{n} < \varepsilon_p$, where ε_p is tolerance and specified by users.

4.4. Spatial filtering

Spatial filtering is sometimes required to stabilise the calculations. In Flowsquare, the second order explicit filter can be used with a relaxation parameter as follows:

$$\hat{f}_i = (1-\omega_{fil}) f_i + \omega_{fil} (f_{i+1} + 2 f_{ij} + f_{i-1}) / 4 \quad . \quad (53)$$

The relaxation parameter ω_{fil} ranges from 0 to 1. Minimum possible ω_{fil} is recommended.

4.5. Boundary conditions

Various boundary conditions can be used in Flowsquare. Boundary conditions are roughly classified into two types: Dirichlet boundary condition and Neumann boundary condition. Dirichlet BC is also called fixed BC, because it specifies the values that a solution takes on the boundary of the computational domain. In contrast, Neumann BC specifies values that the derivative of a solution is to take on the boundary of the domain. The value is determined to make the derivative zero ($f_1 = f_2$ and $f_N = f_{N-1}$). In Flowsquare, these two types of BC are used appropriately depending on the physical quantities.

4.5.1. Inflow boundary

On the inflow boundary, P is set using Neumann BC. Other quantities such as ρ, u, v, T, c, ξ and E are specified according to Dirichlet BC depending on the simulation mode. Some of these values are explicitly set by users, while the others are computed appropriately during the simulation. Inflow boundaries can be set only on the boundaries of the computational domain ($i=1, i=N_x$, $j=1$ and $j=N_y$).

4.5.2. Outflow boundary

The minimum pressure P on the outflow boundaries is always set to be a reference pressure P_0 (Dirichlet BC) which is explicitly set by users. On other outflow boundary locations, P is set according to Neumann BC. Other quantities such as ρ, u, v, T, c and ξ are specified according to Neumann BC.

4.5.3. Periodic boundary

One or both of two directions can be set as periodic boundaries. On the periodic boundaries, derivatives are simply computed (shown only for 2nd order scheme in Eq. 34, but equally applied to other schemes as well) as:

$$\left(\frac{\partial f}{\partial x_i} \right)_1 = (f_2 - f_N)/(2\Delta x_i), \quad (54)$$

$$\left(\frac{\partial f}{\partial x_i} \right)_N = (f_1 - f_{N-1})/(2\Delta x_i). \quad (55)$$

4.5.4. Local scalar boundary

Sometimes, users want to set source of scalar, c or ξ, for Eq. (12) or (25) inside the domain. Such boundaries can be used as tracer in the case of non-reactive flows, or as non-reactive area for a convenience in the case of reactive case. For this type of boundary, the user-defined scalar value is set on the boundary. Flow velocity and other quantities are computed by solving the transport equations as usual.

4.5.5. Wall boundary

Wall boundaries can be set at any locations in the domain. On the wall boundary, the velocity is set as zero. All other quantities are solved using the transport equations, although temperature can be fixed on the wall (optional).

4.5.6. Moving boundary

Moving boundary is basically moving wall boundary. The displacement velocity of the wall can be specified by users and the movement can be transient (if the moving wall has gone, it's gone) or periodic (when moving wall left the domain it comes back from the other side of the domain). The velocity on the moving wall is fixed to the velocity of displacement. All other quantities are solved using the transport equations, although temperature can be fixed on the wall (optional).

4.5.7. Air-flow boundary

Only for the simulation of premixed reacting flows, pure-air stream can be set on the boundary of the numerical domain. The velocity and temperature are set (Dirichlet BC) explicitly by users. The mixture fraction between progress variable and air, ξ_c, for Eqs. (13) and (25) is set as 0 in the pure-air stream.

4.6 Initial velocity perturbation

In Flowsquare, velocity perturbation, (u_p, v_p), can be added to the initial global velocity field. There are three mode of perturbation.

Mode 1: single mode

$$\begin{aligned} u_p = +u_{mag} &\cdot \sin\left(2\pi \cdot \frac{i-1}{N_x-1} n_{wave}\right) \\ &\cdot \cos\left(2\pi \cdot \frac{j-1}{N_x-1} n_{wave}\right) \\ -u_{mag} &\cdot \cos\left(2\pi \cdot \frac{i-1}{N_x-1} n_{wave}\right) \\ &\cdot \sin\left(2\pi \cdot \frac{j-1}{N_x-1} n_{wave}\right) \end{aligned}, \qquad (56)$$

$$\begin{aligned} v_p = -u_{mag} &\cdot \cos\left(2\pi \cdot \frac{i-1}{N_x-1} n_{wave}\right) \\ &\cdot \sin\left(2\pi \cdot \frac{j-1}{N_x-1} n_{wave}\right) \\ -u_{mag} &\cdot \sin\left(2\pi \cdot \frac{i-1}{N_x-1} n_{wave}\right) \\ &\cdot \cos\left(2\pi \cdot \frac{j-1}{N_x-1} n_{wave}\right) \end{aligned}, \qquad (57)$$

Mode 2: multi modes

$$u_p = +\sum_{n=0}^{M} \left[\begin{aligned} +u_{mag} &\cdot \sin\left(\pi \cdot \frac{i-1}{N_x-1} \frac{n_{wave}}{2^{n-1}}\right) \\ &\cdot \cos\left(\pi \cdot \frac{j-1}{N_x-1} \frac{n_{wave}}{2^{n-1}}\right) \\ -u_{mag} &\cdot \cos\left(\pi \cdot \frac{i-1}{N_x-1} \frac{n_{wave}}{2^{n-1}}\right) \\ &\cdot \sin\left(\pi \cdot \frac{j-1}{N_x-1} \frac{n_{wave}}{2^{n-1}}\right) \end{aligned} \right], (58)$$

$$v_p = -\sum_{n=0}^{M} \begin{bmatrix} +u_{mag} \cdot \cos\left(\pi \cdot \dfrac{i-1}{N_x - 1} \dfrac{n_{wave}}{2^{n-1}} \right) \\ \cdot \sin\left(\pi \cdot \dfrac{j-1}{N_x - 1} \dfrac{n_{wave}}{2^{n-1}} \right) \\ -u_{mag} \cdot \sin\left(\pi \cdot \dfrac{i-1}{N_x - 1} \dfrac{n_{wave}}{2^{n-1}} \right) \\ \cos\left(\pi \cdot \dfrac{j-1}{N_x - 1} \dfrac{n_{wave}}{2^{n-1}} \right) \end{bmatrix}, \quad (59)$$

Mode 3: multi modes (random amplitudes)

$$u_p = +\sum_{n=0}^{M} \begin{bmatrix} +u_{rnd} \cdot \sin\left(\pi \cdot \dfrac{i-1}{N_x - 1} \dfrac{n_{wave}}{2^{n-1}} \right) \\ \cdot \cos\left(\pi \cdot \dfrac{j-1}{N_x - 1} \dfrac{n_{wave}}{2^{n-1}} \right) \\ -u_{rnd} \cdot \cos\left(\pi \cdot \dfrac{i-1}{N_x - 1} \dfrac{n_{wave}}{2^{n-1}} \right) \\ \cdot \sin\left(\pi \cdot \dfrac{j-1}{N_x - 1} \dfrac{n_{wave}}{2^{n-1}} \right) \end{bmatrix}, \quad (60)$$

$$u_p = -\sum_{n=0}^{M} \begin{bmatrix} +v_{rnd} \cdot \cos\left(\pi \cdot \dfrac{i-1}{N_x - 1} \dfrac{n_{wave}}{2^{n-1}} \right) \\ \cdot \sin\left(\pi \cdot \dfrac{j-1}{N_x - 1} \dfrac{n_{wave}}{2^{n-1}} \right) \\ -v_{rnd} \cdot \sin\left(\pi \cdot \dfrac{i-1}{N_x - 1} \dfrac{n_{wave}}{2^{n-1}} \right) \\ \cdot \cos\left(\pi \cdot \dfrac{j-1}{N_x - 1} \dfrac{n_{wave}}{2^{n-1}} \right) \end{bmatrix}, \quad (61)$$

where u_{mag} and n_{wave} are user defined parameters, and $M = \log_2 n_{wave}$. The perturbation amplitudes, u_{rnd} and v_{rnd} are

$$u_{rnd} = u_{mag} R_1(n) \quad , \quad v_{rnd} = u_{mag} R_2(n), \quad (62)$$

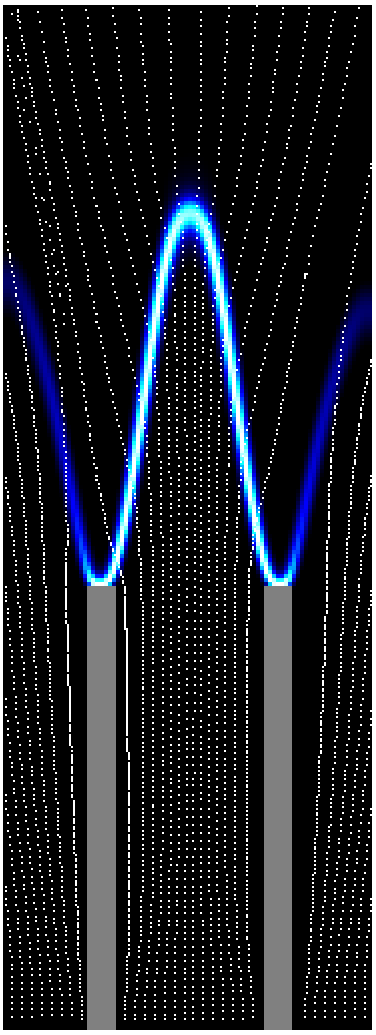
where $R_1(n)$ and $R_2(n)$ are random numbers ranging from 0 to 1, generated for each *n*. Note these random numbers for u_p and v_p are different in general.

5. Literatures to consult with would be:

Poinsot, T. and Veynante, D. (2005) *Theoretical and numerical combustion.* R.T. Edwards, Inc.

Cant, R. S. and Mastorakos, E. (2008) *An introduction to turbulent reacting flow.* Imperial College Press.

Sana, O. and Kara, K. (2013) *Numerical assessments of high-order accurate shock capturing schemes: Kelvin-Helmholtz type vertical structures in high-resolutions.* Computers and Fluids.


A Bunsen flame simulated using Flowsquare.