OCTA

Integrated simulation system for soft materials

Multi-Phase Dynamics Program

Muffin

version 5.1

User's Manual

- Volume II -

Multi-Fluid Phase Dynamics Simulator

PhaseSeparation

OCTA User's Group

January 01 2016

Authors of the Manual

Makoto Sasaki and Takashi Taniguchi

Programers

Takashi Taniguchi (FDM) and Makoto Sasaki (FEM)

Version 5.0 release

Programer, Authors of the Manual Tatsuya Yamaue, Taku Ozawa

Version 5.1 release

Programer, Authors of the Manual Taku Ozawa

Acknowledgment

This work is supported by the national project, which has been entrusted to the Japan Chemical Innovation Institute (JCII) by the New Energy and Industrial Technology Development Organization (NEDO) under METI's Program for the Scientific Technology Development for Industries that Creates New Industries.

Contents

List of Figures

Chapter 1

Theoretical background of PhaseSeparation

The PhaseSeparation of MUFFIN is a simulator which deals with the dynamics of phase separation induced by the change of thermodynamics variables or external fields.

The main target of this simulator is the dynamics of the polymer mixture system in a liquid state. Considering the dynamics of a system in a liquid state, key fields relevant to a certain phenomenon depend on a problem. It is desired to use various equations for calculating chemical potential and to use various calculation techniques for performing a simulation. Thus, the PhaseSeparation simulator is designed to have flexibility, which makes it possible to construct a simulator by combining fields relevant to a phenomenon. In the following section 1.1, the calculation principle of a multiphase fluid system will be explained.

1.1 Basic equations of PhaseSeparation

1.1.1 Equation of a volume fraction

The MUFFIN fluid simulator deals with the dynamics of the multi-component mixture system under a flow. Now, we consider a fluid mixture which consists of M components. The temperature of the system is assumed to be a constant T_o. This system is expressed by a spatial distribution of the volume fraction field of each component. The volume fraction of the α component in a volume element $v_o = a^3$ (a is monomer size here) at a position $\boldsymbol{r}$ is expressed as $\psi_\alpha(\boldsymbol{r})$. [1] The subscript α specifies a component, and can take the value of $\{0, 1, \cdots, M-1\}$. The time evolution of the volume fraction of each component is described by the following equation of continuity,

$$\frac{\partial \psi_\alpha}{\partial t} = -\nabla \cdot \boldsymbol{J}_\alpha^{(Total)}, \tag{1.1}$$

where $\boldsymbol{J}_\alpha^{(Total)}$ of the right-hand side expresses the flux of a component α. This flux $\boldsymbol{J}_\alpha^{(Total)}$ consists of the following three contributions,

$$\boldsymbol{J}_\alpha^{(Total)} = \boldsymbol{J}_\alpha^{(h)} + \boldsymbol{J}_\alpha^{(d)} + \boldsymbol{J}_\alpha^{(r)}, \tag{1.2}$$

where $\boldsymbol{J}_\alpha^{(h)}$ means the flux of the component α induced by the hydrodynamics flow, and is expressed as $\boldsymbol{J}_\alpha^{(h)} = \boldsymbol{v}\psi_\alpha$ using the local velocity $\boldsymbol{v}$ of the fluid. The second contribution $\boldsymbol{J}_\alpha^{(d)}$ denotes the diffusion flux induced by a thermodynamics force. Assuming this diffusion flux is proportional to the spatial gradient of chemical potential, it is given as

$$\boldsymbol{J}_\alpha^{(d)} = -\sum_{\alpha'} \mathcal{L}_{\alpha\alpha'} \nabla \mu_{\alpha'}, \tag{1.3}$$

where, $\mathcal{L}_{\alpha\alpha'}$ is a transport coefficient and μ_α is a chemical potential. When the expression of free-energy F is given by the Flory-Huggins theory, the chemical potential can be expressed as

$$\mu_\alpha^{(FH)} = \frac{\delta F}{\delta \psi_\alpha(\boldsymbol{r})}. \tag{1.4}$$

[1] In the below, Greek characters like α, β are used as suffixes indicating components, and i, j, k are used as suffixes indicating x, y or z.

When the calculation method of chemical potential is not specified, the chemical potential will be written by μ_α. Superscripts as follows are used to specify a calculation method.

$$\begin{aligned}
\mu_\alpha^{(FH)} &: \text{Flory-Huggins De Gennes} \\
\mu_\alpha^{(GL)} &: \text{Symmetric Ginzburg Landau free energy} \\
\mu_\alpha^{(OK)} &: \text{Ohta-Kawasaki} \\
\mu_\alpha^{(ADF)} &: \text{Approximate Density Functional Theory} \\
\mu_\alpha^{(SCF)} &: \text{the self consistent mean field theory}
\end{aligned}$$

The transport coefficient is usually assumed to be diagonal as $\mathcal{L}_{\alpha\alpha'} = \mathcal{L}_\alpha \delta_{\alpha\alpha'}$ in many cases, and this assumption is also used here. So the diffusion flux is expressed as $\boldsymbol{J}_\alpha^{(d)} = -\mathcal{L}_\alpha \nabla \mu_\alpha$. The ψ_α dependency of $\mathcal{L}_\alpha$ may be taken as

$$\mathcal{L}_\alpha = L_\alpha \psi_\alpha \qquad (L_\alpha = \text{Const.}), \tag{1.5}$$

so that it agree with the diffusion coefficient in single phase state. When the composition dependence of a transport coefficient is not important for the system, it may be simplified as

$$\mathcal{L}_\alpha = L_\alpha = Const.. \tag{1.6}$$

The equation (1.5) is used for electrolytes, while equation (1.6) is used for other systems. The diffusion flux can be induced by other thermodynamics forces. The diffusion flux may be a function of some thermodynamics forces such as the diffusion flow in a viscoelastic system or the one induced by a temperature gradient. The third contribution $\boldsymbol{J}_\alpha^{(r)}$ is a random current induced by the thermal noise. Its magnitude is determined to satisfy the following fluctuation dissipation theorem.

$$\langle J_{\alpha i}^{(r)}(\boldsymbol{r}, t) J_{\alpha i}^{(r)}(\boldsymbol{r'}, t') \rangle = 2 k_B T \mathcal{L}_\alpha \delta_{ij} \delta(\boldsymbol{r} - \boldsymbol{r'}) \delta(t - t'). \tag{1.7}$$

Finally, the equation of the time evolution for the volume fraction ψ_α is given as

$$\frac{\partial \psi_\alpha}{\partial t} = -\nabla \cdot (\boldsymbol{v} \psi_\alpha) - \nabla \cdot \boldsymbol{J}_\alpha, \qquad \boldsymbol{J}_\alpha = -\mathcal{L}_\alpha \nabla \mu_\alpha + \boldsymbol{J}_\alpha^{(r)}. \tag{1.8}$$

1.1.2 Equation for fluid

In this section, we consider the equation of the fluid. Because we are considering a system with large viscosity like a polymeric system, its Reynolds number is small enough. Such a system is described by the Navier-Stokes equation ignoring the convection term (Stokes approximation) :

$$\rho \frac{\partial \boldsymbol{v}}{\partial t} = \nabla \cdot \big(\eta \boldsymbol{D}\big) - \nabla p + \boldsymbol{K}, \tag{1.9}$$

where $\boldsymbol{v}$ is the velocity, p is the pressure, ρ is the density, and η is the viscosity. $\boldsymbol{D}_{ij}$ is a velocity gradient tensor defined by

$$\boldsymbol{D}_{ij} = \partial v_i / \partial x_j + \partial v_j / \partial x_i.$$

The third term $\boldsymbol{K}(\boldsymbol{r})$ of the right-hand side of the Stokes equation is a body force. Since a dynamic balance is satisfied in a very short time when the viscosity is large enough, it can be assumed that the time change of a flow is small enough to make $\partial \boldsymbol{v} / \partial t = 0$. So we can use the equation of fluid as follows;

$$\nabla \cdot (\eta \boldsymbol{D}) - \nabla p + \boldsymbol{K} = \boldsymbol{0}. \tag{1.10}$$

We impose an incompressibility condition expressed by the following equation

$$\nabla \cdot \boldsymbol{v} = 0. \tag{1.11}$$

Using eq.(1.11), we can derive the following Poisson equation for the pressure

$$\Delta p = \nabla \nabla (\eta \boldsymbol{D}) + \nabla \cdot \boldsymbol{K}. \tag{1.12}$$

Summary

Basic equations

$$\frac{\partial \psi_\alpha}{\partial t} = -\nabla \cdot (\boldsymbol{v}\psi_\alpha) - \nabla \cdot \boldsymbol{J} \tag{1.13}$$

$$\rho \frac{\partial \boldsymbol{v}}{\partial t} = \nabla \cdot \big(\eta \boldsymbol{D}\big) - \nabla p + \boldsymbol{K}, \qquad \nabla \cdot \boldsymbol{v} = 0 \tag{1.14}$$

In the PhaseSeparation simulator of MUFFIN, the simulation of various systems is realized by changing the function which calculates a chemical potential and so on.

1.1.3 Free energy and chemical potential of M component polymer mixture system

The free energy based on the Flory-Huggins-deGenne theory is written as,

$$F^{(FH)} = \frac{k_B T}{v_o} \int dV \Big[\sum_{\alpha=0}^{M-1} \frac{\psi_\alpha}{N_\alpha} \ln \psi_\alpha + \sum_{\alpha<\alpha'}^{M-1} \chi_{\alpha\alpha'} \psi_\alpha \psi_{\alpha'} + \frac{1}{2} \sum_{\alpha<\alpha'}^{M-1} C_{\alpha\alpha'}(\psi_\alpha, \psi_{\alpha'}) [\nabla(\psi_\alpha - \psi_{\alpha'})]^2 \Big]. \tag{1.15}$$

In the eq.(1.15) the coefficient C_{ij} is given as

$$C_{\alpha\alpha'} = \frac{a^2}{18 \psi_\alpha \psi_{\alpha'}}, \tag{1.16}$$

by a random phase approximation (RPA). Here, v_0 is given by $v_0 = a^3$ using monomer size a. $\chi_{\alpha\alpha'}$ is known as Flory's χ-parameter. For the chemical potential $\mu_\alpha = \delta F / \delta \psi_\alpha$, when we define $\hat{\mu}_\alpha$ as chemical potential of a component α calculated as $\delta F / \delta \psi_\alpha$ making $\psi_0,\ \psi_1,\ \psi_2, \cdots \psi_{M-1}$ independent variables, μ_α is written as

$$\mu_\alpha = \hat{\mu}_\alpha - \hat{\mu}_0, \tag{1.17}$$

$$\hat{\mu}_\alpha^{(FH)} = \frac{k_B T}{v_o} \left[\frac{1}{N_\alpha} \ln \psi_\alpha + \sum_{\alpha'=0}^{M-1} \Big[\chi_{\alpha\alpha'} \psi_{\alpha'} - \frac{1}{2} \frac{\partial C_{\alpha\alpha'}}{\partial \psi_\alpha} \{\nabla(\psi_\alpha - \psi_\alpha)\}^2 - \nabla \{ C_{\alpha\alpha'} \nabla(\psi_\alpha - \psi_{\alpha'}) \} \Big] \right]. \tag{1.18}$$

Driving force term $\boldsymbol{K}$ is given by

$$\boldsymbol{K} = - \sum_{\alpha=1}^{M-1} \psi_\alpha \nabla \mu_\alpha^{(FH)}. \tag{1.19}$$

1.1.4 Free energy and chemical potential of a M component system under an electric field

The charge density in a tiny volume v_o at the position $\boldsymbol{r}$ can be written as

$$\rho_e(\boldsymbol{r}) = \sum_{\alpha=0}^{M-1} \rho_{e\alpha} \psi_\alpha(\boldsymbol{r}), \tag{1.20}$$

where $\rho_{e\alpha} = e\zeta_\alpha$ and ζ_α is a valence density of the α component. The dielectric constant in a mixed state is assumed to be given by a following simple mixing rule:

$$\epsilon(\boldsymbol{r}) = \sum_{\alpha=0}^{M-1} \epsilon_\alpha \psi_\alpha(\boldsymbol{r}), \tag{1.21}$$

where ϵ_α is the dielectric constant of the α component. Using the expressions the free energy is written as follows

$$F = F_{mix} + \int dV \big[-\frac{1}{2} \epsilon(\{\psi_\alpha\}, T) \boldsymbol{E}^2 + \frac{1}{2} \rho_e(\{\psi_\alpha\}) \Phi \big], \tag{1.22}$$

where $\boldsymbol{E} = \boldsymbol{E}(\boldsymbol{r})$ is the electric field and $\Phi = \Phi(\boldsymbol{r})$ is the electric potential (scalar potential). The chemical potential is written as

$$\mu_\alpha^{(X)} = \mu_{\alpha mix}^{(X)} - \frac{1}{2}(\epsilon_\alpha - \epsilon_0)\boldsymbol{E}^2 + (\rho_{e\alpha} - \rho_{e0})\Phi, \tag{1.23}$$

where $\mu_{\alpha mix}^{(X)}$ denotes a chemical potential coming from a mixing free energy and the superscript X expresses the calculation method ($X = FH, ADF$ and SCF, *etc.*). The body force $\boldsymbol{K}(\boldsymbol{r})$ can be written using the expression of this chemical potential

$$\boldsymbol{K} = -\sum_{\alpha=1}^{M-1} \psi_\alpha \nabla \mu_\alpha. \tag{1.24}$$

Although eq.(1.24) is the same expression as the one in the previous section, in addition to the effect of the osmotic pressure and a surface tension, the Maxwell stress and the body force coming from the charge are newly included in $\boldsymbol{K}$.

Since the electric field $\boldsymbol{E}$ and the scalar potential are added as new fields which are necessary for describing the chemical potential and the driving force term of this system, the equations to solve them are required. Supposing that $\nabla \times \boldsymbol{E} = 0$ is satisfied for the electric field, the electric field are expressed as $\boldsymbol{E} = -\nabla\Phi$ using scalar potential Φ. Using the Maxwell equation: $\mathrm{div}\boldsymbol{D} = \rho_e(\boldsymbol{r})$ and $\boldsymbol{D} = \epsilon\boldsymbol{E}$, the electrostatic potential is calculated by

$$\nabla\cdot\Big[\epsilon(\boldsymbol{r})\nabla\Phi\Big] = -\rho_e(\boldsymbol{r}). \tag{1.25}$$

1.2 Equations for various systems and similarity

In this section, we explain the similarity of equations for various systems using four examples. The first example is an equation which describes the Stokes flow. Here, we assume that no body force is applied on the fluid.

Stokes flow

$$\nabla \cdot (\eta \boldsymbol{D}) - \nabla p + \boldsymbol{K} = 0 \tag{1.26}$$
$$\nabla \cdot \boldsymbol{v} = 0 \tag{1.27}$$
$$\boldsymbol{K} = 0 \tag{1.28}$$

The second example is the equations which describe the dynamics of a multi-component system (a system of polymer blend or block copolymer, the phase separation under electric field). The effect of a hydrodynamic flow can also be taken into account by combining these equations with the equation of fluid. Some approximation theories exist to obtain the chemical potential of an M component system (Flory-Huggins, Approximate Density Functional Theory, Self-Consistent Field Theory, *etc*), and it is necessary to use them properly according to a required level of approximation. In this simulator, the phase separation will be dealt with using the Flory Huggins theory. It is possible to calculate chemical potential by other calculation methods by extending this simulator.

Multi-component mixture system

$$\frac{\partial \psi_\alpha}{\partial t} = -\nabla \cdot (\psi_\alpha \boldsymbol{v}) - \nabla \cdot \boldsymbol{J}_\alpha \tag{1.29}$$
$$\boldsymbol{J}_\alpha = -L_\alpha \nabla \mu_\alpha + \boldsymbol{J}_\alpha^{(r)} \tag{1.30}$$
$$\boldsymbol{K} = \sum_{\alpha=1}^{M-1} \boldsymbol{K}_\alpha = -\sum_{\alpha=1}^{M-1} \psi_\alpha \nabla \mu_\alpha \tag{1.31}$$

and Eq.(1.26) and Eq.(1.27)

The third example is an electrolyte system. Although there is a difference between concentration C_α and volume-fraction ψ_α, it turns out that the equations themselves are completely equivalent to the ones of the M component mixture under an electric field.

Electrolyte

$$\frac{\partial C_\alpha}{\partial t} = -\nabla \cdot (\boldsymbol{v} C_\alpha) - \nabla \cdot \boldsymbol{J}_\alpha \qquad \boldsymbol{J}_\alpha = -D_\alpha \Big[\nabla C_\alpha + \frac{\rho_{e\alpha} \boldsymbol{E}(\boldsymbol{r})}{2 k_B T} \Big] \tag{1.32}$$
$$\boldsymbol{K} = -k_B T \sum_\alpha \Big[\nabla C_\alpha + \frac{\rho_{e\alpha} \boldsymbol{E}(\boldsymbol{r})}{2 k_B T} \Big] \tag{1.33}$$
$$F = \int d^d \boldsymbol{r} k_B T C_\alpha \ln C_\alpha + \frac{1}{2} \int d^d \boldsymbol{r} \int d^d \boldsymbol{r'} \frac{C_\alpha C_\beta q_\alpha q_\beta}{4\pi\epsilon |\boldsymbol{r} - \boldsymbol{r'}|} \tag{1.34}$$

and Eq.(1.26) and Eq.(1.27)

An immiscible electroviscous fluid (block copolymer under electric field), an electrolyte, and a polyelectrolyte can be described by completely similar equations. The only difference is the expression of the chemical potential.

The fourth example is the equations which describe viscoelastic phase separation in a polymer solution system. The equations are derived based on the two fluid model. In eq.(1.35) - (1.39), ψ_p is the volume fraction of a polymer. In this system, the network stress resulting from entanglements of polymers plays a very important role. So it is necessary to calculate the network stress using a constituitive equation *etc*.

Viscoelastic phase separation

$$\frac{\partial \psi_p}{\partial t} = -\nabla \cdot (\boldsymbol{v}\psi_p) - \nabla \cdot \boldsymbol{J} \tag{1.35}$$

$$\boldsymbol{J} = -L\Big[\psi_p \nabla \mu - \nabla \cdot \sigma^{(N)}\Big] \tag{1.36}$$

$$L = \frac{1}{\zeta}\psi_s^2 \psi_p \simeq \frac{a^2}{6\pi\eta_s \psi_p} \qquad \text{(in } \Theta \text{ solvent)} \tag{1.37}$$

$$\boldsymbol{K} = -\psi_p \nabla \mu + \nabla \cdot \sigma^{(N)} \tag{1.38}$$

$$\nabla \cdot \sigma^{(N)} : \text{Contribution from network stress} \tag{1.39}$$

and Eq.(1.26) and Eq.(1.27)

As described above, it turns out that various systems can be described by the equation group with the same structure. Expressions of the chemical potential and the body force determine the characteristic of these systems.

1.3 Dimensionless expressions of parameters and equations

1.3.1 The unit of time and length

In the numerical calculation, it is important to make equations and parameters dimensionless. Using the characteristic time and spatial scale of a system, the dimensionless control parameters of the target system can be obtained. Let us return to a set of basic equations eqs.(1.13) and (1.14) again.

Basic equations

$$\frac{\partial \psi_\alpha}{\partial t} = -\nabla \cdot (\boldsymbol{v}\psi_\alpha) - \nabla \cdot \boldsymbol{J}_\alpha \tag{1.40}$$

$$\rho \frac{\partial \boldsymbol{v}}{\partial t} = \nabla \cdot \big(\eta \boldsymbol{D}\big) - \nabla p + \boldsymbol{K}, \qquad \nabla \cdot \boldsymbol{v} = 0 \tag{1.41}$$

In the following, the definition of dimensionless variables and dimensionless parameters are explained.

Notice

In the following, a tilde sign will be attached to the dimensionless variable.

[The space unit in a simulation]

In a system for which we perform simulations, there are some characteristic lengths which have physical meanings. For example, they are the characteristic size $L(t)$ of domain, the interface width λ, the gyration radius R_g of a polymer, the monomer size a, *etc.* For lengths characteristic to the system, it is very important to specify the shortest length that must be expressed by the continuum picture. Because, if a grid size larger than the reasonable short length which appears in a system is used, the precision of calculation will significantly decrease and calculation will often fail. The shortest length which must be expressed by the continuum picture among these characteristic lengths is **a width of an interface** λ between two phases. If many phases coexist, you have to find out the shortest width of interface. Since the width of the interface is the minimum length which appears in this system, the size of mesh must be small enough to express the smooth interface.

Although there is no quantitative results from a theoretical analysis, it is known empirically that the space-lattice size Δx which is about $1/4$ of the interface width λ is sufficient to express the interface configuration with a good precision (five lattice points in the interface). This length is expressed by $\xi \simeq \lambda/4$. Since ξ only differs from the interface width λ at most about 4 times, we call this length as the interface width

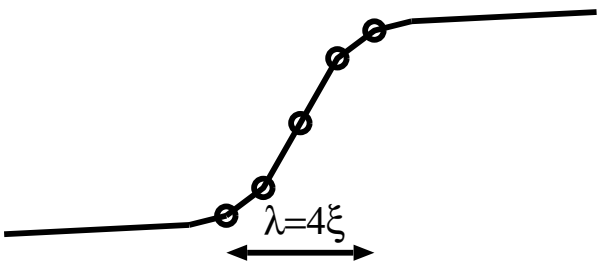

Figure 1.1: Explanation for interface width. ξ is about 1/4 of λ.

hereafter. The interface width depends not only on the kinds of materials, but also on the χ-parameter, the temperature, *etc.* For example, the interface width becomes very large near the critical point, and diverges at the critical point. It is very important to know how the interface width ξ depends on physical constants such as the χ-parameter (temperature). Since the M component system is considered now, there is ${}_MC_2 = M(M-1)/2$ kinds of combination of phases forming interfaces. For such a case, we have to find out the minimum of width ξ, and makes it the minimum length which exists in a system. This minimum length is taken as the unit of a space ξ. How does it depend on the χ-parameter or the degree of polymerization? Considering the interface of two phases near the critical point of two phase (α, α') and using eq.(1.15) for free energy, the interface width $\xi_{\alpha\alpha'}$ between two phases (α, α') will be given as

$$\xi_{\alpha\alpha'} \equiv \sqrt{\frac{C_{\alpha\alpha'}}{\Delta\chi_{\alpha\alpha'}}} = \sqrt{\frac{1}{\Delta\chi_{\alpha\alpha'}} \cdot \frac{a^2}{18\bar{\psi}_\alpha \bar{\psi}_{\alpha'}}}, \tag{1.42}$$

where $\Delta\chi_{\alpha\alpha'} \equiv \chi_{\alpha\alpha'} - \chi^{(c)}_{\alpha\alpha'}$, $\chi^{(c)}_{\alpha\alpha'} \equiv (\sqrt{N_\alpha} + \sqrt{N_{\alpha'}})^2/2N_\alpha N_{\alpha'}$. At first, we have to calculate $\xi_{\alpha\alpha'}$, then we determine the minimum width $\xi_{\beta\beta'}$ by using the following equation:

$$\xi_{\beta\beta'} \equiv \underset{<\alpha,\alpha'>}{\mathrm{Min}} \xi_{\alpha\alpha'}. \tag{1.43}$$

(In order to reduce symbol, the combination of the minimum interface width will be described as $\xi \equiv \xi_{\beta\beta'}$. If there is no confusion for the following abbreviation, $\Delta\chi_{\beta\beta'}$ will be written as $\Delta\chi$.) Using $e_o = (k_B T/v_0)\Delta\chi$ as a unit of energy density, the chemical potential will be scaled as

$$\begin{aligned} \tilde{\mu}_\alpha &= \mu_\alpha/e_o \\ &= \frac{1}{\Delta\chi_{\beta\beta'}}\Big[\frac{1}{N_\alpha}\ln\psi_\alpha + \sum_{\alpha'=0}^{M-1}\chi_{\alpha\alpha'}\psi_{\alpha'}\Big] - \sum_{\alpha'=0}^{M-1}\frac{C_{\alpha\alpha'}}{C_{\beta\beta'}}\tilde{\Delta}(\psi_\alpha - \psi_{\alpha'}), \end{aligned} \tag{1.44}$$

where $\tilde{\Delta} \equiv \Delta \cdot \xi^2_{\beta\beta'}$.

[Time unit in the simulation]

Next, we consider the unit of time. In order to determine the unit of time τ, it is important to find out the fastest transport phenomena in the system.

As described in section 1.1, two transport mechanism exist in the system. These are the diffusional transport and the hydrodynamics one. Therefore, in order to determine the unit of time, it is necessary to compare the quickest transport by the diffusion with the hydrodynamic transport [2]. These two characteristic times are defined as follows using the shortest interface width ξ which appears in the phase-separation phenomenon of this system.

Two time scales

(1) Being $\tau_\alpha^{(D)}$ a time required for component α to diffuse over the distance of the interface width ξ, τ_D is the minimum value among $\tau_\alpha^{(D)}$ ($\tau_D = \underset{\alpha}{\mathrm{Min}}\ \tau_\alpha^{(D)}$).

(2) Time which is needed a substance to be transported by the distance ξ by a flow U which exists in a system (an imposed flow field or interfacial tension).

The characteristic time originating from a diffusion (1) can be estimated by the following equation,

$$\tau_o \equiv \xi^2/D_\alpha^o, \qquad D_o \equiv k_B T L_\alpha, \tag{1.45}$$

where L_α is the transport coefficient. However, since we should consider a cooperative diffusion (counter diffusion), the time to diffuse by a distance ξ is not τ_o in eq.(1.45) but depends on χ-parameter [3]. We can get mutual diffusion coefficient $D_{\alpha\alpha' coop}$ of two phases $\alpha - \alpha'$ by

$$D_{\alpha\alpha'\ coop} \equiv D_\alpha^o\ \Delta\chi_{\alpha\alpha'}, \tag{1.46}$$

where $1/\chi_{\alpha\alpha'}$ is a susceptibility. This corresponds to the cooperative-diffusion coefficient of the α component when the α' component is a solvent. The fastest mutual-diffusion process is determined by the following equation,

$$D_{coop} = \underset{<\alpha\alpha'>}{\mathrm{Max}}\ D_{\alpha\alpha'\ coop}. \tag{1.47}$$

Therefore, the shortest transportation time τ_D by the cooperative diffusion is estimated by the following equation

$$\tau_D = \xi^2/D_{coop}. \tag{1.48}$$

Next, we estimate characteristic time τ_H of the hydrodynamic transportation (2). If the magnitude of the characteristic flow field is U, τ_H is given as

$$\tau_H = \xi/U. \tag{1.49}$$

The ratio of the above two characteristic times serves as an important quantity. We define this ratio g_0 as

$$g_0 = \tau_D/\tau_H. \tag{1.50}$$

How is the magnitude of a characteristic flow field U determined? The magnitude of a characteristic velocity is determined by the balance between the viscosity term (the second term of the right-hand side of equation (1.41)) and the body force term ($\boldsymbol{K}$of equation (1.41)). However, because of the generality of a simulator, it is impossible to make an exact estimation for a general situation. It will be estimated approximately assuming that the body force comes from an interfacial tension. The viscous force can be estimated to be $\eta L_D^{-2} U$ supposing the scale of spatial change of velocity is the scale of domain size. Here, L_D is the characteristic size of domain. The magnitude of body force coming from an interfacial tension can be estimated from the Laplace's equation. Supposing that the spatial scale of the pressure gradient is the scale of the width of the interface, it is written as $\nabla p \simeq p/\xi$. Supposing that the gradient of the pressure can be written using the interfacial tension γ as $p = \gamma/L_D$ by Laplace's formula, the gradient of the pressure can be estimated as $\gamma/\xi L_D$.

[2] In the estimation here, since the transport by diffusion depends on components, an attention will be paid only to the quickest transportation among them.

[3] Since susceptibility diverges near the critical point, a cooperative-diffusion coefficient becomes remarkably small.

The magnitude of a characteristic velocity is determined by a balance of these two forces as

$$\text{Viscosity} \sim \text{body force(interfacial tension)} \tag{1.51}$$

$$\eta_{min} L_D^{-2} U \sim \frac{\gamma}{\xi L_D} \tag{1.52}$$

where, η_{min} is the smallest viscosity in all the components ($\eta_{min} \equiv \mathrm{Min}\,\eta_\alpha$). For this viscosity, U becomes the maximum. In this estimation, it is assumed that the characteristic scale of spatial change of flow is L_D (= characteristic domain size). From the equation (1.52), the characteristic velocity U can be estimated as

$$U \sim \frac{\gamma}{\bar{\eta}_{min}} \frac{L_D}{\xi} \tag{1.53}$$

The interfacial tension γ is defined as

$$\gamma \simeq \frac{\sqrt{2} k_B T}{a^2} \frac{(\Delta\chi_{\beta\beta'})^{3/2}}{\chi_c^2 \sqrt{N_\beta N_{\beta'} \bar{\psi}_\beta \bar{\psi}_{\beta'}}}, \qquad \chi_c = \frac{1}{2}\Big(\frac{1}{\sqrt{N_b}} + \frac{1}{\sqrt{N_{b'}}}\Big)^2.$$

The scale of the velocity field estimated by the formula (1.53) is not suitable as a unit of a velocity field because it depends on the characteristic domain size. So, $U_o = \gamma/\eta_{min}$ is used as a unit of a velocity field. The velocity field $\tilde{\boldsymbol{v}}$ is scaled as using U_o as

$$\tilde{\boldsymbol{v}} = \frac{\boldsymbol{v}}{U_o}, \qquad U_o \equiv \frac{\gamma}{\eta_{min}}.$$

The dimensionless pressure $\tilde{p}$, the body force $\tilde{\boldsymbol{K}}$, and the viscosity $\tilde{\eta}$ are defined as

$$\tilde{p} = \frac{\xi}{\eta_{max}\, U_o}\, p, \qquad \tilde{\boldsymbol{K}} = \frac{\xi}{e_o}\, \boldsymbol{K}, \qquad \tilde{\boldsymbol{J}} = \frac{\xi}{e_o}\, \boldsymbol{J}, \qquad \tilde{\eta} = \frac{\eta}{\eta_{max}}.$$

Here, η_{max} means the maximum value ($\eta_{max} \equiv \mathrm{Max}\,\eta_\alpha$) in the viscosities for all the components.

After estimating the characteristic time τ_D and τ_H using actual physical constants, it turns out to be $\tau_H \ll \tau_D$ for polymer mixture systems [4]. Then, we use characteristic time τ_H as a unit of time, and this will be written as τ from now on. Using this τ a dimensionless time is defined as

$$\tilde{t} = t/\tau.$$

[**Dimensionless equations**]

Using the spatial and time units, dimensionless equations are obtained as follows.

Basic dimensionless equations 1 : when the inertia of fluid cannot be ignored

$$\frac{\partial \psi_\alpha}{\partial \tilde{t}} = -\tilde{\nabla} \cdot (\tilde{\boldsymbol{v}} \psi_\alpha) - \tilde{D}_\alpha \tilde{\nabla} \cdot \tilde{\boldsymbol{K}}_\alpha \tag{1.54}$$

$$Re \cdot \tilde{\rho} \frac{\partial \tilde{\boldsymbol{v}}}{\partial \tilde{t}} = -\tilde{\nabla}\tilde{p} + \tilde{\nabla} \cdot \big(\tilde{\eta}\tilde{\boldsymbol{D}}\big) + \mathrm{Ca}^{-1}\tilde{\boldsymbol{K}}, \qquad \tilde{\nabla} \cdot \tilde{\boldsymbol{v}} = 0 \tag{1.55}$$

$$\tilde{D}_\alpha \equiv D_\alpha |\Delta\chi| \frac{\tau}{\xi^2}, \qquad \mathrm{Ca}^{-1} = \frac{k_B T}{v_o} \frac{\Delta\chi \xi}{\eta_{max} U_o}, \qquad Re = \frac{\bar{\rho}\, \xi\, U_o}{\eta_{max}} \tag{1.56}$$

Here, $\tilde{D}_\alpha$ is a dimensionless diffusion coefficient. It should be noticed that the Ca in eq.(1.56) is different from the usual capillary number Ca ($Ca \equiv \eta U/\gamma$, U: the magnitude of a characteristic flow). It is also necessary to notice that the number Re is different from the usual Reynolds number. This Re has usually very small value in the polymer system, so the inertia of a fluid can be ignored. Therefore, the equations which should be solved are finally as follows.

[4] When the diffusion coefficient is $D_{coop} = 1.0 \times 10^{-9} \sim 10^{-11}$ cm^2/sec, the interface width $\xi \sim 5$ nm, the interfacial tension 10.0 mN/m, and viscosity 1.0 Pa $\cdot$ sec, $\tau_D \sim 10^{-3}$ sec and $\tau_H \sim 10^{-6}$ sec.

Basic dimensionless equations 1 : when the inertia of fluid can be ignored

$$\frac{\partial \psi_\alpha}{\partial \tilde{t}} = -\tilde{\nabla}\cdot(\tilde{\boldsymbol{v}}\psi_\alpha) - \tilde{D}_\alpha \tilde{\nabla}\cdot\tilde{\boldsymbol{K}}_\alpha \tag{1.57}$$

$$0 = -\tilde{\nabla}\tilde{p} + \tilde{\nabla}\cdot\left(\tilde{\eta}\tilde{\boldsymbol{D}}\right) + \mathrm{Ca}^{-1}\tilde{\boldsymbol{K}} \qquad \tilde{\nabla}\cdot\tilde{\boldsymbol{v}} = 0 \tag{1.58}$$

$$\tilde{D}_\alpha \equiv D_\alpha|\Delta\chi|\frac{\tau}{\xi^2}, \qquad \mathrm{Ca}^{-1} = \frac{k_B T}{v_o}\frac{\Delta\chi\xi}{\eta_{max}U_o} \tag{1.59}$$

1.3.2 Dimensionless expression of interaction term in the free energy

Here, we consider non-dimensional expressions for the parameters which appear in the interaction energy density except for the term related to the χ-parameter. We write terms other than the χ-parameter terms of equation (1.23) as $f_{int}^{(ex)}$

$$f_{int}^{(ex)} = \rho_e(\boldsymbol{r})\Phi(\boldsymbol{r}) - \frac{\epsilon(\boldsymbol{r})}{2}\boldsymbol{E}^2(\boldsymbol{r}). \tag{1.60}$$

As a unit of the electric field, we use E_o=1kV/mm. We can get a dimensionless scalar potential $\tilde{\Phi}(\tilde{\boldsymbol{r}}) = \Phi(\boldsymbol{r})/\phi_o$ using $\phi_o \equiv E_o\xi$. The dielectric constant $\epsilon(\boldsymbol{r})$ is scaled by the maximum value ϵ_{max} among dielectric constants of each component. Using the scales defined so far, the interaction free energy density is turned into dimensionless one

$$\tilde{f}_{int}(\tilde{\boldsymbol{r}}) = \sum_{\alpha}^{M-1}\tilde{\rho}_{e\alpha}\psi_\alpha(\tilde{\boldsymbol{r}})\tilde{\Phi}(\tilde{\boldsymbol{r}}) - \frac{1}{2}B\sum_{\alpha}^{M-1}\tilde{\epsilon}_\alpha\psi_\alpha(\tilde{\boldsymbol{r}})\tilde{\boldsymbol{E}}^2(\tilde{\boldsymbol{r}}), \tag{1.61}$$

$$\tilde{\rho}_{e\alpha} \equiv \frac{e\nu_\alpha Z_\alpha\phi_o}{k_B T}\cdot\frac{1}{|\Delta\chi|}, \qquad B \equiv \frac{\epsilon_{max}E_o^2 v_o}{k_B T}\cdot\frac{1}{|\Delta\chi|}, \tag{1.62}$$

where the coefficient B is the ratio between the characteristic electromagnetic energy in a volume element v_o and the interaction free energy between monomers. The electromagnetic effect becomes dominant as B becomes larger. The $\tilde{\rho}_{e\alpha}$ in eq.(1.62) expresses the ratio of the Coulomb energy and the interaction free energy between monomers. The Coulomb interaction becomes dominant as $\rho_{e\alpha}$ becomes larger.

1.3.3 Dimensionless Poisson equation

When the dielectric constant is scaled by the maximum dielectric constant ϵ_{max}, the electric field is scaled by E_o and the equation (1.20) (1.25) are used, a dimensionless Maxwell equation is obtained as

$$\tilde{\nabla}\cdot\left[\ \tilde{\epsilon}(\tilde{\boldsymbol{r}})\tilde{\nabla}\tilde{\Phi}(\tilde{\boldsymbol{r}})\ \right] = -\sum_{\alpha=0}^{M-1}\tilde{\rho}_{e\alpha}B^{-1}\psi_\alpha(\tilde{\boldsymbol{r}}). \tag{1.63}$$

1.3.4 Actual and dimensionless parameters required for input

To get a value of a dimensionless parameter, at first you have to input the following basic parameters (actual value with a unit) .

[basic parameters (actual value with a unit)]

Notation	Meanings
M	number of components
a	monomer size
N_α	the degree of polymerization of each component($\alpha = 0, \cdots M-1$)
$\chi_{\alpha\alpha'}$	χ-parameter ($_MC_2$ combinations)
η_α	viscosity of each component
D_α	diffusion coefficient of each component
T	temperature
ψ_α	The preparation volume fraction of each component($\alpha = 0, \cdots M-1$)

Using the above values, the spatial unit ξ and the time unit τ can be derived.

[space unit and time unit]

Meanings	Notation	Expression
space unit	ξ	$\xi = \underset{<\alpha,\alpha'>}{\mathrm{Min}} \sqrt{\frac{1}{\Delta\chi_{\alpha\alpha'}} \cdot \frac{a^2}{9\bar{\psi}_\alpha\bar{\psi}_\alpha}}$ $\Delta\chi_{\alpha\alpha'} \equiv \chi_{\alpha\alpha'} - \chi^{(c)}_{\alpha\alpha'}$ $\chi^{(c)}_{\alpha\alpha'} \equiv \frac{(\sqrt{N_\alpha}+\sqrt{N_{\alpha'}})^2}{2N_\alpha N_{\alpha'}}$: critical value
time unit	τ	$\tau = \frac{a^2\xi\eta_{min}}{\sqrt{2}k_BT} \frac{\chi_c^2\sqrt{N_\beta N_{\beta'}\bar{\psi}_\beta\bar{\psi}_{\beta'}}}{(\Delta\chi)^{3/2}}$ When ξ is given for a combination (β, β'), $\Delta\chi$ is $\Delta\chi_{\beta\beta'}$.

[Conversion from an actual to a dimensionless parameter]

Dimensionless parameter	meanings	input and dimensionless parameter	input parameter in MUFFIN
$\tilde{\eta}_\alpha$	viscosity of each components	$\tilde{\eta}_\alpha = \eta_\alpha/\eta_{max}$ $\eta_{\max} \equiv \underset{\alpha}{\mathrm{Max}}\,\eta_\alpha$	`VISCOSITY`
$\tilde{D}_\alpha$	diffusion coefficient	$\tilde{D}_\alpha = D_\alpha \lvert\Delta\chi\rvert \dfrac{\tau}{\xi^2}$	`DIFFUSION_COEFFICIENT`
$\tilde{\dot{\gamma}}$	shear rate	$\tilde{\dot{\gamma}} = \dot{\gamma}\tau$	`SHEAR_RATE`
$\tilde{R}_i$	radius of a droplet	$\tilde{R}_i = R_i/\xi$	`RADIUS_OF_DROPLET`
$\tilde{\rho}_{e\alpha}$	charge density	$\tilde{\rho}_{e\alpha} = \dfrac{\rho_{e\alpha} a^3 E_o \xi}{k_B T} \cdot \dfrac{1}{\Delta\chi}$ $E_o \equiv 1.0$ kV/mm	`CHARGE_DENSITY`
B	electric energy	$B = \dfrac{a^3\ \epsilon_{max}\ E_o^2}{k_B T} \cdot \dfrac{1}{\lvert\Delta\chi\rvert}$	`B`
$\tilde{\epsilon}_\alpha$	relative dielectric const.	$\tilde{\epsilon}_\alpha = \epsilon_\alpha/\epsilon_{max}$ $\epsilon_{max} \equiv \underset{\alpha}{\mathrm{Max}}\,\eta_\alpha$	`DIELECTRIC_CONSTANT`
Ca	–	Ca= $\dfrac{k_B T}{v_o} \dfrac{\Delta\chi\xi}{\eta_{max} U_o}$	`CA`

[What to do before performing simulation by MUFFIN]

1. Survey the values of the basic input variables(parameters) of a system to Simulate
2. Calculate the dimensionless spatial and time units of the simulation from basic variables using a tool for conversion to dimensionless parameters.
3. Estimate Re using a calculation tool from the values of basic input variables, and judge whether the inertia term of the fluid equation is important for the system or not. It serves as a guide to judge whether you should use a module including the inertia term, or that without the inertia term.
4. Consider boundary conditions:
   System size
   Periodic, pseudo-periodic
   Existence of wall(s)
   Lees-Edwards condition
5. Determine fields to be considered.
6. Determine initialization modules to be used for each field.
7. Determine boundary condition modules to be used for each field according to the consideration in 4.
8. Determine time-evolution modules to be used for each field.
9. Determine analysis modules to be used for each field, and a time interval for the simulation.
10. You will get a list of parameters necessary for the simulation, and you can get dimensionless parameters by using, for example, a tool "MuffinMujigen.py" included in the MUFFIN system.

1.4 FDM phase separation simulator PhaseSeparation_FDM

The PhaseSeparation_FDM is a phase separation simulator of multicomponent fluid systems using the finite difference method. The characteristic feature of this simulator are as follows.

- The finite difference method of a three-dimensional Euler picture is used as the calculation method.
- A slow viscous flow (Stokes flow) can be dealt with.
- The target is a multicomponent polymer fluid which is described by the volume fractions of each component.
- Boundary conditions can be given on the six boundaries of a rectangular geometry.

List of selectable fields

Selectable fields	notation
Volume fraction field	ψ_α
Chemical potential field	μ_α
Flux field (without hydrodynamics effect)	$J_{i\alpha} \quad (i = x, y \ or \ z)$
Velocity field	$V_i \quad (i = x, y \ or \ z)$
Pressure field	P
Scalar electric potential field	Φ

The greek index α expresses a component which takes a value of $\alpha = 0, \cdots N_c - 1$ (N_c: the number of components).

1.4.1 Calculation model

The following set of equations is used as a basic model of a multicomponent fluid.

$$\frac{\partial \psi_\alpha}{\partial t} = -\nabla \cdot (\boldsymbol{v}\psi_\alpha) - \nabla \cdot \boldsymbol{J}_\alpha, \tag{1.64}$$

$$\boldsymbol{J}_\alpha = -L_\alpha \psi_\alpha \nabla \mu_\alpha, \tag{1.65}$$

$$-\nabla p + \nabla(\eta\{\nabla \boldsymbol{v} + (\nabla \boldsymbol{v})^t\}) + \boldsymbol{K} = 0, \tag{1.66}$$

$$\boldsymbol{K} = -\sum_\alpha \psi_\alpha \nabla \mu_\alpha, \tag{1.67}$$

$$\nabla \cdot \boldsymbol{v} = 0, \tag{1.68}$$

where $\psi_\alpha(\boldsymbol{r})$ is the volume fraction of component α, L_α is the Onsager coefficient and μ_α is the chemical potential. $\boldsymbol{v}$ is fluid velocity field, p is pressure field and η is viscosity coefficient. $\boldsymbol{K}$ is the body force which works as the driving force of a fluid field. The Stokes equation is used for the equation of fluid velocity field.

1.4.2 The boundary conditions on the wall surfaces for each field

In the multiphase fluid simulator using the finite difference method, the shape of a system is rectangular. Therefore, each field must have boundary conditions for the following six boundaries.

Boundary conditions of volume fraction field ψ_α

Possible boundary conditions for the volume fraction field are as follows.

- **Periodic boundary condition**
  When a periodic boundary condition is applied in x-direction,

  $$\psi_\alpha(x, y, z) = \psi_\alpha(x + L_x, y, z).$$

  The similar condition can be applied for y-direction and z-direction.

- **Biased periodic boundary condition**
  A biased periodic condition is similar to the periodic boundary condition, but at two facing boundaries the value of the field may have a non-zero gap. When this condition is applied for x-direction,

  $$\psi_\alpha(x, y, z) = \psi_\alpha(x + L_x, y, z) + A_x,$$

  where A_x is a value of gap for the x-direction.

- **Wall boundary condition**
  At each of six boundaries of rectangular geometry, a gradient of the field along the direction perpendicular to the boundary can be set to zero.

  $$\boldsymbol{n} \cdot \nabla\psi_\alpha(x, y, z)|_{wall} = 0,$$

  where $|_{wall}$ denotes the value on the wall.

- **Bulk boundary condition**
  At each of six boundaries of rectangular geometry, a constant value can be given for the field. It means that the value of the volume fraction is constant outside of the boundary (bulk).

  $$\psi_\alpha(x, y, z)|_{Boundary} = \mathrm{Constant}_\alpha.$$

Boundary conditions for chemical potential field μ_α

Possible boundary conditions for the chemical potential field are as follows.

- **Periodic boundary condition**
  When the periodic boundary condition is applied in x-direction,

  $$\mu_\alpha(x, y, z) = \mu_\alpha(x + L_x, y, z).$$

  The similar condition can be applied for y-direction and z-direction.

- **Wall boundary condition**
  At each of six boundaries of rectangular geometry, a gradient of the field along the direction perpendicular to the boundary can be set to zero.

  $$\boldsymbol{n} \cdot \nabla\mu_\alpha(x, y, z)|_{wall} = 0,$$

  where $|_{wall}$ denotes the value on the wall.

- **Lees Edwards boundary condition**
  The Lees Edwards boundary condition can be used if a shear is applied to the system. This boundary condition is only applicable for z-direction boundaries in this simulator.

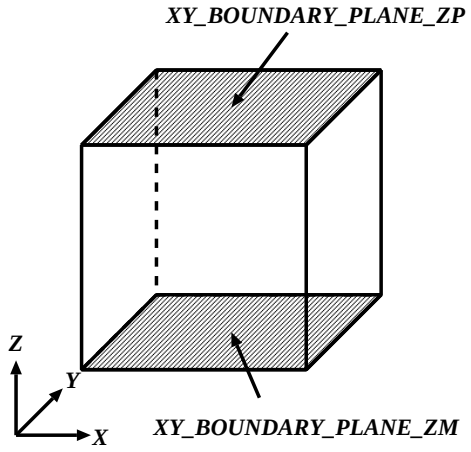

Figure 1.2: two XY plane boundaries perpendicular to Z-axis

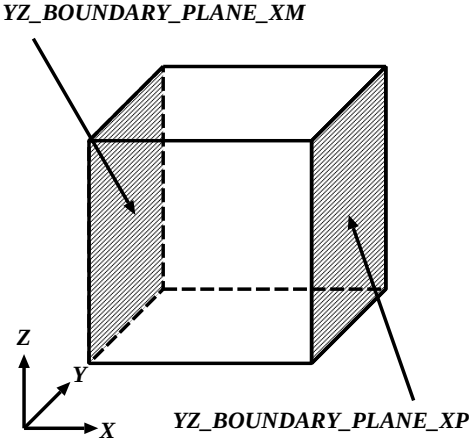

Figure 1.3: two YZ plane boundaries perpendicular to X-axis

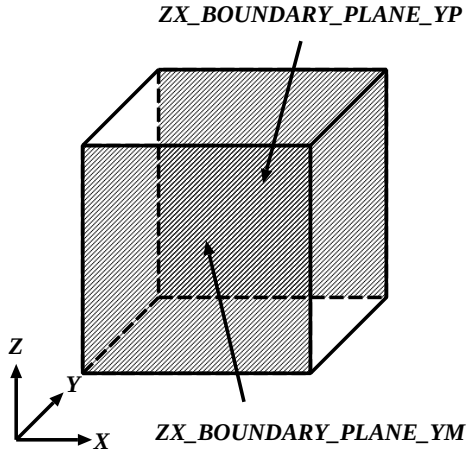

Figure 1.4: two ZX plane boundaries perpendicular to Y-axis

Boundary conditions for flux field $\boldsymbol{K}_\alpha$

Possible boundary conditions for the flux field are as follows.

- **Periodic boundary condition**
  When a periodic boundary condition is applied in x-direction,

  $$K_{\alpha x}(x, y, z) = K_{\alpha x}(x + L_x, y, z).$$

  The similar condition can be applied for y-direction and z-direction.

- **Wall boundary condition**
  At each of six boundaries of rectangular geometry, the flux can be set to zero. It means that the flux by diffusion is zero on the boundary wall.

  $$\boldsymbol{K}_\alpha(x, y, z)|_{wall} = \boldsymbol{0},$$

  where $|_{wall}$ denotes the value on the wall.

- **Bulk boundary condition**
  At each of six boundaries of rectangular geometry, the gradient of the flux in the direction perpendicular to the boundary can be set to zero. It means that the gradient of the flux is zero because of a constant value of the volume fraction field outside of the boundary (bulk).

  $$(\boldsymbol{n} \cdot \nabla)\boldsymbol{K}_\alpha(x, y, z)|_{Boundary} = \boldsymbol{0}.$$

- **Lees Edwards boundary condition**
  The Lees Edwards boundary condition can be used if a shear is applied to the system. This boundary condition is only applicable for z-direction boundaries in this simulator.

Boundary conditions for velocity field $\boldsymbol{v}$

Possible boundary conditions for the velocity field are as follows.

- **Periodic boundary condition**
  When a periodic boundary condition is applied in x-direction,

  $$\boldsymbol{v}(x, y, z) = \boldsymbol{v}(x + L_x, y, z).$$

  The similar condition can be applied for y-direction and z-direction.

- **Constant velocity boundary condition**
  At each of six boundaries of rectangular geometry, the velocity can be set to a constant vector $\boldsymbol{v}_o$. This means that the boundary wall is moving with the constant velocity:

  $$\boldsymbol{v}(x, y, z)|_{wall} = \boldsymbol{v_o},$$

  where, $|_{wall}$ denotes the value on the wall, and $\boldsymbol{v}_o \perp \boldsymbol{n}$.

- **When a constant pressure boundary condition is applied**
  The gradient of the velocity should be set to zero on the boundary:

  $$(\boldsymbol{n} \cdot \nabla)\boldsymbol{v}(x, y, z)|_{wall} = \boldsymbol{0}.$$

- **Lees Edwards boundary condition**
  The Lees Edwards boundary condition can be used if a shear is applied to the system. This boundary condition is only applicable for z-direction boundaries in this simulator.

Boundary conditions for pressure field P

Possible boundary conditions for the pressure field are as follows.

- **Periodic boundary condition**
  When a periodic boundary condition is applied in x-direction,

  $$P(x, y, z) = P(x + L_x, y, z).$$

  The similar condition can be applied for y-direction and z-direction.

- **Biased periodic boundary condition**
  A biased periodic condition is similar to the periodic boundary condition, but at two facing boundaries the value of the field may have a non-zero gap. When this condition is applied for x-direction,

  $$P(x, y, z) = P(x + L_x, y, z) + A_x,$$

  where A_x is a value of gap for the x-direction.

- **When a constant velocity boundary condition is applied**
  The gradient of the pressure should be set to zero on the boundary.

  $$\boldsymbol{n} \cdot \nabla P(x, y, z)|_{Boundary} = 0.$$

- **Constant pressure boundary condition**
  At each of six boundaries of rectangular geometry, the value of the pressure can be set to a constant value P_o.

  $$P(x, y, z)|_{Boundary} = P_o.$$

- **Oscillating pressure on a boundary**
  The pressure oscillates with a constant frequency on the boundary.

  $$P(x, y, z)|_{Boundary} = P_o + \delta P \cdot \sin(\omega t),$$

  where, P_o is a constant pressure, δP is a oscillation amplitude, ω is a frequency and t is the time.

- **Lees Edwards boundary condition**
  The Lees Edwards boundary condition can be used if a shear is applied to the system. This boundary condition is only applicable for z-direction boundaries in this simulator.

Boundary conditions for electric potential field

Possible boundary conditions for the electric potential field are as follows.

- **Periodic boundary condition**
  When a periodic boundary condition is applied in x-direction,

  $$\phi(x, y, z) = \phi(x + L_x, y, z).$$

  The similar condition can be applied for y-direction and z-direction.

- **Biased periodic boundary condition**
  A biased periodic condition is similar to the periodic boundary condition, but at two facing boundaries the value of the field may have a non-zero gap. When this condition is applied for x-direction,

  $$\phi(x, y, z) = \phi(x + L_x, y, z) + A_x,$$

  where A_x is a value gap for the x-direction.

- **A surface charge density is given on a boundary(Neumann boundary condition)**
  A constant surface charge on a boundary can be given by the gradient of the electric potential.

$$\boldsymbol{n} \cdot \nabla\phi(x, y, z)|_{Boundary} = -\sigma/\epsilon_r\epsilon_o.$$

  In a dimensionless form,

$$\boldsymbol{n} \cdot \nabla\phi(x, y, z)|_{Boundary} = -q/\mathsf{R},$$

  where the definition of dimensionless parameters are given by equations (**??**), and (**??**).

- **Constant potential on a boundary (Dirichlet boundary condition)**
  At each of six boundaries of rectangular geometry, the electric potential can be set to a constant value ϕ_o.

$$\phi(x, y, z)|_{Boundary} = \phi_o \qquad (= \text{Constant}).$$

- **Oscillating electric potential on a boundary**
  The electric potential oscillates with a constant frequency on the boundary.

$$\phi(x, y, z)|_{Boundary} = \phi_o + \delta\phi \cdot \sin(\omega t),$$

  where ϕ_o is a constant potential, δP is a oscillation amplitude, ω is a frequency, and t is the time.

- **Lees Edwards boundary condition**
  The Lees Edwards boundary condition can be used if a shear is applied to the system. This boundary condition is only applicable for z-direction boundaries in this simulator.

1.5 FEM phase separation simulator PhaseSeparation_FEM

The PhaseSeparation_FEM is a phase separation simulator of multicomponent fluid systems using the finite element method. The characteristic feature of this simulator are as follows.

- The finite element method of a three-dimensional Euler picture is used as the calculation method. The supported element type is tetrahedral element with a linear interpolation shape function.
- A slow viscous flow (Stokes flow) can be dealt with.
- The target is multicomponent polymer fluid which is described by the volume fractions of each component.
- The Dirichlet condition, the Neumann condition and a periodic boundary condition can be used as a boundary condition (periodic boundary condition is available only for rectangular geometry).

Compared with the simulator using the finite difference method, this FEM simulator has more flexibility in geometry, and, in general, boundary condition treatment is easier. On the other hand, it requires a longer computation time and a larger memory compared to the FDM simulator.

List of selectable fields

Selectable fields	notation
Volume fraction field	ψ_α
Chemical potential field	μ_α
Flux field (without hydrodynamics effect)	$J_{i\alpha} \quad (i = x, y \ or \ z)$
Velocity field	$V_i \quad (i = x, y \ or \ z)$
Pressure field	P
Scalar electric potential field	Φ

The greek index α expresses a component which takes a value of $\alpha = 0, \cdots N_c - 1$ (N_c : the number of components).

1.5.1 Calculation model

The basic calculation model of this simulator is almost similar to that of the FDM simulator. The following set of equations is used as a basic model of a multicomponent fluid,

$$\frac{\partial \psi_\alpha}{\partial t} = -\nabla \cdot (\boldsymbol{v}\psi_\alpha) - \nabla \cdot L_\alpha \boldsymbol{K}_\alpha, \tag{1.69}$$

$$\boldsymbol{K}_\alpha = -\psi_\alpha \nabla \mu_\alpha, \tag{1.70}$$

$$\boldsymbol{J}_\alpha \equiv L_\alpha \boldsymbol{K}_\alpha, \tag{1.71}$$

$$Re\rho \frac{\partial \boldsymbol{v}}{\partial t} = -\nabla p + \nabla(\eta\{\nabla \boldsymbol{v} + (\nabla \boldsymbol{v})^t\}) + \boldsymbol{K}, \tag{1.72}$$

$$\boldsymbol{K} = \sum_\alpha \boldsymbol{K}_\alpha + \boldsymbol{F}_{ext}, \tag{1.73}$$

$$\nabla \cdot \boldsymbol{v} = 0. \tag{1.74}$$

where $\psi_\alpha(\boldsymbol{r})$ shows the volume fraction of each component α, L_α is the Onsager coefficient, and μ_α is the chemical potential of each component. $\boldsymbol{v}$ is the fluid flow velocity, p is the pressure field, η is the the viscosity coefficient, and $\boldsymbol{K}$ is the driving force of a fluid flow composed of $\boldsymbol{K}_\alpha$ which comes from a gradient of the volume fraction and the chemical potential of each component, and $\boldsymbol{F}_{ext}$ is an external force. The equation for the flow field is the Navier Stokes equation without the convection term. In this simulator, we use mainly the Stokes approximation ignoring inertia term $\partial \boldsymbol{v}/\partial t = 0$. Please refer to the chapter 1 about the detail of meanings of the equations. Dimensionless equations are the same as those of FDM simulator described in section 1.3.

1.5.2 Flow field calculation method in FEM simulator

As a calculation method for the fluid flow by FEM, many methods have been developed . We adopted for MUFFIN's FEM simulators a method called "velocity correction method" which is a kind of multi-step method and considered to be suited to large scale three-dimensional simulations. In this method, we discretize the Navier-Stokes equation in time, as follows (consider a dimensionless equation of $\rho = 1$)

$$\frac{\boldsymbol{v}^{(n+1)} - \boldsymbol{v}^{(n)}}{\Delta t/Re} = -\nabla p^{(n+1)} + \nabla\eta(\nabla\boldsymbol{v}^{(n)} + (\nabla\boldsymbol{v}^{(n)})^t) + \boldsymbol{K}^{(n+1)}, \tag{1.75}$$

where variables with (n) is for the current time step (known value) and those with $(n+1)$ are for the next step to be calculated. We write $\Delta t/Re$ as Δt in the following.

By applying the divergence operator to this equation and considering an incompressibility condition $\nabla \cdot \boldsymbol{v} = 0$, we get a Poisson equation for the pressure.

$$\boldsymbol{v}^* = \boldsymbol{v}^{(n)} + \Delta t \left\{ \nabla\eta(\nabla\boldsymbol{v}^{(n)} + (\nabla\boldsymbol{v}^{(n)})^t) + \boldsymbol{K}^{(n+1)} \right\}, \tag{1.76}$$

$$\nabla^2 p^{(n+1)} = \frac{1}{\Delta t}\nabla \cdot \boldsymbol{v}^*. \tag{1.77}$$

The variable $\boldsymbol{v}^*$ in these equations can be called as an "intermediate velocity". The pressure field is calculated using this intermediate velocity. Substituting the calculated pressure to equation (1.75), we get the velocity of the next time step as

$$\boldsymbol{v}^{(n+1)} = \boldsymbol{v}^* - \Delta t\nabla p^{(n+1)}. \tag{1.78}$$

This method is called as "velocity correction method" because the intermediate velocity is corrected by the gradient of the pressure in eq.(1.78).

We perform FEM discretization in space by applying the weighted residual method for equations (1.76),(1.77) and (1.78). In our FEM program, the velocity, the pressure and the force fields are approximated by trial functions which are linear combinations of a first order interpolation function $L_I(\boldsymbol{r})$ whose value on the FEM nodes I are unity and zero on nodes other than node I:

$$\boldsymbol{v}(\boldsymbol{r}) = \sum_I L_I(\boldsymbol{r})\boldsymbol{v}_I, \tag{1.79}$$

$$p(\boldsymbol{r}) = \sum_I L_I(\boldsymbol{r})p_I, \tag{1.80}$$

$$\boldsymbol{K}(\boldsymbol{r}) = \sum_I L_I(\boldsymbol{r})\boldsymbol{K}_I. \tag{1.81}$$

We use a lumped mass matrix on each node for the intermediate velocity calculation (1.76) and the velocity correction (1.78) to enable an explicit time evolution scheme. On the other hand, the calculation of the Poisson equation (1.77) for the pressure needs an implicit method.

As a calculation method of fluid flow, there are "direct" methods which solve velocity field and pressure field equations at once with an implicit solution procedure. In such methods, however, we must solve linear equations of a degree of freedom at least four times of the number of FEM nodes. Moreover, to get solution in such methods, we actually need to add more nodes for velocity field. In the velocity correction method, the implicitly solved variable is only a scalar field p, so we can expect that in large scale problems the velocity correction method requires less memory and calculation time compared to the direct methods.

A demerit of the velocity correction method is that we must use a fine time mesh to avoid a divergence of the calculation, because we are using an explicit time evolution other than for the pressure field. In the fluid flow calculation, a generally required condition for the time mesh value for the calculation stability is the CFL (Courant, Friedrich, Levy) condition as

$$\Delta t \leq \frac{h}{3U}, \tag{1.82}$$

where h is the minimum spatial mesh width and U is the absolute value of the velocity. This condition must be satisfied on all calculation nodes.

The CFL condition is, however, necessary when the convection term is dominant. Because our model neglects the convection term, a more important condition for time step is given by the following equation

which is determined by viscosity coefficient relating to the diffusion of momentum of flow field (viscosity stress):

$$\Delta t \leq \frac{h^2}{9\eta}. \tag{1.83}$$

1.5.3 The Stokes flow calculation

In the Stokes flow, the inertia term $\partial \boldsymbol{v}/\partial t$ is zero. We use the time evolution procedure iteratively until the velocity field converges so that the Stokes flow is attained. In this case the time step Δt is the one for "imaginary" time for iterations to get a stationary state and it is not the physical one. In our FEM simulator, the time interval for the Stokes flow calculation is given as UDF input parameter "DT_FOR_V" and the real time step which is used for time evolution of fields other than the velocity and the pressure is given as input parameter "DT".

1.5.4 Relationship between the FEM discretization and a boundary condition

The finite element method has a characteristic feature in the treatment of the Neumann boundary condition which gives a component of a gradient of a physical quantity normal to a boundary: $\boldsymbol{n} \cdot \nabla f$ where $\boldsymbol{n}$ is the outward normal vector of a boundary surface. The feature is;

Neumann condition

No need to explicitly give the Neumann condition of a type $\boldsymbol{n} \cdot \nabla f = 0$ (natural boundary condition).

For example, we consider the Poisson equation which appears in the calculation of the pressure and the calculation of the electric potential:

$$\nabla^2 f = \sigma. \tag{1.84}$$

In the discretization by the weighted residual method adopted by this simulator, we impose a condition that the residual of equation which is integrated with arbitrary weight function $W(r)$ becomes zero (solution by weak form):

$$\int d\boldsymbol{r}(\nabla^2 f(\boldsymbol{r}) - \sigma(\boldsymbol{r}))W(\boldsymbol{r}) = 0. \tag{1.85}$$

In the weighted residual method by Galerkin-Ritz type, the weight function $W(r)$ is interpolated by the same interpolation function $L_I(\boldsymbol{r})$ as that for physical quantity $f(\boldsymbol{r})$:

$$f(\boldsymbol{r}) = \sum_I L_I(\boldsymbol{r}) f_I, \tag{1.86}$$

$$W(\boldsymbol{r}) = \sum_I L_I(\boldsymbol{r}) W_I. \tag{1.87}$$

The Laplacian term for f in eq. (1.85) is decomposed by partial integration into a volume and a surface integration terms as

$$\int d\boldsymbol{r}(\nabla^2 f(\boldsymbol{r}) - \sigma(\boldsymbol{r}))W(\boldsymbol{r}) = -\int d\boldsymbol{r}\nabla f \nabla W - \int d\boldsymbol{r}\sigma(\boldsymbol{r})W(\boldsymbol{r}) + \int dSW(\boldsymbol{n} \cdot \nabla f) = 0. \tag{1.88}$$

When we express terms including f in the equation above using the interpolation equation in each FEM element e, we get:

$$\int_e d\boldsymbol{r}(\nabla^2 f(\boldsymbol{r}))W(\boldsymbol{r}) = -\int_e d\boldsymbol{r}(\sum_J f_J \nabla L_J)(\sum_I W_I \nabla L_I) + \int_e dS(\sum_I L_I(\boldsymbol{r})W_I)(\boldsymbol{n} \cdot \nabla f). \tag{1.89}$$

From the continuity of the weighting function W, surface integration terms on element surfaces for adjacent elements cancel out in eq.(1.88), so the surface integration term in eq.(1.89) remains only on the system boundary. Requiring that eq.(1.89) should be fulfilled for any value of W_I on all nodes I, we get a linear equation f_I as

$$\sum_J f_J \left[\int_e d\boldsymbol{r}\nabla L_J \nabla L_I\right] = -\int_e d\boldsymbol{r}\sigma L_I + \int_e dS L_I(\boldsymbol{n} \cdot \nabla f). \tag{1.90}$$

The complete linear equation is obtained by a summation of both sides of eq.(1.90) for all elements including node I.

The explanation so far shows that the treatment of the Neumann condition is to calculate the surface integration term. So, in the case that $\boldsymbol{n} \cdot \nabla f = 0$ is required, apparently we need not calculate the surface integration term because it should be zero. For some field variables in our FEM programs, only the Neumann boundary condition of type $\boldsymbol{n} \cdot \nabla f = 0$ is possible, and for such a variable, the default boundary condition is the Neumann condition of this type automatically.

The Neumann boundary condition can also be applied for equations other than Poisson equations. Refer to the detailed explanation for boundary conditions of each field in the following subsection.

1.5.5 Boundary condition specification by the partial region condition

In FEM simulators, we can calculate in any geometry which can be expressed by FEM mesh. So, except for the mesh type "UNSTRUCTURED_RECT", we can not take a way to give a pair of X,Y or Z boundaries and a pair of boundary conditions as used in FDM simulator. In the FEM simulators, boundary conditions are specified by "partial region condition"s defined in input UDF of MUFFIN (UDF data path name is "region_condition[]").

A partial region condition is a combination of "partial region" which expresses a part of space, and a condition for a field defined in the region. For example, when you want to give a boundary condition of a constant velocity 1.0 in the X-direction on the Y-direction boundary, you should give a partial region condition as follows.

- Name of partial region : BOUNDARY_VERTEX_YMAX
- Name of field : Velocity
- Symbol of condition: D_VX (make X-component of a vector field to a constant value – Dirichlet condition)
- Array of values corresponding to the condition: Here a numerical value string is given.

An UDF data for this partial region condition are as follows,

```
{ "YMAX_Vx" "BOUNDARY_VERTEX_YMAX" "Velocity" "D_VX" [ "1.0" ] }
{ "YMAX_Vy" "BOUNDARY_VERTEX_YMAX" "Velocity" "D_VY" [ "0" ] }
{ "YMAX_Vz" "BOUNDARY_VERTEX_YMAX" "Velocity" "D_VZ" [ "0" ] }
```

The string "`YMAX_Vx`" etc. is a KEY data of a partial region condition, and it can be specified arbitrarily by users.

"`BOUNDARY_VERTEX_YMAX`" is a partial region name. In this case a partial region created automatically when mesh type (UDF data path name is `parameter.mesh_parameter.type`) is "`UNSTRUCTURED_RECT`" (unstructured rectangular-shaped region), and it is one of a Y-direction boundaries (boundary having larger y-coordinate value). Other than internally defined partial regions, user can make partial regions by themselves (UDF data path name is mesh.partial_region[]).

"`Velocity`" is the name of the velocity field. "`D_VX`","`D_VY`" and "`D_VZ`" mean Dirichlet conditions for X,Y and Z component of the velocity. This condition fixes the value of the field at nodes (vertices) on the specified partial region. In our FEM simulator, condition symbols starting with "D" are Dirichlet conditions, and those starting with "N" are Neumann conditions.

The last input items "`["1.0"]`" and "`["0"]`" set values for Dirichlet conditions for each velocity component. Some other conditions need more string data to give the additional information.

You should take care that you can give the boundary condition for each of X,Y,Z component of velocity separately. The reason why we have not provided a partial region condition for a vector is to make it possible to set only a component perpendicular to a wall to zero without imposing any constraints for other components ("slipping" flow on a wall). In the region conditions above, if you want to give slipping flow on "`YMAX`" boundary, you must give only the following condition, and do not give any conditions for X and Z components:

```
{ "YMAX_Vy" "BOUNDARY_VERTEX_YMAX" "Velocity" "D_VY" [ "0" ] }
```

1.5.6 Application of a partial region condition other than a boundary condition

Although the main purpose of a partial region condition is to give a boundary condition on a boundary surface, it can also be used to setup a condition as the original purpose – "setting a condition to a specified space region." – as described in this section.

[Set an initial value in a part of space]

You can use a partial region condition to set a constant value on node groups specified as a partial region. For example, you can give to the volume fraction field (VolumeFraction) in a partial region "I_CONSTANT_VALUE_FOR_A_COMPONENT":

```
{ "phi1" "partial_region_A" "VolumeFraction"
  "I_CONSTANT_VALUE_FOR_A_COMPONENT" [ "1" "0.65" ] }
```

For this partial region condition data, the value of the filed "VolumeFraction" of a specified component is set to a specified one in a partial region whose KEY is "partial_region_A". "`[ "1" "0.65" ]`" means the value of component 1 is set to 0.65.

In general, the condition name starting with "I" is used for initialization of a field, and used only in the procedures used for initialization. In the case of VolumeFraction,
"`INITIALIZE_BY_PARTIAL_REGION_CONDITION`" is an example of a procedure (command) for initialization.

[Set a constant value to be fixed in a part of space]

The Dirichlet boundary condition fixes a field value to a specified one anytime on a boundary. It is possible, however, to set a value for internal points to be constant in time. Conditions whose names start with "D" can be used on a partial region condition including internal points.

A typical application of such a condition is to put an "obstacle" in a flow field by setting the value of velocity to zero in an internal partial region.

1.5.7 Boundary conditions in PhaseSeparation_FEM

You can use the following type of boundary conditions (partial region conditions) in the PhaseSeparation_FEM simulator.

- Periodic boundary condition:

  This condition is possible only when an "UNSTRUCTURED_RECT" type of mesh is specified with geometrical periodic boundary. In this case periodic boundaries are treated "geometrically" for all fields, so that the fields must satisfy the periodic boundary condition. So you need not specify the periodic condition explicitly as the partial region conditions in input UDF, or, in other word, you can't use boundary conditions other than the periodic condition on geometrical periodic boundaries.

- Dirichlet condition:

  Set a constant value in time for a partial region.

- Neumann condition:

  A component of the gradient vector of a field, or a vector field itself, normal to a boundary surface is given as a constant value. As explained in the section 1.5.4, when the value of the normal component is zero, you need not specify any boundary conditions in input UDF for some fields.

The FEM simulator does not support the "biased periodic condition" and the Lees Edwards boundary condition supported in the FDM simulators.

Boundary conditions of volume fraction field ψ_α

Possible boundary conditions for the volume fraction field are as follows.

- **Periodic boundary condition**
  This condition is applied when the mesh type is "UNSTRUCTURED_RECT" and the mesh has a periodic boundary. When a periodic boundary is set in X direction, the condition requires:

  $$\psi_\alpha(x, y, z) = \psi_\alpha(x + L_x, y, z).$$

  When a periodic condition is used for Y or Z direction, the similar condition is applied for each direction. In our FEM simulator, a periodic boundary condition is required by geometry, so you need not, or cannot, specify any periodic boundary condition as a partial region condition in input UDF.

- **Wall boundary condition** (Neumann condition)
  This condition sets the normal component of the diffusion flux $\boldsymbol{J}_\alpha$ to zero.

  $$\boldsymbol{n} \cdot \boldsymbol{J}_\alpha(x, y, z)|_{wall} = 0.$$

  It also means;

  $$\boldsymbol{n} \cdot \nabla\psi_\alpha(x, y, z)|_{wall} = 0,$$

  where $|_{wall}$ indicates a field value on the wall. When the boundary is not a geometrical periodic boundary, and if no boundary condition is given for ψ_α, this wall condition is applied as default.

- **Bulk boundary condition** (Dirichlet condition)
  This condition means that the value of the volume fraction is constant outside of the boundary (bulk). A constant value is given for the field on a boundary surface.

  $$\psi_\alpha(x, y, z)|_{Boundary} = \text{Constant}.$$

Boundary conditions for chemical potential field

Possible boundary conditions for the chemical potential field are as follows.

- **Periodic boundary condition**
  This condition is applied when the mesh type is "UNSTRUCTURED_RECT" and the mesh has a periodic boundary. When a periodic boundary condition is set in X direction, the condition requires:

  $$\mu_\alpha(x, y, z) = \mu_\alpha(x + L_x, y, z).$$

  When a periodic condition is used for Y or Z direction, the similar condition is applied for each direction. The periodic boundary condition is required by geometry, so you need not, or cannot, specify any periodic boundary condition as a partial region condition in input UDF.

Boundary conditions for flux field $\boldsymbol{K}_\alpha$

Possible boundary conditions for the flux field are as follows.

- **Periodic boundary condition**
  This condition is applied when the mesh type is "UNSTRUCTURED_RECT" and the mesh has a periodic boundary. When a periodic boundary is set in X direction, the condition requires:

  $$\boldsymbol{K}_\alpha(x, y, z) = \boldsymbol{K}_\alpha(x + L_x, y, z).$$

  When a periodic condition is used for Y or Z direction, the similar condition is applied for each direction. The periodic boundary condition is required by geometry, so you need not, or cannot, specify any periodic boundary condition as a partial region condition in input UDF.

- **Wall boundary condition**
  This condition sets the normal component of the flux $\boldsymbol{J}_\alpha$ to be zero.

  $$\boldsymbol{n} \cdot \boldsymbol{J}_\alpha(x, y, z)|_{wall} = 0,$$

  where $|_{wall}$ indicates a field value on the wall. This condition is identical to the wall boundary condition for ψ_α.

  When the boundary is not a geometrical periodic boundary, and if no boundary condition is given for $\boldsymbol{K}_\alpha$, this wall condition is applied as default.

Boundary conditions for velocity field $\boldsymbol{v}$

Possible boundary conditions for the velocity field are as follows.

- **Periodic boundary condition**
  This condition is applied when the mesh type is "UNSTRUCTURED_RECT" and the mesh has a periodic boundary. When a periodic boundary is set in X direction, the condition requires:

  $$\boldsymbol{v}(x, y, z) = \boldsymbol{v}(x + L_x, y, z).$$

  When a periodic condition is used for Y or Z direction, the similar condition is applied for each direction. The periodic boundary condition is required by geometry, so you need not, or cannot, specify any periodic boundary condition as a partial region condition in input UDF.

- **Constant velocity boundary condition**
  On a boundary (or generally in a partial region), the velocity can be set to a constant value $\boldsymbol{v}_o$.

  $$\boldsymbol{v}(x, y, z)|_{wall} = \boldsymbol{v_o},$$

  where $|_{wall}$ denotes the value on the wall.

  It can be used for a case that the boundary wall is moving with a constant velocity (including the case of fixed boundary $\boldsymbol{v_o} = 0$). As described in the section 1.5.5, it is possible to make "obstacle" in a flow by imposing the velocity to be a constant value for an internal partial region.

Boundary conditions for pressure field p

Possible boundary conditions for the pressure field are as follows.

- **Periodic boundary condition**
  This condition is applied when the mesh type is "UNSTRUCTURED_RECT" and the mesh has a periodic boundary. When a periodic boundary is set in X direction, the condition requires:

  $$P(x, y, z) = P(x + L_x, y, z).$$

  When a periodic condition is used for Y or Z direction, the similar condition is applied for each direction. The periodic boundary condition is required by geometry, so you need not, or cannot, specify any periodic boundary condition as a partial region condition in input UDF.

- **Constant pressure boundary condition**(Dirichlet condition)
  On a boundary surface (or generally in a partial region), the pressure can be set to a constant value P_o.

  $$P(x, y, z)|_{Boundary} = P_o.$$

Boundary conditions for electric potential field

Possible boundary conditions for the electric potential field are as follows.

- **Periodic boundary condition**
  Applied when the mesh type is "UNSTRUCTURED_RECT" and the mesh has a periodic boundary. When a periodic boundary is set in the X direction, the condition requires:

  $$\phi(x, y, z) = \phi(x + L_x, y, z).$$

  When a periodic condition is used for Y or Z direction, the similar condition is applied for each direction. The periodic boundary condition is required by geometry, so you need not, or cannot, specify any periodic boundary condition as a partial region condition in input UDF.

- **A surface charge density is given on a boundary (Neumann boundary condition)**
  A constant surface charge on a boundary can be given by the gradient of electric potential.

  $$\boldsymbol{n} \cdot \nabla\phi(x, y, z)|_{Boundary} = -\sigma/\epsilon_r\epsilon_o.$$

  In dimensionless expression,

  $$\boldsymbol{n} \cdot \nabla\phi(x, y, z)|_{Boundary} = -q/\mathsf{R},$$

  where definition of dimensionless parameters are given by equations (**??**), and (**??**).

- **Constant potential on a boundary (Dirichlet boundary condition)**
  On a boundary surface (or generally in a partial region), the electric potential can be set to a constant value ϕ_o.

  $$\phi(x, y, z)|_{Boundary} = \phi_o \qquad (= \text{Constant}).$$

Chapter 2

Sample problems of PhaseSeparation

2.1 Sample problems of PhaseSeparation_FDM

This section shows the applications for phase separation dynamics simulator - PhaseSeparation_FDM - using the finite difference method.

Input UDF files and output files corresponding to these applications are prepared in a directory of the Muffin distribution as a sub-directory according to the problem.

2.1.1 Application 0: Flory-Huggins model phase separation(1)

This is a two-dimensional phase separation simulation with 32x32 mesh (xz-plane), using the Flory-Huggins model equation. The simulation does not include the effect of hydrodynamics. Periodic boundary conditions are applied for all directions.

1. Mesh parameters:
   Set the parameter.mesh_parameter.type to "SIMPLERECTANGULAR".
   In order to create two-dimensional 32x32 meshes input
   parameter.mesh_parameter.axes[] as follows

axes[]	values[]	Input
[0]	[0]	0.0
[0]	[1]	31.0
[0]	[2]	31.0
[1]	[0]	0.0
[1]	[1]	0.0
[1]	[2]	0.0
[2]	[0]	0.0
[2]	[1]	31.0
[2]	[2]	31.0

2. Periodic boundary condition:
   In our FDM simulators, mesh data do not include periodic condition, so
   parameter.mesh_parameter.periodic[] must be input values 0 and, and periodic condition should be specified for each field.

3. Structured mesh: In order to use structured mesh in the FDM program, give
   parameter.mesh_parameter.index_rule[] values 2, 1 and 0.

4. Solver parameter: It is not necessary to input this parameter for this case.

5. Common physical parameter:
   Input DT=1.0e-3, FINAL_STEP=100000,
   INTERVAL_OF_MONITORING=10 and INTERVAL_OF_UDF_OUTPUT=5000.

6. Physical parameter: Input physical constants shown as follows.

Parameter	input value
NUMBER_OF_COMPONENTS	2
POLYMERIZATION_INDEX_N	1, 1
AVERAGED_VOLUME_FRACTION	0.5, 0.5
DEVIATION_FROM_AVERAGED_VOLUME_FRACTION	0.01
SEED_OF_RANDOM_NUMBER	715
DIFFUSION_COEFFICIENT	1.0, 1.0
CHI_01	3.0

7. Field: Remain as a blank.

8. region_condition[]: For each field, input the boundary conditions as follows,

name_of_region	name_of_target	name_of_condition
YZ_BOUNDARY_PLANE_XM_AND_XP	K_Field	PERIODIC
ZX_BOUNDARY_PLANE_YM_AND_YP	K_Field	PERIODIC
XY_BOUNDARY_PLANE_ZM_AND_ZP	K_Field	PERIODIC
YZ_BOUNDARY_PLANE_XM_AND_XP	VolumeFraction	PERIODIC
ZX_BOUNDARY_PLANE_YM_AND_YP	VolumeFraction	PERIODIC
XY_BOUNDARY_PLANE_ZM_AND_ZP	VolumeFraction	PERIODIC
YZ_BOUNDARY_PLANE_XM_AND_XP	ChemicalPotential	PERIODIC
ZX_BOUNDARY_PLANE_YM_AND_YP	ChemicalPotential	PERIODIC
XY_BOUNDARY_PLANE_ZM_AND_ZP	ChemicalPotential	PERIODIC

9. Field[]: Register fields as follows.

registered_field	type	name_of_region	num_of_component	io_flag
VolumeFraction	Scalar	ALL_VERTEX	$(Number_of_Components)	1
ChemicalPotential	Scalar	ALL_VERTEX	$(Number_of_Components)	0
K_Field	Vector	ALL_EDGE	$(Number_of_Components) $(*)$(3)	0
ElectricPotential	Scalar	ALL_VERTEX	1	0
Pressure	Scalar	ALL_VERTEX	1	0
Velocity	Vector	ALL_EDGE	3	0

10. procedures_table_for_initialization[]:
Set an initialization procedure group name to be "SINGLE_PHASE".
Apply a command "CONSTANT_VOLUME_FRACTION_WITH_NOISE" for the field VolumeFraction.

11. procedures_table_for_evolution[]:
Set an evolution procedure name to "PHASE_SEPARATION_01".
Apply commands for fields as,
"FLORY_HUGGINS" for the field ChemicalPotential,
"GRADIENT_CHEMICAL_POTENTIAL" for the field K_Field, and
"SOLVE_EQUATION_OF_CONTINUITY_WITHOUT_FLOW" for the field VolumeFraction.

12. Calculation.

13. After a calculation, read the output UDF into GOURMET, show the VolumeFraction, and observe the phase separation process.

2.1.2 Application 1: Flory-Huggins model phase separation(2)

This example has similar conditions with Ex.00, but in this case there are "walls" in Z-direction boundaries, and periodic boundary conditions are applied in X-direction

1. Mesh parameters: Set the same as Ex.00.
2. Periodic boundary condition: Set the same as Ex.00.
   All elements of parameter.mesh_parameter.periodic[] should be set to zero.
3. Structured mesh: Set the same as Ex.00.
4. Solver parameters: Set the same as Ex.00.
5. Common physical parameters: Set the same as Ex.00.
6. Physical parameters: Set the same as Ex.00.
7. Field: Set the same as Ex.00.
8. region_condition[]: For each field, input boundary conditions as follows.

name_of_region	name_of_target	name_of_condition
YZ_BOUNDARY_PLANE_XM_AND_XP	K_Field	PERIODIC
ZX_BOUNDARY_PLANE_YM_AND_YP	K_Field	PERIODIC
XY_BOUNDARY_PLANE_ZM_AND_ZP	K_Field	ZM_WALL__ZP_WALL
YZ_BOUNDARY_PLANE_XM_AND_XP	VolumeFraction	PERIODIC
ZX_BOUNDARY_PLANE_YM_AND_YP	VolumeFraction	PERIODIC
XY_BOUNDARY_PLANE_ZM_AND_ZP	VolumeFraction	ZM_WALL__ZP_WALL
YZ_BOUNDARY_PLANE_XM_AND_XP	ChemicalPotential	PERIODIC
ZX_BOUNDARY_PLANE_YM_AND_YP	ChemicalPotential	PERIODIC
XY_BOUNDARY_PLANE_ZM_AND_ZP	ChemicalPotential	ZM_WALL__ZP_WALL

9. The procedures_table_for_initialization[] and procedures_table_for_evolution[] are set the same as Ex.00.
10. Calculation.
11. After a calculation, read the output UDF into GOURMET, show the VolumeFraction, and compare the phase separation process with that of EX.00.

2.1.3 Application 2: Flory-Huggins model phase separation(3)

This example almost has the same conditions with example Ex.01. The difference is that we apply a wettability on Z-direction walls.

1. Mesh parameters: Set the same as Ex.00.
2. Periodic boundary condition: Set the same as Ex.00.
   All elements of parameter.mesh_parameter.periodic[] should be set to zero.
3. Structured mesh: Set the same as Ex.00.
4. Solver parameters: Set the same as Ex.00.
5. Common physical parameters: Set the same as Ex.00.
6. Physical parameters: Input physical constants shown as follows. They are almost the same as Ex.00, but an important parameter is GAMMA_S which gives a wettability to a wall.

Parameter	input value
NUMBER_OF_COMPONENTS	2
POLYMERIZATION_INDEX_N	1, 1
AVERAGED_VOLUME_FRACTION	0.5, 0.5
DEVIATION_FROM_AVERAGED_VOLUME_FRACTION	0.01
SEED_OF_RANDOM_NUMBER	715
DIFFUSION_COEFFICIENT	1.0, 1.0
CHI_01	3.0
GAMMA_S	1.0, 0.0

7. Field: Set the same as Ex.00.

8. region_condition[]: Set the same as Ex.01

9. Field[]: Register the same fields as Ex.00.

10. procedures_table_for_initialization[]: Set the same as Ex.00.

11. procedures_table_for_evolution[]:
Add a field "ChemicalPotential" with name_of_func;
"ADD_EFFECT_OF_WETTING_FOR_UNIFORM_Z_WALL"

12. Calculation.

13. After a calculation, the output UDF into GOURMET, show the VolumeFraction, and compare the phase separation process with those of Ex.00 and Ex.01.

2.1.4 Application 3: Uniform electric field

A uniform electric field under two-dimensional 64x64 mesh geometry in XZ plane can be realized as follows.

1. Mesh parameters:
Set the parameter.mesh_parameter.type to "SIMPLERECTANGULAR". In order to generate a two-dimensional 64x64 meshes, input the parameter.mesh_parameter.axes[] as follows.

axes[]	values[]	input
[0]	[0]	0.0
[0]	[1]	63.0
[0]	[2]	63.0
[1]	[0]	0.0
[1]	[1]	0.0
[1]	[2]	0.0
[2]	[0]	0.0
[2]	[1]	63.0
[2]	[2]	63.0

2. Structured mesh:
In order to use a structured mesh, set the parameter.mesh_parameter.index_rule[] to 2, 1 and 0.

3. Solver parameter: Input parameters for electric field solver.

Solver parameter	input
ACCELERATION_VALUE_FOR_E-POTENTIAL	1.8
MAX_ITERATION_FOR_E-POTENTIAL_SOLVER	1.0e5
CONVERGENCE_CRITERION_FOR_E-POTENTIAL	1.0e-6
MONITORING_INTERVAL_OF_E-POTENTIAL_SOLVER	1000

4. Common physical parameter:
   Since a result can be obtained by one step calculation, there is no meaning of DT, but you have to set the value (e.g. DT=1.0). Input FINAL_STEP=1 and INTERVAL_OF_UDF_OUTPUT=1.

5. Physical parameter:
   Input physical constants shown as follows. All parameters other than ELECTRIC_POTENTIAL_AT_XY_PLANE_ZM and ELECTRIC_POTENTIAL_AT_XY_PLANE_ZP have no meanings for this electric field calculation.

Parameter	input
NUMBER_OF_COMPONENTS	2
AVERAGED_VOLUME_FRACTION	0.5, 0.5
CHARGE_DENSITY	0.0, 0.0
DIELECTRIC_CONSTANT	1.0, 1.0
B	1.0
ELECTRIC_POTENTIAL_AT_XY_PLANE_ZM	0.0
ELECTRIC_POTENTIAL_AT_XY_PLANE_ZP	1.0

6. region_condition[]: For each field, input boundary conditions as follows.
   "XY_BOUNDARY_PLANE_ZM_AND_ZP" for the ElectricPotential makes electric field uniform.

name_of_region	name_of_target	name_of_condition
YZ_BOUNDARY_PLANE_XM_AND_XP	VolumeFraction	PERIODIC
ZX_BOUNDARY_PLANE_YM_AND_YP	VolumeFraction	PERIODIC
XY_BOUNDARY_PLANE_ZM_AND_ZP	VolumeFraction	ZM_WALL__ZP_WALL
YZ_BOUNDARY_PLANE_XM_AND_XP	ElectricPotential	PERIODIC
ZX_BOUNDARY_PLANE_YM_AND_YP	ElectricPotential	PERIODIC
XY_BOUNDARY_PLANE_ZM_AND_ZP	ElectricPotential	ZM_DIRICHLET__ ZP_DIRICHLET

7. Field[]: Register fields. Only the ElectricField and the VolumeFraction are the actual working fields.

registered_field	type	name_of_region	num_of_component	io_flag
VolumeFraction	Scalar	ALL_VERTEX	$(Number_of_Components)	1
ChemicalPotential	Scalar	ALL_VERTEX	$(Number_of_Components)	0
K_Field	Vector	ALL_EDGE	$(Number_of_Components) $(*)$(3)	0
ElectricPotential	Scalar	ALL_VERTEX	1	1
Pressure	Scalar	ALL_VERTEX	1	0
Velocity	Vector	ALL_EDGE	3	0

8. procedures_table_for_initialization[]:
   Set an initialization procedure group name to "SINGLE_PHASE".
   Apply a command "CONSTANT_VOLUME_FRACTION" for the field "VolumeFraction" .

9. procedures_table_for_evolution[]:
   Set an evolution procedure name to "TEST_03".
   Apply a command "ELECTRIC_POTENTIAL_SOLVER" for the ElectricPotential field.

10. Calculation.

11. After a calculation, read an output UDF into GOURMET, view or plot the ElectricPotential filed, and you can see a constant gradient for the ElectricPotential, i.e. constant electric field.

2.1.5 Application 4: Electric field change by dielectric constant distribution(1)

A simulation of electric field depending on a spatial distribution of materials having different dielectric constant is performed under two-dimensional 128x128 meshes on XZ plane.

1. Mesh parameters:
   Set the parameter.mesh_parameter.type to "SIMPLERECTANGULAR". In order to create a two-dimensional 128x128 meshes, input the parameter.mesh_parameter.axes[] as follows.

axes[]	values[]	input
[0]	[0]	0.0
[0]	[1]	127.0
[0]	[2]	127.0
[1]	[0]	0.0
[1]	[1]	0.0
[1]	[2]	0.0
[2]	[0]	0.0
[2]	[1]	127.0
[2]	[2]	127.0

2. Solver parameter: Input parameters for the electric field solver.

Solver parameter	input
ACCELERATION_VALUE_FOR_E-POTENTIAL	1.8
MAX_ITERATION_FOR_E-POTENTIAL_SOLVER	1.0e5
CONVERGENCE_CRITERION_FOR_E-POTENTIAL	1.0e-5
MONITORING_INTERVAL_OF_E-POTENTIAL_SOLVER	1000

3. Common physical parameter:
   This is a one step calculation. Input DT=1.0e-3(no meanings here), FINAL_STEP=1 INTERVAL_OF_MONITORING=1, and INTERVAL_OF_UDF_OUTPUT=1.

4. Physical parameter:
   Input physical constants shown as follows. Important parameters are DIELECTRIC_CONSTANT (dielectric constant for each component), ELECTRIC_POTENTIAL_* (boundary electric potential value to create electric field) and a position and size of a droplet.

Parameter	input
NUMBER_OF_COMPONENTS	2
CHARGE_DENSITY	0.0, 0.0
DIELECTRIC_CONSTANT	1.0, 2.0
B	1.0
ELECTRIC_POTENTIAL_AT_XY_PLANE_ZM	0.0
ELECTRIC_POTENTIAL_AT_XY_PLANE_ZP	1.0
NUMBER_OF_DROPLETS	1
RADIUS_OF_DROPLETS	20.0
X_COORDINATE_OF_DROPLET	64
Y_COORDINATE_OF_DROPLET	1
Z_COORDINATE_OF_DROPLET	64

5. region_condition[]: For each field, input boundary conditions as follows.
   "XY_BOUNDARY_PLANE_ZM_AND_ZP" for ElectricPotential makes a uniform electric field.

name_of_region	name_of_target	name_of_condition
YZ_BOUNDARY_PLANE_XM_AND_XP	K_Field	PERIODIC
ZX_BOUNDARY_PLANE_YM_AND_YP	K_Field	PERIODIC
XY_BOUNDARY_PLANE_ZM_AND_ZP	K_Field	ZM_WALL__ZP_WALL
YZ_BOUNDARY_PLANE_XM_AND_XP	VolumeFraction	PERIODIC
ZX_BOUNDARY_PLANE_YM_AND_YP	VolumeFraction	PERIODIC
XY_BOUNDARY_PLANE_ZM_AND_ZP	VolumeFraction	ZM_WALL__ZP_WALL
YZ_BOUNDARY_PLANE_XM_AND_XP	ChemicalPotential	PERIODIC
ZX_BOUNDARY_PLANE_YM_AND_YP	ChemicalPotential	PERIODIC
XY_BOUNDARY_PLANE_ZM_AND_ZP	ChemicalPotential	ZM_WALL__ZP_WALL
YZ_BOUNDARY_PLANE_XM_AND_XP	ElectricPotential	PERIODIC
ZX_BOUNDARY_PLANE_YM_AND_YP	ElectricPotential	PERIODIC
XY_BOUNDARY_PLANE_ZM_AND_ZP	ElectricPotential	ZM_DIRICHLET__ ZP_DIRICHLET

6. Field[]: Register fields as EX.03. Only the “ElectricField” and the “VolumeFraction” are the actual working fields.

7. procedures_table_for_initialization[]:
   Set an initialization procedure group name to ”SINGLE_PHASE”.
   Apply a command ”SET_DROPLETS” for the field ”VolumeFraction”.

8. procedures_table_for_evolution[]:
   Set an evolution procedure name to be ”TEST_04”.
   Apply a command “ELECTRIC_POTENTIAL_SOLVER” for the ElectricPotential field.

9. Calculation.

2.1.6 Application 5: Electric field change by dielectric constant distribution(2)

In this sample a system similar to the sample EX.04 (mesh size is reduced to 64x64) is simulated, but the a droplet evolves in time and effect of electric field is taken in the time evolution.

1. Mesh parameters:
   Set the parameter.mesh_parameter.type to “SIMPLERECTANGULAR”.
   In order to create a two-dimensional 64x64 meshes input the parameter.mesh_parameter.axes[] as follows;

axes[]	values[]	input
[0]	[0]	0.0
[0]	[1]	63.0
[0]	[2]	63.0
[1]	[0]	0.0
[1]	[1]	0.0
[1]	[2]	0.0
[2]	[0]	0.0
[2]	[1]	63.0
[2]	[2]	63.0

2. Solver parameter: Input the same parameters for the electric field solver as EX.04.

3. Common physical parameter: Input DT=1.0e-3, FINAL_STEP=100000,
   INTERVAL_OF_MONITORING=10, and INTERVAL_OF_UDF_OUTPUT=10000.

4. Physical parameter: Input physical constants shown as follows. All parameters are necessary for this simulation.

Parameter	input
NUMBER_OF_COMPONENTS	2
CHARGE_DENSITY	0.0, 0.0
DIELECTRIC_CONSTANT	1.0, 2.0
B	1.0
ELECTRIC_POTENTIAL_AT_XY_PLANE_ZM	0.0
ELECTRIC_POTENTIAL_AT_XY_PLANE_ZP	64.0
NUMBER_OF_DROPLETS	1
RADIUS_OF_DROPLETS	15.0
X_COORDINATE_OF_DROPLET	32
Y_COORDINATE_OF_DROPLET	1
Z_COORDINATE_OF_DROPLET	32

5. region_condition[]: Set the same as EX.04.

6. Field[]: Set the same as EX.04.

7. procedures_table_for_initialization[]:
   Set an initialization procedure group name to "SINGLE_PHASE".
   Apply a command "SET_DROPLETS" for the field "VolumeFraction".

8. procedures_table_for_evolution[]:
   Set an evolution procedure name to "TEST_05".
   Apply commands for fields as,
   "ELECTRIC_POTENTIAL_SOLVER" for the ElectricPotential,
   "FLORY_HUGGINS" and "ADD_ELECTRIC_EFFECT_OF_DIELECTRIC_MEDIUM"
   for the ChemicalPotential, and
   "GRADIENT_CHEMICAL_POTENTIAL" for the field K_Field and
   "SOLVE_EQUATION_OF_CONTINUITY_WITHOUT_FLOW" for the VolumeFraction.

9. Calculation.

2.1.7 Application 6: Poiseuille flow

This is a simulation of a flow of a two-dimensional(XZ) fluid bounded by two Z-walls, and a pressure gap (difference) is applied in X-direction.

1. Mesh parameters:
   Set the parameter.mesh_parameter.type to "SIMPLERECTANGULAR". In order to create a two-dimensional 64x64 meshes input the parameter.mesh_parameter.axes[] as follows.

axes[]	values[]	input
[0]	[0]	0.0
[0]	[1]	63.0
[0]	[2]	63.0
[1]	[0]	0.0
[1]	[1]	0.0
[1]	[2]	0.0
[2]	[0]	0.0
[2]	[1]	63.0
[2]	[2]	63.0

2. Solver parameter: Input parameters as follows for a pressure and velocity field solver.

Solver parameter	input
ACCELERATION_VALUE_FOR_PRESSURE_SOLVER	1.5
MAX_ITERATION_FOR_PRESSURE_SOLVER	1.0e6
CONVERGENCE_CRITERION_FOR_RRESSURE_SOLVER	1.0e-4
MONITORING_INTERVAL_OF_PRESSURE_SOLVER	100
ACCELERATION_VALUE_FOR_VELOCITY_SOLVER	1.5
MAX_ITERATION_FOR_VELOCITY_SOLVER	1.0e6
CONVERGENCE_CRITERION_FOR_VELOCITY_SOLVER	1.0e-4
MONITORING_INTERVAL_OF_VELOCITY_SOLVER	100
SKIP_INTERVAL_VELOCITY_CALCULATION	1

3. Common physical parameter:
   Input DT=1.0e-3, FINAL_STEP=10,
   INTERVAL_OF_MONITORING=10, and INTERVAL_OF_UDF_OUTPUT=5.

4. Physical parameter: Input physical constants as follows.

Parameter	input
NUMBER_OF_COMPONENTS	1
CA	1
PRESSURE_GRADIENT	-0.01
VISCOSITY	1, 1

5. region_condition[]: For each field, input boundary conditions as follows.

name_of_region	name_of_target	name_of_condition
YZ_BOUNDARY_PLANE_XM_AND_XP	K_Field	PERIODIC
ZX_BOUNDARY_PLANE_YM_AND_YP	K_Field	PERIODIC
XY_BOUNDARY_PLANE_ZM_AND_ZP	K_Field	ZM_WALL__ZP_WALL
YZ_BOUNDARY_PLANE_XM_AND_XP	VolumeFraction	PERIODIC
ZX_BOUNDARY_PLANE_YM_AND_YP	VolumeFraction	PERIODIC
XY_BOUNDARY_PLANE_ZM_AND_ZP	VolumeFraction	ZM_WALL__ZP_WALL
YZ_BOUNDARY_PLANE_XM_AND_XP	Pressure	BIASED_PERIODIC
ZX_BOUNDARY_PLANE_YM_AND_YP	Pressure	PERIODIC
XY_BOUNDARY_PLANE_ZM_AND_ZP	Pressure	ZM_WALL__ZP_WALL
YZ_BOUNDARY_PLANE_XM_AND_XP	Velocity	PERIODIC
ZX_BOUNDARY_PLANE_YM_AND_YP	Velocity	PERIODIC
XY_BOUNDARY_PLANE_ZM_AND_ZP	Velocity	ZM_WALL__ZP_WALL

6. Field[]: Register fields. Only the “Velocity”, “Pressure” and “VolumeFraction” are the actual working fields.

registered_field	type	name_of_region	num_of_component	io_flag
VolumeFraction	Scalar	ALL_VERTEX	$(Number_of_Components)	1
ChemicalPotential	Scalar	ALL_VERTEX	$(Number_of_Components)	0
K_Field	Vector	ALL_EDGE	$(Number_of_Components) $(*)$(3)	0
ElectricPotential	Scalar	ALL_VERTEX	1	1
Pressure	Scalar	ALL_VERTEX	1	1
Velocity	Vector	ALL_EDGE	3	1

7. procedures_table_for_initialization[]:
   Set an initialization procedure group name to ”SINGLE_PHASE”.
   Apply a command ”CONSTANT_VOLUME_FRACTION” for the field ”VolumeFraction”.

8. procedures_table_for_evolution[]:
   Set an evolution procedure name to "POISEUILLE_FLOW".
   Apply a command "SOLVE_STOKES_EQUATION_AND_PRESSURE" for the Velocity field.

9. Calculation.

2.1.8 Application 7: Shear induced flow(1)

This is a simulation of a flow in a two-dimensional(XZ) fluid bounded by two Z-walls. The flow is induced by a movement of walls.

1. Mesh parameters:
   Set the parameter.mesh_parameter.type to "SIMPLERECTANGULAR". In order to make a two-dimensional 64x64 meshes, input the parameter.mesh_parameter.axes[] as follows.

axes[]	values[]	input
[0]	[0]	0.0
[0]	[1]	63.0
[0]	[2]	63.0
[1]	[0]	0.0
[1]	[1]	0.0
[1]	[2]	0.0
[2]	[0]	0.0
[2]	[1]	63.0
[2]	[2]	63.0

2. Solver parameter: Input parameters for a pressure and velocity field solver as follows.

Solver parameter	input
LEES_EDWARDS_BC	0
ACCELERATION_VALUE_FOR_PRESSURE_SOLVER	1.5
MAX_ITERATION_FOR_PRESSURE_SOLVER	1.0e6
CONVERGENCE_CRITERION_FOR_RRESSURE_SOLVER	1.0e-4
MONITORING_INTERVAL_OF_PRESSURE_SOLVER	100
ACCELERATION_VALUE_FOR_VELOCITY_SOLVER	1.5
MAX_ITERATION_FOR_VELOCITY_SOLVER	1.0e6
CONVERGENCE_CRITERION_FOR_VELOCITY_SOLVER	1.0e-4
MONITORING_INTERVAL_OF_VELOCITY_SOLVER	100
SKIP_INTERVAL_VELOCITY_CALCULATION	1

3. Common physical parameter:
   Input DT=1.0e-3, FINAL_STEP=10,
   INTERVAL_OF_MONITORING=1, and INTERVAL_OF_UDF_OUTPUT=5.

4. Physical parameter: Input physical constants as follows.

Parameter	input
NUMBER_OF_COMPONENTS	1
CA	1
VISCOSITY	1, 1
VX_AT_XY_PLANE_ZM	-1
VX_AT_XY_PLANE_ZP	1
VY_AT_XY_PLANE_ZM	0
VY_AT_XY_PLANE_ZP	0
VZ_AT_XY_PLANE_ZM	0
VZ_AT_XY_PLANE_ZP	0

5. region_condition[]: For each field, input boundary condition as follows.

name_of_region	name_of_target	name_of_condition
YZ_BOUNDARY_PLANE_XM_AND_XP	K_Field	PERIODIC
ZX_BOUNDARY_PLANE_YM_AND_YP	K_Field	PERIODIC
XY_BOUNDARY_PLANE_ZM_AND_ZP	K_Field	ZM_WALL__ZP_WALL
YZ_BOUNDARY_PLANE_XM_AND_XP	VolumeFraction	PERIODIC
ZX_BOUNDARY_PLANE_YM_AND_YP	VolumeFraction	PERIODIC
XY_BOUNDARY_PLANE_ZM_AND_ZP	VolumeFraction	ZM_WALL__ZP_WALL
YZ_BOUNDARY_PLANE_XM_AND_XP	Pressure	PERIODIC
ZX_BOUNDARY_PLANE_YM_AND_YP	Pressure	PERIODIC
XY_BOUNDARY_PLANE_ZM_AND_ZP	Pressure	ZM_WALL__ZP_WALL
YZ_BOUNDARY_PLANE_XM_AND_XP	Velocity	PERIODIC
ZX_BOUNDARY_PLANE_YM_AND_YP	Velocity	PERIODIC
XY_BOUNDARY_PLANE_ZM_AND_ZP	Velocity	ZM_VELOCITY_SET__ ZP_VELOCITY_SET

6. Field[]: Register fields. Only the “Velocity”, “Pressure” and “VolumeFraction” are the actual working fields.

registered_field	type	name_of_region	num_of_component	io_flag
VolumeFraction	Scalar	ALL_VERTEX	$(Number_of_Components)	1
ChemicalPotential	Scalar	ALL_VERTEX	$(Number_of_Components)	0
K_Field	Vector	ALL_EDGE	$(Number_of_Components) $(*)$(3)	0
ElectricPotential	Scalar	ALL_VERTEX	1	1
Pressure	Scalar	ALL_VERTEX	1	1
Velocity	Vector	ALL_EDGE	3	1

7. procedures_table_for_initialization[]:
   Set an initialization procedure group name to ”SINGLE_PHASE”.
   Apply a command ”CONSTANT_VOLUME_FRACTION” for the field ”VolumeFraction”.

8. procedures_table_for_evolution[]:
   Set an evolution procedure name to ”SHEAR_FLOW”.
   Apply a command “SOLVE_STOKES_EQUATION_AND_PRESSURE”
   for the Velocity field.

9. Calculation

2.1.9 Application 8: Shear induced flow(2)

This is a simulation of a flow in a two-dimensional(XZ) fluid bounded by two Z-walls. A shear flow condition is given by the Lees-Edwards boundary condition.

1. Mesh parameters:
   Set the parameter.mesh_parameter.type to “SIMPLERECTANGULAR”. In order to create a two-dimensional 64x64 meshes, input the parameter.mesh_parameter.axes[] as follows.

axes[]	values[]	input
[0]	[0]	0.0
[0]	[1]	63.0
[0]	[2]	63.0
[1]	[0]	0.0
[1]	[1]	0.0
[1]	[2]	0.0
[2]	[0]	0.0
[2]	[1]	63.0
[2]	[2]	63.0

2. Solver parameter: Input parameters for the pressure and velocity field solver.
   A parameter "LEES_EDWARDS_BC" is set to "1" in order to use the Lees-Edwards boundary condition.

Solver parameter	input
LEES_EDWARDS_BC	1
ACCELERATION_VALUE_FOR_PRESSURE_SOLVER	1.5
MAX_ITERATION_FOR_PRESSURE_SOLVER	1.0e6
CONVERGENCE_CRITERION_FOR_RRESSURE_SOLVER	1.0e-4
MONITORING_INTERVAL_OF_PRESSURE_SOLVER	100
ACCELERATION_VALUE_FOR_VELOCITY_SOLVER	1.5
MAX_ITERATION_FOR_VELOCITY_SOLVER	1.0e6
CONVERGENCE_CRITERION_FOR_VELOCITY_SOLVER	1.0e-4
MONITORING_INTERVAL_OF_VELOCITY_SOLVER	100
SKIP_INTERVAL_VELOCITY_CALCULATION	1

3. Common physical parameter:
   Input DT=1.0e-3, FINAL_STEP=10,
   INTERVAL_OF_MONITORING=1, and INTERVAL_OF_UDF_OUTPUT=5.

4. Physical parameter:
   Input physical constants as follows. A parameter "SHEAR_RATE_XZ" is an important one which gives shear rate (shear and flow are in z- and x-direction, respectively).

Parameter	input
NUMBER_OF_COMPONENTS	1
CA	1
VISCOSITY	1, 1
SHEAR_RATE_XZ	0.1

5. region_condition[]: For each field, input boundary conditions as follows.

name_of_region	name_of_target	name_of_condition
YZ_BOUNDARY_PLANE_XM_AND_XP	K_Field	PERIODIC
ZX_BOUNDARY_PLANE_YM_AND_YP	K_Field	PERIODIC
XY_BOUNDARY_PLANE_ZM_AND_ZP	K_Field	ZM_WALL__ZP_WALL
YZ_BOUNDARY_PLANE_XM_AND_XP	VolumeFraction	PERIODIC
ZX_BOUNDARY_PLANE_YM_AND_YP	VolumeFraction	PERIODIC
XY_BOUNDARY_PLANE_ZM_AND_ZP	VolumeFraction	ZM_WALL__ZP_WALL
YZ_BOUNDARY_PLANE_XM_AND_XP	Pressure	PERIODIC
ZX_BOUNDARY_PLANE_YM_AND_YP	Pressure	PERIODIC
XY_BOUNDARY_PLANE_ZM_AND_ZP	Pressure	LEES_EDWARDS_BC
YZ_BOUNDARY_PLANE_XM_AND_XP	Velocity	PERIODIC
ZX_BOUNDARY_PLANE_YM_AND_YP	Velocity	PERIODIC
XY_BOUNDARY_PLANE_ZM_AND_ZP	Velocity	LEES_EDWARDS_BC

6. Field[]: Register fields. Only the "Velocity", "Pressure" and "VolumeFraction" are the actual working fields.

registered_field	type	name_of_region	num_of_component	io_flag
VolumeFraction	Scalar	ALL_VERTEX	$(Number_of_Components)	0
ChemicalPotential	Scalar	ALL_VERTEX	$(Number_of_Components)	0
K_Field	Vector	ALL_EDGE	$(Number_of_Components) $(*)$(3)	0
ElectricPotential	Scalar	ALL_VERTEX	1	0
Pressure	Scalar	ALL_VERTEX	1	1
Velocity	Vector	ALL_EDGE	3	1

7. procedures_table_for_initialization[]:
   Set an initialization procedure group name to be "SINGLE_PHASE".
   Apply a command "CONSTANT_VOLUME_FRACTION" for the field "VolumeFraction".

8. procedures_table_for_evolution[]:
   Set an evolution procedure name to be "SHEAR_FLOW". Apply a command "SOLVE_STOKES_EQUATION_AND_PRESSURE" for the field "Velocity".

9. Calculation.

2.1.10 Application 9: Droplet in a flow

This is a simulation of a flow in a two-dimensional(XZ) fluid bounded by two Z-walls and a pressure gap (difference) is applied in X-direction like the Poiseuille flow. In the fluid, a droplet having different viscosity is put and moves with a deformation in the "Poiseuille flow".

1. Mesh parameters:
   Set the parameter.mesh_parameter.type to "SIMPLERECTANGULAR". In order to create a two-dimensional 64x64 meshes, input the parameter.mesh_parameter.axes[] as follows.

axes[]	values[]	input
[0]	[0]	0.0
[0]	[1]	63.0
[0]	[2]	63.0
[1]	[0]	0.0
[1]	[1]	0.0
[1]	[2]	0.0
[2]	[0]	0.0
[2]	[1]	63.0
[2]	[2]	63.0

2. Solver parameter: Input parameters for a pressure and velocity field solver as follows.

Solver parameter	input
LEES_EDWARDS_BC	0
ACCELERATION_VALUE_FOR_PRESSURE_SOLVER	1.5
MAX_ITERATION_FOR_PRESSURE_SOLVER	1.0e6
CONVERGENCE_CRITERION_FOR_RRESSURE_SOLVER	1.0e-4
MONITORING_INTERVAL_OF_PRESSURE_SOLVER	100
ACCELERATION_VALUE_FOR_VELOCITY_SOLVER	1.5
MAX_ITERATION_FOR_VELOCITY_SOLVER	1.0e6
CONVERGENCE_CRITERION_FOR_VELOCITY_SOLVER	1.0e-4
MONITORING_INTERVAL_OF_VELOCITY_SOLVER	100
SKIP_INTERVAL_VELOCITY_CALCULATION	100

3. Common physical parameter:
   Input DT=1.0e-3, FINAL_STEP=1000,
   INTERVAL_OF_MONITORING=1, and INTERVAL_OF_UDF_OUTPUT=1000.

4. Physical parameter: Input physical constants as follows.

Parameter	input
NUMBER_OF_COMPONENTS	2
POLYMERIZATION_INDEX_N	1, 1
DIFFUSION_COEFFICIENT	1.0, 1.0
CHI_01	3.0
CA	1
PRESSURE_GRADIENT	-1.0e-3
VISCOSITY	1.0, 0.1
NUMBER_OF_DROPLET	1
RADIUS_OF_DROPLET	10.0
X_COORDINATE_DROPLET	32
Y_COORDINATE_DROPLET	1
Z_COORDINATE_DROPLET	32

5. region_condition[]: For each field, input boundary conditions as follows.

name_of_region	name_of_target	name_of_condition
YZ_BOUNDARY_PLANE_XM_AND_XP	K_Field	PERIODIC
ZX_BOUNDARY_PLANE_YM_AND_YP	K_Field	PERIODIC
XY_BOUNDARY_PLANE_ZM_AND_ZP	K_Field	ZM_WALL__ZP_WALL
YZ_BOUNDARY_PLANE_XM_AND_XP	VolumeFraction	PERIODIC
ZX_BOUNDARY_PLANE_YM_AND_YP	VolumeFraction	PERIODIC
XY_BOUNDARY_PLANE_ZM_AND_ZP	VolumeFraction	ZM_WALL__ZP_WALL
YZ_BOUNDARY_PLANE_XM_AND_XP	ChemicalPotential	PERIODIC
ZX_BOUNDARY_PLANE_YM_AND_YP	ChemicalPotential	PERIODIC
XY_BOUNDARY_PLANE_ZM_AND_ZP	ChemicalPotential	ZM_WALL__ZP_WALL
YZ_BOUNDARY_PLANE_XM_AND_XP	Pressure	BIASED_PERIODIC
ZX_BOUNDARY_PLANE_YM_AND_YP	Pressure	PERIODIC
XY_BOUNDARY_PLANE_ZM_AND_ZP	Pressure	ZM_WALL__ZP_WALL
YZ_BOUNDARY_PLANE_XM_AND_XP	Velocity	PERIODIC
ZX_BOUNDARY_PLANE_YM_AND_YP	Velocity	PERIODIC
XY_BOUNDARY_PLANE_ZM_AND_ZP	Velocity	ZM_WALL__ZP_WALL

6. Field[]: Register fields as follows.

registered_field	type	name_of_region	num_of_component	io_flag
VolumeFraction	Scalar	ALL_VERTEX	$(Number_of_Components)	1
ChemicalPotential	Scalar	ALL_VERTEX	$(Number_of_Components)	0
K_Field	Vector	ALL_EDGE	$(Number_of_Components) $(*)$(3)	0
ElectricPotential	Scalar	ALL_VERTEX	1	1
Pressure	Scalar	ALL_VERTEX	1	1
Velocity	Vector	ALL_EDGE	3	1

7. procedures_table_for_initialization[]:
   Set an initialization procedure group name to "SINGLE_PHASE".
   Apply commands for fields as,
   "SET_DROPLET" for the field VolumeFraction,
   "FLORY_HUGGINS" for the field ChemicalPotential, and
   "GRADIENT_CHEMICAL_POTENTIAL" for the field K_Field.

8. procedures_table_for_evolution[]:
   Set an evolution procedure name to "POISEUILLE_FLOW".
   Apply commands for fields as,
   "FLORY_HUGGINS" for the field ChemicalPotential,

"GRADIENT_CHEMICAL_POTENTIAL" for the field K_Field,
"SOLVE_EQUATION_OF_CONTINUITY_WITH_FLOW" for the field VolumeFraction,
and "SOLVE_STOKES_EQUATION_AND_PRESSURE" for the field Velocity.

2.1.11 Application 10: Flory-Huggins model phase separation(4)

This is a two-dimensional fluid phase separation simulation with 64x64 meshes (xz-plane), using the Flory-Huggins model and taking the effect of flow into account.

1. Mesh parameters:
   Set the parameter.mesh_parameter.type to "SIMPLERECTANGULAR". In order to make two-dimensional 64x64 meshes, input the parameter.mesh_parameter.axes[] as follows.

axes[]	values[]	input
[0]	[0]	0.0
[0]	[1]	63.0
[0]	[2]	63.0
[1]	[0]	0.0
[1]	[1]	0.0
[1]	[2]	0.0
[2]	[0]	0.0
[2]	[1]	63.0
[2]	[2]	63.0

2. Solver parameter: Input parameters for the velocity and pressure field solver.

Solver parameter	input
ACCELERATION_VALUE_FOR_PRESSURE_SOLVER	1.5
MAX_ITERATION_FOR_PRESSURE_SOLVER	1.0e6
CONVERGENCE_CRITERION_FOR_RRESSURE_SOLVER	1.0e-4
MONITORING_INTERVAL_OF_PRESSURE_SOLVER	100
ACCELERATION_VALUE_FOR_VELOCITY_SOLVER	1.5
MAX_ITERATION_FOR_VELOCITY_SOLVER	1.0e6
CONVERGENCE_CRITERION_FOR_VELOCITY_SOLVER	1.0e-4
MONITORING_INTERVAL_OF_VELOCITY_SOLVER	100
SKIP_INTERVAL_VELOCITY_CALCULATION	100

3. Common physical parameter: Input DT=1.0e-3, FINAL_STEP=100000,
   INTERVAL_OF_MONITORING=10, and INTERVAL_OF_UDF_OUTPUT=5000.

4. Physical parameter: Input physical constants as follows.

Parameter	input
NUMBER_OF_COMPONENTS	2
POLYMERIZATION_INDEX_N	1, 1
AVERAGED_VOLUME_FRACTION	0.5, 0.5
DEVIATION_FROM_AVERAGED_VOLUME_FRACTION	0.01
SEED_OF_RANDOM_NUMBER	715
DIFFUSION_COEFFICIENT	1.0, 1.0
CHI_01	3.0
CA	1
VISCOSITY	1, 1

5. region_condition[]: For each field, input boundary conditions as follows.

name_of_region	name_of_target	name_of_condition
YZ_BOUNDARY_PLANE_XM_AND_XP	K_Field	PERIODIC
ZX_BOUNDARY_PLANE_YM_AND_YP	K_Field	PERIODIC
XY_BOUNDARY_PLANE_ZM_AND_ZP	K_Field	PERIODIC
YZ_BOUNDARY_PLANE_XM_AND_XP	VolumeFraction	PERIODIC
ZX_BOUNDARY_PLANE_YM_AND_YP	VolumeFraction	PERIODIC
XY_BOUNDARY_PLANE_ZM_AND_ZP	VolumeFraction	PERIODIC
YZ_BOUNDARY_PLANE_XM_AND_XP	ChemicalPotential	PERIODIC
ZX_BOUNDARY_PLANE_YM_AND_YP	ChemicalPotential	PERIODIC
XY_BOUNDARY_PLANE_ZM_AND_ZP	ChemicalPotential	PERIODIC
YZ_BOUNDARY_PLANE_XM_AND_XP	Pressure	PERIODIC
ZX_BOUNDARY_PLANE_YM_AND_YP	Pressure	PERIODIC
XY_BOUNDARY_PLANE_ZM_AND_ZP	Pressure	PERIODIC
YZ_BOUNDARY_PLANE_XM_AND_XP	Velocity	PERIODIC
ZX_BOUNDARY_PLANE_YM_AND_YP	Velocity	PERIODIC
XY_BOUNDARY_PLANE_ZM_AND_ZP	Velocity	PERIODIC

6. Field[]: Register fields as follows.

registered_field	type	name_of_region	num_of_component	io_flag
VolumeFraction	Scalar	ALL_VERTEX	$(Number_of_Components)	1
ChemicalPotential	Scalar	ALL_VERTEX	$(Number_of_Components)	0
K_Field	Vector	ALL_EDGE	$(Number_of_Components) $(*)$(3)	0
ElectricPotential	Scalar	ALL_VERTEX	1	1
Pressure	Scalar	ALL_VERTEX	1	1
Velocity	Vector	ALL_EDGE	3	1

7. procedures_table_for_initialization[]:
   Set an initialization procedure group name to "SINGLE_PHASE".
   Apply commands for fields as,
   "CONSTANT_VOLUME_FRACTION_WITH_NOISE" for the field VolumeFraction,
   "FLORY_HUGGINS" for the field ChemicalPotential,
   and "GRADIENT_CHEMICAL_POTENTIAL" for the field K_Field,

8. procedures_table_for_evolution[]:
   Set an evolution procedure name to "TEST_MODEL_H".
   Apply commands for fields as,
   "FLORY_HUGGINS" for the field ChemicalPotential,
   "GRADIENT_CHEMICAL_POTENTIAL" for the field K_Field,
   "SOLVE_EQUATION_OF_CONTINUITY_WITH_FLOW" for the field VolumeFraction,
   and "SOLVE_STOKES_EQUATION_AND_PRESSURE" for the field Velocity.

9. Calculation: It may take several or more hours.

2.1.12 Application 11: Coagulation of droplets

This is a two-dimensional simulation of coagulation of droplets in a two-dimensional 64x64 meshes (xz-plane) using the Flory-Huggins model and taking the effect of flow into account.

1. Mesh parameters:
   Set the parameter.mesh_parameter.type to "SIMPLERECTANGULAR". In order to make a two-dimensional 64x64 meshes, input the parameter.mesh_parameter.axes[] as follows.

axes[]	values[]	input
[0]	[0]	0.0
[0]	[1]	63.0
[0]	[2]	63.0
[1]	[0]	0.0
[1]	[1]	0.0
[1]	[2]	0.0
[2]	[0]	0.0
[2]	[1]	63.0
[2]	[2]	63.0

2. Solver parameter: Input parameters for a velocity and pressure field solver.

Solver parameter	input
ACCELERATION_VALUE_FOR_PRESSURE_SOLVER	1.5
MAX_ITERATION_FOR_PRESSURE_SOLVER	1.0e6
CONVERGENCE_CRITERION_FOR_RRESSURE_SOLVER	1.0e-4
MONITORING_INTERVAL_OF_PRESSURE_SOLVER	100
ACCELERATION_VALUE_FOR_VELOCITY_SOLVER	1.5
MAX_ITERATION_FOR_VELOCITY_SOLVER	1.0e6
CONVERGENCE_CRITERION_FOR_VELOCITY_SOLVER	1.0e-4
MONITORING_INTERVAL_OF_VELOCITY_SOLVER	100
SKIP_INTERVAL_VELOCITY_CALCULATION	100

3. Common physical parameter:
   Input DT=1.0e-3, FINAL_STEP=100000,
   INTERVAL_OF_MONITORING=100, and INTERVAL_OF_UDF_OUTPUT=10000.

4. Physical parameter: Input physical constants as follows.

Parameter	input
NUMBER_OF_COMPONENTS	2
POLYMERIZATION_INDEX_N	1, 1
DIFFUSION_COEFFICIENT	1.0, 1.0
CHI_01	3.0
CA	1
VISCOSITY	1.0, 1.0
NUMBER_OF_DROPLET	2
RADIUS_OF_DROPLET	10.0, 10.0
X_COORDINATE_DROPLET	20, 44
Y_COORDINATE_DROPLET	0, 0
Z_COORDINATE_DROPLET	32, 32

5. region_condition[]:
   For each field, input boundary conditions as follows.

name_of_region	name_of_target	name_of_condition
YZ_BOUNDARY_PLANE_XM_AND_XP	K_Field	PERIODIC
ZX_BOUNDARY_PLANE_YM_AND_YP	K_Field	PERIODIC
XY_BOUNDARY_PLANE_ZM_AND_ZP	K_Field	PERIODIC
YZ_BOUNDARY_PLANE_XM_AND_XP	VolumeFraction	PERIODIC
ZX_BOUNDARY_PLANE_YM_AND_YP	VolumeFraction	PERIODIC
XY_BOUNDARY_PLANE_ZM_AND_ZP	VolumeFraction	PERIODIC
YZ_BOUNDARY_PLANE_XM_AND_XP	ChemicalPotential	PERIODIC
ZX_BOUNDARY_PLANE_YM_AND_YP	ChemicalPotential	PERIODIC
XY_BOUNDARY_PLANE_ZM_AND_ZP	ChemicalPotential	PERIODIC
YZ_BOUNDARY_PLANE_XM_AND_XP	Pressure	PERIODIC
ZX_BOUNDARY_PLANE_YM_AND_YP	Pressure	PERIODIC
XY_BOUNDARY_PLANE_ZM_AND_ZP	Pressure	PERIODIC
YZ_BOUNDARY_PLANE_XM_AND_XP	Velocity	PERIODIC
ZX_BOUNDARY_PLANE_YM_AND_YP	Velocity	PERIODIC
XY_BOUNDARY_PLANE_ZM_AND_ZP	Velocity	PERIODIC

6. Field[]: Register fields as follows.

registered_field	type	name_of_region	num_of_component	io_flag
VolumeFraction	Scalar	ALL_VERTEX	$(Number_of_Components)	1
ChemicalPotential	Scalar	ALL_VERTEX	$(Number_of_Components)	0
K_Field	Vector	ALL_EDGE	$(Number_of_Components) $(*)$(3)	0
ElectricPotential	Scalar	ALL_VERTEX	1	1
Pressure	Scalar	ALL_VERTEX	1	1
Velocity	Vector	ALL_EDGE	3	1

7. procedures_table_for_initialization[]:
   Set an initialization procedure group name to "SINGLE_PHASE".
   Apply commands for fields as,
   "SET_DROPLET" for the field VolumeFraction,
   "FLORY_HUGGINS" for the field ChemicalPotential,
   and "GRADIENT_CHEMICAL_POTENTIAL" for the field K_Field.

8. procedures_table_for_evolution[]:
   Set an evolution procedure name to "TWO_DROPLET_COAGULATION".
   Apply commands for fields as,
   "FLORY_HUGGINS" for the field ChemicalPotential,
   "GRADIENT_CHEMICAL_POTENTIAL" for the field K_Field,
   "SOLVE_EQUATION_OF_CONTINUITY_WITH_FLOW" for the field VolumeFraction,
   and "SOLVE_STOKES_EQUATION_AND_PRESSURE" for the field Velocity.

9. Calculation: It may take several or more hours.

2.2 Sample problems of PhaseSeparation_FEM

This section shows the applications for phase separation dynamics simulator - PhaseSeparation_FEM - using the finite element method.

Input UDF files and output files corresponding to these applications are prepared in a directory of the Muffin distribution as a sub-directory according to the problem.

2.2.1 Application 1: Shear flow (Couette flow)

This sample simulates a fluid flow in an infinite slab bounded by two walls(Y-direction boundaries), and the upper wall moves in a constant velocity and the lower wall does not move. A constant X direction velocity gradient in Y-direction will be realized (Couette flow).

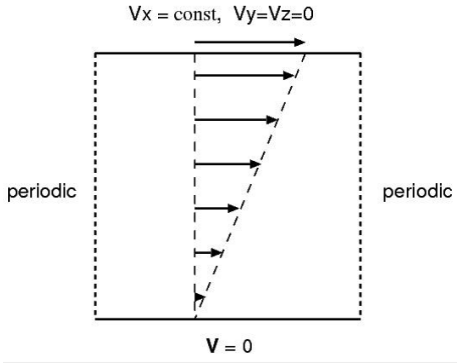

Figure 2.1: PhaseSeparation_FEM Application: Shear flow in a finite slab (Couette flow)

[Input UDF file]

`MUFFIN5/sample/muffin5ebeta/PhaseSeparation/EX01/EX01_in.udf`

[Explanation for input UDF]

- parameter.mesh_parameter:

  Mesh shape is "UNSTRUCTURED_RECT", and 8x8x3 mesh division. A periodic boundary condition is applied in X direction.

- parameter.physical_parameter[] :

  Set viscosity coefficient ("VISCOSITY") to 20.

- region.condition[]

  Boundary conditions are given as partial region conditions as follows.

name	partial region	field	condition name	value
YMIN_P	BOUNDARY_VERTEX_YMIN	Pressure	D	1
YMIN_Vx	BOUNDARY_VERTEX_YMIN	Velocity	D_VX	0
YMIN_Vy	BOUNDARY_VERTEX_YMIN	Velocity	D_VY	0
YMIN_Vz	BOUNDARY_VERTEX_YMIN	Velocity	D_VZ	0
YMAX_Vx	BOUNDARY_VERTEX_YMAX	Velocity	D_VX	$(YMAX_Vx)
YMAX_Vy	BOUNDARY_VERTEX_YMAX	Velocity	D_VY	0
YMAX_Vz	BOUNDARY_VERTEX_YMAX	Velocity	D_VZ	0
ZMIN_Vz	BOUNDARY_VERTEX_ZMIN	Velocity	D_VZ	0
ZMAX_Vz	BOUNDARY_VERTEX_ZMAX	Velocity	D_VZ	0

  - Set a pressure on the bottom wall in Y direction (BOUNDARY_VERTEX_YMIN) to 1.
  - Set a zero velocity on the bottom wall in Y.

– The velocity in X-direction on the Y upper boundary is given by a user defined parameter "YMAX_Vx" in physical_parameter [1].

– Impose a two-dimensional flow in XY plane by setting Z-component of velocity to zero on Z-direction boundaries (BOUNDARY_VERTEX_ZMIN, BOUNDARY_VERTEX_ZMAX).

- dynamics_manager.registered_field[]

  Only 3 fields – Velocity, Pressure and Viscosity – are registered.

2.2.2 Application 2: Poiseuille flow

This is a simulation of a flow in a two-dimensional(XY) fluid bounded by two Y-walls and a pressure gap (difference) is applied in X-direction. The result will be a Poiseuille flow.

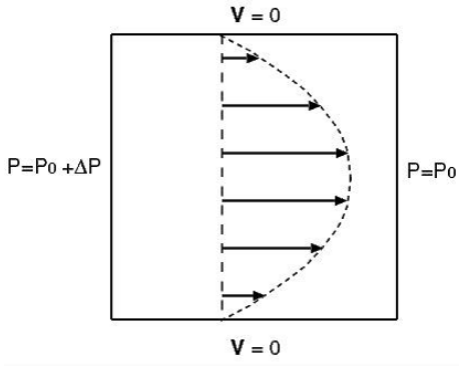

Figure 2.2: PhaseSeparation_FEM Application: Poiseuille flow.

[Input UDF file]

`MUFFIN5/sample/muffin5ebeta/PhaseSeparation/EX02/EX02_in.udf`

[Explanation for input UDF]

- parameter.mesh_parameter:

  Mesh type is "UNSTRUCTURED_RECT", and 16x16x3 mesh division.

- parameter.physical_parameter[] :

  Set the viscosity coefficient ("VISCOSITY") to 10.

- region.condition[]

  Boundary conditions are given as partial region conditions as follows.

[1] Users can use an arbitrary name of parameter which does not written in this manual as "physical_parameter" 's, and use it as an ordinary parameter to define values of other parameters, or values for partial region conditions *etc.*

name	partial region	field	condition name	value
XMIN_p	BOUNDARY_VERTEX_XMIN	Pressure	D	1
XMAX_p	BOUNDARY_VERTEX_XMAX	Pressure	D	0
XMIN_Vy	BOUNDARY_VERTEX_XMIN	Velocity	D_VY	0
XMIN_Vz	BOUNDARY_VERTEX_XMIN	Velocity	D_VZ	0
XMAX_Vy	BOUNDARY_VERTEX_XMAX	Velocity	D_VY	0
XMAX_Vz	BOUNDARY_VERTEX_XMAX	Velocity	D_VZ	0
YMIN_Vx	BOUNDARY_VERTEX_YMIN	Velocity	D_VX	0
YMIN_Vy	BOUNDARY_VERTEX_YMIN	Velocity	D_VY	0
YMIN_Vz	BOUNDARY_VERTEX_YMIN	Velocity	D_VZ	0
YMAX_Vx	BOUNDARY_VERTEX_YMAX	Velocity	D_VX	0
YMAX_Vy	BOUNDARY_VERTEX_YMAX	Velocity	D_VY	0
YMAX_Vz	BOUNDARY_VERTEX_YMAX	Velocity	D_VZ	0
ZMIN_Vz	BOUNDARY_VERTEX_ZMIN	Velocity	D_VZ	0
ZMAX_Vz	BOUNDARY_VERTEX_ZMAX	Velocity	D_VZ	0

  - Set a pressure difference on X direction boundaries (BOUNDARY_VERTEX_XMIN, BOUNDARY_VERTEX_XMAX) to 8.
  - On X-direction boundaries (BOUNDARY_VERTEX_XMIN), make Y and Z components of velocity to zero.
  - On Y-direction boundaries (BOUNDARY_VERTEX_YMIN, BOUNDARY_VERTEX_YMAX) velocity is set to zero.
  - Impose a two-dimensional flow in XY plane by setting Z-component of the velocity to zero on Z-direction boundaries (BOUNDARY_VERTEX_ZMIN, BOUNDARY_VERTEX_ZMAX).

- dynamics_manager.registered_field[]

  Only 3 fields – Velocity, Pressure and Viscosity – are registered.

[Note]

The FEM simulator doesn't have the biased periodic condition supported in the FDM simulator, so instead of setting periodic condition in X direction, here we are set Y- and Z-components of velocity to zero to make Poiseuille flow.

2.2.3 Application 3: Phase separation by Flory-Huggins model of free energy(1)

This is a simulation of phase separation of a two component system (without flow effect) using the Flory-Huggins free energy.

[Input UDF file]

`MUFFIN5/sample/muffin5ebeta/PhaseSeparation/EX03/EX03_in.udf`

[Explanation for input UDF]

- parameter.mesh_parameter:

  The geometry type is "UNSTRUCTURED_RECT", and 32x32x4 mesh division. Set periodic boundary conditions for X,Y and Z directions.

- parameter.physical_parameter[] :

Parameter name(KEY)	value
NUMBER_OF_COMPONENTS	2
POLYMERIZATION_INDEX_N	1, 1
AVERAGED_VOLUME_FRACTION	0.5, 0.5
DEVIATION_FROM_AVERAGED_VOLUME_FRACTION	0.01
SEED_OF_RANDOM_NUMBER	1
Ca	1.0
DIFFUSION_COEFFICIENT	1.0
Chi_01	3.0
MODE_COUPLING_CONSTANT_G0	1.0

- region.condition[]

  No boundary condition is given because all directions are periodic.

- dynamics_manager.registered_field[]

  Register the VolumeFraction, ChecmicalPotential and K_Field.

- dynamics_manager.procedures_table_for_initialization[].command_list[]

  Initialize the VolumeFraction field by a command "CONSTANT_VOLUME_FRACTION_WITH_NOISE".

- dynamics_manager.procedures_table_for_evolution[].command_list[]

  Time evolution procedure is given as follows;

field name	command name
ChemicalPotential	FLORY_HUGGINS
K_Field	GRADIENT_CHEMICAL_POTENTIAL
VolumeFraction	SOLVE_EQUATION_OF_CONTINUITY_WITHOUT_FLOW

2.2.4 Application 4: Phase separation by Flory-Huggins model of free energy(2)

This is a similar calcualtion as the application 3, but the effect of fluid flow field is applied.

[Input UDF file]

`MUFFIN5/sample/muffin5ebeta/PhaseSeparation/EX04/EX04_in.udf`

[Explanation for input UDF]

- parameter.mesh_parameter:

  Mesh shape is "UNSTRUCTURED_RECT", and 32x32x3 mesh division. Set periodic boundary conditions for X,Y and Z directions.

- parameter.solver_parameter[] :

  Perform a Stokes flow calculation by a pressure and velocity solver. The following parameters control the solver.

Parameter name(KEY)	value
MATRIX_SOLVER	ICCG
DT_FOR_V	0.05
MAX_ITERATION_FOR_VELOCITY_SOLVER	500
CONVERGENCE_CRITERION_FOR_ VELOCITY_SOLVER	(DT_FOR_V)(*)$(0.01)
SKIP_INTERVAL_VELOCITY_CALCULATION	50

  – DT_FOR_V is a time mesh for the Stokes flow solver, and the time is not the physical time but only for the iterative flow solver. It is selected to satisfy the condition of eq.(1.83) in section 1.5.2.

  - CONVERGENCE_CRITERION_FOR_VELOCITY_SOLVER is calculated using the parameter DT_FOR_V so that relative change of the velocity less than 0.01 becomes the convergence condition.
  - SKIP_INTERVAL_VELOCITY_CALCULATION=50 means that the velocity pressure solver is activated for every 50 time step. This is to save calculation time by skipping the flow field calculation for most of time steps assuming that the velocity field should chang slowly for time steps.

- parameter.common_physical_parameter:

  Input DT=1.0e-2, FINAL_STEP=2000, INTERVAL_OF_MONITORING=1,
  and INTERVAL_OF_UDF_OUTPUT=500.

- parameter.physical_parameter[] :

Parameter name(KEY)	value
NUMBER_OF_COMPONENTS	2
POLYMERIZATION_INDEX_N	1, 1
AVERAGED_VOLUME_FRACTION	0.5, 0.5
DEVIATION_FROM_AVERAGED_VOLUME_FRACTION	0.01
SEED_OF_RANDOM_NUMBER	1
Ca	1.0
DIFFUSION_COEFFICIENT	1.0
VISCOSITY	1.0
Chi_01	3.0

- region.condition[]

  No boundary condition is given because all directions are periodic.

- dynamics_manager.registered_field[]

  Register not only the VolumeFraction, ChecmicalPotential and K_Field, but also the Velocity, Viscosity and Pressure fields for flow calculation.

- dynamics_manager.procedures_table_for_initialization[].command_list[]

  Initializing the VolumeFraction field by a command
  “CONSTANT_VOLUME_FRACTION_WITH_NOISE”.

- dynamics_manager.procedures_table_for_evolution[].command_list[]

  Time evolution procedure is given as follows.

field name	command name
ChemicalPotential	FLORY_HUGGINS
K_Field	GRADIENT_CHEMICAL_POTENTIAL
VolumeFraction	SOLVE_EQUATION_OF_CONTINUITY_WITH_FLOW
Velocity	SOLVE_STOKES_EQUATION_AND_PRESSURE

Chapter 3

Operation guide of PhaseSeparation

3.1 Commands and parameters for fields of PhaseSeparation_FDM

3.1.1 Input parameters of PhaseSeparation_FDM

Name of Parameters	Meanings and notations in theory
NUMBER_OF_COMPONENTS	number of components
B	dimensionless electric energy B
DIELECTRIC_CONSTANT	relative dielectric constants ϵ_α give for each component
CHARGE_DENSITY	charge density of each component ρ_α
GAMMA_S	surface energy of each component showing the wettability of a wall $\gamma_{s\alpha}$
GAMMA_S_REGION	surface energy of each component showing the wettability of a wall $\gamma_{s\alpha}$ rectangle region with following parameters(delimiter is space) component-id potential xmin ymin zmin xmax ymax zmax
ACCELERATION_VALUE _FOR_E-POTENTIAL	acceleration factor for electric potential calculation
MAX_ITERATION _FOR_E-POTENTIAL_SOLVER	maximum number of iterations for electric potential calculation
CONVERGENCE_CRITERION_FOR _E-POTENTIAL	convergence criterion of electric potential calculation
MONITORING_INTERVAL_OF _E-POTENTIAL_SOLVER	convergence monitoring interval of electric potential calculation
ELECTRIC_POTENTIAL_GRADIENT	initial value of an electric potential gradient
ELECTRIC_POTENTIAL _AT_YZ_PLANE_XM	boundary value of electric potential(YZ plane $-X$ side)
ELECTRIC_POTENTIAL _AT_YZ_PLANE_XP	boundary value of electric potential(YZplane $+X$ side)
ELECTRIC_POTENTIAL _AT_ZX_PLANE_YM	boundary value of electric potential(ZXplane $-Y$ side)
ELECTRIC_POTENTIAL _AT_ZX_PLANE_YP	boundary value of electric potential(ZXplane $+Y$ side)
ELECTRIC_POTENTIAL _AT_XY_PLANE_ZM	boundary value of electric potential(XYplane $-Z$ side)
ELECTRIC_POTENTIAL _AT_XY_PLANE_ZP	boundary value of electric potential(XYplane $+Z$ side)
CONSTANT_TERM_OF _ELECTRIC_POTENTIAL _OSCILLATION_AT_YZ_PLANE_XP	ϕ_o of oscillating boundary electric potential; $\phi(x,y,z)\|_{Boundary} = \phi_o + \delta\phi \cdot \sin(\omega t)$ (YZ plane, $+X$ side)

AMPLITUDE_OF_ELECTRIC_POTENTIAL _OSCILLATION_AT_YZ_PLANE_XP	$\delta\phi$ of oscillating boundary electric potential (YZplane $+X$ side)
FREQUENCY_OF_ELECTRIC_POTENTIAL _OSCILLATION_AT_YZ_PLANE_XP	ω of oscillating boundary electric potential (YZplane, $+X$ side)
GRADIENT_OF_E-POTENTIAL _AT_YZ_PLANE_XM	boundary value of electric potential gradient (YZplane, $-X$ side)
GRADIENT_OF_E-POTENTIAL _AT_YZ_PLANE_XP	boundary value of electric potential gradient (YZplane, $+X$ side)
GRADIENT_OF_E-POTENTIAL _AT_ZX_PLANE_YM	boundary value of electric potential gradient (ZXplane, $-Y$ side)
GRADIENT_OF_E-POTENTIAL _AT_ZX_PLANE_YP	boundary value of electric potential gradient (ZXplane, $+Y$ side)
GRADIENT_OF_E-POTENTIAL _AT_XY_PLANE_ZM	boundary value of electric potential gradient (XYplane, $-Z$ side)
GRADIENT_OF_E-POTENTIAL _AT_XY_PLANE_ZP	boundary value of electric potential gradient (XYplane, $+Z$ side)
SEED_OF_RANDOM_CURRENT	initial random number for random initialization of flux field $K(J)$
STRENGTH_OF_NOISE	deviation for random initialization of flux field $K(J)$
CA	Capillary number
MONITORING_INTERVAL_OF _PRESSURE_SOLVER	convergence monitoring interval of pressure calculation
ACCELERATION_VALUE_FOR _PRESSURE_SOLVER	acceleration factor for pressure calculation
MAX_ITERATION_FOR _PRESSURE_SOLVER	maximum number of iterations for pressure calculation
CONVERGENCE_CRITERION _FOR_PRESSURE_SOLVER	convergence criterion of pressure calculation
VISCOSITY	viscosity coefficient of each component η_α
PRESSURE_GRADIENT	boundary pressure gap for biased periodic condition of pressure
PRESSURE_AT_YZ_PLANE_XM	boundary pressure (YZ plane, $-X$ side)
PRESSURE_AT_YZ_PLANE_XP	boundary pressure (YZ plane, $+X$ side)
PRESSURE_AT_ZX_PLANE_YM	boundary pressure (ZX plane, $-Y$ side)
PRESSURE_AT_ZX_PLANE_YP	boundary pressure (ZX plane, $+Y$ side)
PRESSURE_AT_XY_PLANE_ZM	boundary pressure (XY plane, $-Z$ side)
PRESSURE_AT_XY_PLANE_ZP	boundary pressure (XY plane, $+Z$ side)
CONSTANT_VALUE_OF_P _OSCILLATION_AT _YZ_PLANE_XP	P_o of oscillating boundary pressure ; $P(x,y,z)\|_{Boundary} = P_o + \delta P \cdot \sin(\omega t)$ (YZplane, $+X$ side)
AMPLITUDE_OF_P _OSCILLATION_AT_YZ_PLANE_XP	δP of oscillating boundary pressure (YZplane, $+X$ side)
FREQUENCY_OF_P _OSCILLATION_AT_YZ_PLANE_XP	ω of oscillating boundary pressure (YZplane, $+X$ side)
VELOCITY_RAW_DATA_FILE	input file name for initialization of velocity by file input.
SKIP_INTERVAL _VELOCITY_CALCULATION	time step interval for velocity calculation
VX_AT_YZ_PLANE_XM	X component of boundary velocity(YZ plane, $-X$ side)
VY_AT_YZ_PLANE_XM	Y component of boundary velocity(YZ plane, $-X$ side)
VZ_AT_YZ_PLANE_XM	Z component of boundary velocity(YZ plane, $-X$ side)
VX_AT_YZ_PLANE_XP	X component of boundary velocity(YZ plane, $+X$ side)
VY_AT_YZ_PLANE_XP	Y component of boundary velocity(YZ plane, $+X$ side)

VZ_AT_YZ_PLANE_XP	Z component of boundary velocity(YZ plane, $+X$ side)
VX_AT_ZX_PLANE_YM	X component of boundary velocity(ZX plane, $-Y$ side)
VY_AT_ZX_PLANE_YM	Y component of boundary velocity(ZX plane, $-Y$ side)
VZ_AT_ZX_PLANE_YM	Z component of boundary velocity(ZX plane, $-Y$ side)
VX_AT_ZX_PLANE_YP	X component of boundary velocity(ZX plane, $+Y$ side)
VY_AT_ZX_PLANE_YP	Y component of boundary velocity(ZX plane, $+Y$ side)
VZ_AT_ZX_PLANE_YP	Z component of boundary velocity(ZX plane, $+Y$ side)
VX_AT_XY_PLANE_ZM	X component of boundary velocity(XY plane, $-Z$ side)
VY_AT_XY_PLANE_ZM	Y component of boundary velocity(XY plane, $-Z$ side)
VZ_AT_XY_PLANE_ZM	Z component of boundary velocity(XY plane, $-Z$ side)
VX_AT_XY_PLANE_ZP	X component of boundary velocity(XY plane, $+Z$ side)
VY_AT_XY_PLANE_ZP	Y component of boundary velocity(XY plane, $+Z$ side)
VZ_AT_XY_PLANE_ZP	Z component of boundary velocity(XY plane, $+Z$ side)
SHEAR_RATE_XZ	shear rate for Lees Edwards boundary condition (XZ direction)
MONITORING_INTERVAL_OF _VELOCITY_SOLVER	convergence monitoring interval of velocity calculation
ACCELERATION_VALUE_FOR _VELOCITY_SOLVER	acceleration factor for velocity calculation
CONVERGENCE_CRITERION_FOR _VELOCITY_SOLVER	convergence criterion of velocity calculation
MAX_ITERATION_FOR _VELOCITY_SOLVER	maximum number of iterations for velocity calculation
CHI_mn	χ-parameter of components n,m (only for $m < n$)
POLYMERIZATION_INDEX_N	polymerization index of each component N_α
AVS_DATA_FILE_NAME	file name for initialization of volume fraction by AVS format file input.
AVERAGED_VOLUME_FRACTION	averaged volume fraction as initial value $\psi_{\alpha 0}$
DEVIATION_FROM _AVERAGED_VOLUME_FRACTION	magnitude of noise given to initial value of volume fraction
SEED_OF_RANDOM_NUMBER	initial random number for random initialization of volume fraction field.
VOLUME_FRACTION_GRADIENT _ALONG_X	X-direction gradient of volume fraction ψ_α for Biased Periodic Condition
VOLUME_FRACTION_GRADIENT _ALONG_Y	Y-direction gradient of volume fraction ψ_α for Biased Periodic Condition
VOLUME_FRACTION_GRADIENT _ALONG_Z	Z-direction gradient of volume fraction ψ_α for Biased Periodic Condition
BULK_VOLUME_FRACTION	volume fraction of each component ψ_α for bulk boundary condition
REMOVE_FRACTION_OF_SOLVENT	rate of solvent removal from boundary
INITIAL_SHEAR_STRAIN_ZX	strain for initialization of volume fraction by Affine transformation
NUMBER_OF_DROPLETS	number of droplets placed in initialization of volume fraction.
RADIUS_OF_DROPLET	radius of each droplet placed in initialization of volume fraction.
X_COORDINATE_OF_DROPLET	X coordinate of each droplet placed in initialization of volume fraction.
Y_COORDINATE_OF_DROPLET	Y coordinate of each droplet placed in initialization of volume fraction.
Z_COORDINATE_OF_DROPLET	Z coordinate of each droplet placed in initialization of volume fraction.
NUMBER_OF_LAMELLAE	number of lamellae placed in

	initialization of volume fraction.
DIRECTION_OF_LAMELLAE	direction of lamella placed in initialization of volume fraction("X"|"Y"|"Z")
PHASE_OF_LAMELLAE	phase of lamella placed in initialization of volume fraction("0"|"1")
DIFFUSION_COEFFICIENT	1. diffusion coefficient of each component L_α 2. diffusion on the phase separated structure : diffusion coefficient for each region
ALPHA_OHTA_KAWASAKI	Ohta-Kawasaki model's coefficient α
XI_OHTA_KAWASAKI	Ohta-Kawasaki model's coefficient ξ
G_OHTA_KAWASAKI	Ohta-Kawasaki model's coefficient g
OBSTACLE_DATA_FILE	file name storing position of obstacle meshes
INITIAL_DIFFUSION_FIELD	diffusion on the phase separated structure : intial value for all region
DF_AT_YZ_PLANE_XM	diffusion on the phase separated structure : component of boundary value(YZ plane, $-X$ side)
DF_AT_YZ_PLANE_XP	diffusion on the phase separated structure : component of boundary value(YZ plane, $+X$ side)
DF_AT_ZX_PLANE_YM	diffusion on the phase separated structure : component of boundary value(ZX plane, $-Y$ side)
DF_AT_ZX_PLANE_YP	diffusion on the phase separated structure : component of boundary value(ZX plane, $+Y$ side)
DF_AT_XY_PLANE_ZM	diffusion on the phase separated structure : component of boundary value(XY plane, $-Z$ side)
DF_AT_XY_PLANE_ZP	diffusion on the phase separated structure : component of boundary value(XY plane, $+Z$ side)

3.1.2 PhaseSeparation_FDM - list of fields

Field name	meanings and notation in theory
ChemicalPotential	Chemical potential field μ_α
ElectricPotential	Electric potential field Φ
K_Field	Flux field $\boldsymbol{J}_\alpha(\boldsymbol{K})$
Pressure	Pressure field P
Velocity	Velocity field $\boldsymbol{v}$
VolumeFraction	Volume fraction field ψ_α
Obstacle	Obstacle field
Diffusion_Field	Diffusion field on the phase separated structure
Diffusion_Flux	Diffusion flux on the phase separated structure

K_Field $\alpha = 0$ component of K_Field may be used to store volume force field $\boldsymbol{K}$ in fluid equation.

3.1.3 PhaseSeparation_FDM - commands of fields

ChemicalPotential : chemical potential field - commands

ChemicalPotential	Name
Time evolution	"SET_ZERO"
Time evolution	"SYMMETRIC_GL"
Time evolution	"FLORY_HUGGINS"
Time evolution	"OHTA_KAWASAKI_2"
Time evolution	"ADD_ELECTRIC_EFFECT_OF_DIELECTRIC_MEDIUM"
Time evolution	"ADD_ELECTROSTATIC_EFFECT_USING_CHARGE_DENSITY"
Time evolution	"ADD_EFFECT_OF_WETTING_FOR_UNIFORM_Z_WALL"
Time evolution	"ADD_EFFECT_OF_WETTING_REGION"
Evaluation	"RETURN_TRUE_FUNC"

1. **ChemicalPotential - time evolution commands**

Name	**"SET_ZERO"**
Function	set field values to zero.

Name	**"SYMMETRIC_GL"**
Function	2-component Ginzburg-Landau model $\mu = -\psi + \psi^3 - \Delta\psi(\psi \equiv \psi_1 - \psi_0)$
Dependent filed	VolumeFraction

Name	**"FLORY_HUGGINS"**
Function	Flory-Huggins free energy.
Dependent parameter	NUMBER_OF_COMPONENTS
Dependent parameter	CHI_mn
Dependent parameter	POLYMERIZATION_INDEX_N

Name	**"OHTA_KAWASAKI_2"**
Function	2 components Ohta-Kawasaki model(Proc. of SPIE, 8680, 86801I (2013)) $\mu = -\frac{1}{\xi^2}\eta + g\eta^3 - \nabla^2\eta$, where $\eta = 2\psi_0 - 1$ K_Field command should be "GRADIENT_CHEMICAL_POTENTIAL_ORDER_PARAMETER", VolumeFraction command should be "SOLVE_OHTA_KAWASAKI_2". length is scaled by $R_g(= a\sqrt{N/2}$, where a is Kuhn length). $\frac{1}{\xi^2} = 2f(1-f)(2\chi N - \frac{s(f)}{2f^2(1-f)^2})$, f is number fraction of component 0 in diblock-copolymer, N is number of polymerization, χ is interaction parameter. $s(f)$ is empirical parameter, e.g. $s(f) = 0.9$ for $f = 0.5$
Dependent filed	VolumeFraction
Dependent parameter	XI_OHTA_KAWASAKI
Dependent parameter	G_OHTA_KAWASAKI

Name	**"ADD_ELECTRIC_EFFECT_OF_DIELECTRIC_MEDIUM"**
Function	By using dielectric constants depending on volume fraction field and electric potential field Φ, take an effect of dielectric fluid under electric field into chemical potential.
Dependent filed	ElectricPotential
Dependent parameter	B
Dependent parameter	DIELECTRIC_CONSTANT

Name	**"ADD_ELECTROSTATIC_EFFECT_USING _CHARGE_DENSITY"**
Function	add effect of charge distribution and electric potential field to chemical potential; $(\partial\rho_e/\partial\psi)\Phi$
Dependent filed	ElectricPotential
Dependent parameter	CHARGE_DENSITY

Name	**"ADD_EFFECT_OF_WETTING_FOR_UNIFORM_Z_WALL"**
Function	Add wettability effect on wall to chemical potential. This effect exists only when walls are on $Z = 0$, $Z = L_z$, so, if there are no such a wall boundary, this command does not affect anything on simulation results.
Dependent parameter	GAMMA_S

Name	**"ADD_EFFECT_OF_WETTING_REGION"**
Function	Add wettability effect on designated region.
Dependent parameter	GAMMA_S_REGION

2. **ChemicalPotential - boundary condition (partial condition) commands**

Partial region condition	treatment
PERIODIC	periodic boundary condition
XM_WALL__XP_WALL	Wall boundary condition $\boldsymbol{n} \cdot \nabla \mu_\alpha = 0$ on both $-X$ and $+X$ boundaries
YM_WALL__YP_WALL	Wall boundary condition $\boldsymbol{n} \cdot \nabla \mu_\alpha = 0$ on both $-Y$ and $+Y$ boundaries
ZM_WALL__ZP_WALL	Wall boundary condition $\boldsymbol{n} \cdot \nabla \mu_\alpha = 0$ on both $-Z$ and $+Z$ boundaries
LEES_EDWARDS_BC	Lees-Edwards boundary condition

3. **ChemicalPotential - evaluation commands**

Name	**"RETURN_TRUE_FUNC"**
Function	Always return "true" flag. This function is used to perform an analysis command with a constant time step interval.

ElectricPotential : electric potential field - commands

ElectricPotential	Name
Time evolution	"ELECTRIC_POTENTIAL_SOLVER"
Analysis	"OUTPUT_SNAPSHOT_IN_AVS_FORMAT"
Evaluation	"RETURN_TRUE_FUNC"

1. **ElectricPotential - time evolution commands**

Name	**"ELECTRIC_POTENTIAL_SOLVER"**
Function	Calculate electric potential by solving Poisson equation with iterative method (SOR).
Dependent parameter	ACCELERATION_VALUE_FOR_E-POTENTIAL
Dependent parameter	MAX_ITERATION_FOR_E-POTENTIAL_SOLVER
Dependent parameter	CONVERGENCE_CRITERION_FOR_E-POTENTIAL
Dependent parameter	B
Dependent parameter	MONITORING_INTERVAL_OF_E-POTENTIAL_SOLVER
Dependent parameter	ELECTRIC_POTENTIAL_GRADIENT
Dependent parameter	ELECTRIC_POTENTIAL_AT_YZ_PLANE_XM
Dependent parameter	ELECTRIC_POTENTIAL_AT_YZ_PLANE_XP
Dependent parameter	ELECTRIC_POTENTIAL_AT_ZX_PLANE_YM
Dependent parameter	ELECTRIC_POTENTIAL_AT_ZX_PLANE_YP
Dependent parameter	ELECTRIC_POTENTIAL_AT_XY_PLANE_ZM
Dependent parameter	ELECTRIC_POTENTIAL_AT_XY_PLANE_ZP
Dependent parameter	CONSTANT_TERM_OF_ELECTRIC_POTENTIAL_OSCILLATION_AT_YZ_PLANE_XP
Dependent parameter	AMPLITUDE_OF_ELECTRIC_POTENTIAL_OSCILLATION_AT_YZ_PLANE_XP
Dependent parameter	FREQUENCY_OF_ELECTRIC_POTENTIAL_OSCILLATION_AT_YZ_PLANE_XP
Dependent parameter	GRADIENT_OF_E-POTENTIAL_AT_YZ_PLANE_XM
Dependent parameter	GRADIENT_OF_E-POTENTIAL_AT_YZ_PLANE_XP
Dependent parameter	GRADIENT_OF_E-POTENTIAL_AT_ZX_PLANE_YM
Dependent parameter	GRADIENT_OF_E-POTENTIAL_AT_ZX_PLANE_YP
Dependent parameter	GRADIENT_OF_E-POTENTIAL_AT_XY_PLANE_ZM
Dependent parameter	GRADIENT_OF_E-POTENTIAL_AT_XY_PLANE_ZP

2. **ElectricPotential - boundary condition (partial condition) commands**

Partial region condition	treatment
PERIODIC	periodic boundary condition
BIASED_PERIODIC	Biased Periodic boundary condition
XM_NEUMANN__XP_NEUMANN	Neumann condition on both $-X$ and $+X$ boundaries
XM_NEUMANN__XP_DIRICHLET	Neumann condition on $-X$, and Dirichlet condition on $+X$
XM_DIRICHLET__XP_NEUMANN	Dirichlet condition on $-X$, Neumann condition on $+X$
XM_DIRICHLET__XP_DIRICHLET	Dirichlet condition on both $-X$ and $+X$ boundaries
OSCILLATORY	oscillation potential boundary condition $\phi(x,y,z)\|_{Boundary} = \phi_o + \delta\phi \cdot \sin(\omega t)$
YM_NEUMANN__YP_NEUMANN	Neumann condition on both $-Y$ and $+Y$ boundaries
YM_NEUMANN__YP_DIRICHLET	Neumann condition on $-Y$, and Dirichlet condition on $+Y$
YM_DIRICHLET__YP_NEUMANN	Dirichlet condition on $-Y$, Neumann condition on $+Y$
YM_DIRICHLET__YP_DIRICHLET	Dirichlet condition on both $-Y$ and $+Y$ boundaries
ZM_NEUMANN__ZP_NEUMANN	Neumann condition on both $-Z$ and $+Z$ boundaries
ZM_NEUMANN__ZP_DIRICHLET	Neumann condition on $-Z$, and Dirichlet condition on $+Z$
ZM_DIRICHLET__ZP_NEUMANN	Dirichlet condition on $-Z$, Neumann condition on $+Z$
ZM_DIRICHLET__ZP_DIRICHLET	Dirichlet condition on both $-Z$ and $+Z$ boundaries

3. **ElectricPotential- analysis commands**

Name	**"OUTPUT_SNAPSHOT_IN_AVS_FORMAT"**
Function	output calculation results on an AVS format file(field-data).

4. **ElectricPotential - evaluation commands**

Name	**"RETURN_TRUE_FUNC"**
Function	Always return "true" flag. This function is used to perform an analysis command with a constant time step interval.

K_Field : flux field - commands

K_Field	Name
Initialization	"SET_ZERO"
Time evolution	"GRADIENT_CHEMICAL_POTENTIAL"
Time evolution	"GRADIENT_CHEMICAL_POTENTIAL_ORDER_PARAMETER"
Time evolution	"ADD_RANDOM_CURRENT"

1. **K_Field - initialization commands**

Name	**"SET_ZERO"**
Function	Set field values to zero.

2. **K_Field - time evolution commands**

Name	**"GRADIENT_CHEMICAL_POTENTIAL"**
Function	Calculate flux from volume fraction and chemical potential; $\boldsymbol{K}_\alpha = -\psi_\alpha \nabla \mu_\alpha$

Name	**"GRADIENT_CHEMICAL_POTENTIAL_ORDER_PARAMETER"**
Function	This command should be used when OHTA_KAWASAKI_2 command is used for Chemical Potential. calculate flux from chemical potential. $\boldsymbol{K} = -\nabla\mu$

Name	**"ADD_RANDOM_CURRENT"**
Function	add random noise on flux field.
Dependent parameter	SEED_OF_RANDOM_CURRENT
Dependent parameter	STRENGTH_OF_NOISE

3. **K_Field - boundary condition (partial condition) commands**

Partial region condition	treatment
PERIODIC	periodic boundary condition
XM_WALL__XP_WALL	wall boundary condition on both $-X$ and $+X$ boundaries.
XM_WALL__XP_BULK	wall boundary condition on $-X$, bulk boundary condition on $+X$.
XM_BULK__XP_WALL	bulk boundary condition on $-X$, wall boundary condition on $+X$.
XM_BULK__XP_BULK	bulk boundary condition on both $-X$ and $+X$ boundaries.
YM_WALL__YP_WALL	wall boundary condition on both $-Y$ and $+Y$ boundaries.
YM_WALL__YP_BULK	wall boundary condition on $-Y$, bulk boundary condition on $+Y$.
YM_BULK__YP_WALL	bulk boundary condition on $-Y$, wall boundary condition on $+Y$.
YM_BULK__YP_BULK	bulk boundary condition on both $-Y$ and $+Y$ boundaries.
ZM_WALL__ZP_WALL	wall boundary condition on both $-Z$ and $+Z$ boundaries.
ZM_WALL__ZP_BULK	wall boundary condition on $-Z$, bulk boundary condition on $+Z$.
ZM_BULK__ZP_WALL	bulk boundary condition on $-Z$, wall boundary condition on $+Z$.
ZM_BULK__ZP_BULK	bulk boundary condition on both $-Z$ and $+Z$ boundaries.

Pressure : pressure field - commands

Pressure	Name
Initialization	"SET_ZERO"
Time evolution	"SOLVE_PRESSURE"
Time evolution	"CALCULATION_OF_SOURCE_FIELD"
Time evolution	"ONE_ITERATION"
Time evolution	"CALCULATION_OF_VISCOSITY"
Analysis	"OUTPUT_SNAPSHOT_IN_AVS_FORMAT"
Evaluation	"RETURN_TRUE_FUNC"

1. **Pressure - initialization commands**

Name	**"SET_ZERO"**
Function	Set field values to zero.

2. **Pressure - time evolution commands**

Name	**"SOLVE_PRESSURE"**
Function	Solve pressure field by iterative solution of Poisson equation; $\nabla^2 p = \nabla \cdot [\nabla(\eta\{\nabla \boldsymbol{v} + (\nabla \boldsymbol{v})^t\})] + \nabla \cdot \boldsymbol{K}$
Dependent filed	Velocity
Dependent filed	K_Field
Dependent parameter	ACCELERATION_VALUE_FOR_PRESSURE_SOLVER
Dependent parameter	MAX_ITERATION_FOR_PRESSURE_SOLVER
Dependent parameter	CONVERGENCE_CRITERION_PRESSURE_SOLVER
Dependent parameter	MONITORING_INTERVAL_OF_PRESSURE_SOLVER
Dependent parameter	VISCOSITY
Dependent parameter	PRESSURE_GRADIENT
Dependent parameter	PRESSURE_AT_YZ_PLANE_XM
Dependent parameter	PRESSURE_AT_YZ_PLANE_XP
Dependent parameter	PRESSURE_AT_ZX_PLANE_YM
Dependent parameter	PRESSURE_AT_ZX_PLANE_YP
Dependent parameter	PRESSURE_AT_XY_PLANE_ZM
Dependent parameter	PRESSURE_AT_XY_PLANE_ZP
Dependent parameter	CONSTANT_TERM_OF_PRESSURE_OSCILLATION _AT_YZ_PLANE_XP
Dependent parameter	AMPLITUDE_OF_PRESSURE_OSCILLATION _AT_YZ_PLANE_XP
Dependent parameter	FREQUENCY_OF_PRESSURE_OSCILLATION _AT_YZ_PLANE_XP

Name	**"CALCULATION_OF_SOURCE_FIELD"**
Function	Calculate source term of Poisson equation of pressure field.
Dependent filed	Velocity
Dependent filed	K_Field
Dependent parameter	CA

Name	**"ONE_ITERATION"**
Function	One iteration of solution for Poisson equation of pressure field.
Dependent parameter	MONITORING_INTERVAL_OF_PRESSURE_SOLVER
Dependent parameter	ACCELERATION_VALUE_FOR_PRESSURE_SOLVER
Dependent parameter	MAX_ITERATION_FOR_PRESSURE_SOLVER
Dependent parameter	CONVERGENCE_CRITERION_FOR_PRESSURE_SOLVER

Name	**"CALCULATION_OF_VISCOSITY"**
Function	Calculate viscosity coefficient $\eta = \sum_\alpha \eta_\alpha \psi_\alpha$
Dependent filed	VolumeFraction
Dependent parameter	VISCOSITY

3. **Pressure - boundary condition (partial condition) commands**

Partial region condition	treatment
PERIODIC	periodic boundary condition
BIASED_PERIODIC	Biased Periodic boundary condition
XM_WALL__XP_WALL	set pressure to make ψ field satisfy wall boundary condition on both $-X$, and $+X$ boundaries. (called "wall boundary condition" in this table)
XM_WALL__XP_PRESSURE_SET	wall boundary condition on $-X$, constant pressure on $+X$
XM_WALL__XP_VELOCITY_SET	wall boundary condition on $-X$, constant velocity on $+X$
XM_PRESSURE_SET__XP_WALL	constant pressure on $-X$, wall boundary condition on $+X$
XM_PRESSURE_SET__XP_PRESSURE_SET	constant pressure on both $-X$ and $+X$ boundaries.
XM_PRESSURE_SET__XP_VELOCITY_SET	constant pressure on $-X$, constant velocity on $+X$
XM_VELOCITY_SET__XP_WALL	constant velocity on $-X$, wall boundary condition on $+X$
XM_VELOCITY_SET__XP_PRESSURE_SET	constant velocity on $-X$, constant pressure on $+X$
XM_VELOCITY_SET__XP_VELOCITY_SET	constant velocity on both $-X$ and $+X$ boundaries
OSCILLATORY_BIASED_PERIODIC	oscillating pressure condition on $+X$, $P = 0$ on $-X$
YM_WALL__YP_WALL	wall boundary condition on both $-Y$, and $+Y$ boundaries.
YM_WALL__YP_PRESSURE_SET	wall boundary condition on $-Y$, constant pressure on $+Y$
YM_WALL__YP_VELOCITY_SET	wall boundary condition on $-Y$, constant velocity on $+Y$
YM_PRESSURE_SET__YP_WALL	constant pressure on $-Y$, wall boundary condition on $+Y$
YM_PRESSURE_SET__YP_PRESSURE_SET	constant pressure on both $-Y$ and $+Y$ boundaries.
YM_PRESSURE_SET__YP_VELOCITY_SET	constant pressure on $-Y$, constant velocity on $+Y$
YM_VELOCITY_SET__YP_WALL	constant velocity on $-Y$, wall boundary condition on $+Y$
YM_VELOCITY_SET__YP_PRESSURE_SET	constant velocity on $-Y$, constant pressure on $+Y$
YM_VELOCITY_SET__YP_VELOCITY_SET	constant velocity on both $-Y$ and $+Y$ boundaries
ZM_WALL__ZP_WALL	wall boundary condition on both $-Z$, and $+Z$ boundaries.
ZM_WALL__ZP_PRESSURE_SET	wall boundary condition on $-Z$, constant pressure on $+Z$
ZM_WALL__ZP_VELOCITY_SET	wall boundary condition on $-Z$, constant velocity on $+Z$
ZM_PRESSURE_SET__ZP_WALL	constant pressure on $-Z$, wall boundary condition on $+Z$
ZM_PRESSURE_SET__ZP_PRESSURE_SET	constant pressure on both $-Z$ and $+Z$ boundaries.
ZM_PRESSURE_SET__ZP_VELOCITY_SET	constant pressure on $-Z$, constant velocity on $+Z$
ZM_VELOCITY_SET__ZP_WALL	constant velocity on $-Z$, wall boundary condition on $+Z$
ZM_VELOCITY_SET__ZP_PRESSURE_SET	constant velocity on $-Z$, constant pressure on $+Z$
ZM_VELOCITY_SET__ZP_VELOCITY_SET	constant velocity on both $-Z$ and $+Z$ boundaries
LEES_EDWARDS_BC	Lees-Edwards boundary condition

4. **Pressure- analysis commands**

Name	**"OUTPUT_SNAPSHOT_IN_AVS_FORMAT"**
Function	Output calculation results on an AVS format file(field-data).

5. **Pressure - evaluation commands**

Name	**"RETURN_TRUE_FUNC"**
Function	Always return "true" flag. This function is used to perform an analysis command with a constant time step interval.

Velocity : velocity field - commands

Velocity	Name
Initialization	"SET_ZERO"
Initialization	"READ_VELOCITY_RAWDATA"
Time evolution	"SOLVE_STOKES_EQUATION_AND_PRESSURE"
Analysis	"OUTPUT_SNAPSHOT_IN_RAW_FORMAT"
Analysis	"OUTPUT_SNAPSHOT_IN_AVS_FORMAT"
Evaluation	"RETURN_TRUE_FUNC"

1. **Velocity - initialization commands**

Name	**"SET_ZERO"**
Function	Set velocity to zero.

Name	**"READ_VELOCITY_RAWDATA"**
Function	Input initial value of velocity from a file. ("OUTPUT_SNAPSHOT_IN_RAW_FORMAT")
Dependent parameter	VELOCITY_RAW_DATA_FILE

2. **Velocity - time evolution commands**

Name	**"SOLVE_STOKES_EQUATION_AND_PRESSURE"**
Function	Solve Stokes flow equation for velocity and pressure; $-\nabla p + \nabla(\eta\{\nabla \boldsymbol{v} + (\nabla \boldsymbol{v})^t\}) + \boldsymbol{K} = 0$
Dependent filed	Pressure
Dependent parameter	SKIP_INTERVAL_VELOCITY_CALCULATION
Dependent parameter	VX_AT_YZ_PLANE_XM
Dependent parameter	VY_AT_YZ_PLANE_XM
Dependent parameter	VZ_AT_YZ_PLANE_XM
Dependent parameter	VX_AT_YZ_PLANE_XP
Dependent parameter	VY_AT_YZ_PLANE_XP
Dependent parameter	VZ_AT_YZ_PLANE_XP
Dependent parameter	VX_AT_ZX_PLANE_YM
Dependent parameter	VY_AT_ZX_PLANE_YM
Dependent parameter	VZ_AT_ZX_PLANE_YM
Dependent parameter	VX_AT_ZX_PLANE_YP
Dependent parameter	VY_AT_ZX_PLANE_YP
Dependent parameter	VZ_AT_ZX_PLANE_YP
Dependent parameter	VX_AT_XY_PLANE_ZM
Dependent parameter	VY_AT_XY_PLANE_ZM
Dependent parameter	VZ_AT_XY_PLANE_ZM
Dependent parameter	VX_AT_XY_PLANE_ZP
Dependent parameter	VY_AT_XY_PLANE_ZP
Dependent parameter	VZ_AT_XY_PLANE_ZP
Dependent parameter	SHEAR_RATE_XZ
Dependent parameter	MONITORING_INTERVAL_OF_VELOCITY_SOLVER
Dependent parameter	ACCELERATION_VALUE_FOR_VELOCITY_SOLVER
Dependent parameter	CONVERGENCE_CRITERION_FOR_VELOCITY_SOLVER
Dependent parameter	MAX_ITERATION_FOR_VELOCITY_SOLVER

3. **Velocity - boundary condition (partial condition) commands**

Partial region condition	treatment
PERIODIC	periodic boundary condition
XM_WALL__XP_WALL	set pressure to make ψ field satisfy wall boundary condition on both $-X$, and $+X$ boundaries. (called "wall boundary condition" in this table)
XM_WALL__XP_PRESSURE_SET	wall boundary condition on $-X$, constant pressure on $+X$
XM_WALL__XP_VELOCITY_SET	wall boundary condition on $-X$, constant velocity on $+X$
XM_PRESSURE_SET__XP_WALL	constant pressure on $-X$, wall boundary condition on $+X$
XM_PRESSURE_SET__XP_PRESSURE_SET	constant pressure on both $-X$ and $+X$ boundaries.
XM_PRESSURE_SET__XP_VELOCITY_SET	constant pressure on $-X$, constant velocity on $+X$
XM_VELOCITY_SET__XP_WALL	constant velocity on $-X$, wall boundary condition on $+X$
XM_VELOCITY_SET__XP_PRESSURE_SET	constant velocity on $-X$, constant pressure on $+X$
XM_VELOCITY_SET__XP_VELOCITY_SET	constant velocity on both $-X$ and $+X$ boundaries
YM_WALL__YP_WALL	wall boundary condition on both $-Y$, and $+Y$ boundaries.
YM_WALL__YP_PRESSURE_SET	wall boundary condition on $-Y$, constant pressure on $+Y$
YM_WALL__YP_VELOCITY_SET	wall boundary condition on $-Y$, constant velocity on $+Y$
YM_PRESSURE_SET__YP_WALL	constant pressure on $-Y$, wall boundary condition on $+Y$
YM_PRESSURE_SET__YP_PRESSURE_SET	constant pressure on both $-Y$ and $+Y$ boundaries.
YM_PRESSURE_SET__YP_VELOCITY_SET	constant pressure on $-Y$, constant velocity on $+Y$
YM_VELOCITY_SET__YP_WALL	constant velocity on $-Y$, wall boundary condition on $+Y$
YM_VELOCITY_SET__YP_PRESSURE_SET	constant velocity on $-Y$, constant pressure on $+Y$
YM_VELOCITY_SET__YP_VELOCITY_SET	constant velocity on both $-Y$ and $+Y$ boundaries
ZM_WALL__ZP_WALL	wall boundary condition on both $-Z$, and $+Z$ boundaries.
ZM_WALL__ZP_PRESSURE_SET	wall boundary condition on $-Z$, constant pressure on $+Z$
ZM_WALL__ZP_VELOCITY_SET	wall boundary condition on $-Z$, constant velocity on $+Z$
ZM_PRESSURE_SET__ZP_WALL	constant pressure on $-Z$, wall boundary condition on $+Z$
ZM_PRESSURE_SET__ZP_PRESSURE_SET	constant pressure on both $-Z$ and $+Z$ boundaries.
ZM_PRESSURE_SET__ZP_VELOCITY_SET	constant pressure on $-Z$, constant velocity on $+Z$
ZM_VELOCITY_SET__ZP_WALL	constant velocity on $-Z$, wall boundary condition on $+Z$
ZM_VELOCITY_SET__ZP_PRESSURE_SET	constant velocity on $-Z$, constant pressure on $+Z$
ZM_VELOCITY_SET__ZP_VELOCITY_SET	constant velocity on both $-Z$ and $+Z$ boundaries
LEES_EDWARDS_BC	Lees-Edwards boundary condition

4. **Velocity - analysis commands**

Name	**"OUTPUT_SNAPSHOT_IN_RAW_FORMAT"**
Function	Output calculation results on a file.
Dependent parameter	VELOCITY_RAW_DATA_FILE

Name	**"OUTPUT_SNAPSHOT_IN_AVS_FORMAT"**
Function	Output calculation results on an AVS format file(field-data).

5. **Velocity - evaluation commands**

Name	**"RETURN_TRUE_FUNC"**
Function	Always return "true" flag. This function is used to perform an analysis command with a constant time step interval.

VolumeFraction : volume fraction field - commands

VolumeFraction	Name
Initialization	"READ_AVS_DATA"
Initialization	"ADD_NOISE"
Initialization	"CONSTANT_VOLUME_FRACTION"
Initialization	"CONSTANT_VOLUME_FRACTION_WITH_NOISE"
Initialization	"LINEAR_ALONG_X_DIRECTION"
Initialization	"LINEAR_ALONG_Y_DIRECTION"
Initialization	"LINEAR_ALONG_Z_DIRECTION"
Initialization	"SET_DROPLETS"
Initialization	"ADD_AFFINE_DEFORMATION:SHEAR"
Initialization	"SET_LAMELLAE"
Time evolution	"SOLVE_EQUATION_OF_CONTINUITY_WITH_FLOW"
Time evolution	"SOLVE_EQUATION_OF_CONTINUITY_WITHOUT_FLOW"
Time evolution	"SOLVE_OHTA_KAWASAKI_2"
Analysis	"OUTPUT_SNAPSHOT_IN_AVS_FORMAT"
Evaluation	"RETURN_TRUE_FUNC"
Evaluation	"TRUE_AT_A_CONSTANT_TIME_INTERVAL"

1. **VolumeFraction - initialization commands**

Name	**"READ_AVS_DATA"**
Function	Read initial value from a file in AVS format.
Dependent parameter	AVS_DATA_FILE_NAME

Name	**"ADD_NOISE"**
Function	Add random noise.
Dependent parameter	DEVIATION_FROM_AVERAGED_VOLUME_FRACTION
Dependent parameter	SEED_OF_RANDOM_NUMBER

Name	**"CONSTANT_VOLUME_FRACTION"**
Function	Initialize volume fraction of each component to a constant value. Boundary values are set according to boundary conditions.
Dependent parameter	AVERAGED_VOLUME_FRACTION
Dependent parameter	VOLUME_FRACTION_GRADIENT_ALONG_X
Dependent parameter	VOLUME_FRACTION_GRADIENT_ALONG_Y
Dependent parameter	VOLUME_FRACTION_GRADIENT_ALONG_Z
Dependent parameter	BULK_VOLUME_FRACTION
Dependent parameter	REMOVE_FRACTION_OF_SOLVENT

Name	**"CONSTANT_VOLUME_FRACTION_WITH_NOISE"**
Function	Initialize volume fraction of each component to a constant value, and add noise. Boundary values are set according to boundary conditions.
Dependent parameter	VOLUME_FRACTION_GRADIENT_ALONG_X
Dependent parameter	VOLUME_FRACTION_GRADIENT_ALONG_Y
Dependent parameter	VOLUME_FRACTION_GRADIENT_ALONG_Z
Dependent parameter	BULK_VOLUME_FRACTION
Dependent parameter	REMOVE_FRACTION_OF_SOLVENT
Dependent parameter	DEVIATION_FROM_AVERAGED_VOLUME_FRACTION
Dependent parameter	SEED_OF_RANDOM_NUMBER
Dependent parameter	AVERAGED_VOLUME_FRACTION

Name	**"LINEAR_ALONG_X_DIRECTION"**
Function	Initialize to have a constant gradient in X direction.
Dependent parameter	VOLUME_FRACTION_GRADIENT_ALONG_X
Dependent parameter	AVERAGED_VOLUME_FRACTION

Name	**"LINEAR_ALONG_Y_DIRECTION"**
Function	Initialize to have a constant gradient in Y direction.
Dependent parameter	VOLUME_FRACTION_GRADIENT_ALONG_Z
Dependent parameter	AVERAGED_VOLUME_FRACTION

Name	**"LINEAR_ALONG_Z_DIRECTION"**
Function	Initialize to have a constant gradient in Z direction.
Dependent parameter	VOLUME_FRACTION_GRADIENT_ALONG_Z
Dependent parameter	AVERAGED_VOLUME_FRACTION

Name	**"SET_DROPLETS"**
Function	Put droplets with specified positions and radii (2-component system only)
Dependent parameter	NUMBER_OF_COMPONENTS
Dependent parameter	NUMBER_OF_DROPLETS
Dependent parameter	RADIUS_OF_DROPLET
Dependent parameter	X_COORDINATE_OF_DROPLET
Dependent parameter	Y_COORDINATE_OF_DROPLET
Dependent parameter	Z_COORDINATE_OF_DROPLET

Name	**"ADD_AFFINE_DEFORMATION:SHEAR"**
Function	Add shear deformation for current distribution. (in XY plane , add shear in X direction) Boundary values are set according to boundary conditions.
Dependent parameter	INITIAL_SHEAR_STRAIN_ZX
Dependent parameter	VOLUME_FRACTION_GRADIENT_ALONG_X
Dependent parameter	VOLUME_FRACTION_GRADIENT_ALONG_Y
Dependent parameter	VOLUME_FRACTION_GRADIENT_ALONG_Z
Dependent parameter	BULK_VOLUME_FRACTION
Dependent parameter	REMOVE_FRACTION_OF_SOLVENT

Name	**"SET_LAMELLAE"**
Function	Initialize with lamellae structure (2-component system only)
Dependent parameter	NUMBER_OF_LAMELLAE
Dependent parameter	DIRECTION_OF_LAMELLAE
Dependent parameter	PHASE_OF_LAMELLAE

2. **VolumeFraction - time evolution commands**

Name	**"SOLVE_EQUATION_OF_CONTINUITY_WITH_FLOW"**
Function	One step time integration of equation for volume fraction $\partial\psi_\alpha/\partial t = -g_0\nabla\cdot(\psi_\alpha \boldsymbol{v}) - \nabla\cdot\boldsymbol{J}_\alpha$
Dependent filed	Velocity
Dependent filed	K_Field
Dependent parameter	DT
Dependent parameter	VOLUME_FRACTION_GRADIENT_ALONG_X
Dependent parameter	VOLUME_FRACTION_GRADIENT_ALONG_Y
Dependent parameter	VOLUME_FRACTION_GRADIENT_ALONG_Z
Dependent parameter	BULK_VOLUME_FRACTION
Dependent parameter	REMOVE_FRACTION_OF_SOLVENT

Name	**"SOLVE_EQUATION_OF_CONTINUITY_WITHOUT_FLOW"**
Function	One step time integration of equation for volume fraction $\partial\psi_\alpha/\partial t = -\nabla\cdot\boldsymbol{J}_\alpha$
Dependent filed	K_Field
Dependent parameter	DT
Dependent parameter	VOLUME_FRACTION_GRADIENT_ALONG_X
Dependent parameter	VOLUME_FRACTION_GRADIENT_ALONG_Y
Dependent parameter	VOLUME_FRACTION_GRADIENT_ALONG_Z
Dependent parameter	BULK_VOLUME_FRACTION
Dependent parameter	REMOVE_FRACTION_OF_SOLVENT

Name	**"SOLVE_OHTA_KAWASAKI_2"**
Function	2 compontents Ohta-Kawasaki model(Proc. of SPIE, 8680, 86801I (2013)). One step time integration of equation for volume fraction $\partial\eta/\partial t = -\nabla\cdot\boldsymbol{J} - \alpha\eta$, where $\eta = 2\psi_0 - 1$ "GRADIENT_CHEMICAL_POTENTIAL_ORDER_PARAMETER" should be used for K_Field command. "OHTA_KAWASAKI_2" should be used for ChemicalPotential command. Diffusion coefficient parameter L is not necessary in physical parameter because definition of parameter DT is $L\Delta t$. $\alpha = \frac{3}{f(1-f)}$, where f is number fraction of component 0 in diblock-copolymer.
Dependent filed	VolumeFraction
Dependent filed	K_Field
Dependent parameter	DT
Dependent parameter	ALPHA_OHTA_KAWASAKI

3. **VolumeFraction - boundary condition (partial condition) commands**

Partial region condition	treatment
PERIODIC	periodic boundary condition
BIASED_PERIODIC	biased Periodic boundary condition
XM_WALL__XP_WALL	wall boundary condition on both $-X$ and $+X$ boundaries.
XM_WALL__XP_BULK	wall boundary condition on $-X$, bulk boundary condition on $+X$.
XM_BULK__XP_WALL	bulk boundary condition on $-X$, wall boundary condition on $+X$.
XM_BULK__XP_BULK	bulk boundary condition on both $-X$ and $+X$ boundaries.
YM_WALL__YP_WALL	wall boundary condition on both $-Y$ and $+Y$ boundaries.
YM_WALL__YP_BULK	wall boundary condition on $-Y$, bulk boundary condition on $+Y$.
YM_BULK__YP_WALL	bulk boundary condition on $-Y$, wall boundary condition on $+Y$.
YM_BULK__YP_BULK	bulk boundary condition on both $-Y$ and $+Y$ boundaries.
ZM_WALL__ZP_WALL	wall boundary condition on both $-Z$ and $+Z$ boundaries.
ZM_WALL__ZP_BULK	wall boundary condition on $-Z$, bulk boundary condition on $+Z$.
ZM_BULK__ZP_WALL	bulk boundary condition on $-Z$, wall boundary condition on $+Z$.
ZM_BULK__ZP_BULK	bulk boundary condition on both $-Z$ and $+Z$ boundaries.

4. **VolumeFraction- analysis commands**

Name	**"OUTPUT_SNAPSHOT_IN_AVS_FORMAT"**
Function	Output calculation results on an AVS format file(field-data).

5. **VolumeFraction - evaluation commands**

Name	**"RETURN_TRUE_FUNC"**
Function	Always return "true" flag. This function is used to perform an analysis command with a constant time step interval.

Name	**"TRUE_AT_A_CONSTANT_TIME_INTERVAL"**
Function	Return "true" flag at a constant time interval.
Dependent parameter	FINAL_STEP
Dependent parameter	DIVISION_NUM1

Obstacle : obstacle field - commands

Obstacle	Name
Initialization	"READ_OBSTACLE_DATA"
Initialization	"SET_BOUNDARY_CONDITION"

1. **Obstacle - initialization commands**

Name	**"READ_OBSTACLE_DATA"**
Function	Initialize Obstacle field by reading a file.
Dependent parameter	OBSTACLE_DATA_FILE

Name	**"SET_BOUNDARY_CONDITION"**
Function	Set periodic boundary condition for Obstacle. This command is necessary when a part of surface of obstacle is on a boundary.

2. **Obstacle - boundary condition (partial condition) commands**

Partial region condition	treatment
PERIODIC	periodic boundary condition

Diffuion_Flux : Diffusion flux on the phase separated strucure - commands

This field is used for diffusion calculation on the phase separated structure (e.g. heat conduction). The phase separated structure can be imported by using Action functuin and the strucrure is fixed during the calculation.

Diffusion_Flux	Name
Time evolution 0	"GRADIENT_DIFFUSION_FIELD"

1. **Diffusion_Flux - time evolution commands**

Name	**"GRADIENT_DIFFUSION_FIELD"**
Function	Calculation of the diffusion flux by the gradient of the diffusion field. For the temperature, $\boldsymbol{J} = -D\nabla T$, where D is the local diffusion coefficient.
Dependent parameter	DIFFUSION_COEFFICIENT When starting the calculation, automatically, local diffusion coefficient is calculated by using the volume fraction distribution. $D(x) = \sum_\alpha \psi_\alpha(x) D_\alpha$

2. **Diffusion_Flux - boundary condition (partial condition) commands**

Partial region condition	treatment
PERIODIC	periodic boundary condition
XM_WALL__XP_WALL	wall boundary condition on both $-X$ and $+X$ boundaries.
YM_WALL__YP_WALL	wall boundary condition on both $-Y$ and $+Y$ boundaries.
ZM_WALL__ZP_WALL	wall boundary condition on both $-Z$ and $+Z$ boundaries.

Diffuion_Field : Diffusion field on the phase separated strucure - commands

This field is used for diffusion calculation on the phase separated structure (e.g. heat conduction). The phase separated structure can be imported by using Action functuin and the strucrure is fixed during the calculation.

Diffusion_Field	Name
Initialization 0	"CONSTANT_DIFFUSION_FIELD"
Time evolution 0	"SOLVE_DIFFUSION"

1. **Diffusion_Field - initialization commands**

Name	**"CONSTANT_DIFFUSION_FIELD"**
Function	Initialize the diffusion field to a constant value.
Dependent parameter	INITIAL_DIFFUSION_FIELD

2. **Diffusion_Field - time evolution commands**

Name	**"SOLVE_DIFFUSION"**
Function	Time evolution of the diffusion field by using the diffusion flux. For the temperature, $\partial T/\partial t = -\nabla \cdot \boldsymbol{J}$

3. **Diffusion_Field - boundary condition (partial condition) commands**

Partial region condition	treatment
PERIODIC	periodic boundary condition
XM_WALL__XP_WALL	wall boundary condition on both $-X$ and $+X$ boundaries.
YM_WALL__YP_WALL	wall boundary condition on both $-Y$ and $+Y$ boundaries.
ZM_WALL__ZP_WALL	wall boundary condition on both $-Z$ and $+Z$ boundaries.

3.2 Commands and parameters for fields of PhaseSeparation_FEM

3.2.1 Input parameters of PhaseSeparation_FEM

Name of Parameters	Meanings and notations in theory
NUMBER_OF_COMPONENTS	number of components
VALENCY	valence of each ion component Z_α
Z	valence of each ion component Z_α
POLYMERIZATION_INDEX_N	polymerization index of each component N_α
CHI_mn	χ-parameter of components n,m (only for $m < n$)
DIELECTRIC_CONSTANT	relative dielectric constants ϵ_α Give for each component.
B	dimensionless electric energy B
CHARGE_DENSITY	charge density of each component ρ_α
MATRIX_SOLVER	matrix solver name to be used in pressure solution. Either "ICCG" or "CG" Default is "ICCG".
MATRIX_SOLVER _FOR_ELECTRIC_FIELD	matrix solver name to be used for electric field Either "ICCG" or "CG" Default is "ICCG".
PENALTY_NUMBER_FOR_DIRICHLET_BC	A penalty number to handle Dirichlet condition (a very large number). The default value is 10^{13}.
GRAVITY_X	X component of external force on fluid.
GRAVITY_Y	Y component of external force on fluid.
GRAVITY_Z	Z component of external force on fluid.
DIMENSIONLESS_GRAVITY	gravitational acceleration g.
DIMENSIONLESS_DENSITY	mass density of each component ρ_α.
REYNOLDS	Reynolds number.
DT_FOR_V	time step interval for Stokes flow calculation.
CA	Capillary number
MAX_ITERATION_FOR _VELOCITY_SOLVER	maximum number of iterations for Stokes flow calculation
CONVERGENCE_CRITERION_FOR _VELOCITY_SOLVER	convergence criterion for Stokes flow calculation > 0 : monitor relative change of velocity (default:1.0^{-3}) < 0 : monitor absolute change of velocity (default:1.0^{-3})
VISCOSITY	viscosity coefficient of each component η_α
SKIP_INTERVAL _VELOCITY_CALCULATION	time step interval for velocity calculation
AVERAGED_VOLUME_FRACTION	averaged volume fraction as initial value $\psi_{\alpha 0}$
DEVIATION_FROM_AVERAGED _VOLUME_FRACTION	magnitude of noise given to initial value of volume fraction
SEED_OF_RANDOM_NUMBER	initial random number for random initialization of volume fraction field.
NUMBER_OF_DROPLETS	number of droplets placed in initialization of volume fraction.
RADIUS_OF_DROPLET	radius of each droplet placed in initialization of volume fraction.
X_COORDINATE_OF_DROPLET	X coordinate of each droplet placed in initialization of volume fraction.
Y_COORDINATE_OF_DROPLET	Y coordinate of each droplet placed in initialization of volume fraction.
Z_COORDINATE_OF_DROPLET	Z coordinate of each droplet placed in initialization of volume fraction.
DIFFUSION_COEFFICIENT	diffusion coefficient of each component L_α

3.2.2 PhaseSeparation_FEM - list of fields

Name of Parameters	Meanings and notations in theory
VolumeFraction	Volume fraction field ψ_α
ChemicalPotential	Chemical potential field μ_α
Velocity	Velocity field $\boldsymbol{v}$
K_Field	Flux field $\boldsymbol{J}_\alpha(\boldsymbol{K})$
Viscosity	Viscosity field η
Pressure	Pressure field P
ElectricPotential	Electric potential field Φ
Obstacle	Obstacle filed (defined on FEM cells)

The Obstacle filed is defined on FEM cells, and all other fields are defined on vertex.
K_Field $\alpha = 0$ component of K_Field may be used to store volume force field $\boldsymbol{K}$ in fluid equation.

3.2.3 PhaseSeparation_FEM - commands

ChemicalPotential : chemical potential field - commands

ChemicalPotential	Name
Time evolution	"SET_ZERO"
Time evolution	"SYMMETRIC_GL"
Time evolution	"FLORY_HUGGINS"
Time evolution	"ADD_ELECTRIC_EFFECT_OF_DIELECTRIC_MEDIUM"
Time evolution	"ADD_ELECTROSTATIC_EFFECT_USING_CHARGE_DENSITY"
Time evolution	"ADD_EFFECT_OF_GRAVITY"
Analysis	"OUTPUT_SNAPSHOT_IN_AVS_FORMAT"
Evaluation	"RETURN_TRUE_FUNC"

1. **ChemicalPotential - time evolution commands**

Name	**"SET_ZERO"**
Function	Set field values to zero.
Dependent parameter	NUMBER_OF_COMPONENTS

Name	**"SYMMETRIC_GL"**
Function	Two-component Ginzburg-Landau model $\mu = -\psi + \psi^3 - \Delta\psi$ ($\psi \equiv \psi_1 - \psi_0$)
Dependent field	VolumeFraction
Dependent parameter	NUMBER_OF_COMPONENTS

Name	**"FLORY_HUGGINS"**
Function	Flory-Huggins free energy.
Dependent field	VolumeFraction
Dependent parameter	NUMBER_OF_COMPONENTS
Dependent parameter	CHI_mn
Dependent parameter	POLYMERIZATION_INDEX_N

Name	**"ADD_ELECTRIC_EFFECT_OF_DIELECTRIC_MEDIUM"**
Function	By using dielectric constants depending on volume fraction field and electric potential field Φ, take an effect of dielectric fluid under electric field into chemical potential.
Dependent field	ElectricPotential
Dependent parameter	NUMBER_OF_COMPONENTS
Dependent parameter	DIELECTRIC_CONSTANT
Dependent parameter	B

Name	**"ADD_ELECTROSTATIC_EFFECT_USING_ CHARGE_DENSITY"**
Function	Add effect of charge distribution and electric potential field to chemical potential; $(\partial\rho_e/\partial\psi)\Phi$
Dependent field	ElectricPotential
Dependent parameter	NUMBER_OF_COMPONENTS
Dependent parameter	CHARGE_DENSITY

Name	**"ADD_EFFECT_OF_GRAVITY"**
Function	Add effect of gravity $g(\rho_\alpha - \rho_0)z$
Dependent parameter	NUMBER_OF_COMPONENTS
Dependent parameter	DIMENSIONLESS_GRAVITY
Dependent parameter	DIMENSIONLESS_DENSITY

2. **ChemicalPotential - analysis commands**

Name	**"OUTPUT_SNAPSHOT_IN_AVS_FORMAT"**
Function	Output calculation results on an AVS format file(ucd-data).

3. **ChemicalPotential - evaluation commands**

Name	**"RETURN_TRUE_FUNC"**
Function	Always return "true" flag. This function is used to perform an analysis command with a constant time step interval.

ElectricPotential : electric potential field - commands

ElectricPotential	Name
Time evolution	"ELECTRIC_POTENTIAL_SOLVER"
Analysis	"OUTPUT_SNAPSHOT_IN_AVS_FORMAT"
Evaluation	"RETURN_TRUE_FUNC"

1. **ElectricPotential - time evolution commands**

Name	**"ELECTRIC_POTENTIAL_SOLVER"**
Function	Solve Poisson equation for electric potential $\nabla\cdot(\epsilon\nabla\Phi) = -\rho_e$
Dependent field	VolumeFraction
Dependent parameter	CHARGE_DENSITY
Dependent parameter	DIELECTRIC_CONSTANT
Dependent parameter	NUMBER_OF_COMPONENTS
Dependent parameter	PENALTY_NUMBER
Dependent parameter	MATRIX_SOLVER_FOR_ELECTRIC_FIELD
Dependent parameter	MATRIX_SOLVER

2. **ElectricPotential - partial region condition (boundary condition) commands**

Partial region condition	treatment
I_CONSTANT_VALUE	initialize to a constant value.
D	set to a constant value (Dirichlet condition)
N	$\boldsymbol{n}\cdot(\epsilon\nabla\Phi) = \bar{\boldsymbol{D}}\cdot\boldsymbol{n}$: give surface charge density (Neumann condition)

3. **ElectricPotential - analysis commands**

Name	**"OUTPUT_SNAPSHOT_IN_AVS_FORMAT"**
Function	Output calculation results on an AVS format file(ucd-data).

4. **ElectricPotential - evaluation commands**

Name	**"RETURN_TRUE_FUNC"**
Function	Always return "true" flag. This function is used to perform an analysis command with a constant time step interval.

K_Field : flux field - commands

K_Field	Name
Initialization	"SET_ZERO"
Initialization	"SET_CONSTANT_FORCE"
Time evolution	"SET_ZERO"
Time evolution	"GRADIENT_CHEMICAL_POTENTIAL"
Time evolution	"APPLY_PARTIAL_REGION_CONDITION"
Analysis	"OUTPUT_SNAPSHOT_IN_AVS_FORMAT"
Evaluation	"RETURN_TRUE_FUNC"

1. **K_Field - initialization commands**

Name	**"SET_ZERO"**
Function	Set field values to zero.
Dependent parameter	NUMBER_OF_COMPONENTS

Name	**"SET_CONSTANT_FORCE"**
Function	Apply a constant external force.
Dependent parameter	DIMENSION_OF_SPACE
Dependent parameter	GRAVITY_X
Dependent parameter	GRAVITY_Y
Dependent parameter	GRAVITY_Z

2. **K_Field - time evolution commands**

Name	**"SET_ZERO"**
Function	Set field values to zero.
Dependent parameter	NUMBER_OF_COMPONENTS

Name	**"GRADIENT_CHEMICAL_POTENTIAL"**
Function	Calculate flux for each component from volume fraction and chemical potential $\boldsymbol{K}_\alpha = -\psi_\alpha \nabla \mu_\alpha$
Dependent field	VolumeFraction
Dependent field	ChemicalPotential
Dependent parameter	NUMBER_OF_COMPONENTS

Name	**"APPLY_PARTIAL_REGION_CONDITION"**
Function	Apply partial region conditions.
Dependent parameter	NUMBER_OF_COMPONENTS

3. **K_Field - partial region condition (boundary condition) commands**

Partial region condition	treatment
D_CONSTANT_VALUE_FOR_A_COMPONENT	set flux of a component to a constant value (Dirichlet condition) give component index α, $\boldsymbol{J}_{\alpha x}$, $\boldsymbol{J}_{\alpha y}$ and $\boldsymbol{J}_{\alpha z}$ as data part.

4. **K_Field - analysis commands**

Name	**"OUTPUT_SNAPSHOT_IN_AVS_FORMAT"**
Function	Output calculation results on an AVS format file(ucd-data).

5. **K_Field - evaluation commands**

Name	**"RETURN_TRUE_FUNC"**
Function	Always return "true" flag. This function is used to perform an analysis command with a constant time step interval.

Pressure : pressure field - commands

Pressure	Name
Initialization	"SET_ZERO"
Time evolution	"SOLVE_PRESSURE"
Analysis	"OUTPUT_SNAPSHOT_IN_AVS_FORMAT"
Evaluation	"RETURN_TRUE_FUNC"

1. **Pressure - initialization commands**

Name	**"SET_ZERO"**
Function	Set field values to zero.

2. **Pressure - time evolution commands**

Name	**"SOLVE_PRESSURE"**
Function	Solve Poisson equation for pressure, $\nabla^2 p = \frac{1}{\Delta t}\nabla \cdot \boldsymbol{v}^*$
Dependent field	Pressure
Dependent field	Velocity
Dependent parameter	DT_FOR_V
Dependent parameter	DIMENSION_OF_SPACE
Dependent parameter	PENALTY_NUMBER
Dependent parameter	MATRIX_SOLVER

3. **Pressure - partial region condition (boundary condition) commands**

Partial region condition	treatment
I_CONSTANT_VALUE	initialize to a constant value.
D	set to a constant value (Dirichlet condition)
N	$\boldsymbol{n} \cdot \nabla P = \bar{P}_n$ (Neumann condition)

4. **Pressure - analysis commands**

Name	**"OUTPUT_SNAPSHOT_IN_AVS_FORMAT"**
Function	Output calculation results on an AVS format file(ucd-data).

5. **Pressure - evaluation commands**

Name	**"RETURN_TRUE_FUNC"**
Function	Always return "true" flag. This function is used to perform an analysis command with a constant time step interval.

Velocity : velocity field - commands

Velocity	Name
Initialization	"SET_ZERO"
Initialization	"SET_DIRICHLET_CONDITION"
Time evolution	"SOLVE_VELOCITY_AND_PRESSURE"
Time evolution	"SOLVE_STOKES_EQUATION_AND_PRESSURE"
Boundary condition	"SET_DIRICHLET_CONDITION"
Evaluation	"RETURN_TRUE_FUNC"

1. **Velocity - initialization commands**

Name	**"SET_ZERO"**
Function	Set field values to zero.

Name	**"SET_DIRICHLET_CONDITION"**
Function	Initialize using Dirichlet boundary conditions in partial region conditions.
Dependent field	Velocity

2. **Velocity - time evolution commands**

Name	**"SOLVE_VELOCITY_AND_PRESSURE"**
Function	One time step evolution of velocity by Navier Stokes equation, $Re\frac{\partial \boldsymbol{v}}{\partial t} = -\nabla p + \nabla(\eta\{\nabla \boldsymbol{v} + (\nabla \boldsymbol{v})^t\}) + C_a^{-1}\boldsymbol{K}$ Default value of Re is 1.0.
Dependent field Dependent field Dependent field	Pressure K_Field Viscosity
Dependent parameter Dependent parameter Dependent parameter Dependent parameter	DT REYNOLDS NUMBER_OF_COMPONENTS CA

Name	**"SOLVE_STOKES_EQUATION_AND_PRESSURE"**
Function	Solve Stokes flow equation for velocity and pressure; $\nabla p = \nabla(\eta\{\nabla \boldsymbol{v} + (\nabla \boldsymbol{v})^t\}) + C_a^{-1}\boldsymbol{K}$
Dependent field	Pressure
Dependent field	Velocity
Dependent field	K_Field
Dependent field	Viscosity
Dependent parameter	DT
Dependent parameter	DT_FOR_V
Dependent parameter	SKIP_INTERVAL_VELOCITY_CALCULATION
Dependent parameter	MAX_ITERATION_FOR_VELOCITY_SOLVER
Dependent parameter	CONVERGENCE_CRITERION_FOR_VELOCITY_SOLVER
Dependent parameter	DIMENSION_OF_SPACE
Dependent parameter	NUMBER_OF_COMPONENTS
Dependent parameter	CA

3. **Velocity - partial region condition (boundary condition) commands**

Name	**"SET_DIRICHLET_CONDITION"**
Function	Apply Dirichlet boundary conditions in partial region conditions. When your are using command "SOLVE_VELOCITY_AND_PRESSURE" or "SOLVE_STOKES_EQUATION_AND_PRESSURE", partial region conditions are applied automatically, so you may not need to use this command explicitly.

4. **Velocity - partial region condition (boundary condition) commands**

Partial region condition	treatment
D_VX	set v_x to a constant value (Dirichlet condition)
D_VY	set v_y to a constant value (Dirichlet condition)
D_VZ	set v_z to a constant value (Dirichlet condition)

5. **Velocity - analysis commands**

Name	**"OUTPUT_SNAPSHOT_IN_AVS_FORMAT"**
Function	Output calculation results on an AVS format file(ucd-data).

6. **Velocity - evaluation commands**

Name	**"RETURN_TRUE_FUNC"**
Function	Always return "true" flag. This function is used to perform an analysis command with a constant time step interval.

Viscosity : viscosity field - commands

Viscosity	Name
Time evolution	"CONSTANT_VISCOSITY"
Time evolution	"VISCOSITY_DEPENDING_ON_VOLUME_FRACTION"

1. **Viscosity - initialization commands**

2. **Viscosity - time evolution commands**

Name	**"CONSTANT_VISCOSITY"**
Function	Set uniform viscosity coefficient with η_0 (component 0).
Dependent parameter	VISCOSITY

Name	**"VISCOSITY_DEPENDING_ON_VOLUME_FRACTION"**
Function	calculate viscosity coefficient from that of each component weighting by volume fraction $\eta = \sum_\alpha \eta\alpha\psi_\alpha$
Dependent field	VolumeFraction
Dependent parameter	NUMBER_OF_COMPONENTS
Dependent parameter	VISCOSITY

VolumeFraction : volume fraction field - commands

VolumeFraction	Name
Initialization	"INITIALIZE_BY_PARTIAL_REGION_CONDITION"
Initialization	"CONSTANT_VOLUME_FRACTION"
Initialization	"ADD_NOISE"
Initialization	"CONSTANT_VOLUME_FRACTION_WITH_NOISE"
Initialization	"UNIFORM_CONCENTRATION"
Initialization	"SET_DROPLETS"
Time evolution	"APPLY_PARTIAL_REGION_CONDITION"
Time evolution	"SOLVE_EQUATION_OF_CONTINUITY_WITH_FLOW"
Time evolution	"SOLVE_EQUATION_OF_CONTINUITY_WITHOUT_FLOW"
Analysis	"OUTPUT_SNAPSHOT_IN_AVS_FORMAT"
Evaluation	"RETURN_TRUE_FUNC"

1. **VolumeFraction - initialization commands**

Name	**"INITIALIZE_BY_PARTIAL_REGION_CONDITION"**
Function	Initialize field using initialization partial region conditions ("I_xxx").
Dependent parameter	NUMBER_OF_COMPONENTS

Name	**"CONSTANT_VOLUME_FRACTION"**
Function	Initialize volume fraction of each component to a constant value.
Dependent parameter	NUMBER_OF_COMPONENTS
Dependent parameter	AVERAGED_VOLUME_FRACTION

Name	**"ADD_NOISE"**
Function	Add random noise.
Dependent parameter	NUMBER_OF_COMPONENTS
Dependent parameter	DEVIATION_FROM_AVERAGED_VOLUME_FRACTION
Dependent parameter	SEED_OF_RANDOM_NUMBER

Name	**"CONSTANT_VOLUME_FRACTION_WITH_NOISE"**
Function	Initialize volume fraction of each component to a constant value, and add noise.
Dependent parameter	NUMBER_OF_COMPONENTS
Dependent parameter	AVERAGED_VOLUME_FRACTION
Dependent parameter	DEVIATION_FROM_AVERAGED_VOLUME_FRACTION
Dependent parameter	SEED_OF_RANDOM_NUMBER

Name	**"UNIFORM_CONCENTRATION"**
Function	Initialize concentration of each ion component to a constant value.
Dependent parameter	NUMBER_OF_COMPONENTS
Dependent parameter	AVERAGED_ION_CONCENTRATION

Name	**"SET_DROPLETS"**
Function	Put droplets with specified positions and radii (2-component system only).
Dependent parameter	NUMBER_OF_COMPONENTS
Dependent parameter	NUMBER_OF_DROPLETS
Dependent parameter	RADIUS_OF_DROPLET
Dependent parameter	X_COORDINATE_OF_DROPLET
Dependent parameter	Y_COORDINATE_OF_DROPLET
Dependent parameter	Z_COORDINATE_OF_DROPLET

2. **VolumeFraction - time evolution commands**

Name	**"APPLY_PARTIAL_REGION_CONDITION"**
Function	Apply partial region conditions.
Dependent parameter	NUMBER_OF_COMPONENTS

Name	**"SOLVE_EQUATION_OF_CONTINUITY_WITH_FLOW"**
Function	One step time integration of equation for volume fraction $\partial\psi_\alpha/\partial t = -g_0\nabla\cdot(\psi_\alpha \boldsymbol{v}) - \nabla\cdot\boldsymbol{J}_\alpha$
Dependent field	K_Field
Dependent field	Velocity
Dependent parameter	DT
Dependent parameter	NUMBER_OF_COMPONENTS
Dependent parameter	DIFFUSION_COEFFICIENT

Name	**"SOLVE_EQUATION_OF_CONTINUITY_WITHOUT_FLOW"**
Function	One step time integration of equation for volume fraction $\partial\psi_\alpha/\partial t = -\nabla\cdot\boldsymbol{J}_\alpha$
Dependent field	K_Field
Dependent parameter	DT
Dependent parameter	NUMBER_OF_COMPONENTS
Dependent parameter	DIFFUSION_COEFFICIENT

3. **VolumeFraction - partial region condition (boundary condition) commands**

Partial region condition	treatment
I_CONSTANT_VALUE_FOR_A_COMPONENT	initialize ψ_α to a constant value. Give component index α and ψ_α as data part.
D_CONSTANT_VALUE_FOR_A_COMPONENT	set ψ_α to a constant value (Dirichlet condition). Give component index α and ψ_α as data part.

4. **VolumeFraction - analysis commands**

Name	**"OUTPUT_SNAPSHOT_IN_AVS_FORMAT"**
Function	Output calculation results on an AVS format file(ucd-data).

5. **VolumeFraction - evaluation commands**

Name	**"RETURN_TRUE_FUNC"**
Function	Always return "true" flag. This function is used to perform an analysis command with a constant time step interval.