Manual for using the finite difference solver for solidification using OpenCL

Dasari Mohan and Gandham Phanikumar

November 4, 2021

1 Introduction

This is a multicomponent phase-field solver based on a regular-grid finite-difference discretization, with a simple Euler forward time marching scheme. It is based on the phase-field model presented in *Acta Materialia 55, 4391 (2007).* The solver is parallelized based on OpenCL v1.2 to run across different platforms viz. CPU and GPU (AMD® and Nvidia®) platforms. Before, you run the solver there are certain basic operations one needs to carry out.

2 Infile

The input parameters required for the solver are derived from an infile. This contains information about the domain geometry, the thermodynamic functions(free energy), material properties such as the interfacial energies and their anisotropies as well as boundary conditions. It could also contain special flags related to the running of the solver. The following is a basic description of the keys in the infile. **Each key must end with a semicolon**. Additionally, all lines beginning with "#" will be treated as comments in the infile.

```
##Geometrical dimensions of the simulation domain
DIMENSION = 2;
MESH_X = 256;
MESH_Y = 256;
MESH_Z = 1;
##Discretization, space and time
DELTA_X = 40.0;
DELTA_Y = 40.0;
DELTA_Z = 1.0;
DELTA_t = 320.0;
##Number of phases and composition
NUMPHASES = 2;
NUMCOMPONENTS = 2;
#Running and saving information
NTIMESTEPS = 2000;
NSMOOTH = 10;
SAVET = 200;
## Component and Phase names
## List of components in the alloy system
## {x1,...,xn}: x1 is 1st alloying element, xn is solvent (matrix
    ↪ ) element
## Name of the element should be in accordance with TDB file
COMPONENTS = {NB, NI};
## List of phases in the alloy system
## {phase1,...,phasen}: phase1 is 1st phase, phasen is matrix
    ↪ phase
## Name of the phase should be in accordance with TDB file
PHASES = {FCC_A1, LIQUID};
##Material properties
##GAMMA={12, 13, 14, 23, 24...}
GAMMA = {0.37};
# Diffusivity = {Diagonal:0/1, phase, 11,22,33, 12, 13, 23...};
DIFFUSIVITY = {1, 0, 4.31e-9};
DIFFUSIVITY = {1, 1, 1e-12};
##Gas constant and molar volume
R = 8.314;
V = 7.43e-6;
##Boundary conditions
#0: Free, 1: Neumann, 2: Dirichlet, 3: Periodic, 4: Complex
#Boundary = {phase, X+, X-, Y+, Y-, Z+, Z-}
BOUNDARY = {phi, 1, 1, 1, 1, 0, 0};
BOUNDARY = {mu, 1, 1, 1, 1, 0, 0};
BOUNDARY = {c, 1, 1, 1, 1, 0, 0};
BOUNDARY = {T, 1, 1, 1, 1, 0, 0};
# Boundary = {phi, 1, 1, 0};
# Boundary = {"u", 3, 3, 2, 2};
#Boundary_value = {Value X+, Value X-, Value Y+, Value Y-, Value
    ↪ Z+, Value Z-}
BOUNDARY_VALUE = {phi, 0, 0, 0, 0, 0, 0};
BOUNDARY_VALUE = {mu, 0, 0, 0, 0, 0, 0};
BOUNDARY_VALUE = {c, 0, 0, 0, 0, 0, 0};
BOUNDARY_VALUE = {T, 0, 0, 0, 0, 0, 0};
##Type of simulation
ISOTHERMAL = 1;
BINARY = 1;
T = 1690.0;
##FILEWRITING and OUTPUTTING TO SCREEN
## WRITEFORMAT ASCII/BINARY
##TRACK_PROGRESS: interval of writing out the progress of the
    ↪ simulation to stdout.
WRITEFORMAT = ASCII;
TRACK_PROGRESS = 10;
##Model-specific parameters: Grand-potential model
##Phase-field parameters; epsilon:interface width; it is not the
    ↪ gradient energy coefficient
epsilon = 400.0;
##Anisotropy functions
##Anisotropy mode, FUNCTION_ANISOTROPY=0 is isotropic
```

```
Function_anisotropy = 1;
dab = {0.023};
#Rotation_matrix = {0, 1, Euler_x(ang), Euler_y(ang), Euler_z(ang
    ↪ )};
Rotation_matrix = {0, 1, 0, 0, 0};
#Writing of composition fields along with the chemical potential
    ↪ fields
Writecomposition = 0;
#Noise
Noise_phasefield = 1;
Amp_Noise_Phase = 0.0001;
#TEMPGRADY={BASETEMP, DELTAT, DISTANCE, OFFSET, VELOCITY}
#Tempgrady = {0.96, 0.06, 800.0, 0, 0.016};
#Composition at T
ceq = {0, 0, 3.77440E-02};
ceq = {0, 1, 3.77440E-02};
ceq = {1, 1, 2.41737E-02};
ceq = {1, 0, 2.41737E-02};
#Initial domain composition
cfill = {0, 0, 3.20360E-02};
cfill = {0, 1, 3.20360E-02};
cfill = {1, 1, 1.97734E-02};
cfill = {1, 0, 1.97734E-02};
##Parameters for Kim model
#Time step to start the noise implementation
tNoiseStart = 1000;
TLiquidus = 1696.0;
#Anti-trapping current in composition evolution
atr = 1;
# ID of OpenCL platform
CLplatformID = 0;
# ID of OpenCL device in above platform
CLdeviceID = 0;
# Name of the TDB file
tdbfname = nbni_bolc.tdb;
```

2.1 Simulation geometry, spatial and temporal discretization

```
##Geometrical dimensions of the simulation domain
DIMENSION = 2;
MESH_X = 256;
MESH_Y = 256;
MESH_Z = 1;
```

- DIMENSION: This can take values 2,3 for 2D and 3D simulations respectively. This is a required key in the solver and not mentioning this key might lead to unexpected results

- MESH_X,MESH_Y,MESH_Z: These are the number of grid points in the domain(and not the physical lengths) in the respective X, Y, Z directions. When DIMENSION is 2, the value of MESH_Z will redundant and will be taken as 1.

```
##Discretization, space and time
DELTA_X = 40.0;
DELTA_Y = 40.0;
DELTA_Z = 1.0;
DELTA_t = 320;
```

- The values DELTA_X, DELTA_Y, DELTA_Z correspond to the grid resolution in the X, Y, Z directions respectively. Similarly, DELTA_t corresponds to the temporal discretization(time-step).

2.2 Phases and Components information

```
##Number of phases and composition
NUMPHASES = 2;
NUMCOMPONENTS = 2;
```

- The keys are self-explanatory. NUMPHASES corresponds to the number of phases in the domain, while NUMCOMPONENTS corresponds to the number of components(2, for binary, 3 for ternary etc.)
- These keys are absolutely necessary, please fill carefully for successful running of your codes.

```
COMPONENTS = {NB, NI};
PHASES = {FCC_A1, LIQUID};
```

- COMPONENTS refers to the tuple that consists of the names of the components. The names are separated by commas and the entire tuple needs to placed within {} followed by a semicolon.
- Similarly, PHASES refers to the names of the phases in the domain.

2.3 Boundary conditions

```
##0:FREE, 1: Neumann, 2: Dirichlet, 3: Periodic, 4: Complex
##BOUNDARY = {TYPE, X_LEFT, X_RIGHT, Y_FRONT, Y_BACK, Z_TOP,
    ↪ Z_BOTTOM}
BOUNDARY = {phi, 1, 1, 1, 1, 0, 0};
BOUNDARY = {mu, 1, 1, 1, 1, 0, 0};
BOUNDARY = {c, 1, 1, 1, 1, 0, 0};
BOUNDARY = {T, 1, 1, 1, 1, 0, 0};
```

- Any of the solvers in repository will consist of the following scalar default types. Type "phi" will represent the phase-field order parameters whose number is specified by the key, "NUMPHASES". Depending on the solver whether it is the grand-potential based solver, where "mu" will be the independent variable or if it is the Cahn-Hilliard or Kim-Kim-Suzuki based solvers, where "c" is the independent variable, Type "mu" or "c" will refer to the components in the key "COMPONENTS". Similarly, type "T" will refer to the temperature field. The BOUNDARY key will refer to the boundary condition for the respective Type of field. The following numbers in a given tuple refer to the boundary at the respective (X_LEFT, X_RIGHT, Y_FRONT, Y_BACK, Z_TOP, Z_BOTTOM) boundaries, that refer to the X, Y, Z boundaries in order. The boundary condition is specified by numbers, 0:FREE, 1: Neumann, 2: Dirichlet, 3: Periodic, 4: Complex, where for the DIRICHLET boundary condition the respective boundary can also take in a specified value which can be specified in the following key: BOUNDARY_VALUE. If the key is not present, the default remains the one that is initialized at the start of the simulation and is left unchanged. Further, if for any direction, for eg: X, one of the extremeties is specified as a PERIODIC boundary, then the other X-boundary will also be initialized as PERIODIC irrespective of the entry in the tuple.

```
#BOUNDARY_VALUE = {Type, Value X+, Value X-, Value Y+, Value Y-,
    ↪ Value Z+, Value Z-}
BOUNDARY_VALUE = {phi, 0, 0, 0, 0, 0, 0};
BOUNDARY_VALUE = {mu, 0, 0, 0, 0, 0, 0};
BOUNDARY_VALUE = {c, 0, 0, 0, 0, 0, 0};
BOUNDARY_VALUE = {T, 0, 0, 0, 0, 0, 0};
```

- This key corresponds to a specific value that needs to be specified on a boundary which will be held constant during the duration of the simulation. The definition of this key should follow the BOUNDARY specification and follows the same type of definition, except here the tuple Value X_LEFT, Value X_RIGHT, Value Y_FRONT, Value Y_BACK, Value Z_TOP, Value Z_BOTTOM, refer to values on the respective boundaries. The values in this tuple will only be utilized if the respective boundary condition on that boundary is DIRICHLET. By default, the value will be treated as the same as the one that is initialised at the start of the simulation, which will be utilized if this key is not present.

2.4 Number of iterations, smoothing time-steps and writing interval

```
#Running and saving information
NTIMESTEPS = 2000;
NSMOOTH = 10;
SAVET = 200;
```

- NTIMESTEPS: Total number of iterations that you wish the solver to run. This is not the total time

- NSMOOTH: The number of pre-conditioning steps for smoothening sharp phase-field profiles that are initialised at the start of the simulation
- SAVET: Writing interval, i.e, frequency of writing files of the respective fields

2.5 Material parameters

```
##Gas constant and molar volume
R = 8.314;
V = 7.43e-6;
```

- The values "R" and "V" will refer to the gas constant and the molar volume respectively

```
#DIFFUSIVITY={Diagonal:0/1, phase, 11,22,33..(K-1) diagonal
    ↪ elements, 12, 13, 23...(rest of the elements; rowwise)}
DIFFUSIVITY = {1, 0, 4.31e-9};
```

- The DIFFUSIVITY key refers to the inter-diffusivity matrix which has the tuple in the following form. The first element can taken in values 0/1, "1" refers to as a diagonal matrix and "0" is a full matrix.
- The following element refers to the phase number referring to the phases in the list PHASES. This can take values from 0 to NUMPHASES-1.
- The following elements are the values in the inter-diffusivity matrix. The first elements are the diagonal terms in the matrix, while the following elements correspond rowwise to the off-diagonal terms in the diffusivity matrix.
- If the first element in the tuple is "1", i.e. the matrix is diagonal irrespective of the number of entries in the tuple only the entries corresponding to the diagonal elements in the matrix will be read in.

```
##GAMMA = {12, 13, 14, 23, 24...}
GAMMA = {0.37};
```

- The GAMMA key refers to the interfacial energy $\gamma_{\alpha\beta}$ between the phases $\alpha\beta$. The elements in tuple correspond to all combination of phases forming the interfaces from the list of phases in PHASES numbered from 0 to NUMPHASES-1.
- The elements are numbered in the order $12, 13, 14, 23, 24 \ldots N(N-1)$, $N = NUMPHASES$ where "12" corresponds to the value of the interfacial energy between phase 1 and 2; γ_{12}.
- In the tuple only combinations $\alpha\beta$ are included where $\alpha < \beta$ as the value of the interfacial energy of the $\alpha\beta$ interface is $\gamma_{\alpha\beta}$ which is the same as $\gamma_{\beta\alpha}$.

- Therefore, the total number of elements in the tuple is $\frac{N(N-1)}{2}$.

```
#EIGEN_STRAIN = {phase, exx, eyy, ezz, eyz, exz, exy};
EIGEN_STRAIN = {0, 0.01, 0.01, 0.0, 0.0, 0.0, 0.0};
EIGEN_STRAIN = {1, 0.01, 0.01, 0.0, 0.0, 0.0, 0.0};
```

- The EIGEN_STRAIN key refers to the eigen-strain tensor in a given phase. The information about the elements of the eigen-strain are derived from the elements in the tuple.
- The first element refers to the phase number in the list PHASES and can take in values from 0 to NUMPHASES-1.
- The following elements in tuple are mapped to the eigen-strain matrix in the following order $exx, eyy, ezz, eyz, exz, exy$.
- This is an important key required for problems where there are coherent interfaces and the phase transformation is coupled with the stress distribution arising due to coherency strains at the interface.

```
#VOIGT_ISOTROPIC = {phase, c11, c12, c44};
VOIGT_ISOTROPIC = {0, 270, 187.5, 125.0};
VOIGT_ISOTROPIC = {1, 270, 187.5, 125.0};
```

- VOIGT_ISOTROPIC is the key that refers to the elastic stiffness properties in the Voigt notation for an isotropic material.
- The first element in the tuple refers to the phase which will be a number from 0 to NUMPHASES-1.
- The following elements are the values $C11, C12, C44$ in order.

```
#VOIGT_CUBIC = {phase, c11, c12, c44};
VOIGT_CUBIC = {0, 270, 187.5, 125.0};
VOIGT_CUBIC = {1, 270, 187.5, 125.0};
```

- VOIGT_CUBIC is the key that refers to the elastic stiffness properties in the Voigt notation for a cubic material.
- The first element in the tuple refers to the phase which will be a number from 0 to NUMPHASES-1
- The following elements are the values $C11, C12, C44$ in order

```
#VOIGT_TETRAGONAL = {phase, c11, c12, c13, c33, c44, c66};
```

- VOIGT_TETRAGONAL is the key that refers to the elastic stiffness properties in the Voigt notation for a tetragonal material.

- The first element in the tuple refers to the phase which will be a number from 0 to NUMPHASES-1.
- The following elements are the values $C11, C12, C13, C33, C44, C66$ in order.

```
##FILEWRITING and OUTPUTTING TO SCREEN
## WRITEFORMAT ASCII/BINARY
##TRACK_PROGRESS: interval of writing out the progress of the
    ↪ simulation to stdout.
WRITEFORMAT = ASCII;
TRACK_PROGRESS = 10;
```

- The key WRITEFORMAT mentions the type of the output files to be written. The possible values are ASCII or BINARY that are self-explanatory.
- For the case when the type chosen is BINARY, the format is BIG-ENDIAN such that it matches the BINARY type required for PARAVIEW.
- TRACK_PROGRESS is a key that will inform the solver about the frequency with which the progress of the simulation will be written to stdout.
- The number can be anything other than 0.

2.6 Model specific parameters

The following parameters are corresponding to the multi-component phase-field model as described briefly below: The microstructure is described by phase-field order parameter $\phi(\boldsymbol{r}, t)$. The order parameter is defined such that $\phi(\boldsymbol{r}, t) = 0$ represents liquid phase and $\phi(\boldsymbol{r}, t) = 1$ represents solid phase.

The total free energy function if the system is given by

$$F = \int_V \left[\frac{\varepsilon_\phi^2}{2} |\nabla\phi|^2 + W g(\phi) + h(\phi) f^S + [1 - h(\phi)] f^L \right] dV \tag{1}$$

where ε_ϕ is the gradient enrgy coefficient, $g(\phi) = \phi^2(1-\phi)^2$ is the double-well potential and W is the potential height. $h(\phi)$ is the interpolation function. f^p is free energy density of a phase p. In the present solver, f^p is Gibbs energy calculated from CALPHAD based developed thermodynamic database.

The concentration c_i of the ith solute is given as

$$c_i = h(\phi) c_{iS} + [1 - h(\phi)] c_{iL} \equiv \mu_i \tag{2}$$

where c_{iS} and c_{iL} are concentrations in bulk sold and liquid, respecivley. c_{iS} and c_{iL} are restricted by the equal chemical potential condition

$$\frac{\partial f^S}{\partial c_{iS}} = \frac{\partial f^L}{\partial c_{iS}} \equiv \mu_i \tag{3}$$

The Newton-Raphson method is employed for solving c_{iS} and c_{iL} from equations 2 and 3

The evolution equations for ϕ and c_i are as follows:

$$\frac{1}{M_\phi}\frac{\partial \phi}{\partial t} = \varepsilon_\phi^2 \nabla^2 \phi - W\frac{\partial g(\phi)}{\partial \phi} - \frac{\partial h(\phi)}{\partial \phi}\left[f^S - f^L - \sum_{i=1}^{n}(c_{iS} - c_{iL})\mu_i\right] \tag{4}$$

where M_ϕ is phase-field mobility

$$\frac{\partial c_i}{\partial t} = \nabla \cdot \left(h(\phi)\sum_{j=1}^{n} D_{ij}^S \nabla c_{jS} + [1 - h(\phi)]\sum_{j=1}^{n} D_{ij}^L \nabla c_{jL}\right) \tag{5}$$

D_{ij}^p is the diffusivity in the phase solid ($p = S$) or liquid ($p = L$)

Introducing the anti-trapping current term into equation 5 (for $D_S \ll D_L$)

$$\frac{\partial c_i}{\partial t} = \nabla \cdot [1 - h(\phi)]\sum_{j=1}^{n} D_{ij}^L \nabla c_{jL} + \nabla \cdot j_{at_i} \tag{6}$$

$h(\phi) = \phi$ and the anti-trapping term, j_{at} is given as

$$j_{at_i} = \frac{\varepsilon_\phi}{\sqrt{2W}}(c_{iL} - c_{iS})\frac{\partial \phi}{\partial t}\tilde{n} \tag{7}$$

A four fold anisotropy is introduced in the ε_ϕ

$$\varepsilon(\vec{\eta}) = \varepsilon_0(1 - 3\epsilon_4)\left(1 + \frac{4\epsilon_4}{1 - 3\epsilon_4}\frac{\phi_x^4 + \phi_y^4}{|\nabla\phi|^4}\right) \tag{8}$$

where $\vec{\eta} = \frac{\nabla\phi}{|\nabla\phi|}$, $\phi_{r_i} = \frac{\partial\phi}{\partial r_i}; r_i = x, y, z$, ε_0 is the isotropic value and ϵ_4 is strength of capillary anisotropy

With this information, one can now follow the parameters in the file corresponding to the model

```
#Phase-field parameters; epsilon:interface width; it is not the
    ↪ gradient energy coefficient
epsilon = 400.0;
Function_anisotropy = 1;
##dab = {12, 13, 14, 23, 24...}
dab = {0.04};
```

- "epsilon" is the interface width.
- Function_anisotropy is the key to relate to the particular form of $\varepsilon_\phi(\vec{\eta})$ that is utilized as the gradient energy function.
- For a value of zero, the system is isotropic with respect to the change in the interfacial energy with orientation of the interface normal.
- A value of 1 corresponds to a particular function as described in Eqn.8 where the anistropy function is present only in the gradient energy term.
- The "dab" key in the infile is relevant in the presence of anisotropy in the interfacial energies when the key Anisotropy_type=4.

- The "dab" key refers to the strength of the interfacial energy anisotropy ϵ_4 between the phases $\alpha\beta$. The elements in tuple correspond to all combination of phases forming the interfaces from the list of phases in PHASES numbered from 0 to NUMPHASES-1.
- The elements are numbered in the order $12, 13, 14, 23, 24 \ldots N(N-1)$, $N = NUMPHASES$ where "12" corresponds to the value of the strength of the interfacial energy anisotropy between phases 1 and 2; $\epsilon_{4_{12}}$.
- In the tuple only combinations $\alpha\beta$ are included where $\alpha < \beta$ as the value of the anisotropy strength of the $\alpha\beta$ interface is $\delta_{\alpha\beta}$ which is the same as $\delta_{\beta\alpha}$.
- Therefore, the total number of elements in the tuple is $\frac{N(N-1)}{2}$.

```
#Rotation_matrix = {0, 1, Euler_x(ang), Euler_y(ang), Euler_z(ang
    ↪ )};
Rotation_matrix = {0, 1, 0, 0, 0};
```

- The "Rotation_matrix" key refers to the rotation of the anisotropy function by a given Euler angle description given by $\theta_x, \theta_y, \theta_z$, where θ_x refers to the angle of rotation about X-axis, θ_y refers to the angle of rotation about Y-axis and θ_z is the angle of rotation about Z-axis.
- The values are taken in as a tuple, where the first two elements correspond to numbers from 0 to NUMPHASES-1 corresponding to combinations of phases that can form an interface.
- This rotation operation will be reflected in the interfacial energy anisotropy of the relevant interface. For example if the first two elements are 0 and 1 then the anisotropy function of the interface between the phases 0 and 1 will be influenced.
- The following three elements represent the three Euler angles $\theta_x, \theta_y, \theta_z$, where θ_x as described before.

```
ceq = {0, 0, 3.77440E-02};
ceq = {0, 1, 3.77440E-02};
ceq = {1, 1, 2.41737E-02};
ceq = {1, 0, 2.41737E-02};
cfill = {0, 0, 3.20360E-02};
cfill = {0, 1, 3.20360E-02};
cfill = {1, 1, 1.97734E-02};
cfill = {1, 0, 1.97734E-02};
```

- The key "ceq" refers to the equilibrium compositions of the phases. The considered equilibrium is between the last phase NUMPHASES-1 and any of the other phases at the temperature provided in the key "T"
- `T` represents the simulation domain temperature in case of isothermal solidification.

- The first two elements of the tuple ”ceq={}” are numbers between 0 and N = NUMPHASES-1, which correspond to the list of N phases.
- Thereafter, the tuple consists in order, the compositions of the K = NUMCOMPONENTS-1 components.
- The required values are corresponding to all combinations ”ceq=$\{a, a, 0, 1 \ldots K\}$” and secondly ”ceq=$\{a, N-1, 0, 1 \ldots K\}$” where $a \in [0, N-1]$. The former corresponds to the composition of the ”a” phase in equilibrium with the ”N-1” phase, and the latter set corresponds to the composition of the ”N-1” phase in equilibrium with the ”a” phase.
- The remaining possibilities can be skipped.
- The key ”cfill” is of the same type as ”ceq” and is utilized while initializing the domain. It is safe to start with having the ”cfill” the same as ”ceq”, unless one wants to start with a supersaturation.

```
##Temperature
ISOTHERMAL = 1;
T = 1690.0;
```

- The key ”T” sets the value of the temperature for isothermal simulations when the key ”ISOTHERMAL = 1”.

```
Noise_phasefield = 1;
Amp_Noise_Phase = 0.0001;
tNoiseStart = 1000;
```

- ”Noise_phasefield” is a key to impose noise in the phase-field evolution equations at the interface. This is typically utilized for dendritic simulations.
- When ”Noise_phasefield=1”, the amplitude of the noise is specified by the key ”Amp_Noise_Phase” and noise introduced from the timestep given by the key ”tNoiseStart”
- `TLiquidus`: Liquidus temperature of the alloy system
- `atr`: `atr=1`, enables anti-trapping current in composition evolution equation and equations 4 and 6 are solved. For `atr=0`, equations 4 and 5 are solved.
- `CLplatformID`: ID of OpenCL platform. The OpenCL platform can be any of availabe CPU or GPU configured with OpenCL drivers and run time libraries.
- `CLdeviceID`: ID of OpenCL device in chosen OpenCL platform

- `tdbfname`: Name of the TDB file. The TDB file is read by `GEdata_writer.py` and executes it for generation of Gibbs energies and their derivatives. It is adviced to follow the names of elements and phases as given in TDB file. The components names in `COMPONENTS` are given such that {`x1,...,xn`}, x1 is 1st alloying element, xn is solvent (matrix) element. Name of the element should be in accordance with TDB file. Similarly, for phase names in `PHASES` are given such that {`phase_1,...,phase_n`}, phase_1 is 1st phase, phase_n is matrix phase. Name of the phase should be in accordance with TDB file. *pycalphad* is required for execution of `GEdata_writer.py`.

3 Compiling and running

- The solver can be compiled using the command "make" in the command line in a terminal that is opened in the folder which contains the "Makefile".
- After compilation the executable "kim_soldfn.out" will be created.
- The solver takes in three command line arguments, the first is the name of the Input file, the second the name of the filling file containing the information about how the domain should be initialized and thirdly the name of the output file.
- GEdata_writer.py is used for generation of Gibbs energy and its derivatives. The generated Gibbs energy information is stored in directory 'solverloop/GibbsEnergyFunctions'
- To generate Gibbs energies and execute the program run "./kimsldfn.sh Input.in Filling.in Output". It is always safe to run this command for execution of the code.
- If Gibbs energies are already generated then the solver can be run as ./kim_soldfn.out name_of_infile name_of_filling_file name_of_output_file.
- The output of the program will be written in a folder "DATA" with the filename "name_of_output_file_timestep.vtk" for the separate timesteps.
- The files at all the timesteps can be opened as a group in the software "paraview" and the fields according to the names given in the COMPONENTS and PHASES keys can be visualized.