Manual of Coupled Atomistic-Continuum Program (CACP)

Section 1. Introduction to concurrent coupled atomistic-continuum model

CACP has following three major components:

1) Domain partition + geometry design
2) Individual solver for FE and MD
3) Staggered solving and information passing

Domain partition + geometry design

Domain partition relates to the basic geometry information of the simulation sample, for atomistic system, the atoms are created by lammps input file, controlled by commands such as lattice, box, create_atoms etc. For continuum system, the mesh file "single.inp" is created by the mesh generator, controlled by the information of nodes and elements together with orientation information by "texture.dat, elem_graint.dat". The material property of atomistic system is defined using the interatomic potential, in continuum defined by constitutive relation provided in "umat*.f".

The interface region are defined in "couple_run.inp" together with subroutine Voronoi_partition() in "ESCM_couple.f". Where the nodes and atoms falls into interface region are called interface nodes, and interface atoms. the connection between interface atoms and interface nodes will be constructed by Voronoi partition.

Individual solver for FE and MD

In continuum domain, the finite deformation problem is solved using update Lagrange method. The equilibrium configuration is obtained by quasi-newton solver, the matrix inversion is calculated using superLU package.

In atomistic domain, molecular dynamics is conducted, the equilibrium can only be reached in an statistical sense, meaning the time average of system parameters such as stress, strain, energy, temperature becomes stable. The evolution of MD is controlled by time integration scheme together with ensemble control. In CACP, the standard choice is velocity verlet time integration for all atoms. And Langevin dynamics and gradient damping applied on interface atoms. The parameters for timestep and ensemble control in CACP is same as in regular MD.

Staggered solving and information passing

The staggered solving process means instead of directly solve all degrees of freedom at once, we solve two systems separately, iteratively evolve both system until it converges to the equilibrium solution. Choosing this strategy is motivated by two reasons:

i) When using molecular dynamics (finite temperature atomistic simulation), there is no clear definition of equilibrium, solving two systems together corresponds to molecular static simulation results, in which case, the temperature fields will be added by other approach with certain assumptions. Notice this is still doable, (finite temperature Quasi-continuum). If using MD directly, the temperature field is a natural thing.

ii) It can be easier and more efficiently implemented. Each system only needs to provide a interface to deliver/receive the passing information. For example, in CACP lammps is used for MD with extract/distribute methods on per-atom info.

The iterative process for one load increment can be decomposed into four steps, same structure will be seen in the main code "fem_uel.f"

i) AtoC step: Extracting the displacement of all interface atoms from LAMMPS processors, pass them to FE processors, calculate the displacement boundary condition of interface nodes from atom-node relation.

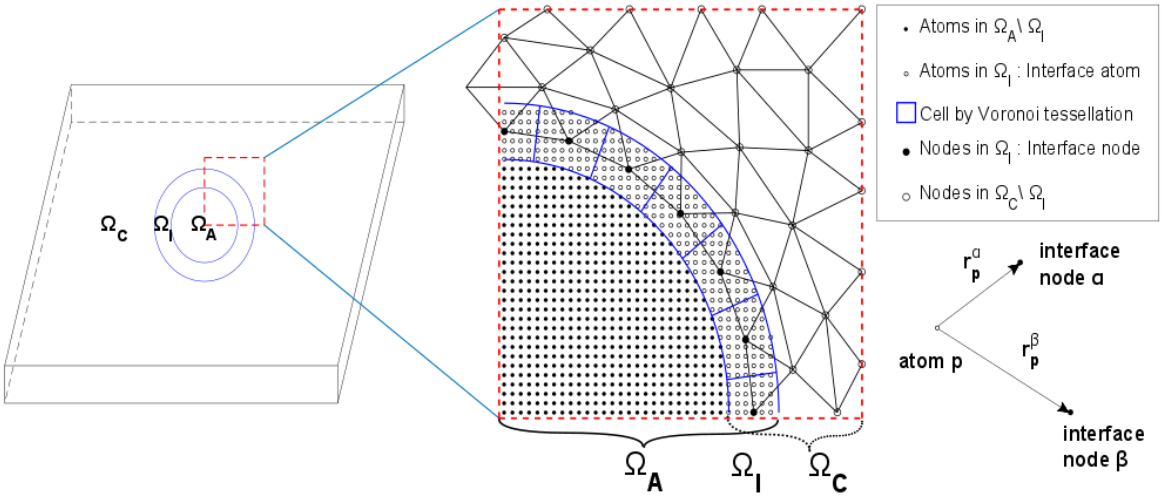
ii) FE solve step: solve the equilibrium configuration of all FE degrees of freedom based on external load boundary condition on boundary nodes (force or disp) and displacement boundary condition on interface nodes. Calculate the reaction force on interface nodes.

iii) CtoA step: Use the reaction force of all interface nodes to calculate the external force on interface atoms, then pass the external force to LAMMPS processors and distribute accordingly.

iv) MD step: evolve the MD system based on external force for a given time (default 1 ps). Calculate the time average displacement on interface atoms.

Combining step i) – iv), it forms a complete loop of staggered method.

Flow chart and figure:



JOHNS HOPKINS WHITING SCHOOL of ENGINEERING

Numerical Implementation

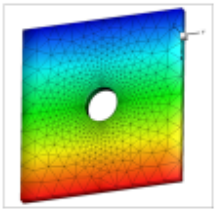
Computational Mechanics Research Laboratory CMRL

FEM

Static problem finding equilibrium configuration

Solve for reaction force of any boundary node I with displacement B.C.

R_I



Handshake region

Boundary atoms external force on atom j

$$f_j^{\text{ext}} = w_j R_I^C$$

Boundary nodes displacement on node I

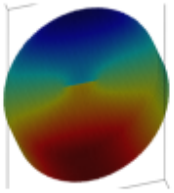
$$u_I^C = \sum_j^{m_I} w_j \langle u_j^A \rangle$$

Molecular Dynamics

Velocity Verlet and NVE ensemble

Solve for displacement field and its time average for any atom j in handshake region

$u_i(t), \langle u_i(t) \rangle$



Section 2. Details of CACP, important variable, subroutines or files

Folder Lammps_src/:

- atom.h/cpp, atom_style_vec.h/cpp: deal with per-atom information used for coupled model
  add two variables: **fext, *IVC_id.
  fext[i][j] defines the external force on direction j on atom i,
  IVC_id[i] defines which interface node atom i belongs to.
  For non-interface atoms, fext[i][j] = 0.0, IVC_id[i] = 0

- domain.h/cpp: deal with global information used for coupled model
  add 3 variables: nnode_inter, *natoms_IVC, *natoms_SVC
  nnode_inter: number of interface nodes
  *natoms_IVC: number of interface atoms associated with each node
  *natoms_SVC: number of surface atoms associated with each node

- fix_external.cpp controls how does the external force applied on interface atoms.
  This files corresponds to one command that is necessary when running MD in a coupled model.
  Three parameters controls the external force, t_start, t_apply, t_evaluate
  T_start: is the time when coupled model starts, before which fext wont be calculated or applied.
  T_apply: is every how many timestep the external force is applied, should be 1 in most cases.
  T_evaluate: is every how many timestep the external force is re-evaluated. This is related to the time increment and MD timestep, say time increment is 1 ps and MD timestep is 2 fs, then T_evaluate should be (1ps/2fs) = 500.

  There are also some compute class which provides some necessary/useful insights of the MD system.

- compute_neighlist.h/cpp, compute_FeFp.h/cpp computes the per-atom deformation gradient based on least-square fit. It can also decompose into elastic and plastic component, the details decomposition can be found in Jiaxi Zhang's Ph.D. Dissertation.

- compute_Fp_IVC.h/cpp computes the average plastic deformation gradient at interface region for each interface node based on compute_FeFp.

- Compute_SVC_force.h/cpp, this will be called by fix_external.cpp in order to reduce ghost force for deformed configuration, refer to paper: "An embedded statistical method for coupling molecular dynamics and finite element analyses", Saether et al. (2008)

Folder main_code/:

- fem_uel.f: The main code, deal with pre-processing and staggered solving. The FE quasi-newton method is also implemented in this file. Details can be found in the comments.

- pardis.h this file defines the global variables and some upper bounds of FE simulation.

- ESCM_couple.h/f the variables used for coupled simulation are defined in ESCM_couple.h. the important subroutines for coupled simulation are implemented in ESCM_couple.f
  Important subroutines: ATC_update, CTA_update, Voronoi_partition,

- loadvc_C.f calculate the force, residue on each d.o.f. and stress on each IP for a given configuration. this subroutine is called in fem_uel.f calls subroutine defined in uel_disp_t.f

- disinc_C.f applies external load via boundary condition on boundary d.o.f. also delivers the boundary condition on interface nodes by calling ATC subroutine

- uel_disp_t.f calculate B matrix based on element type (here linear tetrahedron element)

- umat_nonlinear.f calculate the stress from strain at IP based on material constitutive equation
  The code currently includes nonlinear elastic constitutive relation for Ni, Al, Cu. For CPFE model, plasticity flow rule needs to be defined.

- DG_calc.f Calculate deformation gradient at interface region from atomistic and continuum model. Both average value and per interface node value are calculated. In general, comparing average value from atomistic and continuum model ("AveFA.out", "AveFC.out") should provide a good idea of whether the material model used in FE and MD are consistent or not.

- libfwrapper.c To allow fortran calls the method given in "library.h" as interface for lammps, this file provides the c wrapper for those methods.

- superlu_c2f_dwrap.c, dcreate_dist_matrix.c, init_slu.f, superlu_mod.f90, superlupara.f90: superLUa related file

Section 3. Input, output and Parameters

3.1. Run command and input files

Run command: "*mpirun -np n CACP md.inp"*

CACP is the path of CACP program compiled executable, n is the number of processors used for CACP, md.inp is the path of CACP program MD input file.

Global setting: loadtime_creep.inp, loadtime.inp, couple_run.inp, slu_proc_distrib.inp

FE part: single.inp, elem_grain.dat, texture.dat

MD part: MD_run.in (name is flexible, consistent with run_CACP.batch file)

- single.inp: defines FE mesh, boundary nodes, interface nodes, timestep, running time
- elem_grain.dat, texture.dat: orientation information for each element (the length should be more than number of elements)
- MD_run.in format and most part same as conventional MD simulation, details refer to the comments in file
- loadtime_creep.inp, loadtime.inp: applied strain rate, allows strain rate jump by define the time step range and corresponding strain rate.
- Slu_proc_distrib.inp: defines the SuperLU processor decomposition for FE processors.
- couple_run.inp:
  defines the how does external load is applied during staggered solving process (N_load, relax)
  defines the MD steps for each time increment
  defines the interface region, (default defined using cylindrical coord system, $r1 < r < r2$)

Details:

1. FEM input file refer to cpfem manual
2. MD input file refer to lammps manual
3. The couple part: refer to comments in input files in input folder. Mainly couple_run.inp, MD.inp