HNGD stand-alone C++ code

Florian Passelaigue

This document presents a description of a stand-alone C++ version of the modified Hydride Nucleation-Growth-Dissolution (HNGD) model.

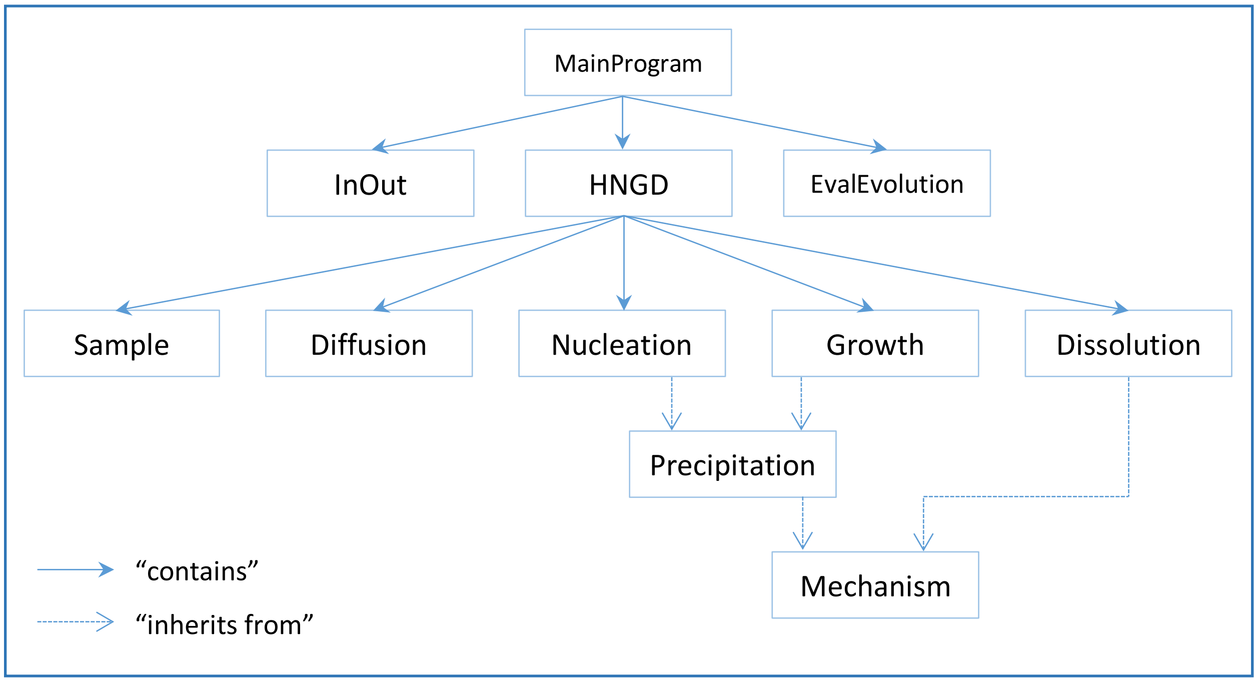
Initial HNGD development: E. Lacroix, P.-C. A. Simon, A. T. Motta, and J.D. Almer. Zirconium Hydride Precipitation and Dissolution Kinetics in Zirconium Alloys. *Zirconium in the Nuclear Industry: 19th International Symposium*, STP 1622, pages 67– 91, 2019

Initial implementation of this stand-alone code: F. Passelaigue "Hydride Nucleation-Growth-Dissolution Model: Implementation in BISON", *Master Thesis*, The Pennsylvania State University (2020)

Modified HNGD development: Passelaigue, F., Simon, P.-C. A., Motta, A. T. "Predicting the hydride rim by improving the solubility limits in the Hydride Nucleation-Growth-Dissolution (HNGD) model." *Journal of Nuclear Materials* (submitted)

Code structure

Overview



MainProgram

This file contains the `main()` function of the program. It takes the argument given by the user to determine which input files to use, and then read these input files via the `InOut` class' reading methods. Once the simulation parameters are read, they are transferred to the `HNGD` instance so the system can be initialized.

This file also manages the time loop, using the `HNGD` object to compute each time step and a couple of `EvalEvolution` instances to determine whether the current state should be written in the output, via the `InOut` static functions.

Three lines contain a comment `/*custom*/` in this file, showing where the user needs to make adjustments to fit their needs.

The first one marks the location of the definition of the string variable `path_exec`. This is the path to the folder that will contain the outputs and the folder named `input_files` where the program will look for the correct inputs.

The second one marks where to define how many quantities the user wants to be in the output. By default, there are 5 output variables: the total hydrogen content, the solid solution concentration, the hydride content, the solubility and the supersolubility. If the user wants to make a change, they also need to make modifications in `InOut.c`. This is detailed in the next section.

The third and last one is optional. It gives the possibility to play a sound file at the end of the simulation. This can be convenient in case of long simulations.

InOut

This class contains only static methods, used to read the input files and write the output. As detailed later in this document, a simulation requires four input files to work. Each type of input file has a method associated with in the `InOut` class. During the initial step before the simulation starts, this class is used to write a verification file that makes possible to check that all parameters were correctly passed to the program. If it is not the case, the input files format should be checked. During the simulation, the output file is written when requested by the main function.

As mentioned in the previous section, if the user wants to make changes to the output of the simulation, a couple of changes need to be made in this class. Again, the lines are marked with a `/*custom*/` comment.

The first one shows where to tell the program how to access a given quantity. Here are a few examples:

- For the first output to be the total hydrogen profile. This profile is stocked in the `Sample` object of the HNGD model:

  ```
  listVector[0] = hngd.returnSample()->returnTotalContent();
  ```

- For the second output to be the hydrogen flux. This quantity is stocked in the `Diffusion` object of the HNGD model:

```
listVector[1] = hngd.returnDiff()->returnFlux();
```

- For the third output to be the growth kinetic factor. This parameter is stocked in the Growth object of the HNGD model:

```
listVector[2] = hngd.returnGro()->returnKinetics();
```

The second `/*custom*/` comment shows where to put the names of the output variables.

EvalEvolution

This class computes the evolution of a given profile compared to a reference state. This is done by comparing the average relative difference to a criterion fixed by the user:

$$\frac{1}{N}\sum_{k}\left|\frac{current_k - ref_k}{ref_k}\right| > criterion$$

If the relationship above is true, then the current state takes over as reference, and it is printed in the output file. The default value for the criterion is 5%.
The code uses two instances of this class, managed by `MainProgram`: one for the hydrogen profile, the other for the temperature profile.

Sample

This class is used to create the geometry of the system. It also computes the effective solubility and effective supersolubility profiles. If needed, a bias can be introduced so that the discretization is finer at one end of the sample

Diffusion

This class computes the diffusion flux of hydrogen in solid solution. It accounts for two components: Fick's law and Soret effect. This flux is described by the following equation:

$$J = -D\nabla C_{ss} - \frac{DQ^* C_{ss}}{RT^2}\nabla T$$

Where C_{ss} is the solid solution concentration, T is the temperature, D is the diffusion coefficient, Q^* is the heat of transport, and R the perfect gas constant.

Mechanism

This is a superclass for hydride nucleation, growth, and dissolution. It defines the quantities (kinetics, driving force, rate) that exist in each of these phenomena. Each daughter classes specifies how their kinetic factor and driving force are computed, and the rate of the mechanism is given by the product of these two quantities.

Dissolution

This class computes the rate of dissolution at each position of the system. The kinetic factor follows an Arrhenius law, and the driving force is given by the difference between the concentration in solid solution and the solubility:

$$\frac{\partial C_{ss}}{\partial t} = -K_{D0} e^{-E_D/kT} (C_{ss} - TSS_D)$$

Dissolution occurs if hydrides are present in an undersaturated matrix, i.e. $C_{ss} < TSS_D$.

Precipitation

Hydride nucleation and growth have common factors, so an intermediary class was created. `Precipitation` inherits from `Mechanism` and contains the functions to compute the factors $f_\alpha\ v_o$ and χ that appear in the equations for nucleation and growth, as well as the hydride formation energy that depends on the temperature (3rd polynomial fit)

Nucleation

This class computes the kinetic factor and driving force for hydride nucleation:

$$\frac{\partial C_{ss}}{\partial t} = -K_{N0} v_\alpha e^{-E_{th}/kT} (C_{ss} - TSS_P)$$

The kinetic factor follows a modified Arrhenius law and the driving force is the difference between the concentration in solid solution and the supersolubility. Nucleation occurs only if the solid solution concentration is above the supersolubility limit.

Growth

This class computes the kinetic factor and driving force for hydride growth. The kinetic factor has two components: diffusion-controlled and reaction-controlled, that follow modified Arrhenius laws and such as

$$\left.\begin{aligned} K_{mob} &= K_{mob0} v_\alpha x_\alpha e^{-E_G/kT} \\ K_{th} &= K_{th0} v_\alpha x_\alpha e^{-E_{th}/kT} \end{aligned}\right\} K_G = \left({}^{1}/_{K_{mob}} + {}^{1}/_{K_{th}} \right)^{-1}$$

The driving force for growth is given by the Johnson-Mehl-Avrami-Kolmogorov model, giving the rate of growth

$$\frac{\partial C_{ss}}{\partial t} = -K_G (C_{tot} - TSS_D) p (1-x)(-\ln(1-x))^{1-1/p} \quad with \quad x = \frac{C_{tot} - C_{ss}}{C_{tot} - TSS_D}$$

Where x is the advancement of the reation, $p = 2.5$ is the Avrami parameter for platelet growth, and C_{tot} is the local total hydrogen content.

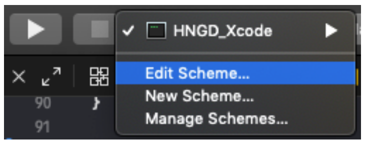
Input files

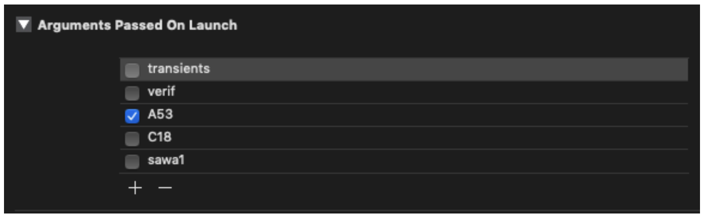
Overview

There are four types of input files needed to run a simulation: settings, physics, temperature and hydrogen. Their names must follow the formats `simulationName_set.txt`, `simulationName_phys.txt`, `simulationName_temp.txt` and `simulationName_hyd.txt` respectively.

The simulation name needs to be passed as an argument when the program is launched. If you work on a Mac computer, here are two ways to do so:

- Terminal command: `<path to the executable> simulationName`
- Using XCode:




Arguments Passed On Launch

- [ ] transients
- [ ] verif
- [x] A53
- [ ] C18
- [ ] sawa1

Settings

This file contains the numerical parameters of the simulation, separated by an empty line. In order, the number of nodes used to discretize the sample, the bias of the geometry, the size of the sample (in cm), the time step, the maximum time between two outputs, the evolution criterion, and the type of geometry (1D linear or 1D polar). The program includes an automatic management of the time step. To use it the user must put a negative value as the time step. Otherwise the code will use the fixed time step given in this file.

Physics

This file contains the values necessary to compute the physical parameters: kinetic factors, diffusion, and solubility/supersolubility. The table below shows the order and default values of these parameters:

Parameter	Default value
Dissolution kinetics preexponential factor	$1.11 \times 10^{3} s^{-1}$
Diffusion activation energy	$0.46\ eV/at$
Nucleation kinetics preexponential factor	$2.75 \times 10^{-5} s^{-1}$
Formation energy polynomial coefficient 0	$0.5655\ eV/mol$
Formation energy polynomial coefficient 1	$4 \times 10^{-4}\ eV/mol/K$
Formation energy polynomial coefficient 2	$2 \times 10^{-7}\ eV/mol/K^{2}$
Formation energy polynomial coefficient 3	$3 \times 10^{-10}\ eV/mol/K^{3}$
Diffusion-controlled growth preexponential factor	$5.35 \times 10^{5} s^{-1}$
Reaction-controlled growth preexponential factor	$1.6 \times 10^{-5} s^{-1}$
Diffusion-controlled growth activation energy	$0.9\ eV/at$
Supersolubility preexponential factor	$3853\ wt.ppm$
Supersolubility activation energy	$25249\ J/mol$
Solubility preexponential factor	$101999\ wt.ppm$
Solubility activation energy	$35459\ J/mol$
Diffusion preexponential factor	$1.08 \times 10^{-2}\ cm^{2}/s$
Heat of transport	$25500\ J/mol$
Supersolubility decrease time	$10^{4}\ s$
High hydride content solubility dependency	1.05
Low hydride content solubility dependency	$120\ wt.ppm$

Hydrogen

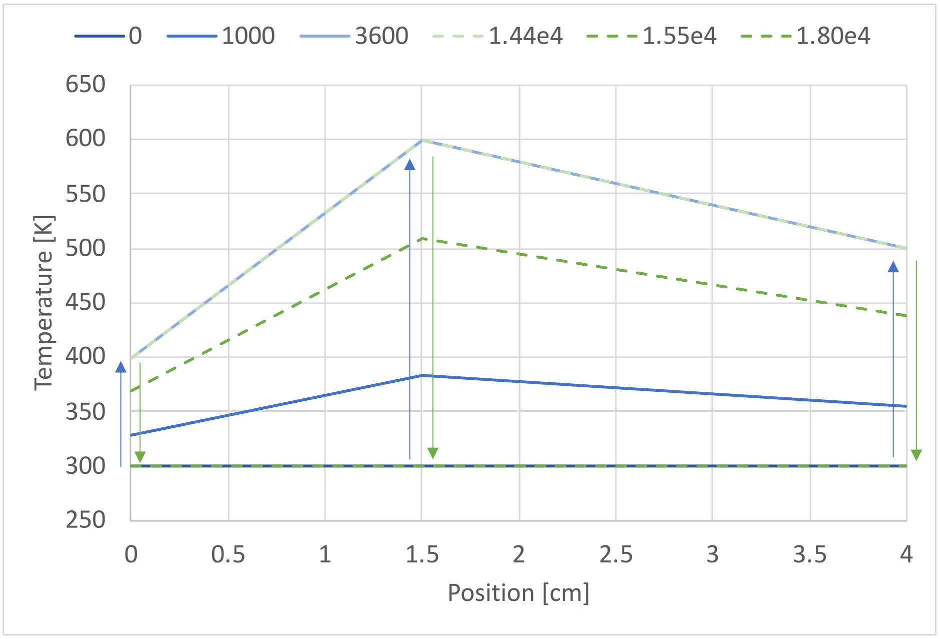
This file contains the initial hydrogen profile of the simulation. The first line contains the positions where the hydrogen content is specified, and the second line contains the hydrogen content of each of these positions. The program interpolates linearly between each position.

Temperature

This file contains the temperature treatment applied to the sample. Like above, the first line contains the positions where the temperature is defined, and the following lines contain the successive temperature profiles, preceded by the time when it should be attained. Below is an example for a sample starting at room temperature, heated-up to a “triangular” profile in an hour, maintained for three hours, and brought back to room temperature in one minute. The sample is 4cm long and the maximum temperature is reached at 1.5cm from the colder end.

```
0            1.5     4
0            300     300    300
3600         400     600    500
1.44e4       400     600    500
1.80e4       300     300    300
```

The program interpolates linearly between the positions and time stamps:



In the case of a polar geometry simulation, the values in the first line must be given in rad.

Two example cases using different geometries are available in the repository.