TABLE OF CONTENTS



# 1.-Installation.md

KIMERA is written in the standard C++11. Therefore, in order to install it, we will need the proper C++ compiler. Moreover, KIMERA is paralelizated by means of the OPENMP library so such library is also requiered.

For the moment, I have managed the installation of KIMERA in some Windows and Linux machines:

**REQUERIMENTS:**

-a **C++ compiler**. I have used TDM-GCC on Windows and GCC on linux. You need 4.9.2 or higher (the current version on 21/12/2020 for TDM-GCC is 9.2.0 and for GCC is 10.2)

-**openmp library**.

WINDOWS:

1. Installation of TDM-GCC
    - Installation of openMP library, **which is available as an optional component in the TDM-GCC compiler installation**

      Image
    - After the installation of TDM-GCC and the openMP library, restart your system.
2. Compilation of KIMERA.

    In the command prompt (you can access the windows command prompt by writting `cmd` after preessig Windows key and 'r' key), go to your KIMERA folder by indicating the path to it.

    ```
    cd C:\Path\to\your\KIMERA\folder\
    ```

    for example:

    ```
    cd C:\Programs\Kimera\
    ```

    compile it with the following command, change the compilator paths if neccesary. (In this example we have installed the 64 version in the default installation folder):

    ```
    C:\TDM-GCC-64\bin\g++.exe -L\C:\TDM-GCC-64\lib\gcc\x86_64-w64-mingw32\9.2.0\include\ -Wl,-rpath=C:\TDM-GCC-64\lib\gcc\x86_64-w64-mingw32\9.2.0\include\ -Wall -fexceptions -fopenmp -std=c++11 -O3 include\Atom.h include\Box.h include\Cell.h include\Control.h include\Data_printer.h include\Data_reader.h include\Event.h include\Event_definition.h include\Event_stack.h
    ```

`include\Event_stack_unstuck.h include\Linked_neighbour.h include\Models.h include\Position.h include\RandomGenerator.h include\Record.h include\Simulation.h include\Sym_equation.h include\Tracker.h include\Util.h main.cpp src\Atom.cpp src\Box.cpp src\Cell.cpp src\Control.cpp src\Data_printer.cpp src\Data_reader.cpp src\Event.cpp src\Event_definition.cpp src\Event_stack.cpp src\Event_stack_unstuck.cpp src\Linked_neighbour.cpp src\Models.cpp src\Position.cpp src\RandomGenerator.cpp src\Record.cpp src\Simulation.cpp src\Sym_equation.cpp src\Tracker.cpp src\Util.cpp -Iinclude -o Kimera.exe`

> TIP: If you get an error of 'Permission denied' (see Image), You must deselect the 'Read-only' attribute of your KIMERA folder (see Image) (left click -> Properties in the KIMERA folder).

3. (OPTIONAL, BUT RECOMMENDED) Install zip program. KIMERA will automatically zip your output files. zip 3.0

   For Windows to recognise the 'zip' command, you will need to 'edit the system environment variables'.
   1- image
   2- image
   3- image
   4- add new line with the path where zip have been installed, for example by default `C:\Program Files (x86)\GnuWin32\bin`
   5- Ok to all

4. USAGE TIPS

   A. It is better to use KIMERA from the command prompt by specifying the path to the input file

   `Kimera.exe path\to\kimera\inputfile\kimera.input`

   for example

   `Kimera.exe C:\Programs\Kimera\examples\AB_Kossel_crystal_random\kimera.input`

   same, but print the log in a text file:

   `Kimera.exe C:\Programs\Kimera\examples\AB_Kossel_crystal_random\kimera.input > log.txt`

   B. Nevertheless, you can also move your input files to the same folder as Kimera.exe, and double-click it.

LINUX:

1) Installation of GCC. Write the following commands in the terminal:

```
`sudo apt update`

`sudo apt install build-essential`

`sudo apt-get install manpages-dev`

- Installation of openMP library

`sudo apt-get install libomp-dev`
```

2) Compilation of KIMERA. In the terminal, go to KIMERA folder

```
 for example

 `cd  /home/user/Kimera`

 execute the makefile with `make`

 If it doesn't work, try to specify the path to your compliler (see and edit the makefile in the KIM
ERA folder)

 make sure that the program has execution rights:

 `sudo chmod 777 Kimera.exe`
```

3) (OPTIONAL, BUT RECOMMENDED) Install zip program. KIMERA will automatically zip your output files.

```
 `sudo apt-get install zip`
```

4) USAGE TIPS

```
 A. It is better to use KIMERA from the terminal by specifying the path to the input file

 `./Kimera.exe  path/to/kimera/inputfile/kimera.input`

 for example

 `./Kimera.exe /home/user/Kimera/examples/AB_Kossel_crystal_random/kimera.input`

  same, but print the log in a text file:

 `./Kimera.exe /home/user/Kimera/examples/AB_Kossel_crystal_random/kimera.input > log.txt`

 B. Nevertheless, you can also move your input files to the same folder as Kimera.exe, and execute i
t.
```

# 2.-Input-structure.md

KIMERA is a command base program in which the user define the system and the simulation parameters by means of commands on an input file. The commands are read by KIMERA subsequently so the effect of a command may be changed or neglected by the effect of a later one. The list with all the commands can be found in this repository in 'intructions/command_list.txt'. In the next step we will classify them in several categories for clearence purposes:

- **Creation of the simulation box**. Unit cell parameters, system dimensions, or Periodic Boundary Conditions (PBC) are essential to start building up your system. These commands are listed in section 2A. Simulation box.
- **Simulation box modifications**. The actual topography and shape of the dissolving system can be specified by commands acting as system modifiers. The initial system box in 2A. Simulation box can be sculpted and changed to procude the desired one. The user can use an external script to help them to create their system. For example, In order to introduce in your system hundrends of adatoms (lonely atoms on the surface) at random postions, an external script to print the commands that will be later used in the KIMERA input file is very handy. The topography and shape commands are listed in section 2B. Topography. Moreover, the constituent atoms or particles of your system can be also modified after the system creation. Commands intended fo this porpuse are listed in section 2C. Particle type modifiers.
- **Events definition**. In the KMC algorithm, in order to study the time evolution of the dissolving system, it is necessary to know beforehand the possible transitions, or events, that may happen in it. KIMERA recognises several commands to do the event definition as detailed as possible. The event definition commands are listed in section 2D. Events definition.
- **Reactive surface**. KIMERA creates from the whole system a list with the initially reactive atoms for performance porpuses. Since it may not be to our liking, 2E. Reactive surface contains the commands that modify it.
- **Output requests**. The output of KIMERA consists of several files inteded for visualization and data analysis. The commands that handle the output files are listed in section 2F. Output requests.
- **Other parameters**. The finishing condition, the random generator seed, the desired number of cores to run the simulation,

the distance threshold, or the searching algorithm (lineal or binary) are some of the side parameters that can be set. The commands that modify them are listed in 2G. Other parameters.

## Comments

Any word different from a command is not considered by the program and can be treated as a comment. Nevertheless, each command is a close statement and it is not possible to make any comments in bettwen. Since all commands in KIMERA are in capital letters, in our examples we comment using lowercase letters starting with two hypens '--' to make a clearer differentiation.

## System exportation

Through this wiki and the command list, you will find commands with comments like `--(*)`, `--(**)` or `--(***)`. These marks are only important when using the system exportation feature. KIMERA can use the system from a previous simulation to save computational time, but it creates restrictions in the change of the simulation parameters. specifically, commands from 2A. Simulation box have to be identical to the previous simulation, 2B. Topography, 2C. Particle type modifiers and 2E. Reactive surface commands have no longer effect, the structure of 2D. Events definition has to be the same though the values of the parameters can change, and all the commands related to 2E. Output requests and most of the 2G. Other parameters commands can be freely modified.

To recapitulate, marks indicate:

```
--(*)   The change of these parameters is possible.
--(**)  This command is compulsary as was in the original definition. Only the change of the ED and
EP, FFD, DG* values is possible.
--(***) They are not possible to be used or have no effect.
```

The command that export a previous system is:

```
READ_SYSTEM_FROM_KIMERA_FILE       (text)path_kimera_file
```

where ´path_kimera_file´ indicates the path of the kimera file in your computer.

Finally, three important points are highlighted:

1- It is necessary to request the program to print the KIMERA file in the previous simulation. You can find the details to do it in the E. Output requests section.

2- The format of this file is specific to KIMERA and can be found in this wiki, or in the 'instructions/KIMERA_format.txt' in this repository.

3- It would be possible for the users to use other programs to create their system to later study its dissolution in KIMERA if the proper file transformation is done.

## Parameters type

Along this wiki and in the command list file, it can be observed that the command input parameters can be of this three types:

- (text): Only one word, or several words without spacing.
- (int): A natural number. 0,1,2,3,...
- (double): A decimal number.

You can set a double parameter with an int (for example 1 would be threated as 1.0). Nevertheless the opposite is not posible (2.6 would be threated as 2).

## Parameter units

The units of the input parameters are the following:

- Energy barriers: _$k_B$_T_) units. _$k_B$ is the Boltzman constant ($1.380\times10^{-23}$ J $K^{-1}$) and _T_ the temperature (K).

> Tip: _$k_B$_T_/$N_A$ = 2.494 kJ $mol^{-1}$ at 300 K. $N_A$ Avogadro constant.

- Time: seconds (s).
- Fundamental frequencies: ($s^{-1}$).
- Distance: Angstrom (Å).
- Mass: Atomic mass unit) (Dalton or u).

# 2A.-Simulation-box.md

> --(**) These command is compulsary as was in the original definition.

> --(***) They have no effect.

The first step to create your system is to define the parameters of the unit cell. The unit cell can correspond to a real mineral, like quartz, or to a theorical one, like a Kossel crystal. Three considerations can be highlighted.

1. You can define as a unit cell, a supercell consisting of several unit cells, or even the hole system. As we will see in section Events definition, this will give more versatility to your simulation, but the program performance is reduced.
2. The particles of your unit cell can represent actual atoms, or coarse grains.
3. You can define 'phantom positions' inside your unit cell without physical meaining but helping define the events of your system.

The commands for defining the unit cell are the following:

```
CELL_A                          (double)            --(**)
CELL_B                          (double)            --(**)
CELL_C                          (double)            --(**)

CELL_ALPHA                      (double)            --(**)
CELL_BETA                       (double)            --(**)
CELL_GAMMA                      (double)            --(**)
```

There are two ways of defining the positions of atoms or particles inside the unit cell.

1- They can be specified directly in the input file:

```
POSITION        (text)atom_type     (double)x   (double)y   (double)z   (double)occupancy   --
(***)
```

where the `occupancy` (number bettwen 0 and 1) represents the likelihood to find that element in that position.

2- Or they can also be read from external files with a simple xyz format, useful to load experimental crystal structures of mineralogist databases (AMS, for example):

```
READ_POSITIONS_FROM_XYZ_FILE      (text)path_xyz_file               --(***)
```

where `path_xyz_file` indicates the path of the xyz file in your computer. These two ways are complementary, so you can define additional positions to the ones in the xyz file.

> Tip: There are some crystallographic structures in which one unit cell position can be occupied by different elements with certain likelihood. This likelihood, named occupancy, is not contemplated in a simple xyz file. Therefore, if the occupancy feature needs to be used, the way to proceed is to introduce them with the `POSITION` command by coping them from the xyz file, but taking into account the occupancy value which is available in the cif file.

> Tip: The xyz file is a simplification of the cif file. You can usually download both files from the mineralogist database, or at least the cif one. There are tools, like VESTA program that allows you to transform your cif file into a xyz file.

Once we have the unit cell with the positions defined, we replicate it in the three space directions

```
DIMENSION_A                    (int)              --(**)
DIMENSION_B                    (int)              --(**)
DIMENSION_C                    (int)              --(**)
```

We can consider the resulting simulation box to have periodicity in any of the three space directions, which is known as periodic boundary conditions (PBC).

```
PERIODICITY                    (optional)A  (optional)B  (optional)C
```

Usually, dissolution simulations are done in slab models to mimic macroscopic surfaces, i.e. with periodic boundary conditions (PBC) in x and y, making the reactive surface perpendicular to the z axis. In other cases we may want to simulate particles, so PBC are not considered.

> Tip: Studies of different planes are possible by unit cell transformations with external programs such as VESTA.

Once our simulation box is defined, we can modify its shape as described in the next section 2B. Topography or change the constituent particles or atoms, as described in section 2C. Particle type modifiers.

# 2B.-Topography.md

> --(***) All the commands listed here are not considered if system exportation from a previous simulation is done

All the possible features you can find in a mineral surface are named as surface topography. The mineral topography has been demostrated to play an important role in the dissolution rate and mechanims. Therefore, KIMERA gathers a list of commands as complete as possible to create topographical features. Indeed, there are several ways to obtain the same simulation system.

The resulting simulation box of the previous step is modified by removing or adding regions of atoms to create the desired morphology and system shape.

There are two ways of performing the changes:

**1-** By aiming the geometric position.

For example if we want to remove form our system a cube, we introduce the following command:

```
REMOVE_CUBE                    (double)x    (double)y    (double)z    (double)side
```

or if we had it previously removed, we introduce next command to recover it

```
ADD_CUBE                       (double)x    (double)y    (double)z    (double)side
```

Additionally, we can set the cube as insoluble

```
DEFINE_INSOLUBLE_CUBE          (double)x    (double)y    (double)z    (double)side
```

For the moment, KIMERA has these commands available for cubes, spheres, ellipsoids, and planes:

```
ADD_SPHERE                     (double)x    (double)y    (double)z    (double)radius
REMOVE_SPHERE                  (double)x    (double)y    (double)z    (double)radius
DEFINE_INSOLUBLE_SPHERE        (double)x    (double)y    (double)z    (double)radius

ADD_ELLIPSOID                  (double)x    (double)y    (double)z    (double)radiusx     (doub
le)radiusy     (double)radiusz
REMOVE_ELLIPSOID               (double)x    (double)y    (double)z    (double)radiusx     (doub
le)radiusy     (double)radiusz
DEFINE_INSOLUBLE_ELLIPSOID     (double)x    (double)y    (double)z    (double)radiusx     (doub
le)radiusy     (double)radiusz
```

Note that the ellipsoid and sphere cases can be also described by using the ellipsoid general equation $Ax^2+By^2+Cz^2+Dxy+Exz+Fyz+Gx+Hy+Jz+K<1$ (Indeed the previous commands does not manage to represent rotated ellipsoids).

```
ADD_GENERAL_ELLIPSOID                (double)A    (double)B    (double)C    (double)D    (double)E
(double)F    (double)G    (double)H    (double)J    (double)K
REMOVE_GENERAL_ELLIPSOID             (double)A    (double)B    (double)C    (double)D    (double)E
(double)F    (double)G    (double)H    (double)J    (double)K
DEFINE_INSOLUBLE_GENERAL_ELLIPSOID   (double)A    (double)B    (double)C    (double)D    (double)E
(double)F    (double)G    (double)H    (double)J    (double)K
```

The planes are defined with the general equation of a plane $Ax+By+Cz+D=0$.

```
DEFINE_INSOLUBLE_PLANE               (double)A    (double)B    (double)C    (double)D    (double)dista
nce
REMOVE_PLANE                         (double)A    (double)B    (double)C    (double)D    (double)dista
nce
ADD_PLANE                            (double)A    (double)B    (double)C    (double)D    (double)dista
nce
```

Here the distance parameter set the thickness of the plane (distance is half the tickness).

> Tip: Geometric tools like Geodebra3d can be very helpful to create your system.

**2-** By aiming the system cells. The changes are produced in the cells of the system. These commands only allow changes in specific cells, or planes of cells:

```
DEFINE_AB_INSOLUBLE_CELLS            (int)pos_cell_a    (int)pos_cell_b    (int)pos_cell_c    (int)lengt
h_a    (int)length_b
DEFINE_BC_INSOLUBLE_CELLS            (int)pos_cell_a    (int)pos_cell_b    (int)pos_cell_c    (int)lengt
h_b    (int)length_c
DEFINE_AC_INSOLUBLE_CELLS            (int)pos_cell_a    (int)pos_cell_b    (int)pos_cell_c    (int)lengt
h_a    (int)length_c

REMOVE_AB_PLANE_BY_CELLS             (int)pos_cell_a    (int)pos_cell_b    (int)pos_cell_c    (int)lengt
h_a    (int)length_b
REMOVE_BC_PLANE_BY_CELLS             (int)pos_cell_a    (int)pos_cell_b    (int)pos_cell_c    (int)lengt
h_b    (int)length_c
REMOVE_AC_PLANE_BY_CELLS             (int)pos_cell_a    (int)pos_cell_b    (int)pos_cell_c    (int)lengt
h_a    (int)length_c

ADD_AB_PLANE_BY_CELLS                (int)pos_cell_a    (int)pos_cell_b    (int)pos_cell_c    (int)lengt
h_a    (int)length_b
ADD_BC_PLANE_BY_CELLS                (int)pos_cell_a    (int)pos_cell_b    (int)pos_cell_c    (int)lengt
h_b    (int)length_c
ADD_AC_PLANE_BY_CELLS                (int)pos_cell_a    (int)pos_cell_b    (int)pos_cell_c    (int)lengt
h_a    (int)length_c

ADD_CELL                             (int)pos_cell_a    (int)pos_cell_b    (int)pos_cell_c

REMOVE_CELL                          (int)pos_cell_a    (int)pos_cell_b    (int)pos_cell_c
```

Note that in all the cases, the `pos_cell` goes from 0 to `DIMENSION_A`-1, `DIMENSION_B`-1 and `DIMENSION_C`-1 respectively (see previous section).

#

## Dislocations

Special mention is given to this topographical feature because it has been experimentally observed that they play a key role in the dissolution rate and dissolution mechanisms. Dislocations can be screw or edge. Both types of dislocations are usually represented in the bibliography by considering a lineal defect where a line of atoms is removed. This approximation accounts that the energy bond in this spots is so weaken that can be neglectible. A finer approximation can be done in KIMERA, as we will explain in Question and answers section, but this basic approach is also contemplated in KIMERA.

Same as before, we can define a dislocation by aiming the geometric position, or by aiming the system cells:

**1-** By aiming the geometric position.

It allows a more complete definition since the angle respect to the axes can be defined, and the dislocation radius can be different than a unit cell multiple.

```
ADD_XY_DISLOCATION                  (double)x    (double)y    (double)radius
(optional line) FROM_Z_TO_Z         (double)bot_z        (double)top_z
(optional line) ANGLE_XZ_ANGLE_YZ   (double)angle_xz     (double)angle_yz

ADD_XZ_DISLOCATION                  (double)x    (double)z    (double)radius
(optional line) FROM_Y_TO_Y         (double)bot_y        (double)top_y
(optional line) ANGLE_XY_ANGLE_ZY   (double)angle_xz     (double)angle_yz

ADD_YZ_DISLOCATION                  (double)y    (double)z    (double)radius
(optional line) FROM_X_TO_X         (double)bot_x        (double)top_x
(optional line) ANGLE_YX_ANGLE_ZX   (double)angle_yx     (double)angle_zx
```

Two optional lines control the length of the dislocation, where does it start and where does it end, and the angle respect to the axes.

**2-** By aiming the system cells.

Same commands as before can be used to remove atoms linearly and define a dislocation but the inclination is lo longer available. They can be only defined along a, b and c directions.

```
REMOVE_AB_PLANE_BY_CELLS            (int)pos_cell_a   (int)pos_cell_b   (int)pos_cell_c   (int)lengt
h_a   (int)length_b
REMOVE_BC_PLANE_BY_CELLS            (int)pos_cell_a   (int)pos_cell_b   (int)pos_cell_c   (int)lengt
h_b   (int)length_c
REMOVE_AC_PLANE_BY_CELLS            (int)pos_cell_a   (int)pos_cell_b   (int)pos_cell_c   (int)lengt
h_a   (int)length_c
```

## Selective topography

All the previous commands can be used selectively by targeting atoms or particles of certain type. They are identical to the previous ones, but they have the `TO_TYPE` surname and an additional input parameter with the target atom type.

**1-** By aiming the geometric position.

```
ADD_XY_DISLOCATION_TO_TYPE                    (text)atom_type  (double)x    (double)y   (double)ra
dius
                                                 (optional line) FROM_Z_TO_Z          (double)bot_
z       (double)top_z
                                                 (optional line) ANGLE_XZ_ANGLE_YZ    (double)angl
e_xz    (double)angle_yz

ADD_XZ_DISLOCATION_TO_TYPE                    (text)atom_type  (double)x    (double)z   (double)ra
dius
                                                         (optional line) FROM_Y_TO_Y          (doubl
e)bot_y       (double)top_y
                                                         (optional line) ANGLE_XY_ANGLE_ZY    (doubl
e)angle_xz    (double)angle_yz

ADD_YZ_DISLOCATION_TO_TYPE                    (text)atom_type  (double)y    (double)z   (double)ra
dius
                                                         (optional line) FROM_X_TO_X          (doubl
e)bot_x       (double)top_x
                                                         (optional line) ANGLE_YX_ANGLE_ZX    (doubl
e)angle_yx    (double)angle_zx

ADD_CUBE_TO_TYPE                              (text)atom_type  (double)x    (double)y    (double)z
    (double)side
REMOVE_CUBE_TO_TYPE                           (text)atom_type  (double)x    (double)y    (double)z
    (double)side
DEFINE_INSOLUBLE_CUBE_TO_TYPE                 (text)atom_type  (double)x    (double)y    (double)z
    (double)side

ADD_SPHERE_TO_TYPE                            (text)atom_type  (double)x    (double)y    (double)z
    (double)radius
REMOVE_SPHERE_TO_TYPE                         (text)atom_type  (double)x    (double)y    (double)z
    (double)radius
DEFINE_INSOLUBLE_SPHERE_TO_TYPE               (text)atom_type  (double)x    (double)y    (double)z
    (double)radius

ADD_ELLIPSOID_TO_TYPE                         (text)atom_type  (double)x    (double)y    (double)z
    (double)radiusx     (double)radiusy    (double)radiusz
REMOVE_ELLIPSOID_TO_TYPE                      (text)atom_type  (double)x    (double)y    (double)z
    (double)radiusx     (double)radiusy    (double)radiusz
DEFINE_INSOLUBLE_ELLIPSOID_TO_TYPE            (text)atom_type  (double)x    (double)y    (double)z
    (double)radiusx     (double)radiusy    (double)radiusz

ADD_GENERAL_ELLIPSOID_TO_TYPE                 (text)atom_type  (double)A    (double)B    (double)C
    (double)D    (double)E    (double)F    (double)G    (double)H    (double)J    (double)K
REMOVE_GENERAL_ELLIPSOID_TO_TYPE              (text)atom_type  (double)A    (double)B    (double)C
    (double)D    (double)E    (double)F    (double)G    (double)H    (double)J    (double)K
DEFINE_INSOLUBLE_GENERAL_ELLIPSOID_TO_TYPE    (text)atom_type  (double)A    (double)B    (double)C
    (double)D    (double)E    (double)F    (double)G    (double)H    (double)J    (double)K

DEFINE_INSOLUBLE_PLANE_TO_TYPE                (text)atom_type  (double)A    (double)B    (double)C
    (double)D    (double)distance
REMOVE_PLANE_TO_TYPE                          (text)atom_type  (double)A    (double)B    (double)C
    (double)D    (double)distance
ADD_PLANE_TO_TYPE                             (text)atom_type  (double)A    (double)B    (double)C
    (double)D    (double)distance
```

**2-** By aiming the system cells.

```
DEFINE_AB_INSOLUBLE_CELLS_TO_TYPE               (text)atom_type  (int)pos_cell_a   (int)pos_cell_b
 (int)pos_cell_c   (int)length_a   (int)length_b
DEFINE_BC_INSOLUBLE_CELLS_TO_TYPE               (text)atom_type  (int)pos_cell_a   (int)pos_cell_b
 (int)pos_cell_c   (int)length_b   (int)length_c
DEFINE_AC_INSOLUBLE_CELLS_TO_TYPE               (text)atom_type  (int)pos_cell_a   (int)pos_cell_b
 (int)pos_cell_c   (int)length_a   (int)length_c

REMOVE_AB_PLANE_BY_CELLS_TO_TYPE                (text)atom_type  (int)pos_cell_a   (int)pos_cell_b
 (int)pos_cell_c   (int)length_a   (int)length_b
REMOVE_BC_PLANE_BY_CELLS_TO_TYPE                (text)atom_type  (int)pos_cell_a   (int)pos_cell_b
 (int)pos_cell_c   (int)length_b   (int)length_c
REMOVE_AC_PLANE_BY_CELLS_TO_TYPE                (text)atom_type  (int)pos_cell_a   (int)pos_cell_b
 (int)pos_cell_c   (int)length_a   (int)length_c

ADD_AB_PLANE_BY_CELLS_TO_TYPE                   (text)atom_type  (int)pos_cell_a   (int)pos_cell_b
 (int)pos_cell_c   (int)length_a   (int)length_b
ADD_BC_PLANE_BY_CELLS_TO_TYPE                   (text)atom_type  (int)pos_cell_a   (int)pos_cell_b
 (int)pos_cell_c   (int)length_b   (int)length_c
ADD_AC_PLANE_BY_CELLS_TO_TYPE                   (text)atom_type  (int)pos_cell_a   (int)pos_cell_b
 (int)pos_cell_c   (int)length_a   (int)length_c

ADD_CELL_TO_TYPE                                (text)atom_type  (int)pos_cell_a   (int)pos_cell_b
 (int)pos_cell_c
REMOVE_CELL_TO_TYPE                             (text)atom_type  (int)pos_cell_a   (int)pos_cell_b
 (int)pos_cell_c
```

In addtion to change the topography and shape of our system, we can perform changes in the type of the atoms, as we will see in the next section.

# 2C.-Particle-type-modifiers.md

> --(***) All the commands listed here are not considered if system exportation from a previous simulation is done

Once we have our system defined with the desired shape and topography, we can change locally the constituent atoms or particles. Very similar commands to the shown ones in the previous section do the job. When applying, a region of the system sees its type changed into the desired final atom type:

```
CHANGE_AB_PLANE_TYPE                              (int)pos_cell_a   (int)pos_cell_b   (int)pos_cell_c
  (int)length_a   (int)length_b    (text)final_atom_type
CHANGE_AC_PLANE_TYPE                              (int)pos_cell_a   (int)pos_cell_b   (int)pos_cell_c
  (int)length_a   (int)length_c    (text)final_atom_type
CHANGE_BC_PLANE_TYPE                              (int)pos_cell_a   (int)pos_cell_b   (int)pos_cell_c
  (int)length_b   (int)length_c    (text)final_atom_type

CHANGE_CUBE_TYPE                                 (double)x    (double)y    (double)z    (double)side
(text)final_atom_type

CHANGE_XY_DISLOCATION_TYPE                       (double)x    (double)y   (double)radius  (text)final_
atom_type
                                                  (optional line) FROM_Z_TO_Z          (double)bot_z
     (double)top_z
                                                  (optional line) ANGLE_XZ_ANGLE_YZ    (double)angle_x
z    (double)angle_yz

CHANGE_XZ_DISLOCATION_TYPE                       (double)x    (double)z   (double)radius   (text)final
_atom_type
                                                  (optional line) FROM_Y_TO_Y          (double)bot_y
     (double)top_y
                                                  (optional line) ANGLE_XY_ANGLE_ZY    (double)angle_x
z    (double)angle_yz

CHANGE_YZ_DISLOCATION_TYPE                       (double)y    (double)z   (double)radius   (text)final
_atom_type
                                                 (optional line) FROM_X_TO_X          (double)bot_x
    (double)top_x
                                                 (optional line) ANGLE_YX_ANGLE_ZX    (double)angle_yx
    (double)angle_zx

CHANGE_PLANE_TYPE                                (double)A    (double)B    (double)C    (double)D   (d
ouble)distance   (text)final_atom_type
CHANGE_SPHERE_TYPE                               (double)x    (double)y    (double)z    (double)radius
  (text)final_atom_type
CHANGE_ELLIPSOID_TYPE                            (double)x    (double)y    (double)z    (double)radius
x     (double)radiusy    (double)radiusz    (text)final_atom_type
CHANGE_GENERAL_ELLIPSOID_TYPE                    (double)A    (double)B    (double)C    (double)D    (
double)E   (double)F   (double)G   (double)H   (double)J   (double)K    (text)final_atom_type
```

Note that:

- The parameters of this last command `CHANGE_GENERAL_ELLIPSOID_TYPE` indicate the coeficients of the ellipsoid general equation $Ax^2+By^2+Cz^2+Dxy+Exz+Fyz+Gx+Hy+Jz+K<1$.
- The parameters of the `CHANGE_PLANE_TYPE` command indicate the coeficients of the plane general equation $Ax+By+Cz+D=0$.
- The `pos_cell` goes from 0 to `DIMENSION_A`-1, `DIMENSION_B`-1 and `DIMENSION_C`-1 respectively (see 2A. Simulation box section).

## Selective particle type modifiers

All the previous commands can be used selectively targeting atoms or particles of certain type. They are identical to the previous ones, but they have the `TO_TYPE` surname and an additional input parameter with the target type:

```
CHANGE_AB_PLANE_TYPE_TO_TYPE                    (text)atom_type  (int)pos_cell_a   (int)pos_cell_b
(int)pos_cell_c   (int)length_a   (int)length_b    (text)final_atom_type
CHANGE_AC_PLANE_TYPE_TO_TYPE                    (text)atom_type  (int)pos_cell_a   (int)pos_cell_b
(int)pos_cell_c   (int)length_b   (int)length_c    (text)final_atom_type
CHANGE_BC_PLANE_TYPE_TO_TYPE                    (text)atom_type  (int)pos_cell_a   (int)pos_cell_b
(int)pos_cell_c   (int)length_a   (int)length_c    (text)final_atom_type

CHANGE_CUBE_TYPE_TO_TYPE                        (text)atom_type  (double)x    (double)y    (double)z
   (double)side  (text)final_atom_type

CHANGE_XY_DISLOCATION_TYPE_TO_TYPE              (text)atom_type  (double)x    (double)y   (double)rad
ius  (text)final_atom_type
                                                 (optional line) FROM_Z_TO_Z          (double)bot_z
     (double)top_z
                                                 (optional line) ANGLE_XZ_ANGLE_YZ    (double)angle_x
z    (double)angle_yz

CHANGE_XZ_DISLOCATION_TYPE_TO_TYPE              (text)atom_type  (double)x    (double)z   (double)rad
ius   (text)final_atom_type
                                                 (optional line) FROM_Y_TO_Y          (double)bot_y
     (double)top_y
                                                 (optional line) ANGLE_XY_ANGLE_ZY    (double)angle_x
z    (double)angle_yz

CHANGE_YZ_DISLOCATION_TYPE_TO_TYPE              (text)atom_type  (double)y    (double)z   (double)rad
ius   (text)final_atom_type
                                                (optional line) FROM_X_TO_X          (double)bot_x
    (double)top_x
                                                (optional line) ANGLE_YX_ANGLE_ZX    (double)angle_yx
    (double)angle_zx

CHANGE_PLANE_TYPE_TO_TYPE                       (text)atom_type  (double)A    (double)B    (double)C
   (double)D   (double)distance   (text)final_atom_type
CHANGE_SPHERE_TYPE_TO_TYPE                      (text)atom_type  (double)x    (double)y    (double)z
   (double)radius  (text)final_atom_type
CHANGE_ELLIPSOID_TYPE_TO_TYPE                   (text)atom_type  (double)x    (double)y    (double)z
   (double)radiusx     (double)radiusy    (double)radiusz    (text)final_atom_type
CHANGE_GENERAL_ELLIPSOID_TYPE_TO_TYPE           (text)atom_type  (double)A    (double)B    (double)C
   (double)D    (double)E   (double)F   (double)G   (double)H   (double)J   (double)K    (text)fina
l_atom_type
```

At this point we have our system perfectly defined. In the next step we are going to define the dissolution events that may happen in the system.

# 2D.-Events-definition.md

> *--(*) If system exportation from a previous simulation is done, all the commands listed here are compulsary as they were in the original definition. Only the change of the ED, EP, FFD, FFP, and DG* values is possible.

The dissolution reactions (or events as commonly known in KMC algorithm) have a rate to happen that follows an Arrhenius equation:

$$r = f_{\mathrm{f}} \cdot \exp\left(-\frac{E_{\mathrm{B}}}{k_{\mathrm{B}} \cdot T}\right)$$

Where the pre-exponential factor _f_$_{f}$ is named fundamental frequency ($s^{-1}$), and _E_$_{B}$ ($kJ\ mol^{-1}$) is the energy barrier for the transition, _k_$_{B}$ the Boltzman constant ($1.380\times10^{-23}\ J\ K^{-1}$) and _T_ the temperature (K).

It is neccesary to take also into account the precipitation events to study the dissolution at close to equilibrium conditions, when the dissolution is unlikely to happen and the Gibbs free energy $\Delta G^{*}$ has values closer to 0. Therefore, the previous equation is splitted in the two following ones:

$$\begin{cases} r_{\mathrm{D}} = f_{\mathrm{D}} \cdot \exp\left(-\dfrac{E_{\mathrm{D}}}{k_{\mathrm{B}} \cdot T}\right) \\ r_{\mathrm{P}} = f_{\mathrm{P}} \cdot \exp\left(-\dfrac{E_{\mathrm{P}} - \Delta G^{*}}{k_{\mathrm{B}} \cdot T}\right) \end{cases}$$

Here the fundamental frequency has splitted in _f_D and _f_P, and the energy barrier in _E_D and _E_P. The Gibbs free energy $\Delta G^{*}$ is related to the macroscopical $\Delta G$ but such relation is dependent with the chosen model. For a deeper explanation, refer to the article. If we make a proper description of the events and know these model parameters, which can be calculated by ab initio simulations or obtained in the bibliography, we can manage an acurrate time evolution of a dissolving mineral.

> Tip: we can ensure very far form equilibrium conditions by considering a very low value of $\Delta G^{*}$ when the precipitation term is neglected.

The command that starts with the event definition is `DEFINE_DISSOLUTION_EVENT`. We will usually use this command once for each type of atom (or particle) in our system.

From the experiments we know that the energy barriers of the reactions changes with the neighbourhood of an atom. We have designed KIMERA in the same way. The definition of the dissolution events is done by indicating the Arrhenius equation parameters, which change with the local neighbourhood. The energy barrier _E_B for an atom or particle to be removed from the system changes with the number of 'contributors' or neighbours around it.

$$E_{\mathrm{D}} = \sum_{i=1}^{n} E_{\mathrm{d}_i} \qquad E_{\mathrm{P}} = \sum_{i=1}^{n} E_{\mathrm{p}_i}$$

where _E_di and _E_pi are the dissolution and precipitation energy barrier contribution to the total _E_D and _E_P respectively and _n_ is the total number of them.

As KIMERA uses an on-lattice description, we can identify each set of contributors according to their distance to the particle which is dissolving. As we will see in the 2G. Other parameters section, we can set the threshold of this distance to give us more versatility in grouping the contributors. Since there may be two different types of contributors at the same distance to the target dissolving atom, we also need to indicate the contributor atom type.

KIMERA search for the contributors in same cell of the target atom, and in the 26 sourrounding cells (up, down, sides, and diagonals). If we need to recognise a contributor out of this range, we will need to redefine the unit cell as a supercell containing a bigger piece of mineral.

The energy barrier of dissolution and precipitation can be determined by more than one different sets of contributors. Every time we use a `NEIGHBOUR` command, like `NEIGHBOUR`, `NEIGHBOUR_LINKED`, `NEIGHBOUR_LINKED_DISSOLVED`, `NEIGHBOUR_DIRECT_LIST`, `NEIGHBOUR_LINKED_DIRECT_LIST` or `NEIGHBOUR_LINKED_DISSOLVED_DIRECT_LIST`, we are refering to a different set and so we need to specify the distance to them and their type. The `DEFINE_DISSOLUTION_EVENT` command needs at least one of the previous to indicate the rate of the dissolution. There is no limit in the number of contributors.

Typically in the bibliography, the energy barrier has been considerered to change linearly with the contributors. The commands that offer a simpler lineal description are `NEIGHBOUR`, `NEIGHBOUR_LINKED` and `NEIGHBOUR_LINKED_DISSOLVED`. Nevertheless, KIMERA also allows to set a specific contribution for each contributor. The commands that allow specific description have the `DIRECT_LIST` surname; `NEIGHBOUR_DIRECT_LIST`, `NEIGHBOUR_LINKED_DIRECT_LIST` and `NEIGHBOUR_LINKED_DISSOLVED_DIRECT_LIST`.

The `LINKED` feature specify that the contribution is only consedered if other atoms are still on the system. To difine the 'linked' atoms, two distances are given; one to the contributor atom, and other to the target dissolving atom (or origin atom) in which we are defining the event. There is not limit in the number of 'linked' atoms.

Same as before, but if the contribution is only considered when other atoms are not longer in the system, we have the commands with `LINKED_DISSOLVED` surname.

Finally, we can specify one special kind of atoms, named `AFFECTED`, which automatically leave the mineral when the 'father' atoms do. The number of 'affected' atoms has no limit.

To recapitulate, the events defintion command list is the following:

```
DEFINE_DISSOLUTION_EVENT            (text)target_atom_type

    (optional line) NEIGHBOUR                    (text)target_neighbour_type   (double)distance_to_n
eighbour       (double)ED_in_KT_units       (double)EP_in_KT_units        (int)neighbours_to_bulk


    (optional line) NEIGHBOUR_LINKED             (text)target_neighbour_type   (double)distance_to_n
eighbour       LINK  (text)target_linked_type    (double)distance_from_origin_to_linked    (double)d
istance_from_neighbour_to_linked

                                                 LINK  (text)target_linked_type    (double)di
stance_from_origin_to_linked    (double)distance_from_neighbour_to_linked

                                                 LINK  ......................    .........
.............................    ......................................    (double)ED_in_KT_uni
ts       (double)EP_in_KT_units      (int)neighbours_to_bulk

    (optional line) NEIGHBOUR_LINKED_DISSOLVED   (text)target_neighbour_type   (double)distance_to_n
eighbour       LINK  (text)target_linked_type    (double)distance_from_origin_to_linked    (double)d
istance_from_neighbour_to_linked

                                                 LINK  (text)target_linked_type    (double)di
stance_from_origin_to_linked    (double)distance_from_neighbour_to_linked

                                                 LINK  ......................    .........
.............................    ......................................    (double)ED_in_KT_uni
ts       (double)EP_in_KT_units      (int)neighbours_to_bulk

    (optional line) NEIGHBOUR_DIRECT_LIST        (text)target_neighbour_type   (double)distance_to_
neighbour      LIST_LENGTH   (int)list_length       (double)ED1   (double)EP1    (double)ED2   (double
)EP2   .........    .........    (int)neighbours_to_bulk

    (optional line) NEIGHBOUR_LINKED_DIRECT_LIST  (text)target_neighbour_type   (double)distance_to_
neighbour      LINK  (text)target_linked_type    (double)distance_from_origin_to_linked    (double)d
istance_from_neighbour_to_linked

                                                 LINK  (text)target_linked_type    (double)di
stance_from_origin_to_linked    (double)distance_from_neighbour_to_linked

                                                 LINK  ......................    .........
.............................    ......................................


                                                 LIST_LENGTH   (int)list_length       (double
)ED1   (double)EP1    (double)ED2   (double)EP2   .........    .........    (int)neighbours_to_bulk

    (optional line) NEIGHBOUR_LINKED_DISSOLVED_DIRECT_LIST   (text)target_neighbour_type   (double)d
istance_to_neighbour       LINK  (text)target_linked_type    (double)distance_from_origin_to_linked
   (double)distance_from_neighbour_to_linked

                                                                 LINK  (text)target_linked_type
  (double)distance_from_origin_to_linked    (double)distance_from_neighbour_to_linked

                                                                 LINK  ......................
   ....................................    ......................................


                                                                 LIST_LENGTH   (int)list_length
     (double)ED1   (double)EP1    (double)ED2   (double)EP2   .........    .........    (int)neighbou
rs_to_bulk
    (optional line) AFFECTED                     (text)target_affected_type    (double)distance_to_a
ffected
                                                             .........                    .........
.................    ........................
    (optional line) FFD                          (double)dissolution_fundamental_frequency
    (optional line) FFP                          (double)precipitation_fundamental_frequency
    (optional line) DG*                          (double)event_delta_G
```

where `DG*` is $\Delta G^*$, `FFD` is $f_D$ and `FFP` is $f_P$.

Next following points are important to be noted:

- We have to define one event (or more) for each atom type in our system, or consider them as 'affected'. Otherwise they will never dissolve.
- Related with the previous, if we have defined 'B' atoms as neighbours of 'A' atoms, we must also indicate that 'A' atoms are neighbours of 'B' atoms to tell KIMERA they are interconnected.

> Tip: We must define events for 'B' atoms respect to the number of 'A' atoms, but if the existence of atoms A has not any influence on the dissolution of the first, we can just consider a very high value of energy barriers _E_$_D$ and _E_$_P$.

- The `DIRECT_LIST` commands needs the specification of _n_, the number of contributors of that type, in the `list_length` parameter.
- In the `neighbours_to_bulk` parameter we specify if a type of neighbour makes the target atom a bulk atom. A bulk atom does not suffer dissolution until the quantity of that neighbour type is lower than the number we have specified. This represents the accessibility of the atom to react with the solvent after the dissolution of one of its neighbours. If a '0' value is considered, that type of neighbour has not effect in the bulk definition.

At this point we have the system with their possible events perfectly defined. KIMERA automatically creates then a reactive surface in where the dissolution is going to start. As the reactive surface may not be to our liking, the next section shows the commands that allow to modify it.

# 2E.-Reactive-surface.md

> --(***) All the commands listed here are not considered if system exportation from a previous simulation is done

KIMERA automatically creates an initial reactive surface from the whole system which greatly increases the simulation performance. In order to define it, KIMERA considers the PBC, the atoms defined as insoluble, and the atoms defined as bulk atoms. Nevertheless, the initial surface may not be well defined. Impurities, dislocations, or other features that decrease the homogeneity of the atomic structure produce initial reactive sites that should not initially exist since they are not in contact with the solvent (the interior of a dislocation, for example). Therefore, similar commands to previous ones explained in the Topography and modifiers sections are available to modify it:

```
REMOVE_AB_PLANE_FROM_SURFACE                  (int)pos_cell_a   (int)pos_cell_b   (int)pos_cell_c
  (int)length_a   (int)length_b
ADD_AB_PLANE_TO_SURFACE                       (int)pos_cell_a   (int)pos_cell_b   (int)pos_cell_c
  (int)length_a   (int)length_b

REMOVE_AC_PLANE_FROM_SURFACE                  (int)pos_cell_a   (int)pos_cell_b   (int)pos_cell_c
  (int)length_a   (int)length_c
ADD_AC_PLANE_TO_SURFACE                       (int)pos_cell_a   (int)pos_cell_b   (int)pos_cell_c
  (int)length_a   (int)length_c

REMOVE_BC_PLANE_FROM_SURFACE                  (int)pos_cell_a   (int)pos_cell_b   (int)pos_cell_c
  (int)length_b   (int)length_c
ADD_BC_PLANE_TO_SURFACE                       (int)pos_cell_a   (int)pos_cell_b   (int)pos_cell_c
  (int)length_b   (int)length_c

REMOVE_CUBE_FROM_SURFACE                      (double)x   (double)y   (double)z   (double)side
ADD_CUBE_TO_SURFACE                           (double)x   (double)y   (double)z   (double)side

REMOVE_XY_DISLOCATION_FROM_SURFACE            (double)x   (double)y   (double)radius
                                              (optional line) FROM_Z_TO_Z          (double)bot_z
     (double)top_z
                                              (optional line) ANGLE_XZ_ANGLE_YZ    (double)angle_x
z    (double)angle_yz

ADD_XY_DISLOCATION_TO_SURFACE                 (double)x   (double)y   (double)radius
                                              (optional line) FROM_Z_TO_Z          (double)bot_z
     (double)top_z
                                              (optional line) ANGLE_XZ_ANGLE_YZ    (double)angle_x
z    (double)angle_yz

REMOVE_XZ_DISLOCATION_FROM_SURFACE            (double)x   (double)z   (double)radius
                                              (optional line) FROM_Y_TO_Y          (double)bot_y
```

```
                                                    (optional line) FROM_Y_TO_Y          (double)bot_y
     (double)top_y
                                                    (optional line) ANGLE_XY_ANGLE_ZY    (double)angle_x
z    (double)angle_yz

ADD_XZ_DISLOCATION_TO_SURFACE                       (double)x    (double)z    (double)radius
                                                    (optional line) FROM_Y_TO_Y          (double)bot_y
     (double)top_y
                                                    (optional line) ANGLE_XY_ANGLE_ZY    (double)angle_x
z    (double)angle_yz

REMOVE_YZ_DISLOCATION_FROM_SURFACE                  (double)y    (double)z    (double)radius
                                                    (optional line) FROM_X_TO_X          (double)bot_x
     (double)top_x
                                                    (optional line) ANGLE_YX_ANGLE_ZX    (double)angle_y
x    (double)angle_zx

ADD_YZ_DISLOCATION_TO_SURFACE                       (double)y    (double)z    (double)radius
                                                    (optional line) FROM_X_TO_X          (double)bot_x
     (double)top_x
                                                    (optional line) ANGLE_YX_ANGLE_ZX    (double)angle_y
x    (double)angle_zx

REMOVE_PLANE_FROM_SURFACE                           (double)A    (double)B    (double)C    (double)D    (
double)distance
ADD_PLANE_TO_SURFACE                                (double)A    (double)B    (double)C    (double)D    (
double)distance

REMOVE_SPHERE_FROM_SURFACE                          (double)x    (double)y    (double)z    (double)radiu
s
ADD_SPHERE_TO_SURFACE                               (double)x    (double)y    (double)z    (double)radiu
s

REMOVE_ELLIPSOID_FROM_SURFACE                       (double)x    (double)y    (double)z    (double)radiu
sx     (double)radiusy     (double)radiusz
ADD_ELLIPSOID_TO_SURFACE                            (double)x    (double)y    (double)z    (double)radiu
sx     (double)radiusy     (double)radiusz

REMOVE_GENERAL_ELLIPSOID_TO_SURFACE                 (double)A    (double)B    (double)C    (double)D
(double)E    (double)F    (double)G    (double)H    (double)J    (double)K
ADD_GENERAL_ELLIPSOID_TO_SURFACE                    (double)A    (double)B    (double)C    (double)D
(double)E    (double)F    (double)G    (double)H    (double)J    (double)K
```

Note that:

- The parameters of these last commands `ADD_GENERAL_ELLIPSOID_TO_SURFACE` and `REMOVE_GENERAL_ELLIPSOID_FROM_SURFACE` indicate the coeficients of the ellipsoid general equation $Ax^2+By^2+Cz^2+Dxy+Exz+Fyz+Gx+Hy+Jz+K<1$.
- The parameters of the `ADD_PLANE_TO_SURFACE` and `REMOVE_PLANE_FROM_SURFACE` commands indicate the coeficients of the plane general equation $Ax+By+Cz+D=0$.
- The `pos_cell` goes from 0 to `DIMENSION_A`-1, `DIMENSION_B`-1 and `DIMENSION_C`-1 respectively (see 2A. Simulation box section)

## Selective modification of the reactive surface

All the previous commands can be used selectively targeting atoms or particles of certain type. They are identical to the previous ones, but they have the `TO_TYPE` surname and an additional input parameter with the target type:

```
REMOVE_AB_PLANE_FROM_SURFACE_TO_TYPE            (text)atom_type  (int)pos_cell_a   (int)pos_cell_b
 (int)pos_cell_c   (int)length_a   (int)length_b
ADD_AB_PLANE_TO_SURFACE_TO_TYPE                 (text)atom_type  (int)pos_cell_a   (int)pos_cell_b
 (int)pos_cell_c   (int)length_a   (int)length_b

REMOVE_AC_PLANE_FROM_SURFACE_TO_TYPE            (text)atom_type  (int)pos_cell_a   (int)pos_cell_b
 (int)pos_cell_c   (int)length_a   (int)length_c
ADD_AC_PLANE_TO_SURFACE_TO_TYPE                 (text)atom_type  (int)pos_cell_a   (int)pos_cell_b
 (int)pos_cell_c   (int)length_a   (int)length_c
```

```
REMOVE_BC_PLANE_FROM_SURFACE_TO_TYPE            (text)atom_type  (int)pos_cell_a   (int)pos_cell_b
 (int)pos_cell_c   (int)length_b   (int)length_c
ADD_BC_PLANE_TO_SURFACE_TO_TYPE                 (text)atom_type  (int)pos_cell_a   (int)pos_cell_b
 (int)pos_cell_c   (int)length_b   (int)length_c

REMOVE_CUBE_FROM_SURFACE_TO_TYPE                (text)atom_type  (double)x    (double)y    (double)z
    (double)side
ADD_CUBE_TO_SURFACE_TO_TYPE                     (text)atom_type  (double)x    (double)y    (double)z
    (double)side

REMOVE_XY_DISLOCATION_FROM_SURFACE_TO_TYPE      (text)atom_type  (double)x    (double)y   (double)ra
dius
                                                (optional line) FROM_Z_TO_Z          (double)bot_z
     (double)top_z
                                                (optional line) ANGLE_XZ_ANGLE_YZ    (double)angle_x
z    (double)angle_yz

ADD_XY_DISLOCATION_TO_SURFACE_TO_TYPE           (text)atom_type  (double)x    (double)y   (double)ra
dius
                                                (optional line) FROM_Z_TO_Z          (double)bot_z
     (double)top_z
                                                (optional line) ANGLE_XZ_ANGLE_YZ    (double)angle_x
z    (double)angle_yz

REMOVE_XZ_DISLOCATION_FROM_SURFACE_TO_TYPE      (text)atom_type  (double)x    (double)z   (double)ra
dius
                                                (optional line) FROM_Y_TO_Y          (double)bot_y
     (double)top_y
                                                (optional line) ANGLE_XY_ANGLE_ZY    (double)angle_x
z    (double)angle_yz

ADD_XZ_DISLOCATION_TO_SURFACE_TO_TYPE           (text)atom_type   (double)x    (double)z   (double)r
adius
                                                (optional line) FROM_Y_TO_Y          (double)bot_y
     (double)top_y
                                                (optional line) ANGLE_XY_ANGLE_ZY    (double)angle_x
z    (double)angle_yz

REMOVE_YZ_DISLOCATION_FROM_SURFACE_TO_TYPE      (text)atom_type  (double)y    (double)z   (double)ra
dius
                                                (optional line) FROM_X_TO_X          (double)bot_x
     (double)top_x
                                                (optional line) ANGLE_YX_ANGLE_ZX    (double)angle_y
x    (double)angle_zx

ADD_YZ_DISLOCATION_TO_SURFACE_TO_TYPE           (text)atom_type  (double)y    (double)z   (double)ra
dius
                                                (optional line) FROM_X_TO_X          (double)bot_x
     (double)top_x
                                                (optional line) ANGLE_YX_ANGLE_ZX    (double)angle_y
x    (double)angle_zx

REMOVE_PLANE_FROM_SURFACE_TO_TYPE               (text)atom_type  (double)A    (double)B    (double)C
    (double)D    (double)distance
ADD_PLANE_TO_SURFACE_TO_TYPE                    (text)atom_type  (double)A    (double)B    (double)C
    (double)D    (double)distance

REMOVE_SPHERE_FROM_SURFACE_TO_TYPE              (text)atom_type  (double)x    (double)y    (double)z
    (double)radius
ADD_SPHERE_TO_SURFACE_TO_TYPE                   (text)atom_type  (double)x    (double)y    (double)z
    (double)radius

REMOVE_ELLIPSOID_FROM_SURFACE_TO_TYPE           (text)atom_type  (double)x    (double)y    (double)z
    (double)radiusx     (double)radiusy    (double)radiusz
ADD_ELLIPSOID_TO_SURFACE_TO_TYPE                (text)atom_type  (double)x    (double)y    (double)z
    (double)radiusx     (double)radiusy    (double)radiusz

REMOVE_GENERAL_ELLIPSOID_TO_SURFACE_TO_TYPE     (text)atom_type  (double)A    (double)B    (double)C
    (double)D    (double)E    (double)F    (double)G    (double)H    (double)J    (double)K
ADD_GENERAL_ELLIPSOID_TO_SURFACE_TO_TYPE        (text)atom_type  (double)A    (double)B    (double)C
    (double)D    (double)E    (double)F    (double)G    (double)H    (double)J    (double)K
```

At this point we can perform our simulation to study the mineral dissolution. In the next step we will see the neccesary commands to indicate KIMERA the output files we need.

# 2F.-Output-requests.md

> --(*) All the commands listed here can be freely modified if system exportation from a previous simulation is done

The KMC algorithm within KIMERA makes the dissolution events happen one by one. In each simulation step, the program evaluates to print the information of the simulation in the output files according to what has been specified by the user. Each command enables the printing of a type of output file. The number in the command indicates the steps that are going to be printed uniformely, in time or in steps, depending on the ending clause (see next section).

The available output files are the following:

#

```
DATA_ANALYSIS                       (int)
```

It constains information of the quantity and fraction of the dissolved atoms of each type with time, as well as the derivative. Moreover, if we are studying a particle (with no PBC in any direction), it also reports information about the gyradius, but we need to specify the mass of each type of atom (see 2G. Other parameters section). This file is essential to study the time evolution of the dissolution rate.

#

```
MEAN_DISSOLVED_ANALYSIS             (int)
```

This file constains information of the avarage quantity of neighbours an atom has when it dissolves. This file is essential to obtain the $\Delta G$ and $\Delta G^*$ relation.

#

```
LAYER_ANALYSIS                      (optional)A   (optional)B   (optional)C   (int)
```

This command prints a maximun of three different files, one for each system direction. They contain information of the time evolution of the quantity of atoms that each system layer has. Each system layer has a tickness of one unit cell.

#

```
BOX_FRAMES                          (int)
```

This command prints the file with snapshots of the whole system with time, in LAMMPS format. As this file contains several times all the system positions, the size of the file may be too big. It is readable by visualization tools as OVITO.

#

```
SURFACE_FRAMES                      (int)
```

This command allows to obtain a similar file to the previous one but it only contains the positions of the reactive surface, thus it is much smaller. Same as before, the output file has LAMMPS format and can be viewed by OVITO program.

```
INITIAL_KIMERA_STATE
FINAL_KIMERA_STATE
```

If these commands are used, the initial and the final states of the system are printed respectively. They have their own KIMERA format, as seen in 3. KIMERA format section. These output files can be used in subsequent simulations to save time, although the

use of the commands are limited (see 2. Input structure section). In the case that `FINAL_KIMERA_STATE` has been used and KIMERA encounters an error during the simulation, it still prints the final state just before the error.

#

At this point, the only step remaining is to indicate when we want our simulation to finish. This, and other side commands like the name we give to the simulation or the number of cores we want to use to to run it are explained in the next 2G. Other parameters section.

# 2G.-Other-parameters.md

> --(*) All the commands listed here can be freely modified if system exportation from a previous simulation is done, with the expection of the `SEED_BOX` command, which its change does not have any effect.

One of the most important commands of the simulation is to tell KIMERA when to stop it, the ending clause. The number of the snapshots of the output files are calculated respect to this ending clause. A lower than expeced quantity of snapshots may be obtained if the system has expired before the ending clause. On the other hand, if we consider a too short ending clause, we may not appreciate any evolution of the system.

There are two ways two set the ending clause, indicating the `TARGET_TIME` or indicating the `TARGET_STEP`. In KIMERA it is easier to use the `TARGET_STEP` command. As one step removes one particle of the system, we know that the maximun number of steps are the total maximun of atoms (we have to rest the atoms defined as insoluble and the atoms removed by the topography commands). The `TARGET_TIME` is more difficult to know beforehand in a KMC algorithm. Nevertheless KIMERA can make an estimation of it using the `ESTIMATE_TIME` command. The following three commands are incompatible with each other:

```
TARGET_TIME                     (double)
ESTIMATE_TIME
TARGET_STEP                     (int)
```

#

If we are performing an study of a particle with no PBC conditions, we can obtain the time evolution of the gyradius with the `DATA_ANALYSIS` command (see previous 2F. Output requests section). Nevertheless, we need to indicate the mass of the constituent atoms of our system in atomic mass units) (Dalton or u). One command for each atom type.

```
SET_MASS                        (text)element       (double)mass
```

#

Some of the loops of KIMERA are parallelized. This means that we can reduce our simulation time if we run it using several cores.

```
PARALLELIZE_SIMULATION              (int)cores
```

#

One of the most time consuming steps of the KMC algorithm is the searching of the event that is going to happen in an array of values. There are different ways to do this searching that offer different performances. KIMERA can use the named binary search or the named lineal seach, the first one by default. Although the binary search is more efficient, it is not parallelized, so the lineal search can be considered if we are using several cores.

```
LINEAL_SEARCH
```

> Tip: If you need your simulation soon, use lineal search with several cores. If you are runing several set of simulations (to get the dispersion of the dissolution rate value for example) use 1 core for each one with binary search. More details are given in the article.

As we have explained in the 2A. Simulation box section, we can specify the occupancy of the atoms in the unit cell. KIMERA

creates randomly the system taking into account that occupancy. But we can give a seed to recreate always the same system.

```
SEED_BOX                          (int)
```

#

The KMC algorithm uses random numbers to describe stochastic processes as is the dissolution of a mineral. Therefore, we will never get two identical simulations unless we indicate it with the simulation seed.

```
SEED_SIMULATION                   (int)
```

Note that the `SEED_SIMULATION` and the `PARALLELIZE_SIMULATION` commands are incompatible if the number of cores > 1.

#

As explained in the 2D. Events definition section, KIMERA recognises an atom as neighbour by indicating the distance to it. It may happen that we want to group several neighbours into a contributor set whose distance is sligthly different to the target atom. We can group them all by changing the distance threshold with the following command.

```
DISTANCE_ACCURACY                 (double)
```

> Tip: For example, imagine a mineral in where a 'A' atom has 4 neighbouring 'B' atoms, two of them at 1.550 Å and the other two at 1.570 Å. To group them all, we can set a `distance_to_neighbour` of 1.560 Å and then change `DISTANCE_ACCURACY` to 0.011 Å.

#

When defining the events, we are usually going to indicate the value of ΔG, $f_d$ and $f_p$ for each event. Other option is to indicate its value using the following general commands. The event definiton without any of the commands `DG*`, `FFD` and `FFP` will see their values replaced with the values of these ones, respectively:

```
DELTA_G*                          (double)
DISSOLUTION_FUNDAMENTAL_FREQUENCY       (double)
PRECIPITATION_FUNDAMENTAL_FREQUENCY     (double)
```

#

We can give to our simulation a name. All the output files will get this name.

```
WORK_NAME              (text)
```

#

These were all the commands of KIMERA. in the next 4. Questions and answers section you can find the answer to some question you may have.

# 3.-KIMERA-format.md

The goal of a Kimera file is to enhance the efficiency of succesive simulations with the same system. The data in a kimera file is as follows:

1- Cells and their corresponding atoms forming the box of the system. Next code in applied in the format of `type_atom`:

```
1:  NORMAL
-1: INSOLUBLE
-2: REVOMED_ATOM (it is already dissolved)
```

2- Neighbours of each atom and the distance to each one.

3- Affected atoms for every atom. An affected atom dissolves as soon as the father does.

4- Linked atoms, related with the neighbour by the order that they follow in this file. The contribution of one neighbour is only considered if the linked atom exist. Next code in applied in the format of `type_link`:

```
0:  NOT LINKED
1:  LINKED
-1: LINKED TO A DISSOLVED ATOM
```

```
CELL 0
0   Si   1   2.505   0   0
1   Si   1   -1.2525   2.16939   3.64667
2   Si   1   1.2525   2.16939   1.82333
CELL 1
9   Si   1   2.505   -9.79754e-013   5.47
10   Si   1   -1.2525   2.16939   9.11667
11   Si   1   1.2525   2.16939   7.29333
12   O   1   1.59543   0.92112   6.38167
CELL (int)cell_id
(int)atom_id    (text)element   (int)type_atom   (double)position_x  (double)position_y  (double)pos
ition_z
(int)atom_id    (text)element   (int)type_atom   (double)position_x  (double)position_y  (double)pos
ition_z
(int)atom_id    (text)element   (int)type_atom   (double)position_x  (double)position_y  (double)pos
ition_z
(int)atom_id    (text)element   (int)type_atom   (double)position_x  (double)position_y  (double)pos
ition_z
...   ...   ...   ...   ...   ...
...   ...   ...   ...   ...   ...
CELL (int)cell_id
(int)atom_id    (text)element   (int)type_atom   (double)position_x  (double)position_y  (double)pos
ition_z
(int)atom_id    (text)element   (int)type_atom   (double)position_x  (double)position_y  (double)pos
ition_z
...   ...   ...   ...   ...   ...
...   ...   ...   ...   ...   ...
NEIGHBORS
0   2   3.09832   407   3.09832   405   5.01   45   5.01   1   5.66774   856   5.66774   406   4.424
16   451   4.42416
1   2   3.09832   4052   3.09832   54   3.09832   4059   3.09832   4051   5.01   451   5.01   496
5.01   4456   5.01   0   5.66774
2   0   3.09832   1   3.09832   45   3.09832   451   3.09832   407   5.01   47   5.01   4052   5.01
  452   5.01   4059   5.66774
(int)atom_id  (int)neighbour_id  (double)distance_atom_neighbour  (int)neighbour_id  (double)distanc
e_atom_neighbour  ...   ...
(int)atom_id  (int)neighbour_id  (double)distance_atom_neighbour  (int)neighbour_id  (double)distanc
e_atom_neighbour  ...   ...
(int)atom_id  (int)neighbour_id  (double)distance_atom_neighbour  (int)neighbour_id  (double)distanc
e_atom_neighbour  ...   ...
...   ...   ...   ...   ...   ...   ...
...   ...   ...   ...   ...   ...   ...
...   ...   ...   ...   ...   ...   ...
AFFECTED
0   3   411
1   7   8   4054   4055
2   3   5   6   8
(int)atom_id    (int)affected_id    (int)affected_id    ...   ...
(int)atom_id    (int)affected_id    (int)affected_id    ...   ...
...   ...   ...   ...   ...   ...
...   ...   ...   ...   ...   ...
LINKED
0  L  0  L  0  L  1   407  L  1   2  L  1   2  L  1   407  L  1   407  L  1   2
1  L  0  L  0  L  0  L  0  L  1   4052  L  1   2  L  1   54  L  1   4059  L  1   2  L  1   4052  L
1   54
2  L  0  L  0  L  0  L  0  L  1   0  L  1   45  L  1   1  L  1   451  L  1   1  L  1   451  L  1   4
51  L  1   1
(int)atom_id  L (int)type_link  (int)linked_id  (int)linked_id   ...    L (int)type_link  (int)linke
d_id  (int)linked_id   ...
(int)atom_id  L (int)type_link  (int)linked_id  (int)linked_id   ...    L (int)type_link  (int)linke
d_id  (int)linked_id   ...
...  ... ...  ...  ...   ...    ... ...  ...  ...   ...
...  ... ...  ...  ...   ...    ... ...  ...  ...   ...
```

# 4.-Questions-and-anwers.md

1- **Is KIMERA useful to study precipitation?**

No, it is not. Altough KIMERA takes into account the precipition of the ions back into the mineral, precipitation is not explecitly

considered.

2- **I want to study the dissolution of an specfic plane of the mineral using a slab model, how can I do it?**

You have 'simply' to change the orientation of your mineral unit cell. VESTA program allows to do this transformation. In the next link you have an example. Note that you may have to change the unit cell parameters in your input file.

3- **Is there any way to define a dislocation with higher detail rather that the proposed one in the Topography section?**

Instead of using the dislocations commands of the Topography section, you can change the type of the atoms where you want to define the dislocation, using one of these commands of the Particle type modifiers section:

```
CHANGE_XY_DISLOCATION_TYPE                  (double)x    (double)y    (double)radius  (text)final_
atom_type
                                             (optional line) FROM_Z_TO_Z          (double)bot_z
     (double)top_z
                                             (optional line) ANGLE_XZ_ANGLE_YZ    (double)angle_x
z    (double)angle_yz

CHANGE_XZ_DISLOCATION_TYPE                  (double)x    (double)z    (double)radius   (text)final
_atom_type
                                             (optional line) FROM_Y_TO_Y          (double)bot_y
     (double)top_y
                                             (optional line) ANGLE_XY_ANGLE_ZY    (double)angle_x
z    (double)angle_yz

CHANGE_YZ_DISLOCATION_TYPE                  (double)y    (double)z    (double)radius   (text)final
_atom_type
                                            (optional line) FROM_X_TO_X          (double)bot_x
    (double)top_x
                                            (optional line) ANGLE_YX_ANGLE_ZX    (double)angle_yx
    (double)angle_zx
```

And then define the desired contribution to the energy barrier of that type of atom in the events definition.

# Home.md

Welcome to the KIMERA wiki!

KIMERA is a scientific tool for the study of mineral dissolution. It implements a reversible Kinetic Monte Carlo (KMC) method to study the time evolution of a dissolving system, obtaining the dissolution rate and information about the atomic scale dissolution mechanisms. KIMERA is an on-lattice) KMC program that allows to define the dissolution process in multiple ways, uses a wide diversity of event types to mimic the dissolution reactions, and defines the mineral structure in great detail, including topographic defects, dislocations, and point defects. Therefore, KIMERA ensures to perform numerous studies with great versatility. In addition, it offers a good performance thanks to its parallelization and efficient algorithms within the KMC method. In this repository, we present the code features and show some examples of its capabilities. KIMERA is controllable via user commands, it is written in object-oriented C++, and it is distributed as open-source software.

In this wiki you will find instructions guiding you through the installation of KIMERA, and the creation of input files to run your simulations.