| 1. ETUDIANT : | Felipe POBLADOR OZONAS |
| 2. FILIRE : | Master MaNuEn, Phelma |
| 3. ANNEE UNIVERSITAIRE : | 2011/2012 |
| 4. TITRE DU STAGE : | Phase Field method for studying nuclear fuel microstructure changes during irradiation |
| 5. LABORATOIRE : | CEA Cadarache |
| 6. MAITRE DE STAGE : | Laurence NOIROT |
| 7. COORDONNEES POSTALES ET MAIL : | laurence.noirot@cea.fr  
CEA/Cadarache  
Saint-Paul-Lez-Durance  
Bt. 151, p 45, RdC |
Phase Field method for studying nuclear fuel microstructure changes during irradiation

Felipe Poblador Ozonas
Supervisor : Laurence NOIROT
Tutor: Marc FIVEL

5/04/2012 - 14/09/2012
To my grandparents.

Acknowledgements

I want to thank to my supervisor, Laurence NOIROT, her help to solve the problems I had to face during these months.

I want to thank to Nadir BOUHAMOU the code he provided me when I arrived to Cadarache, for visualize the results and convert the numbers into images, without this code that report would not exist.

I want to thank to all the people in the laboratory his welcome, especially to Renaud MASSON, the chief of the laboratory.

I want to thank to Rgine BOUSQUET her help in administration tasks.
Contents

0 Presentation of the center 4

1 Introduction 6

2 Phase Field Model 8
   2.1 Free energy of the system ......................... 8
   2.2 Kinetic Equations .............................. 10
   2.3 Non-dimensionalization .......................... 11
   2.4 Parameters of the Model ......................... 12
      2.4.1 Presentation of the physical parameters ...... 12
      2.4.2 Non-dimensionalized parameters ............... 12
      2.4.3 Values of the non-dimension parameters taken in the calculations .................. 13

3 Numerical Scheme 14
   3.1 Time and Space Discretization .................... 14
   3.2 Boundary Conditions ............................. 15
   3.3 Non-dimension Parameters and Input File .......... 15
   3.4 Initial Conditions ............................... 18
   3.5 Short User Manual ............................... 20
      3.5.1 How to Compile ............................... 21
      3.5.2 How to Execute ............................... 21
      3.5.3 How to Visualize and Post-Process ............ 21

4 Implementation 23
   4.1 Applications and Environment .................... 23
   4.2 Developement .................................. 23
   4.3 Processing for visualization ...................... 26
   4.4 Text outputs .................................. 27
Chapter 0

Presentation of the center

Cadarache is one of the ten research centers of the Atomic and Alternative Energy Commission (CEA). This is one of the most important centers of research and technological development for energy in Europe. Established on the commune of Saint-Paul-Lez-Durance (Bouches-du-Rhone), the CEA/Cadarache center, created in October 14th, 1959, is located about forty kilometers north of Aix-en-Provence at the borders of three other departments (Alpes de Haute-Provence, Var and Vaucluse).

The activities of the CEA/Cadarache center are gathered on several platforms for research and development (R&D) technologies, essentially for nuclear energy (fission and fusion) but also for new energy technologies and studies on ecophysiology and plant microbiology.

In support of these R&D, the Cadarache center has a platform that brings services necessary to:

- The management of nuclear materials, waste and discharges from nuclear installations and the general means to monitor facilities and environmental safety.
- The operation of research facilities (water supply, water treatment systems, electricity).

Several collaborations have been developed with regional and national (EDF, AREVA) companies, the three universities of Aix-Marseille and research institutes [French Institute for Research and Exploitation of the Sea (Ifremer)], the center of research and education in the geosciences environment [Cerege] as part of scientific collaborations, education and the creation of mixed laboratories.
About 90 PhD students are welcomed by the research teams of Cadarache within the framework of agreements with universities. There are also many exchanges and cooperation within the framework of European research programs and international nuclear energy (fission and fusion). Between 150 and 200 researchers from twenty different nations work each year on the Cadarache center as part of European and international scientific collaborations.

The LSC Laboratory

I’ve been working under the supervision of Laurence NOIROT in the ‘Laboratoire de Simulation du Comportement des combustibles’ (LSC) directed by Renaud MASSON. The LSC is in charge of the code development and simulation platforms of the nuclear fuels. The LSC depends on the ‘Service d’Etudes et de Simulation du Comportement des combustibles’ (SESC) which is integrated in the ‘Département d’Etudes des Combustibles’ (DEC), in Cadarache. All of them under the ‘Direction de l’Energie Nucléaire’ (DEN).
Chapter 1

Introduction

The starting point of the present work was a scientific article written by Srujam ROKKAM [9] which was dedicated to studying the evolution of voids in metals under irradiation, using the phase field model.

The phase field method is a technique for simulating microstructural evolution and it is based on expressing the total free energy functional of the heterogeneous material in terms of the free energy of its constituent phases and interfaces. The microstructure considered in [9] is an homogeneous matrix with a given concentration of vacancies and a second phase called void which is a phase with a concentration of vacancies equal to one. Following [9], the evolution of this system, where the shape of the two phases can change, is represented by two functions that are continuous in space and time: those are the phase field variables. The first one is the vacancy concentration and the second one, $\eta$, named ”order parameter”, is a field which equals one in a void and zero in the solid. The phase field variables have nearly constant values in the matrix and void phases. But the interface is defined by a narrow region where the phase field variables gradually vary between the values of matrix and void phases. This receives the name of diffuse interface (see Figure 1.1). The evolution of the phase field variables gives the position and shape of the interfaces. The main advantage of the phase field method is that, thanks to the diffuse interface description, the problem can be solved on a very simple and fixed mesh, which does not depend on the position of the interfaces during microstructural evolution.

The temporal evolution of the phase field variables is described by a set of partial differential equations, which are solved numerically. The equations
are derived based on general thermodynamic and kinetic principles and contain a number of phenomenological parameters related to the physical properties of the material. These parameters are determined based on experimental and theoretical information. Different thermodynamic driving forces for microstructure evolution, such as chemical bulk free energy, chemical interfacial energy and elastic strain energy, and different transport processes, such as heat and mass diffusion, can be considered at the same time. In this work elastic strain energy is not considered and the temperature is set uniform.

The paper [9] gives the equations needed in the phase field model. The work done during the internship was to implement this model in a new code, from scratch. Of course, the first step after implementation is to compare the simulations done with this new code with those given in the paper [9] in order to validate the model. The evolution of a void microstructure in a system under the condition of dynamic vacancy supersaturation was the main objective of this study, as well as the process of void formation and growth due to irradiation. In the article [9], the method was applied to pure metal. We have the objective to apply it to $UO_2$ nuclear fuel under irradiation.

This study presents the steps followed to develop the code, explaining the problems faced and the procedure used to solve them. Some of the results obtained for comparison with the article are presented. Then a chapter is dedicated to the adaptation to the nuclear fuel.
Chapter 2

Phase Field Model

2.1 Free energy of the system

The system considered in this study is a heterogeneous system, with varying concentration in the matrix and interface between matrix and void phases. The voids are the result of the condensation of vacancies from a vacancy supersaturated solution of vacancies and lattice atoms. The term supersaturation ($S_v$) refers to the ratio of vacancy concentration in the matrix above its thermal equilibrium value, $S_v = \frac{c_v}{c_0}$.

The vacancy distribution is described by the vacancy concentration field, denoted by $c_v(x, t)$. Besides this concentration field a non-conserved phase field variable or order parameter $\eta(x, t)$ is also used. The order parameter takes a constant value within the individual phases and varies smoothly across interfaces, it varies continuously, from $\eta = 0$ in the matrix phase, to $\eta = 1$ in the void phase, over a narrow diffuse interface between the two phases.

To develop the phase field model, the starting point is the free energy of the system. Being this system a crystal with two phases, a matrix phase with defects (vacancies) and a void phase.

The expression for the free energy chosen in the article [9] is, after neglecting the elastic mechanical energy:

$$\Psi[c_v, \eta] = N \int_\Omega [h(\eta)\Psi^m(c_v) + w(c_v, \eta) + k_v |\nabla c_v|^2 + k_\eta |\nabla \eta|^2]d\Omega \quad (2.1)$$
Being \( N \) the number of lattice sites per unit volume. The first two terms under the integral sign of equation (2.1), multiplied by \( N \), are the system bulk free energy per unit volume.

Below the integral sign the defect free-energy term \( \Psi^m(c_v) \) is written in the form:

\[
\Psi^m(c_v) = E^f_v c_v + k_B T \left[ c_v \ln(c_v) + (1 - c_v) \ln(1 - c_v) \right] \tag{2.2}
\]

where \( E^f_v \) is the vacancy formation energy, \( k_B \) the Boltzmann constant and \( T \) the absolute temperature. Note that \( c_v \) is a concentration in fraction of sites, \( N c_v \) is the concentration in vacancies/m\(^3\). The function \( h(\eta) \) has the expression: \( h(\eta) = (\eta - 1)^2 (\eta + 1)^2 \).

The term \( w(c_v, \eta) \) is a Landau-type, responsible for the bi-stability in the system, it is expressed as follow:

\[
w(c_v, \eta) = -A(c_v - c^0_v)^2 \eta(\eta + 2)(\eta - 1)^2 + B(c_v - 1)^2 \eta^2 \tag{2.3}
\]

where \( c^0_v \) is the thermal equilibrium vacancy concentration in the solid, given by the expression \( c^0_v \approx \exp(-E^f_v/k_B T) \), and \( A \) and \( B \) are constant prefactors. The first two terms in the integral of equation (2.1) represent the free-energy density of the homogeneous system. The two energy terms are designed to define two stable wells: one at \( c_v = c^0_v \) and \( \eta = 0 \) corresponding to the matrix phase with vacancy concentration equal to the thermal concentration, and another at \( c_v = 1 \) and \( \eta = 1 \) corresponding to the void phase. This is due to the fact that the local energy density \( \Psi^m(c_v) \) has a minimum at \( c_v = c^0_v \). The Landau term \( w(c_v, \eta) \) is chosen so as to retain the energy well defined by \( \Psi^m(c_v) \) and to contribute a stable well at \( c_v = 1 \) and \( \eta = 1 \), corresponding to a pure vacancy phase (voids). The shape function \( h(\eta) \) ensures that the contribution of the energy density \( \Psi^m(c_v) \) is nullified when \( c_v = 1 \) and \( \eta = 1 \). The gradient energy coefficients \( k_\eta \) and \( k_v \) of equation (2.1) characterize the energy penalties corresponding to the inhomogeneities in phase field and vacancy field, respectively. The corresponding energy terms are reminiscent of the gradient terms in the spinodal decomposition theory of Cahn and Hilliard [2] and the theory of antiphase boundary of Allen and Cahn [1]. The gradient energy terms in equation (2.1) account for the field inhomogeneity across the diffuse void-matrix phase.
2.2 Kinetic Equations

From the free-energy of the system, the kinetic equations can be obtained. Following the phase field approach, those are derived:

$$\frac{\partial c_v}{\partial t}(x, t) = \nabla \cdot \frac{M_v}{N} \nabla \frac{1}{N} \delta \Psi + \xi(x, t) + P_v(x, t)$$  \hspace{1cm} (2.4)

$$\frac{\partial \eta}{\partial t}(x, t) = -L \frac{\delta \Psi}{\delta \eta} + \zeta(x, t)$$  \hspace{1cm} (2.5)

Where the terms $\xi(x, t)$ and $\zeta(x, t)$ are the noise terms and $P_v(x, t)$ is the term related with the vacancy generation due to the irradiation. The noise terms are used in the article [9] to model the nucleation of voids.

The kinetic equations for the space and time evolution of the phase field variables $c_v(x, t)$ and $\eta(x, t)$ are derived:

$$\frac{\partial c_v}{\partial t}(x, t) = \nabla \cdot \frac{M_v}{N} \nabla \left[ h(\eta) \frac{\partial \Psi^m(c_v, \eta)}{\partial c_v} + \frac{\partial w(c_v, \eta)}{\partial c_v} - 2k_v \nabla^2 c_v \right] + \xi(x, t) + P_v(x, t)$$  \hspace{1cm} (2.6)

$$\frac{\partial \eta}{\partial t}(x, t) = -LN \left[ \frac{\partial h(\eta)}{\partial \eta} \cdot \Psi^m + \frac{\partial w(c_v, \eta)}{\partial \eta} - 2k_\eta \nabla^2 \eta \right] + \zeta(x, t)$$  \hspace{1cm} (2.7)

Where the terms $| \nabla c_v |^2$ and $| \nabla \eta |^2$ of equation (2.1) give the terms $\nabla^2 c_v$ and $\nabla^2 \eta$ respectively, see Appendix C.

Developing the partial derivatives, the equation (2.6) remains as follows:

$$\frac{\partial c_v}{\partial t}(x, t) = \nabla \cdot \frac{M_v}{N} \nabla \left[ \frac{(\eta - 1)^2(\eta + 1)^2}{2} \left[ E_v^f + k_B T[\ln(c_v) - \ln(1 - c_v)] \right] + 2A(c_v - c_v^0)(\eta^4 - 3\eta^2 + 2\eta) + 2B(c_v - 1)\eta^2 - 2k_v \nabla^2 c_v \right] + \xi(x, t) + P_v(x, t)$$  \hspace{1cm} (2.8)

For the order parameter, from equation (2.7):

$$\frac{\partial \eta}{\partial t}(x, t) = -LN \left[ 4\eta(\eta - 1)(\eta + 1) \right] \left[ E_v^f c_v + k_B T[c_v \ln(c_v) + (1 - c_v) \ln(1 - c_v)] \right] - A(c_v - c_v^0)^2(4\eta^3 - 6\eta + 2) + 2B(c_v - 1)^2\eta - 2k_\eta \Delta \eta \right] + \zeta(x, t)$$  \hspace{1cm} (2.9)

For the model used in this study, the mobility of the vacancies ($M_v$) has been considered as constant in space and time in all the simulations.
2.3 Non-dimensionalization

The previous expressions (2.8) and (2.9) can be reduced using a length scale
\( l = \sqrt{\kappa_\eta / k_B T} \), time scale of \( \tau = l^2 / D_v \) (where \( D_v \) is the diffusivity of the va-
cancies; the value of \( M_v \) is obtained by the approximation \( M_v = N D_v / k_B T \)
making the mobility isotropic and uniform over the entire do-
main), and energy scale \( k_B T \).

The normalization of spatial distance by \( l \) and time by \( \tau \) leads to the
following changes from equation (2.8):

\[
\frac{\partial c_v}{\partial \tilde{t}}(x, t) = \nabla \cdot \tilde{M}_v \nabla \left[ (\eta - 1)^2 (\eta + 1)^2 \tilde{E}_v^f + \ln(c_v) - \ln(1 - c_v) \right] \\
-2\tilde{A}(c_v - c_0^v)(\eta^4 - 3\eta^2 + 2\eta) + 2\tilde{B}(c_v - 1)^2 \eta^2 - 2k_v \nabla^2 c_v \\
+ \tilde{\xi}(x, t) + \tilde{P}_v(x, t) \tag{2.10}
\]

For the order parameter, from equation (2.9):

\[
\frac{\partial \eta}{\partial \tilde{t}}(x, t) = -\tilde{L}[4\eta(\eta - 1)(\eta + 1) \tilde{E}_v^f c_v + c_v \ln(c_v) + (1 - c_v)\ln(1 - c_v)] \\
-\tilde{A}(c_v - c_0^v)(4\eta^3 - 6\eta + 2) + 2\tilde{B}(c_v - 1)^2 \eta - 2k_v \nabla^2 \eta + \tilde{\zeta}(x, t) \tag{2.11}
\]

These equations are simplified by the terms \( \tilde{\Psi}_v \) and \( \tilde{\Psi}_\eta \) as:

\[
\frac{\partial c_v}{\partial \tilde{t}}(x, t) = \nabla \cdot \tilde{M}_v \nabla \tilde{\Psi}_v + \tilde{\xi}(x, t) + \tilde{P}_v(x, t) \tag{2.12}
\]

\[
\frac{\partial \eta}{\partial \tilde{t}}(x, t) = -\tilde{L} \tilde{\Psi}_\eta + \tilde{\zeta}(x, t) \tag{2.13}
\]

With:

\[
\tilde{\Psi}_v = (\eta - 1)^2 (\eta + 1)^2 \tilde{E}_v^f + \ln(c_v) - \ln(1 - c_v) \\
-2\tilde{A}(c_v - c_0^v)(\eta^4 - 3\eta^2 + 2\eta) + 2\tilde{B}(c_v - 1)^2 \eta^2 - 2k_v \nabla^2 c_v \tag{2.14}
\]

and

\[
\tilde{\Psi}_\eta = 4\eta(\eta - 1)(\eta + 1) \tilde{E}_v^f c_v + c_v \ln(c_v) + (1 - c_v)\ln(1 - c_v) \\
-\tilde{A}(c_v - c_0^v)(4\eta^3 - 6\eta + 2) + 2\tilde{B}(c_v - 1)^2 \eta - 2k_\eta \nabla^2 \eta \tag{2.15}
\]

The process of non-dimensionalization is detailed in the Appendix B.
2.4 Parameters of the Model

2.4.1 Presentation of the physical parameters

The different parameters that are needed to compute the phase field model are:

\( N \) : Lattice sites per unit volume \([\text{sites} \text{m}^{-3}]\).

\( D_v \) : Diffusivity of the vacancies (used to obtain \( M_v \) and time and length scales) \([\text{m}^2 \text{s}^{-1}]\).

\( E_f^v \) : Vacancy formation energy \([J]\).

\( k_B \) : Boltzman constant \([1.3806488 \cdot 10^{-23} \text{JK}^{-1}]\).

\( T \) : Absolute temperature \([\text{K}]\).

\( c_0^v \) : Thermal equilibrium vacancy concentration in the solid, given by \( E_f^I \), \( k_B \), and \( T \); \([c_0^v \approx \exp(-E_f^I/k_B T) = \text{fraction of sites(\%)}]\).

\( M_v \) : Mobility of the vacancies \( (\frac{M_v}{N} = D_v/k_B T \left[\frac{m^2}{\text{s}}\right])\).

\( k_v \) and \( k_\eta \) : Energy penalties corresponding to the inhomogeneities in the vacancy concentration and phase field, respectively \([Jm^2]\).

\( A \) and \( B \) : Parameters of the Landau-type \( w(c_v, \eta) \), with the same unit as \( E_f^I \) \([J]\).

\( L \) : Mobility of the phase field \( \eta(\vec{x}, t) \left[\frac{m^3}{\text{s}}\right]\).

\( \Delta x \) : Length step for the mesh \([m]\).

2.4.2 Non-dimensionalized parameters

Using \( \tau = l^2/D_v \) and \( l = \sqrt{k_\eta/k_B T} \), these are the relations between the non-dimension and the dimension parameters:

- \( \tilde{t} = t/\tau \)
- \( \tilde{\Delta}x = \Delta x/l \)
- \( \tilde{\nabla}^2 = l^2 \nabla^2 \)
• $k_v = \frac{k_v}{k_\eta}$
• $\tilde{k}_\eta = 1$
• $\tilde{M}_v = 1$
• $\tilde{L} = L N \tau k_B T$

2.4.3 Values of the non-dimension parameters taken in the calculations

The values of $\Delta \tilde{t}$, $\tilde{t}_{\text{final}}, \tilde{\Delta} x$, $\tilde{k}_v$, $\tilde{L}$ taken in the computations are presented in 3.3.

The values used by the authors of the article [9] were:

• $\Delta \tilde{t} = 5 \cdot 10^{-5}$
• $\tilde{\Delta} x = 1$ for 128 nodes and $\tilde{\Delta} x = 0.5$ for 256 nodes.
• $\tilde{k}_v = 0.5$
• $\tilde{k}_\eta = 1$ by definition of the non-dimensionalization.
• $\tilde{M}_v = 1$ by definition of the non-dimensionalization.
• $\tilde{L} = 1$
• $\tilde{A} = 9.09$
• $\tilde{B} = 9.09$
• $\tilde{E}_v^f = 9.09$

Note that, for the 128 nodes mesh, we have $\Delta x = l$. 
Chapter 3

Numerical Scheme

3.1 Time and Space Discretization

The kinetic equations are solved by using a 5-point central finite difference scheme in space and a forward Euler marching scheme in time. The discretization leads to:

\[
\begin{align*}
\frac{c_v|^{n+1}_{i,j} - c_v|^{n}_{i,j}}{\Delta t} + \frac{\Delta \bar{t}}{(\Delta \bar{x})^2} M_v (\tilde{\Psi}_v|^{n}_{i+1,j} + \tilde{\Psi}_v|^{n}_{i,j+1} + \tilde{\Psi}_v|^{n}_{i-1,j} + \tilde{\Psi}_v|^{n}_{i,j-1} - 4\tilde{\Psi}_v|^{n}_{i,j}) \\
+ \Delta \bar{t} \cdot \tilde{P}_v|^{n}_{i,j} &= 0 \quad (3.1) \\
\eta|^{n+1}_{i,j} - \eta|^{n}_{i,j} - \Delta \bar{t} L \tilde{\Psi}_\eta|^{n}_{i,j} + \Delta \bar{t} \tilde{\zeta}|^{n}_{i,j} &= 0 \quad (3.2)
\end{align*}
\]

Where the \( \tilde{\Psi} \) terms are:

\[
\begin{align*}
\tilde{\Psi}_v|^{n}_{i,j} &= (\eta - 1)^2(\eta + 1)^2[E_v^f + ln(c_v) - ln(1 - c_v)] \\
&- 2\tilde{A}(c_v - \bar{c}_v)(\eta^4 - 3\eta^2 + 2\eta) + 2\tilde{B}(c_v - 1)\eta^2 - 2\tilde{k}_v \tilde{\nabla}^2 c_v|^{n}_{i,j} \quad (3.3) \\
\tilde{\Psi}_\eta|^{n}_{i,j} &= 4\eta(\eta - 1)(\eta + 1)[E_v^f c_v + c_v ln(c_v) + (1 - c_v)ln(1 - c_v)] \\
&- \tilde{A}(c_v - \bar{c}_v)(4\eta^3 - 6\eta + 2) + 2\tilde{B}(c_v - 1)^2 \eta - 2\tilde{k}_\eta \tilde{\nabla}^2 \eta|^{n}_{i,j} \quad (3.4)
\end{align*}
\]

With \( c_v = c_v|^{n}_{i,j} \) and \( \eta = \eta|^{n}_{i,j} \), where \( n \) stands for the time step number and \( i, j \) are the indexes of the point in the mesh.

The Laplacian terms \( \tilde{\nabla}^2 c_v \) and \( \tilde{\nabla}^2 \eta \) are evaluated using a 5-point central difference scheme or a 9-point central difference scheme (see Appendix A).
3.2 Boundary Conditions

The boundary conditions defined in the code are periodic, as the mesh is a regular square with a defined number of nodes in each direction, this periodicity sets the volume object of study surrounded by the same volume. The fields are defined in a matrix with row and column as coordinates, starting with zero as the first coordinate and going until the number of nodes less one, so the coordinates corresponding to the neighbors of the first and last coordinate are ‘−1’ and ‘number of nodes’, which are not valid coordinates for the field matrix. To solve this, the code defines the neighbors of the boundary lines and columns as:

- The row or column corresponding to coordinate ‘−1’ corresponds to the row or column at the last coordinate of the matrix; as it is to say the coordinate ‘number of nodes less one’.

- The row or column corresponding to coordinate ‘number of nodes’ corresponds to the row or column at the first coordinate of the matrix, as it is to say, the coordinate ‘0’.

3.3 Non-dimension Parameters and Input File

The input of the program is given in a ‘.xml’ file, where the user can give the values for the parameters of the model and can define the values for initial concentration inside the cavity ($c_v = 1$) and outside the cavity, the size of the voids, the number of iterations, the number of iterations between output files and the number of nodes.

Those values are read by the code from the xml file and used in the simulation. An example of input file is given in Appendix D. Following the scheme of the xml file, the parameters are explained:

**nND nodes**=”256”: Number of nodes per direction for the mesh.

**DATA ITER** Input for the iteration values.

- **nbIterations** = ”100000”: Number of iterations for the simulation.
- **interval** = ”10000”: Interval between iterations to generate an output.

**NOISE** Input for the noise and source term.
• **noise** = "0": If the value is 0 no noise/source terms will be applied, opposite if it is 1.

• **NoiseStop** = "30000": Sets the number of iterations the noise/source term will affect the simulation.

**TEXT** texts="0": If the value is 1 the simulation will generate ‘.txt’ outputs, containing concentration profiles and phase field profiles, see Section 4.4.

**VISUAL** vtk="1": Set to 1, the simulation generates the ‘.vtk’ and ‘.med’ files (for the visualization).

**SCHEME** scheme="1": This option sets the time scheme that will be used in the simulation ("1" is Euler forward marching scheme, and "2" is Runge-Kutta order 4 scheme).

**CORRECT** Input for the correction of fluxes.

• **correction** = "2": The corrections are explained in sections 5.2 (option "1") and 5.4 (option "2") .

• **FmaxFactor** = "1.0": This factor is used in correction "2", the value $F_{\text{max}} \text{total}$ is proportional to $(c_v^0 - c_v)/F_{\text{maxFactor}}$. If $F_{\text{maxFactor}} = "1.0"$, the concentration at the next iteration $c_v^{n+1} = c_v^0$.

**DISCRETIZATION** The discretization of the inner Laplacians is defined here.

• **discr** = "1": There are three options that are explained in Appendix A. Option "1" is equation A.3, option "2" is equation A.4, option "3" is equation A.2.

• **alfa** = "0.3": This is the value of the alfa paramater in case of using discr = "3".

**DATA1** Values for the mesh and vacancy information.

• **deltaT** = "5e−5": Time step used in the simulation.

• **deltaX** = "0.5": Distance between nodes in the mesh (set = 1 if nND is 128 to calculate the same box).

• **epscv** = "1e−10": Value used when computing the logarithms (see Chapter 5 *Problems while programming*).
• $Sv = "300"$: Supersaturation of vacancies at the beginning of the simulation $\frac{C_v^0}{C_v}$.
• $P_{casc} = "0.25"$: Given value to model the cascade source $\tilde{P}_v = P_{casc} \ast rand(\tilde{x}_{rand}, \tilde{t})$.

GENERAL DATA Values for the parameters of the phase field model.

• $M_{vd} = "1"$: Value for $\tilde{M}_v$.
• $E_{fvd} = "9.09"$: Value for $\tilde{E}_f$.
• $Ad = "9.09"$: Value for $\tilde{A}$.
• $Bd = "9.09"$: Value for $\tilde{B}$
• $C0v = "1.1277534880926000e -04"$: Value for $c_v^0$.
• $kvd = "0.5"$: Value for $\tilde{k}_v$.
• $knd = "1"$: Value for $\tilde{k}_\eta$.
• $Ld = "1"$: Value for $\tilde{L}$.

CASE case = "11": Define the Initial Conditions for the simulation (see section 3.4 Initial Conditions).

VALUES1 Values used if the case is 11, 12 or 13 (Center Void).

• $valC = "Sv \ast C0v"$: Initial vacancy concentration $c_v^0_{i,j}$ outside the void.
• $valOP = "0"$: Value for $\eta^0_{i,j}$ outside the void.
• $valin = "1"$: Value for the $c_v^0_{i,j}$ and $\eta^0_{i,j}$ inside the void.
• $rad = "11.48"$: Non-dimensioned radius of the void (Real radius value is rad $\ast l$).

VALUES2 Values used if the case is 21, 22 or 23 (Constant Field).

• $initConcentration = "Sv \ast C0v"$: Vacancy concentration $c_v^0_{i,j}$.
• $initPhase = "0.0"$: Value for $\eta^0_{i,j}$.

VALUES3 Values used if the case is 3** (Void Band).

• $valBand = "1"$: Value for the $c_v^0_{i,j}$ and $\eta^0_{i,j}$ inside the band.
• $valOut = "Sv \ast C0v"$: Vacancy concentration $c_v^0_{i,j}$ outside the band.
• \textit{width} = "10": Non dimensioned width of the band (Real width value is \textit{width} \times \textit{l}).

\textbf{VALUES4} Values used if the case is 4* (Two Voids).

• \textit{rowCenter1} = "127.5": Vertical real position in the mesh for the center point of void one is 127.5 \times \Delta x.
• \textit{colCenter1} = "79": Horizontal real position in the mesh for the center point of void one is 79 \times \Delta x.
• \textit{rowCenter2} = "127.5": Vertical real position in the mesh for the center point of void two is 127.5 \times \Delta x.
• \textit{colCenter2} = "175": Horizontal real position in the mesh for the center point of void two is 79 \times \Delta x.
• \textit{radius1} = "8": Non-dimensioned radius of the first void (Real radius value is \textit{radius1} \times \textit{l}).
• \textit{radius2} = "4": Non-dimensioned radius of the second void (Real radius value is \textit{radius2} \times \textit{l}).
• \textit{valin} = "1": Value for the \( c_v |_{i,j}^0 \) and \( \eta |_{i,j}^0 \) inside the voids.
• \textit{valout} = "Sv \times C0v": Value for the \( c_v |_{i,j}^0 \) outside the voids.

\textbf{RANDOM} Value for the random functions.

• \textit{muNoise} = "0.0": Mean value for the random number used to compute the noise term.
• \textit{sigmaNoise} = "0.0": Standard Deviation of \textit{muNoise} for the random number used to compute the noise term.

\textbf{3.4 Initial Conditions}

The implemented code has the option to create different initial condition fields, which can be chosen in the input file. The options are named as CASE followed by a number for each initial condition and another number for the simulation options (noise term, cascade term ...). The complete list of options is the following:

\textbf{CASE1*}: a center void you can define

• case 11: only a random term
• case 12: only a source term at one point each iteration. $P[\text{row}][\text{col}] = \text{rand} * P_{\text{casc}}$ with $\text{rand} \in [0, 1]$
• case 13: both terms

**CASE2***: a constant field with defined values for $C_v$ and $\eta$
• case 21: only a random term
• case 22: only a source term at one point each iteration. $P[\text{row}][\text{col}] = \text{rand} * P_{\text{casc}}$ with $\text{rand} \in [0, 1]$
• case 23: both terms

**CASE3***: a field with a Band (parameters defined by user)
• case 31*: a horizontal Band (used for validate the model: only allows nND=256)
  1. case 311: only a random term
  2. case 312: only a source term at one point each iteration. $P[\text{row}][\text{col}] = \text{rand} * P_{\text{casc}}$ with $\text{rand} \in [0, 1]$
  3. case 313: both terms
• case 32*: a vertical Band
  1. case 321: only a random term
  2. case 322: only a source term at one point each iteration. $P[\text{row}][\text{col}] = \text{rand} * P_{\text{casc}}$ with $\text{rand} \in [0, 1]$
  3. case 323: both terms
• case 33*: a diagonal Band
  1. case 331: only a random term
  2. case 332: only a source term at one point each iteration. $P[\text{row}][\text{col}] = \text{rand} * P_{\text{casc}}$ with $\text{rand} \in [0, 1]$
  3. case 333: both terms

**CASE4***: a field with two voids you can define
• case 41: only a random term
• case 42: only a source term at one point each iteration. $P[\text{row}][\text{col}] = \text{rand} * P_{\text{casc}}$ with $\text{rand} \in [0, 1]$
• case 43: both terms
If the user wants to do a simulation without any noise term, the input parameter NOISE can be set to 0. But a simulation with $P_{\text{case}} = 0$ will give the same result for the CASE*2, the difference is that if the NOISE parameter is 0, no computation will be done for the noise term, so gives a shorter simulation time, by shortening the computer time consumption.

### 3.5 Short User Manual

The content of the folder before any compilation has to be:

- envDev.sh
- Makefile
- MATRIX.cxx
- MATRIX.hxx
- phaseFieldBaseMATRIX.cxx
- phaseFieldBaseMATRIX.hxx
- ModelPHASEFIELDMATRIX.cxx
- ModelPHASEFIELDMATRIX.hxx
- ModelPHASEFIELDserie.cxx
- ModelPHASEFIELDserie.hxx
- mainPHASEFIELDMATRIX.cxx
- mainPHASEFIELDserie.cxx
- ticpp.cpp
- ticpp.h
- ticpprc.h
- tinystr.cpp
- tinystr.h
- tinyxml.cpp
3.5.1 How to Compile

The Makefile gives the option to compile in serial or in parallel by typing:

- make: “make” without any option compiles the parallel code, generating the “testPHASEFIELDparallel” executable
- make serie: compiles in serial, generating the “testPHASEFIELDserie” executable

3.5.2 How to Execute

After compiling, the user has to set the input file (input_ok.xml) with the desired parameters, defining the Initial Conditions, the type of simulation and the output files he/she wants to obtain. The executable reads the input file and use the values saved in the same.

To execute the simulation, the user has to set the environment first by typing “source envDev.sh”, and then the simulation can be launched by typing “./testPHASEFIELDparallel” or “./testPHASEFIELDserie”, depending on the executable he/she wants to use. The program runs for about one hour to do a 100000 iterations simulation with a 256 × 256 mesh, and about 8 hours for one million iterations with the same mesh size. Running in a single processor (serie) or in two processors (parallel), which takes a bit less time.

3.5.3 How to Visualize and Post-Process

Once the simulation has finished, the folder where the user run the simulation will contain many files. Most of them are the `.vtk` files, which are generated if the users sets the viusalization input parameter “vtk=1”. The number of `.vtk` files depends on the number of iterations set in the input (nbIterations=”100000”) and the interval between iterations (interval=”1000”), in this case one obtains 101 files (100000/1000 + 1) the initial conditions and the result at iterations 1000, 2000 ... 100000. In order to visualize the results
with Paraview, the user has to Open the “phaseField_Case**-***.vtk” from the folder he/she has launched the simulation and then click on “Apply”. This opens all the ‘.vtk’ files thus the whole simulation can be seen as a film by clicking on the “play” button in Paraview. The time value showed by Paraview corresponds to the number of iteration divided by the interval that corresponds to the actual image or ‘.vtk’ file. Paraview has an option to save the results as an animation (“Save Animation”) in the “File” menu, generating an ‘.avi’ file.

Most of the ‘.txt’ output files are generated in order to open them with Excel or similar. In order to convert the file to a ‘.xcls’ files, open the text file, set the tabulation with spaces and change the dot for comma. This gives a table with the values of the text file. Other text files give information about the simulation, like “Data.txt” (resume of the input file, start and end time) or “Vacancies.txt” (mean vacancy concentrations and total number of vacancies).
Chapter 4

Implementation

4.1 Applications and Environment

The implementation of the Phase Field Model has been developed in the C++ language and compiling with GNU g++-4.3 compiler. The OS is Debian GNU/Linux 5.0.10. The simulations are visualized with the Paraview software using the output ‘.vtk’ files. The program also generates a ‘.med’ file which can be read by Salome, this software is more powerful but needs a bigger computer than what I had. All this output files are generated thanks to MEDMEM library. The use of the OpenMP was introduced at the last part of the internship, by recomendation of Marc FIVEL, to do faster simulations thanks to the use of Multi Processor; the code is splitting the ”for” parts in the number of processors available in the computer. The user can choose to compile the program in parallel or serie by the comands make or make serie, respectively.

4.2 Developement

The program calculates the vacancy concentration field and the phase field following the equations described above. In order to achieve this objective, the first step was the implementation of the equations with a new Matrix class which can stock the values of the fields and allows easy access to the values. This matrix has as many elements as the number of nodes in the mesh; in practice, we used a 128x128 or a 256x256 mesh. The equations are separated in different parts for simplicity in terms of programming and understanding. The basic level of the program which contains the equations of the model calculates through several functions the values of all the terms
used in the main equations (3.1) and (3.2) to obtain the fields at the next
iteration. These terms are calculated at each point of the field, that is to
say that the values of each term are also stocked in a Matrix-type object.
The program is written by following logical steps in terms of calculating the
values.
The sum of the noise term and the source term, $P$, is calculated in the function called $CalcP$, and saved in a Matrix. $CalcP$ needs a mean value and the standard deviation to generate one random number for the Noise term, given by a Gauss distribution. The source term is given by a random number between zero and one-thousand (generated by the random function of C++) and then divided by 1000 in order to obtain a random number between zero and one, $rand_{0-1}$. The noise term is generated at each point of the field by the random number. The cascade term is generated in only one point of the field, $M_{\text{rand}}$. This point is randomly defined by the C++ random function and has to be in the matrix (not in a void). For that purpose, the value of $\eta$ at this point is considered and only if it is below a certain value (set at 0.5 in our study), the source term is applied at the point. The strength of the cascade term is given by the random number between zero and one described before and a constant value ($P_{\text{cas}}$) given for a single cascade. For a point $M$ in the solid:

$$P(M) = P_{\text{cas}} \times rand_{0-1} \times \delta_{M_{\text{rand}}} + rand(\mu_{\text{Noise}}, \sigma_{\text{Noise}}).$$

Where $\delta_{M_{\text{rand}}}$ is the Kronecker symbol (1 if $M = M_{\text{rand}}$ and 0 if $M \neq M_{\text{rand}}$). This equation contains the source term (first term) and the noise term (second term) as a general description, but the user can choose between applying one of the terms, both of them or none of them. In practice, the noise term was never used in this work because we wanted to control the vacancy balance in our tests.

$CalcPSIvn$ and $CalcPSIOPn$

The terms corresponding to the equations (3.3) and (3.4) are calculated for each point of the field and saved in another Matrix-type object by the functions $CalcPSIvn$ and $CalcPSIOPn$ which give the values of the right hand side of the equations at each point of the field at the current iteration. Inside these functions, the function $CalcLap$ is called to compute the values of the Laplacians $\tilde{\nabla}^2 c_v$ and $\tilde{\nabla}^2 \eta$.

Once the values of the $\tilde{\Psi}_v$ and $\tilde{\Psi}_\eta$ terms are calculated, the function $CalcLap$ is called again to compute the first Laplacian which appears in the equation (2.4) and correspond to the term between parentheses in the second term of equation (3.1). This second call of the function $CalcLap$ will be replaced by another function explained in the next chapter because of the problems due to the discretization of the Laplacian and the values of the vacancy concentration.

$CalcCnp1$ and $CalcOPnp1$
With all these values the program has all the data necessary to compute the values of the vacancy concentration field and the phase field of the next iteration. The functions $CalcCnp1$ and $CalcOPnp1$ calculate the values of the vacancy concentration field and the phase field at each point for the iteration $n + 1$ respectively.

Once this first part was working and giving no more errors with compilation and execution, the next step was to provide a code which transforms these values into a '.vtk' file and allows to visualize and study the results with the numerical values obtained by the program.

4.3 Processing for visualization

As was mentioned before, the code containing the equations of the Phase Field Model and the code developed to represent the values of the model were coupled. For this task, the classes Matrix and double** were used because the code which provides the output files for the visualization reads and write the values of the fields using double** while the result of the field given by the first code gives the results in a Matrix class.

The functions $D2M$ and $M2D$ were defined to convert the data from one class to the other.

A new class called MODELPHASEFIELD was defined. In this class the fields resulting of each iteration are saved in a list for both vacancy concentration and phase fields. In the next iteration the result is read from this list and used to compute the current iteration. In a first aproach all the iterations were added to the list causing a massive use of RAM memory but this was changed and the final choice was to save just the result of the last iteration.

The resulting fields are saved in a '.med' file readable by Salome and a '.vtk' file readable by Paraview, with both, the vacancy concentration and phase fields. In order to avoid consuming CPU time saving the fields in the '.med' file and generating the '.vtk' file, just some iterations are saved for post-process study; the interval between two iterations where the fields are saved is given by the user in the input file.

The option to generate or not the visualization files is included in the input file. Introducing in the VISUAL layer ‘1’ if the user wants to obtain the '.med' and '.vtk' for each iteration defined in the interval, or '0' to obtain the output just for the first iteration and the last iteration.
4.4 Text outputs

The simulation generates several `.txt` files in order to represent the results of the iteration defined in the interval. These files contain different information. These first files contain information given for all the cases:

- **Data.txt** provides the general information of the simulation, it is a printing of the input file and the starting and ending dates.
- **Vacancies.txt** gives information about the total number of vacancies in the volume, the mean vacancy concentration corresponding to void phase and the mean vacancy concentration corresponding to matrix phase.
- **FREE.txt** has the information related with the free energy of the volume. It is divided in three: the free energy associated to the matrix phase, the free energy of the interfaces and the free energy of the void phases.
- **VOIDS.txt** gives the number of voids and the void fraction of the volume.

The following files are generated for certain cases, if the `texts` option is set to ‘1’:

- **CV.txt** is the horizontal vacancy concentration profile at the middle row.
- **OP.txt** is the horizontal order parameter profile at the middle row.
- **Radius.txt** is generated for cases1* and gives the mean radius of the center void.
- **Width.txt** is generated for cases3** and gives the width of the Band.

All these files can be post-processed with Excel by opening as text file and following the pop-up, clicking on limited, tabulated and separated with spaces and changing the decimal sign for point instead of comma in the advanced options.
Chapter 5

Problems while programming

We tested the program in the situation of one pre-existing void surrounded by a super-saturated solid ($S_v = c_v/c_v^0 > 1$).

5.1 Logarithms

The code used is written in C++, so all the formulas used by the model are implemented and joined in a single code. The formulas have some numerical problems near certain values of concentration. These values are zero and one, the limit values. At these values the result of the logarithms is NaN (not a number) for $\ln(1 - c_v)$ when $c_v$ is one, and also for $\ln(c_v)$ when $c_v$ is zero. The evolution of the concentration is given by the previous concentration, so it’s necessary to avoid the logarithms of these values in the program. These problem is extensible for negative values and values above one of the concentration. The first attempt to avoid the singular case was by fixing the concentration with a value just below one and just above zero, so the logarithm could be computed. But this solution was not conserving the balance of vacancies in the volume, and thus was discarded.

The solution takes into account the value of the concentration, the result of the logarithms is computed by two functions $\logc(C)$ (when computing $\ln(1 - c_v)$ and $\ln(c_v)$) and $c\logc(C)$ (when computing $(1 - c_v) * \ln(1 - c_v)$ and $c_v * \ln(c_v)$). The first function returns $\ln(C)$ if $C > 0$ and $\ln(1e - 12)$ if $C \leq 0$. The second function returns $C * \ln(C) = 0$ (with $C = c_v$ or $C = (1 - c_v)$) when $1 - epscv \leq C \leq 1 + epscv$ or when $-epscv \leq C \leq epscv$, as is to say when $C$ is near zero or one; for any other value of $C$, the function returns $C * \logc(C)$ where $\logc(C)$ is the function defined before.
5.2 Concentration out of limit

Another problem was faced while programming, this was to observe values of the concentration out of range, that is to say values above one and values below zero. One of the problems was the value of the logarithms at these points, as mentioned before. But also to observe concentrations above one which is physically impossible. As the previous section, to avoid this problem the first attempt was to set a fix value for the vacancy concentration if this was above one (the value was set to 0.9999) and the same if the values was below zero (the value was set to $1e^{-5}$), not only avoiding the singular cases of the logarithms, but also the problem of a concentration greater than one or below zero. This solution showed a void without moving (nor growth or shrinkage) but the vacancies surrounding it were disappearing, which again produced a situation where the balance of vacancies was not null. So this option was not valid to solve the problem.

Another strategy was found instead of correcting the concentration, we decided to correct the fluxes (this strategy conserves the balance of vacancies). Indeed, the flux of vacancies going from $(i + 1, j)$ to $(i, j)$ is proportional to $\tilde{\Psi}_v |_{i+1,j}^n - \tilde{\Psi}_v |_{i,j}^n$ and the term in parenthesis in equation (3.1) is a sum of fluxes from all the neighbors of point $(i, j)$. We banned the flux of vacancies going to a point which concentration of vacancies was already one (exactly when $c_v > cvlim1 = 1 - epscv$), and also the flux of vacancies going out from a point which concentration was already 0 (exactly $c_v < cvlim0 = epscv$). This correction is done at each point taking into account the value of the concentration and the flux, with a correction of the flux, if necessary.

As can be seen in the Figure 5.1, both problems disappear with the correction.

5.3 Shape of the voids

This problem appeared while running simulations with an excess of vacancies in the matrix ($Sv > 100$) and a mesh size of 256 nodes with a deltaX of 0.5. When using a mesh of 128 nodes, the problem appeared with even less supersaturation. The shape of the void, which initially was defined as a circle, was resulting a square after some iterations. This transformation is emphasized in two cases: when the meshing is too coarse or when the energy penalty parameter of the concentration gradient $k_v$ is too low.
Figure 5.1: The figure shows the value of the vacancy concentration in function of the position of the mesh at the initial conditions and at T=50. For the code without any correction, the concentration of vacancies inside the void is bigger than one and the void does not grow; while for the corrected code (correcting the fluxes) the concentration inside the void is one and the cavity broadens.
Our first strategy to avoid this shape transformation was to set a value of $k_v$ higher than the used in the article. The first improve was observed with $k_v = 1$ instead of 0.5, but the best results were obtained when using a value of $k_v = 2$, with this value the shape of the void did not change, so the visualizations were showing a circle for the whole simulation. This result is quite logical because the more energy is associated to the surface, the smaller surface the system will tend to, and the surface of a circle is smaller than the surface of a square of the same width.

However, using this strategy implies changing the parameter values used in the article. In front of these difficulties (necessity to correct the fluxes, surprising shape of the void) that were not mentioned in the article [9], we decided to contact the authors. Of course they said that they also had to deal with the logarithms, to develop a strategy to avoid values of $c_v$ out of $[0, 1]$, and explained that the problem of the square voids is common in many phase-field studies and comes from the discretization of the Laplacian operator. That is why we decided to use a 9-point central difference scheme when computing the Laplacian terms $\nabla^2 c_v$ and $\nabla^2 \eta$ in equations (3.3) and (3.4), that is to say the inside laplacians. The expression used is (see Appendix A):

$$\nabla^2 G = \frac{1}{l^2}(G|_{i+1,j}^n + G|_{i-1,j}^n + G|_{i,j+1}^n + G|_{i,j-1}^n + 0.5 \cdot G|_{i+1,j+1}^n + 0.5 \cdot G|_{i+1,j-1}^n + 0.5 \cdot G|_{i-1,j+1}^n + 0.5 \cdot G|_{i-1,j-1}^n + 6 \cdot G|_{i,j}^n) \quad (5.1)$$

The results obtained with this discretization of the Laplacian showed the void as a circle (in 2D simulations) what corresponds to the minimum surface energy. The difference between the results can be seen in Figure 5.2.

### 5.4 Values below $c_v^0$

Another problem we had to face during the tests was the observation of oscillations around the value of $c_v^0$, when the solid reaches its equilibrium. These oscillations come from the correction carried on to avoid the values of vacancy concentration below zero, the correction cancels the flux if the concentration is near zero, but the values of the vacancy concentration can still be below $c_v^0$, that it’s not possible. An improvement of the correction
Figure 5.2: The figures show the shape of the void before (‘a’ and ‘b’) and after (‘c’ and ‘d’) using the 9-point central difference scheme for the inside Laplacians in equations (3.3) and (3.4) at T=0 and T=5 (100000 iterations with $\Delta t = 5e^{-5}$).
was developed in order to avoid this. The new correction is more complex and needs more computer time to solve, but the results are more correct. The function \texttt{CalcLapFluxCorr} is computing the laplacian term:

\[
(\tilde{\Psi}_v |_{i+1,j}^n + \tilde{\Psi}_v |_{i,j+1}^n + \tilde{\Psi}_v |_{i,j}^n + \tilde{\Psi}_v |_{i,j-1}^n - 4\tilde{\Psi}_v |_{i,j}^n)
\]

after the correction of the fluxes: the function computes the values of the fluxes for the X coordinates and the Y coordinates and save the results in two matrix class objects, one for the X and another for the Y axes. The flux is computed by the difference \(\tilde{\Psi}_v |_{i,j}^n - \tilde{\Psi}_v |_{i,j}^n\) and saved in \(X[i,j]\) for the X axe, and \(\tilde{\Psi}_v |_{i,j}^n - \tilde{\Psi}_v |_{i,j}^n\) and saved in \(Y[i,j]\) for the Y axe.

The results are used by the function to correct the flux of vacancies in order to avoid any vacancy concentration below \(c_{0v}\). To do that, the concentration corresponding to the next time-step is compared with the value of \(c_{0v}\) and if this concentration is below \(c_{0v}\) the flux are corrected. The correction of the flux is done by computing the maximum value of the flux that can go out from the point \([i,j]\) leaving a concentration of vacancies \(c_{0v}\) in that point for the next time-step \((F_{\text{max}_{total}})\), this maximum value is then proportionally distributed between all the fluxes that go out from the point \([i,j]\). By dividing \(F_{\text{max}_{total}}\) by the sum of all the outgoing fluxes, we obtain \(F_{\text{max}}\) which is the factor that distributes the maximum flux (see Figure 5.3). This correction does not take into account the fluxes of vacancies that go into the point, in order to have a correction independent of the order of rastering the mesh.

After the correction is done at all the points, the value of:

\[
(\tilde{\Psi}_v |_{i+1,j}^n + \tilde{\Psi}_v |_{i,j+1}^n + \tilde{\Psi}_v |_{i,j}^n + \tilde{\Psi}_v |_{i,j-1}^n - 4\tilde{\Psi}_v |_{i,j}^n),
\]

is computed at each point of the mesh with the corrected values.
Figure 5.3: The fluxes that are going out are multiplied by a coefficient (F_{max}), in order that the sum of the exiting fluxes equals to the maximum flux that can leave a concentration not below \( c_0 \) (\( F_{max_{total}} \)). The entering flux is not changed. This strategy ensures that the correction of a flux depends only on what happens in one cell, and thus that the correction is independent on the order which is done.
Chapter 6

Validation of the model

6.1 Tests

The tests we carried on were focused in the understanding of the model and the comparison of the results with those given in the article [9]. For that purpose, using the cases described in Section 3.4 some tests have been done. The cases starting with a Band (case3**) were developed to compare the results with a 1D phase field model programmed in Excel, because the symmetry of the field causes a 1D behavior in the evolution of the fields, that allows to compare a single row (if vertical band) or column (if horizontal band) with the results obtained in the simulation done with the Excel model.

The tests of the center void (case1*) where the main objective. With these tests, the model was compared with the code used in the article, by comparing different aspects as the evolution of the void in terms of shape and size and the concentration outside the void. The results obtained with the corrections in the model were becoming more and more similar to those given in the article [9].

Some tests were done for the cases of nucleation, from a uniform concentration field and applying the source term. And also for the case with two voids, to observe the movement of vacancies from the small void to the matrix and then to the big void.

The results showed in this work were performed with the SVN version rev.14 of the PHASEFIELD depot. The correction applied in this version is the last one, corrected fluxes around $c_0^0$ with the Fmax factor, giving to this Fmax factor a value of two. In the last version (rev.16) of the SVN depot, an option to choose the correction is introduced in the input file, as
well as the option to choose the time scheme (Euler or Runge-Kutta) and the discretization of the inner Laplacians (see Appendix A). This version (rev.16) has different examples in the TEST repertoire with the input files used in the simulations, it also contains an example input file with the parameters used in the article [9] and another with the parameters used for the UO₂ simulations.

6.2 Comparison of Results

The results obtained with our code were compared with the results given in the article in order to confirm the validity of ours. The fact is that our results show a faster evolution of the vacancy concentration field. Comparing the figure of the article [9] with the results we obtain, as can be observed in Figure 6.1, the order parameter profile is wider in our simulation at the same simulation time \( \tilde{t} \).

As can be observed in Figure 6.3, the vacancies are completely absorbed by the void before the end of the simulation, that means in a shorter time than it is showed in the article [9]. In our case the vacancy concentration outside the void reaches the thermal equilibrium level faster. The difference between the evolution in our simulations and the results given in the article [9] can be compared with the Table 6.1. The values give a evolution much faster in our case.

<table>
<thead>
<tr>
<th>( S_v )</th>
<th>Article [9]</th>
<th>Our Results</th>
</tr>
</thead>
<tbody>
<tr>
<td>100</td>
<td>( \lambda = 0.268; \ n = 0.5 )</td>
<td>( \lambda = 0.596; \ n = 0.5415 )</td>
</tr>
<tr>
<td>200</td>
<td>( \lambda = 0.348; \ n = 0.5 )</td>
<td>( \lambda = 0.765; \ n = 0.5837 )</td>
</tr>
<tr>
<td>300</td>
<td>( \lambda = 0.412; \ n = 0.5 )</td>
<td>( \lambda = 0.896; \ n = 0.5990 )</td>
</tr>
</tbody>
</table>

Table 6.1: Void growth as a functions of time in supersaturated systems; parameters for the power law \( \Delta R = \lambda t^n \). Obtained from the graph of Figure 6.2.

We checked our result verifying the overall vacancy balance. There is no loss or creation of vacancies in the system: the cavity growth corresponds exactly to the amount of vacancies that were in the solid at the initial stage. If we consider a volume in 2D of 256 nodes, as it is to say 255 cells in each direction (X and Y); with \( \Delta X = \Delta Y = 0.5 \), we have a surface of \( L^2 = 127.5 \times 127.5 \). Considering a void with an initial radius of \( R = 11, 48, \)
the surface of this void is \( S_{\text{ini}} = \pi R^2 = 414.03 \). With \( c_v^0 = 1.13 \cdot 10^{-4} \) and a supersaturation \( S_v = 300 \), \( c_v \text{ ini} = S_v c_v^0 \). The area of the matrix phase is \( S_{\text{matrix}} = L^2 - \pi R^2 = 15842.22 \); \( c_v \text{ ini} \times S_{\text{matrix}} \) is the fraction of the surface matrix occupied by vacancies, and also (considering \( c_v^0 << c_v \text{ ini} \)) the maximum increment of the void surface, \( \Delta S_{\text{max}} \). We calculate \( \Delta S_{\text{max}} = c_v \text{ ini} \times S_{\text{matrix}} = 537.051 \). The surface of the void at the end \( (c_v = c_v^0 \text{ outside the void}) \) will be \( S_{\text{end}} = S_{\text{ini}} + \Delta S_{\text{max}} = 951.08 \), that gives a radius \( R_{\text{end}} = \sqrt{S_{\text{end}}/\pi} = 17.3993 \) and a \( \Delta R = 17.4 - 11.48 \approx 6 \). That value is also found in the simulations as can be seen in the graph of the Figure 6.3, the small difference is due to the fact that the square regular mesh doesn’t allow to have a real circle, and the function that computes the radius has the same problem.

We also asked the authors to check if the parameters presented in the article [9] had no mistake. Their answer was that they were correct “to the best of their memory”, but they had clearly no time to really check. So, as we had obtained good qualitative behaviour, we did not insisted more on this subject.
Figure 6.1: The graphs show the order parameter profile at $\bar{t} = 50$. The upper graph is given in the article [9] and the other graph shows our results.
Figure 6.2: In the picture we can observe a zoom on the first part of the simulation (from $\tilde{t} = 0$ to $\tilde{t} = 5$), that shows a behaviour similar to the result obtained by the authors of the article [9] (upper image) for the whole simulation (until $\tilde{t} = 50$).
Figure 6.3: The figure shows the entire simulation, until $\tilde{t} = 50$. As can be observed, the radius remains constant at the end of the simulation. This is because the vacancy concentration outside the void is already near $c_v^0$. 
6.3 Nucleation tests

Nucleation process has also been tested successfully. In particular, the visualization of the simulation shows that:

- the supersaturation reach a high value at all the volume before starting the nucleation, called incubation period
- when the supersaturation is high enough, the nucleation takes place, called nucleation period
- the smaller voids decrease at the profit of the bigger ones, called growth period
- when two voids happen to coalesce, the spherical shape is reached again for the resulting void

This behavior can be observed in the Figure 6.4.
Figure 6.4: The figure shows the evolution of the volume under irradiation at times $t=1$ (a), $t=50$ (b), $t=75$ (c) and $t=100$ (d).
6.4 Other tests

Other tests were performed introducing a $1/2$ prefactor in the discretization of the internal Laplacians. This term comes from the equation we use to discretize the internal Laplacians, the fact that we consider our equation as a combination of the standard 5-point central difference scheme and a 5-point central difference scheme considering the diagonal cells instead of the “first neighbor” cells. The contact surface between the center cell and the diagonal neighbor cells is $l \sqrt{2}$ and this gives the 0.5 prefactor for these cells. The equations is:

$$\nabla^2 G = \frac{1}{2l^2} (G^n_{i+1,j} + G^n_{i-1,j} + G^n_{i,j+1} + G^n_{i,j-1}$$

$$+ 0.5 * G^n_{i+1,j+1} + 0.5 * G^n_{i+1,j-1} + 0.5 * G^n_{i-1,j+1} + 0.5 * G^n_{i-1,j-1}$$

$$- 6 * G^n_{i,j}) \quad (6.1)$$

The use of this prefactor is equivalent to use a value of $k_v$ and $k_\eta$ divided by two. The shape of the voids with this $1/2$ prefactor show a “octagonal shape” due to the equivalent low value of the gradient energy penalties.

The time discretization used in the article [9] is the Euler forward marching scheme. We tried to use the Runge Kutta order 4 scheme in order to give the solution more stability and observe less oscillations when the concentration in the matrix is close to $c^0_v$. This solution is compared with the Euler scheme for both corrections and different values of the $Fmax$ factor in the last correction. The simulations with this scheme are much longer because of the Runge Kutta scheme. The comparison of the evolution of the radius and the oscillations around $c^0_v$ for the different cases is shown in the figure 6.5.
Figure 6.5: Comparison between the different options (see Table 6.2) for a center void with $S_v = 300$. The upper graph shows the evolution of the radius and the lower one shows the behavior of $c_v$ around $c_v^{0}$ (outside the void) at Time=50.
<table>
<thead>
<tr>
<th>Correction</th>
<th>Time Scheme</th>
<th>Discretization</th>
<th>FmaxFactor/alfa</th>
<th>Test Done</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 (Old)</td>
<td>1 (Euler)</td>
<td>1 (A.3)</td>
<td>-/-</td>
<td>yes</td>
</tr>
<tr>
<td>1 (Old)</td>
<td>1 (Euler)</td>
<td>2 (A.4)</td>
<td>-/-</td>
<td>no</td>
</tr>
<tr>
<td>1 (Old)</td>
<td>1 (Euler)</td>
<td>3 (A.2)</td>
<td>-/-</td>
<td>no</td>
</tr>
<tr>
<td>1 (Old)</td>
<td>2 (RK4)</td>
<td>1 (A.3)</td>
<td>-/-</td>
<td>yes</td>
</tr>
<tr>
<td>1 (Old)</td>
<td>2 (RK4)</td>
<td>2 (A.4)</td>
<td>-/-</td>
<td>no</td>
</tr>
<tr>
<td>2 (Fmax)</td>
<td>1 (Euler)</td>
<td>1 (A.3)</td>
<td>1,2,3/-</td>
<td>yes</td>
</tr>
<tr>
<td>2 (Fmax)</td>
<td>1 (Euler)</td>
<td>2 (A.4)</td>
<td>2/-</td>
<td>yes</td>
</tr>
<tr>
<td>2 (Fmax)</td>
<td>1 (Euler)</td>
<td>3 (A.2)</td>
<td>2/0.3</td>
<td>yes</td>
</tr>
<tr>
<td>2 (Fmax)</td>
<td>2 (RK4)</td>
<td>1 (A.3)</td>
<td>1,2,3/-</td>
<td>yes</td>
</tr>
<tr>
<td>2 (Fmax)</td>
<td>2 (RK4)</td>
<td>2 (A.4)</td>
<td>-/-</td>
<td>no</td>
</tr>
<tr>
<td>2 (Fmax)</td>
<td>2 (RK4)</td>
<td>3 (A.2)</td>
<td>-/-</td>
<td>no</td>
</tr>
</tbody>
</table>

Table 6.2: The table shows the different combinations of correction, discretization and time scheme. The results of the tests with a “yes” are in the folder RESall.
Chapter 7

Application in the $UO_2$ Nuclear Fuel

7.1 Input for $UO_2$ nuclear fuel

The parameters used in the simulations mentioned earlier in this study use the values given in the article [9], in order to compare the results, but these values does not represent any special material, are just to validate the model. We now want to apply this model to $UO_2$. What we call a vacancy in the model will be a Schottky defect in $UO_2$, that is to say, a Uranium and two Oxigen atoms missing.

We have access to some of the physical parameters required: $E_{fv}$ and $D_v$. The $E_{fv}$ is computed in the PhD thesis [3]. There are three different types of Schottky defects, leading to three energies for $E_{fv}$:

<table>
<thead>
<tr>
<th></th>
<th>$S_1$</th>
<th>$S_2$</th>
<th>$S_3$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$E_{fv}$ (eV)</td>
<td>3.32</td>
<td>2.54</td>
<td>2.82</td>
</tr>
</tbody>
</table>

Table 7.1: Energies associated to the Schottky defects.

We will take the lower value $E_{fv} = 2.54$ eV.

For the vacancy diffusion coefficient $D_v$, we have a relation between the Uranium diffusion coefficient ($D_U$) and $D_v$, if we notice that $c_v$ (in site fraction) can also be seen as the probability of a site to be a vacancy. And, if we suppose that, when $D_U$ was measured $c_v$ was equal to $c_0^v$ in the
material, then we have: \( D_U = c_v^0 D_v \). The value of the Uranium diffusion coefficient has been experimentally evaluated by Hj. MATZKE [6], giving the following relation:

\[
D_U = 6.5 \cdot 10^{-5} \exp \left( \frac{-64902}{T} \right) \left( \frac{m^2}{s} \right) \tag{7.1}
\]

From the previous expressions and using:

\[
c_v^0 \approx \exp \left( \frac{-E_f^v}{k_B T} \right) \tag{7.2}
\]

We have:

\[
D_v = 6.5 \cdot 10^{-5} \exp \left( \frac{-35452}{T} \right) \tag{7.3}
\]

Using \( E_f^v = 2.54 \text{ eV} \) and a temperature of \( T = 1273K \), which is a usual temperature in the center part of nuclear fuel pellets, we can calculate the values for the non-dimension parameters, obtaining:

- \( \tilde{E}_f^v = \frac{E_f^v}{k_B T} = 23.13 \)
- \( c_v^0 = 9.01 \cdot 10^{-11} \)

In order to have the value in Joules [J] the constant

\[
k_B = 1.3806505 \cdot 10^{-23} J/K \] and with a temperature of \( T = 1273K \),

\[
k_B T = 1.757568087 \cdot 10^{-20}.\]

We have decided to study a square domain of 50nm × 50nm discretized on a 128 × 128 mesh. This means that the real \( \Delta x = 3.93 \text{Å} \). Besides, for the 128 × 128, we had the relation \( \Delta x = \tilde{\Delta} x \times l = l \). This choice imposes \( l \), and hence \( k_\eta \) (see 2.3).

### 7.2 Identification of \( \tilde{k}_v \)

In order to know all the parameters, we still need to find the value of \( k_v \) associated to the \( UO_2 \). To do that we tried to obtain the energy of the system with a vacancy concentration outside the void \( c_v = c_v^0 \), that means that almost all the energy is due to the interface between void and matrix phases. For that purpose a function is computing the energy of each point of the mesh, and tells to which region it corresponds (outside the void if \( \eta < 10^{-3} \), inside the void if \( \eta > 1 - 10^{-3} \) or at the interface if
$10^{-3} \leq \eta \leq 1 - 10^{-3}$). These values can be compared with the experimental value of the surface energy [4, 8], we have:

$$\gamma = 0.41 \times (0.85 - 1.4 \cdot 10^{-4}(T_K - 273)) \text{ in } [J/m^2] \quad (7.4)$$

The function computing the free energy during the simulation FREE, has the equation:

$$\tilde{E} = \frac{1}{(\Delta \tilde{x})^2} [(\eta - 1)^2(\eta + 1)^2[\tilde{E}_{\tilde{\eta}} + c_v \ln(c_v) + (1 - c_v)\ln(1 - c_v)]
- \tilde{A}(c_v - c_v^0)^2 \eta(\eta + 2)(\eta - 1)^2 + \tilde{B}(c_v - 1)^2 \eta^2 + \tilde{k}_v |\tilde{\nabla}c_v|^2 + \tilde{k}_\eta |\tilde{\nabla}\eta|^2] \quad (7.5)$$

With $|\tilde{\nabla}c_v|^2$ and $|\tilde{\nabla}\eta|^2$ computed in the function GradSQR by the equation:

$$|\tilde{\nabla}M_{(i,j)}|^2 = \left(\frac{M_{(i+1,j)} - M_{(i-1,j)}}{2\Delta \tilde{x}}\right)^2 + \left(\frac{M_{(i,j+1)} - M_{(i,j-1)}}{2\Delta \tilde{x}}\right)^2 \quad (7.6)$$

With $M_{(i,j)}$ the vacancy concentration field or the order parameter field. The energy obtained at each point is then added to the correspondent region of the volume.

Once we have a value for the free energy ($\tilde{E}$), it is divided by $\pi R^2$, in order to obtain the value for the surface energy $\tilde{\gamma}$. If the desired value is with units ($[J/m^2]$), it has to be multiplied by $k_B T/l^2$ to have the dimension value.

The simulations with the $UO_2$ where focused in the obtention of the value of $k_v$. This simulations had the same value for all the parameters ($S_v$, $\Delta \tilde{t}$, $\tilde{t}_{end}$, ...) except the value of $k_v$, so after the simulations we obtained a value for the free energy of the system for each value of $k_v$. A graph of the surface energy in function of $k_v$ is presented in the Figure 7.1. The equation obtained from the trendline gives the relation:

$$\gamma = 0.2205 \times (k_v)^{0.4588} \text{ in } [J/m^2] \quad (7.7)$$

If we consider the equation (7.4) and a temperature of $T = 1273K$, the value of the surface energy at this temperature is $\gamma_{\text{exp}} = 0.2911 J/m^2$. This surface energy corresponds to a $k_v = 1.832$.  

48
The concentration outside the void is not exactly $c_0^v$ as it was expected at the end of the simulation. Actually, theoretically, as it is explained in [7], the equilibrium vacancy concentration in the vicinity of a void of radius $\rho$ should be:

$$c_0^v(\rho) = \exp \left[-\frac{E_f^v}{k_B T} + \frac{\Omega}{k_B T} \frac{2\gamma}{\rho} \right]$$  \hspace{1cm} (7.8)

We calculate this quantity and compare it to the result of the code. In a $UO_2$ nuclear fuel $\Omega = 40.9 \times 10^{-30} \text{ m}^3$, and $\gamma$ is the surface energy in $[J/\text{m}^2]$. In our study we consider $E_f^v = 2.54\text{eV}$, that gives $E_f^v/k_B T = E^f_v = 23.13$. The term $\frac{\Omega}{k_B T} \frac{2\gamma}{\rho}$, using $\gamma = \frac{E^f_v}{\pi R_{adim}^2 k_B T}$ and $\rho = R_{adim} l$, can be expressed as $\frac{2\Omega E_f^v}{\pi R_{adim}^2 l^2}$. Which leads to:

$$c_0^v(\rho) = \exp \left[-\frac{E_f^v}{k_B T} + \frac{2\Omega E_f^v}{\pi R_{adim}^2 l^2} \right]$$  \hspace{1cm} (7.9)

With $E$ the free energy computed by the program, $R_{adim}$ is the radius of the void without dimension obtained with the simulation and $l$ is the length scale, in our case $l = 3.93\text{Å}$.

In our study, $E_f^v = 350.231$ and $R_{adim} = 11.296$; these values lead to a $c_0^v(\rho) = 4.98 \times 10^{-10}$. This is higher than $c_0^v = c_0^v(\rho = \infty) = 9.01 \times 10^{-11}$, but the concentration outside the void that we obtain with the code is:
\( c_v \text{out} \approx 1.2 \cdot 10^{-8} \). The difference may be due to the correction of the flux or the insufficient simulation time, that don’t allow to reach the real final value. Despite this difference between the theoretical result and our code for the concentration in the bulk, the approach is still valid because the energy associated with the interface is much higher than the energy outside the void and inside, as can be read in the table 7.2.

<table>
<thead>
<tr>
<th></th>
<th>( E_{\text{outside}} )</th>
<th>( E_{\text{interface}} )</th>
<th>( E_{\text{inside}} )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( E )</td>
<td>0.00652245</td>
<td>350.231</td>
<td>0.000635687</td>
</tr>
</tbody>
</table>

Table 7.2: Energies of the different regions.

So, the idea of identifying \( \tilde{E} \) to \( \gamma \pi R^2 \) still stands.

### 7.3 \( \tilde{A}, \tilde{B} \) and \( \tilde{L} \)

These parameters are chosen so that the energy map presents two wells. But there is no clear method to relate them to physical values, which is a drawback of this phase field approach. We have taken the same relation as in [9] for the parameters \( \tilde{A} \) and \( \tilde{B} \), which is to give the same value as \( \tilde{E}_f \). In the case of the \( UO_2 \) nuclear fuel we used \( \tilde{A} = \tilde{B} = 23.13 \).

The parameter \( \tilde{L} \) is related with the mobility of the \( \eta \) field, as \( M_v \) is with the \( c_v \) field. In this case we have taken the same value for both, as in the article [9]. So, \( \tilde{L} = 1 \) for all the simulations.

### 7.4 Time scale for \( UO_2 \)

Using \( D_v \) and \( l \), we calculate \( \tau = 2.95 \cdot 10^{-3} \text{s} \). The test that we have performed in Chapter 7 lead to a non-dimensional time \( \tilde{t} = 50 \). In this case the real final time would be \( t = \tilde{t} \tau = 0.1475 \text{s} \), which is not very interesting! This probably means, that, in the case of \( UO_2 \) \( \Delta \tilde{t} \) should be taken higher to see something happening.
Chapter 8

Conclusions

The code was developed in order to obtain a program able to compute the evolution of the vacancy concentration in a certain volume, starting from an article which give all the equations needed to use the phase field model for that purpose [9]. The fact is that the results given in the article were not easy to obtain. At the beginning because no clue about the corrections needed to have any coherent result are given, but also because depending on the corrections applied and how you define certain functions, the results are really different starting from the same initial conditions.

The importance of the discretization, the meshing and the gradient energy penalty parameter affects the visualizations and can lead to very different results depending on the way they are chosen. In the nucleation tests, the gradient energy penalty does not affect only on the shape of the voids, but also the behaviour of the volume under irradiation; the lower the energy penalty is, the bigger the number of voids. The nucleation starts at different values of vacancy concentration depending on the value of this parameter, and also the maximum number of voids varies with that parameter.

This internship has been a good experience for mixing the code development and the physics behind them, it has allowed me not only to learn more about the C++ language, but also to increase my knowledge in defect behaviour inside the nuclear fuel under irradiation and their evolution.

The phase field model is a high time consuming method, simulating a small real time and the domain that can be studied is small in comparison
with others like Kinetic Monte Carlo (KMC), for example. After the development of this code we are trying to use a hybrid model based on the SPPARKS code together with Veena TIKARE from Sandia National Laboratories, using the phase field model combined with the KMC. This technic should give faster simulations (less computer time consumption) and could be applied in larger domains and longer real time simulations.
Appendix A

Discretization of the inner Laplacian term

The discretization used to develop the Laplacian terms was initially a 5-point central difference scheme, as the authors of the article [9] used, accordingly with the following expressions:

$$\nabla^2 G = \frac{1}{l^2} (G_{i+1,j}^n + G_{i-1,j}^n + G_{i,j+1}^n + G_{i,j-1}^n - 4 \times G_{i,j}^n) \quad (A.1)$$
Demonstration for 2D:

\[
\nabla^2 c = \text{div}(\vec{\text{grad}} c) \\
\vec{\varphi} = \text{grad}(c) = \frac{c_m - c_{i,j}}{l} \\
\text{div}\vec{\varphi} = \frac{d\varphi_x}{dx} + \frac{d\varphi_y}{dy} \\
f_V \nabla^2 c dV = f_S \vec{\varphi} dS
\]

Green-Ostrogradski formula.

Discretization:

\[
\nabla^2 c \cdot V_{\text{cell}} = \\
\sum_{m \in (i,j)\text{neighbors}} S_i \cdot \varphi_i \\
\nabla^2 c \cdot l^2 = \\
\sum_{m \in (i,j)\text{neighbors}} l \frac{c_m - c_{i,j}}{l} \\
\nabla^2 c = \frac{1}{l^2} \sum_{m \in (i,j)\text{neighbors}} (c_m - c_{i,j}) \\
= \frac{1}{l^2} \left[(\sum_m c_m) - 4c_{i,j}\right]
\]

Figure A.1: Discretization of the Laplacian.
Trying to obtain a good shape evolution of a central void by changing the discretization of the Laplacian to a 9-point central difference scheme, we found different options. The option used is defined in the section 5.2, this option gives the best results in terms of shape of the voids we visualized, but may not be the best option in terms of calculus and errors. Another option we tried is given in the article [5], the equation is:

\[ \nabla^2 U_{i,j} \approx (2\alpha - 4)U_{i,j} + (1 - \alpha)(U_{i+1,j} + U_{i-1,j} + U_{i,j+1} + U_{i,j-1}) + (\alpha/2)(U_{i+1,j+1} + U_{i-1,j-1} + U_{i-1,j+1} + U_{i+1,j-1}) \] (A.2)

Which has the best results in terms of error [5] with \( \alpha = 1/3 \), but gives a squared representation of the voids.

The results showed in this report are obtained with the equation given in 5.2:

\[ \nabla^2 G = \frac{1}{l^2}(G\big|_{i+1,j} + G\big|_{i-1,j} + G\big|_{i,j+1} + G\big|_{i,j-1}) + 0.5 \ast G\big|_{i+1,j+1} + 0.5 \ast G\big|_{i+1,j-1} + 0.5 \ast G\big|_{i-1,j+1} + 0.5 \ast G\big|_{i-1,j-1} - 6 \ast G\big|_{i,j} \] (A.3)

The last option of the discretizations that has been used during the internship is:

\[ \nabla^2 G = \frac{1}{2l^2}(G\big|_{i+1,j} + G\big|_{i-1,j} + G\big|_{i,j+1} + G\big|_{i,j-1}) + 0.5 \ast G\big|_{i+1,j+1} + 0.5 \ast G\big|_{i+1,j-1} + 0.5 \ast G\big|_{i-1,j+1} + 0.5 \ast G\big|_{i-1,j-1} - 6 \ast G\big|_{i,j} \] (A.4)

This option is more consistent with the mathematical origin of the equation. It considers a mix of two 5-point central difference schemes, one considering the “normal” neighbors and the other with the diagonal neighbors; this one with a contact surface of \( \sqrt{2l} \) that leads to the 0.5 term multiplying the diagonal cells.
Appendix B

Non-dimensionalization

The non-dimension expressions are noted with the tilde (˜). The length scale is \( l = \sqrt{k_\eta/k_BT} \), the time scale is \( \tau = l^2/D_v \) and the energy scale is \( k_BT \). The non-dimension time is \( \tilde{t} = t/\tau \) and the dimensionless Laplacian \( \tilde{\nabla}^2 = l^2\nabla^2 \). The parameters \( E_f^0, A \) and \( B \) are normalized with respect to \( k_BT \). The function \( \Psi^m(c_v) \) is normalized by \( k_BT \) too. The mobility of the vacancies and the mobility of the void/matrix interface are now: \( \tilde{M}_v = D_v\tau/l^2 \) and \( \tilde{L} = LN\tau(k_BT) \). Finally \( \tilde{k}_v = k_v/k_\eta \) and \( \tilde{k}_\eta = 1 \).

In order to non-dimensionalize the expression (2.6), the equation (2.1) is functionaly derived in function of \( c_v \):

\[
\frac{1}{N} \frac{\delta \Psi}{\delta c_v} = h(\eta) \frac{\partial \Psi^m}{\partial c_v} + \frac{\partial w(c_v, \eta)}{\partial c_v} - 2k_v \nabla^2 c_v \tag{B.1}
\]

Developing the terms:

\[
\frac{\partial \Psi^m}{\partial c_v} = k_BT \left[ h^f + \ln(c_v) - \ln(1-c_v) \right] \tag{B.2}
\]

\[
\frac{\partial w(c_v, \eta)}{\partial c_v} = k_BT \left[ -2\tilde{A}(c_v - c_0^v)\eta(\eta + 2)(\eta - 1)^2 + 2\tilde{B}(c_v - 1)\eta^2 \right] \tag{B.3}
\]

\[
-2k_v \nabla^2 c_v = -2k_v \tilde{k}_\eta k_\eta \nabla^2 c_v = -2k_v \tilde{k}_v l^2 k_BT \nabla^2 c_v = -2\tilde{k}_v \tilde{k}_v \nabla^2 c_v \tag{B.4}
\]

Coupling the expressions:

\[
\frac{1}{N} \frac{\delta \Psi}{\delta c_v} = k_BT \times \left[ h(\eta) \left[ \tilde{E}_v^f + \ln(c_v) - \ln(1-c_v) \right] - 2\tilde{A}(c_v - c_0^v)(\eta^4 - 3\eta^2 + 2\eta) + 2\tilde{B}(c_v - 1)\eta^2 - 2\tilde{k}_v \tilde{k}_\eta \nabla^2 c_v \right] \tag{B.5}
\]

\[
\tilde{\Psi}_v
\]

56
Continuation to obtain equation (2.6):

\[
\frac{\partial c_v}{\tau \partial t} = \frac{1}{l} \nabla \cdot \left( \frac{M_v k_B T}{l} \tilde{\nabla} (\tilde{\Psi}_v) \right) + \xi(x, t) + P_v(x, t) \tag{B.6}
\]

And finally:

\[
\frac{\partial c_v}{\partial t} = \tilde{\nabla} \cdot \tilde{M}_v \tilde{\nabla} (\tilde{\Psi}_v) + \tau \xi(x, t) + \tau P_v(x, t) + \xi(x, t) + \tilde{P}_v(x, t) \tag{B.7}
\]

Which leads to the equation (2.10) given in the Section 2.3. Where \( \tilde{M}_v = \tau M_v \frac{k_B T}{N} \times k_B T, \xi(x, t) = \tau \xi(x, t) \) and \( \tilde{P}_v(x, t) = \tau P_v(x, t) \).

Following now the same procedure for the order parameter \( \eta \), the non-dimensionalization of equation (2.7) starts from the functional derivative of equation (2.1) in function of \( \eta \):

\[
\frac{\delta \Psi}{\delta \eta} = -Nk_B T \left[ \frac{\partial h(\eta)}{\partial \eta} \tilde{\Psi}^m + \frac{\partial \tilde{w}(c_v, \eta)}{\partial \eta} - 2\tilde{k}_\eta \tilde{\nabla}^2 \eta \right] \tag{B.8}
\]

With \( \tilde{\Psi}^m = \tilde{E}_v^m c_v + c_v \ln(c_v) + (1 - c_v) \ln(1 - c_v) \)

\[
\frac{\partial h(\eta)}{\partial \eta} = 4\eta(\eta - 1)(\eta + 1) \tag{B.9}
\]

\[
\frac{\partial \tilde{w}(c_v, \eta)}{\partial \eta} = -\tilde{A}(c_v - c_v^0)^2(4\eta^3 - 6\eta + 2) + 2\tilde{B}(c_v - 1)^2 \eta \tag{B.10}
\]

And finally:

\[
\frac{\partial \eta}{\partial t} = -\tilde{L} \left[ \frac{\partial h(\eta)}{\partial \eta} \tilde{\Psi}^m + \frac{\partial \tilde{w}(c_v, \eta)}{\partial \eta} - 2\tilde{k}_\eta \tilde{\nabla}^2 \eta \right] + \tilde{\xi}(x, t) \tag{B.11}
\]

Which, making the substitutions with the equations (B.9) and (B.10), leads to the equation (2.11) given in the Section 2.3, noting that \( \tilde{L} = L \tau k_B T N \), \( \tilde{\xi}(x, t) = \tau \xi(x, t) \) and \( \tilde{k}_\eta = \frac{1}{\tau} \frac{k_B T}{l} \). We note that, by the way \( l \) and \( \tau \) have been chosen, the non-dimensionalization leads to \( \tilde{k}_\eta = 1 \) and \( \tilde{M}_v = \tau M_v \frac{k_B T}{N} \times k_B T = \tau D_T \frac{k_B T}{l} \times k_B T = 1 \).
Appendix C

Demonstration of the Gradient

Starting from these equations:

\[ F = \int_V f_0(c) + \frac{\lambda c}{2} (\nabla c)^2 dV \quad (C.1) \]

\[ \mu(r_0) = \frac{\delta F}{\delta c(r_0)} = \lim_{\epsilon \to 0} \frac{F(c + \epsilon \delta r_0) - F(c)}{\epsilon} \quad (C.2) \]

Using \( f_0(c + \epsilon \delta r_0) \approx f_0(c) + \epsilon \delta r_0 \frac{\partial f_0}{\partial c} \) and \( \nabla (c + \epsilon \delta r_0) = \nabla c + \nabla (\epsilon \delta r_0) \)

\[ \mu(r_0) = \int_V \frac{\partial f_0}{\partial c} \delta r_0 dV + \frac{\lambda c}{2} \lim_{\epsilon \to 0} \frac{1}{\epsilon} \int_V 2 \nabla c \cdot \nabla (\epsilon \delta r_0) + (\nabla (\epsilon \delta r_0))^2 dV \quad (C.3) \]

\[ \int_V \frac{\partial f_0}{\partial c} \delta r_0 dV = \frac{\partial f_0}{\partial c}(r_0) \quad (C.4) \]

\[ \int_V \nabla c \nabla (\epsilon \delta r_0) dV = \int_V \sum_{X=x,y,z} \frac{\partial c}{\partial X} \frac{\partial (\epsilon \delta r_0)}{\partial X} dV = \sum_{X=x,y,z} \int_V \frac{\partial c}{\partial X} \frac{\partial (\epsilon \delta r_0)}{\partial X} dV \quad (C.5) \]

For example, for \( X = x \) we integrate by parts.

\[ \int_y \int_z \left( \int_{x=-\infty}^{+\infty} \frac{\partial c}{\partial x} \frac{\partial (\epsilon \delta r_0)}{\partial x} dx \right) dydz = \int_y \int z \left( \left. \left[ \frac{\partial c}{\partial x} \epsilon \delta r_0 \right]_{x=-\infty}^{+\infty} \right) - \epsilon \int_{x=-\infty}^{+\infty} \frac{\partial^2 c}{\partial x^2} \delta r_0 dx \right) dydz \quad (C.6) \]
The term (1) is null because $\delta_{r_0}$ is equal to zero for $x = \pm \infty$.

$$
\int_y \int_z -\epsilon \frac{\partial^2 c}{\partial x^2}(x_0, y, z) \delta_{y_0} \delta_{z_0} dy dz = -\epsilon \frac{\partial^2 c}{\partial x^2}(x_0, y_0, z_0) = -\epsilon \frac{\partial^2 c}{\partial x^2}(r_0) \quad \text{(C.7)}
$$

Doing the same for $X = y$ and $X = z$ leads to:

$$
\int_V \vec{\nabla}c \cdot \vec{\nabla}(\epsilon \delta_{r_0}) = -\epsilon \left[ \frac{\partial^2}{\partial x^2}(r_0) + \frac{\partial^2}{\partial y^2}(r_0) + \frac{\partial^2}{\partial z^2}(r_0) \right] = -\epsilon \nabla^2 c \big|_{r_0} \quad \text{(C.8)}
$$

The last term $(\vec{\nabla}(\epsilon \delta_{r_0}))^2 = \epsilon^2 (\vec{\nabla}(\delta_{r_0}))^2$, so $\frac{(\vec{\nabla}(\epsilon \delta_{r_0}))^2}{\epsilon}$ is proportional to $\epsilon$ and tends to zero when $\epsilon$ tends to zero.
Appendix D

Input File

Here is an example of the ‘.xml’ input file.
Figure D.1: Input file with the values used in the simulations for the $UO_2$. 
Bibliography


SUMMARY (english): This study shows the steps followed to develop the phase field model code starting from an article which gives the main equations and the results obtained. The phase field model is used to study the evolution of the microstructures. The total free energy functional of the heterogeneous material in terms of the free energy of its constituent phases and interfaces guides the evolution of the fields.

This study considers a metal with two phases, a matrix phase with point defects, namely vacancies, and a void phase. The vacancy concentration varies smoothly from the matrix to the void phase (diffuse interface). The void is the result of the condensation of vacancies from the surrounding matrix with a super-saturated vacancy concentration. The model uses the form of coupled Cahn-Hilliard and Allen-Cahn type equations to govern the dynamics of the vacancy concentration field and the void microstructure in the matrix, respectively.

The free energy equation:

$$\Psi[c_v, \eta] = N \int_\Omega [h(\eta)\Psi^m(c_v) + \omega(c_v, \eta) + k_v |\nabla c_v|^2 + k_\eta |\nabla \eta|^2] d\Omega$$

and the kinetic equations derived from the previous equation:

$$\frac{\partial c_v}{\partial t}(x, t) = \nabla \cdot \left( \frac{M_v}{N} \frac{1}{\Psi} \frac{\delta \Psi}{\delta c_v} \right) + \xi(x, t) + P_v(x, t)$$

$$\frac{\partial \eta}{\partial t}(x, t) = -L \frac{\delta \Psi}{\delta \eta} + \zeta(x, t)$$

are the equations used by the code to simulate the evolution of the phase fields. Starting from these equations, applying a non-dimensionalization scale and a discretization, the code computes the values of the fields at each point of the mesh and at each time step.

The results are saved in “.vtk” files in order to use Paraview to see the evolution of the fields. Some output files as the vacancy concentration profile, the radius of the void or the free energy are given by the code for post-processing. The main time consuming part of the study was to solve the problems we had to face while programming: values of the concentration out of range, logarithms of zero and negative values and the shape of the voids were the most significant. The correction of the vacancy flux solved the first, a pseudo-code for the logarithms solved the second and a new discretization of the laplacian solved partially the third (it depends on the refinement of the mesh).
All the results were compared with those given in the article, showing a correct qualitative behaviour.
We then identified the parameters needed to compute the phase field applied to the $UO_2$ nuclear fuel.
SUMMARY (french): Cette étude montre les étapes suivies pour développer le modèle de champ de phase à partir d’un article qui donne les principales équations et les résultats obtenus. Le modèle de champ de phase est utilisé pour étudier l’évolution des microstructures. La fonctionnelle “énergie libre totale” du matériau hétérogène (somme des énergies libres de ses phases constitutives et interfaces) guide l’évolution des champs.
Cette étude considère un métal avec deux phases, une phase de matrice avec les défauts ponctuels, à savoir les postes vacants, et une phase vide. La concentration de lacunes varie continuellement à partir de la matrice à la phase de vide (interface diffuse). Le vide est le résultat de la condensation de lacunes dans la matrice environnante où la concentration de lacunes est sursaturée. Le modèle utilise la forme couplée des équations type de Cahn-Hilliard et d’Allen-Cahn pour régir la dynamique du champ de concentration de lacunes et la microstructure des vides dans la matrice, respectivement.
L’équation de l’énergie libre:
\[
\Psi[c_v, \eta] = N \int_\Omega \left[ h(\eta) \Psi^m(c_v) + w(c_v, \eta) + k_v |\nabla c_v|^2 + k_\eta |\nabla \eta|^2 \right] d\Omega
\]
et les équations cinétiques obtenus à partir de l’équation précédente:
\[
\frac{\partial c_v}{\partial t}(x, t) = \nabla \cdot \left( \frac{M_v}{N} \nabla \frac{1}{N} \frac{\delta \Psi}{\delta c_v} \right) + \xi(x, t) + P_v(x, t)
\]
\[
\frac{\partial \eta}{\partial t}(x, t) = -L \frac{\delta \Psi}{\delta \eta} + \zeta(x, t)
\]
sont les équations utilisées par le code pour simuler l’évolution des champs de phase. À partir de ces équations, et après adimensionalisation et discrétisation, le code calcule les valeurs des champs en chaque point du maillage et à chaque pas de temps.
Les résultats sont enregistrés dans des fichiers “.Vtk” afin de pouvoir utiliser Paraview pour visualiser l’évolution des champs. Des fichiers de sortie comme le profil de concentration de lacunes, le rayon du vide ou de l’énergie libre sont données par le code pour post-traitement.
La difficulté principale du stage a été de résoudre les problèmes que nous avons dû affronter pendant la programmation: les valeurs de la concentration hors limites, logarithmes de zéro et des valeurs négatives et la forme des vides sont les plus importantes. La correction du flux des lacunes a résolu le premier problème, un pseudo-code pour les logarithmes
a résolu le deuxième et une discrétisation nouvelle des laplacians a résolu partiellement le troisième (cela dépend de la finesse du maillage). Tous les résultats ont été comparés à ceux donnés dans l’article, montrant un comportement qualitatif correct.

Nous avons ensuite obtenu par identification les paramètres nécessaires pour calculer les champs de phase appliqués au combustible nucléaire $UO_2$. 