Publishing House
AKAPIT

COMPUTER METHODS IN MATERIALS SCIENCE
Informatyka w Technologii Materiałów

Vol. 16, 2016, No. 1

APPLICABILITY AND PERFORMANCE ANALYSIS OF THE PHASE-FIELD MODELLING BASED ON THE CAHN-HILLIARD METHOD FOR THE BINARY ALLOYS

KAMIL J. DUDEK*, DANUTA SZELIGA

Akademia Górniczo-Hutnicza im. Stanisława Staszica, al. A. Mickiewicza 30, 30-059 Kraków, Poland
**Corresponding author: kamdud@agh.edu.pl*

Abstract

The Cahn-Hilliard method describes the driving force of the process of phase separation. Considering a binary (two-phase) alloy of conserved composition variable, the C-H model makes it possible to trace the process at which the domain creates pure subdomains of each component. Gururajan et al. (2005) have offered a simplified reference implementation of the aforementioned model, using the C language and FFTW libraries. It allows to perform a reverse Fourier transform to acquire a domain composition profile at a given time. Being a proof-of-concept, this approach does not stress on performance and the dedicated tools, like OpenPhase might not provide the flexibility necessary for new implementations. The following work aims on analyzing the C-H algorithm with regards of optimization, taking the Gururajan, among others, as a reference problem. Utilizing the models accompanied by replicable results facilitates the analysis of FFTW and GSL libraries, answering whether to search the replacement implementations and where (if necessary) work on code performance and parallelization. Finally, the improved model is intended to be used on external (non-random) initial composition, taken from ongoing simulations, in hope of preparation of the pearlitic steel model in the future.

Key words: imaging, visualization, iipsrv, modeling

1. INTRODUCTION

The phase-field model (PF) is an increasingly popular method of predicting the microstructure evolution. Among the multitude of advantages, this method makes it possible to neglect the initial assumptions about the grain size and/or shape. The theory behind creating the phase field model consists of several basic pieces, which are discussed in the first part of the following paper.

2. THE ORDER PARAMETER

The order parameter is a way of escape from the limitations of a traditional "sharp interface model" (ShInM). Such a model assumes that the proper description of a microstructure can be achieved by describing the locations of the interfacial boundaries, because the interfaces between the domains are defined as infinitely sharp. The consequence of this approach is the need of finding the moving interface track and, in the end, much higher mathematical complexity, that limits the area of application of such a model. PF describes the interface in a diffuse manner. It means that the entire microstructure is defined using the continuous functions of space and time, encapsulating the phase-field variables. They indicate the same values as the sharp model inside the domains, but express the domain transition in a radically different manner: continuously. The model therefore does not contain the explicit interface boundaries(and it has to be noted that the accurate sharp model is unknown for a wide area of problems (Provatas & Elder, 2005), replacing them with the equations covering the entire system. It can be said (Bhadeshia, 2004) that the state of the entire

ISSN 1641-8581

system (microstructure) is represented by an order parameter variable φ. The sharp value of φ equal the properties of the sharp model, and the gradual values ($0 < \varphi < 1$) represent the interface. This continuous manner of describing the interface leads to the superset of the values of the order parameter φ over the entire microstructure. Defining those values means applying the phase field over the system.

3. FREE ENERGY FUNCTIONAL

Of course, describing the material state by means of the phase field φ values does not include the driving force that is responsible for the microstructure evolution. The most general, thermodynamically speaking, way of grasping the driving force is presenting it as a way to reduce the free energy of the system. In case of the energy composition, the theory behind "energy sources" is much more conventional that the phase-field approach itself. So the general free energy F in the system is, in most, models (Moelans et al., 2007) composed of bulk free energy, interfacial energy and elastic-strain energy. More detailed models include magnetic, electromagnetic and electrostatic interactions with the material. What differentiates the PF model from the classical approach is relation between the energy and the phase field order parameters. This excludes the possibility of assuming the homogeneity of the free energy across the entire investigated system. Despite being initially counter-intuitive and radically different than the well-known Gibbs f.e. model, expressing the free energy as the functional opens multiple possibilities.

4. DECOMPOSED OBJECTS OF THE FREE ENERGY FUNCTION

The most significant problem that has to be faced while composing the phase field model is determining the correct free energy equation. The energy flux is determined by the boundaries of the thermodynamic energy functional, but the equation itself is strictly depended on the process we wish to simulate (Boettinger & Warren, 2014). For example, in the simulation of a solidification of a single phase in a binary liquid, under the isothermal conditions, the free energy is expressed as a functional defined in the works of Wheeler, and McFadden (Wheeler & Boettinger, 1992):

$$F = (\phi, C, T) = \int_V \{f(\emptyset, C, T) + K_1 K_2\} \, \mathrm{d}\vec{V} \qquad (1)$$

Where $K_1 = \frac{k_\emptyset}{z}|\nabla\phi|^2$ and $K_2 = \frac{k_c}{2}|\nabla c|^2$ are the coefficients that hold the diffuse specifics of the interface, i.e. govern the interfacial energy. It is the direct development of the statement formed by Cahn-Hilliard team. The f function is the homogeneous free energy density, a temperature-dependent feature of the system. Formulating the generalized free energy functional shifts the problem to defining the actual free energy density. Wheeler and Boettinger (1992) suggests the ideal solution by expressing the free energy density as a function of the phase field and concentration, thus:

$$f(\phi, c) = c f_B(\phi) + (1 - c) f_A(\phi) + \frac{RT}{V_{molar}}[c \ln c \, (1 - c) \ln(1 - c)] \qquad (2)$$

Which decomposes the f.e. even further, by introducing the free energy equations for pure compositions (A and B), as the functions of phase. Kobayashi (1992) used two functions, NIST guidelines use the f function with regards of φ, C, but also T, so they introduce multiple instances, along with the interpolation functions. The much-desired simplifications, assuming the ideal solution circumstances, lead to the form with the pure component free energy equation and the necessary double-well potential functions:

$$\begin{cases} f_{A\oplus\mathrm{B}}(\phi, T) = \\ L_{A\oplus\mathrm{B}} \dfrac{\left(T - T_{melting}^{A\oplus\mathrm{B}}\right)}{T_{melting}^{A\oplus\mathrm{B}}} p(\phi) + \dfrac{W_{A\oplus\mathrm{B}}}{2} g(\phi) = \\ = \check{L}_{A\oplus\mathrm{B}}(T) p(\phi) + \dfrac{W_{A\oplus\mathrm{B}}}{2} g(\phi) \\ g' = \dfrac{\partial g}{\partial\phi} = \dfrac{\partial}{\partial\phi}(\phi^2(1 - \phi)^2) = 2\phi(1 - \phi)(1 - 2\phi) \\ p' = \dfrac{\partial p}{\partial\phi} = \dfrac{\partial}{\partial\phi}\left(\phi^3(6\phi^2 - 15\phi + 10)\right) = \\ = 30\phi^2(\phi - 1)^2 = 30g(\phi) \end{cases} \qquad (3)$$

Because the general phase field equation requires to satisfy the completeness of form $\frac{\partial\phi}{\partial t} = \widetilde{M}(\widetilde{K_\phi} - \frac{\partial f}{\partial\phi})$ (which states that the evolution of the order parameter in time is dependent on the change on the free energy density with regards to the phase), we need to construct the $\frac{\partial f}{\partial\phi}$ term, as simple as $\frac{\partial f}{\partial\phi} = (1 - C)\frac{\partial f_A}{\partial\phi} + (0 - C)\frac{\partial f_B}{\partial\phi}$ which, being aware of $f_{A\oplus\mathrm{B}}(\phi, T)$, is in fact nothing more than:

$$\frac{\partial f}{\partial \phi} = (1-C)\left(\check{L}_A(T)p'^{(\phi)} + \frac{W_A}{2}g'^{(\phi)}\right) + (0-C)\left(\check{L}_B(T)p'^{(\phi)} + \frac{W_B}{2}g'^{(\phi)}\right) \qquad (4)$$

Which concludes the basic description of the system.

4.1. Penalizing the diffuse and sharp changes of φ

While constructing the phase field evolution equation $\frac{\partial \phi}{\partial t}$, all the phenomena responsible for the energy have to be included in in the functional of the concentration (the Ginsberd-Landau free energy functional).

$$\begin{cases} \dfrac{\partial \phi}{\partial t} = \nabla \cdot \left(\mathrm{M}\nabla \dfrac{\delta F}{\delta \phi}\right) \\ \mu = \dfrac{\delta F}{\delta \phi} \\ \dfrac{\partial \phi}{\partial t} = M\nabla^2\left(\dfrac{\partial f}{\partial \phi} + W\nabla^2\phi\right) \end{cases} \qquad (5)$$

where the free energy function is, again, a double potential well. Components of μ are responsible for "penalizing" the system for evolving by means of energy change. The entire process is a combination of sharp and diffuse changes of the energy density, so the f describes the states of diffuse values in the order parameter domain ($0 < x < 1$), while e is the cost of sharp, jump changes. Setting up those two events results in phase separation into subdomain, but with the smooth interface transition.

5. DIFFUSION AND DIFFUSIVITY

The previous paragraph introduced an example phase field problem denoted as $\frac{\partial \phi}{\partial t} = \nabla \cdot \left(\mathrm{M}\nabla \frac{\delta F}{\delta \phi}\right)$, where the important, yet only slightly distinct expressions appear, notably the $\nabla \cdot u$ and $\nabla\varphi$ (divergence and gradient). In order to simplify the notation (which is part of the PF problem formulation issue), there is a need to re-evaluate the terms used to describe the flux and diffusion. The necessary chemical entities are:

- The concentration order parameter φ_c, but in a molar form $\left[\frac{mol}{m^3}\right]$(how much substance in one unit)
- The diffusivity D, as $\left[\frac{m^2}{s}\right]$, related to the velocity of the particles the undergo the diffusion. It encloses the very specifics of the process.
- The flux J, being the consequence of the diffusion, shows how much substance flows through the control area at a time, thus $\left[\frac{mol}{m^3 \cdot s}\right]$

While the flux is a simple derivative while considered in a 1D system $\left(J = -D\frac{\partial \phi}{\partial x}\right)$, that derivate has to be elevated to a generalized form if defined in higher dimensions: $\vec{J} = -D\nabla\phi$. Taking into account that the $\frac{\partial \phi}{\partial t}$-form problems must follow the boundaries of the mass conservation equation and simplifying the Fick equation set, it is suggested to introduce the terms (which will in the future facilitate the process of implementation):

$$\begin{cases} \overrightarrow{-J} = D\nabla\phi = Dgrad(\phi) = \overrightarrow{flux} \\ \nabla \cdot \overrightarrow{flux} = diffusion(D_\phi) \\ \dfrac{\partial \phi}{\partial t} = \nabla \cdot \overrightarrow{-J} = \nabla \cdot \overrightarrow{flux} = diffusion(D_\phi) \end{cases} \qquad (6)$$

It introduces the implicit "artificial' function *diffusion(X)*, that takes the extended term consisting of system's free energy manipulators (like the double-well free energy function). Such a constructed operation might be proven useful in the later stages.

6. BASIC EXAMPLE AT ONE DIMENSION

Using the fastest and most convenient method (with FiPy), let us construct the initial model for further development, in order to explain the specifics of the solver. By means of the general phase field term and the decomposed components of the free energy equation, the simplified PF in 1D mesh of x is:

$$\begin{cases} -\dfrac{\partial \phi}{\partial t} = M\nabla^2\left(\dfrac{\partial f}{\partial \phi} - a\nabla^2\phi\right) \\ f(\phi, T) = L\dfrac{(T - T_{melting})}{T_{melting}}p(\phi) + \dfrac{W}{2}g(\phi) \\ g(\phi) = \phi^2(1-\phi)^2 \\ p(\phi) = \phi^3(6\phi^2 - 15\phi + 10) \end{cases} \qquad (7)$$

The p and g functions are the example, arbitrary equations, that conform to the double potential well requirement. By performing the necessary derivations the following form is obtained:

$$-\frac{\partial f}{\partial \phi} = (W\phi(1-\phi)(1-2\phi) + \frac{(T - T_{melting})}{T_{melting}}30\phi^2(1-\phi)^2) = m_\phi\phi(1-\phi) \qquad (8)$$

Providing the analytical solution of x, after Beckermann's equation of steady state in infinite domains, leads to the solution convergence.

7. REFERENCE EXAMPLE AT TWO DIMENSIONS

Let us consider the two-dimensional space of elementary volumes with the initial composition defined as an order parameter fluctuation around the mean IF value of 0.5 (*uniflux* area distribution):

$$\begin{cases} uniFlux(\Omega_0) = \phi_0 = {}^1/_2 \\ var(\Omega_0) = \left(sd(\Omega_0)\right)^2 = \sigma^2 = 0.01 \\ gaussianNoise(\phi, \phi_0, \sigma) = \\ \left(\sigma\sqrt[2]{2\pi}\right)^{-1} \times e^{\frac{(\phi-\phi_0)(\phi-\phi_0)}{2var(\Omega_0)}} = \\ = \frac{10}{\sqrt{2\pi}} e^{-50\left(\phi-{}^1/_2\right)^2} \\ \therefore \left(\phi^0(i,j) \cong \frac{1}{2}\right) \wedge \left(\frac{\sum \phi^0_{n\times n}(i,j)}{n \times n} = \frac{1}{2}\right) \end{cases} \tag{9}$$

The fluctuations in the system will not require the threshold level, i.e. each of them has a potential to grow. That leads to the spontaneous separation of the composition into two pure subdomains made of pure components. The process starts with the homogenous system $\left(\frac{\sum \phi^0_{n\times n}(i,j)}{n\times n} = \frac{1}{2}\right)$ and is called the *spinoidal decomposition* (Elliot, 1989). Such a system evolves by means of two intertwining effects of energy change (penalizing the diffuse and sharp changes), so the process of phase segregation is the effect of the minimization (or "decay") of the free energy. So the phase field problem can be described as a way at which the system evolves due to the changes of the energy:

$$\begin{cases} \frac{\partial f^{(1)}_{diffuse}}{\partial \phi} = \\ \frac{\partial}{\partial \phi}\left(\frac{a^2}{2}(\phi^2 - \phi)^2\right) = \\ 2a^2\phi^3 - 3a^2\phi^2 + a^2\phi \\ F_{sharp} = \epsilon^2 \nabla^2 \phi \\ \frac{\partial \phi}{\partial t} = \nabla \cdot D\nabla \left(\frac{\partial f^{(1)}_{diffuse}}{\partial \phi} - F_{sharp}\right) = \\ \nabla \cdot D\nabla\left((2a^2\phi^3 - 3a^2\phi^2 + a^2\phi) - \epsilon^2\nabla^2\phi\right) \end{cases} \tag{10}$$

Knowing that $\nabla f = \sum_{i=1}^{n}\left(\frac{\partial f}{\partial x_i} \cdot \widehat{u_i}\right)$ and that our function $diffusion(D_\phi)$ takes the argument to perform the "$\nabla \cdot D\nabla\phi$" operation while increasing the depth the more arguments it gets $\left(\nabla \cdot \left(D\nabla\left(\nabla \cdot (D\nabla\,\phi)\right)\right)\right.$ etc.), we get:

$$\begin{cases} D_1 = diffusion(a^2(6\phi(\phi-1)+1) = \\ \qquad \nabla \cdot Da^2(6\phi(\phi-1)+1)\nabla\phi \\ D_2 = diffusion(D, \epsilon^2) = \nabla \cdot D\epsilon^2\nabla^2\phi \end{cases} \tag{11}$$

While the *diffusion()* name might be an overstatement (as it can describe the general flow of the value-set over the descent direction of its gradient), it grasps the specifics of the C-H 2D system perfectly, thus:

$$\frac{\partial \phi}{\partial t} = D_1 - D_2 \tag{12}$$

The above form is the one accepted by the software using finite element/difference/volume methods. Other forms, like the PyMKS indirectly-sourced (Cheng & Rutenberg, 2005) $\dot{\phi} = \nabla^2(\phi^2 - \phi) - \gamma^2\nabla^2\phi$ require reformulating with regards to the Laplacian operator $lap(f) = \Delta f = \sum_{i=1}^{n}\frac{\partial f^2}{\partial x_i^2} = div(grad(f)) =$
$\nabla \cdot \nabla f = \nabla^2 f$.

Basic decomposition leads to the descent of evolution speed and after the visible division, the solute undergoes the well-known process of Ostwald ripening (Baldan, 2002). In order to achieve this result, one of the phases must evolve into a significant advantage over the other. Independent basic subdomains of pure component take spherical forms and diffusion forces them to get absorbed into a bigger subdomain of the same component.

8. OVERVIEW OF THE IMPLEMENTATION STRATEGIES

Clear representation of the problem using the aforementioned equations makes it possible to implement it as a numerical algorithm. Provatas and Elder explore the most easily-applicable methods of expressing the phase field problem, using the algebraic equations for evaluation of the corresponding PDEs (notably finite difference and finite volume methods). The continuous system, as well as PF equations, are discretized, and the solutions of the equations represent the values of the PF variable set across the lattice points. Let us consider the conserved (Cahn-Hilliard) equation of the flux conservation, as: $\frac{\partial \phi_{conserved}}{\partial t} = \nabla \cdot (\mathrm{M}\nabla\mu)$ also known as $\frac{\partial c}{\partial t} = -\nabla \cdot \vec{J}$. The system is represented as a mesh with nodes, consisting of "volumes". Each node is anchored in the geometrical center of the volume. In other words, points of the computational mesh (point mesh) are surrounded by the volumes that form a mesh out of themselves (volume mesh). The finite volume method implies integrating the equation over the volume of the finite volume unit and using the Ostrogradsky's theorem to decompose the divergence.

$$\begin{cases} \int_{\Omega_{FVM}} \frac{\partial \phi_c}{\partial t} dxdydz = -\int_{\Omega_{FVM}} \nabla \vec{J} dxdydz \\ \iiint_{\Omega} (\nabla \vec{F}) d\Omega = \oiint_S (\vec{F}\vec{n}) dS \\ \int_{\Omega_{FVM}} \nabla \vec{J} dxdydz = -\int_{A_{FVM}} \vec{J} d\vec{s} \end{cases} \tag{13}$$

The "evolution" mechanism related to time "ticking" the consecutive time steps is a simple forward difference, so the main time step equation for evolving the system using FVM will therefore be:

$$-\frac{\phi_c^{n+1}(i,j) - \phi_c^n(i,j)}{\Delta t} = \left(\vec{J}^n_{right} - \vec{J}^n_{left}\right) + \left(\vec{J}^n_{top} - \vec{J}^n_{bottom}\right) \tag{14}$$

where the corresponding R, L, T, B symbols mark the directions (right, left, top, bottom). The finite volume method solvers have been widely implemented, so as long as the problem is properly defined in a FVM-approachable manner of PDEs, there is a multitude of tools that can be used to process the computations. This includes OpenFVM(dedicated only to solving the already-populated areas using the finite volume method), OpenFOAM, PyMKS and FiPy. Many of them are painfully underdocumented, but OpenFOAM excels in its versatility, and FiPy offers a friendly syntax and well-described examples. The initial phase field model, accompanied by this paper and described later, was run against the FiPy environment.

8.1. FiPy

FiPy is a PDE solver created with the Python programming language to solve the problems described using the finite volume method. It offers the syntax that significantly facilitates the process of creating a PF model, while being viewer-independent and easy to connect with other software (which is a much-desired feature). On top of that, it is ready to perform distributed parallelized computations using MPI. All these qualities makes FiPy a very promising platform, not only for the general computations, but also as an auxiliary (or "glue") environment. Establishing the FiPy worker requires the functional Python platform (with strong preference towards a working PIP manager), NumPy, PySparse, SciPy, Matplotlib and Mayavi. All of them are either readily available or easily accessible on current Red Hat family of operating systems.

8.2. CH-MuSE

Researching the possible adaptations of the model in other software, showed the ubiquity of the approach created by Gururajan (2005). Another common approach towards solving the phase field problem family is using a semi-implicit method by means of Fourier Space and Fourier Transform. Specifics of the Fourier space make advantage of the wave vector: the discrete Fourier transformation replaces the continuum field equations with algebraic equations oriented on the Fourier space. Provatas explains the general (and quite broad) process of constructing the Fourier equations, but the most notorious example is the microstructure evolver reference implementation of Rapaport (Rapaport, 2002) by Gururajan (2005). The selected conserved parameter equation solved in it is as follows:

$$\begin{cases} \left(\frac{\partial \phi_c}{\partial t} = \nabla M \nabla \mu\right) \xrightarrow{dimensionless} \\ \left(\frac{\partial \phi_c}{\partial t} = \nabla^2(h - 2\nabla^2 \phi_c)\right) \\ \left(\frac{\partial \{\phi_c\}_g}{\partial t} = -g^2(\{h\}_g + 2g^2\{h\}_g)\right) \xrightarrow{discretize} \\ \left(\frac{\phi_c(g, t+\Delta t) - \phi_c(g,t)}{\Delta t} = -g^2\left(\{h\}_g - 2g^4 \phi_c(g, t+\Delta t)\right)\right) \end{cases} \tag{15}$$

Where the first equation is responsible for the dimensionless form, and the second one is a discretization. The implementation (CH-MuSE, Cahn-Hilliard Microstructure Evolution) takes the pre-defined composition profile *φ_c* at the time step t and calculates the h state along with its Fourier transform. It allows calculating the composition profile at the next time step, with the inverse Fourier transform. CH-MuSE shows how the phase creation (here, via separation) has an energy cost, balancing itself from the processes of decomposition and interfacing. program ends after satisfying the requested number of time steps and starts with the initial composition from "uniform noise": the randomlygenerated fluctuations, using the GNU Scientific Library RNG. The Fourier Transform itself is conducted using the Fastest Fourier Transformation in The World (FFTW) library. Popularity of the Gururajan implementation, despite being originally single-threaded and limited, marks its presence in multitude of discussions related to creating the phase-field solvers, many of them still ongoing, without the actual solver surfaced. In the process of creating the implementation, following the solutions that extensively rely on external libraries might not be as fruitful as expected.

8.3. OpenFOAM

The numerical solver toolbox OpenFOAM is a software for fluid oriented continuum mechanics (CFD, computational fluid dynamics) that provides a wide range of solvers for flow dynamics, meshing, turbulence modeling and data post-processing. Contrary to other environments, the OpenFOAM is invoked by importing (including) the header sets into own programs, but the features offered by the toolbox are quite extensive. FOAM is commonly praised for its stability, maturity and optimization. There exist (although not many) phase field implementations made with OpenFOAM. As with the entire PF subject, most of the code is strictly adapted to a specific case, non-universal and not reusable. There is a certain degree of independence in the works of Takada (GitHub resources, 2011), coded by Donaldson and ported into the update FOAM by Weiss (Man & Kleijn, 2014). The proposed solution uses the Cahn-Hilliard equation together with Navier-Stokes coupling for the phenomenon of the immiscible fluid transport, as explored by Man and Kleijn (2014).

8.4. OpenPhase

Rich selection of use cases for OpenFOAM is juxtaposed with the OpenPhase project that focuses only on the phase field simulations. Similarly to FOAM, it offers a functionality in the library form, to invoke from own implementations. It heavily relies on the GSL and FFTW libraries and does not yet work properly in the distributed parallelized systems (like MPI). OpenPhase is a promising project, but its complexity, as well as early stage of development, might make the efforts disproportionate in regards to amount of initial work with creating the model. This statement has to be re-evaluated while dealing with the more complex systems, as the current difficulty of defining the PF systems cripples the ease of use if this otherwise well-documented project.

8.5. PyMKS

It is a relatively new and still unusual approach towards solving the problems in material science, with the multiscale scope. It links not only multiple scopes of problem-solving, but also utilizes multiple, rarely-coupled methods, like signal processing, machine learning and spatial statistics (Yamanaka, 2007) (GitHub resources, 2011). PyMKS allows to compute the system states using the hard-coded equation, imported from the pymks.datasets:

$$\dot{\phi} = \nabla^2(\phi^3 - \phi) - \gamma^2\nabla^4\phi \longrightarrow$$
$$\frac{\partial\phi}{\partial t} = \nabla\cdot\nabla(\phi^3 - \phi) - \nabla\cdot\gamma^2\nabla^2\phi \qquad (16)$$

The software does not use the finite-unit-type numerical method, but works with the semi-implicit spectral scheme with periodic boundary conditions, which is often accompanied with the Fourier Transforms, as noted by Moelans (2007) and Gururajan (2005). Friendliness of the environment is comparable to the level of FiPy. It might correlate with the Python language used in both those projects.

8.6. Micress

There is a commercial toolchain suitable for dealing with the PF problems, called Micress. The exemplary Micress implementations include the models for dendritic growth and peritectic reaction (directional solidification phenomenon). The provided documentation shows an extensive and solveragnostic method of describing the material and all the driving forces of the processes. Yet, the availability of the software and a scarce number of walkthrough solutions discourage from exploring this tool as a first choice. Nonetheless, after the model develops further, using Micress should be reconsidered. That should however occur after dealing with the fundamentals of the investigated PF problem.

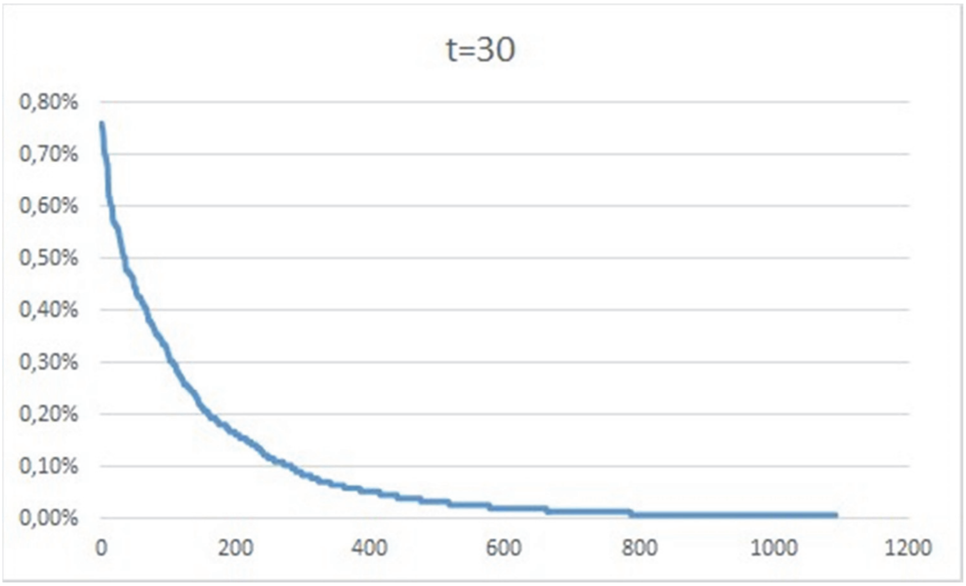

Fig. 1. *Strong-end value of the phase parameter after t=30*

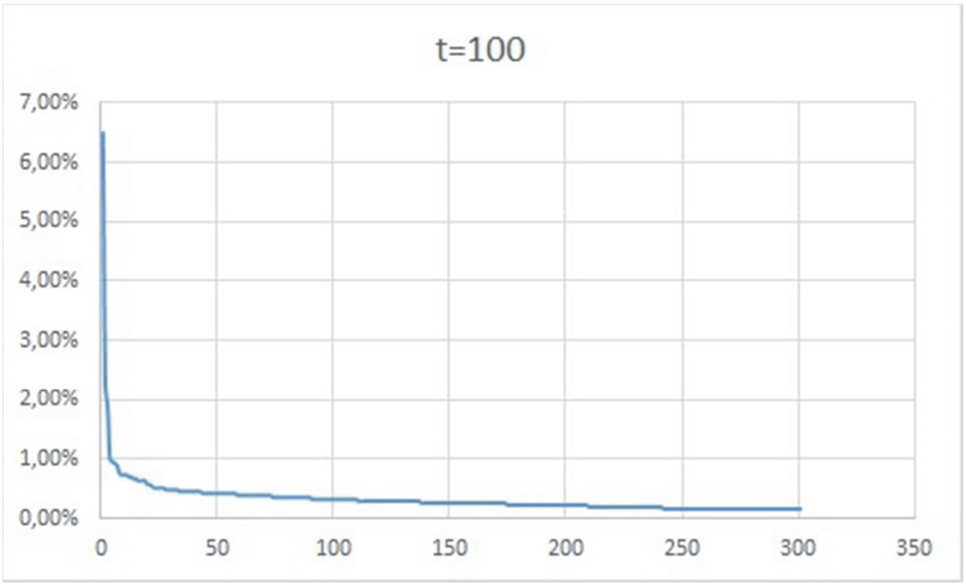

Fig. 2. *Strong-end value of the phase parameter after t=100*

9. FIPY MODEL

The reference, exemplary implementation of the of the 2D spinoidal decomposition, as described in point 7. The space is initiated by the equation 9, and the process undergoes the transformations presented by the equation set No. 10. The combination of Python, FiPy and the PF problem definition results in a resource-demanding, yet relatively efficient process. The recent workstation-class multicore CPU is sufficient for obtaining the results in a timely manner, as long as a sufficient amount of memory is provided (in this case, 16GB flat, non-paged area of memory via 1.(3) GHz bus). Due to the nature of the process (energy functional minimization), the speed of the simulation diminishes with time. That's why it is recommended to dump the result map to the graphics not after fixed step, but rather after prolonged periods. 800 time steps take 7 hours to complete on the aforementioned hardware. The best way to interpret to specifics of the process is to consider the nature of the initial noise and compare the number of colors as well as peaking values.

Figure 1 shows a big number of independent colors, with a steadily decreasing descent. That signifies the Gaussian distribution generally maintained after 30 time steps. Almost 1300 colors are also a strong hint towards the noise distribution, as well as a seemingly random set of most common colo occurrence. That means that the decomposition did not yet occur, but the energy-driven processes already become to get visible.

Figure 2 shows a sharp just in the color composition, drastically reducing the span of the palette. After multiple time steps (figure 3), color profile suggests a small set of order parameter values, oscillating around the edges of the set, which signifies the decomposition took place, which was the purpose of the model. The processed area is depicted on figure 4.

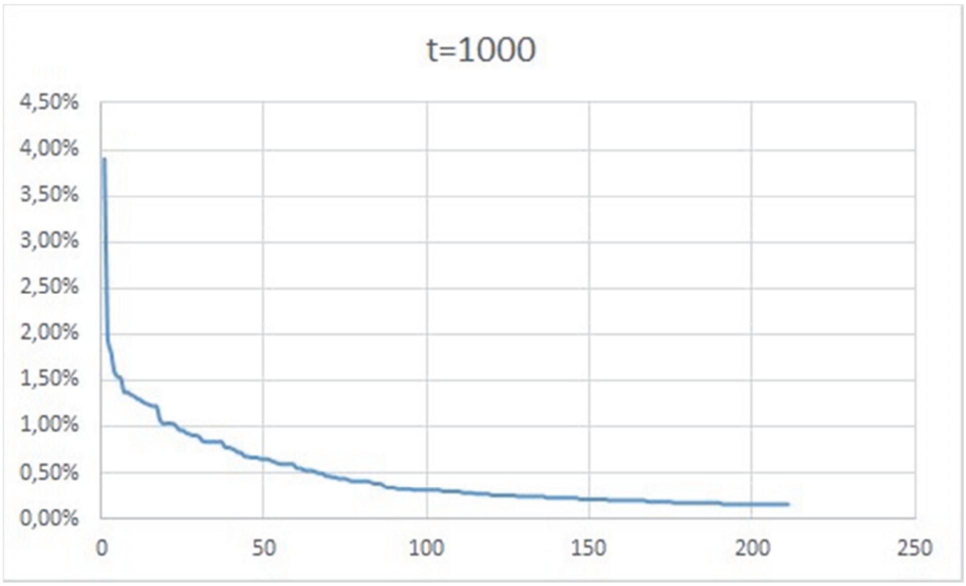

Fig. 3. *Strong-end value of the phase parameter after t=1000*

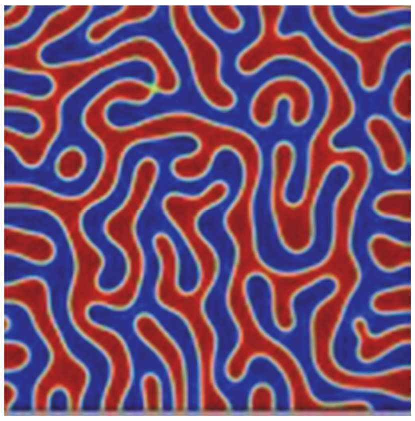

Fig. 4. *The expected state of area after stabilization.*

10. SOFTWARE EXPERIENCE DISCUSSION

Pinpointing the areas of possible improvements in the current solutions for the phase field simulations is not a trivial task. In order to identify the most glaring issues, first the most obvious statements need to be gotten out of the way.

– The straightforward one-to-one performance comparison cannot be formulated using reference problems. Some suites only operate on hard-coded equations, others provide a vast array of parallelization and optimization switches, while the rest would require rebuilding from scratch if the performance was to be taken into account

– The simulations are computationally-demanding, yet can undergo a reasonable distributed/parallel enhancements. Available FEM/FVM/FFT solvers offer the programming environment that facilitates defining the PF problem. Combined with popular scientific libraries, they offer the best realistically-accessible interface for simulating PF.

– Optimizations and stability are not always available in tools that are still under active development (OpenPhase) or try to tackle the problem using innovative approach (PyMKS). There is no clear way of deciding whether the particular problem favors alternate approaches over “direct” methods.

– All of the considered programming environments offer the ways of defining the PF problem under the unsatisfactory usability conditions. This is not caused by those suites being “merely” the C libraries/headers/includes, but rather by a initially surprising complexity in defining the phase field problem. Although all of them share the same rules about describing the system and use the notion of the order parameter transformation $\frac{\partial \phi}{\partial t}$, there are vast differences in declaring the free energy function f, boundary conditions, initializing the space and estimating the thresholds. Depending on the problem, radically different-looking equations (as other forms of energy play the role in the process) need to be introduced, often paired with the analytical boundaries etc. Moelans et al. (2007) offer a step-by-step procedure for constructing the free energy terms and finding them in the calculated phase diagrams or substitutional solutions. Provatas and Elder (2005) explain the mechanisms behind the chemical potential as well as lectures on the solidification and diffusion mechanics, with the essential mathematical footnotes.

The aforementioned problem with a relatively vague (or rather deliberately expanded) method of describing the PF process may look like a first – and easiest – field of improvement. It its then worth noting that all of the mentioned pieces of software aim at achieving exactly that. There is no clear path of improving the PF notations. Additional layers of abstraction might grow out of their scope (and in consequence, usefulness). Maintaining the unified symbols is undoubtedly helpful in the editorial stage (pre-code), as the literature exhibits a curious carelessness between particular notations, especially as the time goes (i.e. compare Elliot (1989) and Moelans et al. (2007)), but is less helpful later. It is therefore essential to remember that the current PF description possibilities are complex, functional and sufficient. The task of validating the PF/CH solvers has been performed few times (Kalidindi, 2012), some of the reviews are incomplete or lead to "dead-ends": tools created for a specific purpose and then abandoned or empty/pulled projects (Pimpalgaonkar, 2011; Wheeler & Boettinger, 1992). Creating a new base platform for defining (not to mention computing) the PF problems will probably not develop above the notion of "reinventing", yet there is without doubt place for utilizing the interconnecting auxiliary code ("glue code") to make the PF equations more straightforward. These best practices guidelines are developed as the work continues.

11. FURTHER PROCEEDINGS

The process of formulating the PF problem and the description of available solver ecosystems is the effect of the first preparation to introducing the phase field modeling as yet another simulation scope at the alma mater faculty. It requires the evaluation of the current solutions with regards to connecting them to the software already created, like the multiscale models. A possible interesting way of introducing PF to the current proceedings is seen in the PyMKS suite and the friendliness of the FiPy toolkit (along with the predictable finite volume method-based solver underneath) might facilitate the development of the more advanced models. The current assessment is conducted over the simulation of the microstructure evolution of the ferrite-pearlite steel with the PF method, as suggested by Yamanaka and Takaki (2007). The aim is not to recreate the process programmatically, but to deploy the model by implementing the proposed equations in the reviewed software. Working towards defining the intermediate PF definition language and model creation best practices is an entry point for creating new models in line with the current projects at the Faculty.

REFERENCES

Baldan, A., 2002, *Review: Progress in Ostwald ripening theories and their applications to nickel-base superalloys,Part I: Ostwald ripening theories*, Department of Metallurgical and Materials Engineering, Mersin University.

Bhadeshia, H. K. D. H., 2004, *Materials Science and Metallurgy*, MP6, L15.

Boettinger, W. J., Warren, J. A., 2014, *NIST Materials Science and Engineering Laboratory*.

Cheng, M., Rutenberg, A. D., 2005, Maximally fast coarsening algorithms, *Physical Reviev E*, 72, 055701R.

Elliott, C. M., 1989, The Cahn-Hilliard model for the kinetics of phase separation, Mathematical Models for Phase Change Problems, *IS of Numerical Mathematics*, 88, 2-3.

Gururajan, M. P., 2005, *Phase field modelling of microstructural evolution using the Cahn-Hilliard equation: A report to accompany CH-muSE.*

Kalidindi, S. R., 2012, Computationally-Efficient Fully-Coupled Multi-Scale Modeling of Materials Phenomena Using Calibrated Localization Linkages, *ISRN Materials Science*, 2012, 305692.

Man, E., Kleijn, C. R., 2014, *Numreical Studies on Phase Field Diffusion and Flow Solvers*, Resources of the Transport Phenomena Research Group, Delft University of Technology, Faculty of Applied Sciences.

Mœlans, N., Blanpain, B., Wollants, P., 2007, An introduction to phase-field modeling of microstructure evolution, *Computer Coupling of Phase Diagrams and Thermochemistry,* 32, 268-294.

Pimpalgaonkar, H., 2011, Phase Field FOAM.

Provatas, N., Elder, K., 2005, *Phase-Field Methods in Material Science*, Wiley-VCH.

Rapaport, D. C., 2002, *The Art of Molecular Dynamics Simulation, Numerical Recipes: The Art of Scientific Computing*, Cambridge University Press.

Wheeler, A. A., Boettinger, W. J., 1992, Phase-field model of solute trapping during solidification, *Physical Review E*, 47, 3, 1893-1909.

Yamanaka, A., Takaki, T., 2007, Coupled simulation of microstructural formation and deformation behavior of ferrite–pearlite steel by phase-field method and homogenization method, *Materials Science and Engineering A*, 480, 1-2, 244-252.

ZASTOSOWANIE I WYDAJNOŚĆ MODELI OPARTYCH O METODĘ POLA FAZOWEGO Z WYKORZYSTANIEM RÓWNANIA CAHNA-HILLIARDA DLA ROZTWORÓW BINARNYCH

Streszczenie

Wśród technik symulacji rozwoju mikrostruktury, coraz większą popularność zdobywa metoda pola fazowego (phase-field method, PFM). Metoda zakłada płynną zmianę faz w miejsce przejścia ostrego. Główną jej zaletą jest względna prostota opisu układu z wykorzystaniem sprzężonych równań różniczkowych, które podlegają typowej dyskretyzacji za pomocą powszechnie stosowanych metod numerycznych. Opis układu za pomocą ciągłych pól pozwala zastosować metodę PFM w złożonych problemach. Niniejsza praca przedstawia analizę możliwości implementacji modeli opartych na metodzie pola fazowego i potencjalne problemy związane z konstruowaniem modelu oraz omawia wzorcową implementację opartą o wybrane zestawy oprogramowania. Przegląd dostępnych narzędzi programistycznych naświetla niedociągnięcia obecnych rozwiązań, potencjalne mozliwości wzrostu wydajności oraz, co nie mniej istotne, zmierza do ułatwienia sposobu, w jakim wyrażane jest zagadnienie PFM. Problemem wejściowym w pracy jest proces reprezentatywnego rozkładu roztworu binarnego wedle równania typu Cahn-Hilliard. Minimalizacja energii swobodnej w układzie jest siłą pędną separacji układu na dwie domeny złożone jednorodnie z czystych składników. Zjawisko zachodzi na podstawie˙ równania CH. Zaproponowano zbiory do zaprogramowania istniejących solverów numerycznych oraz gotowy pakiet obliczeniowy framework, o nazwie OpenPhase. O ile dla podstawowych zagadnień obliczenia z wykorzystaniem pakietu przebiegają prawidłowo, to jest on często niewystarczająco elastyczny dla procesu budowania bardziej złożonych modeli. Zachodzi więc konieczność rozpoznania technik stosowanych w ww. oprogramowaniu i ich analiza pod kątem wydajności i zastosowania w innych, niż referencyjne, przypadkach, celem budowy nowych modeli. Przeglądowy charakter owego zadania unaocznia skromny zasób gotowych rozwiązań, szczątkowe możliwości przeprogramowania (wynikające z dostosowania do konkretnych przypadków) oraz poważne trudności w tworzeniu narzędzi o bardziej uniwersalnym spektrum zastosowań, czy nawet bardziej złożonych modeli. Analiza równania Cahna-Hilliarda dla zachowanego parametru metody pola fazowego oraz przegląd dostępnych rozwiązań punktem wyjścia do dalszej pracy, której celem jest stworzenie modelu PFM pozwalającego badać tworzenie struktur ferryt/perlit w sposób zbliżony do symulacji wedle modelu proponowanego przez Uniwersytet Kobe (Boettinger & Warren, 2014), wykorzystującego PFM, metodę elementów skończonych w ramach teorii homogenizacji (w formie FEH, Finite Element Homogenization).

Received: February 24, 2016
Received in a revised form: June 26, 2016
Accepted: August 17, 2016