Procedia Computer Science

Volume 29, 2014, Pages 980–990

ICCS 2014. 14th International Conference on Computational Science

Modeling phase-transitions using a high-performance, Isogeometric Analysis framework

Philippe Vignal[1], Lisandro Dalcin[2], Nathan.O Collier[3] , and Victor.M. Calo[4]

[1] Center for Numerical Porous Media (NumPor),
Materials Science and Engineering (MSE),
King Abdullah University of Science and Technology, Thuwal, Saudi Arabia
philippe.vignal@kaust.edu.sa

[2] Consejo Nacional de Investigaciones Científicas y Técnicas, Santa Fe, Argentina
Center for Numerical Porous Media (NumPor),
King Abdullah University of Science and Technology, Thuwal, Saudi Arabia
dalcinl@gmail.com

[3] Center for Numerical Porous Media (NumPor),
King Abdullah University of Science and Technology, Thuwal, Saudi Arabia
nathaniel.collier@gmail.com

[4] Center for Numerical Porous Media (NumPor),
Applied Mathematics and Computational Science (AMCS),
Earth Sciences and Engineering (ErSE),
King Abdullah University of Science and Technology, Thuwal, Saudi Arabia
victor.calo@kaust.edu.sa

Abstract

In this paper, we present a high-performance framework for solving partial differential equations using Isogeometric Analysis, called PetIGA, and show how it can be used to solve phase-field problems. We specifically chose the Cahn-Hilliard equation, and the phase-field crystal equation as test cases. These two models allow us to highlight some of the main advantages that we have access to while using PetIGA for scientific computing.

Keywords: phase-fields, Cahn-Hilliard equation, phase-field crystal equation, isogeometric analysis

1 Introduction

Increasingly and steadily, scientific progress is being made thanks to advanced computing capabilities, and an improved understanding of physical problems. If the proper model is used for the problem under consideration, one can many times get through simulation reliable and accurate predictions of properties that are not easily accessible or in the worst cases, not measurable, in a laboratory setting. Nonetheless, coming up with these models is not a trivial task.

 Selection and peer-review under responsibility of the Scientific Programme Committee of ICCS 2014
© The Authors. Published by Elsevier B.V.
doi:10.1016/j.procs.2014.05.088

Along these lines, one of the areas that has recently garnered more attention in computational materials science is phase-field modelling [18], with the goal of representing phase-transitions. These models rely on the mathematical definition of a free energy functional that characterises a system, and accounts for both the components that constitute the system, as well as the interaction between them. A wide variety of phase transitions have been successfully modelled with them [18, 19].

Models usually take the form of nonlinear, partial differential equations in the phase-field framework, and they can have numerical and physical issues that need to be addressed [20]. When trying to solve problems involving phase transitions, the complications usually arise in the interfacial area, where a correct solution might require a very high spatial and/or temporal resolution. It is in such cases where efficient computational resources are needed, that take advantage of efficient libraries for scientific computing using state of the art implementations and technologies. This was the idea behind the inception of PetIGA [6, 8], an open source library that built the Isogeometric Analysis capability on top of PETSc (Portable, Extensible Toolkit for Scientific Computation). The choice to base it on Isogeometric Analysis, a finite element method (FEM) with NURBS basis functions with some improved capabilities over traditional FEM, comes from the fact that this basis is well suited to solve higher-order partial differential equations [7], as the NURBS-based spaces can be constructed to possess an arbitrary degree of inter-element continuity. This gives a lot of flexibility in terms of the choice made for the spatial discretization of a particular partial differential equation when compared to classical finite elements [15].

PetIGA was built over PETSc to make use of a wide array of tools already available for the solution of partial differential equations, namely time-stepping algorithms and solvers (both linear and nonlinear). Along with them, PETSc also has built-in monitors that can be used to speed up the prototyping process when one is trying to solve a new equation, which allow to quickly identify the real sources of error. In this work, we start by giving a brief tutorial on some of the capabilities PetIGA has, and follow by discussing the solution of two phase-field models that have been extensively researched the past few years: the Cahn-Hilliard equation (CH), and the phase-field crystal equation (PFC).

2 PetIGA.

PetIGA is a software framework that implements a NURBS-based finite element method, popularly known as isogeometric analysis (IGA) [7]. By being built on top of PETSc, one gets access to a useful collection of algorithms and data structures for the solution of partial differential equations (PDEs). PETSc was written so as to be applied to a wide range of problem sizes. It has successfully been used in large-scale simulations and shown to have good scaling results [2, 3]. This is why it was a sensible choice as a platform to develop PetIGA.

2.1 Isogeometric analysis basics.

IGA was proposed initially in [16]. It was motivated by the desire to find a technique for solving PDEs which would simplify the problem of converting geometric descriptions for discretizations in the engineering design process. After creating a computer aided design (CAD), the process of converting the representation to a form that is suitable for analysis is considered to be the main bottleneck of the engineering analysis process, consuming up to 80% of the time [7]. By using the NURBS basis directly, isogeometric methods avoid the time-consuming step of

generating the intermediate geometry by using the CAD representations, as NURBS are capable of representing complex geometries more precisely.

Added to this, IGA introduced a new level of refinement, on top of the common h-refinement (i.e. increasing the number of elements in a mesh) and p-refinement (i.e. increasing the polynomial order of the basis function), known as k-refinement, which allows to arbitrarily increase the continuity across elements and can thus be used in discretizations that require C^{p-1}-continuity, where p represents the polynomial order. Work has been done on these higher-order continuous basis functions and it has been observed, both numerically and theoretically [1, 11], that they possess better accuracy per degree of freedom than standard C^0 basis functions. The development of efficient, scalable solvers is also important though, as the weak form discretizations with the higher-order continuous basis functions result in linear systems that are expensive to solve [4, 5].

2.2 The philosophy behind PetIGA

By extending PETSc, that had the idea of parallelism embedded in its implementation, we got direct access to different modular components available for the solution of PDEs, such as direct and iterative solvers, different types of preconditioners and explicit and implicit time-stepping schemes. PetIGA was implemented with the idea in mind that a user, willing to solve a nonlinear and time-dependent partial differential equation in a finite element setting, should only need to know how to get the variational formulation of the equation he was trying to solve. Once this was achieved, he would have access to a framework that allowed him to transfer his code to a supercomputer without any additional work, allowing him to speed up his solution process through the use of more processors. We did not want a user to have to deal with MPI, or have to link codes to well known routines that are already implemented and highly optimized, but instead focus time on actually solving equations, and gaining a better understanding of them through simulation. The goal was achieved with PetIGA, a library where the job of the user is that of assembling the vectors and matrices stemming from the discretization.

The software takes care of assembling the local contributions to a globally distributed data structure in distributed memory architectures, many times a complicated task to complete. This was achieved in big part thanks to PETSc, which provides the support to do so. It also provides data structures called *distributed arrays* (DAs) for managing data, for methods whose discretization is based on a structured grid topology. This particular component optimises the partition of the computational domain among the different processors used, and is the main reason why we developed the IGA object. This object leads to a nearly optimal partition of the degrees of freedom of the domain for a single patch of NURBS-based isogeometric finite elements by trying to minimise the load imbalance among processors. More information on how the actual assembly of the different matrices and vectors is done can be found in [6], and tutorials for the software are being developed. The object-oriented setting of PetIGA hides the details of parallelism and problem assembly, such that the actual application codes are short and easy to read. The user only needs to provide the integrand of the weak form.

3 High-order phase-field models: the Cahn-Hilliard (CH) and the phase-field crystal (PFC) equations.

Phase-field models have received a lot of attention from many different fields, being a truly interdisciplinary area with problems related to numerical analysis, physics and thermodynam-

ics [18]. In this section we give a brief review on the basic structure of phase-field frameworks, as well as some background on the Cahn-Hilliard (CH) and phase-field crystal equations (PFC).

3.1 Phase-field modelling

In these models, a phase-field parameter ϕ, continuous over the whole space, is used to represent the different phases. The physical behaviour of this phase-field is then usually defined through the minimisation of a free energy functional $\mathcal{F}[\phi(\mathbf{r})]$ that characterises the system, defined as

$$\mathcal{F}[\phi(\mathbf{r})] = \int_{\Omega} \left\{ g(\phi) + \alpha \left|\nabla\phi\right|^2 + \beta \left(\Delta\phi\right)^2 \right\} d\mathbf{V}, \tag{1}$$

where $g(\phi)$ is used to represent the bulk free energy and can take either the form of a logarithmic function (as in CH in this work) or a polynomial (as in PFC), the parameters α and β are constants that take values between -1 and 1, and the gradient and Laplacian terms model the physics of the interfacial effects. This framework allows for an implicit tracking of the interface, compared to sharp-interface models, and leads to less stiff systems where arbitrary rules do not have to be used to track the interface [18]. To set the PDE, a variational derivative of the selected free energy is taken, $\dfrac{\delta\mathcal{F}[\phi(\mathbf{r})]}{\delta\phi}$, which is then coupled to the evolution of ϕ in time, such that

$$\frac{\partial\phi}{\partial t} = -\nabla^a \cdot \left(-M(\phi)\nabla^a \frac{\delta\mathcal{F}[\phi(\mathbf{r})]}{\delta\phi} \right) \tag{2}$$

where $M(\phi)$ represents a mobility for the component(s) being considered, a takes a value of one if the phase-field represents a conserved quantity (i.e. density) or zero otherwise, and t stands for time. We now proceed to define the partial differential equations considered in this work.

3.2 The Cahn-Hilliard equation.

The CH equation is a fourth-order, nonlinear and time-dependent, partial differential equation. In this work, we consider the dimensionless version, adapted from [12], where the free energy $\mathcal{F}_{CH}$ is defined as

$$\mathcal{F}_{CH} = \int_{\Omega} \left((c)\log(c) + (1-c)\log(1-c) + 2\theta c(1-c) + \frac{\theta}{3\gamma}|\nabla c|^2 \right) d\mathbf{V}, \tag{3}$$

where the phase-field parameter c represents concentration, γ is a dimensionless number related to the thickness of the interfaces and is given a value of 3000, and θ, equal to 3/2, is a dimensionless number that represents a critical transition ratio. The CH equation has successfully been used to model spinodal decomposition, a physical mechanism in which a mixture phase-separates when θ takes a value greater than one. Considering that the concentration is a conserved quantity, the PDE becomes

$$\begin{aligned} \frac{\partial c}{\partial t} &= \nabla \cdot \left(M_c \nabla \frac{\delta\mathcal{F}_{\mathcal{CH}}}{\delta c} \right) \\ &= \nabla \cdot \left(M_c \nabla \left(\mu_c - \Delta c \right) \right) \end{aligned} \tag{4}$$

where μ_c represents a chemical potential defined as

$$\mu_c = \frac{1}{2\theta}\log\frac{c}{1-c} + 1 - 2c.$$

Finally, the strong form of this equation can be expressed as

$$\begin{cases} \dfrac{\partial c}{\partial t} - \nabla \cdot \left(M_c \nabla \left(\mu_c - \Delta c\right)\right) = 0 & \text{on } \Omega \times]0,T] \\ c = c_0 & \text{on } \Omega \times \{t=0\} \\ M_c \nabla \left(\mu_c - \Delta c\right) \cdot n = s & \text{on } \Gamma_s \times]0,T] \\ M_c \lambda \nabla c \cdot \boldsymbol{n} = 0 & \text{on } \Gamma \times]0,T] \\ c = g & \text{on } \Gamma_D \times]0,T] \end{cases} \tag{5}$$

To get the weak form for this equation and be able to implement the residual in PetIGA, we let $\mathcal{V}$ denote the trial and weighting function spaces, and consider periodic boundary conditions in all directions. The variational formulation, obtained by multiplying the strong form by a test function w and integrating by parts, is stated as: find $c \in \mathcal{V}$ such that $\forall w \in \mathcal{V}$,

$$\left(w, \dot{c}\right)_\Omega + \left(\nabla w, M_c \nabla \mu_c + \nabla M_c \Delta c\right)_\Omega + \left(\Delta w, M_c \Delta c\right)_\Omega = 0, \tag{6}$$

where $\dot{c} = \dfrac{\partial c}{\partial t}$, $\left(.,.\right)_\Omega$ represents the $\mathcal{L}^2$ inner product over the domain Ω and $\mathcal{V}$ needs to be $\mathcal{H}^2$-conforming. We use the Galerkin method (although collocation is also available in PetIGA) to discretize the infinite dimensional problem and derive the semidiscrete formulation which is then stated as: find $c^h \in \mathcal{V}^h \subset \mathcal{V}$ such that $\forall w^h \in \mathcal{V}^h \subset \mathcal{V}$ such that

$$\left(w^h, \dot{c}^h\right)_\Omega + \left(\nabla w^h, M_c^h \nabla \mu_c^h + \nabla M_c^h \Delta c^h\right)_\Omega + \left(\Delta w^h, M_c^h \Delta c^h\right)_\Omega = 0. \tag{7}$$

We suppose that the discrete space $\mathcal{V}^h$ is spanned by the linear combination of basis functions N_A, which are $\mathcal{C}^1$-continuous B-spline basis functions that are associated to the global degree of freedom A, that runs from 1 to n_d, where n_d represents the dimension of the discrete space. With regards to PetIGA, if one is able to get equation (7) and an initial condition, testing of the residual can already be done to check if the system converges to a solution, or compare the results to a benchmark problem. This can save time while prototyping a code or model. Being a nonlinear time-dependent problem, the CH model requires the use of a Jacobian if one sets the problem up with a Newton-type scheme. Another advantage of PETSc is that it is able to approximate the Jacobian through finite-differencing. The user does not even need to code it to start getting results. With regards to the time-discretization, we employ the adaptive scheme from [12], which uses the generalised-α method.

3.3 The phase-field crystal equation

The PFC equation is a sixth-order, nonlinear time-dependent partial differential equation. Even though it was initially developed to study problems related to solidification [9] at atomic length scales and diffusive time scales, it has since then been used to tackle issues in crack propagation, dislocation dynamics, and formation of foams [10].

In the equation, the order parameter ϕ represents an atomistic density field, which is periodic in the solid state and uniform in the liquid one. The free energy functional for the phase-field crystal equation in its dimensionless form is given by [9, 17]

$$\mathcal{F}_{PFC} = \int_\Omega \left[\Psi(\phi) + \frac{1}{2}\left(\phi^2 - 2|\nabla\phi|^2 + (\Delta\phi)^2\right)\right] d\mathbf{V}, \tag{8}$$

where $\Psi(\phi) = \frac{1}{4}\phi^4 - \frac{\epsilon}{2}\phi^2$. Notice that the sign in front of the gradient term $|\nabla\phi|$ is negative in this case, whereas it is positive in the case of CH. This has physical implications in the

behaviour of the equations, as in CH, interfaces (i.e. where the gradients are present) will be energetically penalised and so the equation will tend to minimise them, whereas in PFC they will be energetically favoured because of their negative contribution to the free energy. To get the PDE, we apply the same procedure as before (again with $a = 1$ as density is a conserved quantity) and get

$$\begin{aligned}\frac{\partial \phi}{\partial t} &= \nabla \cdot \left(M_\phi \nabla \frac{\delta \mathcal{F}_{PFC}}{\delta \phi} \right) \\ &= \Delta \left((1+\Delta)^2 \phi + \Psi'(\phi) \right)\end{aligned} \tag{9}$$

where the mobility M_ϕ is taken to be equal to one, and $\Psi'(\phi) = \phi^3 - \epsilon\phi$. By considering the boundary conditions, the strong form of the problem can then be stated as follows: over the spatial domain Ω and the time interval $]0, T[$, given $\phi_0 : \bar{\Omega} \longmapsto R$, find $\phi : \bar{\Omega} \times [0, T] \longmapsto R$ such that

$$\begin{cases} \dfrac{\partial \phi}{\partial t} = \Delta \left[(1+\Delta)^2 \phi - \epsilon\phi + \phi^3 \right] & \text{on } \Omega \times]0, T] \\ \nabla \left((1+\Delta)^2 \phi - \epsilon\phi + \phi^3 \right) \cdot \mathbf{n} = 0 & \text{on } \Omega \times]0, T] \\ \nabla(2\phi + \Delta\phi) \cdot \mathbf{n} = 0 & \text{on } \Omega \times]0, T] \\ \nabla\phi \cdot \mathbf{n} = 0 & \text{on } \Omega \times]0, T] \\ \phi(\mathbf{x}, 0) = \phi_0(\mathbf{x}) & \text{on } \bar{\Omega} \end{cases} \tag{10}$$

The following splitting of the equation is proposed

$$\frac{\partial \phi}{\partial t} = \Delta\sigma, \tag{11a}$$

$$\sigma = (1+\Delta)^2 \phi + \Psi'(\phi) \tag{11b}$$

where $\Psi'(\phi) = -\epsilon\phi + \phi^3$. We chose this formulation to compare results with the formulation presented in [21]. The functional space $\mathcal{V} \in \mathcal{H}^2$ is defined, where $\mathcal{H}^2$ corresponds to the Sobolev space of square integrable functions with square integrable first and second derivatives. A weak form of equations (11a) and (11b) is derived by multiplying them by test functions $q \in \mathcal{H}^2$ and $w \in \mathcal{H}^2$, respectively, and integrating by parts. Assuming periodic boundary conditions in every direction, one can state the problem as: find $\phi \in \mathcal{V}$ and $\sigma \in \mathcal{V}$ such that for all $q \in \mathcal{V}$ and $w \in \mathcal{V}$

$$\left(q, \dot{\phi} \right)_\Omega + (\nabla q, \nabla\sigma)_\Omega = 0, \tag{12a}$$

$$(w, \sigma)_\Omega - (w, \Psi'(\phi) + \phi)_\Omega + (\nabla w, 2\nabla\phi)_\Omega - (\Delta w, \Delta\phi)_\Omega = 0. \tag{12b}$$

To derive a finite element approximation to the problem, we pick the finite dimensional space $\mathcal{V}^h \subset \mathcal{V}$ and derive a semi discrete formulation. The problem is then to find ϕ^h, $\sigma^h \in \mathcal{V}^h$ such that for all q^h, $w^h \in \mathcal{V}^h$

$$\left(q^h, \dot{\phi}^h \right)_\Omega + \left(\nabla q^h, \nabla\sigma^h \right)_\Omega = 0, \tag{13a}$$

$$\left(w^h, \sigma^h \right)_\Omega - \left(w^h, \Psi'(\phi^h) + \phi^h \right)_\Omega + \left(\nabla w^h, 2\nabla\phi^h \right)_\Omega - \left(\Delta w^h, \Delta\phi^h \right)_\Omega = 0. \tag{13b}$$

We again suppose that the discrete space $\mathcal{V}^h$ is spanned by the linear combination of basis functions N_A, which are $\mathcal{C}^1$-continuous B-spline basis functions that are associated to the global degree of freedom A, that runs from 1 to n_d, where n_d represents the dimension of the discrete space.

With regards to the implementation, we let the global vectors of degrees of freedom associated to ϕ_n^h and σ_n^h be $\mathbf{\Phi}_n$ and $\mathbf{\Sigma}_n$, respectively. The residual vectors for this formulation are then given by

$$\mathbf{R}^{\phi}(\mathbf{\Phi}_n, \mathbf{\Phi}_{n+1}, \mathbf{\Sigma}_{n+1}); \quad \mathbf{R}^{\phi} = \{R_A^{\phi}\}; \quad A = 1, ..., n_b, \tag{14}$$

$$\mathbf{R}^{\sigma}(\mathbf{\Phi}_n, \mathbf{\Phi}_{n+1}, \mathbf{\Sigma}_{n+1}); \quad \mathbf{R}^{\sigma} = \{R_A^{\sigma}\}; \quad A = 1, ..., n_b, \tag{15}$$

where $R_A^{\phi} = 0$ and $R_A^{\sigma} = 0$ are defined as

$$\left(q^h, \frac{[[\phi_n^h]]}{\Delta t}\right)_{\Omega} + \left(\nabla q^h, \nabla \sigma^h\right)_{\Omega} = R_A^{\phi}, \tag{16}$$

$$\begin{aligned}&\left(w^h, \sigma^h\right)_{\Omega} - \left(w^h, \frac{1}{2}\Psi'(\phi_{n+1}^h + \phi_n^h)\right)_{\Omega} \\ &- \left(w^h, \phi_{n+1/2}^h\right)_{\Omega} + 2\left(\nabla w^h, \nabla \phi_{n+1/2}^h\right)_{\Omega} - \left(\Delta w^h, \Delta \phi_{n+1/2}^h\right)_{\Omega} = R_A^{\sigma}.\end{aligned} \tag{17}$$

where $[[\phi_n^h]] = \phi_{n+1}^h - \phi_n^h$ and $\phi_{n+1/2}^h = \frac{1}{2}\left(\phi_{n+1}^h + \phi_n^h\right)$. A system of nonlinear equations for $\mathbf{\Phi}_{n+1}$ and $\mathbf{\Sigma}_{n+1}$ is obtained by having the residual vectors be equal to zero, and solved using Newton's method. By having $\mathbf{\Phi}_{n+1,(i)}$ and $\mathbf{\Sigma}_{n+1,(i)}$ be the *ith* iteration of Newton's algorithm, the iterative procedure was defined as in [13]: taking $\mathbf{\Phi}_{n+1,(0)} = \mathbf{\Phi}_n$ and $\mathbf{\Sigma}_{n+1,(0)} = \mathbf{\Sigma}_n$. Then, for $i = 1, .., \mathrm{i}_{max}$,

(1) Compute residuals using $\mathbf{\Phi}_{n+1,(i)}$, $\mathbf{\Sigma}_{n+1,(i)}$, denoted as $\mathbf{R}_{(i)}^{\phi}$ and $\mathbf{R}_{(i)}^{\sigma}$, respectively.

(2) Compute the tangent matrix $\mathbf{K}_i$ using the *ith* iterates. This matrix is given by

$$\mathbf{K}_{(i)} = \begin{pmatrix} \mathbf{K}_{(i)}^{\phi\phi} & \mathbf{K}_{(i)}^{\phi\sigma} \\ \mathbf{K}_{(i)}^{\sigma\phi} & \mathbf{K}_{(i)}^{\sigma\sigma} \end{pmatrix}.$$

(3) Solve the linear system

$$\begin{pmatrix} \mathbf{K}_{(i)}^{\phi\phi} & \mathbf{K}_{(i)}^{\phi\sigma} \\ \mathbf{K}_{(i)}^{\sigma\phi} & \mathbf{K}_{(i)}^{\sigma\sigma} \end{pmatrix} \begin{pmatrix} \delta\mathbf{\Phi}_{n+1,(i)} \\ \delta\mathbf{\Sigma}_{n+1,(i)} \end{pmatrix} = - \begin{pmatrix} \mathbf{R}_{(i)}^{\phi} \\ \mathbf{R}_{(i)}^{\sigma} \end{pmatrix}$$

(4) Update the solution such that

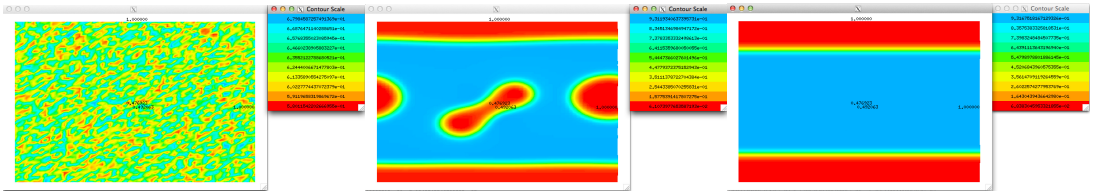
$$\begin{pmatrix} \mathbf{\Phi}_{n+1,(i+1)} \\ \mathbf{\Sigma}_{n+1,(i+1)} \end{pmatrix} = \begin{pmatrix} \mathbf{\Phi}_{n+1,(i)} \\ \mathbf{\Sigma}_{n+1,(i)} \end{pmatrix} + \begin{pmatrix} \delta\mathbf{\Phi}_{(i+1)} \\ \delta\mathbf{\Sigma}_{(i+1)} \end{pmatrix}$$

This algorithm needs to be repeated until the norm of both residual vectors is reduced to a given tolerance of their initial value.

We found it interesting to analyze the behavior of the free energy with different implicit time marching schemes. And herein lies one of the other advantages of PETSc: the code to solve the equation using Backward Euler, the theta method, or generalized-α is exactly the same, and does not need to be recompiled if one wishes to switch between schemes. An added advantage, is the fact that options can be directly specified as command line arguments.

4 Numerical Results.

In this section we present some numerical results for both equations, showing how PetIGA can be used to visualise the solution in real time, as well as some of the features that make it a



(a) $t = 0$ (b) $t = 7.78 * 10^{-4}$ (c) $t = 0.1$

Figure 1: Spinodal decomposition. Snapshots of the numerical approximation to the concentration at different time steps. The computational mesh is composed of $64^2 C^1$ quadratic elements.

```
  Testing hand-coded Jacobian, if the ratio is O(1.e-8), the hand-coded Jacobian is probably correct.
  Run with -snes_check_jacobian_view [viewer][:filename][:format] to show difference of hand-coded and finite difference Jacobian.
6.08549e-10 = ||J - Jfd||//J|| 0.0703301  = ||J - Jfd||
```

Figure 2: Output for the **-snes_check_jacobian** argument. The hand-coded jacobian is compared to PETSc's finite difference approximation.

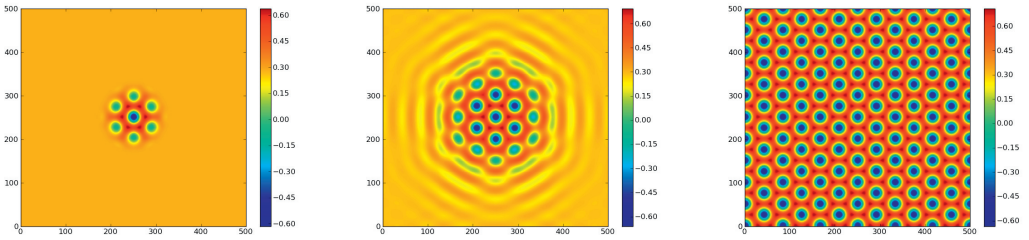
useful tool in solving high-order, partial differential equations. The examples that are dealt with are taken from [12] for CH, and from [20] for PFC.

4.1 Spinodal decomposition using the Cahn-Hilliard equation.

The initial condition was set to be

$$c(t = 0, \mathbf{x}) = 0.63 + \hat{c}, \tag{18}$$

over the whole domain (the unit square), with $\hat{c}$ a uniform random variable with values contained in the interval $[-0.05, 0.05]$. As the value of θ lies above one, we can see the occurrence of a phase separation during the evolution of the concentration in Figure 1, where the command to execute the program looks like this:
mpiexec -n 4 ./CahnHilliard2D -N 64 -p 2 -C 1 -ts_monitor_draw_solution
where **mpiexec -n 4** specifies the number of processors one is using (in this case four), **CahnHilliard2D** is the name of the executable being used, and N, p and C control the number of elements, the polynomial order and the degree of continuity of the basis functions used to solve the problem, respectively. The snapshots shown are taken at different times during the phase separation, and the solution presented in [12] is recovered, which confirms the correct implementation of the algorithm proposed. These snapshots are generated in real-time, as PETSc has 2D visualisation capabilities to make the debugging process easier, and all one needs to do is add $-$**ts_monitor_draw_solution** as an argument in the command line when executing the program. By adding $-$**snes_mf**, one can use the finite-difference calculated jacobian for the residual that was established, or check the explicitly coded jacobian using the option $-$**snes_check_jacobian** and get an output as shown in Figure 2. Tolerances can also be set as command line arguments, both for the linear (ksp) and nonlinear (snes) solvers. By being connected to PETSc, one really gets a large number of very useful tools to monitor and identify problems while trying to solve complicated partial differential equations. All the options available can be shown by adding **-help** as an argument.



(a) $t = 0$ (b) $t = 20$ (c) $t = 150$

Figure 3: Crystal growth in a supercooled liquid. Snapshots of the numerical approximation to the atomistic density field of the phase field crystal equation using Crank-Nicolson. The computational mesh is composed of $190^2 C^1$ quadratic elements. The time step is 1.0.

4.2 Crystal growth using the phase-field crystal equation.

For the following example, we simulate crystal growth in a supercooled liquid. The initial condition is given by

$$\phi_0(\mathbf{x}) = \bar{\phi} + A\left(\cos\left(\frac{qy}{\sqrt{3}}\right)\cos(qx) - 0.5\cos\left(\frac{2q}{\sqrt{3}}y\right)\right) \tag{19}$$

where $\bar{\phi}$ represents the average value of $\phi(\mathbf{x})$, A is the amplitude of the fluctuations in density, and q represents a wavelength related to the lattice constant [18]. These parameters take the values

$$\epsilon = 0.325; \quad \bar{\phi} = \frac{\sqrt{\epsilon}}{2}; \quad q = \frac{\sqrt{3}}{2}; \quad A = \frac{4}{5}\left(\bar{\phi} + \frac{\sqrt{15\epsilon - 36\bar{\phi}^2}}{3}\right).$$

$\bar{\phi}$ can be seen as the value given to the density of the liquid, whereas the second term of equation (19) really sets the structure for the solid. In this case, an initial solid nucleus is placed in the center of a square domain, and is surrounded by liquid of the same material that is below its critical temperature of fusion. This leads to an evolution which sees the solid phase grow as the liquid crystallizes around it. The domain size chosen is $\bar{\Omega} = \left[\frac{20\pi}{q}\right] \times \left[\frac{12\sqrt{3}\pi}{q}\right]$. The initial condition was chosen so as to only have one hexagon of atoms lying in the center of the domain. The evolution of the single crystal is shown in figure 3. The growth of the crystalline phase can be observed, with features being similar to the ones presented in [13, 14]. We also observed the free energy to be non-increasing at each time step.

5 Conclusions.

In this paper, a high-performance implementation of isogeometric analysisis is presented. The software is built in such a way that the user only has to worry about coding the integrand of the weak form and the initial condition. PetIGA is a useful tool for the solution of high-order partial differential equations that arise in phase-field models. The prototyping stage is simplified as compared to other frameworks, as the monitors we get access to through PETSc speed up the

debugging process. The framework allows the user to focus on how to solve a problem instead of how to implement it.

6 Future Work

Given the possibility of generating higher-order continuous spaces, different formulations are possible when solving these equations with IGA. Even though the number of degrees of freedom increases with each split of the equation, that doesn't necessarily translate to longer computational times. Properties such as energy-stability and mass conservation also need to be guaranteed for the numerical scheme used to be valid. Our future work will tackle theses problems. We will analyse the numerical complexity of the splits, as well as develop numerical schemes that possess the aforementioned properties.

6.1 Acknowledgments

This work was supported by NumPor, the centre for Numerical Porous Media.

References

[1] I. Akkerman, Y. Bazilevs, V.M. Calo, T.J.R. Hughes, and S. Hulshoff. The role of continuity in residual-based variational multiscale modeling of turbulence. *Computational Mechanics*, 41:371–378, 2008.

[2] S. Balay, K. Buschelman, V. Eijkhout, W.D. Gropp, D. Kaushik, M.G. Knepley, L. Curfman McInnes, B.F. Smith, and H. Zhang. PETSc users manual. Technical Report ANL-95/11 - Revision 3.0.0, Argonne National Laboratory, 2008.

[3] S. Balay, K. Buschelman, W. D. Gropp, D. Kaushik, M. G. Knepley, L. Curfman McInnes, B. F. Smith, and H. Zhang. PETSc Web page, 2010. http://www.mcs.anl.gov/petsc.

[4] N. Collier, L. Dalcin, D. Pardo, and V.M. Calo. The cost of continuity: performance of iterative solvers on isogeometric finite elements. *SIAM Journal on Scientific Computing*, 35(2):A767–A784, 2013.

[5] N. Collier, D. Pardo, L. Dalcin, M. Paszynski, and V.M. Calo. The cost of continuity: A study of the performance of isogeometric finite elements using direct solvers. *Computer Methods in Applied Mechanics and Engineering*, 213-216(0):353–361, 2012.

[6] N.O. Collier, L. Dalcín, and V.M. Calo. PetIGA: High-performance isogeometric analysis. *arxiv*, abs/1305.4452, 2013. http://arxiv.org/abs/1305.4452.

[7] J.A. Cottrell, T.J.R. Hughes, and Yuri Bazilevs. *Isogeometric Analysis: Toward Unification of CAD and FEA*. John Wiley and Sons, 2009.

[8] L. Dalcin and N. Collier. Petiga: A framework for high performance isogeometric analysis. `https://bitbucket.org/dalcinl/petiga`, last viewed January 2014, 2012.

[9] K.R. Elder, M. Katakowski, M. Haataja, and M. Grant. Modeling elasticity in crystal growth. *Phys. Rev. Lett.*, 88:245705, Jun 2002.

[10] H. Emmerich, H. Löwen, R. Wittkowski, T. Gruhn, G.I. Tóth, G. Tegze, and L. Gránásy. Phase-field-crystal models for condensed matter dynamics on atomic length and diffusive time scales: an overview. *Advances in Physics*, 61(6):665–743, 2012.

[11] J.A. Evans, Y. Bazilevs, I. Babuška, and T.J.R. Hughes. n-widths, sup-infs, and optimality ratios for the k-version of the isogeometric finite element method. *Computer Methods in Applied Mechanics and Engineering*, 198(21-26):1726–1741, 2009.

[12] H. Gómez, V. M. Calo, Y. Bazilevs, and T.J.R. Hughes. Isogeometric analysis of the Cahn-Hilliard phase-field model. *Computer Methods in Applied Mechanics and Engineering*, 197(49-50):4333–4352, 2008.

[13] H. Gómez and X. Nogueira. An unconditionally energy-stable method for the phase field crystal equation. *Computer Methods in Applied Mechanics and Engineering*, 249-252(0):52–61, 2012.

[14] Z. Hu, S.M. Wise, C. Wang, and J.S. Lowengrub. Stable and efficient finite-difference nonlinear-multigrid schemes for the phase field crystal equation. *Journal of Computational Physics*, 228(15):5323 – 5339, 2009.

[15] T.J.R. Hughes. *The finite element method: linear static and dynamic finite element analysis*. Dover Publications, 2000.

[16] T.J.R. Hughes, J.A. Cottrell, and Y. Bazilevs. Isogeometric analysis: CAD, finite elements, NURBS, exact geometry and mesh refinement. *Computer Methods in Applied Mechanics and Engineering*, 194:4135–4195, 2005.

[17] A Jaatinen and T Ala-Nissila. Extended phase diagram of the three-dimensional phase field crystal model. *Journal of Physics: Condensed Matter*, 22(20):205402, 2010.

[18] N. Provatas and K. Elder. *Phase-Field Methods in Materials Science and Engineering*. Wiley-VCH, 1st edition, 2010.

[19] R. Spatschek, C. Müller-Gugenberger, E. Brener, and B. Nestler. Phase field modeling of fracture and stress-induced phase transitions. *Phys. Rev. E*, 75:066111, Jun 2007.

[20] P. Vignal, L. Dalcin, D.L. Brown, N.O. Collier, and V.M. Calo. Energy-stable time discretizations for the phase-field crystal equation. in preparation, 2014.

[21] P.A. Vignal, N. Collier, and V.M. Calo. Phase field modeling using petiga. *Procedia Computer Science*, 18(0):1614 – 1623, 2013. 2013 International Conference on Computational Science.