Commun. Comput. Phys.
doi: 10.4208/cicp.240715.080716a

Vol. **21**, No. 1, pp. 65-92
January 2017

A Robust and Efficient Adaptive Multigrid Solver for the Optimal Control of Phase Field Formulations of Geometric Evolution Laws

Feng Wei Yang[1,*], Chandrasekhar Venkataraman[2], Vanessa Styles[1] and Anotida Madzvamuse[1]

[1] *Department of Mathematics, University of Sussex, UK.*
[2] *School of Mathematics & Statistics, University of St Andrews, UK.*

Received 24 July 2015; Accepted (in revised version) 8 July 2016

Abstract. We propose and investigate a novel solution strategy to efficiently and accurately compute approximate solutions to semilinear optimal control problems, focusing on the optimal control of phase field formulations of geometric evolution laws. The optimal control of geometric evolution laws arises in a number of applications in fields including material science, image processing, tumour growth and cell motility. Despite this, many open problems remain in the analysis and approximation of such problems. In the current work we focus on a phase field formulation of the optimal control problem, hence exploiting the well developed mathematical theory for the optimal control of semilinear parabolic partial differential equations. Approximation of the resulting optimal control problem is computationally challenging, requiring massive amounts of computational time and memory storage. The main focus of this work is to propose, derive, implement and test an efficient solution method for such problems. The solver for the discretised partial differential equations is based upon a geometric multigrid method incorporating advanced techniques to deal with the nonlinearities in the problem and utilising adaptive mesh refinement. An in-house two-grid solution strategy for the forward and adjoint problems, that significantly reduces memory requirements and CPU time, is proposed and investigated computationally. Furthermore, parallelisation as well as an adaptive-step gradient update for the control are employed to further improve efficiency. Along with a detailed description of our proposed solution method together with its implementation we present a number of computational results that demonstrate and evaluate our algorithms with respect to accuracy and efficiency. A highlight of the present work is simulation results on the optimal control of phase field formulations of geometric evolution laws in 3-D which would be computationally infeasible without the solution strategies proposed in the present work.

AMS subject classifications: 15A29, 35K58, 68W10

*Corresponding author. *Email addresses:* F.W.Yang@sussex.ac.uk (F. W. Yang), cv28@st-andrews.ac.uk (C. Venkataraman), V.Styles@sussex.ac.uk (V. Styles), A.Madzvamuse@sussex.ac.uk (A. Madzvamuse)

 ©2016 Global-Science Press

https://doi.org/10.4208/cicp.240715.080716a Published online by Cambridge University Press

Key words: Optimal control, geometric evolution law, phase field, multigrid, parallel, mesh adaptivity, two-grid solution strategy.

1 Introduction

The optimal control of geometric evolution equations or more generally free boundary problems arises in a number of applications. In image processing the tracking of deformable objects may be formulated as the optimal control of a suitably chosen evolution law [1]. A number of applications arise from problems in material science such as the control of nanostructure through electric fields [2,3]. An important and topical application area is the image driven modelling of biological processes, such as tumour growth [4] or cell migration [5], in which parameters (or functions) in a model are estimated from experimental imaging data. In a recent study we proposed an optimal control approach to whole cell tracking [6], i.e., the reconstruction of whole cell morphologies in time from a set of static images, in which the cell tracking problem was formulated as the optimal control of a geometric evolution equation [6]. In general the approximation of such optimal control problems is computationally intensive both in terms of central processing unit (CPU) time and memory. Hence, the development of robust and efficient solvers for such problems with a view to reducing CPU time (or simply wall-clock time) and memory requirements is a worthwhile research direction.

In the current work we consider the optimal control of geometric evolution laws of forced mean curvature flow type. We denote by $\Gamma(t)$, a closed oriented smoothly evolving $d-1$ dimensional hypersurface in $\mathbb{R}^d$, $d=2,3$ with outward pointing unit normal $\boldsymbol{v}$. The motion of $\Gamma(t)$ satisfies a volume constrained mean curvature flow with forcing, i.e., given an initial surface $\Gamma(0)$, the velocity $\boldsymbol{V}$ of Γ is given by

$$\boldsymbol{V}(\boldsymbol{x},t)=(-\sigma H(\boldsymbol{x},t)+\eta(\boldsymbol{x},t)+\lambda_V(t))\boldsymbol{v}(\boldsymbol{x},t),\quad \boldsymbol{x}\in\Gamma(t),\ t\in(0,T], \tag{1.1}$$

where $\sigma>0$ represents the surface tension, H denotes the mean curvature (which we take to be the sum of the principal curvatures) of Γ, η is a space time distributed forcing and λ_V is a spatially uniform Lagrange multiplier enforcing volume constraint. We assume we are given an initial interface Γ^0 and a target interface Γ_{obs} both of which are smooth closed oriented $d-1$ dimensional hypersurfaces.

The optimal control problem, which is the focus of the current work, consists of finding a space time distributed forcing η in (1.1) such that with $\Gamma(0)=\Gamma^0$, the interface position at time T corresponding to the solution of (1.1), $\Gamma(T)$, is "close" to the observed data Γ_{obs}. We have deliberately refrained from stating precisely what is meant by $\Gamma(T)$ being close to Γ_{obs} as in the sharp interface setting it is not obvious what constitutes a good choice of metric to measure the difference between two surfaces. In particular standard measures such as the Haussdorff distance are typically non-smooth and this complicates the approximation of the optimal control problem. Moreover, the theory of optimal control of geometric evolution laws is in its infancy, in fact only recently has progress been

https://doi.org/10.4208/cicp.240715.080716a Published online by Cambridge University Press

made on the optimal control of parabolic equations on evolving surfaces even in the case of prescribed evolution [7]. On the other hand, the theory for the optimal control of semi-linear parabolic equations is more mature (see, for example, [8]) and if one considers a diffuse interface representation of the surfaces then standard measures of distance may be used (e.g., L_2). In light of the above, we consider the phase field approximation of (1.1) given by the volume constrained Allen-Cahn equation, see (2.1). We approximate our initial and target data Γ^0 and Γ_{obs} by diffuse interface representations ϕ_0 and ϕ_{obs} respectively (see for example [5] for details on how to construct such representations).

Our strategy for approximating the solution to the optimal control problem consists of an iterative adjoint based solution method, c.f., Sections 2 and 3. The method is particularly computationally intensive for a number of reasons.

1. The iterative adjoint based solution method we employ necessitates multiple solves of the forward and adjoint problems.

2. As the state equation is of Allen-Cahn type grid adaptivity for the solution of the forward equation is mandatory for large simulations particularly in 3-D.

3. The computed state enters the adjoint equation which is posed backwards in time. Hence the state equation must first be solved over the whole time interval with the computed states stored and then the adjoint equation is solved backwards in time. Thus the algorithm requires large amounts of data storage.

4. We want to consider small values of the interfacial width parameter as many of the applications from cell biology that we are interested in involve interfaces with large curvatures and small scale features which we wish to resolve with our diffuse interface approximation. This imposes strong restrictions on the grid for the solution of the state problem.

In this work we focus on developing a robust and efficient solver for the problem. We employ a fast parallel adaptive multigrid solution method for the forward equation. The use of adaptive grids and parallelisation allows us to compute with relatively small values of the interfacial width parameter. For the adjoint equation we make the observation that as the PDE is linear it may be possible to relax the restrictions on the grid needed for the solution of the state equation. Hence we employ a parallel multigrid solver for the adjoint equation on a uniform grid that is typically coarser than the adaptive grid used for the approximation of the state. We also consider a simple adaptive strategy for our iterative steepest descent based algorithm for the update of the control.

Major findings in the present work include that the two-grid solution strategy we propose in which the state equation is solved on an adaptive grid and the adjoint problem is solved on a coarse uniform grid appears to have only a minor detrimental impact on accuracy whilst the savings in terms of memory and CPU time are considerable as the state is only stored on the coarse grid. We also propose an adaptive algorithm for

https://doi.org/10.4208/cicp.240715.080716a Published online by Cambridge University Press

the iterative update for approximating the optimal control η. We expect that our findings are of relevance beyond the optimal control of Allen-Cahn equations alone and this could include the development of efficient schemes for the optimal control of semilinear parabolic equations in general.

The remainder of this article is structured as follows. In Section 2 we present the optimal control problem and formally derive the optimality conditions used in the algorithm for its approximation. In Section 3 we describe our solver for the optimal control problem which is the main focus in this work. Firstly, in Section 3.1 we summarise the procedures required for solving this optimal control problem. We also outline the fully discrete methods for the approximation of the state and adjoint equations in Section 3.2. We discuss the two-grid solution strategy in Section 3.3; and the adaptive-α algorithm in Section 3.1 which improves the control update. Section 4 contains results of our numerical experiments with the implemented solver. We use a 2-D benchmark problem to demonstrate the convergence of the proposed model, our multigrid performance where a linear complexity is shown, effectiveness of the adaptive-α algorithm and two-grid strategy. We use a 3-D benchmark problem to illustrate the importance of the proposed two-grid solution strategy in terms of the saved memory spaces. We also show 2-D and 3-D irregular shapes. Finally in Section 5 we summarise our major findings and discuss directions for future work.

2 Optimal control of a forced Allen-Cahn equation with volume constraint

As outlined in Section 1 the prototype state equation we consider in this work consists of the volume constrained Allen-Cahn equation with forcing,

$$\begin{cases} \epsilon\frac{\partial}{\partial t}\phi(\boldsymbol{x},t)=\epsilon\triangle\phi(\boldsymbol{x},t)-\epsilon^{-1}G'(\phi(\boldsymbol{x},t))+\eta(\boldsymbol{x},t)+\lambda(t), & \text{in } \Omega\times(0,T], \\ \phi(\boldsymbol{x},0)=\phi^0(\boldsymbol{x}), & \text{in } \Omega, \\ \nabla\phi(\boldsymbol{x},t)\cdot\boldsymbol{v}_\Omega(\boldsymbol{x})=0, & \text{on } \partial\Omega, \end{cases} \tag{2.1}$$

where $\phi(\boldsymbol{x},t)$ is the phase field variable, Ω is the spatial domain, $\epsilon>0$ is the parameter governing the width of the diffuse interface, $G(\phi)=\frac{1}{4}(1-\phi^2)^2$ is a double well potential which has minima at ±1 and λ is a time-dependent constraint on the mass that models a volume constraint [9] and $\boldsymbol{v}_\Omega$ is the normal to $\partial\Omega$.

As the volume enclosed by the target and initial interfaces may differ, i.e., $\int_\Omega\phi^0\neq\int_\Omega\phi_{obs}$, enforcing conservation of mass is inappropriate, instead we proceed as in [6] and enforce a constraint on the linear interpolation of the mass of the initial and target diffusive interfaces. To this end we define

$$M_\phi(t):=\int_\Omega\left[\phi^0(\boldsymbol{x})+\frac{t}{T}\left(\phi_{obs}(\boldsymbol{x})-\phi^0(\boldsymbol{x})\right)\right], \tag{2.2}$$

https://doi.org/10.4208/cicp.240715.080716a Published online by Cambridge University Press

and the volume constraint $\lambda(t)$ in (2.1) is then determined such that for $t\in(0,T]$

$$\int_\Omega \phi(\boldsymbol{x},t)=M_\phi(t). \tag{2.3}$$

In order to formulate our optimal control problem we introduce the objective functional J, which we seek to minimise

$$J(\phi,\eta)=\frac{1}{2}\int_\Omega(\phi(\boldsymbol{x},T)-\phi_{obs}(\boldsymbol{x}))^2d\boldsymbol{x}+\frac{\theta}{2}\int_0^T\int_\Omega\eta^2(\boldsymbol{x},t)d\boldsymbol{x}dt, \tag{2.4}$$

where $\theta>0$ is a regularisation parameter. The first term of the right-hand side of (2.4) is the so called fidelity term which measures the distance between the solution of the model and the target data ϕ_{obs} and the second term is the regularisation which is necessary to ensure a well-posed problem [8].

The optimal control problem we consider in this work may now be stated as the following minimisation problem. Given initial data ϕ^0 and target data ϕ_{obs}, find a space-time distributed forcing $\eta^*:\Omega\times[0,T)\to\mathbb{R}$ such that with ϕ a solution of (2.1) with initial condition $\phi(\cdot,0)=\phi^0(\cdot)$, the forcing η^* solves the minimisation problem

$$\min_\eta J(\phi,\eta), \text{ with } J \text{ given by (2.4).} \tag{2.5}$$

In order to apply the theory of optimal control of semilinear PDEs for the solution of the minimisation problem, we briefly outline the derivation of the optimality conditions, for further details see for example [8,10]. Adopting a Lagrangian approach, we introduce the Lagrange multiplier (adjoint state) p and the Lagrangian functional $\mathcal{L}(\phi,\eta,p)$ defined by

$$\mathcal{L}(\phi,\eta,p)=J(\phi,\eta)-\int_0^T\int_\Omega\left(\epsilon\frac{\partial}{\partial t}\phi-\epsilon\triangle\phi+\epsilon^{-1}G'(\phi)+\eta+\lambda\right)p. \tag{2.6}$$

Assuming the existence of an optimal control η^* and the associated optimal state ϕ^* and requiring stationarity of the Lagrangian at (ϕ^*,η^*) yields the (formal) first order optimality conditions [8,10]

$$\delta_\phi\mathcal{L}(\phi^*,\eta^*,p)\phi=0,\quad \forall\phi:\phi(\boldsymbol{x},t=0)=0, \tag{2.7}$$
$$\delta_\eta\mathcal{L}(\phi^*,\eta^*,p)\eta=0,\qquad \forall\eta. \tag{2.8}$$

Condition (2.7) yields the linear parabolic adjoint equation posed backwards in time

$$\begin{cases}\frac{\partial}{\partial t}p(\boldsymbol{x},t)=-\triangle p(\boldsymbol{x},t)+\epsilon^{-2}G''(\phi(\boldsymbol{x},t))p(\boldsymbol{x},t), & \text{in } \Omega\times[0,T),\\ p(\boldsymbol{x},T)=\phi(\boldsymbol{x},T)-\phi_{obs}(\boldsymbol{x}), & \text{in } \Omega,\\ \nabla p(\boldsymbol{x},t)\cdot\boldsymbol{v}_\Omega(\boldsymbol{x})=0, & \text{on } \partial\Omega\times[0,T).\end{cases} \tag{2.9}$$

Condition (2.8) together with the Riesz representation theorem yields the optimality condition for the control [8]

$$\delta_\eta\mathcal{L}(\phi^*,\eta^*,p)=\theta\eta^*+\frac{1}{\epsilon}p=0. \tag{2.10}$$

https://doi.org/10.4208/cicp.240715.080716a Published online by Cambridge University Press

We note that our approach to the optimal control problem involving the formulation of the adjoint problem appears to require a smooth potential G. The formulation of the adjoint problem is to our best knowledge an open problem for other widely used, but non smooth or unbounded, potentials such as the obstacle or logarithmic potential.

3 Numerical solution methods

In this section, we outline the numerical methods for solving the proposed optimal control of geometric evolution laws. For ease of exposition, we separate this complex solution procedure into three parts. The first part, which deals with the update of the optimal control η, is discussed in Section 3.1, with the assumption that the solutions of ϕ and p have already been obtained. In Section 3.2, we describe the second part that involves the spatial and temporal discretization schemes for the forward (ϕ) and adjoint (p) problems, i.e., the phase field Allen-Cahn equation (2.1) with volume constraint and the adjoint equation (2.9) respectively. In order to solve this problem efficiently and accurately, we employ several state-of-the-art algorithms, described in Section 3.3, including an important in-house two-grid solution strategy, parallelised multigrid solution methods as well as dynamic adaptive mesh refinement. This forms the third and last part of our solution procedure.

3.1 Adaptive iterative update for the optimal control

The control η is updated and obtained through an iterative approach, where the Allen-Cahn and adjoint equations (2.1) and (2.9) respectively, for each fixed time frame $[0,T]$ have to be solved repeatedly. The computational requirement for finding a satisfactory η is large. This is mainly due to two reasons; first the necessity of repeatedly solving both the Allen-Cahn and adjoint equations sequentially and second, the state (ϕ) and the forcing (η) must be stored at all iterations.

We denote a superscript ℓ for the η iteration, and at $\ell=0$, we take

$$\eta^{\ell=0}=0 \ \text{ on } \Omega\times[0,T) \tag{3.1}$$

as our initial guess for the control. A better initial guess for the control may be necessary for certain examples or applications, however for the benchmark examples presented in this paper the simple constant zero initial guess stated above was sufficient for convergence.

For the purpose of demonstration, let us assume that both the state and adjoint equations are solved by some known method with an acceptable accuracy. With this assumption, a gradient-based iterative update of the control, following the steepest descent approach, is employed using (2.10) and the update is given by

$$\eta^{\ell+1}=\eta^{\ell}-\alpha\left(\theta\eta^{\ell}+\frac{1}{\epsilon}p^{\ell}\right), \ \text{ on } \Omega\times[0,T), \tag{3.2}$$

https://doi.org/10.4208/cicp.240715.080716a Published online by Cambridge University Press

where $\ell+1$ denotes the next η iteration and ℓ indicates the current η iteration.

The whole procedure is repeated until the objective function J (see (2.4)) satisfies some pre-defined tolerances. There are two criteria: an absolute criterion and a relative criterion. The former terminates when the obtained J is smaller than a given fixed constant and the latter takes the difference between the current J and the previous one, and terminates when the difference between the two falls below a certain prescribed tolerance.

Due to the nature of the iterative update presented in (3.2), we expect each update on the control η to reduce the objective functional J. Hence we design an adaptive algorithm based upon this observation. We may start with an arbitrary value of α, namely α^ℓ. If the computed objective function J with α^ℓ is smaller than the previous one, we increase the value of $\alpha^{\ell+1}$ and continue the computation. However, if J gets larger, this means the value α^ℓ is not suitable, and the computation with η^ℓ need to be re-calculated using a smaller α (or the default minimum choice α_{min}).

We summarise this adaptive procedure in Algorithm 1. Note $\mathcal{P}_l$ and $\mathcal{P}_u$ are percentages, unless otherwise stated, we set them to be 50% and 110%, respectively. In Section 4.4 we illustrate the effectiveness of this adaptive-α procedure.

Algorithm 1 Adaptive-α

1.While the difference between consecutive Js is still large or J has not reached below a pre-defined tolerance **do**
2. Solve the forward Allen-Cahn equation in $\Omega\times(0,T]$
3. Compute the objective functional J^ℓ
4. if $J^\ell > J^{\ell-1}$ **and** $\ell>0$ **then**
    $\alpha = max(\alpha\times\mathcal{P}_l,\alpha_{min})$
    restart = **TRUE**
  else if $J^\ell < J^{\ell-1}$ **and** $\ell>0$ **then**
    $\alpha = \alpha\times\mathcal{P}_u$
    restart = **FALSE**
  end if
5. if restart == **FALSE then**
    Solve the backward adjoint equation in $\Omega\times[T,0)$
    Backup the current η
    Compute the next η using α
    Continue to the next η iteration
  else
    Compute a new η using the latest backup with α
    Restart the current η iteration
  end if
6. End

https://doi.org/10.4208/cicp.240715.080716a Published online by Cambridge University Press

3.2 Space-time discretizations of the forward Allen-Cahn and adjoint equations

At the beginning of each η iteration, we start by approximating the phase field equation (2.1). The spatial discretization scheme is a central finite difference method (FDM) with a standard seven-point stencil in 3-D on Cartesian grids with cell-centred vertices. Although for illustrative purposes, the discrete system presented here is in 3-D, the use of a standard five-point stencil in 2-D is straightforward. We assume N is the number of grid points in each coordinate direction, h is the uniform grid spacing (i.e. $h=\triangle x=\triangle y=\triangle z$), subscripts i,j,k are used to indicate each grid point and each point has Cartesian coordinate (x,y,z). We refer the reader to [14] for details on adaptive grid spacing and the use of adaptive multigrids. For the temporal discretization scheme we employ a fully-implicit second-order backward differentiation formula (BDF2) [11]. With a given end time T, we assume a uniform time step size τ. For a time discrete sequence, f, we denote by $f^n:=f(t_n)$. The standard BDF1 (also known as backward Euler method) is employed for the very first time step.

The result after applying the described discretisation to (2.1) is the following algebraic system arising at each time step, find $\phi_{i,j,k}^{n+1,\ell+1}$ such that,

$$\begin{aligned}&\epsilon\frac{\phi_{i,j,k}^{n+1,\ell+1}-\frac{4}{3}\phi_{i,j,k}^{n,\ell+1}+\frac{1}{3}\phi_{i,j,k}^{n-1,\ell+1}}{\tau}\\&=\frac{2\epsilon}{3}D\left(\phi_{i,j,k}^{n+1,\ell+1}\right)-\frac{2\left(-\phi_{i,j,k}^{n+1,\ell+1}+\left(\phi_{i,j,k}^{n+1,\ell+1}\right)^3\right)}{3\epsilon}+\frac{2\eta_{i,j,k}^{n+1,\ell+1}}{3}+\frac{2\lambda^{n+1}}{3},\end{aligned}\tag{3.3}$$

where $\ell+1$ denotes the current η iteration, $n+1$, n and $n-1$ indicate solutions from the current, previous and the one before the previous time steps, respectively. We denote the 3-D Laplacian operator D as

$$D\left(\phi_{i,j,k}\right)=\frac{\phi_{i+1,j,k}+\phi_{i-1,j,k}+\phi_{i,j+1,k}+\phi_{i,j-1,k}+\phi_{i,j,k+1}+\phi_{i,j,k-1}-6\phi_{i,j,k}}{h^2}.\tag{3.4}$$

Within each time step, while solving for the solution of the above system, we are also required to satisfy a given mass constraint. This is done by iteratively determining the time-dependent, spatially-uniform volume constraint λ for the imposed mass constraint. Therefore, the system in (3.3) has to be solved multiple times, until a stopping criterion for λ is met. We denote this λ iteration using a superscript Λ, and its update follows the multi-step approach presented in [9], which is given as

$$\lambda^{n+1,\Lambda+1}=\lambda^{n+1,\Lambda}+\frac{\left(\lambda^{n+1,\Lambda}-\lambda^{n+1,\Lambda-1}\right)\left[M_\phi^{n+1}-\int_\Omega\phi^{n+1,\Lambda}\right]}{\left(\int_\Omega\phi^{n+1,\Lambda}-\int_\Omega\phi^{n+1,\Lambda-1}\right)},\quad\text{for }\Lambda>1,\tag{3.5}$$

where M_ϕ is defined in (2.2), $\Lambda+1$, Λ and $\Lambda-1$ indicate values of λ from the current, previous and the one before the previous λ iterations, respectively. We follow [9] in using

https://doi.org/10.4208/cicp.240715.080716a Published online by Cambridge University Press

the initial guesses

$$\lambda^{\Lambda=0}=-\frac{2\epsilon}{\tau}+1,\quad \lambda^{\Lambda=1}=\frac{2\epsilon}{\tau}-1. \tag{3.6}$$

The stopping criterion used here is based upon the difference between consecutive values of λ. Providing a tolerance tol_λ, we consider the algorithm to have converged when $|\lambda^{n+1,\Lambda+1}-\lambda^{n+1,\Lambda}|<tol_\lambda$.

From our experience, using the initial guesses in (3.6) often led to more than three λ iterations within each time step (with, say, a typical choice of $tol_\lambda=0.01$). For later computations, we can improve these initial guesses with known (already computed) values. Within the first and second η iterations, from time step $n=3$ onwards, we choose the computed $\lambda^{n-1,\ell}$ and $\lambda^{n,\ell}$, where $\ell=1,2$, as our improved initial guesses. When we are at the third η iteration or beyond, we choose the two computed λ (corresponding to the current time step) from the previous two η iterations as initial guesses. It should be observed that the computed solution ϕ at each time step has to be stored in order to compute the adjoint state p later.

Having solved the algebraic system arising from the discretizations of the phase field representation of (2.1) forward in time and stored the obtained solutions, we have only completed the first of two parts of the solution procedure. The second part is to discretize and solve (2.9). We employ the described central FDM and BDF2 as our discretization schemes, and the resulting algebraic system for the adjoint state p is the following,

$$\begin{aligned}&\frac{p_{i,j,k}^{n+1,\ell+1}-\frac{4}{3}p_{i,j,k}^{n+2,\ell+1}+\frac{1}{3}p_{i,j,k}^{n+3,\ell+1}}{\tau}\\&=-\frac{2}{3}D\left(p_{i,j,k}^{n+1,\ell+1}\right)+\frac{2}{3}\left(\frac{-1+3\left(\phi_{i,j,k}^{n+1,\ell+1}\right)^2}{\epsilon^2}p_{i,j,k}^{n+1,\ell+1}\right).\end{aligned} \tag{3.7}$$

Note that the adjoint equation is posed backwards in time and its terminal condition is stated in (2.9). The BDF1 method needs to be employed for the first time step. For every subsequent time step the corresponding solution of ϕ that has been previously computed and stored enters as data in the adjoint equation.

3.3 Techniques for improving algorithm efficiency

In this section, we explain several state-of-the-art algorithms that are used for obtaining the solution of (3.3)-(3.7) efficiently. First of all, we describe an in-house two-grid solution strategy that significantly improves the CPU time for solving the adjoint equation as well as massively reducing the memory requirement for storing all the solutions. Secondly, we briefly mention a parallel, adaptive multigrid solution method that we use to solve the arising algebraic system at each time step. Since this has been described in previous works, here we only briefly illustrate that the two-grid solution strategy can be a natural extension to the standard multigrid V-cycles. We refer the interested reader to [12–15] for

https://doi.org/10.4208/cicp.240715.080716a Published online by Cambridge University Press

further details. Thirdly, the adaptive mesh refinement (AMR) can be included in the two-grid solution strategy in a straightforward manner. This significantly improves the CPU time for solving the Allen-Cahn equation. The final technique is the adaptive-α algorithm which has already been mentioned in Section 3.1, here we omit details of this algorithm.

We propose a two-grid solution strategy that exploits a key difference between the forward Allen-Cahn and the backward adjoint equations. As is well known for the Allen-Cahn equation the parameter ϵ determines the thickness of the diffuse interfacial region, Γ_ϵ, that approximates the hypersurface Γ. In order for the Allen-Cahn equation to reliably approximate mean curvature flow the interfacial region has to be well resolved. Typically eight grid points are required across the width of the diffusive interface, see [16]. On the other hand, in our numerical simulations we observed that the solution of the backward adjoint equation has less spatial variation, see Figs. 4(c) and 4(d) in which the optimised solutions of the adjoint equation at time $t = 0.0625$ and $t = 0.125$ are displayed, and so a milder restriction on the grid size h in the interfacial region is expected when solving this equation. Our numerical tests (see Section 4) suggest that such a strategy can dramatically reduce the memory requirement and increase computational efficiency without significantly compromising accuracy.

Within this implemented robust in-house two-grid solution strategy, we solve the Allen-Cahn equation on a grid hierarchy where its finest grid has sufficient grid points for the chosen ϵ; then to improve efficiency the backward adjoint equation is solved using only part of the grid hierarchy. As we stated previously, during the computation, at least solutions from two variables of ϕ, p and η from all time steps in the current η iteration are stored. If a very fine mesh resolution is required, this undoubtedly imposes a severe requirement for the memory, even in a parallel setting. To relax this constraint, we store all-time-step solutions only on the coarser grid where we solve the adjoint problem. When the stored solutions are required on finer grids (where ϕ is solved), an interpolation is used to transfer the stored solutions to that grid level. We illustrate this two-grid solution strategy in Fig. 1.

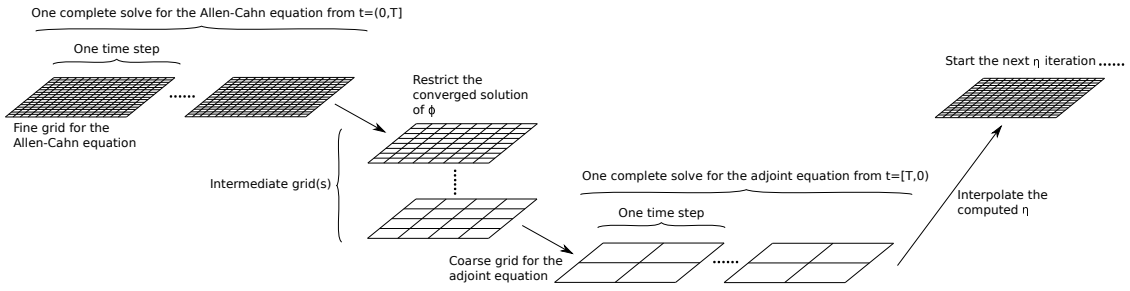


Figure 1: Sketch illustrating our in-house two-grid solution strategy, where the adjoint equation is solved on a much coarser grid. The storage for all-time-step solutions is done on such a grid so as to reduce the memory requirement.

https://doi.org/10.4208/cicp.240715.080716a Published online by Cambridge University Press

Remark 3.1. When solving such optimal control problems we note that apart from the computational complexity, in terms of CPU time, it should be noted that there is also a very large memory requirement. More specifically, η exists on all internal grid points for all time steps. The η update (3.2) requires the solution of the adjoint equation p on every grid point at every time step. For the adjoint equation (2.9), the solution of the state equation ϕ on every space-time grid point is also required. Thus, when we solve the Allen-Cahn equation, the solutions of ϕ and η are stored on every grid point for all time steps. The requirements on memory storage can become significant as the number of grid point increases.

To give the reader a brief idea, say one uses a 2-D grid consisting of 512^2 grid points with 50 forward and backward time steps for the Allen-Cahn and the adjoint equation respectively. The solutions of 2 variables (ϕ and η) are stored using a double-precision format, i.e., each float-point number occupies 8 bytes in the memory. This setting requires around 210 megabytes of memory space. However, if this simulation is done in 3-D on a uniform grid with the resolution of 512^3, the memory requirement becomes approximately 107 gigabytes. Such a simulation may not be feasible if parallelism and other advanced techniques are not employed. To this end, we couple our two-grid solution strategy with a parallelisation of a domain decomposition approach, as well as dynamic AMR to enable 3-D simulations.

The software framework used here is called Campfire v2.0. Comparing with its previous versions used in [12–14], the latest software received some significant changes to its structure in order to deal with a forward and a backward solve, as well as additional parallel memory allocations. This framework contains a geometric nonlinear multigrid solution method with a full approximation scheme (FAS), as well as a multi-level adaptive technique (MLAT) variant. However, when solving the linear adjoint equation, this FAS multigrid reduces to the standard linear multigrid method [17]. The multigrid methods are widely known to be one of the fastest numerical methods with a linear complexity [17–19], and we later demonstrate this in the paper. Its parallelisation comes from a domain decomposition technique, and the Message Passing Interface is used for parallel communication. Here using Fig. 2, we illustrate that the presented in-house two-grid solution strategy can be comfortably extended into the multigrid V-cycles.

Considering we are using phase field approximation, dynamic AMR is commonly employed so the interfacial region is well captured and resolved and the computations are reduced in other regions where the phase field variable tends to be constant. On the other hand, if the adjoint equation is solved with the dynamic AMR, further complications would occur involving the storage of the structure of the evolved mesh. This can be practically unfeasible with the memory requirement that is mentioned in the Remark 3.1 earlier. Therefore, we employ dynamic AMR for the solutions of the forward Allen-Cahn equation and solve the backwards adjoint PDE on a fixed (and much coarser) grid. For instance, the finest grid shown in Fig. 1 may be a dynamically adaptive grid, as well as the intermediate grids. However, the coarse grid that we solve the adjoint equation stays

https://doi.org/10.4208/cicp.240715.080716a Published online by Cambridge University Press

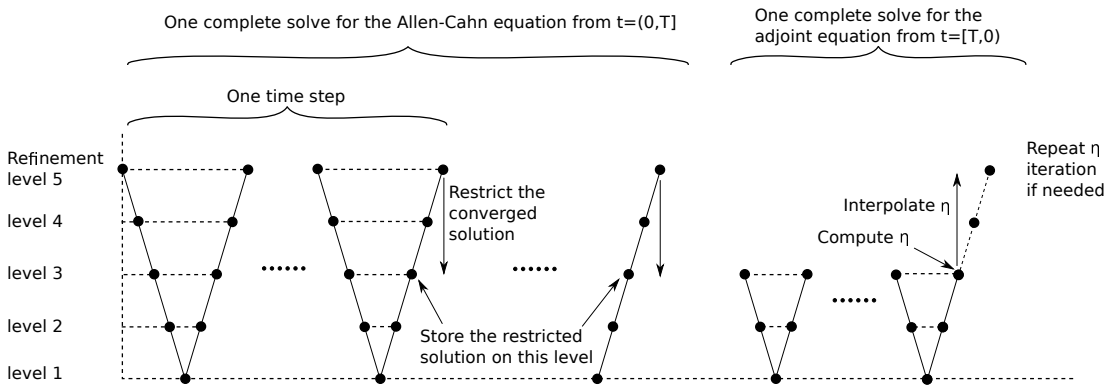


Figure 2: Sketch demonstrating our in-house multi-depth V-cycle multigrid strategy where the adjoint equation is solved on a much coarser grid. The storage for all-time-step solutions is done on such a grid so as to reduce the memory requirement.

fixed. We refer the interested reader to [13, 14] for the description of the dynamic AMR and its associated dynamic load-balancing in parallel.

4 Numerical experiments

All the results shown in this section were generated using the local HPC cluster provided and managed by the University of Sussex. This HPC cluster consists of 3000 computational units. The models of the computational units are AMD64, x86_64 or 64 bit architecture, made up of a mixture of Intel and AMD nodes varying from 8 cores up to 64 cores per node. Each unit is associated with 2GB memory space. Most of the simulations in this paper were executed using $4-32$ cores. The parallel scalability of our multigrid solver, Campfire, has been discussed in earlier publications such as [12–14], where in [12], they successfully scaled up to one thousand computational cores on the UK national supercomputer HECToR in 2013. We refer the reader to [13, 14] for a detailed explanation and related results for the parallel scalability of our software.

4.1 A 2-D benchmark example

We start with a benchmark 2-D example. The initial data is a circle centred at $(2,2)$ with radius 1. We use a hyperbolic tangent function to obtain a continuous interfacial region with a width of $\mathcal{O}(\epsilon)$

$$\phi^{t=0} = \tanh\left(\frac{-\left[(x-2)^2+(y-2)^2-1\right]}{\epsilon}\right). \tag{4.1}$$

https://doi.org/10.4208/cicp.240715.080716a Published online by Cambridge University Press

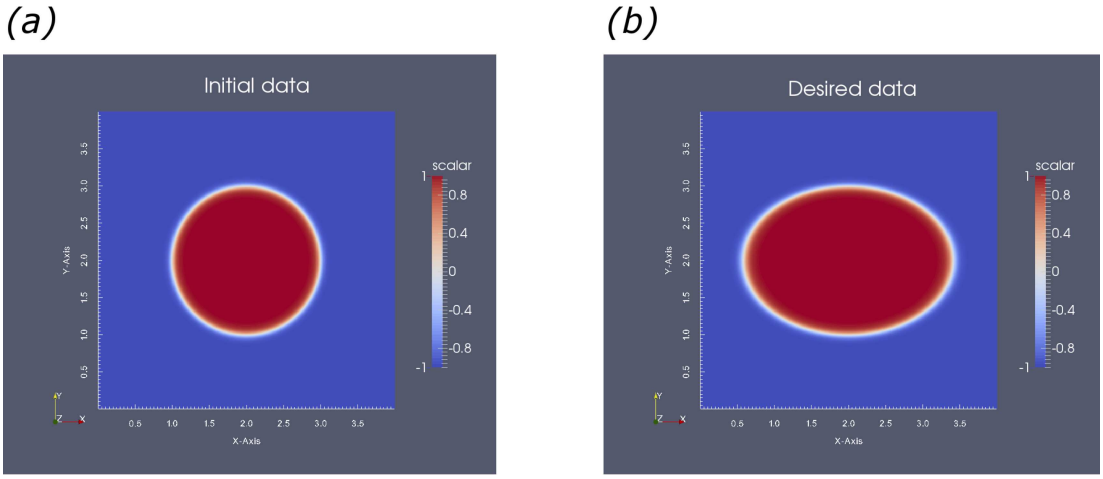


Figure 3: (a) shows the initial data (i.e. (4.1)) and (b) shows the desired data (i.e. (4.2)). The colour version of this figure is online.

The desired data is an ellipse:

$$\phi_{obs} = \tanh\left(\frac{-\left[\frac{(x-2)^2}{2}+(y-2)^2-1\right]}{\epsilon}\right). \tag{4.2}$$

Both the initial and desired shapes are illustrated in Fig. 3.

For illustrative purposes, we take the computational domain $\Omega=(0,4)^2$. The choices of parameters are given in Table 1. At each time step, the infinity norm of the residual is assessed, and the computation is said to have converged when this norm is smaller than 1.0×10^{-11}.

Table 1: The parameters of the optimal control problem for the 2-D examples.

Name	Description	Value
α	Step size for the control update	0.1
θ	Regularisation parameter	0.01
ϵ	Width of the diffuse interface	0.1
T	Total time duration	0.125
τ	Time step size	(varies)

We present the computed solutions from a uniform grid with a resolution of 1024^2. We select a time step $\tau=7.8125\times10^{-4}$, which yields 160 time steps. For the purpose of demonstration, we run 50 η iterations. In a multigrid setting, we use a 16^2 grid as the

https://doi.org/10.4208/cicp.240715.080716a Published online by Cambridge University Press

coarsest grid, and grids like 32^2, 64^2 etc. are the intermediate grids in the V-cycle hierarchy. The multigrid hierarchy is used for all simulations (16^3 is used as the coarsest grid for 3-D simulations) and is fairly standard, for clarity we forgo mentioning the multigrid setting later on.

The computed solution of ϕ at $t=0.0625$ (halfway through the time series) is presented in Fig. 4(a). The computed final shape after the last time step is illustrated in Fig. 4(b). The corresponding solutions of the adjoint p at time $t=0.0625$ and $t=T=0.125$ are included in Figs. 4(c) and (d). The corresponding solutions of η are shown in Figs. 4(e) and (f). We can see that, as expected for a circle evolving into an ellipse, the most forcing in (c) is placed at the left and the right. We also observe that in (d), the forcing is highly positive on the inner side of the phase field interface, and highly negative on the other, in order to keep the shape from further unwanted expansion or shrinking. The reductions in the objective functional J from the 50 η iterations are shown using a semi-log plot in Fig. 7 as "fixed alpha".

4.2 Convergence tests for the benchmark 2-D example

In this subsection, we report on numerical evidence that the proposed optimal control model converges as we refine both spatially and temporally.

We use the simulation that was described in the previous subsection as a benchmark. To recap, it was solved on a grid with the resolution of 1024^2 and 160 of time steps.

In order to conduct the convergence tests, the optimal control model here is solved independently on the following grids: 64^2, 128^2, 256^2 and 512^2. We use a time step size $\tau=0.0125$ for the 64^2 simulation, giving 10 time steps. Then the choices of τ is halved each time we use a finer grid. This results in 128^2 to have 20, 256^2 to have 40 and 512^2 to have 80 time steps, respectively. The other parameters for these simulations are the same as those shown in Table 1. However, instead of running the η iteration to a constant, we change the stopping criterion so that the η iteration stops when the objective functional J is below 0.065. Note this new stopping criterion requires 24 η iterations to be satisfied for the 1024^2 simulation and roughly the same number of iterations for other simulations.

We assess the solutions generated at three specific times: $t_1=0.0125$, $t_2=T/2=0.0625$ and $t_3=T=0.125$. Note t_1 is at the end of the first time step of the 64^2 simulation, and subsequently the end of the second, fourth and eighth time steps of the 128^2, 256^2 and 512^2 simulations respectively.

To compare the solutions spatially at the corresponding t_1, t_2 and t_3, further procedures are required. This is because the solutions are generated with different resolutions of grids. As mentioned earlier, we use the solution from 1024^2 grid as the benchmark, therefore all the solutions from other simulations other than 1024^2 are interpolated to the uniform 1024^2 grid. Note this process uses a standard bilinear interpolation [17], which is also the one used in our multigrid solver.

The interpolated solutions are compared with the benchmark solutions at t_1, t_2 and

https://doi.org/10.4208/cicp.240715.080716a Published online by Cambridge University Press

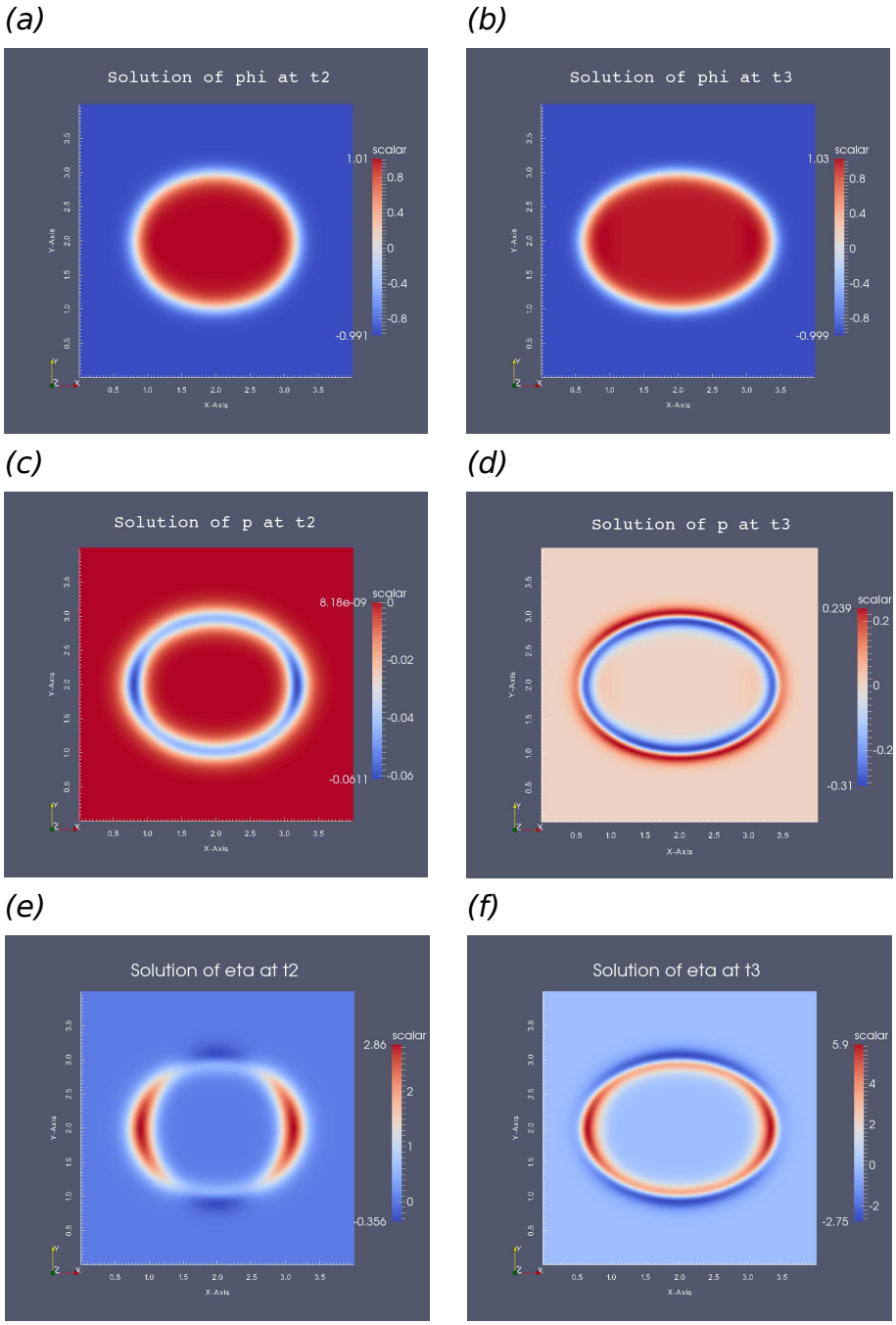


Figure 4: (a) illustrates the computed solution of ϕ halfway through (i.e. $t=0.0625$) and (b) shows the computed final shape (i.e. $t=T=0.125$). The corresponding solutions of adjoint p are included in (c) and (d) and the solutions of η are shown in (e) and (f). The colour version of this figure is online.

https://doi.org/10.4208/cicp.240715.080716a Published online by Cambridge University Press

Table 2: The convergence tests for the solutions of ϕ, adjoint p and η.

$L_2(\Omega)$ error for ϕ			
	$m=t_1$	$m=t_2$	$m=t_3$
$d^m_{64^2}$	3.4264×10^{-2}	4.9226×10^{-2}	6.8561×10^{-2}
$d^m_{128^2}$	1.9721×10^{-2}	4.8058×10^{-2}	6.0397×10^{-2}
$d^m_{256^2}$	8.1793×10^{-3}	2.2300×10^{-2}	3.4557×10^{-2}
$d^m_{512^2}$	2.7850×10^{-3}	8.0407×10^{-3}	1.3192×10^{-2}
$L_2(\Omega)$ error for adjoint p			
	$m=t_1$	$m=t_2$	$m=t_3$
$d^m_{64^2}$	1.6773×10^{-2}	1.8048×10^{-2}	4.9344×10^{-2}
$d^m_{128^2}$	9.9721×10^{-3}	1.0158×10^{-2}	3.1554×10^{-2}
$d^m_{256^2}$	7.9290×10^{-3}	8.5311×10^{-3}	2.2551×10^{-2}
$d^m_{512^2}$	6.5082×10^{-3}	7.5551×10^{-3}	1.4901×10^{-2}
$L_2(\Omega)$ error for η			
	$m=t_1$	$m=t_2$	$m=t_3$
$d^m_{64^2}$	1.6976×10^{-1}	2.0752×10^{-1}	7.5240×10^{-1}
$d^m_{128^2}$	1.1923×10^{-1}	1.5793×10^{-1}	6.2554×10^{-1}
$d^m_{256^2}$	8.5093×10^{-2}	1.0601×10^{-1}	5.3023×10^{-1}
$d^m_{512^2}$	3.2344×10^{-2}	3.9359×10^{-2}	2.6302×10^{-1}

t_3, we use the $L_2(\Omega)$ error defined as follows

$$d^m_l := \frac{\sum_{i=1}^N \sum_{j=1}^N (\phi^{m,l}_{i,j} - \phi^{m,1024^2}_{i,j})^2}{N\times N}, \quad m=t_1,t_2,t_3,\ l=64^2,128^2,256^2,512^2, \tag{4.3}$$

where $\phi^{m,l}_{i,j}$ is the computed and interpolated value of ϕ on grid l and $N=1024$ is the number of internal grid points (after interpolation) on one axis. This is repeated for the solutions of the adjoint p and η. We summarise the convergence tests in Table 2. It can be seen from this table that as we refine both spatially and temporally, the solutions of the model appear to converge.

4.3 The multigrid performance on the benchmark 2-D example

Within our software framework, the algebraic system arising from each time step is solved by a multigrid solver. Here in this subsection, we assess, numerically, the performance of our multigrid solver.

First of all, we present multigrid convergence rates by plotting the infinity norm of the residual at the end of each V-cycle from a typical time step. Furthermore, since two different equations are solved separately, we consider them separately and illustrate these results in Fig. 5. The eight lines in the two plots in Fig. 5 are nearly parallel to each other

https://doi.org/10.4208/cicp.240715.080716a Published online by Cambridge University Press

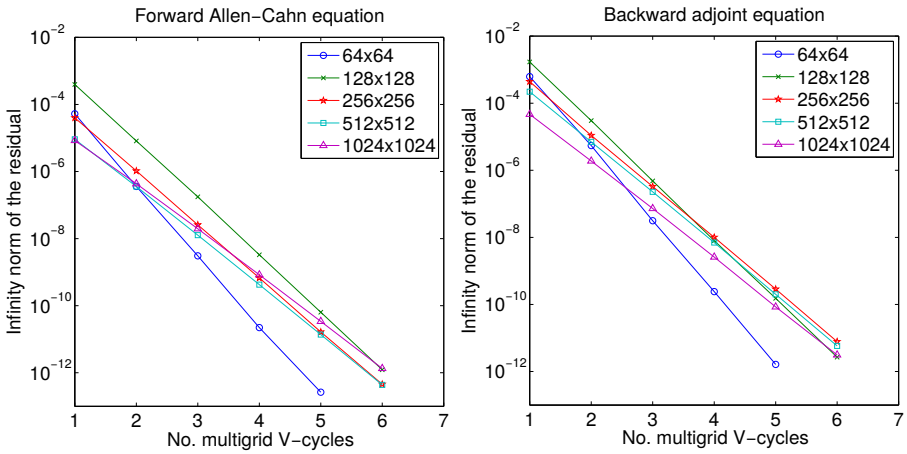


Figure 5: The multigrid convergence rates for the forward Allen-Cahn and backward adjoint equations. The colour version of this figure is online.

which suggests the reductions in the infinity norms of the residuals are independent of grid sizes.

We demonstrate the linear complexity of our multigrid solver in Fig. 6. Five simulations (64^2, 128^2, 256^2, 512^2 and 1024^2) are timed with a single computational core and the averages of the CPU costs for 10 η iterations are plotted as shown in Fig. 6. For clarity, we plot a red line of slope 1. The reason why the forward solver costs more is because, the volume constraint typically requires 2 - 3 choices of λ (more in the first two η iterations, since the guesses for λ are poor), meaning the algebraic system from the forward equation has to be solved 2–3 times within each time step. On the other hand, the algebraic system for the backward equation requires only one solve.

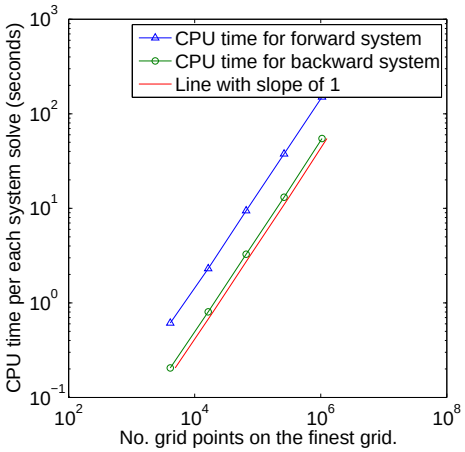


Figure 6: A log-log plot to illustrate the linear complexity of our multigrid solver. For comparisons, a line of slop 1 is included. The colour version of this figure is online.

https://doi.org/10.4208/cicp.240715.080716a Published online by Cambridge University Press

4.4 Adaptive-α algorithm with the benchmark 2-D example

An additional improvement to the efficiency may come from using the described adaptive-α algorithm (see Algorithm 1). In this subsection, we illustrate the effectiveness of this approach.

The two most influential parameters in the algorithm are $\mathcal{P}_l$ and $\mathcal{P}_u$ which control the incremental and decremental portions of the step size respectively. For the purpose of demonstration, we use the described 1024^2 simulation with 160 time steps. As mentioned previously, the reductions of the objective functional J within a fixed 50 η iterations is shown in Fig. 7(a). The initial choice of $\alpha^{\ell=0}=0.1$ is the same as the one used in the fixed α simulations previously.

Here we choose three different incremental parameters: $\mathcal{P}_u^1=110\%$, $\mathcal{P}_u^2=120\%$ and $\mathcal{P}_u^3=130\%$. The corresponding decremental parameters are $\mathcal{P}_l^1=50\%$, $\mathcal{P}_l^2=40\%$ and $\mathcal{P}_l^3=30\%$, respectively. Thus the more increases to the step size, the harder the penalisations become. For comparison, simulations with these three pairs of parameters are done with 50 fixed η iterations. We plot the reductions of the objective functional J when using the adaptive-α algorithm in Fig. 7(a). Note that in this figure, we only show the J from the successful iterations. For completeness, we illustrate the evolutions of α from all four simulations in Fig. 7(b), where the decreases in the values of α indicate the failed attempts. A trade-off can be observed from these two figures: a larger incremental parameter leads to a faster convergence, however, this may result in more failed attempts and thus in turn results in more computational time. In this case the gains in efficiency of the adaptive-α approach against a fixed value of α are evident. The use of an adaptive α is motivated by the fact that in general our initial guess for the solution to the optimal control problem may be poor and hence large step sizes may be admissible in the steepest descent update as we are far from local minima. As we approach the local minima smaller step sizes are necessary to prevent overshoot and hence some adaptivity in the parameter α is expected

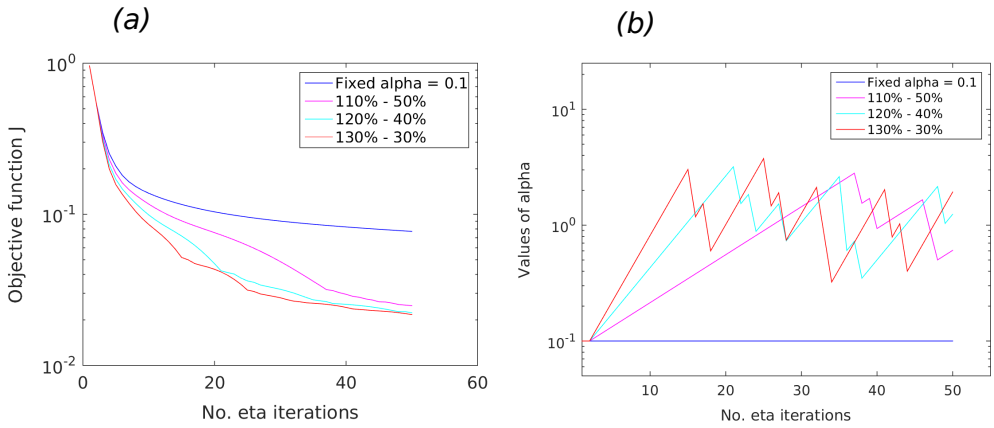


Figure 7: A semi-log plot shows the reductions of the objective functional J from using constant and adaptive αs in (a). A semi-log plot shows the changes in the values of α in (b). The colour version of this figure is online.

https://doi.org/10.4208/cicp.240715.080716a Published online by Cambridge University Press

to be desirable. More involved algorithms for the selection of an optimal parameter α, e.g., via a line search [8], may also be worthwhile topics of future investigation.

4.5 Two-grid solution strategy with dynamic AMR on the benchmark 2-D example

In Remark 3.1 we note that, due to the very large number of degrees of freedom that are typically required in order to accurately resolve phase field representation of interfaces with large curvatures, simulations on uniform grids in three space dimensions are unlikely to be feasible. To this end we introduce the two-grid AMR solution strategy proposed in Section 3.3. In this subsection, we investigate the effectiveness of this two-grid solution strategy, in two space dimensions, and illustrate its robustness with the use of dynamic AMR.

We consider a two-grid simulation where we solve the forward Allen-Cahn equation on a 1024^2 uniform grid while the adjoint equation and the storage of all the solutions, η, ϕ, p, takes place on a 64^2 uniform grid. The solutions of this two-grid simulation are compared with solutions using a standard (one-grid) 64^2 uniform grid simulation. We note that in both simulations all solutions are stored on a grid with resolution of 64^2.

For simplicity, we take 160 time steps for both simulations so that temporal errors have less influence. Like the convergence tests shown in Subsection 4.2, we solve the system until J gets below 0.065. In order to compute the error, the solutions from both simulations are interpolated and compared against solutions from the benchmark simulation of Section 4.2 that used a uniform grid of 1024^2 for both the forward and adjoint problems.

We illustrate the errors in Table 3, where

$$d^m_{1024^2-64^2} := \frac{\sum_{i=1}^N \sum_{j=1}^N (\phi_{i,j}^{m,1024^2-64^2} - \phi_{i,j}^{m,1024^2})^2}{N \times N}, \quad m = t_1, t_2, t_3,$$

with $\phi_{i,j}^{m,1024^2-64^2}$ denoting the ϕ solution from the two-grid simulation.

From this table we see that solving ϕ on a finer grid, while solving for p on a coarse grid and storing all solutions on the coarse grid, not only results in a reduction of the error in ϕ but it also results in a reduction of the errors of p and η. However, since the number of degrees of freedom in the two-grid (1024^2-64^2) simulation is considerably larger than the number of degrees of freedom in the standard 64^2 simulation, this behaviour is somewhat expected. In the next simulation we conduct a comparison between a two-grid simulation that solves the Allen-Cahn equation on an adaptive 256^2 grid with dynamic AMR and the adjoint equation on a uniform 64^2 grid, with a simulation on a standard 128^2 uniform grid. In this comparison the number of degrees of freedom in the two simulations is comparable, 17200 (maximum number of degrees of freedom occurred within the forward solve) for the two-grid simulation versus $128^2 = 16384$.

Both simulations have 160 time steps. The errors are shown in Table 4. From this table, we can see that for the two-grid simulation only the errors in ϕ are better. This is expected

https://doi.org/10.4208/cicp.240715.080716a Published online by Cambridge University Press

Table 3: Comparisons of errors between a two-grid simulation (1024^2-64^2) and a standard 64^2 simulation.

$L_2(\Omega)$ error for ϕ			
	$m=t_1$	$m=t_2$	$m=t_3$
$d^m_{64^2}$	3.1224×10^{-2}	4.7216×10^{-2}	6.3531×10^{-2}
$d^m_{1024^2-64^2}$	8.1840×10^{-3}	1.6194×10^{-2}	8.6616×10^{-3}
$L_2(\Omega)$ error for adjoint p			
	$m=t_1$	$m=t_2$	$m=t_3$
$d^m_{64^2}$	1.3722×10^{-2}	1.2018×10^{-2}	4.0874×10^{-2}
$d^m_{1024^2-64^2}$	6.1444×10^{-3}	4.9266×10^{-3}	9.9998×10^{-3}
$L_2(\Omega)$ error for η			
	$m=t_1$	$m=t_2$	$m=t_3$
$d^m_{64^2}$	1.2176×10^{-1}	1.8872×10^{-1}	6.9943×10^{-1}
$d^m_{1024^2-64^2}$	9.9110×10^{-2}	1.3768×10^{-1}	4.2063×10^{-1}

Table 4: Comparisons of errors between an adaptive two-grid simulation (256^2-64^2) with AMR and a standard 128^2 uniform grid simulation.

$L_2(\Omega)$ error for ϕ			
	$m=t_1$	$m=t_2$	$m=t_3$
$d^m_{128^2}$	1.2327×10^{-2}	2.6664×10^{-2}	3.8450×10^{-2}
$d^m_{256^2-64^2}$	8.6270×10^{-3}	1.6895×10^{-2}	3.2925×10^{-2}
$L_2(\Omega)$ error for adjoint p			
	$m=t_1$	$m=t_2$	$m=t_3$
$d^m_{128^2}$	9.2021×10^{-3}	9.7122×10^{-3}	3.0233×10^{-2}
$d^m_{256^2-64^2}$	1.0004×10^{-2}	1.4886×10^{-2}	2.7392×10^{-2}
$L_2(\Omega)$ error for η			
	$m=t_1$	$m=t_2$	$m=t_3$
$d^m_{128^2}$	9.7196×10^{-2}	1.2153×10^{-1}	5.3632×10^{-1}
$d^m_{256^2-64^2}$	7.7932×10^{-2}	1.4930×10^{-2}	5.9167×10^{-1}

as the adjoint is solved on a coarser grid (i.e. 64^2) and η, ϕ and p are stored on this coarse grid. On the other hand, it is important to note that we can store all the solutions on a coarser grid as well as solving the adjoint equation there without compromising too much on accuracy; this is crucial for 3-D simulations. In Fig. 8 we show two snapshots of our dynamic AMR at $t=t_1$ and $t=T$.

4.6 3-D example

We mentioned previously in Remark 3.1 that solving a 512^3 3-D simulation using a standard uniform grid requires memory of over 100 gigabytes space. Using the two-grid

https://doi.org/10.4208/cicp.240715.080716a Published online by Cambridge University Press

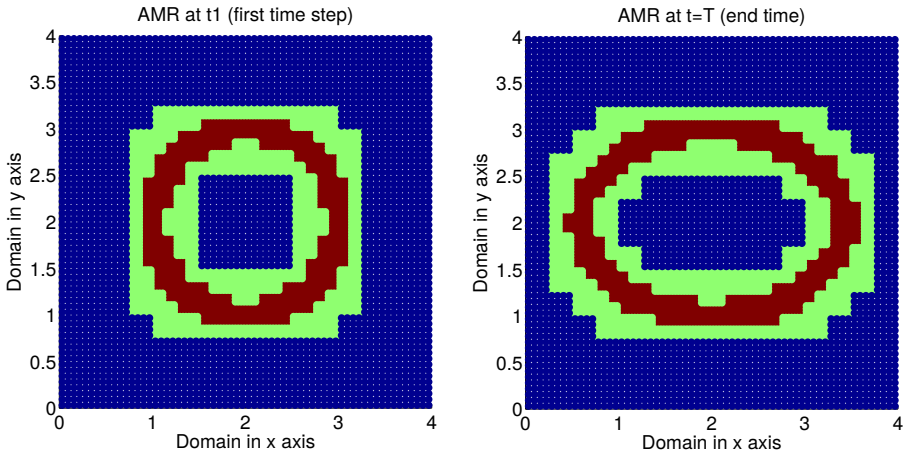


Figure 8: Two colour plots showing the dynamic AMR in our solver. The blue region shows the 64^4 grid; light green region indicates the 128^2 grid; and finally red region illustrates the finest 256^2 grid. The colour version of this figure is online.

solution strategy and dynamic AMR for the phase field variable, we can run a 512^3 simulation with less than 20 gigabytes memory requirement. This simulation is done in a 3-D domain $\Omega=(0,1)^3$. We choose a uniform 64^3 to be the grid where we store all the solutions and solve the adjoint equation. The finest grid is an adaptive grid and if it was uniform, it would have the resolution of 512^3. The temporal domain is $(0,T]=(0,0.001]$, with a time step size $\tau=5\times10^{-5}$. We use the same α and θ shown in Table 1. More importantly, we choose an $\epsilon=0.02$, with the finest grid, we can ensure there are roughly 10 grid points in the interfacial region in each axis direction. It is worth noting that the interfacial region of this simulation *can not* be resolved accurately on any coarser grids than proposed here [16].

We define the initial shape to be a sphere

$$\phi^{t=0}=\tanh\left(\frac{-\left[(x-0.5)^2+(y-0.5)^2+(z-0.5)^2-0.25^2\right]}{\epsilon}\right) \tag{4.4}$$

and the desired data to be an ellipsoid

$$\phi_{obs}=\tanh\left(\frac{-\left[\frac{(x-0.5)^2}{2}+(y-0.5)^2+(z-0.5)^2-0.25^2\right]}{\epsilon}\right). \tag{4.5}$$

The zero-isosurface of ϕ for both the initial and desired shapes are illustrated in Fig. 9(a) and (b) respectively. Following a fixed 15 η iterations, we present two plots of the zero-isosurface of ϕ in Figs. 9(c) and (d). The solution in (c) is halfway through the temporal domain and the solution in (d) is the computed final shape. We use colours and

https://doi.org/10.4208/cicp.240715.080716a Published online by Cambridge University Press

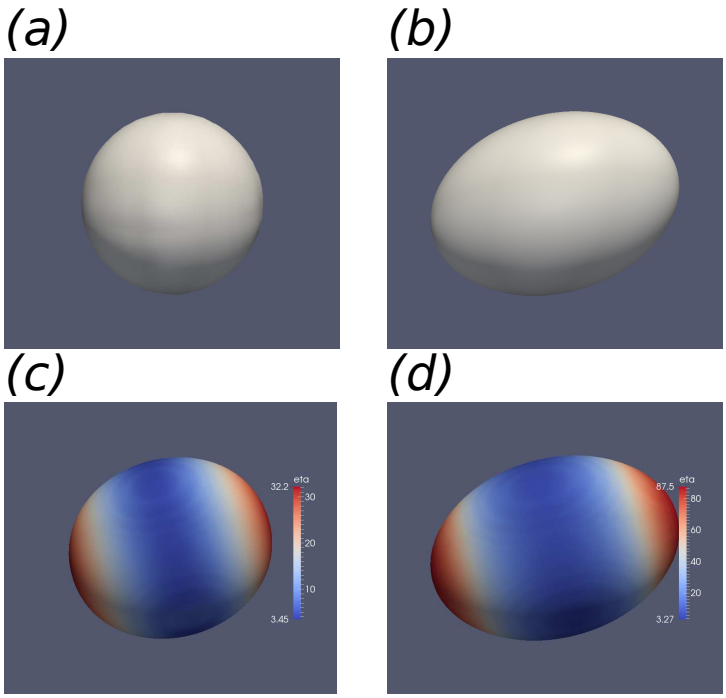


Figure 9: Figures (a) and (b) show the zero-isosurface of ϕ of initial data (i.e. (4.4)) and desired data (i.e. (4.5)) respectively; (c) and (d) illustrate the zero-isosurface of computed solutions halfway through (i.e. $t=T/2$) and the final shape (i.e. $t=T=0.001$) respectively. We use colours to indicate the corresponding solutions of η on the zero-isosurface. The colour version of this figure is online.

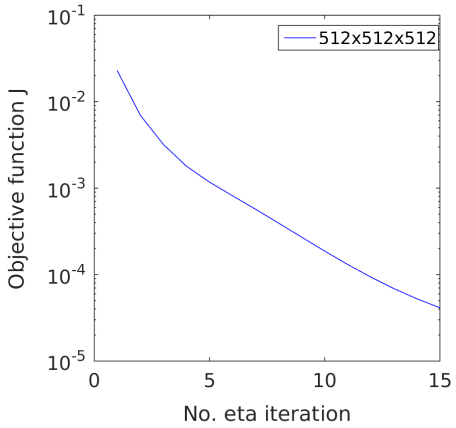


Figure 10: A semi-log plot shows the reductions of the objective function J. The colour version of this figure is online.

colour-value indicator on the side to demonstrate the corresponding solutions of η on the zero-isosurface. The reductions of the objective function J are shown in Fig. 10.

https://doi.org/10.4208/cicp.240715.080716a Published online by Cambridge University Press

4.7 Irregular shapes

In all our previous simulations we used relatively simple shapes for illustrative purposes only. In this subsection, we show some irregular shapes in both 2-D and 3-D in order to illustrate that the proposed optimal control approach is capable of dealing with general interfaces.

We start with a 2-D example which takes a circle as the initial shape and the desired shape is the following

$$\phi_{obs} = \max\left\{ \tanh\left(\frac{-\left[\frac{[(x-2)+(y-2)]^2}{6} + \frac{[(y-2)-(x-2)]^2}{1} - 1 \right]}{\epsilon} \right), \right.$$
$$\left. \tanh\left(\frac{-\left[\frac{[(x-2)+(y-2)]^2}{6} + \frac{[(y-2)-(x-2)]^2}{1} - 1 \right]}{\epsilon} \right) \right\}. \tag{4.6}$$

We take the computational domain $\Omega = (0,4)^2$ and use the parameters presented in Table 1. This simulation is solved using a two-grid approach which has a 512^2 grid for the Allen-Cahn and a 64^2 grid for the adjoint equation. We set $T = 0.05$ and use a time step size $\tau = 0.001$. The initial and desired data are illustrated in Fig. 11. We present our results in Fig. 12, which include the solutions of ϕ at the first time step, the halfway mark (i.e. $t = 0.0025$) and the end time, together with their corresponding control η.

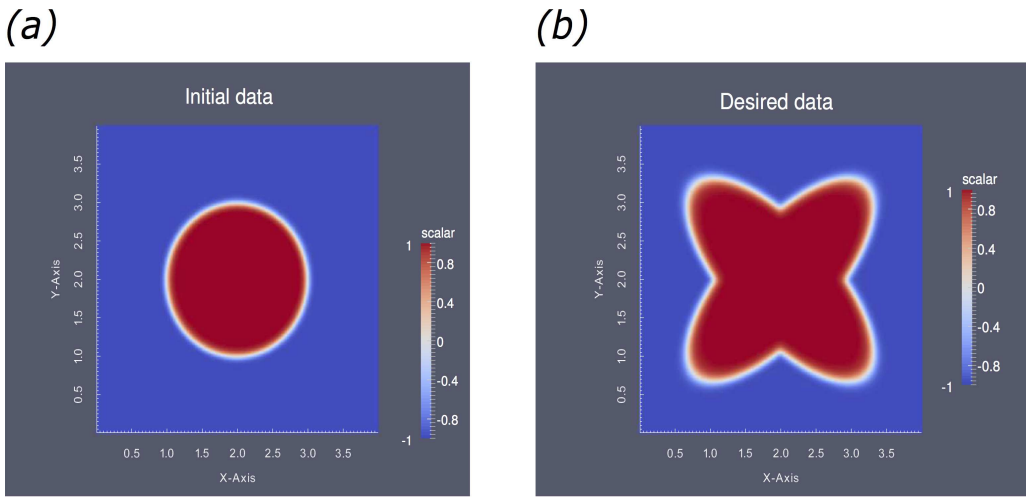


Figure 11: (a) shows the initial data from (4.1) and (b) illustrates the desired data from (4.6). The colour version of this figure is online.

https://doi.org/10.4208/cicp.240715.080716a Published online by Cambridge University Press

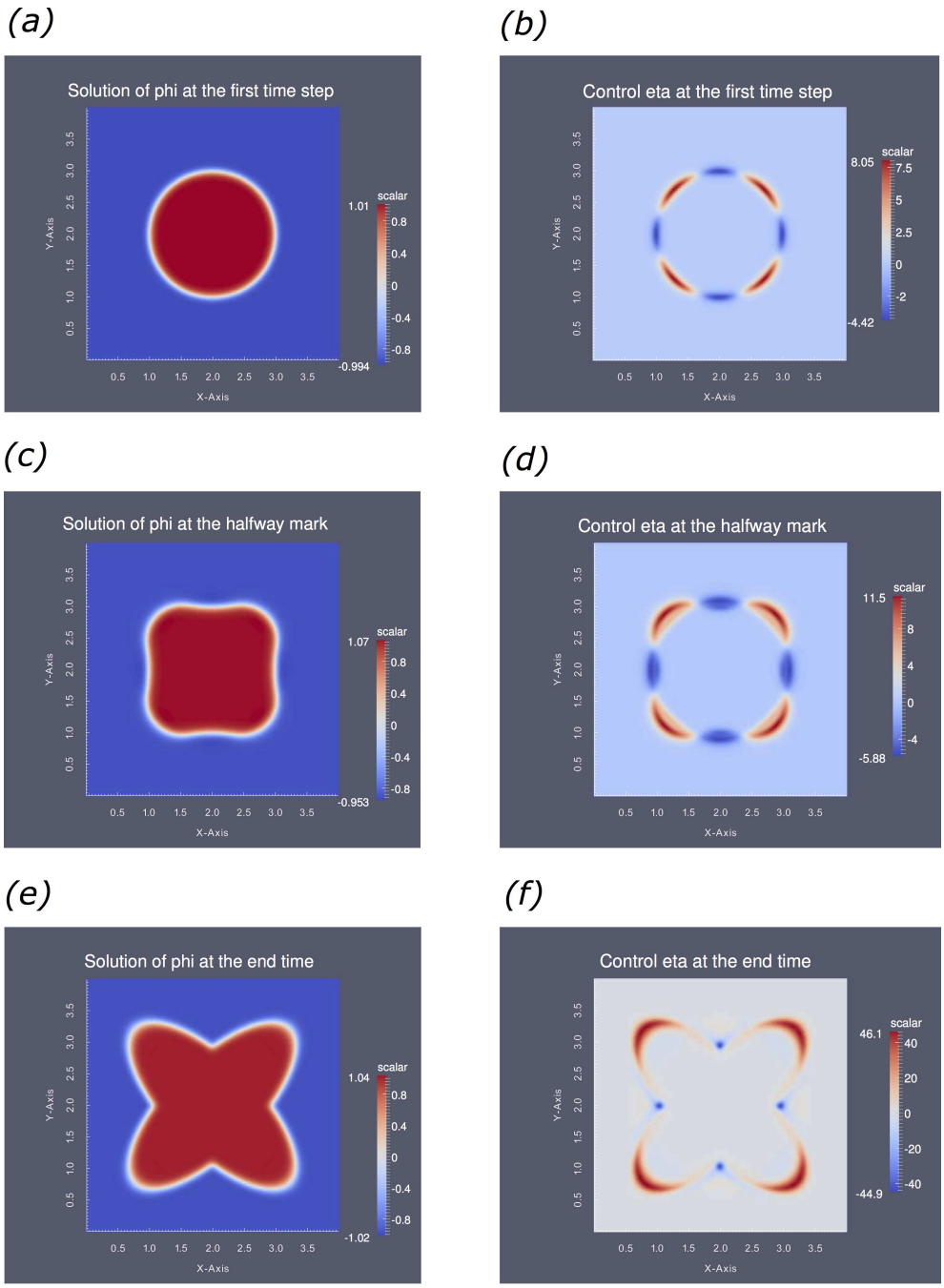


Figure 12: (a) and (b) show the solutions of ϕ and η at the first time step, respectively; (c) and (d) illustrate the solutions at the halfway mark (i.e. $t=0.0025$); (e) and (f) show the solutions at the final time T. The colour version of this figure is online.

https://doi.org/10.4208/cicp.240715.080716a Published online by Cambridge University Press

We define two 3-D shapes as follows

$$\phi^0 = \tanh\left(\frac{-\left[2\left((x-2)-(z-2)^2\right)^2+(y-2)^2+(z-2)^2-1\right]}{\epsilon}\right), \tag{4.7}$$

$$\phi_{obs} = \tanh\left(\frac{-\left[\left((y-2.3)-(z-2.3)^2\right)^2+2(x-2.3)^2+(z-2.3)^2-1\right]}{\epsilon}\right). \tag{4.8}$$

The simulation has the same setting as the one described in Subsection 4.6 and we illustrate the zero-isosurface together with the values of the optimal control η on this isosurface in Fig. 13.

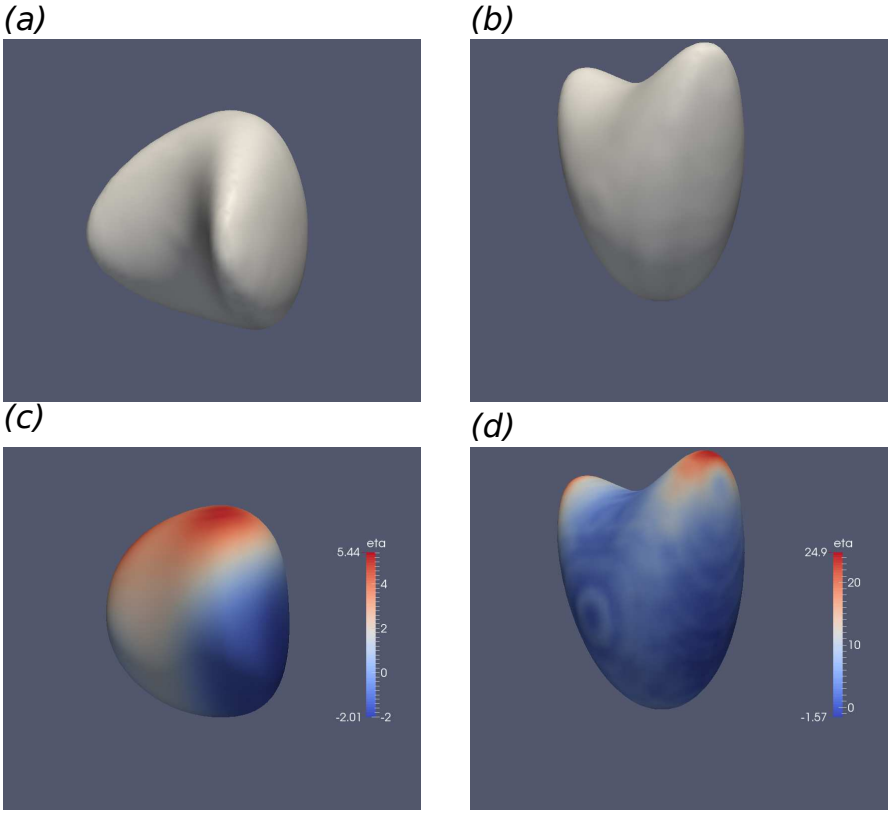


Figure 13: Figures (a) and (b) show the zero-isosurface of ϕ of initial data and desired data respectively; (c) and (d) illustrate the zero-isosurface of computed solutions halfway through (i.e. $t=T/2$) and the final shape (i.e. $t=T=0.001$) respectively. We use colours to indicate the corresponding solutions of η on the zero-isosurface. The colour version of this figure is online.

https://doi.org/10.4208/cicp.240715.080716a Published online by Cambridge University Press

5 Conclusion

In this work, we focussed on the development of robust and efficient solution procedures for the approximation of the optimal control of geometric evolution laws using phase field formulations. The problems under consideration arise naturally in many applications [1–3, 8, 10]. Such optimal control problems are very computationally demanding and memory hungry especially when posed in three space dimensions. Thus the development of efficient, robust and accurate solvers is of much importance. We have described, in detail, a solution procedure that combines a number of state-of-the-art algorithms to improve overall efficiency. We employed a steepest descent approach for the iterative computation of the optimal control. We introduced an adaptive-step-size algorithm which tries to use as large a step size as possible to reduce the number of iterations needed. Robust and efficient solvers for both the forward (Allen-Cahn) and adjoint equations, based on FAS multigrid methods with MLAT are described together with their parallel implementation which is crucial for minimising wall clock time due to the massive memory requirements. We discussed the use of mesh refinement which dramatically reduces the number of degrees of freedom required for the solution of the forward problem, and is crucial in terms of reducing the computational complexity. A major finding of this work is that a two-grid solution strategy, in which the forward equation is solved on an adaptively refined grid whilst the adjoint problem is solved on a coarser grid, thus significantly reducing CPU and memory requirements, appears to lead to only a minor loss in accuracy. We have implemented our algorithms and conducted detailed tests and benchmarks of our solution methods using 2-D and 3-D examples. The conclusion is that our solution algorithms can significantly improve efficiency while maintaining an acceptable accuracy.

Possible further work, which is the focus of our current work is the application of the methodologies described in this article to real world problems. In particular, as investigated in Blazakis et al. [6], the optimal control problem we solve may be useful for whole cell tracking and reconstruction of dynamic cell morphologies from static imaging data. A particular advantage of the present approach is that, in contrast to the majority of existing whole cell tracking algorithms, aspects of the physics of cell migration may be encoded in the forward model and hence in the recovered trajectories. Using the techniques introduced in this paper, we may consider 3-D examples of cell tracking as well as other scenarios involving tracking multiple cells over long time horizons which requires high spatial resolution and integration over a large time interval. Our solution methods are not restricted to forward models involving phase field formulations of geometric evolution laws; we expect that our solution methodologies are likely to be a robust and efficient option for problems involving the control of semilinear parabolic PDEs in general.

Data Management

All the computational data output is included in the present manuscript.

https://doi.org/10.4208/cicp.240715.080716a Published online by Cambridge University Press

Acknowledgments

All authors acknowledge support from the Leverhulme Trust Research Project Grant (RPG-2014-149). The work of CV, VS and AM was partially supported by the Engineering and Physical Sciences Research Council, UK grant (EP/J016780/1). This work (AM) has also received funding from the European Union's Horizon 2020 research and innovation programme under the Marie Sklodowska-Curie grant agreement No 642866. The work of CV is partially supported by an EPSRC Impact Accelerator Account award. The authors (FWY, CV, VS, AM) thank the Isaac Newton Institute for Mathematical Sciences for its hospitality during the programme (Coupling Geometric PDEs with Physics for Cell Morphology, Motility and Pattern Formation; EPSRC EP/K032208/1). AM was partially supported by Fellowships from the Simons Foundation. AM is a Royal Society Wolfson Research Merit Award Holder generously funded by the Royal Society and the Wolfson Foundation.

References

[1] Papadakis, N. and Mémin, E., Variational optimal control technique for the tracking of deformable objects, Computer Vision, IEEE 11th International Conference, 1-7, 2007.
[2] Haußer, F., Rasche, S. and Voigt, A., The influence of electric fields on nanostructures-simulation and control, Mathematics and Computers in Simulation, 80(7), 1449-1457, 2010.
[3] Haußer, F., Rasche, S. and Voigt, A., Control of nanostructures through electric fields and related free boundary problems, in: Constrained Optimization and Optimal Control for Partial Differential Equations, 561-572, 2012.
[4] Hogea, C., Davatzikos, C. and Biros, G., An image-driven parameter estimation problem for a reaction–diffusion glioma growth model with mass effects, Journal of mathematical biology, 56(6), 793-825, 2008
[5] Croft, W., Elliott, C.M., Ladds, G., Stinner, B., Venkataraman, C. and Weston, C., Parameter identification problems in the modelling of cell motility, Journal of Mathematical Biology, 71(2), 399-436, 2015.
[6] Blazakis, K.N., Madzvamuse, A., Reyes-Aldasoro, C.C., Styles, V. and Venkataraman, C., Whole cell tracking through the optimal control of geometric evolution laws, Journal of Computational Physics, 297, 495-514, 2015.
[7] Vierling, M., Parabolic optimal control problems on evolving surfaces subject to point-wise box constraints on the control–theory and numerical realization, Interfaces and Free Boundaries, 16(2), 137-173, 2014.
[8] Tröltzsch, F., Optimal control of partial differential equations: theory, methods and applications, AMS Bookstore, 112, 2010.
[9] Blowey J. and Elliott C., Curvature dependent phase boundary motion and parabolic double obstacle problems, In Degenerate Diffusions, 51, 52, 55, 19-60, 1993.
[10] Hinze, M., Pinnau, R., Ulbrich, M. and Ulbrich, S., Optimization with PDE Constraints, Mathematical Modelling: Theory and Applications, 23, 2009.
[11] Emmerich, E., Stability and error of the variable two-step BDF for semilinear parabolic problems, Journal of Applied Mathematics and Computing, 19, 33-55, 2005.

https://doi.org/10.4208/cicp.240715.080716a Published online by Cambridge University Press

[12] Bollada, P., Goodyer, C., Jimack, P., Mullis, A. and Yang, F., Thermalsolute phase field three dimensional simulation of binary alloy solidification, Journal of Computational Physics, 287, 130-150, 2015.
[13] Yang, F.W., Goodyer, C.E., Hubbard, M.E. and Jimack, P.K., Parallel implementation of an adaptive, multigrid solver for the implicit solution of nonlinear parabolic systems, with application to a multi-phase-field of tumour growth, Proceedings of the Fourth International Conference on Parallel, Distributed, Grid and Cloud Computing for Engineering, paper 39, editors: P. Ivanyi and B.H.V. Topping, 2015.
[14] Yang, F.W., Goodyer, C.E., Hubbard, M.E. and Jimack, P.K., An Optimally Efficient Technique for the solution of systems of nonlinear parabolic partial differential equations, Advances in Engineering Software, doi:10.1016/j.advengsoft.2016.06.003, In Press, 2016.
[15] Goodyer, C.E., Jimack, P.K., Mullis, A.M., Dong, H.B. and Xie, Y., On the Fully Implicit Solution of a Phase-Field Model for Binary Alloy Solidification in Three Dimensions, Advances in Applied Mathematics and Mechanics, 4, 665-684, 2012.
[16] Deckelnick, K., Dziuk, G. and Elliott C.M., Computation of geometric partial differential equations and mean curvature flow, Acta Numerica, 139-232, 2005.
[17] Trottenberg, U., Oosterlee, C. and Schuller, A., Multigrid, Academic Press, 2001.
[18] Brandt, A., Multi-Level Adaptive Solutions to Boundary-Value Problems, Mathematics of Computation, 31, 333-390, 1977.
[19] Briggs, W.L., Henson, V.E. and McCormick, S.F., A Multigrid Tutorial, Society for Industrial and Applied Mathematics, 2000.

https://doi.org/10.4208/cicp.240715.080716a Published online by Cambridge University Press