Numerical and computational efficiency of solvers for two-phase problems

O. Axelsson[2,3], P. Boyanova[1,4], M. Kronbichler[1], M. Neytcheva[1], X. Wu[1]

[1]Uppsala University, Box 337, 751 05 Uppsala, Sweden
[2]King Abdulaziz University, Jeddah, Saudi Arabia
[3]IG – AS, Studentska 1768, 708 00, Ostrava-Poruba, Czech Republic
[4]IICT – BAS, Acad. G. Bonchev Str., bl. 25A, 1113 Sofia, Bulgaria

Abstract

We consider two-phase flow problems, modelled by the Cahn-Hilliard equation. In our work, the nonlinear fourth-order equation is decomposed into a system of two second-order equations for the concentration and the chemical potential.

We analyse solution methods based on an approximate two-by-two block factorization of the Jacobian of the nonlinear discrete problem. We propose a preconditioning technique that reduces the problem of solving the non-symmetric discrete Cahn-Hilliard system to the problem of solving systems with symmetric positive definite matrices where off-the-shelf multilevel and multigrid algorithms are directly applicable. The resulting solution methods exhibit optimal convergence and computational complexity properties and are suitable for parallel implementation.

We illustrate the efficiency of the proposed methods by various numerical experiments, including parallel results for large scale three dimensional problems.
Keywords Cahn-Hilliard, preconditioning, inexact Newton method, parallel solver

1 Introduction

Multiphase phenomena are observed everywhere in nature, like gas bubbles in oil, ice melting, wet steam, crystal growth, spinodal decomposition. Multiphase processes play an important role in practical problems in areas of modern technological industries, biology, medicine, and environmental remediation, to name a few. Relevant applications are found in aerospace, atmospheric, biological, chemical, civil, mechanical, and nuclear systems. However, due to the highly complex nature of multiphase flows, the fundamental understanding of the integrated fluid physics for these phenomena is still not as well developed as those for single-phase flows. Therefore, numerical simulations can play a critical role in this area.

In this paper, we consider two-phase problems as modelled by the phase-field method, also known as the diffuse interface model. It represents interfaces by thin but nonzero thickness regions, where the density, viscosity and other physical quantities undergo rapid but smooth transitions. In order to distinguish one phase from the other, an order parameter, c, is introduced. It assumes distinct values in the bulk phases (for two-phase flow, $c = 1$ in one bulk phase and $c = -1$ in the other), and varies continuously over a thin interfacial region ($-1 < c < 1$). The location of the interface can be defined as the collection of all points where the phase field variable is in a certain range of values, $|c| \leq \delta$, $\delta \ll 1$. The most appealing feature of the phase-field method is that, since the solution of all governing equations does not require any knowledge of an a priori location of the interfaces in the entire computational domain, interface coalescence and separation can be captured naturally without any special procedures (cf. [29, 6]). In order to properly model relevant physical phenomena, the interface layer is usually considered to be quite thin. Due to the need of high resolution in numerical simulations, the discrete problem can often be of a huge size, especially in 3D. Thus, in practical applications, the use of numerically and computationally efficient solution methods becomes crucial.

The major mathematical tool used in the phase-field model is the Cahn-Hilliard (C-H) equation (cf. [11, 12, 25]), which, in order to account for fluid motion, is coupled to the Navier-Stokes (N-S) equation, see e.g. [15, 30, 21, 20]. The C-H problem is a fourth order partial differential equation for the concentration c, that in the case of two phases can be written in the following form:

$$\begin{aligned} \frac{\partial c}{\partial t} + (\mathbf{u}\cdot\nabla)c - \omega\Delta(\Psi'(c) - \varepsilon^2\Delta c) &= 0,\ (\mathbf{x},t) \in \Omega_T \equiv \Omega\times(0,T) \\ \mathbf{n}\cdot\nabla c = 0,\ \ \mathbf{n}\cdot\nabla(\Psi'(c) - \varepsilon^2\Delta c) &= 0,\ \mathbf{x}\in\partial\Omega,\ t\in(0,T) \\ c(\mathbf{x},0) &= c_0(\mathbf{x}),\ \mathbf{x}\in\Omega. \end{aligned} \tag{1}$$

The usual way to solve the C-H – N-S system is via operator splitting and whenever the C-H equation has to be solved, the velocity field, $\mathbf{u}$, is considered to be known. The thickness of the interfaces is controlled by the parameter ε, that is a small positive constant. In order to resolve the interfaces in the numerical simulations, the spatial step size should be chosen as $h = \varepsilon/r$, $r > 2$. In our studies we consider a common form of the double-well potential function $\Psi(c)$, namely $\Psi(c) = \frac{1}{4}(c^2-1)^2$. In the above formulation, ω is a problem parameter that is typically equal to one when the process of phase separation and coarsening is driven by diffusion only, as in models used in metallurgy. In such models there is no convection due to fluid flow and the velocity field, $\mathbf{u}$, is equal to zero. For two-phase flows $\omega = 1/Pe$, where Pe is the Peclet number. This so-called mobility ω describes the relative strength of diffusive mass transport, compared to convective mass transport.

Problem (1) can be rewritten in an equivalent form, in terms of a coupled system of two second order partial differential equations, see e.g. [16, 19], for both the concentration c and the

chemical potential $\eta = \Psi'(c) - \varepsilon^2 \Delta c$,

$$\begin{aligned} \eta - \Psi'(c) + \varepsilon^2 \Delta c &= 0, \quad (\mathbf{x}, t) \in \Omega_T \\ -\omega \Delta \eta + \frac{\partial c}{\partial t} + (\mathbf{u} \cdot \nabla) c &= 0, \quad (\mathbf{x}, t) \in \Omega_T \\ \mathbf{n} \cdot \nabla c = \mathbf{n} \cdot \nabla \eta &= 0, \quad \mathbf{x} \in \partial\Omega,\ t \in (0, T) \\ c(\mathbf{x}, 0) &= c_0(\mathbf{x}),\ \mathbf{x} \in \Omega. \end{aligned} \tag{2}$$

To discretize (2) in space, we use a finite element method. In this work we use piecewise linear basis functions for the two unknown functions c and η. When applying the backward Euler method, the fully discretized system takes the following form

$$\begin{aligned} M\boldsymbol{\eta}^k - \mathbf{f}(\mathbf{c}^k) - \varepsilon^2 K \mathbf{c}^k &= 0, \\ \omega \Delta t_k K \boldsymbol{\eta}^k + M\mathbf{c}^k + \Delta t_k W \mathbf{c}^k - M\mathbf{c}^{k-1} &= 0, \end{aligned} \tag{3}$$

where M and K are mass and stiffness matrices, W is the matrix resulting from discretizing the convection term, and $\mathbf{c}^k, \boldsymbol{\eta}^k$ are the unknown finite element vectors at a time step $t_k = t_0 + k\Delta t$. The nonlinear term $\mathbf{f}(\mathbf{c}^k)$ is a vector with components

$$f_i(\mathbf{c}^k) = \int_\Omega \Psi' \left(\sum_{l=1}^{N} c_l^k \chi_l(\mathbf{x}) \right) \chi_i(\mathbf{x}) d\mathbf{x}, \tag{4}$$

where we have denoted the finite element basis functions by $\chi_i(\mathbf{x}), i = 1, ..., N$.

Remark 1.1 The backward Euler method and fixed time step Δt in this paper are chosen only for notational simplicity. The ideas, presented in this paper, are applicable for the more general θ-method for any value $\theta \in (0, 1]$, and also for adaptive time stepping algorithms.

Different linearization techniques can be applied in order to solve the nonlinear system (3). Here, we mention two possible approaches.

N1. Applying Newton method. System (3) can be written in vector notations as

$$\mathbf{F}(\mathbf{y}^k) = 0,$$

where $\mathbf{y}^k = ((\boldsymbol{\eta}^k)^T, (\mathbf{c}^k)^T)^T$ is the combined vector of unknowns, and $\mathbf{F}(\mathbf{y}^k)$ is a vector-valued function with components $F_i(\mathbf{y}^k)$ that correspond to the i-th row in the system, $i = 1, ..., 2N$. The exact form of the Jacobian $\mathcal{J}(\mathbf{y})$ of $\mathbf{F}(\mathbf{y})$ can be derived in a straightforward manner. It can be shown that $\mathcal{J}(\mathbf{y})$ has the following block two-by-two form (see also [9]),

$$\mathcal{J}(\mathbf{y}) = \begin{bmatrix} M & -J(\mathbf{c}) - \varepsilon^2 K \\ \omega \Delta t K & M + \Delta t W \end{bmatrix}, \tag{5}$$

where $J(\mathbf{c})$ is the Jacobian of the nonlinear term only. It follows from (4) that $J(\mathbf{c}) = \sum_{e \in \mathcal{T}} j_e M_e$, where M_e are the element mass matrices and j_e are coefficients depending on the concentration

$\mathbf{c}$, $-1 \leq j_e \leq 2$. Note that when applying the classical Newton method, one has to solve systems with the matrix (5), computed for the concentration at a previous Newton step.

N2. Using a linearly stabilized splitting scheme. A key idea in using gradient stable time marching schemes is to divide the nonlinear terms into a stabilizing and a growth term that are then separated across the time step. For certain choices of the splitting, the resulting problem for the approximate solution at the new time step is linear, see e.g [17, 18], where linear schemes for the non-convective C-H problem are discussed. The matrix of the system to be solved in those cases has the same structure as (5) but with a different block instead of $J(\mathbf{c})$. However, the block has similar algebraic form – it can be assembled from element mass matrices, multiplied by coefficients, that are either constant or depend on the concentration at the previous time step.

Based on the approaches N1 and N2, the construction of numerically and computationally efficient linear solvers for two-phase problems involves some efficient treatment of systems with nonsymmetric matrices of the form (5). In Section 2 we describe an efficient preconditioner for such systems. Section 3 presents two inexact Newton methods, based on approximating the exact Jacobian of the nonlinear problem using this preconditioner. In Section 4 we discuss the parallel implementation of the algorithms. In Section 5 the behaviour of the methods is illustrated by a series of numerical experiments, both for two and three space dimensions. Some concluding remarks are given in Section 6.

2 Efficient preconditioning techniques

Consider the matrix

$$A = \begin{bmatrix} M & -J - \varepsilon^2 K \\ \delta K & M + \Delta t W \end{bmatrix}, \tag{6}$$

where $\delta = \omega \Delta t$. It is shown in [4, 10], that under certain relations between the time and spatial discretization steps, Δt and h, neglecting the convection and nonlinear terms ($\Delta t W$ and J, respectively) results in a high quality approximation A_0 of A,

$$A_0 = \begin{bmatrix} M & -\varepsilon^2 K \\ \delta K & M \end{bmatrix}.$$

In [4, 10], the derivations of the conditions on Δt related to h are based on the properties of the finite element matrices in two space dimensions. Further, it is also shown that, in turn, A_0 can be preconditioned in an optimal manner by a matrix $\widehat{A}_0$ of the following form

$$\widehat{A}_0 = \begin{bmatrix} M & -\varepsilon^2 K \\ \delta K & M + 2\varepsilon\sqrt{\delta} K \end{bmatrix}. \tag{7}$$

The spectral equivalence of A_0 and $\widehat{A}_0$ is a general result, related to the structure of A_0, and is relevant for any matrix of such a structure.

We show next that we can use $\widehat{A}_0$ directly to approximate A. This turns out to be very efficient due to two important factors, namely the good approximation properties of $\widehat{A}_0$ and the computational efficiency of solving systems with it, that we discuss in turn.

Approximation properties

It follows from (6) and (7) that

$$A = \widehat{A}_0 + \begin{bmatrix} 0 & -J \\ 0 & \Delta t W - 2\varepsilon\sqrt{\delta}K \end{bmatrix},$$

and, after a series of equivalence transformations, it can be shown that the preconditioned matrix has the form

$$\widehat{A}_0^{-1}A = I + \begin{bmatrix} 0 & -M^{-1}J + \varepsilon^2 M^{-1}KQ \\ 0 & Q \end{bmatrix}, \tag{8}$$

where

$$Q = (I + \varepsilon\sqrt{\delta}M^{-1}K)^{-2}(\Delta t M^{-1}W - 2\varepsilon\sqrt{\delta}M^{-1}K + \delta M^{-1}KM^{-1}J).$$

The eigenvalues of $\widehat{A}_0^{-1}A$ can be seen as perturbations of the unit value. They are real if and only if the eigenvalues of Q are real. We show that this holds if $W = 0$ and $J = 0$. To this end it is convenient to make a similarity transformation of Q, that is, $\widetilde{Q} = M^{\frac{1}{2}}QM^{-\frac{1}{2}}$. It is readily seen that

$$\widetilde{Q} = (I + \varepsilon\sqrt{\delta}\widetilde{K})^{-2}(\Delta t\widetilde{W} - 2\varepsilon\sqrt{\delta}\widetilde{K} + \delta\widetilde{K}\widetilde{J}),$$

where $\widetilde{K} = M^{-\frac{1}{2}}KM^{-\frac{1}{2}}$, $\widetilde{W} = M^{-\frac{1}{2}}WM^{-\frac{1}{2}}$ and $\widetilde{J} = M^{-\frac{1}{2}}JM^{-\frac{1}{2}}$.

If $W = 0$ and $J = 0$ then the eigenvalues ξ of $\widetilde{Q}$, and hence of Q, equal $\xi = -\dfrac{2\mu}{(1+\mu)^2} = -\dfrac{\sigma}{1+\sigma}$, where μ is an eigenvalue of $\varepsilon\sqrt{\delta}\widetilde{K}$ and $\sigma = \dfrac{2\mu}{1+\mu^2}$. It follows that $0 < \sigma \leq 1$ and $0 \geq \xi \geq -\dfrac{1}{2}$. Hence the eigenvalues λ of $\widehat{A}_0^{-1}A$ satisfy $\dfrac{1}{2} \leq \lambda \leq 1$.

For a nonzero term W and Jacobian matrix J, we obtain the estimate

$$\|\widetilde{Q}\| \leq \Delta t\frac{\|\widetilde{W}\|}{\|(I + \varepsilon\sqrt{\delta}\widetilde{K})^2\|} + 2\varepsilon\sqrt{\delta}\frac{\|\widetilde{K}\|}{\|(I + \varepsilon\sqrt{\delta}\widetilde{K})^2\|} + \delta\frac{\|\widetilde{K}\|\|\widetilde{J}\|}{\|(I + \varepsilon\sqrt{\delta}\widetilde{K})^2\|}.$$

Since $\widetilde{K}$ is symmetric and positive definite and $\|(I + \varepsilon\sqrt{\delta}\widetilde{K})^2\| \geq 4\varepsilon\sqrt{\delta}\|\widetilde{K}\|$, it holds

$$\|\widetilde{Q}\| \leq \frac{1}{2} + \Delta t\|\widetilde{W}\| + \frac{1}{4}\frac{\sqrt{\omega\Delta t}}{rh}\|\widetilde{J}\| \leq 1/2 + \zeta, \tag{9}$$

where we recall that $\delta = \omega\Delta t$, $h = \varepsilon/r$, and denote $\zeta = \Delta t\|\widetilde{W}\| + \dfrac{1}{4}\dfrac{\sqrt{\omega\Delta t}}{rh}\|\widetilde{J}\|$. Due to the properties of the matrix J, it holds that $\|\widetilde{J}\| = O(1)$. It follows that we can control the eigenvalues of $\widehat{A}_0^{-1}A$ to be small perturbations of the unit eigenvalues by choosing Δt sufficiently small. Based on the estimate (9), we formulate the following proposition for the spectrum of the preconditioned matrix (8).

Proposition 1 *Half of the eigenvalues λ of the preconditioned matrix $\widehat{A}_0^{-1}A$ are equal to one, and the other half belong to a disc, centred at one, with a radius $1/2 + \zeta$, where,*

(1) for diffusion driven problems ($\omega = 1$), ζ is close to zero, when $\Delta t < h^2$,

(2) for convection-diffusion problems ($\omega = 1/Pe, Pe \gg 1$), ζ is close to zero, when $\Delta t < h$.

Remark 2.1 Proposition 1 is based on spectral and norm properties of the finite element matrices. The presented result is true, for example, for bilinear finite elements in two dimensions and trilinear finite elements in three dimensions. These elements are used for the numerical experiments in this article. However, the analysis technique is applicable for any choice of finite element basis functions.

Computational efficiency

Now we consider the solution of systems with the matrix $\widehat{A}_0$,

$$\widehat{A}_0 \begin{bmatrix} \mathbf{x}_1 \\ \mathbf{x}_2 \end{bmatrix} = \begin{bmatrix} M & -\varepsilon^2 K \\ \delta K & M + 2\varepsilon\sqrt{\delta} K \end{bmatrix} \begin{bmatrix} \mathbf{x}_1 \\ \mathbf{x}_2 \end{bmatrix} = \begin{bmatrix} \mathbf{f}_1 \\ \mathbf{f}_2 \end{bmatrix}. \tag{10}$$

Since $\widehat{A}_0$ has the following exact block factorization (cf. [2, 4])

$$\widehat{A}_0 = \begin{bmatrix} I & 0 \\ \delta K M^{-1} & I + \varepsilon\sqrt{\delta} K M^{-1} \end{bmatrix} \begin{bmatrix} M & 0 \\ 0 & M \end{bmatrix} \begin{bmatrix} I & -\varepsilon^2 M^{-1} K \\ 0 & I + \varepsilon\sqrt{\delta} M^{-1} K \end{bmatrix},$$

the inverse $\widehat{A}_0^{-1}$ has the following explicit form:

$$\begin{aligned}
\widehat{A}_0^{-1} &= \begin{bmatrix} I & \varepsilon^2 M^{-1} K (I + \varepsilon\sqrt{\delta} M^{-1} K)^{-1} \\ 0 & (I + \varepsilon\sqrt{\delta} M^{-1} K)^{-1} \end{bmatrix} \begin{bmatrix} M^{-1} & 0 \\ 0 & M^{-1} \end{bmatrix} \times \\
&\quad \begin{bmatrix} I & 0 \\ -\delta K M^{-1} (I + \varepsilon\sqrt{\delta} K M^{-1})^{-1} & (I + \varepsilon\sqrt{\delta} K M^{-1})^{-1} \end{bmatrix} \\
&= \begin{bmatrix} M^{-1} & \dfrac{\varepsilon}{\sqrt{\delta}} (I - (I + \varepsilon\sqrt{\delta} M^{-1} K)^{-1}) M^{-1} \\ 0 & (M + \varepsilon\sqrt{\delta} K)^{-1} \end{bmatrix} \times \\
&\quad \begin{bmatrix} I & 0 \\ -\dfrac{\sqrt{\delta}}{\varepsilon} (I - (I + \varepsilon\sqrt{\delta} K M^{-1})^{-1}) & M(M + \varepsilon\sqrt{\delta} K)^{-1} \end{bmatrix} \\
&= \begin{bmatrix} H(2I - MH) & \dfrac{\varepsilon}{\sqrt{\delta}} (I - HM) H \\ -\dfrac{\sqrt{\delta}}{\varepsilon} H (I - MH) & HMH \end{bmatrix},
\end{aligned}$$

where $H = (M + \varepsilon\sqrt{\delta} K)^{-1}$.

Then, the exact solution of (10) is obtained as

$$\begin{bmatrix}\mathbf{x}_1\\ \mathbf{x}_2\end{bmatrix} = \widehat{A}_0^{-1}\begin{bmatrix}\mathbf{f}_1\\ \mathbf{f}_2\end{bmatrix} = \begin{bmatrix} H\left[(2\mathbf{f}_1 - MH\mathbf{f}_1) + \dfrac{\varepsilon}{\sqrt{\delta}}(\mathbf{f}_2 - MH\mathbf{f}_2)\right] \\ -H\left[\dfrac{\varepsilon}{\sqrt{\delta}}(\mathbf{f}_1 - MH\mathbf{f}_1) - MH\mathbf{f}_2\right] \end{bmatrix}.$$

We perform a series of further transformations to simplify the expression for the solution,

$$\begin{aligned}\begin{bmatrix}\mathbf{x}_1\\ \mathbf{x}_2\end{bmatrix} &= \begin{bmatrix} \dfrac{\varepsilon}{\sqrt{\delta}}H\left[\dfrac{\sqrt{\delta}}{\varepsilon}(2\mathbf{f}_1 - MH\mathbf{f}_1) + (\mathbf{f}_2 - MH\mathbf{f}_2)\right] \\ -H\left[\dfrac{\sqrt{\delta}}{\varepsilon}(\mathbf{f}_1 - MH\mathbf{f}_1) - MH\mathbf{f}_2\right] \end{bmatrix} \\ &= \begin{bmatrix} \dfrac{\varepsilon}{\sqrt{\delta}}H\left[\dfrac{\sqrt{\delta}}{\varepsilon}\mathbf{f}_1 + \mathbf{f}_2\right] + \dfrac{\varepsilon}{\sqrt{\delta}}H\left[\dfrac{\sqrt{\delta}}{\varepsilon}\mathbf{f}_1 - MH\left(\dfrac{\sqrt{\delta}}{\varepsilon}\mathbf{f}_1 + \mathbf{f}_2\right)\right] \\ -H\left[\dfrac{\sqrt{\delta}}{\varepsilon}\mathbf{f}_1 - MH\left(\dfrac{\sqrt{\delta}}{\varepsilon}\mathbf{f}_1 + \mathbf{f}_2\right)\right] \end{bmatrix} \\ &= \begin{bmatrix} \dfrac{\varepsilon}{\sqrt{\delta}}(\mathbf{g} + \tilde{\mathbf{g}}) \\ -\tilde{\mathbf{g}} \end{bmatrix},\end{aligned} \tag{11}$$

where $\mathbf{g} = (M + \varepsilon\sqrt{\delta}K)^{-1}\left(\dfrac{\sqrt{\delta}}{\varepsilon}\mathbf{f}_1 + \mathbf{f}_2\right)$ and $\tilde{\mathbf{g}} = (M + \varepsilon\sqrt{\delta}K)^{-1}\left(\dfrac{\sqrt{\delta}}{\varepsilon}\mathbf{f}_1 - M\mathbf{g}\right)$.

We see from (11) that the solution of system (10) can be computed by the following steps:

1. Compute $\mathbf{b}_1 = \dfrac{\sqrt{\delta}}{\varepsilon}\mathbf{f}_1 + \mathbf{f}_2$;
2. Solve $(M + \varepsilon\sqrt{\delta}K)\,\mathbf{g} = \mathbf{b}_1$;
3. Compute $\mathbf{b}_2 = M\mathbf{g} - \dfrac{\sqrt{\delta}}{\varepsilon}\mathbf{f}_1$;
4. Solve $(M + \varepsilon\sqrt{\delta}K)\,\mathbf{x}_2 = \mathbf{b}_2$;
5. Compute $\mathbf{x}_1 = \dfrac{\varepsilon}{\sqrt{\delta}}(\mathbf{g} - \mathbf{x}_2)$.

Thus, we only need to solve two systems with $M + \varepsilon\sqrt{\delta}K$, perform one matrix multiplication with M and three vector updates. Then, the computational complexity $\mathcal{N}(\widehat{A}_0)$ of solving systems with the preconditioner $\widehat{A}_0$ is

$$\mathcal{N}(\widehat{A}_0) = (3 + d_M)N + 2\mathcal{N}(M + \varepsilon\sqrt{\delta}K),$$

where d_M is the maximum number of nonzero elements per row in the mass matrix M, and $\mathcal{N}(M + \varepsilon\sqrt{\delta}K)$ is the computational complexity of solving a system with the symmetric positive definite matrix $M + \varepsilon\sqrt{\delta}K$. Multilevel and multigrid methods can be applied at steps 2 and 4 in the algorithm. This leads to optimal computational complexity $\mathcal{N}(\widehat{A}_0)$, namely, linearly proportional to the number of degrees of freedom. Note that, for constant meshes, the construction of a multigrid-type preconditioner is done only once for the whole solution process.

Remark 2.2 Note that the matrix $M + \varepsilon\sqrt{\delta}K$ is more diagonally dominant than the stiffness matrix K alone. Moreover, for certain problem parameters, $M + \varepsilon\sqrt{\delta}K$ can exhibit properties similar to those of a mass matrix. Therefore, to save computational effort in each iteration step, using an unpreconditioned conjugate gradient method, even though not guaranteed to be optimal, can lead to efficient computations. This effect is illustrated numerically in Section 5.

3 Inexact Newton methods

To solve the arising nonlinear system, we use a Newton type method. Provided that the Jacobian is invertible and Lipschitz continuous, and $\mathbf{y}_s$ is sufficiently close to the exact solution $\mathbf{y}^*$, Newton's method usually converges rapidly, namely, it holds

$$\|\mathbf{y}_{s+1} - \mathbf{y}^*\| \leq L\|\mathbf{y}_s - \mathbf{y}^*\|^2,$$

where L is a constant.

The general idea of Newton's method is to linearize $\mathbf{F}$ around a current guess, $\mathbf{y}_s$, and compute a correction $\boldsymbol{\delta}_s$,

$$\mathbf{F}'(\mathbf{y}_s)\boldsymbol{\delta}_s = -\mathbf{F}'(\mathbf{y}_s), \tag{12}$$

expecting that $\mathbf{y}_{s+1} = \mathbf{y}_s + \boldsymbol{\delta}_s$ is a better approximation of the solution of the nonlinear problem than $\mathbf{y}_s$. The method is attractive due to its rapid convergence from any sufficiently good initial guess $\mathbf{y}_0$. However, there are two drawbacks with this method. First, at every Newton iteration, we have to solve a system with the Jacobian of $\mathbf{F}$ exactly. In the case of large-scale problems, this can be very expensive. Second, when $\mathbf{y}_0$ is not sufficiently close to $\mathbf{y}^*$ there is no guarantee for achieving global convergence.

When the system with the Jacobian matrix (12) is solved exactly, the method is referred to as *exact* Newton method. When the correction is obtained by solving (12) inexactly (up to some proper accuracy), or when (12) is replaced by a system with a matrix that approximates the exact Jacobian $\mathbf{F}'(\mathbf{y}_s)$, the method is referred to as *inexact* Newton method.

In this work, the nonlinear operator $\mathbf{F}$, which we deal with, is differentiable, that is, $\mathbf{F}'$ exists and is nonsingular. We assume that we possess a good enough initial guess $\mathbf{y}_0$. We consider the following two variants of an inexact Newton method:

(i1) Solve (12) by a preconditioned iterative method. The algorithm takes the form:

Given $\mathbf{y}_0$,
For $s = 0, 1, 2 \cdots$ **do** until convergence
Compute $\boldsymbol{\delta}_s$ as a solution of the system

(i1-1) $\mathbf{F}'(\mathbf{y}_s)\boldsymbol{\delta}_s = -\mathbf{F}(\mathbf{y}_s) + \mathbf{r}_s$,
where $\dfrac{\|\mathbf{r}_s\|}{\|\mathbf{F}(\mathbf{y}_s)\|} \leq \tau_s$, $\tau_s \in (0, 1)$

(i1-2) Update $\mathbf{y}_{s+1} = \mathbf{y}_s + \varsigma_s \boldsymbol{\delta}_s$, $\varsigma_s \in (0, 1]$

(i2) Replace $\mathbf{F}'(\mathbf{y}_s)$ by an approximation $\widehat{\mathbf{F}}'(\mathbf{y}_s)$. Then, the method reads:
Given $\mathbf{y}_0$,
For $s = 0, 1, 2 \cdots$ **do** until convergence
Compute $\boldsymbol{\delta}_s$ as a solution of the system

(i2-1) $\widehat{\mathbf{F}}'(\mathbf{y}_s)\boldsymbol{\delta}_s = -\mathbf{F}(\mathbf{y}_s)$

(i2-2) Update $\mathbf{y}_{s+1} = \mathbf{y}_s + \varsigma_s \boldsymbol{\delta}_s$, $\varsigma_s \in (0, 1]$

As it turns out, for the Cahn-Hilliard problem a proper choice is $\widehat{\mathbf{F}}'(\mathbf{y}_s) = \widehat{A}_0$. Thus, $\widehat{\mathbf{F}}'(\mathbf{y}_s) = \widehat{\mathbf{F}}'$ does not depend on $\mathbf{y}_s$ and we can efficiently solve systems with it to an arbitrary accuracy. We use $\widehat{A}_0$ in our work as a preconditioner for the iterative method in (i1-1) as well as in (i2-1).

The method (i1) is studied, e.g., in [14] and it is shown that for a sufficiently accurate initial approximation $\mathbf{y}_0$, the method is locally convergent. The sequence $\{\tau_s\}$, $s = 1, 2, \cdots$ is referred to as the *forcing sequence*. The choice of τ_s affects local convergence properties. Clearly, for $\tau_s = 0$, we obtain the exact Newton method.

A well-known effect of using inexact Newton methods of type (i1) is that when an iterative method is used in (i1-1), the stopping criterion for the iterations may be chosen much smaller than that needed to meet the condition on the norm of the scaled residual for a given τ_s. Further discussion on this issue falls out of the scope of this work.

Consider the inexact Newton method (i1). Assume that to solve systems with $\mathbf{F}'(\mathbf{y}_s)$ we use a preconditioned iterative method with a preconditioner $\widehat{\mathbf{F}}'$, which is nonsingular and for which the condition number of $\widehat{\mathbf{F}}'^{-1}\mathbf{F}'(\mathbf{y}_s)$ is much smaller than that of $\mathbf{F}'(\mathbf{y}_s)$. The system is modified as below:

$$\widehat{\mathbf{F}}'^{-1}\mathbf{F}'(\mathbf{y}_s)\boldsymbol{\delta}_s = -\widehat{\mathbf{F}}'^{-1}\mathbf{F}(\mathbf{y}_s) + \widehat{\mathbf{r}}_s\,,$$

where the scaled residual is $\widehat{\mathbf{r}}_s = \widehat{\mathbf{F}}'^{-1}(\mathbf{F}'(\mathbf{y}_s)\boldsymbol{\delta}_s + \mathbf{F}(\mathbf{y}_s))$, $\dfrac{\|\widehat{\mathbf{r}}_s\|}{\|\widehat{\mathbf{F}}'^{-1}\mathbf{F}(\mathbf{y}_s)\|} \leq \tau_s$, $\tau_s \in (0, 1)$.

Globally convergent inexact Newton methods have been studied, e.g., in [1, 27, 8]. The global convergence is derived for the so-called *damped inexact Newton method*, for which, as indicated in (i1-2) and (i2-2), the solution is updated as

$$\mathbf{y}_{s+1} = \mathbf{y}_s + \varsigma_s \boldsymbol{\delta}_s,$$

where $\varsigma_s \in (0, 1]$ is the so-called *damping parameter*. The meaning of $\varsigma_s < 1$ is that when we are far away from the exact solution, we could better take a shorter step along the so-computed search

direction $\boldsymbol{\delta}_s$, since it is obtained via an inexact Newton method. However, in our experiments, due to the observed fast convergence, we use $\varsigma_s = 1$.

Inexact Newton methods of type (i1) are studied in [1, 27]. The usual assumptions in the analysis are that $\mathbf{F}$ is Fréchet-differentiable and $\mathbf{F}'$ is Lipschitz-continuous. Then, for a linear, uniformly bounded preconditioner $\widehat{\mathbf{F}}'$, $\|\widehat{\mathbf{F}}'\| \leq \alpha$, it holds

$$\|\widehat{\mathbf{F}}'^{-1}\mathbf{F}'(\mathbf{a}) - \widehat{\mathbf{F}}'^{-1}\mathbf{F}'(\mathbf{b})\| \leq \|\widehat{\mathbf{F}}'^{-1}\|\|\mathbf{F}'(\mathbf{a}) - \mathbf{F}'(\mathbf{b})\| \leq \frac{1}{\alpha}L\|\mathbf{a} - \mathbf{b}\|,$$

where L is the corresponding Lipschitz constant. Thus, the preconditioned operator $\widehat{\mathbf{F}}'^{-1}\mathbf{F}'$ is Hölder-continuous, and the inexact Newton method is globally convergent, see e.g. [1]. Actually, a smaller bound can be derived if we take the Lipschitz constant of $\widehat{\mathbf{F}}'^{-1}\mathbf{F}'(\mathbf{y})$ instead. Furthermore, this constant is independent of any affine mapping part of $\mathbf{F}(\mathbf{y})$, see e.g. [1].

Inexact Newton methods of type (i2) are analysed in [8]. Under the same assumptions for the properties of the operators, as in the case of method (i1), global convergence is shown when

$$\|\mathbf{F}'(\mathbf{y}_s) - \widehat{\mathbf{F}}'\| \leq \alpha\|\widehat{\mathbf{F}}'^{-1}\mathbf{F}'(\mathbf{y}_s) - I\| \leq \alpha\rho, \rho < 1. \tag{13}$$

The analysis in Section 2 shows that (13) holds true for the C-H problem with the considered preconditioner $\widehat{\mathbf{F}}' = \widehat{A}_0$.

Under the above conditions and suitable choice of ς_s, both methods (i1) and (i2) are globally convergent and retain a superlinear rate of convergence.

Remark 3.1 As has been shown e.g. in [3] and [5], an alternative choice of an inexact Newton method can be based on using two meshes in space, a coarse and a fine mesh. Thereby, at each iteration step one solves the non-linear problem on the coarse mesh, interpolates the coarse-mesh solution to the fine mesh and performs some (few) non-linear iterations on the fine mesh. When one is interested mainly in a small L_2-norm of the error rather than in a small residual norm then the coarse mesh solution gives a sufficiently accurate approximation on the fine mesh, and it suffices with only few (often just one) of the more costly non-linear iterations on the fine mesh. In our problem, when thin interfaces must be resolved, we expect, however, that the coarse mesh solution will not be accurate enough. Therefore, we do not consider the two-level method any further in this paper.

4 Parallel implementation

The main procedures that constitute the solver for the C-H equation are the following:

- Assembly of matrices;
- Matrix-vector multiplications and vector operations within the nonlinear and linear solution methods;

- Preconditioned iterative methods to be applied for the linear systems with a symmetric positive definite matrix $M + \varepsilon\sqrt{\delta}K$, and in case of method (i1), for the systems with the nonsymmetric matrix A.

For all of these tasks, there are known techniques for parallelization, as well as intensive ongoing research and implementation of methods, accessible via numerous publicly available software packages (see e.g. [23, 26, 13, 28]). For the numerical tests, presented in this article, we use the open software libraries deal.II and Trilinos, see [13, 28]. The implementation is based on a fully distributed mesh oriented paradigm, that allows programs to scale to large machine and problem sizes, see [7] for examples also for other problems.

5 Numerical results

We use the following problems to evaluate the proposed preconditioning techniques for the Cahn-Hilliard equation in two and three space dimensions:

Problem 1: Only diffusion

Consider the system (1) in $\Omega_T = [0,1] \times [0,1] \times [0,1] \times [0,T]$ with parameters $\omega = 1, \varepsilon = 0.0625$ and $\mathbf{u} = (0,0,0)$. The process of phase separation and coarsening takes place only due to diffusion. Figure 1 illustrates the evolution of a binary mixture in time.

Problem 2: Diffusion combined with convection

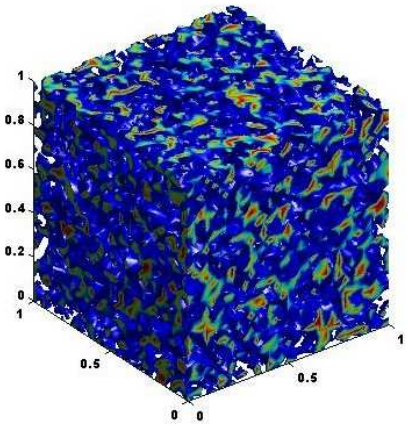
Consider the system in $\Omega_T = [-0.5, 0.5] \times [0,1] \times [0,1] \times [0,T]$ with parameters $\omega = 1/300, \varepsilon = 0.1$ and $\mathbf{u} = (1,0,0)$. The initial condition is assumed to be $C_0 = -\tanh(10x_1)$. Figure 2 illustrates the movement of the front between the two phases in the direction of the horizontal wind.

Problem 3: A rising bubble

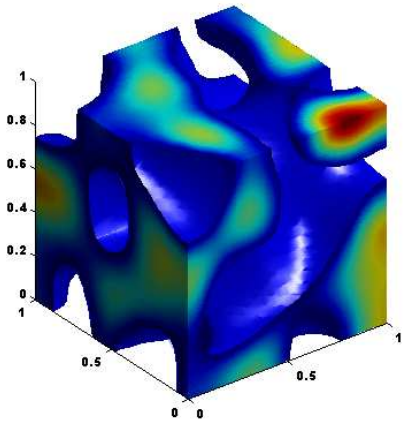
We consider the rise of light bubble according to the benchmark study defined in [24], which consists of solving the coupled N-S – C-H system in $\Omega_T = [0,1] \times [0,2] \times [0,T]$ with parameters $\omega = 2000, \varepsilon = 0.009$. The initial condition is taken as $\tanh(d/\varepsilon)$, where d is the distance to the initial interface. The initial interface between the phases, as well as its evolution in time, is presented in Figure 3.

To solve Problem 1 and 2, we use the proposed inexact Newton methods. The numerical tests are performed both in Matlab and in C++ using deal.II (see [13]), for the case of trilinear conforming finite elements, see also [31]. The stopping criterion for the Newton iterations at each time step is always taken to be $||\Delta \mathbf{y}_s^k|| < 10^{-6}$. In Matlab, the iterative method used to solve the arising system is the Generalised Conjugate Gradient - Minimal Residual (GCG-MR) method, and the solution process is stopped when the norm of the residual is reduced by a factor 10^{-6} or the norm itself is smaller than 10^{-12}. In deal.II, the iterative method used to solve the arising systems is GMRES with left preconditioning.

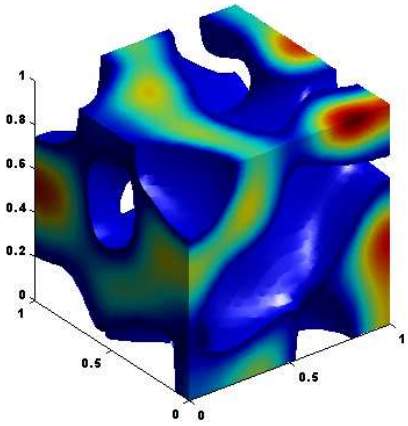
Results for the convergence of the nonlinear method, as well as of the inner linear solvers, are presented in Tables 1-7. We report number of iterations, averaged over five time steps. Each table cell contains two or three integer digits of the form N_1/N_2 or $N_1/N_2/N_3$, where N_1 denotes the average number of Newton iterations per time step, N_2 is the average number of GCG-MR ite-



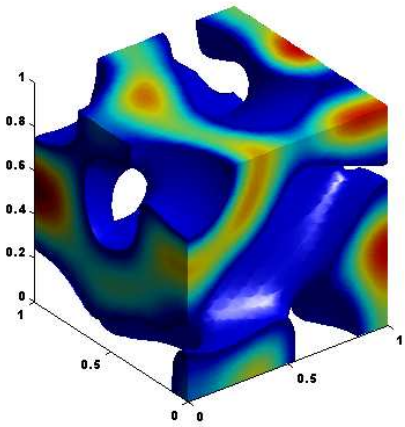
(a) Initial condition



(b) $t = 0.01875$

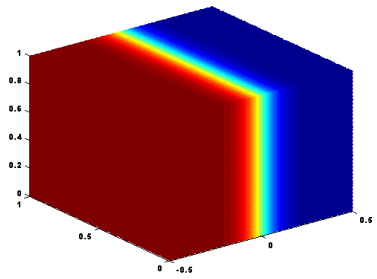


(c) $t = 0.0375$

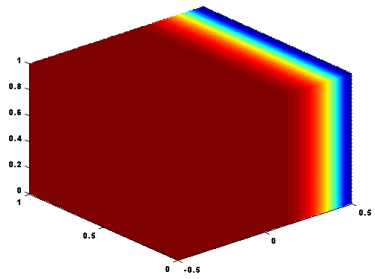


(d) $t = 0.065625$

Figure 1: P1: Phase separation and coarsening due to diffusion



(a) Initial front



(b) $t = 0.4687$

Figure 2: P2: Front movement due to convection

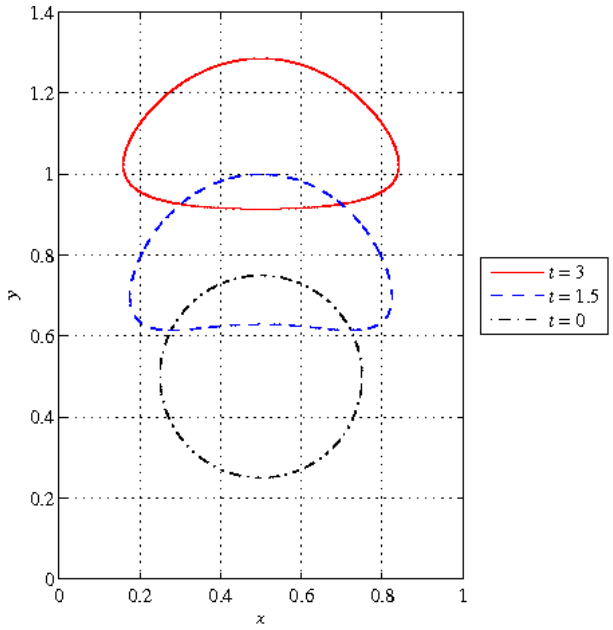


Figure 3: Problem 3: A rising bubble

	Δt				
Size	h	$h/2$	$h/4$	$h/10$	h^2
			$Matlab$		
9826	> 50	> 50	8 / 41	3 / 12	2 / 10
71874	> 50	> 50	3 / 11	2 / 10	2 / 9
549250	> 50	> 50	3 / 8	3 / 7	3 / 7

Table 1: Problem 1: Iteration counts for method (i1)

	Δt				
Size	h	$h/2$	$h/4$	$h/10$	h^2
			$Matlab$		
9826	> 50	> 50	8 / 44 / 14	3 / 12 / 14	2 / 11 / 13
71874	> 50	> 50	3 / 11 / 21	3 / 9 / 17	3 / 8 / 15
			$deal.II$		
9826	> 50	> 50	> 50	3/ 13 / 12	3 / 11 / 12
71874	> 50	> 50	3 / 17 / 18	3 / 11 / 16	3 / 8 / 13
549250	> 50	> 50	3 / 12 / 31	3 / 11 / 24	3 / 9 / 16

Table 2: Problem 1: Iteration counts for method (i1), CG used for the systems with $M + \varepsilon\sqrt{\beta}K$

	Δt				
Size	h	$h/2$	$h/4$	$h/10$	h^2
			$Matlab$		
9826	> 50	> 50	12 / 58 / 3	2 / 13 / 3	2 / 11 / 3
71874	> 50	> 50	3 / 11 / 3	2 / 10 / 3	2 / 9 / 3
549250	> 50	> 50	3 / 8 / 3	3 / 7 / 3	3 / 7 / 3

Table 3: Problem 1: Iteration counts for method (i1), PCG with AMG preconditioner used for the systems with $M + \varepsilon\sqrt{\beta}K$

rations or GMRES iterations per Newton iteration and N_3, whenever present, shows the average number of AMG-preconditioned conjugate gradient (PCG) iterations or the standard unpreconditioned conjugate gradient (CG) iterations to solve iteratively systems with $M + \varepsilon\sqrt{\delta}K$.

Tables 1 to 3 (Problem 1) and Tables 5 to 7 (Problem 2) illustrate the numerical performance of the inexact Newton method (i1), where we use $\widehat{A}_0$ as a preconditioner for A. First we test the convergence when systems with $M + \varepsilon\sqrt{\delta}K$ are solved via a direct method (Tables 1 and 5), next when these are solved by using unpreconditioned CG method, and finally when PCG is applied with the AMG precondioner HSL-MI20, see [22]. The stopping criterion for the two inner solvers (both CG and PCG) is taken so that the residual is reduced by a factor 10^{-3}. We observe, that using an inexact inner solver for the systems with $M + \varepsilon\sqrt{\delta}K$ does not affect the convergence of the outer solvers. When the optimal AMG preconditioner is used, the overall method exhibits optimal behaviour, namely, the number of iterations do not increase with the problem size. Such behaviour is observed for small enough time steps, $\Delta t = h/4$ for Problem 1 and $\Delta t = h$ for Problem 2.

We also test the inexact Newton method (i2) where the exact Jacobian A is replaced by the matrix $\widehat{A}_0$. The results are presented in Table 4 and 8, where we show the number of nonlinear iterations only. As expected, method (i2) converges in more nonlinear steps than method (i1), however, in this case we save the computational cost of the iterative solver for the nonsymmetric system with A.

Size	Δt				
	h	$h/2$	$h/4$	$h/10$	h^2
			$Matlab$		
9826	> 50	> 50	> 50	35	23
71874	> 50	> 50	47	21	20
			$deal.II$		
9826	> 50	> 50	> 50	37	27
71874	> 50	> 50	50	25	20
549250	> 50	> 50	28	22	18

Table 4: Problem 1: Iteration counts for method (i2)

Size	Δt				
	h	$h/2$	$h/4$	$h/10$	h^2
			$Matlab$		
9826	4 / 10	4 / 9	4 / 7	3 / 8	3 / 7
71874	4 / 7	4 / 6	4 / 6	4 / 6	3 / 6
549250	4 / 6	4 / 6	4 / 5	3 / 6	3 / 6

Table 5: Problem 2: Iteration counts for method (i1)

In order to evaluate the performance of the two proposed Newton methods, we present timing results for the nonlinear solvers on one processor in Table 9, and in parallel in Tables 10 and 11. The numerical experiments are done on a cluster with HP SL170h G6 compute servers. Each compute server, or node, has two Intel Xeon 5520 Quad core (Nehalem 2.26 Ghz, 8MB cache) processors. All nodes are interconnected with a 4:1 oversubscribed DDR Infiniband fabric. We use the optimised version of the deal.II library, which is compiled with aggressive compiler optimisations (C++ compiler flag with $-O2$; C compiler flag with $-O3$).

In Table 9 we collect the average wall time T_w per nonlinear solve for the methods (i1) and (i2), when CG or PCG is used for the systems with $M + \varepsilon\sqrt{\beta}K$. One regular refinement of the mesh in three dimensions, when using brick elements, results in about eight times increase of the problem size. In Tables 9 – 11 we introduce R_w to be the factor $R_w = \dfrac{T_w(h)}{T_w(2h)}$ to measure how the execution time grows from one mesh refinement to the next. As seen in Table 9, R_w is smaller than eight for the case when the optimal AMG preconditioner is used. As expected, using unpreconditioned CG method does not guarantee optimal order overall method, however the elapsed wall time increases with a factor less than 12 for (i1) and less than 9 for (i2).

In order to evaluate the parallelization properties of the methods, we perform tests to see how the solution time varies with the number of processors for a fixed problem size per processor, i.e., a so-called weak scalability test. We consider a problem size on a single core of about half a million. The discrete problem after one regular refinement, with system size of about

Size	Δt				
	h	$h/2$	$h/4$	$h/10$	h^2
			$Matlab$		
9826	4 / 10 / 10	4 / 8 / 9	4 / 7 / 9	3 / 8 / 10	3 / 7 / 11
71874	4 / 9 / 11	4 / 7 / 10	4 / 6 / 10	3 / 7 / 8	3 / 6 / 8
			$deal.II$		
9826	4 / 17 / 7	4 / 13 / 9	4 / 11 / 8	4 / 9 / 10	3 / 10 / 12
71874	4 / 15 / 10	4/ 12 / 9	4 / 10 / 8	3 / 10 / 7	3 / 9 / 9
549250	4 / 13 / 14	4 / 11 / 12	3/ 11 / 11	3 / 10 / 9	3 / 8 / 9

Table 6: Problem 2: Iteration counts for method (i1), CG used for the systems with $M + \varepsilon\sqrt{\beta}K$

Size	Δt				
	h	$h/2$	$h/4$	$h/10$	h^2
			$Matlab$		
9826	4 / 10 / 2	4 / 8 / 2	4 / 7 / 3	3 / 7 / 3	3 / 7 / 3
71874	4 / 9 / 3	4 / 7 / 3	4 / 6 / 3	3 / 7 / 2	3 / 6 / 3
549250	4 / 7 / 3	4 / 6 / 3	4 / 6 / 3	4 / 6 / 3	3 / 6 / 3

Table 7: Problem 2: Iteration counts for method (i1), PCG with AMG preconditioner used for the systems with $M + \varepsilon\sqrt{\beta}K$

Size	Δt				
	h	$h/2$	$h/4$	$h/10$	h^2
			$Matlab$		
9826	34	25	23	21	20
71874	28	25	23	21	18
			$deal.II$		
9826	31	23	19	17	16
71874	28	21	19	16	14
549250	22	20	18	15	10

Table 8: Problem 2: Iteration counts for method (i1)

Size	71874	549250	4293378
		(i1), CG	
Iter. count	3 / 11 / 17	3/ 12 / 24	3 / 10 / 38
T_w	3.9854	39.1209	442.1032
R_w		9.80	11.30
		(i2), CG	
Iter. count	26/15	20/21	18/34
T_w	10.5436	70.6468	574.6676
R_w		6.7	8.14
		(i1), PCG with AMG	
Iter. count	2 / 12 / 5	2 / 11 / 5	2 / 10 / 5
T_w	4.82	36.86	287.40
R_w		7.64	7.79
		(i2), PCG with AMG	
Iter. count	26/4	20/5	18/5
T_w	5.02	33.42	236.97
R_w		6.66	7.09

Table 9: Problem 1: Iteration counts, average wall time T_w in seconds and factor R_w of increase of T_w after one refinement of the mesh, for applying methods (i1) and (i2) on one processor

four million, is solved on eight cores that reside on one compute node. The problem after two refinements is solved on eight nodes, with eight cores each. In that case, communication occurs both on the chip and via the network connecting the compute servers. The largest number of degrees of freedom, that we consider, is about 270 million, and the problem is solved on 64 nodes, i.e., 512 cores.

First, we consider the methods when applying an unpreconditioned inner solver, see Table 10. In this case, all computational modules of the algorithm are straightforwardly parallelizable, since only matrix-vector and vector operations are needed. We observe that the factor of increase R_w of the computational time per nonlinear solve, in the case when the type of communication for two consecutive problem sizes is the same (last columns in the table), is almost equal to the factor of increase of the number of CG iterations R_{CG}. We also see that the method (i2) entails more nonlinear iterations, however it has a lower computational cost than (i1). The larger the size of the matrix (i.e. the more the levels of refinement) is, the more remarkable the advantage of the method (i2) becomes.

As a next step, we perform parallel tests for solving the nonlinear problem using PCG method with AMG preconditioner for the systems with $M + \varepsilon\sqrt{\beta}K$, see Table 11. The resulting algorithms have better execution times than those in Table 10. Also, the advantages of method (i2) are more clearly visible.

We also demonstrate the applicability of the preconditioning techniques, presented in Section 2, for the case when the approach N2 is used to treat the nonlinear problem. We consider Problem

Size	549250	4293378	33949186	270011394
No. cores	1	8	64	512
		(i1)		
Iter. count	3 / 12 / 24	3 / 10 / 38	3 / 11 / 57	3 / 10 / 103
T_w	39.1	136.4	293.9	520.7
R_{CG}		1.32	1.65	1.64
R_w		3.49	2.16	1.77
		(i2)		
Iter. count	20	18	17	14
T_w	70.2	127.4	217.2	355.6
R_w		1.81	1.71	1.64

Table 10: Problem 1: Weak scalability test, CG used for systems with $M + \varepsilon\sqrt{\beta}K$. Iteration counts, average wall time T_w in seconds, factor R_{CG} of increase of the inner CG iterations and factor R_w of increase of T_w after one refinement of the mesh

Size	549250	4293378	33949186	270011394
No. core	1	8	64	512
		(i1)		
Iter. count	3 / 11 / 5	3 / 10 / 6	3 / 9 / 6	3 / 9 / 6
T_w	35.36	91.02	145.66	275.20
R_w		2.57	1.60	1.89
		(i2)		
Iter. count	20/5	17/5	17/6	14/6
T_w	31.45	67.53	136.56	140.59
R_w		2.15	2.02	1.03

Table 11: Problem 1: Weak scalability test, PCG with AMG preconditioner used for systems with $M + \varepsilon\sqrt{\beta}K$. Iteration counts, average wall time T_w in seconds and factor R_w of increase of T_w after one refinement of the mesh

	Δt			
Size	h	$h/2$	$h/4$	$h/10$
	$deal.II$			
103362	5	4	4	4
411522	4	4	4	4
1642242	4	3	3	2

Table 12: Problem 3: Iteration counts for solving the linear systems, resulting from applying approach N2, one AMG V-cycle used for the systems with $M + \varepsilon\sqrt{\beta}K$

3, where the C-H equation is discretized in time using BDF-2 scheme and the splitting of the nonlinear term, proposed in [17]. To solve the resulting linear systems for the unknown vectors at consecutive time steps, we use GMRES solver with preconditioner of the form $\widehat{A}_0$ where the subsystems with the matrix $M + \varepsilon\sqrt{\beta}K$ are solved using one AMG V-cycle. Note that in this case we treat the convection term explicitly. Thus, only the nonlinear term is neglected in the preconditioner $\widehat{A}_0$. For the initial guess $\mathbf{y}_0^k$ in the iterative method we use point-wise linear extrapolation, $\mathbf{y}_0^k = 2\mathbf{y}^{k-1} - \mathbf{y}^{k-2}$. The number of GMRES iterations performed per time step, averaged over 20 time steps, are presented in Table 12.

6 Concluding remarks

The efficiency of the solvers for the numerical simulation of complex processes depends on the utilisation of the properties of the considered model and discrete representation. On the other hand, properties like nonlinearity and non-symmetricity are not always straightforward to deal with. Moreover, when taking into consideration the specifics of the problem at hand, general solution approaches are not applicable, or do not perform sufficiently well.

We have developed and analysed solution methods for the C-H equation, that both utilise to a full extent the algebraic properties of the discrete problem, and reduce its solution to the solution of systems with matrices of a well-known type, namely, diagonally dominant symmetric positive definite matrices. The solvers can be implement in sequential, as well as in parallel, by using existing linear algebra methods and software toolboxes. The numerical experiments confirm to a full extend the theoretical analysis of the proposed algorithms.

7 Acknowledgements

The work of the second author (fully) and the fourth and the fifth author (partly) is supported by the Swedish Research Council (VR) via grant VR 2008-5072 *'Finite element preconditioners for algebraic problems as arising in modelling of multiphase microstructures'*. The third author was supported by the Graduate School in Mathematics and Computing (FMB). The support is hereby gratefully acknowledged.

The computations were performed on resources provided by SNIC through Uppsala Multidisciplinary Center for Advanced Computational Science (UPPMAX) under Project p2009040.

We also thank the developers of the HSL MI-20 AMG package for kindly providing us with the AMG-`Matlab` interface.

References

[1] O. Axelsson, On global convergence of iterative methods, *Iterative solution of nonlinear systems of equations* (1982), 1–19, Springer.

[2] O. Axelsson, A. Kucherov, Real valued iterative methods for solving complex symmetric linear systems. *Numerical Linear Algebra with Applications* 7 (2000), 197–218.

[3] O. Axelsson, W. Layton, A two-level method for the discretization of nonlinear boundary value problems, *SIAM J. Numer. Anal.* 30, No.6, 2359–2374, 1996.

[4] O. Axelsson, M. Neytcheva, Operator splittings for solving nonlinear, coupled multiphysics problems with an application to the numerical solution of an interface problem. TR 2011-009, Institute for Information Technology, Uppsala University, April 2011.

[5] O. Axelsson, A. Padiy, On a two-level Newton-type procedure applied for solving nonlinear elasticity problems, *Int. J. Numer. Meth. Engng*, 2000, **49**, 1479–1493.

[6] V. E. Badalassi, H. D. Ceniceros, S. Banerjee, Computation of multiphase systems with phase field models, *Journal of Computational Physics*, 32(2003): 371-397, Elsevier.

[7] W. Bangerth, C. Burstedde, T. Heister, M. Kronbichler, Algorithms and Data Structures for Massively Parallel Generic Finite Element Codes, *ACM Transactions on Mathematical Software*, 38(2), 2011.

[8] R. E. Bank, D. J. Rose, Global Approximate Newton Methods, *Numer. Math.* 37, 1981, 279–295.

[9] P. Boyanova, M. Do-Quang, M. Neytcheva, Solution methods for the Cahn-Hilliard equation discretized by conforming and non-conforming finite elements, TR 2011-004, Institute for Information Technology, Uppsala University, March 2011.

[10] P. Boyanova, M. Do-Quang, M. Neytcheva, Efficient preconditioners for large scale binary Cahn-Hilliard models, Computational Methods in Applied Mathematics , to appear.

[11] J.W. Cahn, On spinodal decomposition, *Acta Metallurgica* 9 (1961) 795–801.

[12] J.W. Cahn, J. Hilliard, Free energy of a nonuniform system. I. Interfacial free energy, *Journal of Chemical Physics* 28 (1958) 258–267.

[13] The deal.II library, `http://www.dealii.org/`.

[14] R. S. Dembo, S. C. Eisenstat, T. Steihaug, Inexact newton methods, *SIAM Journal on Numerical analysis* (1982), 400–408, JSTOR.

[15] M. Do-Quang, G. Amberg, The splash of a ball hitting a liquid surface: Numerical simulation of the influence of wetting, *Physics of Fluids*, 21, 022102 (2009).

[16] C.M. Elliott, D.A. French, F.A. Milner, A second order splitting method for the Cahn-Hilliard Equation, *Numerische Mathematik*, 54 (1989), 575–590.

[17] D. Eyre, An unconditionally stable one-step scheme for gradient systems. Unpublished article, 1998.

[18] D. Eyre, Unconditionally gradient stable time marching the Cahn-Hilliard equation, *Computational and mathematical Models of Microstructural Evolution*, 1998.

[19] X. Feng, A. Prohl, Error analysis of a mixed finite element method for the Cahn-Hilliard equation, *Numerische Mathematik*, 99 (2004), 47–84.

[20] C.G. Gal, M. Grasselli, Asymptotic behavior of a Cahn-Hilliard-Navier-Stokes system in 2D, *Ann. I. H. Poincare AN* 27 (2010),p. 401–436.

[21] M.E. Gurtin, D. Polignone, J. Vials Two-phase binary fluids and immiscible fluids described by an order parameter, *Math. Models Methods Appl. Sci.* 6 (1996): 815-831.

[22] The HSL Mathematical Software Library, `http://www.hsl.rl.ac.uk/`

[23] The Hypre library, `http://acts.nersc.gov/hypre/`.

[24] S. Hysing, S. Turek, D. Kuzmin, N. Parolini, E. Burman, S. Ganesan, L. Tobiska, Quantitative benchmark computations of two-dimensional bubble dynamics, *Int. J. Numer. Meth. Fluids*, 60 (2009), 1259–1288.

[25] A. Novick-Cohen, The Cahn-Hilliard Equation, *Handbook of Differential Equations. IV Evolutionary Partial Differential Equations*, Editors: C. Dafermos and M. Pokorny, Elsevier 2008, 201–228.

[26] The PETSc library, `http://www.mcs.anl.gov/petsc/`.

[27] D. Ralph, Global convergence of damped Newton's method for nonsmooth equations via the path search, *Mathematics of Operations Research* (1994), 352–389, JSTOR.

[28] The Trilinos library, `http://trilinos.sandia.gov/`.

[29] Y. Sun, C. Beckermann, Sharp interface tracking using the phase-field equation, *Journal of Computational Physics archive*, Volume 220 Issue 2, January, 2007.

[30] W. Villanueva, G. Amberg, Some generic capillary-driven flows, *International Journal of Multiphase Flow*, 32 (2006), 1072–1086.

[31] X. Wu, Preconditioners for the Discrete Cah-Hilliard Equation in Three Space Dimensions, Master thesis, Uppsala University, 2011, http://uu.diva-portal.org/smash/record.jsf?pid=diva2:455088