Unconditionally gradient stable time marching the Cahn-Hilliard equation

David J. Eyre*

May 26, 1998

Abstract

Numerical methods for time stepping the Cahn-Hilliard equation are given and discussed. The methods are unconditionally gradient stable, and are uniquely solvable for all time steps. The schemes require the solution of ill-conditioned linear equations, and numerical methods to accurately solve these equations are also discussed.

1 Introduction.

Numerical time marching schemes for the Cahn-Hilliard equation will be discussed in this paper. The Cahn-Hilliard equation is a leading model in theoretical materials science, and the efficient numerical solution of the equation is needed to understand its dynamics. The equation is a fourth order nonlinear parabolic partial differential equation,

$$\frac{\partial u}{\partial t} = \Delta \left[-\varepsilon^2 \Delta u + f(u) \right] \tag{1}$$

where $u = u(\vec{x}, t)$ and

$$0 < \varepsilon \ll 1, \quad \vec{x} \in \Omega, \quad t > 0. \tag{2}$$

The nonlinear term taken here will be $f(u) = u^3 - u$, but the properties of the numerical methods hold equally for other bistable nonlinear terms. The operator Δ is the spatial laplacian. The linear part of $\Delta f(u)$ is $-\Delta u$, and this term is responsible for the interesting dynamics of the Cahn-Hilliard equation including the instability of constant solutions near $u = 0$. The nonlinear term of $\Delta f(u)$ is Δu^3, and this term stabilizes the flow. The Cahn-Hilliard equation was derived in [2] and numerical issues concerning the equation were surveyed in [3].

The natural and average conserving boundary conditions for the system are

$$\nabla u \cdot \vec{n} \mid_{\vec{x} \in \partial\Omega} = \nabla(\Delta u - f(u)) \cdot \vec{n} \mid_{\vec{x} \in \partial\Omega} = 0 \tag{3}$$

where $\vec{n}$ is the outward normal on the boundary of Ω. These boundary conditions leave the spatial average of $u(\vec{x}, t)$ invariant in time. The Cahn-Hilliard equation dissipates energy, i.e.

$$\frac{d\mathcal{E}(u)}{dt} \leq 0 \tag{4}$$

*Department of Mathematics, University of Utah, Salt Lake City, UT 84112, eyre@math.utah.edu

where

$$\mathcal{E}(u) = \int_\Omega \left[\frac{\varepsilon^2}{2} |\nabla u|^2 + F(u) \right] dx \tag{5}$$

and $f(u) = F'(u)$. Therefore, for all positive times, $F(u(t)) \leq F(u(0))$.

The Cahn-Hilliard equation is notoriously difficult to solve numerically [3] because the equations are stiff due to both the biharmonic operator and the nonlinear operator. Additionally, across the spatial interfaces, the solution undergoes an $O(1)$ change over an $O(\varepsilon)$ interval. To accurately resolve these interfaces a fine discretization of space is required.

Therefore, most time marching schemes require one to use time steps that are many orders of magnitude smaller than the fastest times scales of the equation. For example, if Euler's method is employed to solve these equations with the given scaling and with $\varepsilon = 0.01$, then one is forced to take time steps on the order of 10^{-7} while the fastest interesting dynamic time scales are on the order of 10^{-4}. Therefore, thousands of time steps are required to see even the fastest dynamics. Other dynamic time scales are much longer, making explicit numerical schemes expensive.

Other explicit and most semi-implicit schemes have similar restrictions on their time steps. Alternatively, fully implicit schemes require that one solve a coupled system of nonlinear equations. If one chooses the time step for these schemes to be too large, the nonlinear equations have multiple solutions. Only one of these solutions is correct, and the spurious solutions confuse nonlinear equation solvers, like Newton's Method. Consequently, these solvers will not converge to any solution. Finally, it has been shown [7] that only a small set of fully implicit schemes are gradient stable in the sense that they have a discrete energy that decreases from one time level to the next. Furthermore, all of these schemes are conditionally gradient stable meaning that they only have a discrete gradient structure for small enough time steps. The allowed time steps are roughly one order of magnitude larger than the maximum stable time step allowed by Euler's method.

The goal of this paper is to apply a new semi-implicit method [4] to the Cahn-Hilliard equation that resolves the problems associated with both the stiffness and solvability. The methods are unconditionally gradient stable, which means that the equations have a discrete energy that decreases from one time level to the next for all positive time steps. The schemes also possess the same equilibria as the dynamic equations. The methods are also uniquely solvable for all positive time steps.

The methods require the solution of a ill-conditioned set of linear equations, and this paper also gives methods for accurately solving these systems of equations.

It should be emphasized that while the methods will allow one to take arbitrarily large time steps and not become unstable, the accuracy of the numerical solution will depend on choosing a small enough time step to resolve the dynamics.

2 Gradient stable time stepping schemes.

For simplicity, the time stepping method will be presented as a finite difference scheme in two spatial dimensions. One could use the time stepping with other spatial discretizations and in other spatial domains.

Using finite differences, the Cahn-Hilliard equation can be discretized with centered approximations to the spatial derivative operators on $\Omega = [0,1] \times [0,1]$. Let u_{ij}^n be the discrete approximation of $u(ih, jh, nk)$, where the time step is k, the spatial discretization is $h = 1/m$, and $m = O(\varepsilon^{-1})$. Then the standard centered difference approximation of the laplacian is

$$h^2 \Delta_h u_{ij}^n = u_{i+1,j}^n + u_{i-1,j}^n + u_{i,j+1}^n + u_{i,j-1}^n - 4u_{i,j}^n. \tag{6}$$

This operator will be used to approximate the continuous laplacian on the discrete grid.

The key to achieving the desired stability and solvability features of the time marching scheme is to divide the nonlinear terms into a stabilizing term and into a growth term, and then to separate those terms across the time step. The most logical way to accomplish this is with the time marching scheme

$$\frac{u_{ij}^{n+1} - u_{ij}^n}{k} = -\varepsilon^2 \Delta_h^2 u_{ij}^{n+1} + \Delta_h (u_{ij}^{n+1})^3 - \Delta_h u_{ij}^n. \tag{7}$$

To find u_{ij}^{n+1} via this scheme, one needs to solve the nonlinear equations

$$\left(1 + k\varepsilon^2 \Delta_h^2\right) u_{ij}^{n+1} - k\Delta_h (u_{ij}^{n+1})^3 = (1 - k\Delta_h)\, u_{ij}^n \tag{8}$$

at every grid point. This splitting satisfies the conditions presented in [4] for an unconditionally gradient stable time marching scheme.

If the grid points are gathered into a vector by a lexicographic ordering of the nodes and $\mathbf{U}^n \in \mathbf{R}^{m^2}$ is the solution at $t = nk$, then the difference equations read

$$\left(I + k\varepsilon^2 A_h^2 - kA_h D(\mathbf{U}^{n+1})^2\right) \mathbf{U}^{n+1} = (I + kA_h)\, \mathbf{U}^n \tag{9}$$

where $D(\mathbf{U})$ is a diagonal matrix with the vector $\mathbf{U}$ placed element-wise along the main diagonal. The matrix A_h is the ordered discrete laplacian, it has one zero eigenvalue and its remaining eigenvalues are negative, and it has the familiar block tridiagonal form.

The equations in (9) are a large coupled set of nonlinear equations. Numerical experiments in one spatial dimension show that Newton's method converges in two or three iterations for time steps on the order of the fastest time scales of the dynamics. The convergence requires four or five iterates for time steps less than one, and p additional iterates when $k = 10^p$ for $p > 0$.

While this nonlinear splitting is reasonable and has the smallest local truncation error of the one step methods that are gradient stable, the following splitting of the stabilizing and contracting terms leads to a set of *linear* equations that must be solved at every time step. This time marching scheme is

$$\frac{u_{ij}^{n+1} - u_{ij}^n}{k} = -\varepsilon^2 \Delta_h^2 u_{ij}^{n+1} + \Delta_h \left[(u_{ij}^n)^2 u_{ij}^{n+1}\right] - \Delta_h u_{ij}^n. \tag{10}$$

After ordering the nodes, the difference equations in (10) read

$$\left(I + k\varepsilon^2 A_h^2 - kA_h D(\mathbf{U}^n)^2\right) \mathbf{U}^{n+1} = (I - kA_h)\, \mathbf{U}^n. \tag{11}$$

Notice that the only difference between this method and the previous method is that the nonlinear matrix is evaluated at the previous time step. This scheme is also unconditionally gradient stable, and the local truncation error for this method is the same as for the previous method, $O(k)$. Furthermore the equations are uniquely solvable for all time steps. Therefore since the equations are linear, this method is the recommended scheme.

From this scheme, it is easy to see that the solution will remain bounded, and that the scheme is uniquely solvable for all time steps. The solvability property follows because the difference operators on the left hand side of the matrix equation are all nonnegative definite. Therefore, the matrix equation has strictly positive eigenvalues for all positive time steps. The boundedness follows from comparing the operator on the left hand side with the operator on the right hand side of the equation. Approximating the terms in $D(\mathbf{U})^2$ by their smallest value, one can see that the long wavelength perturbations cannot grow unboundedly because the eigenvalues of the operator on the right hand side are smaller than one in absolute value and the eigenvalues of the operator on the left hand side are larger than one. Similarly, short wavelength perturbations cannot grow unboundedly because their growth is strongly damped by the discrete biharmonic operator, A_h^2.

3 Solving the linear equations.

Ideally when working with discretizations of the laplacian, it makes sense to use FFT based methods to invert the operators because the FFT basis functions are eigenvectors of the difference operators. FFT methods require that the operator be linear and constant coefficient. Unfortunately, while the equations in (11) are linear, they are not constant coefficient equations and, therefore, FFT based methods cannot be used for the inversion.

Either direct or iterative methods could then be used to solve the linear equations. However, inverting this system is difficult because the equations are ill-conditioned. The ill-conditioning arises from the discretization of the biharmonic operator $-\Delta^2$ that is represented by the matrix A_h^2. The eigenvectors of this matrix are sinusoidal, and the large eigenvalues of this matrix scale like m^4 where m is the wave number of the eigenvector and the spatial discretization parameter. The small eigenvalues of A_h are zero, so the small eigenvalues of the update matrix are near one. Therefore, the condition number of the update matrix is $O(\varepsilon^2 m^4)$. For large enough m to make the calculations interesting, this condition number makes naive matrix solution methods unusable.

This section will present a method to precondition the linear equations in order to greatly reduce the condition number of the original matrix. Once the preconditioner is described, iterative methods are given to solve the transformed linear system of equations.

3.1 Preconditioning the system.

Nearly all matrix solution methods fail miserably when the system of equations they are trying to solve is poorly conditioned. Roundoff error accumulates to the point that it dominates the computed answer, and the effect is that the "solution" computed by the chosen method is useless. One logical choice to resolve this problem is to transform the original system of linear equations into a new system of linear equations that have better conditioning, and then to solve the problem in the new transformed variables. This approach, called preconditioning, will be applied to solve the update equations defined in (11). The approach given here results in a matrix whose condition number is nearly one, and is therefore, nearly optimal.

Notice that the only term with nonconstant coefficients in the linear equations defined in (11) are the elements of the diagonal matrix $D(\mathbf{U}^n)^2$. This matrix has nonnegative entries, and once the solution of the Cahn-Hilliard equation has a layered structure, the matrix is nearly the identity everywhere in space except within the layers themselves. Therefore the equations in (11) are *nearly* constant coefficient equations. This suggests that a constant coefficient approximation to the update matrix would make a good preconditioner.

Let $\hat{D} = d^2 I$ be the constant approximation of $D(\mathbf{U}^n)^2$, and define the preconditioner to be

$$M = I + k\varepsilon^2 A_h^2 - kd^2 A_h. \tag{12}$$

The results of numerical tests indicate that taking d^2 equal to the mean of the elements of $D(\mathbf{U}^n)^2$ and taking $d^2 = 1$ are both reasonable choices. It will now be shown that this preconditioner is nearly optimal.

Using the preconditioner, the original equations are replaced with a transformed system of linear equations of the form

$$M^{-1}\mathcal{A}\hat{x} = \hat{b} \tag{13}$$

where

$$\mathcal{A} = I + k\varepsilon^2 A_h^2 - kA_h D(\mathbf{U}^n)^2, \tag{14}$$

and

$$\hat{b} = M^{-1}\left(I - kA_h\right)\mathbf{U}^n, \text{ and } \hat{x} = M^{-1}\mathbf{U}^{n+1}. \tag{15}$$

These condition number of equations is now analyzed.

The effects of this preconditioner on the condition number of the problem are easily seen. First, since A_h has one zero eigenvalue corresponding to a constant eigenvector, the preconditioned matrix will have a smallest eigenvalue nearly equal to 1. To find the largest eigenvalue of the preconditioned matrix, the equations $M^{-1}\mathcal{A}$ are calculated as

$$\begin{aligned}
&\left(I + k\varepsilon^2 A_h^2 - kd^2 A_h\right)^{-1}\left(I + k\varepsilon^2 A_h^2 - kA_h D(\mathbf{U}^n)^2\right) \\
&\quad = \left(I + k\varepsilon^2 A_h^2 - kd^2 A_h\right)^{-1}\left(I + k\varepsilon^2 A_h^2 - kA_h D(\mathbf{U}^n)^2 - kd^2 A_h + kd^2 A_h\right) \\
&\quad = I + \left(I + k\varepsilon^2 A_h^2 - kd^2 A_h\right)^{-1} kA_h\left(d^2 I - D(\mathbf{U}^n)^2\right).
\end{aligned}$$

To estimate the two norm of this latter matrix note that

$$\begin{aligned}
&\|I + \left(I + k\varepsilon^2 A_h^2 - kd^2 A_h\right)^{-1} kA_h\left(d^2 I - D(\mathbf{U}^n)^2\right)\| \\
&\quad \leq \; 1 + \|\left(I + k\varepsilon^2 A_h^2 - kd^2 A_h\right)^{-1} kA_h\left(d^2 I - D(\mathbf{U}^n)^2\right)\| \\
&\quad \leq \; 1 + \|\left(I + k\varepsilon^2 A_h^2 - kd^2 A_h\right)^{-1} kA_h\| \, \|d^2 I - D(\mathbf{U}^n)^2\|
\end{aligned}$$

To analyze the norm of the first system, notice that the eigenvectors of the matrix

$$\left(I + k\varepsilon^2 A_h^2 - kd^2 A_h\right)^{-1} kA_h$$

correspond to the eigenvectors of A_h. Therefore, if X is an eigenvector of A_h with eigenvalue $\lambda < 0$, then

$$\left[\left(I + k\varepsilon^2 A_h^2 - kd^2 A_h\right)^{-1} kA_h\right] X = \left(\frac{-k\lambda}{1 + k\varepsilon^2\lambda^2 - kd^2\lambda}\right) X. \tag{16}$$

The eigenvalues are bounded above by

$$\frac{-k\lambda}{1+k\varepsilon^2\lambda^2-kd^2\lambda} < \frac{1}{d^2},$$

and this value bounds the two norm of the first matrix as well since symmetric matrices two norm equals their largest eigenvalue. Suppose that d^2 is chosen to satisfy $(u_{ij}^n)^2 \leq 2d^2$, then combining this with the last inequality gives

$$\| \left(I + k\varepsilon^2 A_h^2 - kd^2 A_h\right)^{-1} kA_h\| \, \|d^2 I - D(\mathbf{U}^n)^2\| < \frac{1}{d^2} \max_{ij} |d^2 - (u_{ij}^n)^2| < 1. \tag{17}$$

Furthermore, the analysis suggests that an optimal choice for d^2 is to satisfy the minimax criteria

$$\min_d \max_{ij} \frac{1}{d^2} |d^2 - (u_{ij}^n)^2| \tag{18}$$

at every time step. Thus, the condition number of the product $M^{-1}\mathcal{A}$ is less than two which is a nearly optimal preconditioning.

3.2 Fast inversion of the preconditioner.

For the preconditioner to be useful, it must be capable of being inverted quickly. The equations of (12) are constant coefficient linear equations, and can be quickly solved using FFT methods. These methods are described in most textbooks on the numerical solution of partial differential equations (for example, see [1]). Briefly, FFT based methods exploit the fact that the basis functions of the FFT are eigenfunctions of the discrete laplacians. More precisely, the cosine basis functions that are used in a discrete cosine transformation are eigenfunctions of the Cahn-Hilliard equations with the boundary conditions given in (3). The eigenvalues corresponding to these eigenfunctions for the finite difference method presented here are

$$\lambda_{kl} = -4m^2 \left(\sin^2 \frac{k\pi}{2m} + \sin^2 \frac{l\pi}{2m}\right) \tag{19}$$

The eigenvectors of the preconditioner matrix (12) are also discrete cosine functions, and the eigenvalues corresponding to these eigenvectors are the sum of the eigenvalues of the individual matrices, or

$$(I + k\varepsilon^2 A_h^2 - kd^2 A_h)x_{kl} = (1 + k\varepsilon^2 \lambda_{kl}^2 - kd^2 \lambda_{kl})x_{kl} \tag{20}$$

Therefore, to solve a system of the form $Mx = b$ where M is the preconditioner requires three steps;

- Compute the fast cosine transformation $\hat{b}$ of b.
- Divide the elements of $\hat{b}$ by the eigenvalues of M.
- Compute the inverse fast cosine transformation of the result.

This fast solution technique will be used as the basis for the iterative solution of the original system.

3.3 Iterative solutions of the preconditioned system.

Two logical choices for an iterative method to solve the preconditioned are now presented. The first method is an elementary iterative method that is presented to demonstrate the iterative process. For this method, recall the preconditioner matrix M, define N as follows

$$N = kA_h \left(d^2 I - D(\mathbf{U}^n)^2 \right),$$

and define a sequence of approximations $\mathbf{U}_\ell^{n+1}$ to $\mathbf{U}^{n+1}$ as

$$M\mathbf{U}_{\ell+1}^{n+1} = (I - kA_h)\,\mathbf{U}^n - N\mathbf{U}_\ell^{n+1}$$

where ℓ is the iteration parameter. From the definitions, notice that

$$M + N = I + k\varepsilon^2 A_h^2 - kA_h D(\mathbf{U}^n)^2,$$

so if $\mathbf{U}_{\ell+1}^{n+1} = \mathbf{U}_\ell^{n+1}$ then $\mathbf{U}_{\ell+1}^{n+1} = \mathbf{U}^{n+1}$ because the linear equations have a unique solution.

The iterative scheme is to guaranteed to converge if the spectral radius of the iteration matrix $M^{-1}N$ is less that one [5]. The spectral radius of $M^{-1}N$ was bounded below one in equation (17) provided a reasonable choice of d^2 is made. Thus the iteration procedure will converge.

The first method demonstrates a simple iterative solver that is guaranteed to converge with reasonable choices of the parameters that are available. However, far more sophisticated iterative schemes exist that will converge faster than this scheme. The most logical choice to solve the preconditioned system presented in this paper is the method known as conjugate gradients squared (CGS). A formal presentation of this method is beyond the scope of this paper. Its details are presented in [6], and well documented and tested code for CGS is available from *www.netlib.org*.

The convergence rate for CGS applied to this system when d^2 is chosen via the minimax criteria (18) is available from analytic results. These results indicate that the solution will gain one decimal digit of accuracy for every 2.1 iterations. Therefore, 16 to 20 iterations is enough to compute the solution to single precision accuracy, and computing many more iterates is wasteful because the accuracy of the scheme, $O(k)$, doesn't significantly more work on the linear system. Since each CGS iteration requires only $O(m^2)$ effort, the overall effort required to solve the linear system is dominated by the effort to perform the FFTs and is $m^2 \log m$. The system itself has m^2 unknowns, therefore, the method is nearly optimal. Furthermore, since the time marching method is unconditionally stable, one can take large steps and effectively solve the Cahn-Hilliard equation over long time intervals.

References

[1] W. F. Ames, *Numerical Methods for Partial Differential Equations*, Academic Press, New York (1977), pp. 304–307.

[2] J. W. Cahn, Acta Met., 9 (1961), pp. 795–801.

[3] C. M. Elliott, in Mathematical Models for Phase Change Problems, J. F. Rodrigues, ed., Birkhäuser Verlag, Basel, 1989, pp. 35–73.

[4] D. J. EYRE, preprint.

[5] G. GOLUB AND C. F. VAN LOAN, Matrix Computations, Johns Hopkins Press, Baltimore (1983), pp. 353–361.

[6] P. SONNEVELD, SIAM J. Sci. Stat. Comput., 10 (1989), pp. 36-52.

[7] A. M. STUART AND A. R. HUMPHRIES, SIAM Rev., 36 (1994), pp. 226-257.