ADAPTIVE MESH REFINEMENT FOR MULTISCALE NONEQUILIBRIUM PHYSICS

In this article, the authors demonstrate how to use adaptive mesh refinement (AMR) methods for the study of phase transition kinetics. In particular, they apply a block-structured AMR approach to investigate phase ordering in the time-dependent Ginzburg-Landau equations.

A long-standing challenge in both applied and theoretical physics is to fully understand and predict the behavior of systems far from thermodynamic equilibrium,[1–4] including those systems driven by an external force or experiencing a sudden change in environment (such as pressure or temperature). They also include systems transitioning from one metastable or long-lived state to another. The need to accurately model and numerically simulate these processes has become more pressing as efforts to derive well-parameterized evolution equations for mesoscopic nonequilibrium dynamical processes' progress. Simultaneously, it's becoming increasingly important (for example, in phase transition kinetics[5,6]) to be able to validate equations and their solutions against experiment. This validation requires highly accurate and efficient partial differential equation (PDE) solvers.

PDE integrators for these problems aren't typically selected for their space and time accuracy. Instead, researchers have used easily implemented, mostly explicit methods to determine universal features such as domain growth exponents. For modern materials applications, however, space and time resolution are crucial for modeling system behavior. For these applications, the standard low-order or fixed-grid integration approach is prohibitive due to time-step constraints. We address this issue by applying adaptive mesh refinement (AMR) methods in dynamical condensed-matter systems. For ease of exposition, we focus on the time-dependent (real) Ginzburg-Landau (TDGL) equations because they're the prototypical example of a spatially extended nonequilibrium system undergoing a dynamical phase transition.

1521-9615/05/$20.00 © 2005 IEEE
Copublished by the IEEE CS and the AIP

DANIEL F. MARTIN AND PHILLIP COLELLA
Lawrence Berkeley National Laboratory
MARIAN ANGHEL AND FRANCIS J. ALEXANDER
Los Alamos National Laboratory

Nonequilibrium Phenomena

The equations of dynamical critical phenomena, phase separation, and so on typically fall into the reaction-diffusion or reaction-diffusion-advection type.[4,7] They can contain several scalar, vector, or tensor fields, and they're often dissipative, but sometimes they have an inertial manifold.[8–10] In this article, we consider nonequilibrium phenomena of the following type:

- The system (for example, a binary alloy) is at high temperature, in the disordered or single phase.

- The temperature or other control parameter is rapidly changed to a value at which the system is unstable to fluctuations.
- Via nucleation or continuous ordering, coherent structures of the new stable phase grow.
- As time progresses, the ordered domains R grow, typically as some power of time $R(t) \sim t^{\alpha}$.

Because most of the alloy concentration's variation is at the interface between domains, it's worthwhile to use numerical and analytical methods that take advantage of this fact. Relatively recent efforts work in the regime of a very sharp interface.[11–13]

One such approach is AMR, in which the computational mesh is locally refined in regions where greater accuracy is desired. Local refinement of the computational mesh can permit solutions with the accuracy of the finest mesh with only a fraction of the cost of a computation on the equivalent uniformly fine mesh. We use Marsha Berger and Phillip Colella's block-structured approach,[14] in which refinement is organized in logically rectangular regions of the domain. Discontinuities in the computational mesh due to local refinement require special stencils to maintain accuracy. Block-structured refinement allows this overhead at interfaces between coarse and fine meshes to be amortized over many regular operations on the interiors of the refined regions. This approach was initially implemented for gas dynamics,[14] but has been successfully extended to other systems of equations including incompressible fluid dynamics,[15,16] plasma fluid dynamics,[17] and magnetohydrodynamics (MHD).[18]

The Time-Dependent Ginzburg-Landau Equations

Our physical system is characterized by a coarse-grained local order parameter $\phi(x, t)$, which represents a concentration difference between competing phases.

The free energy for this system is given by

$$F[\phi] = \int dr \left(\frac{(\nabla\phi)^2}{2} + V(\phi) \right), \qquad (1)$$

where $V(\phi)$ is a local potential. For simplicity, we choose $V(\phi) = (1/4)\phi^4 - (1/2)\phi^2$, which is a double-well potential.

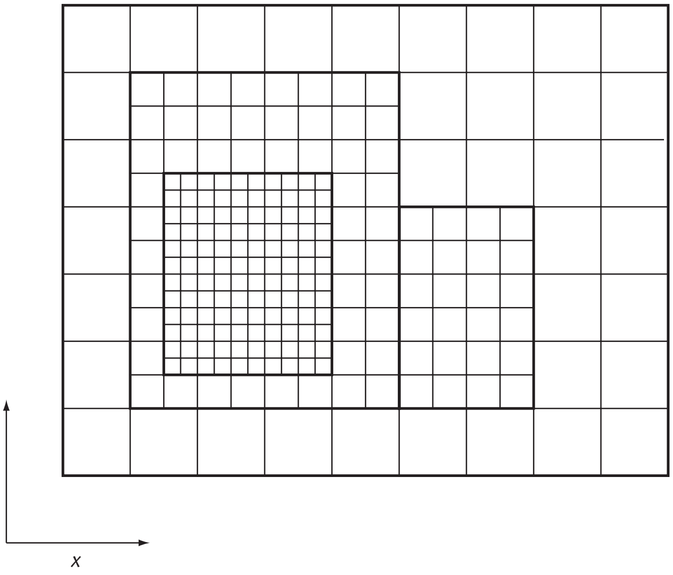

Figure 1. Block-structured local refinement. Two levels of refinement (levels 1 and 2) are nested within the coarse mesh (level 0).

The standard approach to generating a dynamics when the order parameter isn't conserved is to equate the local order parameter's time derivative with the local chemical potential, which in this case is the (negative) of the free energy's variational derivative with respect to the order parameter. Hence,

$$\frac{\partial\phi}{\partial t} = -\frac{\delta F[\phi]}{\delta\phi} = \Delta\phi - V'(\phi)\,. \qquad (2)$$

For simplicity, we've set any kinetic coefficients to unity. Our specific potential V then yields the following dynamics for the order parameter:

$$\frac{\partial\phi}{\partial t} = \Delta\phi - \phi^3 + \phi\,. \qquad (3)$$

These equations model the time development of the concentration difference between competing stable phases. Other evolution equations for nonequilibrium dynamics have similar behavior and will be dealt with elsewhere.

AMR Operators and Notation

As Figure 1 illustrates, we start with a single, uniform Cartesian mesh, with grid spacing h_0 and indexing $\mathbf{i} = (0, N_0 - 1)$. Where refinement is desired, we refine logically rectangular patches by a refinement ratio n^0_{ref}, with cell spacing $h_1 = h_0/n^0_{ref}$. The collection of logically rectangular patches with $h = h_1$ are considered to make up a single AMR level, level 1 (the original base mesh is level 0). If further refinement is desired, we can refine logically rectangular patches of level 1 by n^1_{ref} to create a second level of refinement (level 2).

In general, we can create an arbitrary number of refinement levels. The refinement level ℓ is related to the coarser level $(\ell - 1)$ by the refinement ratio $n_{ref}^{\ell-1} = h_{\ell-1}/h_{\ell}$. Because the solution on finer grids is assumed to be more accurate, only cells that aren't covered by refinement are considered to have "valid" solution values; the collection of cells on a level uncovered by refinement are considered to be the "valid region" on a given AMR level. In our implementation, we've used refinement ratios that are powers of two. To simplify the stencils at the interfaces between fine and coarse regions, we require that refined regions be properly nested[14]—the edge of level ℓ must either be adjacent to level $(\ell - 1)$ cells or be located at the boundary of the computational domain. Implementation is also greatly simplified by the use of a global index space on each level; for the cell-centered discretization used in this work, the cell $\mathbf{i} = (i, j)$ on level ℓ is centered at $\mathbf{x_i} = (x_i, y_j) = ((i + 1/2)h_\ell, (j + 1/2)h_\ell)$.

Subcycling is particularly useful when temporal accuracy is required on all levels in the adaptive mesh hierarchy or when a problem has a very large range of dynamic scales.

Implementing numerical algorithms on locally refined meshes requires care in the construction of operator discretizations at the coarse–fine interface to ensure that the appropriate accuracy is maintained for the resulting solution. We refer to such discretizations on the composite mesh hierarchy as *composite* or *AMR operators*. We'll use a composite discretization of the Laplacian operator,[16] which we'll refer to as $L^{comp}\phi$.

Although other AMR time-accurate applications employ refinement in time and space,[14–18] we've elected to advance cells at all levels of refinement with the same time step (also referred to as a *nonsubcycled algorithm*). This approach has been used in various contexts, including incompressible flow and porous media flow.[19,20] The hyperbolic nature of the equations solved elsewhere[14–16] make refinement in time particularly useful, because it enables all levels to advance at similar CFL (Courant-Friedrichs-Lewy) numbers.

Because the advection schemes employed in these other works have better phase-error accuracy at moderate CFL numbers, subcycling is important to the numerical accuracy on coarser levels for these algorithms.[1] Subcycling is particularly useful when similar temporal accuracy is required on all levels in the adaptive mesh hierarchy or when a problem has a very large range of dynamic scales (resulting in many levels of refinement). The TDGL equations solved in this article produce solutions that quickly evolve into a fairly sharp interface between regions in which the solution is relatively constant spatially. In this case, the solution's accuracy on the coarse levels isn't important because all the dynamics occur at the interface. Therefore, implementing subcycling won't affect solution accuracy in any appreciable way, assuming that the interface between the two phases is completely refined. Subcycled algorithms are more complex, both algorithmically and in code implementation. In the case of the TDGL equations, the subcycled algorithm for a given level (ℓ) would look like this:

1. If $(\ell > 0)$, interpolate boundary conditions in time and space from level $(\ell - 1)$.
2. Advance the solution on level ℓ using a single-level advance from time (t^ℓ) to time $(t^\ell + \Delta t^\ell)$, where $\Delta t^\ell = (1/n_{ref}^{\ell-1})\Delta t^{\ell-1}$.
3. If a level $(\ell + 1)$ exists, advance the solution on the next finer level n_{ref}^{ℓ} times (which will also include advancing any levels finer than $(\ell + 1)$). At this point, both the level ℓ and level $(\ell + 1)$ solutions will have reached the same time.
4. Perform a multilevel elliptic synchronization solve over all levels that have reached time $(t^\ell + \Delta t^\ell)$.
5. Average level $(\ell + 1)$ solution to covered regions of level ℓ.

Note that in a subcycled algorithm, the TDGL equations' elliptic/parabolic nature implies that the synchronization step entails an extra multilevel elliptic solve to enforce the proper matching conditions between coarse and fine levels.[15] In contrast, the nonsubcycled algorithm is the same as the single-level algorithm, except that it uses composite AMR operators.

Because there is no real accuracy benefit to implementing subcycling for the solutions computed in this work, we decided to implement the simpler choice. Adopting a nonsubcycled algorithm results in a much simpler algorithm with fewer elliptic solves; we believe that the benefits of refinement in time are outweighed by the increased algorithmic complexity and computational cost the additional elliptic solves would require in this case.

Integration

We wish to compute solutions to the TDGL equations that are second-order accurate in time and space. Due to the stability constraints of explicit methods for parabolic equations, we elect to use a semi-implicit approach instead. Although the Crank-Nicolson scheme is commonly used for these types of problems, it suffers from accuracy and stability problems in the presence of local refinement. Instead, we use the second-order L_0-stable scheme detailed elsewhere.[21] Although this scheme requires two elliptic solves for each update instead of the one required for Crank-Nicolson, its improved performance in the presence of local refinement makes it a good choice for AMR algorithms.

Mathematically, we can treat the TDGL equations as a heat equation with a nonlinear forcing term:

$$\frac{\partial \phi}{\partial t} = \Delta\phi - \frac{dV(\phi)}{d\phi} , \tag{4}$$

where $V(\phi)$ is the potential.

We discretize ϕ in time and space in a cell-centered manner:

$$\phi^n_{\mathbf{i}} = \phi(\mathbf{x}_{\mathbf{i}}, t^n). \tag{5}$$

The basic advance step from time t^n to time $t^{n+1} = t^n + \Delta t^n$ proceeds as follows:

1. Evaluate source term $S^n = -(V')^n = -dV/d\phi(\phi^n)$.
2. Predict $\tilde{\phi}^{n+1}$ using a forward-difference approximation:

$$\tilde{\phi}^{n+1} \approx \phi^n + \Delta t^n(\Delta\phi - (V')^n). \tag{6}$$

3. Use $\tilde{\phi}^{(n+1)}$ to predict source term $S^{n+1} = -(V')^{n+1}$ for time t^{n+1}.
4. Do initial elliptic solve for parabolic advance and solve for intermediate term ϕ_e:

$$(I - r_2 \Delta t^\ell L^{comp})\phi_e = e \tag{7}$$

$$e = (1 + (1 - a)\Delta t L^{comp})\phi^n + 1/2\Delta t\, [S^n + (1 - 2(a - 1/2)\Delta t L^{comp})S^{n+1}]. \tag{8}$$

5. Do second elliptic solve for parabolic advance and solve for ϕ^{n+1}:

$$(I - r_1 \Delta t L^{comp})\phi^{n+1} = \phi_e. \tag{9}$$

The quantities a, r_1, and r_2 are the values suggested elsewhere:[21]

$$a = 2 - \sqrt{2} - \varepsilon,$$

$$discr = \sqrt{a^2 - 4a + 2} ,$$

$$r_1 = \frac{2a - 1}{a + discr} ,$$

$$r_2 = \frac{2a - 1}{a - discr} ,$$

where ε is a small quantity (we use 10^{-8}).

Solver

We constructed our implementation of this algorithm with the Chombo infrastructure,[22] which simplifies the implementation of the locally adaptive algorithm. To perform the multilevel AMR elliptic solves, we use the Chombo AMR elliptic solver, which is based on a multigrid algorithm.

Results

For this AMR approach to be worthwhile, it must satisfy three things:

1. The AMR algorithm should converge at second-order rates; adding coarse–fine interfaces to the discretization can cause a loss of accuracy, which can impact convergence.
2. Suitably placed local refinement should result in the same solution accuracy as the solution on an equivalent uniformly fine computation mesh.
3. The use of local refinement should result in computational savings over the uniform mesh solution with the same resolution.

Let's examine each of these in more detail.

Convergence of the AMR Algorithm

To demonstrate this algorithm's convergence properties, we solve the sample three-dimensional TDGL system (Equation 3) on a 64.0 × 64.0 × 64.0 domain using an initial condition in which ϕ is initialized to be a sum of sinusoids to create an analytic initial condition that is also complex geometrically.

To evaluate convergence consistently, we specify a refined patch in the center of the domain and refine the entire grid hierarchy. Because the grid hierarchy in this case isn't truly adaptive, we don't expect to get fine-grid accuracy in these computations (although demonstrating the algorithm's overall convergence properties is useful). The so-

Table 1. Convergence results for L_1 arbitrary refine grids, L_1 norm.

Base grid size	Single level		$n_{ref} = 2$		$n_{ref} = 4$	
	Error	Rate	Error	Rate	Error	Rate
32	1.0053e-01	—	9.0256e-02	—	8.7002e-02 —	
64	1.6024e-02	2.65	1.6800e-02	2.42	1.5859e-02	2.46
128	3.9268e-03	2.02	3.8158e-03	2.13	—	—
256	9.9205e-04	1.98	—	—	—	—

Table 2. Convergence results for arbitrary refined grids, L_2 norm.

Base grid size	Single level		$n_{ref} = 2$	$n_{ref} = 4$		
	Error	Rate	Error	Rate	Error	Rate
32	3.0811e-04	—	3.1077e-04	—	3.0196e-04	—
64	5.4665e-05	2.49	5.7526e-05	2.43	5.4348e-05	2.47
128	1.3575e-05	2.01	1.3121e-05	2.12	—	—
256	3.4496e-06	1.98	—	—	—	—

Table 3. Convergence results for arbitrary refined grids, L_∞ norm.

Base grid size	Single level		$n_{ref} = 2$	$n_{ref} = 4$		
	Error	Rate	Error	Rate	Error	Rate
32	1.0257e+00	—	1.0840e+00	—	1.0617e+00	—
64	3.0519e-01	1.75	3.3780e-01	1.68	3.2419e-01	1.71
128	7.8087e-02	1.97	7.8628e-02	2.10	— —	
256	1.9702e-02	1.99	—	—	—	—

lution features cross coarse–fine interfaces in this test, so this test highlights any discretization problems at the coarse–fine interfaces. Because no analytic solution exists for this problem, we compute a solution on a uniform 512^3 fine mesh and treat it as the "exact" solution against which we compare other computed solutions. We compute error norms by averaging the "exact" solution down to the appropriate resolution, subtracting the computed solution, multiplying by the cell volume on each level, and then summing over the valid regions on each level:

$$L_1^{AMR}\phi = \frac{1}{V}\left(\sum_{\ell=0}^{\ell_{\max}} h_\ell^3 \mid Avg(\phi^{exact}) - \phi_\ell \mid \text{on } \Omega_{valid}^\ell\right)$$

$$L_2^{AMR}\phi = \frac{1}{V}\left(\sum_{\ell=0}^{\ell_{\max}} h_\ell^3 \mid Avg(\phi^{exact}) - \phi_\ell \mid^2 \text{ on } \Omega_{valid}^\ell\right)^{\frac{1}{2}}. \quad (10)$$

Because the domain is a 64 × 64 × 64 cube, the L_1 and L_2 error norms are normalized by dividing by the total domain volume, V = 262,144.

Tables 1, 2, and 3 show the test problem's convergence at L_1, L_2, and L_∞; note that because the refinement is placed without regard to the solution, there is no apparent accuracy gain due to refinement. Because the time step and cell spacing are reduced simultaneously (the time step is halved when the cell spacing is halved), second-order convergence in these plots demonstrates convergence in both time and space. As expected, adding refinement in this example doesn't noticeably improve the solution's accuracy, because the refinement isn't placed in an adaptive manner.

AMR Accuracy

For the AMR approach to be worthwhile, a solution with well-placed local refinement should provide accuracy essentially equivalent to a solution computed with a uniform fine mesh. To demonstrate this, we use a different test problem with a more AMR-appropriate initial condition.

Table 4. Errors for Gaussian AMR test (L_1).

Base grid size	Uniform grid error	n_{ref} = 2 error	n_{ref} = 4 error
32	7.0774e-04	2.2495e-04	6.3469e-05
64	2.2375e-04	6.1733e-05	1.5261e-05
128	5.3856e-05	1.3769e-05	—
256	1.3174e-05	—	—

Table 5. Errors for Gaussian AMR test (L_2).

Base grid size	Uniform grid error	n_{ref} = 2 error	n_{ref} = 4 error
32	6.2447e-06	1.9765e-06	5.4413e-07
64	1.9765e-06	5.4237e-07	1.2004e-07
128	5.4230e-07	1.1713e-07	—
256	1.1626e-07	—	—

Table 6. Errors for Gaussian AMR test (L_∞).

Base grid size	Uniform grid error	n_{ref} = 2 error	n_{ref} = 4 error
32	2.2907e-02	7.4154e-03	2.0236e-03
64	7.4154e-03	2.0236e-03	9.7390e-04
128	2.0236e-03	4.3250e-04	—
256	4.3246e-04	—	—

The initial condition for ϕ is the sum of two Gaussians in a $64.0 \times 64.0 \times 64.0$ cubic domain:

$$\phi(0, \mathbf{x}) = \sum_{i=0}^{n} \sigma_i e^{\frac{|\mathbf{x}-\mathbf{x}_i|^2}{\rho_i}}, \qquad (11)$$

where

n = 1
$\mathbf{x}_0$ = (30, 24, 24)
σ_0 = 1
ρ_0 = 25
$\mathbf{x}_1$ = (36, 40, 40)
σ_1 = –1
ρ_1 = 25.

The solution is evolved to time t = 1, using the undivided gradient of ϕ as the indicator for refinement. Tables 4, 5, and 6 show the error norms computed by using Equation 10; base grid size is the number of cells per side on the coarsest AMR level. If the refinement is effective, then the 32^3 base grid solution with n_{ref} = 2 should have the same error as the 64^3 uniform grid solution, and the 32^3 with n_{ref} = 4 solution should have the same accuracy as the 128^3 uniform grid solution. When the refinement is appropriately placed, Tables 4, 5, and 6 demonstrate that the solutions do attain the equivalent uniform grid solution's level of accuracy.

Performance

To demonstrate the AMR approach's effectiveness, we use a simple test problem on a $64 \times 64 \times 64$ domain with homogeneous Dirichlet boundary conditions.

n = 1
$\mathbf{x}_0$ = (30, 24, 24)
σ_0 = 1
ρ_0 = 50
$\mathbf{x}_1$ = (36, 40, 40)
σ_1 = –1
ρ_1 = 50.

We compute this with a base 32^3 mesh (h_0 = 2.0), with two levels of refinement at n_{ref} = 4. As before, we tag cells for refinement based on the undivided gradient of ϕ. To estimate the cost of the equivalent uniform-grid solution, we computed 15 time steps on the equivalent $512 \times 512 \times 512$ uniform grid. Figure 2 shows the solution at the initial time and at time t = 10.0. Figure 3 shows the finest-level grid boxes at the final time (by this point, the first level of refinement has grown to cover the entire domain).

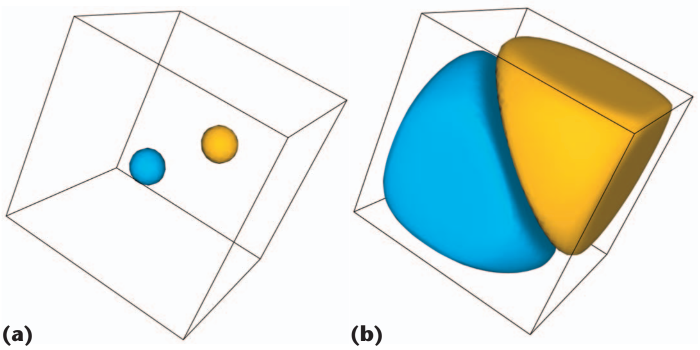

Figure 2. Gaussian test problem. The solutions for isosurfaces of ϕ = (–0.7, 0.7) at (a) the initial condition and (b) time t = 10.

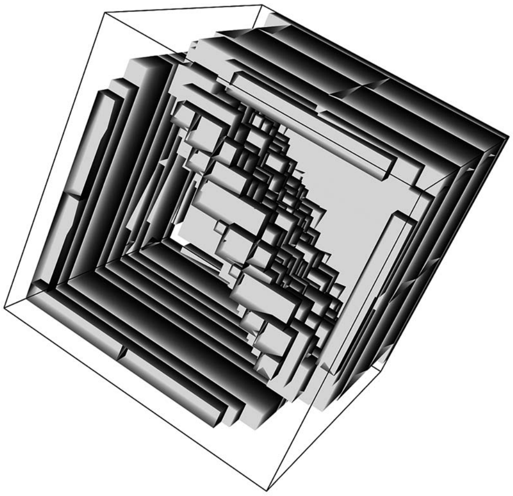

Figure 3. Grid boxes on finest level at solution time t = 10. Grid boxes make up the finest level of refinement at solution time t = 10 for the two-Gaussian problem.

Figure 4 shows the total number of cells advanced per time step for the AMR computation compared to the uniform grid computation. This number slowly increases to approximately 40 percent of the number in the uniform grid case (at the initial time, it's less than 1 percent of the number of cells in the uniform grid case). The smaller problem size can be crucial in fitting the problem onto machines with less memory. The total time spent advancing the solution (not counting I/O and initialization) was 87,942.67 seconds on a 1.8-GHz Opteron machine. As a comparison, running the equivalent single-level computation out to time t = 10 would have taken 291,360 seconds. The AMR computation took 30.2 percent of the execution time and required only 40 percent of the memory the equivalent single-level computation needed. We should note, though, that this is still a fairly small sample problem. We expect that a larger problem with finer resolution will result in more pronounced benefits due to AMR.

The work described here only scratches the surface of potential AMR applications to nonequilibrium problems. A first step beyond would be to treat systems with multiple-order parameter fields, including, for example, the dynamics of liquid crystals and superfluids.

Another direction in which AMR could be applied is modeling systems with a conserved order parameter. The dynamical equations then contain higher order differential operators such as the Cahn-Hilliard equation, in which the Laplacian is replaced by the fourth-order (–ΔΔ) operator. Classical finite difference spatial discretizations aren't well-suited for this problem. Instead, we would consider using a different approach to multiresolution modeling for constant coefficient PDEs.[23]

For the problem examined in this work, the undivided gradient of the solution provided a reasonable indicator for locating refinement. For more complicated problems, this simple approach could be replaced by more appropriate indicators. One approach[14] is to estimate the local truncation error and refine it where it is above a given threshold. Another approach is to determine a physical parameter of the solution as an indicator for refinement (for example, vorticity in a fluid or heat release in a flame).

Acknowledgments

This work, LA-UR-05-0749, was carried out under the auspices of the US Department of Energy at Los Alamos National Laboratory and Lawrence Berkeley National Laboratory. Additional funding came from lab-directed research and development. It was supported at LBNL by the US Department of Energy, Director, Office of Science, Office of Advanced Scientific Computing, Mathematical Information, and Computing Sciences Division under contract DE-AC03-76SF00098.

References

1. R. Zwanzig, *Nonequilibrium Statistical Mechanics*, Oxford Univ. Press, 2001.
2. G. Mazenko, *Fluctuations, Order, and Defects*, Wiley-Interscience, 2003.
3. D.N. Zubarev, V. Morozov, and G. Roepke, *Statistical Mechanics of Nonequilibrium Processes: Volumes I and II*, Akademie Verlag, 1997.

4. S. Dattagupta and S. Puri, *Dissipative Phenomena in Condensed Matter: Some Applications*, Springer-Verlag, 2004.
5. A.J. Bray, "Theory of Phase Ordering Kinetics," *Advances in Physics*, vol. 43, no. 3, 1994, pp. 357–459.
6. J.D. Gunton, M. San Miguel, and P.S. Sahni, *The Dynamics of First Order Phase Transitions*, Academic Press, 1983, p. 267.
7. C. Hohenberg and B.I. Halperin, "Theory of Dynamic Critical Phenomena," *Rev. Modern Physics*, vol. 49, no. 3, 1977, pp. 435–479.
8. P. Constantin et al., *Integral Manifolds and Inertial Manifolds for Dissipative Partial Differential Equations*, Springer-Verlag, 1989.
9. J.C. Robinson, *Infinite-Dimensional Dynamical Systems: An Introduction to Dissipative Parabolic PDEs and the Theory of Global Attractors*, Cambridge Univ. Press, 2001.
10. R. Temam, *Infinite-Dimensional Dynamical Systems in Mechanics and Physics*, Springer-Verlag, 1997.
11. R.F. Almgren and A.S. Almgren, "Phase Field Instabilities and Adaptive Mesh Refinement," *Proc. Materials Week '96: Mathematics of Microstructure Evolution*, Mineral Metals and Materials Soc., 1996, pp. 205–214.
12. N. Provatas, N. Goldenfeld, and J.A. Dantzig, "Adaptive Mesh Refinement Computation of Solidification Microstructures Using Dynamic Data Structures," *J. Computational Physics*, vol. 148, no. 1, 1999, pp. 265–290.
13. N. Goldenfeld, J.H. Jeong, and J.A. Dantzig, "Phase Field Model for Three-Dimensional Dendritic Growth with Fluid Flow," *Physical Rev. E*, vol. 64, no. 4, 2004, pp. 1–14.
14. M.J. Berger and P. Colella, "Local Adaptive Mesh Refinement for Shock Hydrodynamics," *J. Computational Physics*, vol. 82, no. 1, 1989, pp. 64–84.
15. A.S. Almgren et al., "A Conservative Adaptive Projection Method for the Variable Density Incompressible Navier-Stokes Equations," *J. Computational Physics*, vol. 142, no. 1, 1998, pp. 1–46.
16. D. Martin and P. Colella, "A Cell-Centered Adaptive Projection Method for the Incompressible Euler Equations," *J. Computational Physics*, vol. 163, no. 2, 2000, pp. 271–312.
17. P. Colella, M. Dorr, and D. Wake, "Numerical Solution of Plasma-Fluid Equations Using Locally Refined Grids," *J. Computational Physics*, vol. 152, no. 2, 1999, pp. 550–583.
18. R. Samtaney et al., "3D Adaptive Mesh Refinement Simulations of Pellet Injection in Tokamaks," *Computers Physics Comm.*, vol. 164, nos. 1–3, 2004, pp. 220–228.
19. R.D. Hornung and J.A. Trangenstein, "Adaptive Mesh Refinement and Multilevel Iteration for Flow in Porous Media," *J. Computational Physics*, vol. 136, no. 2, 1997, pp. 522–545.
20. M.L. Minion, "A Projection Method for Locally Refined Grids," *J. Computational Physics*, vol. 127, no. 1, 1996, pp. 158–178.
21. E.H. Twizell, A.B. Gumel, and M.A. Arigu, "Second-Order, L_0-Stable Methods for the Heat Equation with Time-Dependent Boundary Conditions," *Advances in Computational Mathematics*, vol. 6, nos. 3 and 4, 1996, pp. 333–352.
22. Applied Numerical Algorithms Group, *The Chombo Framework for Block-Structured Adaptive Mesh Refinement*, tech. report, Lawrence Berkeley Nat'l Lab., 2005; http://seesar.lbl.gov/ANAG/chombo.
23. G.T. Balls and P. Colella, "A Finite Difference Domain Decomposition Method Using Local Corrections for the Solution of Poisson's Equation," *J. Computational Physics*, vol. 180, no. 1, 2002, pp. 25–53.

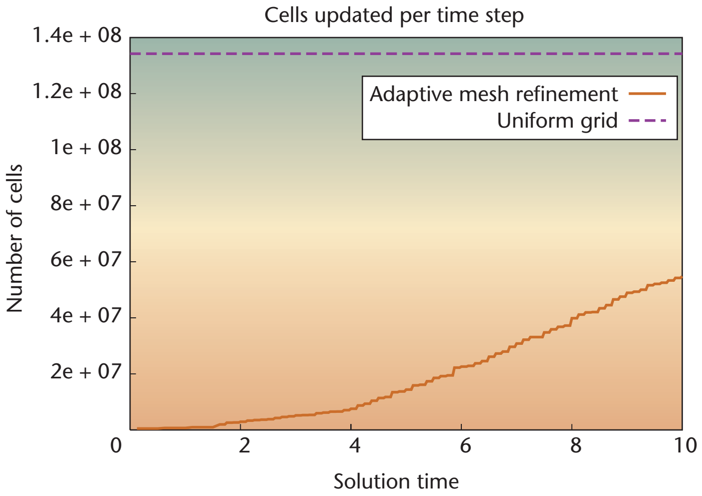

Figure 4. Cells updated per time step. "Solution time" is the time t to which the solution $\phi(\boldsymbol{x}, t)$ has been evolved. "Uniform grid" is the number of cells in the equivalent uniform-grid computation.

Daniel F. Martin is a scientist in the Applied Numerical Algorithms Group at the Lawrence Berkeley National Laboratory (LBNL). His technical interests include applying adaptive mesh refinement techniques to the solution of differential equations and implementing useful scientific software. Martin has a PhD in mechanical engineering from the University of California, Berkeley. Contact him at dfmartin@lbl.gov.

Phillip Colella is a senior staff scientist and group leader of the Applied Numerical Algorithms Group at LBNL. His technical interests include numerical methods for partial differential equations, high-resolution finite difference methods, adaptive mesh refinement, volume-of-fluid methods for irregular boundaries, and programming language and library design for parallel scientific computing. Colella has a PhD in applied mathematics from the University of California, Berkeley. He is a member of the US National Academy of Sciences. Contact him at pcolella@lbl.gov.

Marian Anghel is a technical staff member in the Computer and Computational Sciences Division at Los Alamos National Laboratory. His technical interests include the application of statistical learning theory and numerical acceleration algorithms to the data-driven modeling and identification of nonlinear, multiscale systems. He has a PhD in physics from the University of Colorado, Boulder. Contact him at manghel@lanl.gov.

Francis J. Alexander is a deputy group leader at Los Alamos National Laboratory. His technical interests include statistical mechanics, computational physics, and estimation theory. Alexander has a PhD in physics from Rutgers University. Contact him at fja@lanl.gov.