Journal of Computational Physics 229 (2010) 7942–7955

Contents lists available at ScienceDirect

Journal of Computational Physics

journal homepage: www.elsevier.com/locate/jcp

A Schur complement formulation for solving free-boundary, Stefan problems of phase change

Lisa Lun, Andrew Yeckel, Jeffrey J. Derby *

Department of Chemical Engineering and Materials Science and Minnesota Supercomputing Institute, University of Minnesota, 421 Washington Avenue SE, Minneapolis, MN 55455-0132, USA

ARTICLE INFO

Article history:
Received 20 November 2009
Received in revised form 18 June 2010
Accepted 30 June 2010
Available online 6 July 2010

Keywords:
Stefan problem
Solidification
Crystal growth
Free-boundary
Schur complement
GMRES

ABSTRACT

A Schur complement formulation that utilizes a linear iterative solver is derived to solve a free-boundary, Stefan problem describing steady-state phase change via the Isotherm–Newton approach, which employs Newton's method to simultaneously and efficiently solve for both interface and field equations. This formulation is tested alongside more traditional solution strategies that employ direct or iterative linear solvers on the entire Jacobian matrix for a two-dimensional sample problem that discretizes the field equations using a Galerkin finite-element method and employs a deforming-grid approach to represent the melt–solid interface. All methods demonstrate quadratic convergence for sufficiently accurate Newton solves, but the two approaches utilizing linear iterative solvers show better scaling of computational effort with problem size. Of these two approaches, the Schur formulation proves to be more robust, converging with significantly smaller Krylov subspaces than those required to solve the global system of equations. Further improvement of performance are made through approximations and preconditioning of the Schur complement problem. Hence, the new Schur formulation shows promise as an affordable, robust, and scalable method to solve free-boundary, Stefan problems. Such models are employed to study a wide array of applications, including casting, welding, glass forming, planetary mantle and glacier dynamics, thermal energy storage, food processing, cryosurgery, metallurgical solidification, and crystal growth.

© 2010 Elsevier Inc. All rights reserved.

1. Introduction

The mathematical problem of finding the interface between melt and solid phases for the case of a general temperature field in a freezing liquid is classically referred to as Stefan's second problem [1]. Since the interface is an unknown function of the temperature field, the Stefan problem gives rise to a moving- or free-boundary problem of great relevance for modeling a variety of phase-change processes, including applications areas as diverse as casting, welding, glass forming, planetary mantle and glacier dynamics, thermal energy storage, food processing, and cryosurgery. The motivation for this work is the study of solidification and melt crystal growth processes, in which the kinetics of the phase change from liquid to solid occur very quickly and the location of the melt–solid interface is well represented by the melting-point isotherm in the system [2]. The solidification and crystal growth modeling literature is extensive, and we do not attempt to fully summarize it here; some overviews are available in Refs. [2–9].

In seeking a numerical solution of Stefan, free-boundary problems, the underlying field equations are discretized, including the temperature field but also often including fluid flow and other phenomena, which are then coupled with conditions

* Corresponding author.
E-mail address: derby@umn.edu (J.J. Derby).

0021-9991/$ - see front matter © 2010 Elsevier Inc. All rights reserved.
doi:10.1016/j.jcp.2010.06.046

that represent the melt–solid interface. Methods of representing the interface generally are of two basic types: fixed-grid and deforming-grid. A comprehensive comparison of the relative advantages and disadvantages of these approaches is beyond the scope of this paper. However, we note that fixed-grid methods, such as phase-field [10], enthalpy [11], level set [12], explicit front-tracking [13], volume-of-fluid [14], sharp interface [15], and extended finite element [16] methods, provide for great flexibility to represent topologically complicated melt–solid interfaces. On the other hand, deforming-grid methods make it possible to achieve high accuracy with much less grid refinement than typically required by fixed-grid methods, provided that the interface can be represented without excessive distortion of the computational cells. Indeed, with proper implementation of conditions at the interface, the accuracy of solving for the interface position is comparable to the accuracy of solving the underlying field variables; see, e.g., [17,7].

Even after the choice of field and interface representation, a solution strategy for the discretized problem, which is nonlinear, must be implemented. Most often, iterative solution algorithms for these problems have been constructed around Newton's method. Here, the important distinction is whether Newton's method is implemented in a manner that solves for both field and interface unknowns simultaneously or whether a decoupled approach is taken. A decoupled solution algorithm solves, in turn, the field equations with the interface fixed, followed by an update of the interface position. While easier to implement, this procedure will not yield the efficient, quadratic rate of problem convergence provided by a simultaneous Newton approach. Ettouney and Brown [17] conducted a landmark analysis of these issues and recommended a full Newton approach coupled with an implicit representation of the interface based on the melting point of the system, which they termed the Isotherm–Newton approach.

The Isotherm–Newton approach has proven to be very effective for the solution of two-dimensional problems when direct matrix solution techniques are employed. However, this method suffers when an iterative linear solver is employed to solve the linear equations of each Newton step. This is caused by the poorly conditioned Jacobian matrix that arises from applying the isotherm condition to form residual equations for determination of the melt–solid interface position. This choice, while central to the method's accuracy [17,7], formulates residual equations for the interface position that are implicitly defined via the temperature field variables, leading to a block of zeros along the Jacobian matrix diagonal. Indeed, the Isotherm–Newton formulation of the Stefan problem gives rise to a linear system of saddle point type, which typically proves to be extremely challenging for iterative solution. Benzi et al. [18] provide an extensive overview of the vast literature associated with these problems and the various techniques employed for their iterative solution.

In our experience, iterative linear solvers are prone to fail when applied to the global system of equations arising from the Isotherm–Newton approach. This is especially important for three-dimensional problems, where direct methods often prove to be too expensive to solve even the field equations alone, and linear iterative solvers need be employed. Such failures are rarely reported in the literature, though we shall state here that the results from our prior three-dimensional, Bridgman crystal growth simulation reported in [19] were only attained after the field and interface unknowns were solved using a decoupled iteration, where the field equations were solved on a fixed domain using Newton's method and GMRES, followed by an independent update of the interface position. While successful, this solution strategy resulted in linear convergence rates for the full problem. Notably, simultaneous solution of the entire problem via Newton's method, as posed by the Isotherm–Newton, proved infeasible for these computations using an iterative linear solver. In this case, convergence of the linear solution step was never obtained. Measures to improve performance were either ineffective, such as reordering the equations, or too expensive, notably increasing the size of the Krylov space used in GMRES. We note that other researchers attempting to solve such Stefan-like problems in three dimensions have also resorted to a decoupled solution strategy [20–24].

Hence, our motivation is to formulate a new solution approach that is faithful to the underlying ideas presented in the Isotherm–Newton approach but will result in a formulation far more amenable to solution via linear iterative solvers. We do this via a Schur complement decomposition of a block representation of Newton's method that separates the interface unknowns from the field variables. A more robust formulation will allow for the key advantage of the Isotherm–Newton approach, namely efficient, quadratic convergence, to be attained using scalable, parallel algorithms on large-scale systems.

2. Problem formulation

2.1. Governing equations

We construct a two-dimensional Stefan problem to represent a melt crystal growth process. Specifically, we consider a two-phase, heat-conduction problem in a cylindrical geometry that features a free-boundary representing the phase-change interface. Fig. 1(a) shows a schematic diagram of the geometry of the sample problem. This sample problem is based upon the Bridgman method employed to grow single crystals from the melt; more thorough discussions of this problem are available in Refs. [3,7–9,2].

The temperature field and coordinates are non-dimensionalized with respect to characteristic temperature and length scales, T_c and L, and we write steady-state conservation equations as:

$$\nabla^2 T_s = 0 \quad \text{in } \Omega_s \tag{1}$$

$$\kappa \nabla^2 T_m = 0 \quad \text{in } \Omega_m \tag{2}$$

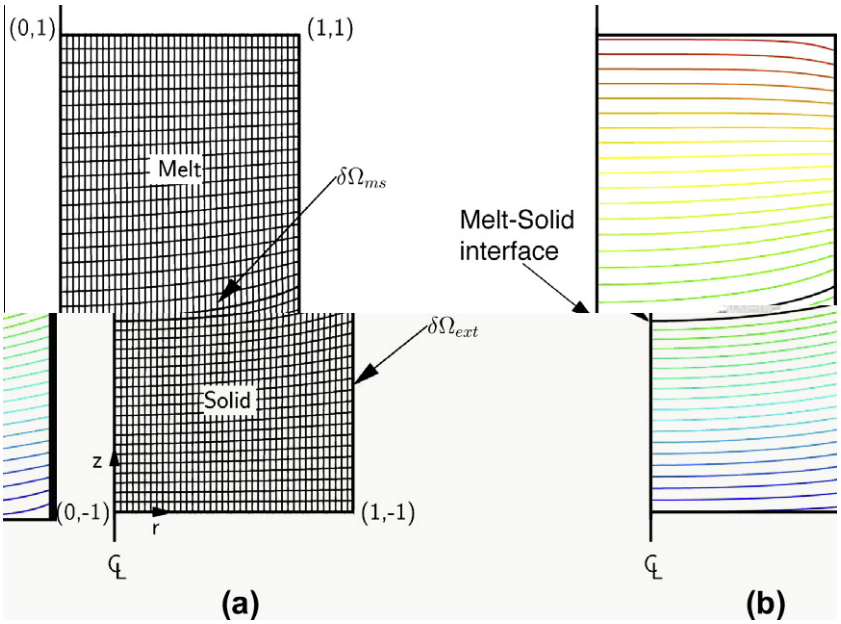

Fig. 1. Schematic of the two-dimensional, Stefan problem with melt and solid phases separated by an interface ($\delta\Omega_{ms}$) that is computed as part of the underlying thermal problem. (a) A representative 40×40 element mesh is shown at convergence along with (b) the solution of the temperature field (the melt–solid interface is located at the melting-point isotherm and is marked by the indicated boundary).

where ∇^2 represents the Laplacian operator, T is dimensionless temperature with subscripts s and m to denote the solid and melt domains over which it is defined, and $\kappa \equiv k_m/k_s$ is the ratio of melt to solid thermal conductivities.

Along the boundary between the melt and solid phases, $\delta\Omega_{ms}$, we impose several, required conditions:

$$\mathbf{n} \cdot \kappa \nabla T_s - \mathbf{n} \cdot \nabla T_m = PeS(\mathbf{n} \cdot \mathbf{e}_z) \tag{3}$$

$$T_s = T_m = T_{mp} \tag{4}$$

Eq. (3) ensures that conductive heat fluxes across the interface, represented on the left-hand-side, balance the amount of latent heat generated by the phase change, given by the product of the Peclet and Stefan numbers. Respectively, these dimensionless groups are defined by $Pe \equiv VL/\alpha$ and $S \equiv \Delta H_f/C_pT_c$, where V is the steady-state solidification rate, α is the thermal diffusivity of the solid, ΔH_f is the latent heat of phase change, and C_p is the heat capacity of the solid. Additional terms in this condition include the unit vector normal to the melt–solid interface, $\mathbf{n}$, the gradient operator, ∇, and the unit vector aligned with the solidification direction, $\mathbf{e}_z$ in this particular problem.

Eq. (4) is written as one condition but actually sets two distinct constraints. First, it demands that the temperature field be continuous between the domains of solid and melt. Second, it demands that the interface between the two domains is located at the melting-point temperature, T_{mp}. This has been referred to as the *isotherm* condition and arises from the mathematical description of a solidifying interface in the limit when phase-change kinetics occur on a time scale much faster than transport time scales in the system [2].

To complete the specification of the test problem, we impose the following boundary conditions:

$$\mathbf{n} \cdot \nabla T = Bi(T - T_f) \quad \text{along } \delta\Omega_{ext} \tag{5}$$

$$\mathbf{n} \cdot \nabla T = 0 \quad \text{along the centerline} \tag{6}$$

Note that in these boundary conditions, we simply refer to the temperature field using the variable T. A more strict interpretation of the multi-domain problem formulation would require the designation of either T_s or T_m; however, the choice is obvious in the context of the above conditions. Eq. (5) includes $\mathbf{n}$, an outward-pointing unit vector, $Bi \equiv hL/k_s$, a dimensionless group defined by a heat transfer coefficient, h. The nondimensional external temperature, T_f, represents an imposed thermal field from a directional solidification furnace (taken to be linear in the axial direction along the outer boundary and constant along the top and bottom of the problem domain), positioned so that the melting point falls somewhere within the problem domain and implicitly moving with a speed identical to the steady-state solidification velocity, V. Eq. (6) enforces symmetry conditions at the system centerline. We provide a much more detailed discussion of this formulation in Ref. [9].

2.2. Discretization and implementation

While the governing equations of the prior section fully define the Stefan, phase-change problem mathematically, considerable challenges are presented by their solution via numerical methods. Our overall solution strategy, now considered fairly routine for such problems, employs a Galerkin finite-element discretization of the conservation equations for the field variables, here the two-dimensional temperature field, using a mesh that is deformed to follow the shape of the phase-change interface. We discuss the most pertinent features below, emphasizing the challenges generic to the Stefan problem but downplaying the details of the actual mechanics of the implementation, specifically the method employed to deform the mesh. We emphasize that the solution approaches tested here are general for the Stefan problem and could be implemented with *any* mesh movement algorithm that is applied in a simultaneous iteration with the field variables.

We define a finite-element mesh over the problem domains, constructed so that the mesh deforms to force specific element edges to lie along the phase-change boundary, $\delta\Omega_{ms}$; see Fig. 1(a). Eqs. (1) and (2) are then discretized using the Galerkin finite-element method (specifically with a biquadratic, Lagrangian representation of the temperature field) and put into the weak form. The weak form provides for straightforward and explicit application of the flux boundary conditions represented by Eqs. (3), (5) and (6). Continuity of the temperature field along the phase-change boundary, $\delta\Omega_{ms}$, is automatically enforced by the construction of the underlying finite-element approximation and satisfies one of the constraints imposed by Eq. (4).

This deforming-mesh implementation conveniently and accurately satisfies several special requirements of the Stefan problem. First, a discontinuous temperature gradient across the melt–solid interface, arising as a consequence of Eq. (3), is naturally represented along the border of adjacent elements with C^0 continuity (as is the case for Lagrangian basis functions). While this effect can also be represented by discontinuity-capturing representations that allow the interface to run through a fixed computational cell [15,16], the simplicity of employing a non-moving mesh in these methods must be weighed against the more involved construction of a suitable, discontinuity-capturing basis. Second, our choice of enforcing the flux balance along the melt–solid interface, Eq. (3), via the weak form of the field equations, rather than supplying a Dirichlet boundary condition to set the interfacial temperature at the melting point, has been demonstrated by Ettouney and Brown [17] to yield a more accurate and convergent numerical solution.

After the above discretization and assignment of boundary conditions, there remains an unsatisfied requirement of the Stefan problem, namely that the specific temperature joining the melt and solid domains must exactly be the melting-point temperature. This condition, set by Eq. (4), asserts that the phase-change interface fall along the melting-point isotherm and is employed to provide information to locate the free boundary. A consideration of how this condition is implemented and satisfied yields some insight about the challenges of solving the Stefan problem. In a general sense, let us represent the position of the melt–solid interface as a vector function

$$\tilde{\mathbf{x}} - \mathbf{G}(\tilde{\mathbf{x}}) = \mathbf{0} \tag{7}$$

where $\tilde{\mathbf{x}}$ denotes the coordinates of the interface. For a free-boundary problem, our goal is to solve for $\mathbf{G}(\tilde{\mathbf{x}})$ in order to locate the interface. The *isotherm condition* demanded by Eq. (4) is then written as

$$T(\tilde{\mathbf{x}}) - T_{mp} = 0 \quad \text{along } \delta\Omega_{ms} \tag{8}$$

Rearranging the prior equation to $\tilde{\mathbf{x}} = \mathbf{G}(\tilde{\mathbf{x}})$ and substituting, we obtain

$$T(\mathbf{G}(\tilde{\mathbf{x}})) - T_{mp} = 0 \tag{9}$$

This isotherm condition is used to form the residual equations for the degrees of freedom associated with $\mathbf{G}(\tilde{\mathbf{x}})$, namely the interface unknowns.

Notice that Eq. (9) explicitly involves only the temperature field and is *implicit* with respect to the interface unknowns. As a consequence, zeros appear along the diagonals in portions of the Jacobian matrix, as discussed in the next section, thus rendering poor condition. Another, less-obvious outcome is that this condition introduces a strongly nonlinear equation into the otherwise linear problem considered here (with respect to the temperature field). This nonlinearity can be readily shown using an alternative formulation of the problem where the interface position is explicitly immobilized via a coordinate transformation; see, e.g., [17,25]. Of course, the solidification problem may include additional nonlinearities when effects such as melt convection and radiant heat transfer are included, but these are ignored in the test problem considered here.

As mentioned above, we implement a deforming-grid approach to track the melt–solid interface and construct a mesh to explicitly conform to its shape by placing specific element edges along it; see Fig. 1(a). Specifically, we employ elliptic grid generation techniques to govern the movement of the deforming mesh [9]. The mesh equations are discretized using biquadratic basis functions to locate the element nodes. The location of the melt–solid interface is solved by satisfying Eq. (9) via a Galerkin weak-form equation using a quadratic basis. The interface location provides boundary conditions to the mesh equations to locate specified nodes of the mesh (that define the boundary between melt and solid domains) along the melt–solid interface. Additional boundary conditions on the mesh equations set the shape of the outer domain while allowing the grid points to slide along the boundaries as needed; specific details can be found in Ref. [26].

It is important to realize that the mesh unknowns are included among all of the other problem unknowns in this formulation. Thus, when the problem is solved (via Newton iterations, as will be explained in the next section) all unknowns are updated simultaneously. We do not decouple the mesh movement from the solution of the field variables nor are field

variable values interpolated from one mesh to another. Indeed, the mesh is simply considered to be part of the overall solution to the problem. In this sense, the specific mechanism employed to move the mesh between iterations is rather unimportant. The same qualitative behavior of the solution algorithm will be displayed whether the mesh is determined by elliptic equations [27–29] (as is done here), by a pseudo-solid movement scheme [30–32], or by algebraic means [33–35]. All methods will give rise to the general forms considered in the following section.

2.3. Solution via Newton's method

We employ Newton's method to solve simultaneously for the field and free-boundary equations specified above. This approach was originally advocated as the Isotherm–Newton method by Ettouney and Brown [17]. To clarify the ensuing presentations, we note that a lower-case, bold symbol denotes a vector and an upper-case, bold symbol denotes a matrix.

After discretizing the steady-state field equations, implementing the boundary conditions, and applying a technique to track the free boundary, our problem is posed as a set of coupled, nonlinear algebraic equations:

$$\mathbf{r}(\mathbf{x}) = \mathbf{0} \tag{10}$$

where $\mathbf{r}$ is the residuals vector, a function of the discretized problem unknowns, $\mathbf{x}$. For reasons that will become clear below, we choose to order the problem unknowns as,

$$\mathbf{x} \equiv \begin{bmatrix} \mathbf{f} \\ \mathbf{i} \end{bmatrix} \tag{11}$$

where $\mathbf{f}$ is a vector of dimension n that contains the degrees of freedom associated with the field variables, here the temperature distribution and nodal coordinates of the mesh, and $\mathbf{i}$ is a vector of dimension m that contains only the unknowns associated with the representation of the interface. For most problems of this type, there are far fewer degrees of freedom associate with the description of the interface than of field quantities, so $m \ll n$.

We solve the nonlinear residual equation iteratively using Newton's method; each iteration has the form

$$\mathbf{J}(\mathbf{x}^{(\ell)})\boldsymbol{\delta}^{(\ell+1)} = -\mathbf{r}(\mathbf{x}^{(\ell)}) \tag{12}$$

where $\mathbf{r}$ is the residual vector, $\mathbf{J}$ is Jacobian matrix defined by $J_{ij} \equiv \partial r_i/\partial x_j$, $\boldsymbol{\delta}$ is the update vector, and the superscript ℓ denotes iteration number. Once the above linear equation is solved for $\boldsymbol{\delta}^{(\ell+1)}$, an updated solution vector is obtained via

$$\mathbf{x}^{(\ell+1)} = \mathbf{x}^{(\ell)} + \boldsymbol{\delta}^{(\ell+1)} \tag{13}$$

This procedure is continued until some suitable measure of convergence is attained.

For the ordering of unknowns adopted here, the linear system for each Newton iteration, Eq. (12), has the following block structure

$$\begin{bmatrix} \mathbf{A} & \mathbf{B} \\ \mathbf{C} & \mathbf{0} \end{bmatrix} \begin{bmatrix} \boldsymbol{\delta}_f \\ \boldsymbol{\delta}_i \end{bmatrix} = \begin{bmatrix} -\mathbf{r}_f \\ -\mathbf{r}_i \end{bmatrix} \tag{14}$$

where $\mathbf{A}$ is an $n \times n$ matrix which is typically banded, $\mathbf{B}$ is an $n \times m$ matrix which is usually dense, and $\mathbf{C}$ is an $m \times n$ matrix that is sparse. The block $\mathbf{0}$ is an $m \times m$ matrix that consists entirely of zeros; this arises from the choice of implementing the isotherm condition, Eq. (9), which is written only in terms of the temperature field unknowns, as a residual for the interface unknowns. Subscripts f and i denote quantities associated with field and interface variables, respectively. Despite the block-diagonal matrix of zeros in the lower-right quadrant, the overall matrix $\mathbf{J}$ is nonsingular and poses little difficulty for solution via a direct method. Note that the in the above Eq. (14) and in ensuing discussions, we do not explicitly indicate the iteration counter, rather we focus instead on how the linear system at each iteration is solved.

3. Implementation issues

3.1. Solvers

3.1.1. Method 1 (M1): direct solution

Solution of the Newton step via a direct solver, such as Gaussian elimination, has traditionally been very effective, yielding robust quadratic convergence for sufficiently good initial guesses. Using these solution methods, the block of zeros in the Jacobian matrix is usually not problematic because of fill-in during factorization, which is an advantage of the ordering of the unknowns specified in the prior section. In practice, pivoting strategies are also usually employed to avoid potential difficulties posed by the diagonal zeros. Finally, the block structure of Eq. (14) can be utilized for special direct solution strategies, such as band solvers augmented with bordering algorithms; see, e.g., [36–39].

The use of a direct solver for the Newton step represented by Eq. (14) is denoted as Method 1 (M1) in the ensuing tests.

3.1.2. Method 2 (M2): iterative solution of the global system

Even though the direct solution approach (M1) is quite robust, the computational effort scales nonlinearly with problem size, as will be discussed in more detail in Section 3.3. Especially for the solution of three-dimensional problems, the effort associated with each Newton step can become prohibitive. An alternative approach is the use of an iterative linear solver for the Newton step that may scale more favorably with problem size and may permit easier and more effective parallelization compared to direct solution methods.

Hence, we denote Method 2 (M2) as the use of an iterative solver for the full Newton step represented by Eq. (14). We specifically employ the restarted Generalized Minimal Residual (GMRES) procedure of Saad and Schultz [40] with no preconditioning. However, the rows of the Jacobian matrix are scaled with respect to the ℓ_2 row norm prior to iterations. We choose not to precondition for two reasons. First, we desire to keep the solution algorithms as simple as possible. Second, the underlying Laplacian equations for the temperature field, Eqs. (1) and (2), lead to a discrete system (notably block matrix $\mathbf{A}$ of Eq. (14)) that is symmetric and diagonally dominant and not in need of preconditioning for convergence. We comment in Section 3.2 on preconditioning techniques for the more difficult part of the problem (the remaining blocks of Eq. (14)) using the Schur method explained in the next section.

3.1.3. Method 3 (M3): iterative solution of the Schur complement

Our past experience in solving large-scale simulations of crystal growth from the melt indicates that iterative solution strategies for the full problem (i.e., solution strategy M2 described previously) often fail, thus motivating a different approach. Using block elimination, Eq. (14) can be rewritten as

$$\begin{bmatrix} \mathbf{A} & \mathbf{B} \\ \mathbf{0} & \mathbf{S} \end{bmatrix} \begin{bmatrix} \boldsymbol{\delta}_f \\ \boldsymbol{\delta}_i \end{bmatrix} = \begin{bmatrix} -\mathbf{r}_f \\ \mathbf{r}_S \end{bmatrix} \tag{15}$$

where $\mathbf{0}$ is an $m \times n$ matrix of zeros, $\mathbf{S} \equiv -\mathbf{C}\mathbf{A}^{-1}\mathbf{B}$ is an $m \times m$ matrix known as the Schur complement, and $\mathbf{r}_S \equiv -\mathbf{r}_i + \mathbf{C}\mathbf{A}^{-1}\mathbf{r}_f$ is its modified right-hand-side.

The solution of this system of equations can proceed in a decoupled manner. Namely, block back-substitution is used to first solve for the interface update vector $\boldsymbol{\delta}_i$ and then for the field update vector $\boldsymbol{\delta}_f$ via

$$\mathbf{S}\boldsymbol{\delta}_i = \mathbf{r}_S \tag{16}$$

$$\mathbf{A}\boldsymbol{\delta}_f = -\mathbf{r}_f - \mathbf{B}\boldsymbol{\delta}_i \tag{17}$$

We expect that this strategy may pose advantages over the original form, Eq. (14), when an iterative linear solver is employed. Specifically, Eq. (16) is a much smaller, $m \times m$ system which may allow for more affordable solution, even if a larger Krylov space is required for the solution of this subproblem. At the same time, solution of the $n \times n$ system of Eq. (17) should be no more difficult to solve than the original field equations on a fixed domain, since the left-hand-side contains only $\mathbf{A}$. The Schur complement formulation therefore represents a "divide and conquer" strategy, where the field and interface equations can be separated, with different solution strategies applied.

The actual implementation of this strategy is complicated by the need to compute the action of $\mathbf{A}^{-1}$ in forming both the Schur complement matrix, $\mathbf{S}$, and the modified right-hand-side vector, $\mathbf{r}_S$ (as indicated by their definitions after Eq. (15) above). This would pose little difficulty if we employed a method to directly factorize $\mathbf{A}$ into LU form which could subsequently be reused with little cost. However, the point of the reformulation of the current problem into Schur complement form is to more readily apply iterative linear solution techniques. We will return to this issue after outlining the general approach in what follows.

Ignoring for the moment how we will actually perform the computations, our strategy is to first form the modified right-hand-side,

$$\mathbf{r}_S = -\mathbf{r}_i + \mathbf{C}\mathbf{A}^{-1}\mathbf{r}_f \tag{18}$$

Vectors $\mathbf{r}_i$ and $\mathbf{r}_f$ and block matrix $\mathbf{C}$ are readily available, but the action of $\mathbf{A}^{-1}$ must be computed. We proceed by solving the linear system of equations

$$\mathbf{A}\mathbf{z} = \mathbf{r}_f, \tag{19}$$

for the n-dimensional vector $\mathbf{z}$. This is immediately followed by the calculation of the modified right-hand-side using the placeholder $\mathbf{z} \equiv \mathbf{A}^{-1}\mathbf{r}_f$ as

$$\mathbf{r}_S = -\mathbf{r}_i + \mathbf{C}\mathbf{z} \tag{20}$$

Next, we must compute the Schur complement matrix

$$\mathbf{S} = -\mathbf{C}\mathbf{A}^{-1}\mathbf{B} \tag{21}$$

Note that matrices $\mathbf{B}$ and $\mathbf{C}$ are available, but again we need to compute the action of $\mathbf{A}^{-1}$. We use a similar technique here as used above for Eq. (19). Here, we solve m systems of linear equations of the form

$$\mathbf{A}\mathbf{Y} = \mathbf{B} \tag{22}$$

to obtain the $n \times m$ placeholder matrix $\mathbf{Y}$. The Schur complement can then be calculated as

$$\mathbf{S} = -\mathbf{CY} \tag{23}$$

The first step of the Newton update is completed by using the computed values of $\mathbf{S}$ and $\mathbf{r}_S$ to solve Eq. (16) for the interface update vector, $\boldsymbol{\delta}_i$. For clarity, this equation is repeated below:

$$\mathbf{S}\boldsymbol{\delta}_i = \mathbf{r}_S \tag{24}$$

We remark that this step involves solving only an m-dimensional system; however, the series of calculations needed to form for the Schur complement, $\mathbf{S}$ and its right-hand-side, $\mathbf{r}_S$ required $m + 1$ solves of n-dimensional systems involving $\mathbf{A}$.

The second step of the Newton update is attained by computing the field update vector, $\boldsymbol{\delta}_f$, from Eq. (17). However, it turns out that we have already done the work needed to solve this equation. We formally rearrange Eq. (17) to

$$\boldsymbol{\delta}_f = -\mathbf{A}^{-1}\mathbf{r}_f - \mathbf{A}^{-1}\mathbf{B}\boldsymbol{\delta}_i \tag{25}$$

and realize that the first term of the right-hand-side has already been computed as $\mathbf{z} \equiv \mathbf{A}^{-1}\mathbf{r}_f$ from Eq. (19) and the second term has already been computed as $\mathbf{Y} \equiv \mathbf{A}^{-1}\mathbf{B}$ from Eq. (22). The final step is then trivial to compute as

$$\boldsymbol{\delta}_f = -\mathbf{z} - \mathbf{Y}\boldsymbol{\delta}_i \tag{26}$$

Now, let us return to the implementation of this approach using a Krylov-subspace-based, iterative linear solver. We realize that several steps from the prior procedures involved the solution of linear systems, namely Eqs. (19), (22) and (24). To solve these systems, we will employ restarted, unpreconditioned GMRES with the same row scaling used in Method 2 of the prior section. Specifically, before dividing into the block components $\mathbf{A}$, $\mathbf{B}$, and $\mathbf{C}$, the rows of the global Jacobian matrix are scaled with respect to the ℓ_2 row norm.

We begin by repeating the Schur subproblem that constitutes the first step of the Newton iteration,

$$\mathbf{S}\boldsymbol{\delta}_i = \mathbf{r}_S \tag{27}$$

On the surface, we expect that this problem may be rather inexpensive to compute, since $\mathbf{S}$ is a small, $m \times m$ matrix. To get started, we employ GMRES to solve Eq. (19) for $\mathbf{z}$ and use its solution to form $\mathbf{r}_S$ via Eq. (20). This constitutes one independent solve of an n-dimensional system using our iterative linear solver.

With the newly formed right-hand-side $\mathbf{r}_S$, we construct a method to use our Krylov-based solver to obtain $\boldsymbol{\delta}_i$ from Eq. (27). A suitable candidate solution for $\boldsymbol{\delta}_i$ is constructed from the Krylov subspace defined by $\mathcal{K}_S = \text{span}\{\mathbf{v}, \mathbf{Sv}, \mathbf{S}^2\mathbf{v}, \ldots\}$, where $\mathbf{v}$ is the initial Krylov vector, in this case set to $\mathbf{v} = \mathbf{r}_S$. To proceed, we must be able to evaluate the products of $\mathbf{S}$ multiplied by the initial Krylov vector $\mathbf{v}$. Formally, this would require the computation of

$$\mathbf{Sv} = -\mathbf{CA}^{-1}\mathbf{Bv} \tag{28}$$

We could compute the terms in Eq. (28) following the general procedure described above. Namely, we would employ a Krylov-based solver for Eq. (22) to construct approximations to each of the columns of $\mathbf{Y}$ from Krylov subspaces of the form $\mathcal{K}_A = \text{span}\{\mathbf{v}, \mathbf{Av}, \mathbf{A}^2\mathbf{v}, \ldots\}$, where the initial Krylov vector, $\mathbf{v}$, is set by the corresponding column of $\mathbf{B}$. The resulting matrix of solutions, $\mathbf{Y} \equiv \mathbf{A}^{-1}\mathbf{B}$, would then be multiplied by $\mathbf{C}$ and $\mathbf{v}$ in Eq. (28). This approach requires m independent iterative solutions of Eq. (22), an $n \times n$ system of equations. However, once $\mathbf{Y}$ is computed, it can be reused to compute successive Krylov vectors with no additional solves.

Seeking a cheaper alternative, we define a new vector, $\mathbf{w}$, which is computed from known quantities as

$$\mathbf{w} \equiv \mathbf{Bv} \tag{29}$$

We now employ this as a right-hand-side for the $n \times n$ system of equations

$$\mathbf{Au} = \mathbf{w} \tag{30}$$

Solution of this single equation yields $\mathbf{u} = \mathbf{A}^{-1}\mathbf{Bv}$. We now form the product of $\mathbf{Sv}$ as follows

$$\mathbf{Sv} = -\mathbf{Cu} \tag{31}$$

(compare to Eq. (28)). The net result is that we have computed $\mathbf{Sv}$ using only a single iterative solve of the n-dimensional system of Eq. (30), a significant reduction of effort over the prior approach. Yet, successive Krylov vectors must be evaluated by forming a new right-hand-side vector and performing an additional iterative solve of Eq. (30). Nevertheless, we expect to obtain an accurate solution of the original m-dimensional Eq. (27) with, at most, m Krylov vectors (m vectors would, in theory, yield a perfect solution for the case of exact arithmetic) and, likely, significantly fewer will be needed. So, we choose this alternative formulation, since it will likely be cheaper and never more expensive than the prior procedure.

With Eq. (27) solved, the interface update vector, $\boldsymbol{\delta}_i$, is available and the first step of the Newton iteration is complete. The second step is straightforward; the known quantities $\mathbf{r}_f$, $\mathbf{B}$, and $\boldsymbol{\delta}_i$ are used to construct a vector,

$$\mathbf{q} \equiv -\mathbf{r}_f - \mathbf{B}\boldsymbol{\delta}_i \tag{32}$$

Table 1
Algorithm for the iterative Schur complement solution of a single Newton step.

I. **Solve** $\mathbf{S}\delta_i = \mathbf{r}_S$ **for** δ_i
  A. Compute $\mathbf{r}_S = -\mathbf{r}_i + \mathbf{C}\mathbf{A}^{-1}\mathbf{r}_f$
    1. Solve $\mathbf{A}\mathbf{z} = \mathbf{r}_f$ for z using iterative solver
    2. Compute $\mathbf{r}_S = -\mathbf{r}_i + \mathbf{C}\mathbf{z}$

  B. Iteratively solve $\mathbf{S}\delta_i = \mathbf{r}_S$, where $\mathbf{S} = -\mathbf{C}\mathbf{A}^{-1}\mathbf{B}$
    1. Form $\mathbf{w} = \mathbf{B}\mathbf{v}$ where $\mathbf{v}$ is the prior Krylov vector associated with the iterative solve of B.
    2. Solve $\mathbf{A}\mathbf{u} = \mathbf{w}$ for vector $\mathbf{u}$ using iterative solver
    3. Form $\mathbf{S}\mathbf{v} = -\mathbf{C}\mathbf{u}$ for the next Krylov vector
    4. Repeat Steps B.1–3 until enough Krylov vectors are attained for the iterative solve of B.

II. **Solve** $\mathbf{A}\delta_f = -\mathbf{r}_f - \mathbf{B}\delta_i$ **for** δ_f
  A. Form $\mathbf{q} = -\mathbf{r}_f - \mathbf{B}\delta_i$
  B. Solve $\mathbf{A}\delta_f = \mathbf{q}$ for δ_f using iterative solver

which is the right-hand-side of Eq. (17). Then this equation, repeated here using $\mathbf{q}$,

$$\mathbf{A}\delta_f = \mathbf{q} \tag{33}$$

is solved for the field variable update vector, δ_f, using a linear iterative solver. With both update vectors now available, the entire solution is updated and the Newton step is repeated until convergence is attained. The algorithm to compute a single Newton iteration is summarized in Table 1.

The major cost of this procedure is the effort expended computing the interface update vector, δ_i, via Eq. (27) and associated operations. In fact, the computation of δ_i requires two nested calls to an iterative linear solver, which we label as outer and inner iterations. The *outer* iteration constructs the candidate solution for δ_i from a successively larger Krylov subspace $\mathcal{K}_S = \{\mathbf{v}, \mathbf{S}\mathbf{v}, \mathbf{S}^2\mathbf{v}, \ldots\}$; however, the equation set that defines δ_i is a relatively small system of dimension m (refer to step I.B. above). The *inner* iteration is invoked for each new Krylov vector required by the outer iteration and arises from the iterative solution of the much larger, n-dimensional system, Eq. (30); refer to step I.B.2. above.

If k_S denotes the number of outer iterations needed to solve for δ_i, then the entire Newton step for Method 3 will require $k_S + 2$ solves of n-dimensional systems. At first glance, this effort may appear to be substantially more than the one iterative solve of the global system (of dimension $n + m \approx n$) needed for each Newton step in Method 2. However, k_S will likely be small; it is guaranteed to be bounded by $k_S \leqslant m \ll n$. More significantly, the block matrix $\mathbf{A}$ in the iterative solves of Method 3 is expected to result in substantially better behavior than that expected from iterative solves involving the global Jacobian matrix $\mathbf{J}$ in Method 2. Thus, the solves associated with the Schur complement form of Method 3 may be substantially easier and cheaper than those required for Method 2. These ideas will be further explored via computations on test problems presented in the next section.

3.2. Preconditioning

We expect the iterative solution of the linear systems involving block matrix $\mathbf{A}$ in M3 to be relatively easy, due to its underlying form (although its $n \times n$ dimension will never make it cheap). Indeed, our results, discussed below, show that the simple, *a priori* row scaling is enough to allow convergence without additional preconditioning.

Benefits may be obtained, however, from preconditioning the Schur subproblem, Eq. (27), thus we consider constructing an approximation of $\mathbf{S}^{-1}$ to be used in this manner. We propose the following:

$$\widetilde{\mathbf{S}}^{-1} = -\left[\mathbf{C}(\mathrm{diag}(\mathbf{A}))^{-1}\mathbf{B}\right]^{-1} \tag{34}$$

where we have replaced $\mathbf{A}$ in the original definition of $\mathbf{S}$ with its diagonal matrix, whose inverse is trivial to form. Then $\widetilde{\mathbf{S}}$ can be computed using a series of matrix multiplications.

We intend to apply $\widetilde{\mathbf{S}}$ as a left preconditioner on Eq. (27), giving

$$\widetilde{\mathbf{S}}^{-1}\mathbf{S}\delta_i = \widetilde{\mathbf{S}}^{-1}\mathbf{r}_S \tag{35}$$

as the equation to solve with GMRES. The Krylov subspace for this problem becomes $\mathcal{K}_{\widetilde{S}} = \{\mathbf{v}, \widetilde{\mathbf{S}}^{-1}\mathbf{S}\mathbf{v}, \ldots\}$, where the initial Krylov vector $\mathbf{v}$ is now defined as $\mathbf{v} \equiv \widetilde{\mathbf{S}}^{-1}\mathbf{r}_S$.

The inverse of $\widetilde{\mathbf{S}}$ must be calculated to implement Eq. (35). However, this is relatively cheap and easy, since $\widetilde{\mathbf{S}}$ is a small, $m \times m$ matrix. Here, we accomplish this via an additional inner iteration nested within the loop computing the $\mathbf{S}\mathbf{v}$ products, whereby the action of $\widetilde{\mathbf{S}}^{-1}$ is computed by solving

$$\widetilde{\mathbf{S}}\mathbf{s} = \mathbf{S}\mathbf{v} \tag{36}$$

where $\mathbf{s}$ is the Krylov vector of the new subspace $\mathcal{K}_{\widetilde{S}}$ described above. The procedure is repeated until sufficient number of Krylov vectors are accumulated for a satisfactory solution to Eq. (35). In our computations, we rely on GMRES to solve

for Eq. (36), although its direct solution using LU decomposition, while more expensive, would also likely be effective. The additional computational expense of this added solve of an $m \times m$ system is insignificant.

3.3. Scaling of computational effort

An analysis of the precise scaling of computational effort for the three approaches requires an in-depth consideration of the discretization of the Stefan problem, including the underlying details of representation of the field quantities, the method employed for front tracking, and the problem geometry and meshing, as well as the solver details, most notably the size of the Krylov spaces employed or other convergence criteria associated with the iterative linear solvers. Such details are discussed in derivations presented in [41]. Here, we outline the leading-order effects that scale computation effort.

We first simplify our analysis by restricting factors that arise from the underlying discretization. Namely, we consider a two-dimensional problem domain that is discretized by a structured mesh containing an equal number of nodes, N_n along each edge, so that the total number of nodes is given by N_n^2. We then assume that the total number of unknowns in the problem, N_t, is simply proportional to the total number of nodes, i.e., $N_t \propto N_n^2$. Note that we may also relate these quantities to those already introduced to quantify the block matrices, so that $N_t = n + m \approx n$ (relying on the assumption that $m \ll n$) and $N_n \approx m$ (which assumes that the one-dimensional free boundary is discretized in a manner comparable to an edge of the domain).

The computational effort associated with M1 is proportional to the number of operations needed to factorize a banded matrix of total dimension N_t and with a bandwidth proportional to N_n, which is a generic outcome of any discretization of a two-dimensional domain. Thus, the computational work associated with each Newton iteration using direct solution, which we denote as W_{M1}, scales to leading order as

$$W_{M1} \propto N_t N_n^2 = N_n^4 \tag{37}$$

Method 2 applies GMRES to solve the linear equation arising from the global Newton step, and the total operation count is dominated by Householder vector generation and orthogonalization, leading to the estimate of computational work of

$$W_{M2} \propto N_t k_J^2 = N_n^2 k_J^2 \tag{38}$$

where k_J is the size of the Krylov subspace employed.

Finally, the total effort required by the iterative Schur complement formulation, M3, depends on several pieces; however, the most expensive involves the nested iterative solves associated with the Schur subproblem. To leading order, we estimate the computational effort as

$$W_{M3} \propto N_t k_S k_A^2 = N_n^2 k_S k_A^2 \tag{39}$$

where k_S and k_A are the sizes of the Krylov subspaces employed to solve the outer and inner solves of the Schur subproblem, respectively.

4. Results and discussion

The model Stefan problem described in Section 2 is employed in the ensuing tests of the different solution strategies. Physical properties are based on the melt and solid phases of cadmium zinc telluride, a II–VI semiconductor crystal of strategic importance for radiation detection and homeland security [42–44], given in Lun et al. [45]. For this test problem we employ the following values for the dimensionless parameters: $\kappa = 0.5$, $PeS = 10^{-3}$, $T_{mp} = 0$, $Bi = 10$, and $T_f = z$. Fig. 1(b) shows the solution of the temperature field and corresponding melt–solid interface computed on the 40×40 element, deforming-grid mesh shown, at convergence, in Fig. 1(a). The same initial guess is employed for each mesh and calculation.

For all cases, we deem the Newton iterations on the Stefan problem to be convergent when both the ℓ_2 norms of the residuals and the update vectors are less than 10^{-4} that of the solution vector. When applying the procedures based on iterative linear solvers, i.e., Methods 2 and 3, we either perform a set number of iterations as determined by the size of the Krylov subspace, as in Sections 4.1 and 4.3, or we iterate the linear solver until the ℓ_2 norm of the linear system residuals vector is less than a certain tolerance, as in Section 4.2. Different values for these limits will be considered in the ensuing tests.

4.1. Convergence

For the cases considered in this section, we employ a 20×20 element mesh over the computational domain. This discretization leads to a problem characterized by 41 nodes along the edges of the domain for the biquadratic, finite-element discretization. This rather small case comprises 1681 temperature unknowns, 3362 mesh unknowns, and 41 unknowns associated with the interface position, yielding 5084 total unknowns. When this problem is put into block form, there are $n = 5043$ field (mesh and temperature) unknowns and $m = 41$ interface unknowns.

When this system is solved using our standard direct solver (M1), the system exhibits quadratic convergence and requires just three Newton iterations. Fig. 2 shows the performance of the iterative solver-based methods M2 (full Jacobian) and M3 (Schur complement) by plotting the number of Newton iterations required for the solution to reach convergence as a

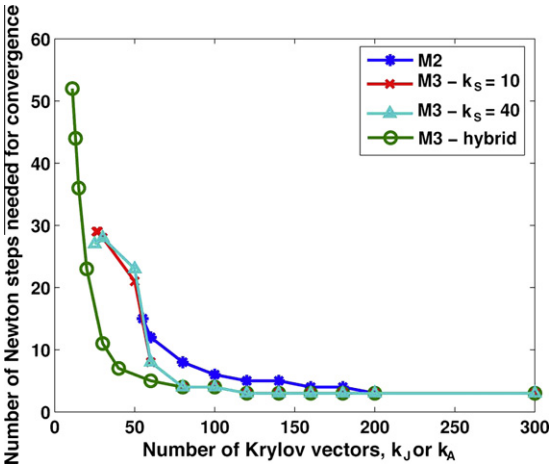

Fig. 2. Comparision of number of Newton iterations required to reach convergence as a function of number of (inner) Krylov vectors.

function of the number of Krylov vectors employed in the solves of the larger systems. Here, k_J denotes the size of the Krylov subspace employed for the solution of Eq. (14) in M2, involving one solve with the full Jacobian matrix, **J**, per Newton iteration. For M3, k_A represents the number of Krylov vectors applied to solve Eqs. (19) and (33) (once each per Newton iteration) and the inner iterations that repeatedly solve Eq. (30) a total of k_S times (the number of inner iterations performed) in each Newton step; all of these solves involve block matrix **A**. Both k_J and k_A are plotted along a common abscissa in the figure.

The number of Newton iterations required for convergence using Method 2, shown by the M2 curve in Fig. 2, decreases as k_J increases, reflecting the more accurate solutions obtained for each Newton step as the size of the Krylov subspace is increased. The system failed to converge for M2 when $k_J < 55$, while strong quadratic convergence (as judged by convergence of the nonlinear problem residual in three Newton steps per the performance of M1) was achieved for cases when $k_J > 180$.

The performance of M3 depends on the size of k_A and on the number of Krylov vectors employed for the outer iteration, k_S, to solve Eq. (27). Fig. 2 shows outcomes for two cases of $k_S = 10$ and $k_S = 40$. As for the case of M2, a larger-sized Krylov subspace, reflected by larger k_A, generally results in a reduced number of Newton iterations required for convergence for both k_S parameters. In addition, both cases give a more reliable solution procedure than M2 for smaller Krylov subspaces, as evidenced by convergence for $k_J \geqslant 26$ when $k_S = 10$ and $k_J \geqslant 25$ when $k_S = 40$. In addition, M3 requires fewer Newton steps for convergence for nearly all cases for comparable Krylov approximations, i.e., when $k_A = k_J$. Of course, the downside for this increase in robustness is a considerable increase in the cost per iteration, a factor of at least $2 + k_S$ times greater than M2. Curiously, M3 shows little improvement in behavior as a function of k_S for the test problem considered here, which suggested the analysis described next.

The case of $k_S = 40$ represents the situation where the Krylov subspace size is the same size (including the initial vector) as Eq. (27), with $m = 41$, thus the solution to the Schur complement subproblem should be nearly exact. However, this does not imply that the interface update vector, $\boldsymbol{\delta}_i$, is exact, since there is some error in solving the inner problem, Eq. (30), that effectively approximates the Schur matrix, **S**. That $\boldsymbol{\delta}_i$ may be computed inaccurately also affects the accuracy of the field update vector, $\boldsymbol{\delta}_f$, since $\boldsymbol{\delta}_i$ is need to form the right-hand-side vector of Eq. (33). To remove the errors associated with an inaccurate computation of the interface update vector, we pose a hybrid implementation of the Schur method, where we compute an "exact" update, $\boldsymbol{\delta}_i$, by evaluating the Schur matrix, **S**, and the modified right-hand-side vector, $\mathbf{r}_S$, using a direct factorization of $\mathbf{A}^{-1}$ and by solving the Schur subproblem, Eq. (27), using a direct method. The remaining field update vector is then computed via solution of Eq. (33) by GMRES with a Krylov subspace of size k_A. Of course, this mixed solution approach is not a practical implementation of M3, since the cost of the "exact" solve of the Schur complement subproblem is nearly as expensive as the cost of the direct approach, M1.

The effect of eliminating the error associated with the interface update is shown in Fig. 2 by the curve labeled M3-hybrid. This hybrid approach proves to be quite robust, with convergence attained for Krylov subspaces as small as $k_A = 11$. In addition, substantially fewer iterations are required for intermediate values of $k_A \approx 20 - 60$. Clearly, controlling the error associated with computing the update vector, $\boldsymbol{\delta}_i$, is important for improving the performance of the Schur complement formulation.

A summary of the conditions needed for Newton's method to converge and to converge quadratically for each case is provided in Table 2.

4.2. Tolerances and preconditioning

In the tests of this section, we consider the effect of changing convergence criteria and preconditioning (as described in Section 3.2) on the performance of the Schur complement approach, M3. The case studies are performed using a 40×40 element mesh that yields $n = 19{,}683$ field (mesh and temperature) unknowns and $m = 81$ interface unknowns.

Table 2
The minimum number of Krylov vectors (k_J for M2 and k_A for M3) needed for Newton's method to converge to the solution and to converge quadratically.

	M2	M3, $k_S = 10$	M3, $k_S = 40$	M3, hybrid
Solution to converge	55	26	25	11
Quadratic convergence	181	110	108	108

Table 3
Effect of tolerances and Schur preconditioning on the number of Newton iterations required to convergence and on number of iterations of outer solver during first Newton iteration (# Newton iterations/# Outer iterations). Cases of failure of Newton iteration to converge are indicated.

ϵ_o	ϵ_i	Preconditioning	
		off	on
1×10^{-4}	1×10^{-5}	3/16	3/16
1×10^{-4}	1×10^{-4}	4/16	3/16
1×10^{-3}	1×10^{-4}	4/13	3/13
1×10^{-3}	1×10^{-3}	fail (2)/13	4/13
1×10^{-2}	1×10^{-3}	fail (1)/8	4/9

In these tests, we vary the convergence tolerance of the outer and inner solver loops, ϵ_o and ϵ_i, respectively, for the use of M3 both without and with preconditioning of the Schur subproblem. Table 3 lists the number of Newton iterations needed for convergence and the number of outer iterations (i.e., the iterations needed to solve Eq. (27) or (35) to the convergence level specified) per Newton iteration for the various cases. The number of outer iterations is an important measure of M3, since it corresponds to the number of Krylov vectors k_S used in the computation and thus represents the approximate factor of additional computational effort required by M3 over M2 (see Eqs. (39) and (38) under the assumption that $k_A \approx k_J$).

The outcome of using nominal values of $\epsilon_o = 1 \times 10^{-4}$, $\epsilon_i = 1 \times 10^{-4}$, as done for all prior computations, is shown on the second row of Table 3. Notice that, while not changing the number of outer iterations, the application of preconditioning does reduce the number of Newton iterations required for convergence, presumably by improving the accuracy of computing the interface update vector, $\boldsymbol{\delta}_i$. A similar outcome can be obtained by reducing the inner iteration tolerance to $\epsilon_i = 1 \times 10^{-5}$ for the same value of $\epsilon_o = 1 \times 10^{-4}$, which would also increase the accuracy of computing $\boldsymbol{\delta}_i$.

Reducing the outer iteration tolerance to $\epsilon_o = 1 \times 10^{-3}$ reduces the number of outer iterations required (and thus the overall computational effort) with no change in the convergence behavior of the nominal case; however, preconditioning again results in one fewer Newton iteration; see row three of the table. Further reducing the tolerances causes the unpreconditioned system to fail to converge, while preconditioning allows for convergence of the Newton steps with as few as $k_S = 9$ outer iterations on the Schur subproblem.

4.3. Scaling of computational effort

The results of the previous section addressed issues of robustness and convergence of the solution methods. Here, we attempt to address the equally important issue of how the computational effort scales for each method as the size of the underlying problem grows. We examine cases corresponding to three different mesh sizes with the same number of elements in each direction, $N_e \times N_e$, for $N_e = 20$, 40, and 80 elements. The sizes of the matrices associated with the two smaller meshes were described in the two prior sections; the 80×80 mesh produced a system with $n = 77{,}763$ and

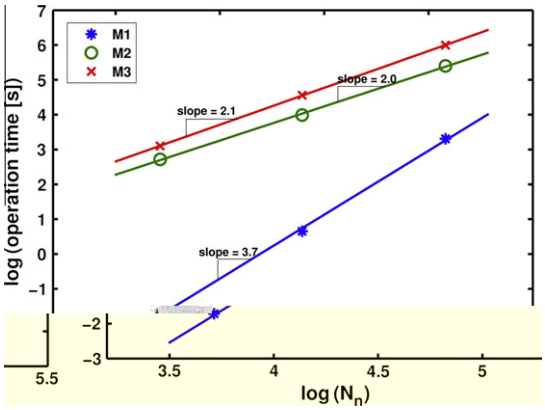

Fig. 3. Plot of computational effort versus problem size for one Newton step using M1, M2, and M3.

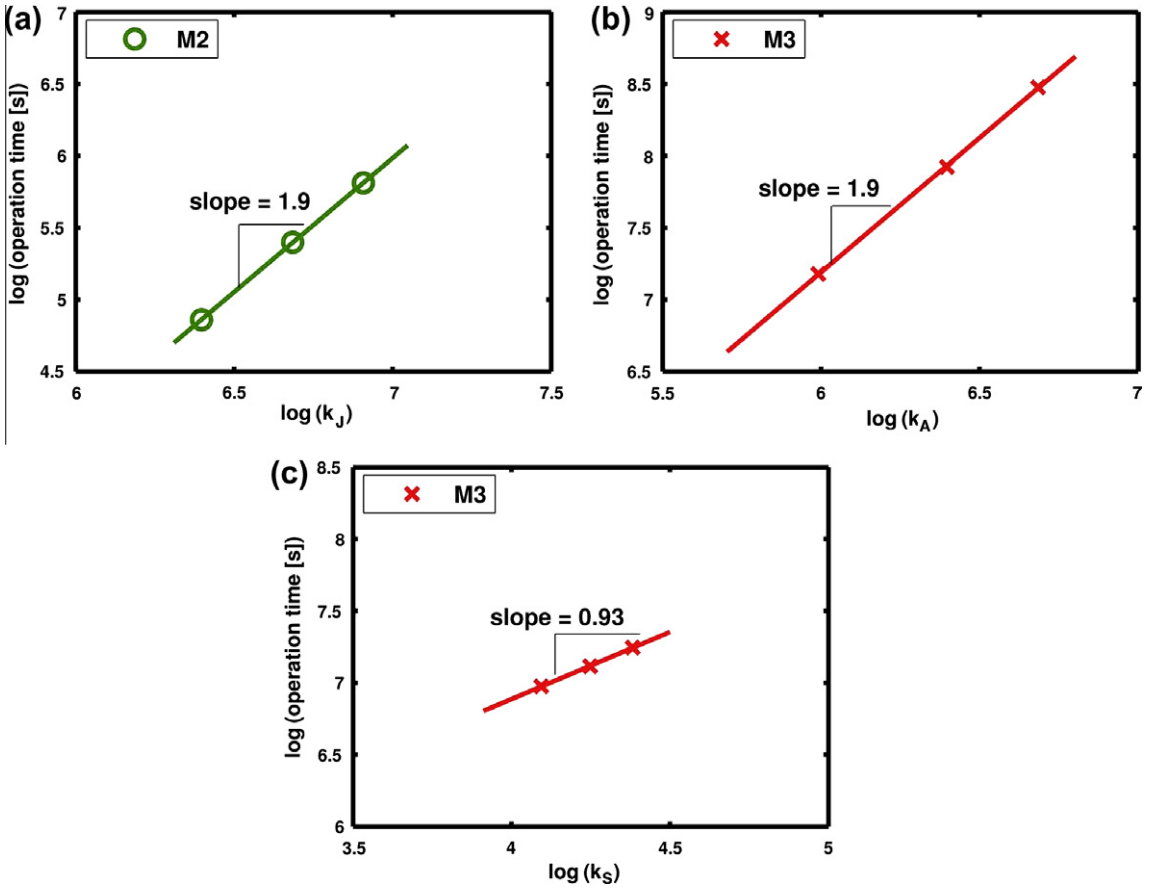

Fig. 4. Scaling of computational effort of (a) method M2 on k_J, (b) method M3 on k_A, and (c) method M3 on k_S.

$m = 161$. Tests were conducted by measuring the amount of time needed to perform one Newton step on a dedicated computer on each of these meshes for the three solution procedures considered here.

Fig. 3 plots the log of computation time for one Newton step versus the log of N_n for the three methods. For M2, the number of Krylov vectors was fixed at $k_J = 800$. For M3, the number of outer and inner Krylov vectors were fixed at $k_S = 20$ and $k_A = 200$. From the slope of the linear fit to the three computations, the computational effort scalings predicted in Section 3.3 are confirmed, namely $W_{M1} \propto N_n^4$ and both W_{M2} and W_{M3} are proportional to N_n^2.

Fig. 4 shows how M2 and M3 solutions computed on the 80×80 mesh scale with the number of Krylov vectors used by GMRES. Again, the plots of the log of computation time versus the log of number of Krylov vectors confirm the analyses presented above. Fig. 4(a) indicates that the work needed for a Newton step via M2 increases with the square of the number of Krylov vectors, $W_{M2} \propto k_J$. Similarly, Fig. 4(b) and (c) confirm that a Newton step in M3 scales proportionally to k_A^2 and k_S, respectively.

5. Concluding remarks

We have presented a new, Schur complement method to solve a free-boundary, Stefan problem comprising a melt–solid interface and its associated heat transfer problem, similar to those that arise in solidification and melt crystal growth models, via a deforming-grid technique. The motivation of this approach is to employ a full Newton iteration strategy that simultaneously solves for the field and interface unknowns, while casting the problem in a form that is more amenable to solution via iterative linear solvers. The use of these solvers will readily allow for parallel techniques to compute large-scale problems.

The initial studies presented here show promise for this new approach, especially for larger computations needed to solve three-dimensional problems. The traditional direct solution of the global Newton iteration (i.e., the Isotherm–Newton method) works extremely well to solve the two-dimensional test problem posed here, yielding quadratic convergence and requiring very few iterations. However, the poor scaling of computational effort of such direct methods makes strategies based upon linear iterative solvers more attractive as the problem size grows. This advantage, however, may be negated if the iterative linear technique fails to accurately solve for a Newton step. In this aspect, the Schur approach is encouraging.

There is evidence from the two-dimensional problem solved here that the Schur complement formulation promises to be more robust than iterative solution of the global system. As shown by the results of Section 4.1, Newton steps solved using GMRES were effective in solving the Stefan problem, even resulting in quadratically converging algorithms when a sufficiently large number of Krylov vectors were employed. Notably, for a fixed problem size, the Schur complement approach (M3) allowed for convergence in as few as half the number of Krylov vectors needed to successfully converge when a linear iterative solver was applied to the original, global Newton step (M2). Results from a hybrid-Schur implementation indicated that even better robustness is possible when the accuracy of the interface update vector, computed from the Schur subproblem, is improved. Consistent with this assertion, the robustness and efficiency of the Schur approach is improved by preconditioning the Schur subproblem, as shown by the outcomes of Section 4.2.

The robustness of the Schur complement approach is tempered by its computational expense for two-dimensional Stefan problems. The results from Section 4.3 corroborated the leading-order scaling of computational effort presented in Section 3.3. Using these estimates, we can make several assertions about the relative computational cost of each method. Fig. 3 showed that both iterative solution methods, M2 and M3, took more time to solve a Newton step than M1 for the problem sizes considered here. However, eventually the M1 curve would overtake the M2 and M3 curves as the problem size is increased, provided the needed numbers of Krylov vectors do not increase at an appreciable rate. However, our scaling results also indicate that the Schur complement method will likely always be more expensive than the iterative global strategy, M2, due to the nested solution loops, for a two-dimensional problem.

Based on our results, the optimal solution strategy for a three-dimensional Stefan problem (i.e., a problem with three-dimensional field variables and a two-dimensional free boundary) is posited to be quite different than that for the two-dimensional problem. The direct solution method, M1, pays a steep penalty, since the total number of equations grows as $N_t = N_n^3$ and the anticipated bandwidth of the Jacobian matrix increases to be proportional to N_n^2 rather than simply N_n in the two-dimensional problem. The computational effort scaling result is then

$$W_{M1,3D} \propto N_t (N_n^2)^2 = N_n^7 \tag{40}$$

The solution strategies that employ linear iterative solvers are less strongly affected; the effort for both changes as the total degrees of freedom change with the number of nodes, $N_t = N_n^3$, to yield

$$W_{M2,3D} \propto N_t k_J^2 = N_n^3 k_J^2 \tag{41}$$

$$W_{M3,3D} \propto N_t k_S k_A^2 = N_n^3 k_S k_A^2 \tag{42}$$

Clearly, as long as the needed number of Krylov vectors k_J and k_A scale with problem size in a manner no worse than $(k_J, k_A) \propto N_n^2 = N_t^{2/3}$, schemes M2 and M3 will be less expensive for the solution of large, three-dimensional problems. Indeed, our experience is that the size of the Krylov space needed for convergence is typically proportional to $N_t^{1/2}$ or less [41] and that, when convergent, iterative techniques provide an immense saving for the solution of three-dimensional problems. Note that the factor k_S for scaling of effort in the Schur approach is rarely expected to be important, since the size of the interface subproblem will always be substantially smaller than that associated with the field variables. If, as expected, the Schur complement formulation proves more robust than M2, the $k_A^2 < k_J^2$ savings in effort realized from $k_A < k_J$ will likely more than overcome the factor of k_S in effort for M3.

Both approaches employing linear iterative solvers for Newton steps can successfully solve the two-dimensional Stefan problem considered here, and both offer clear computational savings for larger problems, especially three-dimensional ones. However, the Schur complement approach promises increased robustness—a successful solution could be computed using significantly fewer Krylov vectors. We believe that this feature of the Schur approach will make it even more valuable for the solution of three-dimensional problems, since, in our experience, often one cannot afford to compute a large enough Krylov space for the convergence of these problems. This may mean the difference between success or failure for a large-scale computation.

We believe the success of this approach is attributable to the Schur complement formulation that maps the computation of the free boundary into a smaller subproblem, whose challenges can be more directly and independently addressed. Furthermore, the larger problem that is left behind involves a matrix block that is likely more amenable to iterative solution than the full Jacobian matrix of the unmapped problem. Namely, solving a linear system with matrix **A** is likely to be easier than solving the linear system with matrix **J**. In addition, appropriate preconditioning of **A** will undoubtedly yield further benefits for the solution methods considered here. We believe that the Schur complement formulation could prove to be an important tool for solving three-dimensional Stefan problems in a reliable and efficient manner.

Acknowledgments

This work has been supported in part by the Minnesota Supercomputing Institute and by the Department of Energy, National Nuclear Security Administration, under Award Numbers DE-FG52-06NA27498 and DE-FG52-08NA28768, the content of which does not necessarily reflect the position or policy of the United States Government, and no official endorsement should be inferred. The authors would also like to thank an anonymous reviewer for several interesting observations on this approach and its further development.

References

[1] J. Stefan, Über die theorie der eisbildung, insbesondereüber die eisbildung im polarmeere, Annalen der Physik und Chemie 42 (1891) 269–286.
[2] J.J. Derby, Crystal growth: crystal growth, bulk (theory and models), in: F. Bassani, J. Liedl, P. Wyder (Eds.), Encyclopedia of Condensed Matter Physics, Academic Press, Amsterdam, 2005, pp. 274–282.
[3] R.A. Brown, Theory of transport processes in single crystal growth from the melt, AIChE J. 34 (1988) 881.
[4] C. Beckermann, C.Y. Wang, Multi-phase/-scale modeling of transport phenomena in alloy solidification, in: C. Tien (Ed.), Annual Review of Heat Transfer Annual Review of Heat VI, Begell House, New York, NY, 1995, pp. 115–198.
[5] W. Shyy, H. Udaykumar, M. Rao, R. Smith, Computational Fluid Dynamics with Moving Boundaries, Taylor& Francis, Washington, DC, 1996.
[6] K.-O. Yu (Ed.), Modeling for Casting and Solidification Processing, CRC Press, 2001.
[7] J.J. Derby, A. Yeckel, Modeling of crystal growth processes, in: G. Müller, J.-J. Métois, P. Rudolph (Eds.), Crystal Growth—From Fundamentals to Technology, Elsevier, Amsterdam, 2004, pp. 143–167.
[8] C. Lan, Recent progress of crystal growth modeling and growth control, Chem. Eng. Sci. 59 (2004) 1437–1457.
[9] A. Yeckel, J.J. Derby, Computer modelling of bulk crystal growth, in: P. Capper (Ed.), Bulk Crystal Growth of Electronic, Optical, and Optoelectronic Materials, John Wiley& Sons, West Sussex, UK, 2005, pp. 73–119.
[10] H. Emmerich, The Diffuse Interface Approach in Materials Science: Thermodynamic Concepts and Applications of Phase-Field Models, Springer-Verlag, Heidelberg, 2003.
[11] V. Voller, A. Brent, C. Prakash, The modeling of heat, mass and solute transport in solidification systems, Int. J. Heat Mass Transfer 32 (1989) 1719.
[12] J. Sethian, P. Smereka, Level set methods for fluid interfaces, Annu. Rev. Fluid Mech. 35 (2003) 341.
[13] J. Glimm, J. Grove, X. Li, K. Shyue, Y. Zeng, Q. Zhang, Three-dimensional front tracking, SIAM J. Sci. Comput. 19 (1998) 703.
[14] M. Bogdanov, S. Demina, S.Y. Karpov, A. Kulik, M. Ramm, Y.N. Makarov, Volume of fluid (VOF) method for the dynamics of free boundaries, J. Comput. Phys. 39 (1981) 201.
[15] H. Udaykumar, R. Mittal, W. Shyy, Computation of solid-liquid phase fronts in the sharp interface limit on fixed grids, J. Comput. Phys. 153 (1999) 535.
[16] J. Chessa, P. Smolinski, T. Belytschko, The extended finite element method (XFEM) for solidification problems, Int. J. Numer. Meth. Eng. 53 (2002) 1959–1977.
[17] H. Ettouney, R. Brown, Finite-element methods for steady solidification problems, J. Comput. Phys. 49 (1983) 118.
[18] M. Benzi, G. Golub, J. Liesen, Numerical solution of saddle point problems, Acta Numerica 14 (2005) 1–137.
[19] A. Yeckel, G. Compère, A. Pandy, J.J. Derby, Three-dimensional imperfections in a model vertical Bridgman growth system for cadmium zinc telluride, J. Cryst. Growth 263 (2004) 629–644.
[20] C.W. Lan, M.C. Liang, Multigrid methods for incompressible heat flow problems with an unknown interface, J. Comput. Phys. 152 (1999) 55–77.
[21] A. Chatterjee, V. Prasad, A full 3-dimensional adaptive finite volume scheme for transport and phase-change processes, Part I: formulation and validation, Numer. Heat Transfer, Part A 37 (2000) 801–821.
[22] O. Weinstein, S. Brandon, Dynamics of partially faceted melt-crystal interfaces III: three-dimensional computational approach and calculations, J. Cryst. Growth 284 (2005) 235–253.
[23] V. Kumar, F. Durst, S. Ray, Modeling moving-boundary problems of solidification and melting adopting an arbitrary Lagrangian–Eulerian approach, Numer. Heat Transfer, Part B 49 (2006) 299–331.
[24] C.W. Lan, C.J. Chen, Dynamic three-dimensional simulation of facet formation and segregation in Bridgman crystal growth, J. Cryst. Growth 303 (2007) 287–296.
[25] J.J. Derby, R.A. Brown, A fully implicit method for simulation of the one-dimensional solidification of a binary alloy, Chem. Eng. Sci. 41 (1986) 37–46.
[26] A. Yeckel, R.T. Goodwin, Cats2D (Crystallization and Transport Simulator), User Manual, unpublished, 2003. Available at <http://www.msi.umn.edu/~yeckel/cats2d.html>.
[27] J. Thompson, Z. Warsi, C.W. Mastin, Boundary-fitted coordinate system for numerical solution of partial differential equations – A review, J. Comput. Phys. 47 (1982) 1–108.
[28] K.N. Christodoulou, L.E. Scriven, Discretization of free surface flows and other moving boundary problems, J. Comput. Phys. 98 (1992) 1–17.
[29] K.N. Christodoulou, S.F. Kistler, P.R. Schunk, Advances in computational methods, in: P.M. Schweizer, S.F. Kistler (Eds.), Liquid Film Coating: Scientific Principles and Their Technological Implications, Chapman& Hall, New York, NY, 1996 (Chapter 9).
[30] D. Lynch, K. O'Neill, Continuously deforming finite elements for the solution of parabolic problems, with and without phase change, Int. J. Numer. Methods Eng. 17 (1981) 81–96.
[31] P. Sackinger, P. Schunk, R. Rao, A Newton–Raphson pseudo-solid domain mapping technique for free and moving boundary problems: a finite element implementation, J. Comput. Phys. 125 (1996) 83–103.
[32] R. Cairncross, P. Schunk, T. Baer, R. Rao, P. Sackinger, A finite element method for free-surface flows of incompressible fluids in three dimensions, Part i: Boundary-fitted mesh motion, Int. J. Numer. Meth. Fluids 33 (2000) 375–403.
[33] S. Kistler, L. Scriven, Coating flows, in: J. Pearson, S. Richardson (Eds.), Computational Analysis of Polymer Processing, Applied Science Publishers, London, 1983, pp. 243–299.
[34] S. Kistler, L. Scriven, Coating flow theory by finite-element and asymptotic analysis of the Navier–Stokes system, Int. J. Numer. Meth. Fluids 4 (1984) 207–229.
[35] L. Ungar, N. Ramprasad, R. Brown, Finite element methods for unsteady solidification problems arising in prediction of morphological structure, J. Sci. Comput. 3 (1988) 77–108.
[36] H.B. Keller, Numerical solution of bifurcation and nonlinear eigenvalue problems, in: Applications of Bifurcation Theory, Academic Press, New York, 1997, pp. 159–384.
[37] P.D. Thomas, R.A. Brown, LU decomposition of matrices with augmented dense constraints, Int. J. Numer. Meth. Eng. 24 (1987) 1451–1459.
[38] B.A. Finlayson, Numerical Methods for Problems with Moving Fronts, Ravenna Park Publishing, Seattle, 1992.
[39] S. Kuppurao, S. Brandon, J.J. Derby, Modeling the vertical Bridgman growth of cadmium zinc telluride I. Quasi-steady analysis of heat transfer and convection, J. Cryst. Growth 155 (1995) 93.
[40] Y. Saad, M.H. Schultz, Gmres: a generalized minimal residual algorithm for solving nonsymmetric linear systems, SIAM J. Sci. Stat. Comput. 7 (1986) 856–869.
[41] L.S. Lun, Modeling and control of cadmium zinc telluride grown via an electro-dynamic gradient freeze furnace, Ph.D. Thesis, University of Minnesota, 2007. Available through University Microfilms, Inc. <http://www.umi.com>.
[42] R. James, T. Schlesinger, J. Lund, M. Schieber, in: T. Schlesinger, R. James (Eds.), Semiconductors for Room Temperature Nuclear Detector Applications, vol. 43, Academic Press, San Diego, 1995, p. 335.
[43] R. James, B. Brunett, J. Heffelfinger, J.V. Scyoc, J. Lund, F. Doty, C. Lingren, R. Olsen, E. Cross, H. Hermon, H. Yoon, N. Hilton, M. Schieber, E. Lee, J. Toney, T. Schlesinger, M. Goorsky, W. Yao, H. Chen, A. Burger, Material properties of large-volume cadmium zinc telluride crystals and their relationship to nuclear detector performance, J. Electron. Mater. 27 (1998) 788.
[44] R. Arlt, V. Ivanov, K. Parnham, Advantages and use of CdZnTe detectors in safeguards measurements, in: Proceedings of the MPA& C Conference, Obninsk, Russia, 2000.
[45] L. Lun, A. Yeckel, P. Daoutidis, J.J. Derby, Decreasing lateral segregation in cadmium zinc telluride via ampoule tilting during vertical Bridgman growth, J. Cryst. Growth 291 (2006) 348–357.