Journal of Computational Physics 229 (2010) 2009–2033

Contents lists available at ScienceDirect

Journal of Computational Physics

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

High-order unconditionally stable FC-AD solvers for general smooth domains I. Basic elements

Oscar P. Bruno [a,*], Mark Lyon [b]

[a] California Institute of Technology, Applied and Computational Mathematics, MC 217-50, 1200 East California Blvd., CA 91125, United States
[b] University of New Hampshire, Department of Mathematics, Kingsbury Hall W348, NH 03824, United States

ARTICLE INFO

Article history:
Received 17 April 2009
Received in revised form 25 October 2009
Accepted 11 November 2009
Available online 18 November 2009

Keywords:
Spectral method
Complex geometry
Unconditional stability
Fourier series
Fourier continuation
ADI
Partial Differential Equation
Numerical method

ABSTRACT

We introduce a new methodology for the numerical solution of Partial Differential Equations in general spatial domains: our algorithms are based on the use of the well-known Alternating Direction Implicit (ADI) approach in conjunction with a certain "Fourier continuation" (FC) method for the resolution of the Gibbs phenomenon. Unlike previous alternating direction methods of order higher than one, which can only deliver unconditional stability for rectangular domains, the present high-order algorithms possess the desirable property of *unconditional stability for general domains*; the computational time required by our algorithms to advance a solution by one time-step, in turn, grows in an essentially linear manner with the number of spatial discretization points used. In this paper we demonstrate the FC-AD methodology through a variety of examples concerning the Heat and Laplace Equations in two and three-dimensional domains with smooth boundaries. Applications of the FC-AD methodology to Hyperbolic PDEs together with a theoretical discussion of the method will be put forth in a subsequent contribution. The numerical examples presented in this text demonstrate the unconditional stability and high-order convergence of the proposed algorithms, as well the very significant improvements they can provide (in one of our examples we demonstrate a one thousand improvement factor) over the computing times required by some of the most efficient alternative general-domain solvers.

1. Introduction

We present a new methodology for the numerical solution of Partial Differential Equations (PDEs) in general spatial domains. Our approach is based on use of the well-known Alternating Direction Implicit (ADI) methodology introduced in Refs. [1–5] in conjunction with a certain "Fourier continuation" method [6–8] for the resolution of the Gibbs phenomenon. Alternating direction algorithms can yield *unconditional stability* at approximately the same cost per time-step as explicit (conditionally stable) finite-difference formulations, and have thus been pursued aggressively over the last half century. The application of alternating direction methods has been hindered by a significant limitation however: previous alternating direction approaches could not be directly applied to PDEs on arbitrary (non-rectangular) domains without reducing the truncation error near the boundary to first-order [9, p. 77]. The Fourier-Continuation Alternating-Direction (FC-AD) methodology introduced in this paper, in contrast, can produce high-order accuracies with unconditionally stable numerics for general geometries in essentially linear time—of the order of a spatial Fast Fourier Transform per time-step. In this paper we introduce the FC-AD methodology for the Heat and Laplace Equations in two- and three-dimensional domains with smooth

* Corresponding author. Tel.: +1 626 395 4548; fax: +1 626 578 0124.
E-mail address: bruno@acm.caltech.edu (O.P. Bruno).

0021-9991/$ - see front matter © 2009 Elsevier Inc. All rights reserved.
doi:10.1016/j.jcp.2009.11.020

boundaries; applications of the FC-AD methodology to Hyperbolic PDEs together with a theoretical discussion of the approach will be presented in [10]. Further extensions of our methods to more general problems, including equations containing mixed derivatives, nonlinear terms, etc., have been successfully demonstrated, and will be put forth in subsequent contributions; extensions to domains with non-smooth boundaries are currently being investigated. A variety of numerical examples presented in this text demonstrate the unconditional stability and high-order convergence of the proposed algorithm, as well the very significant improvements it can provide (in one of our examples we demonstrate a one thousand improvement factor) over the computing times required by some of the most efficient alternative general-domain solvers.

Over the more than 50 years since the introduction of the finite-difference-based ADI algorithm for the Heat and Laplace Equations, many variants of this approach have been put forward (c.f. [11] and references therein), including methods for solution of a variety of linear and nonlinear PDEs [12–21], as well as methods of high-order of spatial and temporal accuracy [22–26]. As suggested above, previous unconditionally stable alternating-direction methods can only achieve high-order accuracy in presence of a formulation of the given PDE on domains given by the union of a finite number of rectangular regions. The few unconditionally stable high-order ADI algorithms that have been applied to non-rectangular geometries [27–29] rely upon domain mappings that translate the given problem into one posed on a rectangular geometry, to which a high-order version of the algorithm is applicable. Unfortunately, however, the construction of such domain mappings is prohibitively laborious for most engineering and scientific applications. To the authors' knowledge, unconditionally stable high-order alternating direction algorithms for general domains without some form of domain mapping had not been produced before the present work.

(It is interesting to note that, even in the context of classical finite difference PDE solvers, *in which the alternating-direction methodology is not used*, the treatment of boundaries and boundary conditions with high-order accuracy entails significant difficulties—even for simple rectangular domains. Recent advances in these regards are reported in [30–32]: these methods produce high-order finite-difference PDE solvers on the basis of a certain "simultaneous approximation term" (SAT) for enforcement of boundary conditions. Development of *alternating direction* schemes using high-order finite-difference discretizations *for rectangular domains*, in turn, is currently an active area of research; see e.g. [25,26,33,34]. A great deal of activity currently focuses in the area of embedded-boundary (EB) methods [35–46]. Like the schemes introduced in the present work, EB methods rely on Cartesian grids to obtain solutions for non-rectangular domains. Most EB methods are explicit and of first or second-order of spatial accuracy; typically EB formulations of higher-order of accuracy have not proven reliably stable for non-rectangular domains. A Fourier-based EB method presented in [47] for the Poisson equation has demonstrated some accuracy for a slowly-oscillatory sinusoidal solution on a smooth four-leaf shaped geometry.)

Alternating direction methodologies have also been previously used in conjunction with spatial differentiation methods which, like the one used introduced in this work, do not rely on finite differencing. In particular, in [16,48–50] an alternating-direction approach was proposed that uses a Fourier basis for differentiation. Applications of previous Fourier-based techniques have thus far been restricted to rectangular geometries in spite of efforts seeking generalization to general domains [16,48]. (In fact, the previous Fourier approaches exhibit even less flexibility than their finite-difference counterparts: the latter are applicable to general domains, albeit with first-order accuracy.) Spatial spline collocation methods have also been introduced in this context, see [51–53] and references therein. In particular, cubic-spline interpolation was used [53] in conjunction with an alternating direction embedding scheme to solve elliptic PDEs for non-rectangular geometries, but only conditional stability was obtained (for the Heat-Equation-like discrete scheme whose steady state provides the solution of the given elliptic equation).

Chebyshev methods [50,54], like all methods that rely on use of unevenly spaced discretizations, do not provide a consistent basis for alternating direction methods on complex geometries—unless, as discussed above, domain mappings into rectangular domains are used. A significant decrease of the complexity of the required mappings can be obtained via use of spectral-element methods [55]. Yet construction of adequate three-dimensional finite element meshes, existence of restrictive stability conditions (especially for higher-order spectral methods), and the appearance of considerable pollution error for the lower-order spectral methods often considered, remain significant unresolved issues under investigation in this context.

Our use of the Fourier basis, which relies on Fourier approximation of non-periodic functions, requires resolution of a classical problem in numerical analysis: the Gibbs phenomenon. A variety of methods have been developed to reduce or eliminate the Gibbs phenomenon: see [54,56–58] and references therein. Briefly, the approach used in this paper for the resolution of the Gibbs phenomenon, which we refer to as the FC(Gram) algorithm, is based on a "continuation method" [6–8] which produces the Fourier series of a smooth and periodic extension of a given smooth function. The FC(Gram) continuation strategy we introduce for use in conjunction with our alternating direction algorithm, which differs significantly from previous continuation approaches and which can be computed with FFT speed, is presented in detail in Section 2.

For a given d-dimensional domain $\Omega, d \geqslant 2$, our FC-AD algorithm uses a Cartesian grid given by the intersection of Ω with a Cartesian grid G in all space, as shown in Fig. 1. The PDE is discretized in time and then split into sets of uncoupled spatial ODEs by means of an alternating direction technique. To complete the time-stepping algorithm, each one of the resulting spatial ODEs is then solved with high-order accuracy on the corresponding Cartesian line, call it L, by means of the FC(Gram) continuation method and one-dimensional grids in L as depicted in Fig. 2. (The one-dimensional grid in L equals the union of $L \cap G \cap \Omega$ and the set of boundary points $L \cap \partial\Omega$, which are shown as $\{x_\ell, x_r\}$ in Fig. 2). Since the computational cost required for the FC(Gram) solution of each one of these spatial ODEs is proportional to that of a one-dimensional FFT, the overall cost of a full FC-AD time-step is of the order of $\mathcal{O}(N \log(N))$ operations, where N denotes the size of the full d-dimensional spatial

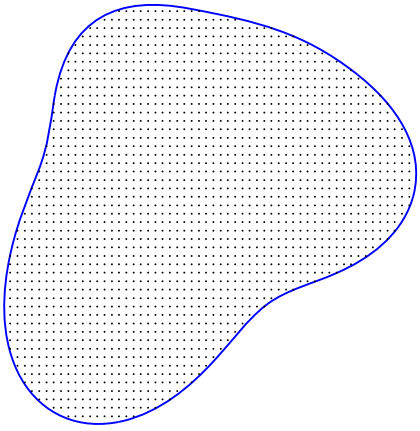

Fig. 1. discretization $\mathcal{D}_\Omega = G \cap \Omega$ of a non-rectangular open domain Ω.

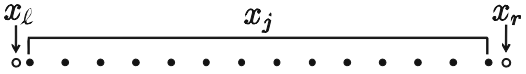

Fig. 2. One-dimensional discretization grid $(L \cap G \cap \Omega) \cup (L \cap \partial\Omega)$ on a typical discretization line L parallel to the x axis; similar discretizations are used on all discretization lines L parallel to each one of the Cartesian coordinate axes. Note the boundary points $L \cap \partial\Omega = \{x_\ell, x_r\}$ that generically do not lie on the regular one-dimensional Cartesian grid. Note that for any geometry (convex or non-convex) that can be realistically resolved by the underlying Cartesian mesh of the FC-AD, the needed end points of each interval can be easily found by using robust root finding techniques in an automated way.

grid. To achieve this reduced computational cost the FC-AD method uses "adequate" FFT implementations, which evaluate an FFT of size n in $\mathcal{O}(n\log(n))$ operations for any integer n, including n given by products of a small number of prime numbers, or even prime values of n. Such FFT algorithms are available, for example, as part of the FFTW library [59]. FFTW is used in all of the implementations demonstrated in this paper; the resulting $\mathcal{O}(N\log(N))$ cost of the FC-AD algorithm is clearly demonstrated by the numerical examples presented in Section 6. Reductions in the $\mathcal{O}(N\log(N))$ proportionality constant could conceivably be obtained by slight modifications of the FC-AD method that avoid evaluation of FFTs of sizes n containing large prime numbers; such modifications have not been pursued as yet.

In view of its high-order accuracy, unconditional stability and accurate handling of boundary conditions for general domains, the FC-AD algorithm offers significant performance advantages over other methods. In particular, its unconditional stability allows for use of significantly larger time-steps than those required by conditionally stable methods: this characteristic is of the highest importance in the context of the diffusive equations considered in this paper, whose stringent explicit-solver CFL conditions require use of extremely small time steps. The FC-AD's high-order accuracy and accurate handling of boundary conditions, on the other hand, generally leads to accurate results with much coarser discretizations that otherwise are necessary. Perhaps the most notable aspect of the proposed algorithm, finally, is that it displays these desirable qualities for PDEs posed in general domains. As a result of these qualities, the method can exhibit excellent performance: above we mentioned, for example, an improvement by a factor of one thousand in the solution of a diffusion equation over that obtained from a previous high-quality solver.

Our paper is organized as follows: after introducing the FC(Gram) Fourier continuation algorithm in Section 2, in Section 3 we describe the alternating direction operator-splitting we use for the Heat Equation (Sections 3.1 and 3.2) and we introduce an efficient iteration strategy leading to a fast Poisson solver (Section 3.3). A brief summary of the overall FC-AD Heat and Poisson solvers, leading into Sections 4 and 5, is provided in Section 3.4. In Section 4 we then present an algorithm, FC-ODE, for the solution of the ODEs that result from the alternating direction splittings mentioned above; a number of boundary projection techniques introduced in this section ensure the unconditional stability of the FC-AD method on general geometries. The overall prescriptions for the FC-AD Heat and Poisson solvers are then detailed in Section 5. A variety of numerical results and comparisons presented in Section 6, finally, demonstrate the properties of our approach: unconditional stability, accuracy, speed, and memory efficiency.

2. The FC(Gram) algorithm

2.1. Background

Consider a smooth function $f \in C^k[x_\ell, x_r]$ for some positive integer k or $k = \infty$, and assume approximate values of the function f are given on a discrete grid $x_j, j = 1, \ldots, n$ within $[x_\ell, x_r]$, for which the relations $x_1 = x_\ell$ and $x_n = x_r$ may or may not be

satisfied. With reference to Fig. 2 we mention that in our application, the FC-AD algorithm to be introduced in Section 3, the FC(Gram) continuation method is applied to the restriction of f to the domain $[x_1, x_n]$ from the slightly larger interval $[x_\ell, x_r]$. Naturally, this restriction operation has an effect on the overall error and error bounds for the FC approximation of f in the interval $[x_\ell, x_r]$. For clarity but without loss of generality, within Section 2 we replace the interval $[x_1, x_n]$ by the unit interval $[0, 1]$ and we use the discrete grid

$$x_j = (j-1)h, \quad j = 1, \ldots, n, \quad h = 1/(n-1), \tag{1}$$

the results of this section apply, via simple transformations, to the case of a general interval $[x_1, x_n]$ and associated (slightly larger) intervals $[x_\ell, x_r]$.

The Gibbs phenomenon is the well-known ringing effect and associated inaccurate approximations that occur as a non-periodic function defined in the interval $[0, 1]$ is expanded in a Fourier series of period 1. Preceding the present work, a certain continuation method [6–8] was introduced for the resolution of the Gibbs phenomenon, which relies on the construction of a periodic function in a domain significantly larger than the domain $[x_1, x_n] = [0, 1]$ containing the discretization points $x_1, \ldots, x_n$. Following [7], we could define a b-periodic continued function $f^c(x), b > 1$, given by a series of the form

$$f^c(x) = \sum_{k \in t(F)} a_k e^{\frac{2\pi i}{b} k x}, \tag{2}$$

where $t(F) = \{k \in \mathbb{N} : -F/2 + 1 \leqslant k \leqslant F/2\}$ for F even and $t(F) = \{k \in \mathbb{N} : -(F-1)/2 \leqslant k \leqslant (F-1)/2\}$ for F odd. The coefficients a_k would then be obtained (as in [6,8,7]) by solving the least-squares system of equations

$$\min_{\{a_k : k \in t(F)\}} \sum_{j=0}^{n-1} \left| \sum_{k \in t(F)} a_k e^{\frac{2\pi i}{b} k x_j} - f(x_j) \right|^2. \tag{3}$$

Numerical results [7] demonstrate that it is advantageous to use Singular Value Decompositions (SVD) to produce the least-squares solutions. In this text we call FC(SVD) the resulting SVD-based method of Fourier continuation, to distinguish it from the FC(Gram) continuation method introduced in what follows. Typical results produced by FC(SVD) are shown in Fig. 3. In the reminder of this section we put forward a new, more complex prescription for the evaluation of continuation functions f^c which, owing to speed and stability considerations (see point 5 and Remark 2.3 below in this section with regards to the stability issue), is appropriate for use in conjunction with the alternating-direction time-stepping strategy.

Although adequate for applications such as high-order surface representations [7], the $O(n^3)$ computational cost of the FC(SVD) is significantly higher than is desirable for use as an element of a PDE solver. In the one-dimensional context relevant to the present work, however, it is possible to use a significantly faster continuation algorithm. In order to introduce this new method (which, we call FC(Gram) in view of its use of Gram Polynomials), we consider a pair of functions: the given function $f(x)$, defined in the interval $[0, 1]$, and the translated function $f(x - d - 1)$, which is defined in the interval $[1 + d, 2 + d]$, see Fig. 4. As indicated in the figure, we consider a small portion on the right-end of the graph of $f(x)$ as well as a small portion on the left-end of the graph of $f(x - d - 1)$, say, the portions defined in the intervals $[1 - \Delta, 1]$ and $[1 + d, 1 + d + \Delta]$. The FC(SVD) algorithm can be applied in these two lines segments to produce a periodic function f_{match}, with periodicity interval $[1 - \Delta, 1 + 2d + \Delta]$, which simultaneously approximates $f(x)$ on the interval $[1 - \Delta, 1]$ and $f(x - d - 1)$ on the interval $[1 + d, 1 + d + \Delta]$; in view of the discussion [7] we know such an approximation is spectrally

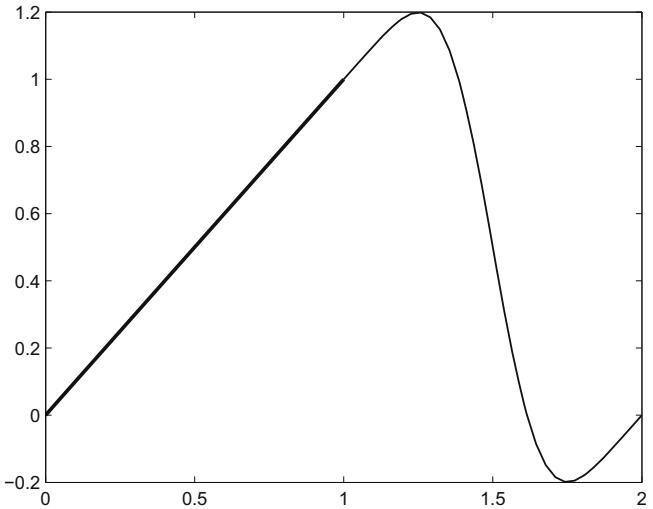

Fig. 3. Smooth periodic function resulting from the FC(SVD) algorithm applied to the function $f(x) = x$ over the unit interval.

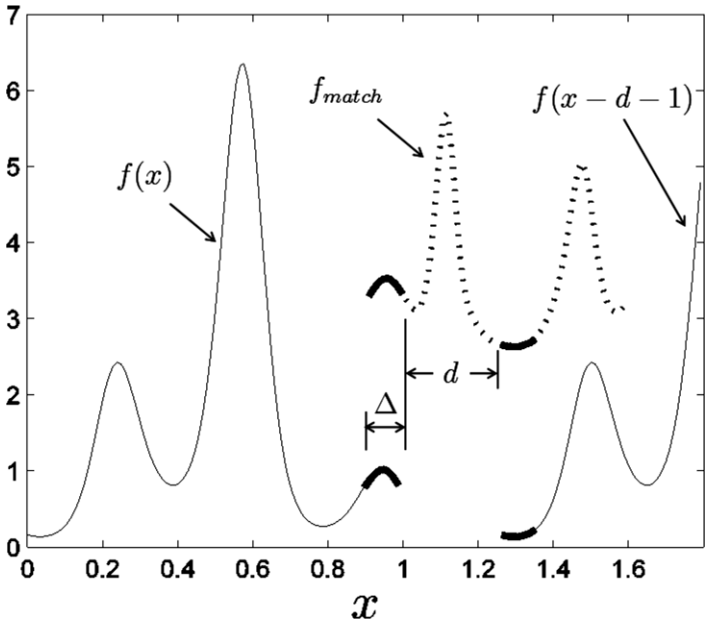

Fig. 4. Calculation of a periodic extension of $f(x) = e^{\sin(5.4\pi x - 2.7\pi) - \cos(2\pi x)}$ using only a small subset of function values ($n_\Delta = 10$). Raised for visibility, the function $f_{\text{match}}(x)$ is displayed in the upper-right portion of the figure.

accurate. The function f_{match}, which is shown in the upper right portion of Fig. 4 (displaced for visibility), can be used to produce a smooth transition between the right-end of $f(x)$ to the left-end of $f(x-d-1)$. Denoting by $C^\infty_{\text{per}}[c,d]$ the space of infinitely differentiable functions defined over the real line with periodicity length $d-c$, we clearly have $f_{\text{match}} \in C^\infty_{\text{per}}[1-\Delta, 1+2d+\Delta]$.

We note that the prescription

$$f^{\text{de}}(x) = \begin{cases} f(x) & \text{for } x \in [0,1] \\ f_{\text{match}}(x) & \text{for } x \in (1, 1+d]\,, \\ f^{\text{de}}(x+1+d) = f^{\text{de}}(x) & \text{for all } x \text{ in } \mathbb{R} \end{cases} \tag{4}$$

defines a $(1+d)$-periodic function that, owing to approximation errors arising in the FC(SVD) is, in fact, discontinuous (thus the acronym "de" which stands for "discontinuous extension"), but which equals a smooth function up to the—small—error arising from the FC(SVD) continuation. A spectrally accurate Fourier continuation for f can be obtained as a discrete Fourier transform of the grid values of the (Δ-dependent) function f^{de}. Naturally we need $1+d$ to be an integer multiple of h. We also take Δ to be an integer multiple of h. For appropriate positive integers N_Δ and N_d, defining

$$\Delta = (N_\Delta - 1)h \quad \text{and} \quad d = (N_d - 1)h, \tag{5}$$

sampling the function f^{de} at the $n + N_d - 2$ evenly spaced points

$$x_j = (j-1)h, \quad j = 1, \ldots, n + N_d - 2, \tag{6}$$

and then evaluating the discrete Fourier transform of the resulting discrete f^{de} values produces, with FFT speed, a high-order accurate Fourier continuation of the function f.

Out of the $n + N_d - 2$ points used in the interval $[0, 1+d]$, the points $x_j, j = n - N_\Delta + 1, \ldots, n$ fall within the set $[1-\Delta, 1]$ and the points $x_j + 1 + d, j = 1, \ldots, N_\Delta$ fall within the set $[1+d, 1+d+\Delta]$, for a total of $2N_\Delta$ points in the set $[1-\Delta, 1] \cup [1+d, 1+d+\Delta]$. Then, for a fixed ratio d/Δ, and taking N_Δ such that $N_\Delta \leqslant Cn^{1/3}$ for some constant C, the total number of operations required to construct the Fourier transform of f^{de} is of order $O((n+N_d)\log(n+N_d))$. Additionally, if N_Δ is chosen so that $N_\Delta \geqslant Cn^\gamma$ for some $\gamma > 0$ and some constant C, then, due to the spectral convergence of the FC(SVD) algorithm [7], spectral convergence results from the present FFT-speed version of the continuation algorithm as well.

Our overall FC(Gram) continuation method results as a slight but necessary variation of the algorithm just described: even for "reasonable" values of the number n of one-dimensional samples, the number $N_\Delta \sim O(n^{1/3})$ of discretization points contained in the boundary intervals, which was chosen sufficiently small as to maintain the FFT-type efficiency, is not large enough to give rise to optimally accurate SVD continuations. For reasonable values of n it is much more efficient to use small values of N_Δ and N_d and to precede the FC(SVD) calculation by a projection into a bases of orthogonal polynomials, the Gram bases for each one of the two intervals, as detailed below. One of the advantages of this procedure is that the values of each one of the polynomials in the orthogonal sets can easily be evaluated at large numbers of points in the corresponding

intervals $[1-\Delta, 1]$ and $[1+d, 1+d+\Delta]$, and, thus, highly accurate FC(SVD) continuation of these basis functions can be constructed; the separate continuations of the basis functions can then be combined to form the continuation of the original function. The evaluation of the FC(SVD) of the various polynomials in the basis can be precomputed and stored, with insignificant memory cost, for each possible (reasonable) choice of N_Δ, N_d and the parameters of the FC(SVD) continuation: obviously, the use of such (bounded cost and, in fact, extremely inexpensive) precomputation does not affect the $O(n\log(n))$ complexity of the FC(Gram) algorithm.

2.2. Precise description of the FC(Gram) algorithm

To provide a complete description of the FC(Gram) algorithm (points 1 through 6 below), we first introduce a number of notations and definitions. Letting Δ be a small n-dependent real number (that tends to zero as $n \to \infty$), we define the spaces

$$C^k[1-\Delta, 1] \quad \text{and} \quad C^k[1+d, 1+d+\Delta] \tag{7}$$

of smooth boundary sections and, for a function $f \in C^k[0,1]$, we call

$$\begin{aligned} f_{\text{left}}(x) &= f(x) && \text{for } x \in [1-\Delta, 1] \quad \left(f_{\text{left}} \in C^k[1-\Delta, 1]\right) \quad \text{and} \\ f_{\text{right}}(x) &= f(x-d-1) && \text{for } x \in [1+d, 1+d+\Delta] \quad \left(f_{\text{right}} \in C^k[1+d, 1+d+\Delta]\right) \end{aligned} \tag{8}$$

the "boundary sections" of f.

Further, we endow the spaces (7) with the semi-positive-definite discrete scalar products

$$\begin{aligned} (h,k)_{\text{left}} &= \sum\nolimits_{j \in S_{\text{left}}} h(x_j)k(x_j) \quad \text{and} \\ (h,k)_{\text{right}} &= \sum\nolimits_{j \in S_{\text{right}}} h(x_j+1+d)k(x_j+1+d), \end{aligned} \tag{9}$$

respectively, where the sets S_{left} and S_{right} are defined in what follows. Selecting an appropriate integer n_Δ such that $I = (N_\Delta - 1)/(n_\Delta - 1)$ is an integer (the "skipping parameter"; see Remark 10 for details on adequate choices of I and n_Δ), each one of the sums defining the scalar products (9) are performed over n_Δ values of j skipping groups of $I-1$ values: $S_{\text{right}} = \{1, I+1, 2I+1, \ldots, (n_\Delta-1)I+1\}$ and $S_{\text{left}} = \{n-(n_\Delta-1)I, n-(n_\Delta-2)I, n-(n_\Delta-3)I, \ldots, n\}$. Thus for $j \in S_{\text{right}}$ we have $x_j + 1 + d \in [1+d, 1+d+\Delta]$, and for $j \in S_{\text{left}}$ we have $x_j \in [1-\Delta, 1]$. In Fig. 5 the points $x_j \in [0, \Delta]$ for $j \in S_{\text{right}}$ (which correspond to the points $x_j + 1 + d$ that are actually used in the scalar product $(\cdot,\cdot)_{\text{right}}$) and the points $x_j \in [1-\Delta, 1]$ for $j \in S_{\text{left}}$ are shown as fully filled-in circles for the cases $I = 1$ and $I = 2$.

Remark 2.1. In the algorithm presented in points 1 through 6 below, the smooth boundary functions f_{left} and f_{right} are approximated with high-order accuracy by their orthogonal projections with respect to the scalar products (9), onto the subspaces of polynomials of degree m. In our examples we use projections onto the space of polynomials of degree $\leqslant m$ with values such as $m = 4$ and $m = 5$. These choices of the numerical values of m are dictated, in part, by *stability* considerations, as discussed in [10].

Remark 2.2. The skipping parameter I was introduced in order to insure convergence of the FC(Gram) approximation. In detail, a bound is presented in [10] for the approximation error that arises as the FC(Gram) algorithm is applied to differentiable functions $f \in C^k[0,1]$ with k large enough. That error bound indicates that, if fixed values of n_d and n_Δ are used, full convergence of the FC(Gram) algorithm to absolute zero error is obtained only if the numbers of discretization points contained in the Δ-sized end intervals increase, even if very slowly, as n grows: a growth of $N_\Delta \sim n^\gamma$ gives an overall convergence rate of $n^{-(m+1)(1-\gamma)}$ at a cost of $O(n\log(n))$ operations provided γ is small enough; per the discussion earlier in this section we

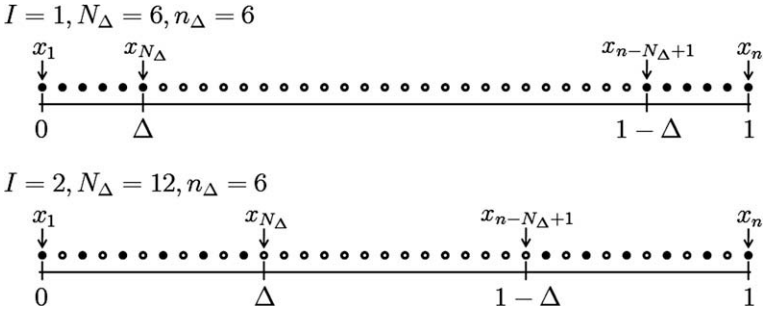

Fig. 5. Depiction of the points x_j for $j \in S_{\text{left}}$ (rightmost fully filled-in points in the figures) and $j \in S_{\text{right}}$ (leftmost fully filled-in points in the figures). Note that for $j \in S_{\text{right}}$, the points x_j, which lie in $[0, \Delta]$, correspond to the points $x_j + 1 + d \in [1+d, 1+d+\Delta]$ that are actually used in the scalar product $(\cdot,\cdot)_{\text{right}}$. The values of I, N_Δ, and n_Δ in the figure were selected for illustrative purposes. As indicated in Remark 10 below, for all numerical calculations presented in this paper the values $I = 1, N_\Delta = 10$, and $n_\Delta = 10$ were used.

must have $\gamma \leqslant 1/3$ to maintain the overall cost below an $O(n\log(n))$ bound. (Use of fixed values of n_d, n_Δ and d/Δ is advantageous in that it allows for use of a fixed set of Gram Polynomials and corresponding continuations; see items 1 and 3 below.) Note that, as N_Δ grows, N_d must also grow, to maintain the fixed ratio of d/Δ we assume in our algorithm. In practice, full machine precision accuracy levels were obtained in all of the examples we have ever considered by using $I = 1$ (no skipping) together with appropriate fixed values of $n_\Delta = N_\Delta$ and $n_d = N_d$. Thus, the parameter values we use throughout the remainder of this paper satisfy

$$I = 1, \quad \Delta = (n_\Delta - 1)h \quad \text{and} \quad d = (n_d - 1)h. \tag{10}$$

On the basis of a variety of numerical experiments, we have selected the parameter values $n_\Delta = 10$ and $n_d = 27$ (and, thus, $d/\Delta = 26/9$), which are used in all of the numerical experiments considered in this paper.

Using these notations and definitions, the FC(Gram) algorithm proceeds as follows:

1. For a given value of m (see Remark 2.1), an orthonormal basis $\mathcal{B}_{\text{left}} = \{P^r_{\text{left}}(x)\}_{r=0}^m$ of the space of polynomials of degree $\leqslant m$ with respect to the scalar product $(\cdot,\cdot)_{\text{left}}$ is obtained by applying the $(\cdot,\cdot)_{\text{left}}$-based Gram–Schmidt orthogonalization process to the set $\{1, x, x^2, \ldots, x^m\}$ in order of increasing degree. Analogously, an orthonormal basis $\mathcal{B}_{\text{right}} = \{P^r_{\text{right}}(x)\}_{r=0}^m$ of the space of polynomials of degree $\leqslant m$ with respect to the scalar product $(\cdot,\cdot)_{\text{right}}$ is obtained by applying the $(\cdot,\cdot)_{\text{right}}$-based Gram–Schmidt orthogonalization process to the set $\{1, x, x^2, \ldots, x^m\}$ in order of increasing degree. (Sets of polynomials that are orthogonal with respect to discrete scalar products such as those presented in Eq. (9) are called Gram Polynomial bases; see e.g. [60, p. 323 and 61].) To avoid loss of orthogonality, in our algorithms these bases are obtained by means of the modified Gram–Schmidt orthogonalization procedure with partial re-orthogonalization (see e.g. [62, p. 107 and 375]).
2. Given a function $f \in C^k[0,1]$, the coefficients

   $$a^r_{\text{left}} = \left(f_{\text{left}}, P^r_{\text{left}}\right)_{\text{left}} \quad \text{and} \quad a^r_{\text{right}} = \left(f_{\text{right}}, P^r_{\text{right}}\right)_{\text{right}} \tag{11}$$

   of the polynomial approximations (projections)

   $$f^p_{\text{left}}(x) = \sum_{r=0}^{m} a^r_{\text{left}} P^r_{\text{left}}(x) \quad \text{and} \quad f^p_{\text{right}}(x) = \sum_{r=0}^{m} a^r_{\text{right}}\, P^r_{\text{right}}(x) \tag{12}$$

   of the smooth boundary Section 8 are obtained.
3. Highly accurate precomputed FC(SVD) continuations $f^{P,Q} \in C^\infty_{\text{per}}[1-\Delta, 1+2d+\Delta]$ are used for certain pairs $\{P, Q\}$ of Gram Polynomials, where $P \in \mathcal{B}_{\text{left}}$ and $Q \in \mathcal{B}_{\text{right}}$, and where $f^{P,Q} \in C^\infty_{\text{per}}[1-\Delta, 1+2d+\Delta]$ is a Fourier continuation of both P and Q. Various types of polynomial pairings are admissible as are methods to effect their joint continuation; full details concerning our prescriptions in these regards are presented in Section 2.3. Here we note that, as indicated in that section, the method we use leads to certain continuation functions f^r_{even} and f^r_{odd}, $r = 0 \ldots m$, which can collectively be used to obtain a Fourier continuation of an arbitrary pair of projections $\{f^p_{\text{left}}, f^p_{\text{right}}\}$. Fig. 6 displays the functions f^r_{even} and f^r_{odd} for $r = 0, r = 1$ and $r = 2$.
4. In view of the prescriptions in Section 2.3, the function f_{match} in Eq. (4) is obtained as the following linear combination of the FC(SVD) continuations mentioned in point 3:

   $$f_{\text{match}}(x) = \sum_{r=0}^{m} \frac{a^r_{\text{left}} + a^r_{\text{right}}}{2} f^r_{\text{even}}(x) + \frac{a^r_{\text{left}} - a^r_{\text{right}}}{2} f^r_{\text{odd}}(x). \tag{13}$$

5. A "discontinuous-projection" function f^{dp} is constructed according to the following formula (which is used in part on the basis of stability considerations, see Remark 2.3 below):

   $$f^{\text{dp}}(x) = \begin{cases} f^p_{\text{right}}(x - 1 - d) & \text{for } x \in [0, \Delta] \\ f(x) & \text{for } x \in (\Delta, 1-\Delta) \\ f^p_{\text{left}}(x) & \text{for } x \in [1-\Delta, 1] \\ f_{\text{match}}(x) & \text{for } x \in (1, 1+d) \\ f^{\text{dp}}(x+1+d) = f^{\text{dp}}(x) & \text{for all } x \text{ in } \mathbb{R} \end{cases}. \tag{14}$$

6. The Fourier continuation f^c of f is obtained as the Fourier series of the form (2), with $F = n + n_d - 2$, that results by using an FFT on the set of function values $f^{\text{dp}}(x_j)$ with $j = 1 \ldots n + n_d - 2$ (see Eq. (6)). This completes the prescription of the FC(Gram) continuation procedure; clearly, the continuation function f^c is obtained at a total cost of $\mathcal{O}(n\log(n))$ operations.

Remark 2.3. The prescription given in Step 5 for f^{dp} was chosen to provide stability in the context of the FC-AD methodology (c.f., the stability analysis presented in [10]); otherwise the first three lines in the definition of f^{dp} could be substituted by a more natural prescription, namely "$f(x)$", without changing the degree of accuracy of the continuation approximations. Note, in particular, that $f^c(x_j)$ generally does not equal $f(x_j)$ for discretization points x_j near the interval boundaries. In the

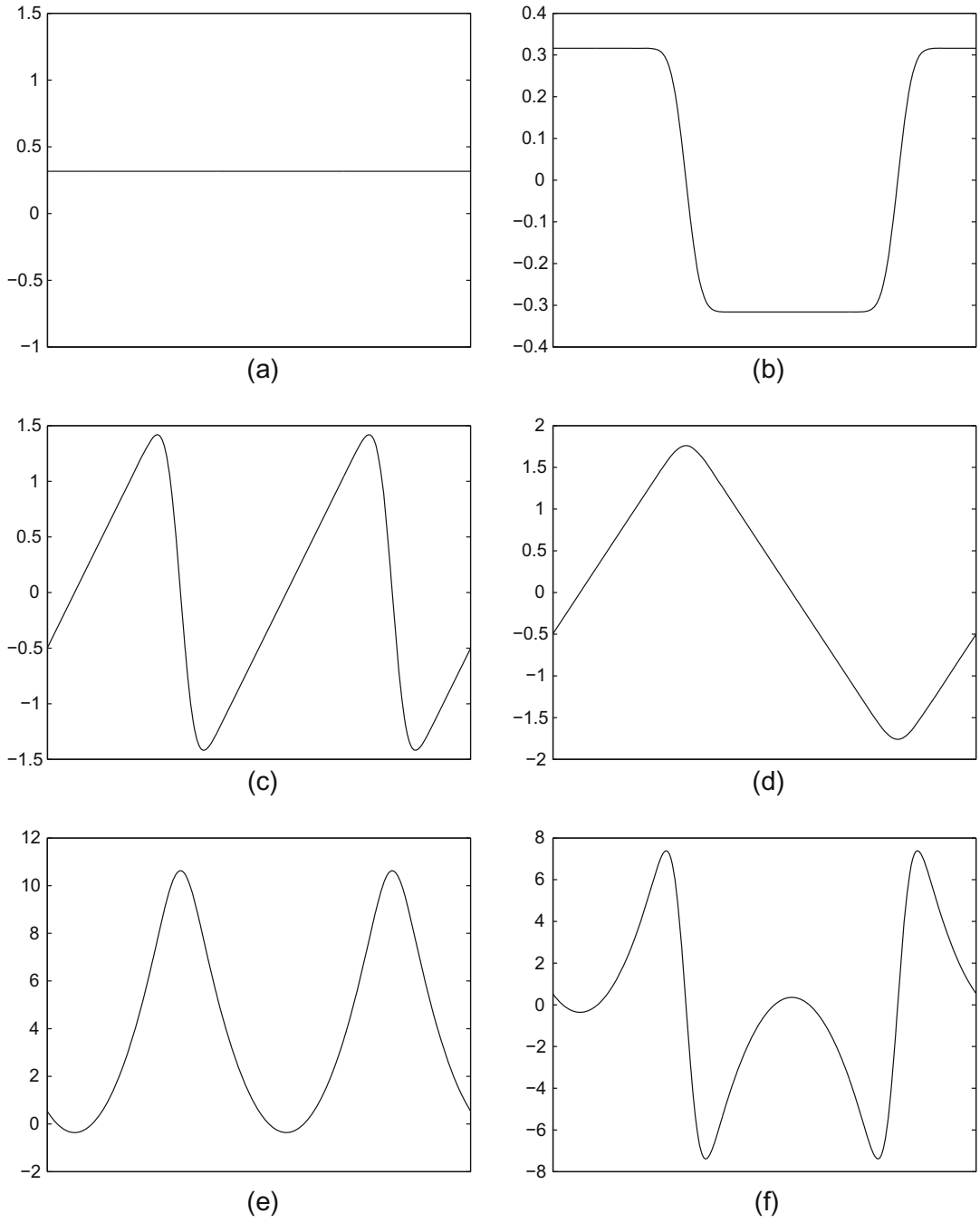

Fig. 6. Functions f^r_{even} (left column) and f^r_{odd} (right column) in the interval $[1-\Delta, 1+2d+\Delta]$ for $r=0$ (first row), $r=1$ (second row) and $r=2$ (third row). By construction, these functions are close approximations of the corresponding Gram polynomials P^r_{left} and $\pm P^r_{\text{right}}$ in the sub-intervals $[1-\Delta, 1]$ and $[1+d, 1+d+\Delta]$, respectively. Note that $f^r_{\text{even}}(x) = f^r_{\text{even}}(x+d+\Delta)$ and $f^r_{\text{odd}}(x) = -f^r_{\text{odd}}(x+d+\Delta)$ for all r.

normalized domain $[0,1]$ considered in this section, for instance, $f^c(x_j)$ will generally differ from $f(x_j)$ for x_j in the intervals $[0,\Delta]$ and $[1-\Delta, 1]$.

2.3. Useful Gram Polynomial pairs and accurate FC(SVD) continuations

Point 3 in the description of the FC(Gram) algorithm above does not detail either how to select pairs $\{P, Q\}$ of Gram Polynomials, $P \in C^k[1-\Delta, 1]$ and $Q \in C^k[1+d, 1+d+\Delta]$ for subsequent continuation via the FC(SVD) approach, or how the corresponding continuations are to be obtained: such details are presented in what follows.

Our choice of polynomial pairs is made in such a way as to require SVDs of the smallest possible dimensionality: with reference to point 1 above, we define the sets of "even" and "odd" polynomial pairs as the sets of all pairs $\{P, Q\}$ and $\{P, -Q\}$, respectively, where $P(x) = P^r_{\text{left}}(x)$ and $Q(x) = P^r_{\text{left}}(x - d - \Delta)$ and where $r \leqslant m$; note that, for each r, $P \in \mathcal{B}_{\text{left}}$ and $Q \in \mathcal{B}_{\text{right}}$—since $P^r_{\text{right}}(x) = P^r_{\text{left}}(x - d - \Delta)$. The even/odd nomenclature corresponds to the fact that for each even (resp. odd) pair, an FC(SVD) continuation can be produced, which will be denoted by f^r_{even} (resp. f^r_{odd}), that uses only even (resp. odd) Fourier coefficients; see Fig. 6. The set of all such continuations of these pairs of Gram Polynomials contains $2m + 2$ elements that can be used to construct the matching function f_{match} in Eq. (13).

Using a set of Υ equispaced matching points $\hat{x}_j$ in the interval $[1 - \Delta, 1]$ (satisfying $\hat{x}_1 = 1 - \Delta$ and $\hat{x}_\Upsilon = 1$), the functions f^r_{even} and f^r_{odd} are obtained as the SVD-based least-squares solution of the over-determined set of equations

$$f^r_{\text{even}}(\hat{x}_j) = \sum_{\substack{k \in t(g) \\ k \text{ even}}} \hat{a}^r_k e^{\frac{\pi i}{d+\Delta}\hat{x}_j k} \approx P^r_{\text{left}}(\hat{x}_j), \quad j = 1 \ldots \Upsilon, \tag{15}$$

$$f^r_{\text{odd}}(\hat{x}_j) = \sum_{\substack{k \in t(g) \\ k \text{ odd}}} \hat{b}^r_k e^{\frac{\pi i}{d+\Delta}\hat{x}_j k} \approx P^r_{\text{left}}(\hat{x}_j), \quad j = 1 \ldots \Upsilon \tag{16}$$

for the unknown Fourier coefficients $\hat{a}^r_k$ and $\hat{b}^r_k$ (see the text following Eq. (2) for the definition of $t(g)$). Note that, in view of the restrictions to k even and odd in Eqs. (15) and (16), respectively, we have $f^r_{\text{even}}(x) = f^r_{\text{even}}(x + d + \Delta)$ and $f^r_{\text{odd}}(x) = -f^r_{\text{odd}}(x + d + \Delta)$ for all r—and, thus, the functions f^r_{even} and f^r_{odd} are indeed joint continuations of $\{P^r_{\text{left}}, \pm P^r_{\text{right}}\}$, respectively. In particular, these functions result from application of a version of the FC(SVD) algorithm to pairs of Gram polynomials.

The explicit values we have used in this paper for the parameters g and Υ are presented in Remark 2.4 below. In particular, the number g of Fourier modes used is set to a value that does not exceed $\Upsilon/2$ (c.f. [7]), in order for the continuations of the Gram Polynomial-pairs to be (roughly) optimally accurate approximations of the corresponding polynomials; to achieve approximations of the desired accuracy, on the other hand, we select appropriately large values of Υ and correspondingly large values for g. Once the functions f^r_{even} and f^r_{odd} have been determined, they can be used to obtain the matching function f_{match} by means of a the linear combination given in Eq. (13).

Remark 2.4. The calculation of the FC(SVD) continuations of the Gram Polynomials thus depends on a small number of parameters: $d/\Delta, n_\Delta, g$, and Υ. A single set of values of these parameters, namely $n_\Delta = 10, d/\Delta = 26/9, g = 63$, and $\Upsilon = 150$, can be used for general applications including those considered here and in [10] for arbitrary accuracy, up to essentially the level of machine precision roundoff; in view of the universality of these parameters, the functions f^r_{even} and $f^r_{\text{odd}}(0 \leqslant r \leqslant 9)$ form a small set that can be precomputed. These parameter values and functions were used, in particular, for all of the numerical examples presented in this paper. The Fourier coefficients of the functions f^r_{even} and f^r_{odd} were calculated and their values at 35 points were stored in a 20×35 double precision matrix F. (Seventy points x_j lie in the interval $1 - \Delta \leqslant x_j < 1 + 2d + \Delta$; only half of the corresponding values $f^r_{\text{even}}(x_j)$ and $f^r_{\text{odd}}(x_j)$ need to be stored, however, since the values of the function f^r_{even} (resp. f^r_{odd}) in the interval $[1 + d, 1 + 2d + \Delta]$ equal the corresponding value (resp. minus the corresponding values) over the interval $[1 - \Delta, 1 + d]$). The calculation leading to the matrix F was performed in Maple's high precision environment to eliminate all errors arising from the conditioning of the FC(SVD) linear system. It was found that with the parameters given above, 48 digits of arithmetic precision were sufficient to guarantee all matrix values were obtained with 16 correct digits (double precision) which were eventually stored. All other computations leading to results in this paper were performed in the standard double precision environment.

Remark 2.5. Since n_Δ is fixed (see Remark 2.4 above), the associated value of Δ may be such that the various cases in Eq. (14) give rise to conflicting definitions. Indeed, for $n < 2n_\Delta$ the segments $[0, \Delta]$ and $[1 - \Delta, 1]$ overlap at least at one point. To avoid this difficulty we use $n \geqslant 2n_\Delta$ points along any line of data. This constraint may be relaxed to $n \geqslant n_\Delta + 1$ points by first approximating the function f over all $n < 2n_\Delta$ points by a m degree polynomial ensuring that $f^p_{\text{left}}(x) = f^p_{\text{right}}(x - 1 - d)$ over their common domain of definition (so that the conflict between the definitions of these functions is avoided). Further, for $n \leqslant n_\Delta$, our experiments indicate that refinement, either global or local (as demonstrated for the FC-AD in [63]) can be performed to insure high-order accuracy while maintaining stability. Finally, note that for some domains Ω, the mesh lines may cross the boundary more than twice (see Fig. 13). In such cases, each segment of data is treated separately.

2.4. FC(Gram) numerical examples

Approximation errors resulting from the FC(Gram) continuation of the function $f(x) = e^{\sin(2.7\pi x) + \cos(\pi x)}$ are shown in Fig. 7, together with corresponding errors for two reference methodologies: the highly competitive (spectral) Barycenter Chebyshev interpolation method [64] and linear interpolation. The maximum error produced by each method (approximated as the maximum error over $20 \cdot n$ equispaced discrete data points) is shown for a range of values of n. Note that the error of the Chebyshev interpolation decreases faster than any power of h, while the FC(Gram) used here is high-order but not spectrally accurate. We point out that, as discussed in the introduction however, the Chebyshev method requires data points arranged in a special fashion, which makes it unsuitable for solution of PDEs in general domains. If an FC(Gram) algorithm based on approximations by Gram Polynomials of degree five at the left and right-ends of the approximation interval is used ($m = 5$), the FC(Gram) contin-

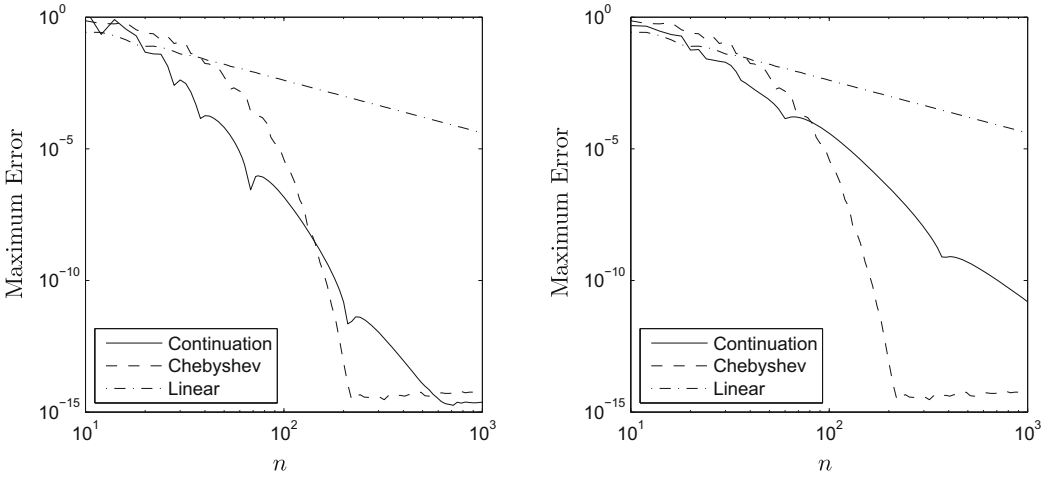

Fig. 7. Maximum interpolation error for the function $f(x) = e^{\sin(2.7\pi x)+\cos(\pi x)}$ with a variety of interpolation schemes including a 10th-order accurate FC(Gram) continuation on left and a sixth-order version on the right.

uation cannot be better than sixth-order accurate. If a ninth-degree polynomial is used instead ($m = 9$), then tenth-order convergence results. Fig. 7 shows the results $m = 9$ on the left and the results with $m = 5$ on the right.

3. Alternating direction splitting

In Sections 3.1 and 3.2 we introduce the alternating direction scheme we use for the Heat Equation in two and three dimensions, respectively; the approach we utilize for the Poisson Equation, in turn, is presented in Section 3.3. A brief outline of the full FC-AD algorithms is given in Section 3.4; the corresponding detailed algorithmic prescriptions are deferred to Section 5. Our FC(Gram)-based algorithm for the solution of the alternating-direction ODEs, which we call FC-ODE, is presented in Section 4.

3.1. Splitting of the two-d Heat Equation

We consider the Dirichlet problem for the Heat Equation

$$\begin{aligned} &u_t = k(u_{xx} + u_{yy}) + Q(x,y,t), \quad (x,y,t) \in \Omega \times (0,T], \\ &u(x,y,t) = G(x,y,t), \quad (x,y) \in \partial\Omega, t \in (0,T], \\ &u(x,y,0) = u_0(x,y), \quad (x,y) \in \Omega, \end{aligned} \tag{17}$$

where $k > 0, \Omega \subset \mathbb{R}^2$ is a smoothly bounded domain, and where Q, G, and u_0 denote given smooth functions. Let $t^n = n\Delta t$ and let u^n and $Q^{n+\frac{1}{2}}$ be equal to $u(x,y,t^n)$ and $Q(x,y,(n+1/2)\Delta t)$, respectively. Using a centered finite-difference scheme around $t = t^{n+\frac{1}{2}} = (n+\frac{1}{2})\Delta t$ to discretize the time derivative, (which results in a splitting corresponding to the classical ADI [1]) yields

$$\frac{u^{n+1}-u^n}{\Delta t} = \frac{k}{2}\frac{\partial^2}{\partial x^2}(u^{n+1}+u^n) + \frac{k}{2}\frac{\partial^2}{\partial y^2}(u^{n+1}+u^n) + Q^{n+\frac{1}{2}} + E_1(x,y,\Delta t), \tag{18}$$

where

$$E_1(x,y,\Delta t) \leqslant \frac{\Delta t^2}{24}\|u_{ttt}\|_{L^\infty(\Omega\times(t^n,t^{n+1}))} + \frac{k\Delta t^2}{8}\|u_{ttxx}\|_{L^\infty(\Omega\times(t^n,t^{n+1}))} + \frac{k\Delta t^2}{8}\|u_{ttyy}\|_{L^\infty(\Omega\times(t^n,t^{n+1}))}, \tag{19}$$

as it follows from use of Taylor series expansions with the Cauchy form of the remainder. Collecting the terms for u^n and u^{n+1} in the above equations gives

$$\left(1 - \frac{k\Delta t}{2}\frac{\partial^2}{\partial x^2} - \frac{k\Delta t}{2}\frac{\partial^2}{\partial y^2}\right)u^{n+1} = \left(1 + \frac{k\Delta t}{2}\frac{\partial^2}{\partial x^2} + \frac{k\Delta t}{2}\frac{\partial^2}{\partial y^2}\right)u^n + \Delta t Q^{n+\frac{1}{2}} + \Delta t E_1(x,y,t). \tag{20}$$

Eq. (20) can be expressed in the factored form

$$\left(1 - \frac{k\Delta t}{2}\frac{\partial^2}{\partial x^2}\right)\left(1 - \frac{k\Delta t}{2}\frac{\partial^2}{\partial y^2}\right)u^{n+1} = \left(1 + \frac{k\Delta t}{2}\frac{\partial^2}{\partial x^2}\right)\left(1 + \frac{k\Delta t}{2}\frac{\partial^2}{\partial y^2}\right)u^n + \frac{k^2\Delta t^2}{4}\frac{\partial^2}{\partial x^2}\frac{\partial^2}{\partial y^2}(u^{n+1}-u^n) + \Delta t Q^{n+\frac{1}{2}} + \Delta t E_1(x,y,t). \tag{21}$$

Again using Taylor series with the Cauchy form of the remainder, we see that

$$E_2(x,y,t) = \frac{k^2\Delta t^2}{4}\frac{\partial^2}{\partial x^2}\frac{\partial^2}{\partial y^2}(u^{n+1}-u^n) \leqslant \frac{k^2\Delta t^3}{4}\|u_{txxyy}\|_{L^\infty(\Omega\times(t^n,t^{n+1}))}, \tag{22}$$

note that $E_2(x,y,t)$ is of order $O(\Delta t^3)$. In order to solve for u^{n+1} in Eq. (21), operators of the form $\left(1-\frac{k\Delta t}{2}\frac{\partial^2}{\partial x^2}\right)$ and $\left(1-\frac{k\Delta t}{2}\frac{\partial^2}{\partial y^2}\right)$ must be inverted. The application of the inverse operators on a function f gives as a result solutions of the one-dimensional boundary value problem

$$-\alpha^2 u'' + u = f \quad u(x_\ell) = B_\ell, \quad u(x_r) = B_r, \tag{23}$$

with

$$\alpha^2 = \frac{k\Delta t}{2}, \tag{24}$$

for given boundary values B_ℓ and B_r prescribed at the boundary points x_ℓ and x_r (see Fig. 2 and Remark 3.2).

Remark 3.1. Whenever the boundary information is clear from the context, we will write

$$\left(1-\frac{k\Delta t}{2}\frac{\partial^2}{\partial x^2}\right)^{-1} = \left(1-\frac{k\Delta t}{2}\frac{\partial^2}{\partial x^2}\right)^{-1}_{x_\ell,x_r;B_\ell,B_r} \tag{25}$$

and

$$\left(1-\frac{k\Delta t}{2}\frac{\partial^2}{\partial y^2}\right)^{-1} = \left(1-\frac{k\Delta t}{2}\frac{\partial^2}{\partial y^2}\right)^{-1}_{x_\ell,x_r;B_\ell,B_r} \tag{26}$$

for the operators that produce the solution u of the boundary value problem (23) from the right-hand side f with boundary values B_ℓ and B_r and boundary locations x_ℓ and x_r.

A numerical algorithm (based on the FC(Gram) continuation method presented in the previous section) for the solution of ODEs such as Eq. (23) is presented in Section 4.

In the special case $Q = 0$, Peaceman and Rachford obtained an approximation $\tilde{u}^n$ to the exact solution u^n using a scheme of the form

$$\left(1-\frac{k\Delta t}{2}\frac{\partial^2}{\partial x^2}\right)\tilde{u}^{n+\frac{1}{2}} = \left(1+\frac{k\Delta t}{2}\frac{\partial^2}{\partial y^2}\right)\tilde{u}^n,$$
$$\left(1-\frac{k\Delta t}{2}\frac{\partial^2}{\partial y^2}\right)\tilde{u}^{n+1} = \left(1+\frac{k\Delta t}{2}\frac{\partial^2}{\partial x^2}\right)\tilde{u}^{n+\frac{1}{2}},$$

(see also [11, p. 176]). In order to account for the presence of the inhomogeneity Q, we use

$$\left(1-\frac{k\Delta t}{2}\frac{\partial^2}{\partial x^2}\right)\tilde{u}^{n+\frac{1}{2}} = \left(1+\frac{k\Delta t}{2}\frac{\partial^2}{\partial y^2}\right)\tilde{u}^n + \frac{\Delta t}{2}Q^{n+\frac{1}{4}}, \tag{27}$$

$$\left(1-\frac{k\Delta t}{2}\frac{\partial^2}{\partial y^2}\right)\tilde{u}^{n+1} = \left(1+\frac{k\Delta t}{2}\frac{\partial^2}{\partial x^2}\right)\tilde{u}^{n+\frac{1}{2}} + \frac{\Delta t}{2}Q^{n+\frac{3}{4}}, \tag{28}$$

where $Q^{n+\frac{1}{4}}$ and $Q^{n+\frac{3}{4}}$ are equal to $Q(x,y,(n+1/4)\Delta t)$ and $Q(x,y,(n+3/4)\Delta t)$, respectively.

It is easy to check that a solution of Eqs. (27) and (28) provide an approximate solution of Eq. (21). To show this we multiply Eq. (27) by $\left(1+\frac{k\Delta t}{2}\frac{\partial^2}{\partial x^2}\right)$ and Eq. (28) by $\left(1-\frac{k\Delta t}{2}\frac{\partial^2}{\partial x^2}\right)$. Noting that the operators $\left(1+\frac{k\Delta t}{2}\frac{\partial^2}{\partial x^2}\right)$ and $\left(1-\frac{k\Delta t}{2}\frac{\partial^2}{\partial x^2}\right)$ commute and subtracting the resulting equations we obtain

$$\left(1-\frac{k\Delta t}{2}\frac{\partial^2}{\partial x^2}\right)\left(1-\frac{k\Delta t}{2}\frac{\partial^2}{\partial y^2}\right)\tilde{u}^{n+1} = \left(1+\frac{k\Delta t}{2}\frac{\partial^2}{\partial x^2}\right)\left(1+\frac{k\Delta t}{2}\frac{\partial^2}{\partial y^2}\right)\tilde{u}^n + \frac{\Delta t}{2}\left(1+\frac{k\Delta t}{2}\frac{\partial^2}{\partial x^2}\right)Q^{n+\frac{1}{4}} + \frac{\Delta t}{2}\left(1-\frac{k\Delta t}{2}\frac{\partial^2}{\partial x^2}\right)Q^{n+\frac{3}{4}}. \tag{29}$$

We now introduce the approximation

$$Q^{n+\frac{1}{2}} = \frac{1}{2}\left(1+\frac{k\Delta t}{2}\frac{\partial^2}{\partial x^2}\right)Q^{n+\frac{1}{4}} + \frac{1}{2}\left(1-\frac{k\Delta t}{2}\frac{\partial^2}{\partial x^2}\right)Q^{n+\frac{3}{4}} + E_3(x,y,t), \tag{30}$$

where applying a Taylor series expansion with the Cauchy form of the remainder to the function Q we obtain the bound

$$E_3(x,y,t) \leqslant \frac{\Delta t^2}{16}\|Q_{tt}\|_{L^\infty(\Omega\times(t^n,t^{n+1}))} + \frac{k\Delta t^2}{8}\|Q_{txx}\|_{L^\infty(\Omega\times(t^n,t^{n+1}))}. \tag{31}$$

It follows that, as claimed above, a solution of Eqs. (27) and (28) is also a solution to Eq. (21) with error equal to

$$\Delta t E_1(x,y,t) + E_2(x,y,t) + \Delta t E_3(x,y,t). \tag{32}$$

Boundary values for $\tilde{u}^{n+\frac{1}{2}}$ in Eq. (27) and for $\tilde{u}^{n+1}$ in Eq. (28) are all that are needed in order to complete the scheme.

Remark 3.2. The boundary values for $\tilde{u}^{n+1}$ are simply given by the value $G(x,y,t^{n+1})$ for the appropriate boundary points $(x,y) \in \partial\Omega$. The boundary values of $\tilde{u}^{n+\frac{1}{2}}$ for the traditional ADI on a square domain are given by

$$\tilde{u}^{n+\frac{1}{2}}(x,y) = \frac{1}{2}\left(1 + \frac{k\Delta t}{2}\frac{\partial^2}{\partial y^2}\right)G(x,y,t^n) + \frac{1}{2}\left(1 - \frac{k\Delta t}{2}\frac{\partial^2}{\partial y^2}\right)G(x,y,t^{n+1}) + \frac{\Delta t}{4}\left(Q\left(x,y,t^{n+\frac{1}{4}}\right) - Q\left(x,y,t^{n+\frac{3}{4}}\right)\right), (x,y) \in \partial\Omega, \tag{33}$$

a relation that follows directly from Eqs. (27) and (28), c.f. [11, p. 176]. In the case of a complex domain, the spatial derivatives of G *are not known a priori*. In order to handle complex domains, our two-dimensional Heat-Equation FC-AD algorithm uses the boundary values

$$\tilde{u}^{n+\frac{1}{2}}(x,y) = G(x,y,(n+1/2)\Delta t). \tag{34}$$

This approximation introduces an additional time discretization error E_4 that satisfies

$$E_4(x,y,t) \leqslant \frac{k\Delta t^2}{4}\|u_{tyy}\|_{L^\infty(\Omega\times(t^n,t^{n+1}))} + \frac{\Delta t^2}{4}\|u_{tt}\|_{L^\infty(\Omega\times(t^n,t^{n+1}))} + \frac{\Delta t^2}{8}\|Q_t\|_{L^\infty(\Omega\times(t^n,t^{n+1}))}. \tag{35}$$

Accounting for the time discretization errors (32) and the error (34) on the boundary values, the overall error arising from one step of our Heat-Equation FC-AD algorithm is of order $O(\Delta t^2)$. This bound predicts an overall FC-AD error of $O(\Delta t)$. Despite this argument, our numerical experiments, including those presented in Section 6, indicate that in practice the overall accuracy of the algorithm remains $O(\Delta t^2)$. In any case, Richardson extrapolation can also be employed to increase the accuracy in time as has been shown in reference [22] and in particular is employed for the FC-AD in [10].

In order to facilitate the description of our algorithm as well as its analysis it is convenient to introduce the notation

$$w^{n+\frac{1}{2}} = \left(1 + \frac{k\Delta t}{2}\frac{\partial^2}{\partial x^2}\right)\tilde{u}^{n+\frac{1}{2}}, \tag{36}$$

and

$$w^{n+1} = \left(1 + \frac{k\Delta t}{2}\frac{\partial^2}{\partial y^2}\right)\tilde{u}^{n+1}. \tag{37}$$

In terms of these new variables and the notation introduced in Remark 3.1 the algorithm becomes

$$w^{n+\frac{1}{2}} = \left(1 + \frac{k\Delta t}{2}\frac{\partial^2}{\partial x^2}\right)\left(1 - \frac{k\Delta t}{2}\frac{\partial^2}{\partial x^2}\right)^{-1}\left(w^n + \frac{\Delta t}{2}Q^{n+\frac{1}{4}}\right), \tag{38}$$

$$w^{n+1} = \left(1 + \frac{k\Delta t}{2}\frac{\partial^2}{\partial y^2}\right)\left(1 - \frac{k\Delta t}{2}\frac{\partial^2}{\partial y^2}\right)^{-1}\left(w^{n+\frac{1}{2}} + \frac{\Delta t}{2}Q^{n+\frac{3}{4}}\right). \tag{39}$$

Remark 3.3. The result of the application of the operators on the right-hand side of Eqs. (38) and (39), e.g. $(1 + \frac{k\Delta t}{2}\frac{\partial^2}{\partial x^2})(1 - \frac{k\Delta t}{2}\frac{\partial^2}{\partial x^2})^{-1}$, to a function f does not actually require differentiation with respect to the relevant independent variable. Indeed, letting e.g.

$$q = \left(1 + \frac{k\Delta t}{2}\frac{\partial^2}{\partial x^2}\right)\left(1 - \frac{k\Delta t}{2}\frac{\partial^2}{\partial x^2}\right)^{-1} f, \tag{40}$$

we have

$$\begin{aligned} &-\alpha^2 u''(x) + u(x) = f(x), \quad u(x_\ell) = B_\ell, \quad u(x_r) = B_r, \quad \text{and} \\ &\alpha^2 u''(x) + u(x) = q(x), \end{aligned} \tag{41}$$

where, again, $\alpha^2 = \frac{k\Delta t}{2}$. Adding these equations we obtain

$$q(x) = 2u(x) - f(x), \tag{42}$$

where $u(x)$ is the solution to the first equation in (41); clearly, a similar result holds for the operator $(1 + \frac{k\Delta t}{2}\frac{\partial^2}{\partial y^2})(1 - \frac{k\Delta t}{2}\frac{\partial^2}{\partial y^2})^{-1}$. It follows that each half-step of (38) and (39) can be computed by solving the first equation in (41) followed by use of Eq. (42) or its y-dependent counterpart.

3.2. Splitting of the three-d Heat Equation

We now consider the Dirichlet problem for the three-d Heat Equation

$$\begin{aligned} &u_t = k(u_{xx} + u_{yy} + u_{zz}) + Q(x,y,z,t), \quad (x,y,z,t) \in \Omega \times (0,T], \\ &u(x,y,z,t) = G(x,y,z,t), \quad (x,y,z) \in \partial\Omega, t \in (0,T], \\ &u(x,y,z,0) = u_0(x,y,z), \quad (x,y,z) \in \Omega, \end{aligned} \tag{43}$$

where $k > 0, \Omega \subset \mathbb{R}^3$ is a smoothly bounded domain, and where Q, G, and u_0 denote given smooth functions. Several Alternating-Direction-type splittings are available for the three-dimensional Heat Equation, including, e.g. the Crank–Nicolson-based Locally One-Dimensional (LOD) scheme [9, p. 71]. In view of the variety of options available, and for the sake of definiteness, here we use a simple approach that, based on implicit Euler time-stepping, delivers first-order accuracy in time. As mentioned above and demonstrated in [10], use of Richardson extrapolation can be made to increase significantly the order of temporal accuracy occurring in a low-temporal-order FC-AD algorithm.

Using implicit Euler time-stepping we have

$$\frac{u^{n+1} - u^n}{\Delta t} = k\frac{\partial^2}{\partial x^2}u^{n+1} + k\frac{\partial^2}{\partial y^2}u^{n+1} + k\frac{\partial^2}{\partial z^2}u^{n+1} + Q^{n+1} + E_1(x,y,z,\Delta t), \tag{44}$$

where E_1 is the error introduced by the time discretization. Collecting the terms for u^{n+1} on the left side of the equation and factoring yields,

$$\left(1 - k\Delta t\frac{\partial^2}{\partial x^2}\right)\left(1 - k\Delta t\frac{\partial^2}{\partial y^2}\right)\left(1 - k\Delta t\frac{\partial^2}{\partial z^2}\right)u^{n+1} = u^n + \Delta t Q^{n+1} + E_2(x,y,z,t), \tag{45}$$

where the error term E_2 is given by

$$E_2(x,y,z,t) = k^2\Delta t^2\left(k\Delta t\frac{\partial^2}{\partial x^2}\frac{\partial^2}{\partial y^2}\frac{\partial^2}{\partial z^2} - \frac{\partial^2}{\partial x^2}\frac{\partial^2}{\partial y^2} - \frac{\partial^2}{\partial y^2}\frac{\partial^2}{\partial z^2} - \frac{\partial^2}{\partial x^2}\frac{\partial^2}{\partial z^2}\right)u^{n+1} + \Delta t E_1(x,y,z,t). \tag{46}$$

To complete the scheme, $G(x,y,z,t^{n+1})$ is used to provide approximate boundary values (with error of the order of $O(Deltat)$) for each of the three inverse operators $(1 - k\Delta t\frac{\partial^2}{\partial x^2})^{-1}, (1 - k\Delta t\frac{\partial^2}{\partial y^2})^{-1}$ and $(1 - k\Delta t\frac{\partial^2}{\partial z^2})^{-1}$ that must be applied in order to obtain u^{n+1} in Eq. (45). A variety of numerical results for the three-d Heat Equation, including those presented in Section 6, indicate that use of such boundary conditions does indeed give rise to a convergent numerical scheme.

3.3. The Poisson Equation: iteration parameters

Since the Poisson Equation,

$$\begin{aligned} &-u_{xx} - u_{yy} = Q(x,y), \quad (x,y) \in \Omega, \\ &u(x,y) = G(x,y), \quad (x,y) \in \partial\Omega, \end{aligned} \tag{47}$$

is the steady-state version to the Heat Equation, the splitting scheme in Eq. (21) could be directly applied to Eq. (47). For efficiency, however, it has long been recognized that the introduction sequences of iteration parameters (that effectively amount to corresponding changes of Δt for each time-step in the Heat Equation solver), can give rise to significantly faster convergence to the solution of the Poisson Equation (c.f. [65, p. 597]). With the iteration parameters γ_n the iteration schemes (38) and (39) adapted for the Poisson Equation becomes

$$w^{n+\frac{1}{2}} = \left(1 + \gamma_n\frac{\partial^2}{\partial x^2}\right)\left(1 - \gamma_n\frac{\partial^2}{\partial x^2}\right)^{-1}(w^n + \gamma_n Q), \tag{48}$$

$$w^{n+1} = \left(1 + \gamma_{n+1}\frac{\partial^2}{\partial y^2}\right)\left(1 - \gamma_n\frac{\partial^2}{\partial y^2}\right)^{-1}\left(w^{n+\frac{1}{2}} + \gamma_n Q\right), \tag{49}$$

with boundary conditions given by $G(x,y)$.

The determination of optimal iteration parameters, γ_n, have been extensively researched in the finite-difference case (c.f., [66,67]). We propose a simple choice of iteration parameters that provide surprisingly efficient performance; our prescription results from the following considerations. The result of an application of the operator $(1 + \gamma_n\frac{\partial^2}{\partial x^2})(1 - \gamma_n\frac{\partial^2}{\partial x^2})^{-1}$ (with periodic boundary conditions) to a Fourier series

$$\mu(x) = \sum_{k \in t(N)} a_k e^{iPxk} \tag{50}$$

is given by

$$\left(1 + \gamma_n\frac{\partial^2}{\partial x^2}\right)\left(1 - \gamma_n\frac{\partial^2}{\partial x^2}\right)^{-1}\mu(x) = \sum_{k \in t(N)} \frac{1 - \gamma_n P^2k^2}{1 + \gamma_n P^2k^2} a_k e^{iPxk}. \tag{51}$$

If we consider the function $\mu(x)$ to be the error in our approximate solution, then components of the error in Fourier modes with index $k \approx \frac{1}{P\sqrt{\gamma_n}}$ will be reduced significantly while the error in modes with index k far from $\frac{1}{P\sqrt{\gamma_n}}$ will remain nearly the same in magnitude. Using a small parameter ε, our selection of iteration parameters γ_n seeks to ensure that every Fourier mode is reduced by at least a factor of ε at least once. We thus use

$$\gamma_{n+1} = \phi\gamma_n \quad \text{with} \quad \phi = (1-\varepsilon)^2/(1+\varepsilon)^2, \tag{52}$$

giving a total of

$$N_I = -2\log(2P/h)/\log(\phi) \tag{53}$$

iteration steps. We have found that $\gamma_0 = \frac{\pi^2}{10P^2}$ works consistently very well for all values of ε. Reductions in ε accompanied by increases in the numbers of iterations give rise to increased accuracies, in turn, provided the resolution of the spatial mesh allows it.

3.4. Overall FC-AD strategy for the two- and three-d heat and poisson equations

In order to provide an adequate lead into the detailed algorithmic prescriptions presented in Sections 4 and 5, in this section we present a brief summary of the overall FC-AD procedure.

Our two-dimensional Heat-Equation algorithms, on one hand, evolve the approximate solution through solution use of discrete versions of Eqs. (38) and (39). The needed spatially-discrete approximate versions of the composite operators on the right hand of these equations are presented in Section 4. The ODE boundary conditions on $\tilde{u}^n$ and $\tilde{u}^{n+\frac{1}{2}}$ needed for application of the inverse differential operators (see Eqs. (36) and (37)), are prescribed as indicated in Remark 3.2.

In the three-dimensional case, on the other hand, the corresponding discrete equations we use result as the error term E_2 is dropped in Eq. (45) and the resulting approximate equations are similarly solved. Iterations for the Poisson solver, finally, proceed similarly except that $\frac{k\Delta t}{2}$ is replaced by γ_n. In particular, the two-d and three-d Heat- and Poisson-Equation solvers proceed via inversion of the same type of discrete differential operators.

Remark 3.4. In the case of the two-d Heat Equation scheme we use, the unknown evolved is not the approximate solution $\tilde{u}^n$. This quantity is actually produced as an intermediate result, however,

$$\tilde{u}^{n+1} = \left(1 - \frac{k\Delta t}{2}\frac{\partial^2}{\partial y^2}\right)^{-1}\left(w^{n+\frac{1}{2}} + \frac{\Delta t}{2}Q^{n+\frac{3}{4}}\right),$$

which is needed in the evaluation of w^{n+1}; see Remark 3.3.

4. FC-ODE algorithm: solution of ODEs by means of Fourier continuation

In view of Section 3.4, the last main element needed for the implementation of the FC-AD solver for the Heat and Poisson Equations is a suitable discrete operator $\mathcal{S}_{\alpha^2,x}$ that approximates the inverse of a simple differential operator with constant coefficients:

$$\mathcal{S}_{\alpha^2,x} \approx \left(1 - \alpha^2\frac{\partial^2}{\partial x^2}\right)^{-1}, \tag{54}$$

once such an operator is available, the composite operators in Eqs. (38), (39), (48) and (49) mentioned in Section 3.4 are then obtained easily; see Remark 4.2.

In detail, then, given a discrete right-hand-side $f = (f_j)$ (an approximation $f_j \approx F(x_j)$ of a smooth right-hand-side $F(x)$), the operator $\mathcal{S}_{\alpha^2,x}$ returns a discrete approximation $\check{u} = (\check{u}_j)$ to the solution u of the ODE (23),

$$\check{u} = \mathcal{S}_{\alpha^2,x}[f], \tag{55}$$

or, including explicitly the dependence on boundary data,

$$\check{u} = \mathcal{S}^{x_\ell,x_r;B_\ell,B_r}_{\alpha^2,x}[f], \tag{56}$$

(see Remark 3.1), where the FC-ODE discrete-solver operator on the right-hand sides of (54) and (56) is defined in what follows.

It is not difficult to obtain such an operator on the basis of the FC(Gram) algorithm. A number of special elements, that are not necessary if one is merely interested in solution of ODEs, are incorporated in the construction of the operator (56) in order to insure the stability and convergence of the resulting overall FC-AD solver—as described below in this section, and studied in [10]. To obtain the operator (56), then, the FC-ODE algorithm proceeds by first approximating the right-hand side discrete data f by an FC(Gram) continuation Fourier series

$$f^c(x) = \sum_{k\in t(n+n_d-2)} a_k e^{2\pi i\frac{xk}{(x_r-x_\ell)(1+d)}}. \tag{57}$$

The solution of

$$-\alpha^2 v''(x) + v(x) = f^c(x), \quad v(x_\ell) = B_\ell, \quad v(x_r) = B_r \tag{58}$$

is then obtained; including the appropriate solution of the associated homogeneous problem, which is needed to meet the boundary conditions, the solution v of (58) is given by

$$v(x) = \sum_{k\in t(n+n_d-2)} \frac{a_k}{1+\frac{4\alpha^2\pi^2k^2}{(x_r-x_\ell)^2(1+d)^2}} e^{2\pi i \frac{xk}{(x_r-x_\ell)(1+d)}} + c_1 h_1(x) + c_2 h_2(x), \tag{59}$$

where c_1 and c_2 are constants chosen to fit the boundary conditions and where h_1 and h_2 are the homogeneous solutions

$$h_1(x) = e^{x/|\alpha|} \quad \text{and} \quad h_2(x) = e^{-x/|\alpha|}. \tag{60}$$

In the context of the FC-AD algorithm (Section 3) we have $\alpha^2 = k\Delta t/2$; in particular $\alpha \to 0$ as $\Delta t \to 0$. In view of the right-hand-side approximation f^c of f used, however, $v(x) - u(x)$ does not converge to zero as $\alpha \to 0$, as it would if a finite-difference ODE solver were used for (23); instead we have

$$u(x_j) - v(x_j) \to f_j - f^c(x_j) \neq 0 \text{ as } \alpha \to 0 \quad \text{for } x_j \text{ near the boundary points } \ x_\ell \text{ and } x_r; \tag{61}$$

see Remark 2.3.

Remark 4.1. In view of the lack of convergence to zero displayed in Eq. (61), use of a scheme such as (58) as part of our FC-AD algorithm would result in certain types of conditional convergence—e.g. lack of convergence as Δt tends to 0 much faster than h tends to zero, resulting from accumulation of the error (61) over a large $O(1/\Delta t)$ number of time-steps for very small values of Δt. Although this theoretical issue is seldom of practical importance, we introduce corrections ensuring that the corrected approximate solution $\check{u}$ of (55) satisfies

$$u(x_j) - \check{u}_j \to 0 \text{ as } \alpha \to 0 \quad \text{for all } x_j. \tag{62}$$

A simple correction that restores the aforementioned convergence to zero as $\alpha \to 0$, (and, additionally, gives rise to *high-order accuracy*) is obtained by means of *low-order* accurate finite-difference techniques: the correction $\eta = (\eta_j)$ is obtained as the solution of the equations

$$-\alpha^2 \frac{\eta_{j+1} - 2\eta_j + \eta_{j-1}}{h^2} + \eta_j = f_j - f^c(x_j), \quad \text{for } j \in \{1,\dots,n_\Delta\} \cup \{n - n_\Delta + 1,\dots,n\}, \tag{63}$$

$$\eta_j = 0 \quad \text{otherwise}, \tag{64}$$

with boundary conditions $\eta_0 = 0, \eta_{n_\Delta+1} = 0, \eta_{n-n_\Delta} = 0$, and $\eta_{n+1} = 0$. (A banded LU Decomposition allows η to be calculated in $O(n_\Delta)$ operations (c.f. [68, p.90])). It is easy to check that the corrected solution $v(x_j) + \eta_j$ satisfies $u(x_j) - (v(x_j) + \eta_j) \to 0$ as $\alpha \to 0$, as desired.

As mentioned above, a number of additional components are needed in the FC-ODE algorithm in order to ensure the stability of the full FC-AD method. These components, which are listed later in this section, were obtained through a significant amount of experimentation. They involve a series of orthogonal projections; as established rigorously in [10] and as demonstrated later in this text, the resulting FC-ODE solver gives rise to an unconditionally stable and high-order accurate overall FC-AD algorithm. To present the FC-ODE stability restoring steps we introduce the following two important definitions. Note the close relationship, as well as a subtle but significant difference, between the use of Gram Polynomials made here and that made in Section 2: the definitions below relate to the domain $[x_\ell, x_r]$ that is slightly larger than the interval $[x_1, x_n]$ considered in Section 2.

Definition 4.1. For a given discrete function f defined at the set of points $\{x_j, j = 1,\dots,n\}$, we define the open boundary projection f^p of f; this definition requires the use of two discrete orthonormal Gram Polynomial bases ([60, p. 323 and 61]). The first such basis consists of a set of polynomials $\{P^r_{o,\text{right}}(x+1+d)\}_{r=0}^m$ of degree $\leqslant m$ that are orthonormal with respect to the discrete scalar product $(g,h)_{o,\text{right}} = \sum g(x_j)h(x_j)$, where the sum extends over the set of points

$$\{x_j, j = 1,\dots,n_\Delta\}.$$

The second Gram Polynomial basis, analogously, $\{P^r_{o,\text{left}}(x)\}_{r=0}^m$ consists of polynomials of degree $\leqslant m$ that are orthonormal with respect to the discrete scalar product $(g,h)_{o,\text{left}} = \sum g(x_j)h(x_j)$ where the sum extends over the set of points

$$\{x_j, j = n - n_\Delta + 1,\dots,n\}.$$

The open boundary projection f^p of f is then defined by

$$f^p_j = \begin{cases} \sum_{r\in\{0,\dots,m\}} P^r_{o,\text{right}}(x_j+1+d) \sum_{k\in\{1,\dots,n_\Delta\}} f_k P^r_{o,\text{right}}(x_k+1+d) & \text{for } j \in \{1,\dots,n_\Delta\} \\ f_j & \text{for } j \in \{n_\Delta+1,\dots,n-n_\Delta\}. \\ \sum_{r\in\{0,\dots,m\}} P^r_{o,\text{left}}(x_j) \sum_{k\in\{n-n_\Delta+1,\dots,n\}} f_k P^r_{o,\text{left}}(x_k) & \text{for } j \in \{n-n_\Delta+1,\dots,n\} \end{cases} \tag{65}$$

Clearly, at the discrete points in the boundary segment $(x_\ell, x_{n_\Delta}]$ (resp. $[x_{n-n_\Delta+1}, x_r)$), f^p equals the projection of f according to the scalar product $(\cdot,\cdot)_{o,\text{right}}$ (resp. $(\cdot,\cdot)_{o,\text{left}}$).

Definition 4.2. For a given discrete function f defined at the set of points $\{x_\ell\} \cup \{x_j, j = 1, \ldots, n\} \cup \{x_r\}$, the closed boundary projection f^b of f is defined as follows. An orthonormal polynomial basis $\{P^r_{c,\text{right}}(x + 1 + d)\}_{r=0}^m$, consisting of polynomials of degree $\leqslant m$ that are orthonormal with respect to the discrete scalar product $(g, h)_{c,\text{right}} = \sum g(x_j)h(x_j)$ over the set of points

$$\{x_j, j = 1, \ldots, n_\Delta\} \cup \{x_\ell\}$$

is formed. Similarly, an orthonormal polynomial basis $\{P^r_{c,\text{left}}(x)\}_{r=0}^m$ is formed on the basis of the discrete scalar product $(f, g)_{c,\text{left}} = \sum f(x_j)g(x_j)$ over the set of points

$$\{x_j, j = n - n_\Delta + 1, \ldots, n\} \cup \{x_r\}.$$

The closed boundary projection f^b of f is then defined by

$$f^b_j = \begin{cases} \sum_{r\in\{0,\ldots,m\}} P^r_{c,\text{right}}(x_j + 1 + d) \sum_{k\in\{1,\ldots,n_\Delta\}\cup\{\ell\}} f_k P^r_{c,\text{right}}(x_k + 1 + d) & \text{for } j \in \{1, \ldots, n_\Delta\} \cup \{\ell\} \\ f_j & \text{for } j \in \{n_\Delta + 1, \ldots, n - n_\Delta\} \\ \sum_{r\in\{0,\ldots,m\}} P^r_{c,\text{left}}(x_j) \sum_{k\in\{n-n_\Delta+1,\ldots,n\}\cup\{r\}} f_k P^r_{c,\text{left}}(x_k) & \text{for } j \in \{n - n_\Delta + 1, \ldots, n\} \cup \{r\} \end{cases}. \tag{66}$$

Clearly, at the discrete points in the boundary segment $[x_\ell, x_{n_\Delta}]$ (resp. $[x_{n-n_\Delta+1}, x_r]$), f^b equals the projection of f according to the scalar product $(\cdot,\cdot)_{c,\text{right}}$ (resp. $(\cdot,\cdot)_{c,\text{left}}$).

We are now ready to introduce our discrete operator (56). To obtain $\breve{u}_j$ we proceed as follows: we

1. Construct the open boundary projection v^p of $v(x_j)$ for $j = 1, \ldots, n$ (Eq. (59)) according to Definition 4.1.
2. Construct the closed boundary projection v^b of $v(x_j)$ for $j = \ell, 1, \ldots, n, r$ according to Definition 4.2.
3. Project the calculated finite-difference solution η (Eq. (64)) into the open Gram Polynomial basis according to Definition 4.1 to obtain the projection η^p.

Our stability corrected solution $\breve{u}$, presented in step 4. below, results as a combination of the projections mentioned in steps 1. through 3. The main goal leading to the specific combination of projections we use is to preserve, at the same time, the validity of (62) and the stability of the algorithm; we obtained our final expression for the solution $\breve{u}$ by seeking to achieve such dual goal through a combination of heuristic arguments and a degree of experimentation. In particular we found that use of the closed boundary projection v^b gives rise to stability but, owing to the inclusion of the boundary points x_ℓ and x_r in the projection scheme, $v^b_j + \eta_j - u(x_j)$ actually does not tend to 0 as $\alpha \to 0$—since for very small values of α relative to the spatial mesh-size, the solutions (60) vanish at all spatial non-boundary discretization points, and, thus, v^b does not change for values of α below a certain positive level. The projection v^p, on the other hand, does exhibit the desired $(v^p_j + \eta_j) - u(x_j) \to 0$ property as $\alpha \to 0$; unfortunately, however, use of v^p alone does not to give rise to unconditionally stable numerics when used in conjunction with the FC-AD methodology.

We found that a linear combination in terms of a variable coefficient $\chi = \chi(\alpha, h)$

$$\chi = \min(25\alpha^2/h^2, 1) \tag{67}$$

induces stability while giving rise to convergence of the boundary values. The specific form (67) and, in particular, the constant 25 used in that expression, were obtained through experimentation, and was found to insure unconditional stability as the resulting ODE solver is used as part of the overall FC-AD method. We note that both v^p and v^b are high-order accurate approximations of v and therefore so is their linear combination. The projection in step 3. was similarly undertaken for stability considerations and since η_j is an error term, this projection has no significant effect on the overall error. Thus:

4. The solution $\breve{u} = (\breve{u}_j)$ is defined by

$$\breve{u}_j = \eta_j - \eta^p_j + (1 - \chi)v^p_j + \chi v^b_j, \tag{68}$$

and the discrete operator (54), finally, is defined by

$$\mathcal{S}_{\alpha^2,x}[f] = \breve{u}. \tag{69}$$

Remark 4.2. Once the approximate solution $\breve{u}_j$ to the differential Eq. (23) has been obtained, our approximation for the function q in Eqs. (40) and (42) is given by

$$\breve{q}_j = 2\breve{u}_j - f_j. \tag{70}$$

5. Heat and Poisson FC-AD solvers: algorithmic prescriptions

All the elements are now in place for the implementation of our unconditionally stable and high-order accurate FC-AD methodology for the Heat and Poisson Equations. (FC-AD algorithms for Hyperbolic PDEs are provided in [10], along with error analyses of the various FC-AD algorithms.) In this section we provide a complete description of the overall FC-AD

methodology for the two-dimensional parabolic and elliptic cases; the three-dimensional algorithms result from analogous implementations of the schemes outlined in Sections 3.2 and 3.3.

In order to facilitate reference to the grid points in the interior of the domain, we introduce some additional notations. Let the bounded open set Ω (the domain of the PDE, see Fig. 1) be contained in $[a_x, b_x] \times [a_y, b_y]$, let

$$\{(x_i, y_j) : 1 \leqslant i \leqslant N_x, 1 \leqslant j \leqslant N_y\} \tag{71}$$

be a Cartesian mesh in the rectangle $[a_x, b_x] \times [a_y, b_y]$, and call $\mathcal{D}_\Omega = G \cap \Omega$ the set of mesh points interior to Ω:

$$\mathcal{D}_\Omega = \{(x_i, y_j) \in [a_x, b_x] \times [a_y, b_y] : (x_i, y_j) \in \Omega\}. \tag{72}$$

For a given point $(x_i, y_j) \in \mathcal{D}_\Omega$, let n_{x_i} be the number of points in $\mathcal{D}_\Omega$ on the same vertical line as (x_i, y_j), and let n_{y_j} be the corresponding number of points in $\mathcal{D}_\Omega$ on the same horizontal line. We also introduce some vector spaces associated with these meshes: we let $\ell_2(n) = \mathbb{R}^n$ be the usual n-dimensional vector space (in our constructions n may be either n_{x_i} or n_{y_j} for some x_i or y_j) and we define the space $\ell_2(\mathcal{D}_\Omega) = \mathbb{R}^{D_\Omega}$—that is the set of all functions from D_Ω to $\mathbb{R}$.

For selected values of the continuation parameters n_Δ, $d/\Delta, g$, and Υ, (all the simulations in this paper use the parameter values given in Remark 2.4), we introduce the following definitions:

Definition 5.1. Let an n point discretization of the interval $[x_\ell, x_r]$, as depicted in Fig. 2, be given, and let L_1 denote the FC-ODE solution operator with boundary-values B_ℓ and B_r: $L_1 = \mathcal{S}^{x_\ell, x_r; B_\ell, B_r}_{\alpha^2, x}$ (see Eq. (56)). Further let L_2 be the linear map from $\ell_2(n) \to \ell_2(n)$ defined by $L_2 f = (2L_1 - I)f$. Note that $\check{q}_j = \sum_{j=1}^n L_{2,ij} f_j$ is an approximation of $q(x) = 2u(x) - f(x)$ (see Eqs. (42) and (70)).

A complete definition of the FC-ODE solution operator $\mathcal{S}^{x_\ell, x_r; B_\ell, B_r}_{\alpha^2, x}$ is given in Section 4 and follows from an application of the FC(Gram) Fourier continuation algorithm (see Section 2) with certain corrections needed for the application to our alternating direction PDE algorithms.

Definition 5.2. For a given $\theta \in \ell_2(\mathcal{D}_\Omega)$, $\theta_{ij} = \theta(x_i, y_j)$ we define the operators $\mathcal{L}^1_x : \ell_2(\mathcal{D}_\Omega) \to \ell_2(\mathcal{D}_\Omega)$, $\mathcal{L}^1_y : \ell_2(\mathcal{D}_\Omega) \to \ell_2(\mathcal{D}_\Omega)$, $\mathcal{L}^2_x : \ell_2(\mathcal{D}_\Omega) \to \ell_2(\mathcal{D}_\Omega)$, and $\mathcal{L}^2_y : \ell_2(\mathcal{D}_\Omega) \to \ell_2(\mathcal{D}_\Omega)$ by

$$\begin{aligned}
\mathcal{L}^1_x[\theta](x_i, y_j) &= \sum_{k=1}^{n_{y_j}} L_{1,ik}\theta_{kj}, \\
\mathcal{L}^1_y[\theta](x_i, y_j) &= \sum_{k=1}^{n_{x_i}} L_{1,jk}\theta_{ik}, \\
\mathcal{L}^2_x[\theta](x_i, y_j) &= \sum_{k=1}^{n_{y_j}} L_{2,ik}\theta_{kj}, \quad \text{and} \\
\mathcal{L}^2_y[\theta](x_i, y_j) &= \sum_{k=1}^{n_{x_i}} L_{2,jk}\theta_{ik}.
\end{aligned} \tag{73}$$

The Heat Equation FC-AD algorithm introduced can easily be expressed in terms of these operators with appropriate boundary values (see Remark 3.2) and $\alpha^2 = \frac{k\Delta t}{2}$: the approximate solution $\check{u}^n = (\check{u}^n_{ij})$ produced at time $t_n = n\Delta t$ is given by

$$\check{u}^n = \mathcal{L}^1_y\left[\mathcal{L}^2_x\left[\check{w}^{n-1}\right]\right], \tag{74}$$

where $\check{w}^n$ is evolved according to

$$\check{w}^n = \mathcal{L}^2_y\left[\mathcal{L}^2_x\left[\check{w}^{n-1}\right]\right] = 2\check{u}^n - \mathcal{L}^2_x\left[\check{w}^{n-1}\right], \tag{75}$$

with initial condition $\check{w}^0_{ij} = w_0(x_i, y_j)$ where

$$w_0(x, y) = \left(1 + \frac{k\Delta t}{2}\frac{\partial^2}{\partial y^2}\right) u_0(x, y). \tag{76}$$

For the Poisson Equation (see Section 3.3), using simple algebraic manipulations we obtain

$$\left(1 + \gamma_n \frac{\partial^2}{\partial y^2}\right)\left(1 - \gamma_{n-1}\frac{\partial^2}{\partial y^2}\right)^{-1} = \frac{\gamma_n}{\gamma_{n-1}}\left(1 + \gamma_{n-1}\frac{\partial^2}{\partial y^2}\right)\left(1 - \gamma_{n-1}\frac{\partial^2}{\partial y^2}\right)^{-1} + \left(1 - \frac{\gamma_n}{\gamma_{n-1}}\right)\left(1 - \gamma_{n-1}\frac{\partial^2}{\partial y^2}\right)^{-1}, \tag{77}$$

which in our discrete setting is simply the linear combination

$$\frac{\gamma_n}{\gamma_{n-1}} L_2 + \left(1 - \frac{\gamma_n}{\gamma_{n-1}}\right) L_1 = \left(1 + \frac{\gamma_n}{\gamma_{n-1}}\right) L_1 - \frac{\gamma_n}{\gamma_{n-1}} I, \tag{78}$$

of the linear maps given in Definition 5.1 with $\alpha^2 = \gamma_{n-1}$. The approximate solution $\check{u}^n = (\check{u}^n_{ij})$ produced after n iterations is thus given by

$$\check{u}^n = \mathcal{L}^1_y\left[\mathcal{L}^2_x\left[\check{w}^{n-1}\right]\right], \tag{79}$$

where $\check{w}^n$ is evolved according to

$$\check{w}^n = \left(1 + \frac{\gamma_n}{\gamma_{n-1}}\right)\check{u}^n - \frac{\gamma_n}{\gamma_{n-1}}\mathcal{L}^2_x\left[\check{w}^{n-1}\right] \tag{80}$$

with initial condition $\check{w}^0_{ij} = 0$.

6. Numerical results

In this section we demonstrate the properties of the FC-AD algorithms described in Section 5 through consideration of a number of results in two and three dimensions and for both the Heat and Poisson Equations. As mentioned in the introduction, our implementations use the FFTW [59] library, which evaluates an FFT of size n in $\mathcal{O}(n\log(n))$ operations for any integer n, including n given by products of a small number of prime numbers, or even prime values of n. As indicated in Section 1, reductions in the overall $\mathcal{O}(N\log(N))$ proportionality constant could conceivably be obtained by slight modifications of the FC-AD method that avoid evaluation of FFTs of sizes n containing large prime numbers; such modifications have not been pursued as yet.

6.1. Parabolic equations: high-order accuracy and unconditional stability in two- and three-dimensional spatial domains

We consider first the FC-AD solution to the PDE (17) where the right-hand-side and boundary conditions are chosen so that the exact solution of the problem is given by

$$u(x,y,t) = \sin(\pi(9x^2 + 4y^2 + 2t)) \tag{81}$$

with $k = 1$, and where the domain Ω is the interior of the curve

$$\begin{aligned} x(\theta) &= a_x(10\sin^2(2\theta) + 3\cos^3(2\theta) + 40)\cos(\theta) + 0.5, \\ y(\theta) &= a_y(10\sin^2(2\theta) + 3\cos^3(2\theta) + 40)\sin(\theta) + 0.5, \end{aligned} \tag{82}$$

$0 \leqslant \theta \leqslant 2\pi$. The constants a_x, and a_y where chosen so that the domain Ω exactly fits the unit square; see Fig. 8. As shown in [10], use of polynomial projections of degrees $m \leqslant 4$ (Eq. (12)) results in unconditional stability for the two-d Heat-Equation FC-AD algorithm described in Section 5. In what follows we use $m = 4$ and we therefore expect unconditional stability and fifth-order spatial convergence; in accordance with Remark 3.2, in turn, we expect second-order of convergence in time.

In order to demonstrate the convergence with respect to the time-step, in the first example a fine spatial discretization was fixed ($h_x = h_y = 1.0 \times 10^{-3}$), and the solver was run with multiple values of Δt to a final time $T = 1.0$. In the left portion of Fig. 9 we display the maximum error thus obtained over the discrete mesh $\mathcal{D}_\Omega$ for all times $t \leqslant T$ as a function of Δt. This graph demonstrates a second-order convergence rate in time until the limiting accuracy of the spatial resolution is reached. (As shown in [10], higher-order accuracy in time can be obtained by incorporating Richardson extrapolation in the FC-AD algorithms.) In a second test, we used a fixed time-step of $\Delta t = 10^{-5}$, and we computed the maximum error for all points

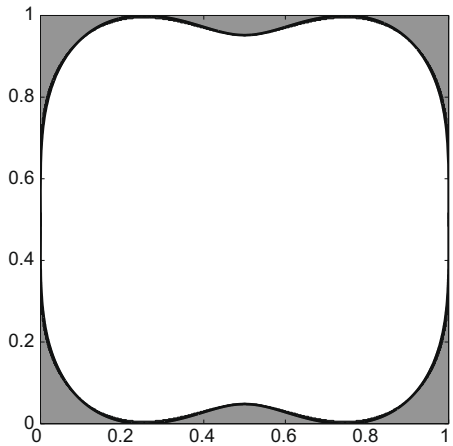

Fig. 8. Domain used for a demonstration of Heat-Equation FC-AD solver. The domain boundary is defined in Eq. (82).

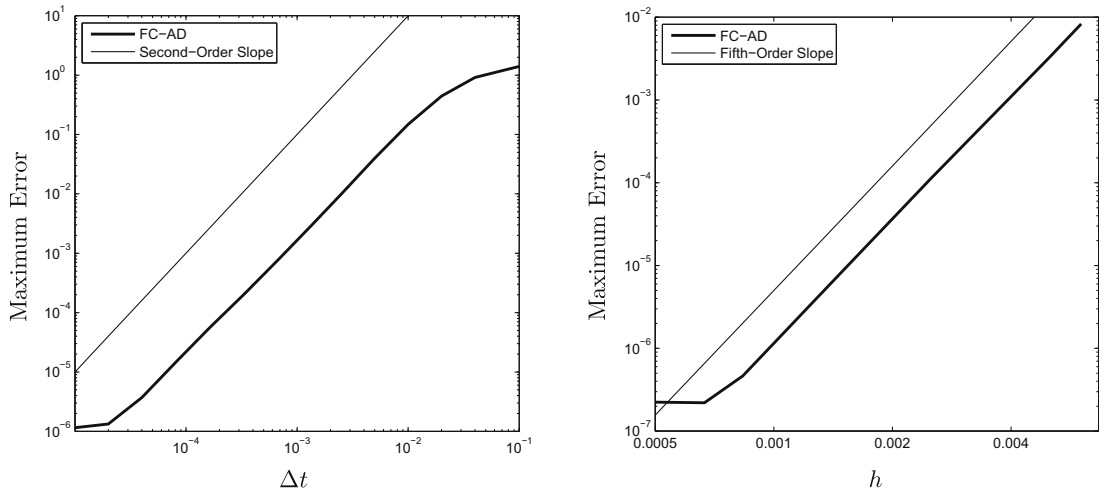

Fig. 9. Convergence of the Heat-Equation FC-AD solver on the domain depicted in Fig. 8. Left: convergence as $\Delta t \to 0$. Right: convergence as $h \to 0$.

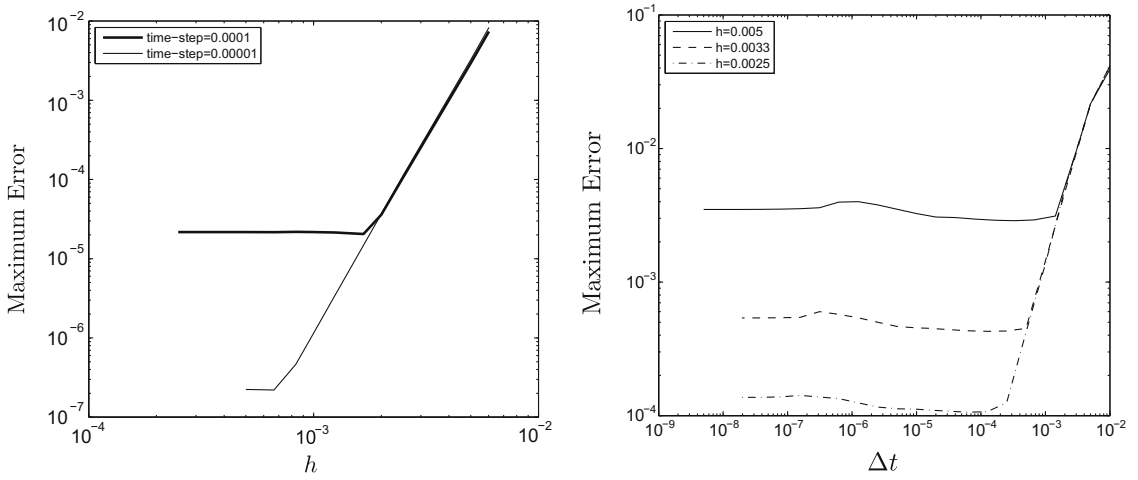

Fig. 10. Left: unconditional stability demonstrated by refining the spatial discretization for fixed $\Delta t = 10^{-4}$ and $\Delta t = 10^{-5}$. Right: maximum error, in time and space, for a wide range of values of Δt and for three fixed spatial resolutions of $h = 5.0 \times 10^{-3}$, $h = 3.3 \times 10^{-3}$, and $h = 2.5 \times 10^{-3}$, demonstrating convergence to the spatial resolution limit as $\Delta t \to 0$.

in space and all time-steps up to a final time of $T = 0.01$. The errors thus obtained are shown in the right portion of Fig. 9 as a function of the spatial mesh-size along with a fifth-order slope line; clearly the expected fifth-order convergence is achieved.

To demonstrate unconditional stability for the problem under consideration, the time-steps $\Delta t = 10^{-4}$ and $\Delta t = 10^{-5}$ were chosen and the mesh-size h was varied. The corresponding maximum errors over the spatial grid for all times $\leqslant 0.01$ are displayed, as a function of h, on the left portion of Fig. 10. The stability limit for a traditional explicit method, which is governed by the minimum distance between two discretization points and the minimum distance of a discretization point to the boundary, for the present problem requires the time-step to be smaller than $O(10^{-7})$; note our solver is stable with a time-step of 10^{-4}. The FC-AD algorithm was also run one thousand time-steps with $\Delta t = 100$ for the previous problem. While the solution was inaccurate, the stability of the approximate solution was nonetheless observed.

To demonstrate the effectiveness of the correction step introduced in Eq. (64), on the other hand, we performed an experiment, the results of which are presented in Fig. 10 right, where Δt is refined up to extremely small values for the fixed spatial resolutions of $h = 5.0 \times 10^{-3}, h = 3.3 \times 10^{-3}$, and $h = 2.5 \times 10^{-3}$. The maximum error at any time-step calculated to a final time $t = 0.01$ is displayed on the right in Fig. 10: clearly, convergence to the various spatial-accuracy levels was achieved. Note a slight bump in each one of the three curves on the right portion of Fig. 10. This bump occurs at precisely the time-step $\Delta t = 25h^2/2$ at which χ (see Eq. (68)) switches from 1 to $25\alpha^2/h^2$, providing evidence of the necessity of this parameter for convergence. For smaller values of Δt, beyond this bump, a strict, albeit small, reduction in the maximum error is observed.

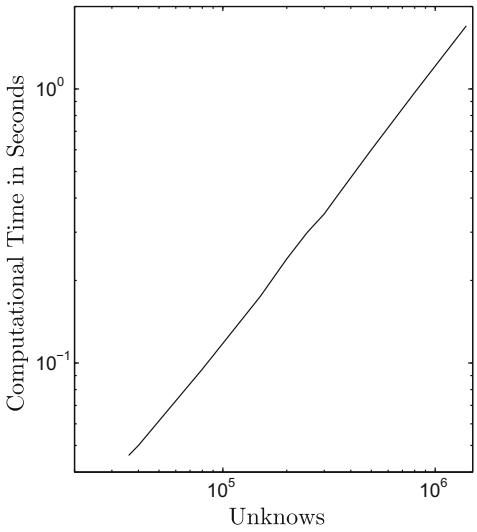

Fig. 11. Computational time for a single time-step of the Heat Equation on a single processing core of a 2.33 GHz Intel Core 2 Duo processor.

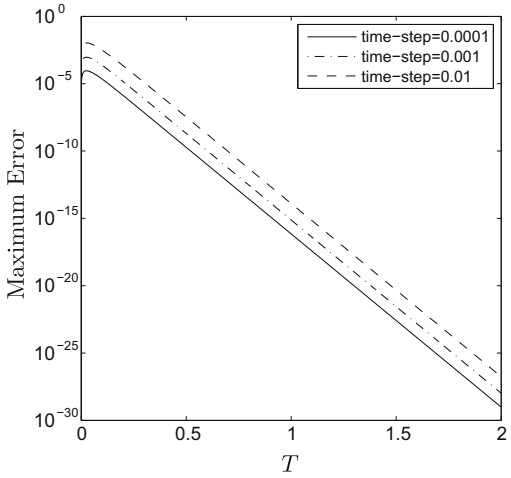

Fig. 12. Maximum error from a full three-dimensional simulation of the Heat Equation for fixed $h = \frac{1}{60}$ and various time-steps demonstrating first-order temporal accuracy and stability well above typical stability limits. The PDE domain consists of the region contained within a unit cube but outside the concentric sphere of radius $r = 0.125$.

To demonstrate the computational speed of the method, and its essentially linear cost, in Fig. 11 we present the computational time required by the algorithm per time-step as a function of the number of spatial unknowns. (Here we used our C implementation of the algorithm, compiled with gcc, on a single 2.33 GHz processing core.) We note that the algorithm requires approximately 1 s per time step with one million spatial unknowns, where each time-step corresponds to a complete pass in both spatial directions. To provide an indication with regards to memory usage, on the other hand, we mention that, as implemented, the code requires approximately 90 MB of memory for a spatial discretization of 4 million unknowns.

We now turn to a three-dimensional example for the Heat Equation. The PDE is posed on the domain consisting of the complement in the unit cube $(0,1)^3$ of the sphere centered at (1/2,1/2,1/2) of radius $r = 1/4$; we use the FC-AD solver described in Section 3.2, for which use of projection degree $m \leqslant 5$ (Eq. (12)) results in unconditional stability. We use $m = 5$ and thus obtain a sixth-order spatially accuracy scheme and first-order accuracy in time (see Eq. (44)), noting, once again, that Richardson extrapolation can be used to increase temporal accuracy; see [10]. The FC-AD was applied to the Heat Equation with right-hand side and boundary conditions chosen so that the exact solution is given by

$$u(x,y,z,t) = e^{-3\pi^2 t}\sin(\pi x)\sin(\pi y)\sin(\pi z). \tag{83}$$

The spatial resolution was fixed at $h = \frac{1}{60}$ and the computation repeated for various time-steps. The results are shown in Fig. 12 demonstrating the first-order time accuracy as well as stability well beyond typical stability limits.

Table 1
Computational times required to produce various accuracies by means of an explicit second-order finite-difference solver for the Heat Equation in a square domain (see Eq. (84)). Computations performed on a 3.4 GHz Pentium D processor.

Accuracy (%)	h	Δt	Comp. time (s)
10	1.52×10^{-2}	5.7×10^{-5}	33
1	4.9×10^{-3}	6.1×10^{-6}	2940
0.1	$<1.67 \times 10^{-3}$	$<7.0 \times 10^{-7}$	>213000

Table 2
Computational times required to produce various accuracies by means of the FC-AD algorithm for the Heat Equation in a square domain (see Eq. (84)). Computations performed on a 3.4 GHz Pentium D processor.

Accuracy (%)	h	Δt	Comp. time (s)
10	7.9×10^{-3}	7.3×10^{-3}	3
1	4.6×10^{-3}	2.3×10^{-3}	24
0.1	2.9×10^{-3}	7.0×10^{-4}	212

Table 3
Computational results for the FC-AD algorithm applied to the Heat Equation over the domain bounded by $(2x-1)^4 + (2y-1)^4 = 1$ using the parameters from Table 2. Performed on a 3.4 GHz Pentium D processor. These results show only minor variations in computational time and accuracy versus those produced by the FC-AD algorithm for the square domain.

Accuracy (%)	h	Δt	Comp. time (s)
9.3	7.9×10^{-3}	7.3×10^{-3}	3.2
0.852	4.6×10^{-3}	2.3×10^{-3}	25
0.0793	2.9×10^{-3}	7.0×10^{-4}	220

6.2. Comparison with finite-difference approaches

Use of finite difference or Finite Element methods for the solution of diffusive equations entails significant difficulties: conditional stability in explicit schemes and large computational cost per time-step in implicit methods. To demonstrate the advantages inherent in the FC-AD schemes we first compare this approach with a finite-difference solver for a parabolic problem in a 2D square domain. (It is important to note here that the requirements of explicit methods can become significantly more taxing for non-rectangular geometries than they are for simple rectangular domains, as for non-rectangular regions the minimum distance from a discretization point to the domain boundary can be arbitrarily small, thus triggering challenging stability constraints.)

In our first comparison we consider a problem for the Heat Equation with $k = 1$ and solution given by

$$u(x, y, t) = \cos(20\pi(2t - x + y)), \tag{84}$$

representing some sort of a thermal cycle in the unit square $(0, 1)^2$. A spatially second-order accurate explicit finite-difference method was used to solve this problem to accuracies of 10% and 1%, and an attempt was made to reduce the error to 0.1%. The time-step $\Delta t = h^2/4$ was chosen to maintain stability in the finite-difference algorithm. The required spatial resolutions are shown in Table 1 along with a lower bound for the computational cost required to produce a 0.1% error. The results were obtained on a single processing core of a 3.4 GHz Pentium D processor.

The same PDE problem was also solved using the FC-AD algorithm with accuracies of 10%, 1%, and 0.1%, and the corresponding timings, on the same hardware as used to produce the results in Table 1, are shown in Table 2. Notice the dramatic difference in computational cost, which results, mainly, from corresponding large reductions in the number of required time-steps. In fact, a time-step of the full FC-AD algorithm was approximately 3–5 times slower than a single step of the explicit finite-difference scheme on equivalent sized meshes. In view of the reduced number of time-steps it requires, the FC-AD algorithm was significantly faster even at the 10% error level. The computations in Table 2 were repeated with the FC-AD algorithm on the non-rectangular domain defined by $(2x-1)^4 + (2y-1)^4 = 1$ with the same spatial and temporal resolutions. The results of this experiment are presented in Table 3: the performance of the FC-AD approach for this geometry is actually (slightly) better than that obtained for the rectangular geometry considered in Table 2.

For an additional comparison, an implicit finite-difference second-order accurate scheme in both time and space, (with Crank–Nicolson time-stepping) was also applied to this problem. The Conjugate Gradient (CG) algorithm without preconditioning was used to solve the linear system at each time-step using the solution at the previous time-step as the initial guess. The tolerance of the CG algorithm was chosen to be as large as possible while still maintaining the desired accuracy level. It was found that for $h = 1.52 \times 10^{-2}$, the best performance was produced with a time-step about 4.4 times larger than that

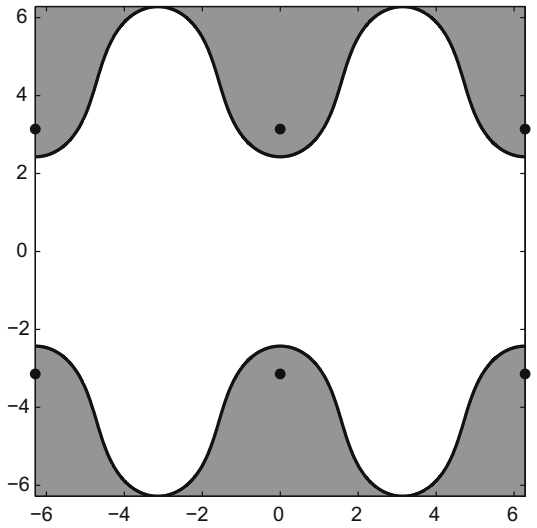

Fig. 13. Domain used for a demonstration of Laplace-Equation FC-AD solver. This domain is contained in the strip $-2\pi \leqslant x \leqslant 2\pi$ and is bounded by the curves $\pm\Psi(x,y) = \log\{\cosh(3\pi) - 1\} - \log\{\cosh(\pi) - 1\}$. The black dots denote the locations of the singularities of the function in (85).

required for the explicit case, but 9 CG iterations were needed on average per time step resulting in approximately double the computational time. By decreasing h to 1.25×10^{-2}, the number of time-steps required for 10% accuracy were further reduced by another factor of 4 and only 8 CG iterations were required. Thus with these parameters, 10% accuracy was reached in approximately 60% of the total time of the explicit FDM which is still considerably more than was required for the FC-AD. We note that had our test problem approached a steady state solution, the benefits of implicit finite-difference scheme with a CG iterative solver would have been more pronounced. For higher accuracies, we were not able to obtain parameters for which the implicit CG method results in an improvement over the computational cost of the explicit version: even though the implicit method allows for use of significantly larger time-steps, large computing times result from the large number of CG iterations required.

In this test we used an un-preconditioned CG iteration for the iterative solver above; an alternative could be considered in which a preconditioned CG solver would be used instead. Without entering in full detail into this vast research area, we note from a recent review [69, p. 425] concerning some of the highest quality preconditioning approaches available, namely incomplete factorization methods, that "Notwithstanding their popularity, incomplete factorization methods have their limitations. These include potential instabilities, difficulty of parallelization, and lack of algorithmic scalability." Additionally, such preconditioning approaches typically entail enormous memory requirements—of the order of a non-trivial percentage of the memory required by a full Cholesky decomposition; see [69, p. 436]. In any case, we note that, for 1% error, a preconditioned CG iteration for the finite-difference method would have to improve the computing time by at least two orders of magnitude over that resulting from the un-preconditioned CG scheme in order to obtain speeds comparable to those obtained by the FC-AD. In view of our example above and taking into account the memory and stability issues arising in preconditioned iterations, it appears that, at least for the types of problems under consideration, the FC-AD algorithm provides a highly competitive alternative to its finite-difference and finite-element counterparts.

6.3. FC-AD Poisson solver

A significantly challenging problem for spectral embedding methods, put forward in [8], provides an interesting test case for our FC-AD Poisson solver. The solution of this is problem is given by

$$\Psi(x,y) = \log\{\cosh(y-\pi) - \cos(x)\} - \log\{\cosh(y+\pi) - \cos(x)\}, \tag{85}$$

and the PDE domain is the region bounded by the curves defined the equations

$$\Psi(x,y) = \pm C, \tag{86}$$

and the lines $-2\pi \leqslant x \leqslant 2\pi$, where $C = \log\{\cosh(3\pi) + 1\} - \log\{\cosh(\pi) + 1\}$. This domain is depicted in Fig. 13; the black dots in the figure mark the location of the singularities in the analytic continuation of the solution, just outside the PDE domain. These singularities, which prevent convergence of a spectral series expansion over the complete rectangle, present a challenging configuration for spectral methods [8]—since convergence of a spectral series within the actual PDE domain often implies convergence over the complete rectangle. Fig. 14 displays numerical results produced by our FC-AD solver, with Dirichlet boundary data given by (85). In order to demonstrate the number of iterations required to reach a given accuracy, a

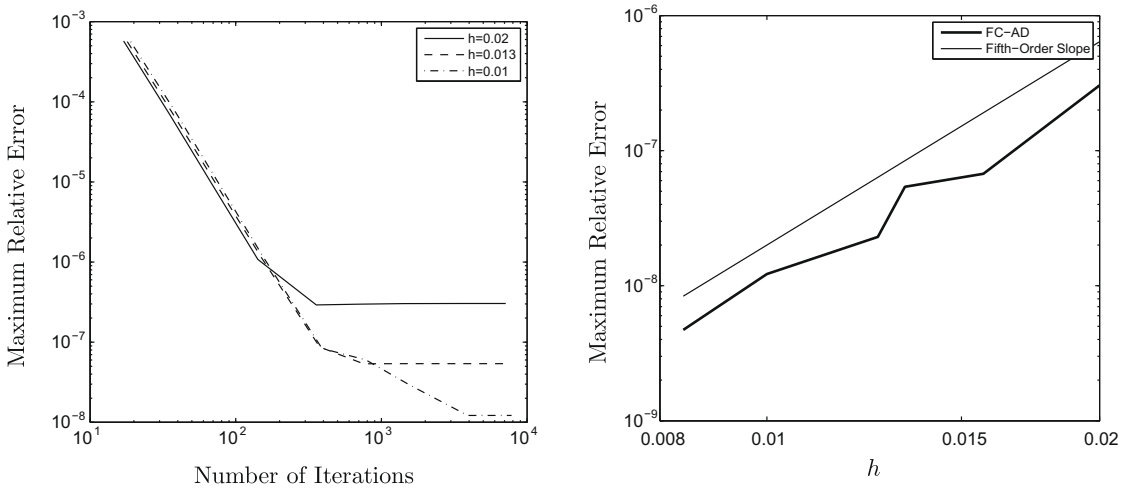

Fig. 14. Convergence results for the Laplace Equation with Dirichlet boundary data imposed on the boundary of the region shown in Fig. 13. The exact solution is given by Eq. (85). Results shown for three mesh spacings $h = 0.02, h = 0.013, h = 0.01$ (which resulted in 274,068, 617,399, and 1,098,396 unknowns, respectively) as a function of the number of iterations are presented on the left figure. The expected fifth-order convergence of the FC-AD algorithm is demonstrated in the right figure.

variety of values of the parameter ε (see Section 3.3) were used for each of three mesh spacings. For each one of three different meshes we used the values $\varepsilon = 0.2, 0.1, 0.05, 0.025, 0.01, 0.005, 0.0025, 0.001$, and 0.0005. For each such ε a value N_I of iterations given by Eq. (53) was used; note that Eq. (53) is a one-to-one relationship between ε and N_I for each h. The left portion of Fig. 14 displays the maximum errors produced by our algorithm over the spatial grid for each value of ε and each mesh as a function of N_I, each data point was obtained, of course, using the corresponding value of ε. Clearly, use of the strategy introduced in Section 3.3 produces rapidly decreasing errors as ε is decreased. Note that with only about 150 iterations, the solution was obtained with very high accuracies for each mesh size in spite of the large meshes used: the finest mesh considered contains over one-million unknowns. This example clearly demonstrates the usefulness of the iteration parameters presented in Section 3.3; in particular we note that the number of iterations needed to reach a given accuracy is nearly independent of the number of unknowns. Additionally the smallest errors obtained for a given mesh size h using the parameters ε listed above are shown, as a function of h, on the right portion of Fig. 14—demonstrating the fifth-order convergence of the method.

7. Conclusions

We have introduced a fast, high-order-accurate approach for numerical solution of Partial Differential Equations on general spatial domains, and we demonstrated its applicability, with unconditional stability, for the solution of the Heat and Poisson equations; applications to a number of other PDE problems are envisioned, see e.g. [10] and a related discussion in Section 1. The methodology combines the splitting of the classical ADI scheme with a new, fast version of the Fourier continuation method introduced previously for the resolution of the Gibbs phenomenon. Running at FFT speeds, our Heat and Poisson FC-AD algorithms were demonstrated to yield highly accurate solutions for general domains at a computational cost that scales nearly linearly with the number of unknowns; the FC-AD memory requirements, that are proportional to the number of unknowns, in turn, are extremely low. In conjunction with the FC-AD's unconditional stability and high-order accuracy (see Section 1 for a brief discussion in these regards), these properties allow for solution of large problems in short computational times—even in a single processing core of a present-day PC. Delivering fifth-order of spatial accuracy, the FC-AD algorithms are the only unconditionally stable alternating-direction schemes for general spatial domains that give rise to spatial accuracies of order higher than one.

Acknowledgments

We gratefully acknowledge support by the Air Force Office of Scientific Research and the National Science Foundation.

References

[1] D. Peaceman, H. Rachford Jr., The numerical solution of parabolic and elliptic differential equations, J. Soc. Ind. Appl. Math. 3 (1) (1955) 28–41.
[2] J. Douglas Jr., Alternating direction methods for three space variables, Numer. Math. 4 (1962) 41–63.
[3] J. Douglas Jr., C. Pearcy, On convergence of alternating direction procedures in the presence of singular operators, Numer. Math. 5 (1963) 175–184.
[4] J. Douglas Jr., J. Gunn, A general formulation of alternating direction methods. Part I. Parabolic and hyperbolic problems, Numer. Math. 6 (1964) 428–453.

[5] J. Douglas Jr., H. Rachford Jr., On the numerical solution of heat conduction problems in two and three space variables, T. Am. Math. Soc. 82 (2) (1956) 421–439.
[6] O. Bruno, Fast, high-order, high-frequency integral methods for computational acoustics and electromagnetics, in: M. Ainsworth, P. Davies, D. Duncan, P. Martin, B. Rynne (Eds.), Topics in Computational Wave Propagation Direct and Inverse Problems Series, Lecture Notes in Computational Science and Engineering, vol. 31, 2003, pp. 43–82.
[7] O. Bruno, Y. Han, M. Pohlman, Accurate, high-order representation of complex three-dimensional surfaces via Fourier-continuation analysis, J. Comput. Phys. 227 (2007) 1094–1125.
[8] J. Boyd, A comparison of numerical algorithms for Fourier extension of the first, second, and third kinds, J. Comput. Phys. 178 (2002) 118–160.
[9] K. Morton, D. Mayers, Numerical Solution of Partial Differential Equations, second ed., Cambridge University Press, Cambridge, 2005.
[10] M. Lyon, O. Bruno, High-order unconditionally-stable FC-AD for general two-dimensional smooth domains II: spatial accuracy, stability, and the Wave Equation, submitted for publication.
[11] J. Strikwerda, Finite Difference Schemes and Partial Differential Equations, second ed., SIAM, Philadelphia, 2004.
[12] T. Namiki, A new FDTD algorithm based on alternating-direction implicit method, IEEE Trans. Microw. Theory 47 (10) (1999) 2003–2007.
[13] M. Lees, Alternating direction methods for hyperbolic differential equations, J. Soc. Ind. Appl. Math. 10 (4) (1962) 610–616.
[14] R. Mohanty, M. Jain, An unconditionally stable alternating direction implicit scheme for the two space dimensional linear hyperbolic equation, Numer. Meth. Part. Differ. Equat. 17 (2001) 684–688.
[15] F. Zheng, Z. Chen, J. Zhang, A finite-difference time-domain method without the Courant stability conditions, IEEE Microw. Guided Wave Lett. 9 (11) (1999) 441–443.
[16] A. Averbuch, L. Vozovoi, Two-dimensional parallel solver for the solution of Navier–Stokes equations with constant and variable coefficients using ADI on cells, Parallel Comput. 24 (1998) 673–699.
[17] K. Badcock, B. Richards, Implicit time-stepping methods for the Navier–Stokes equations, AIAA J. 34 (3) (1996) 555–559.
[18] O. Widlund, On the effects of scaling of the Peaceman–Rachford method, Math. Comput. 25 (113) (1971) 33–41.
[19] O. Widlund, On the rate of convergence of an alternating direction implicit method in a noncommutative case, Math. Comput. 20 (96) (1966) 500–515.
[20] W. Hundsdorfer, J.G. Verwer, Stability and convergence of the Peaceman–Rachford ADI method for initial-boundary value problems, Math. Comput. 53 (187) (1989) 81–101.
[21] J. Gunn, On the two-stage iterative method of Douglas for mildly nonlinear elliptic difference equations, Numer. Math. 6 (1964) 243–249.
[22] B. Fornberg, J. Zuev, J. Lee, Stability and accuracy of time-extrapolated ADI-FDTD methods for solving wave equations, J. Comput. Appl. Math. 200 (2007) 178–192.
[23] G. Fairweather, A. Mitchell, A high accuracy alternating direction method for the wave equation, J. Inst. Math. Appl. 1 (1965) 309–316.
[24] A. Mitchell, G. Fairweather, Improved forms of the alternatin direction methods of Douglas, Peaceman and Rachford for solving parabolic and elliptic equations, Numer. Math. 6 (1964) 285–292.
[25] Z. Tian, Y. Ge, A fourth-order compact ADI method for solving two-dimensional unsteady convection–diffusion problems, J. Comput. Appl. Math. 198 (2007) 268–286.
[26] D. You, A high-order Padé ADI method for unsteady convection–diffusion equations, J. Comput. Phys. 214 (2006) 1–11.
[27] N. Kantartzis, T. Zygiridis, T. Tsiboukis, An unconditionally stable higher order ADI-FDTD technique for the dispersionless analysis of generalized 3-D EMC structures, IEEE Trans. Magn. 40 (2) (2004) 1436–1439.
[28] W. Dyksen, Tensor product generalized ADI methods for separable elliptic problems, SIAM J. Numer. Anal. 24 (1) (1987) 59–75.
[29] R. Lynch, J. Rice, D. Thomas, Direct solution of partial difference equations by tensor product methods, Numer. Math. 6 (1964) 185–199.
[30] M. Carpenter, D. Gottlieb, A. Saul, The stability of numerical boundary treatments for compact high-order finite-difference schemes, J. Comput. Phys. 108 (1993) 272–295.
[31] S. Abarbanel, A. Chertock, Strict stability of high-order compact implicit finite-difference schemes: the role of boundary conditions for hyperbolic PDEs, I, J. Comput. Phys. 160 (2000) 42–66.
[32] S. Abarbanel, A. Chertock, A. Yefet, Strict stability of high-order compact implicit finite-difference schemes: the role of boundary conditions for hyperbolic PDEs, II, J. Comput. Phys. 160 (2000) 67–87.
[33] S. Karaa, J. Zhang, High order ADI method for solving unsteady convection–diffusion problems, J. Comput. Phys. 198 (2004) 1–9.
[34] C. Clavero, J. Gracia, J. Jorge, A uniformly convergent alternating direction hodie finite difference scheme for 2d time-dependent convection–diffusion problems, IMA.J. Numer. Anal. 26 (2006) 155–172.
[35] B. Alpert, L. Greengard, T. Hagstrom, An integral evolution formula for the wave equation, J. Comput. Phys. 162 (2) (2000) 536–543.
[36] S. Abarbanel, A. Ditkowski, A. Yefet, Bounded error schemes for the wave equation on complex domains, J. Sci. Comput. 26 (2006) 67–81.
[37] S. Abarbanel, A. Ditkowski, Asymptotically stable fourth-order accurate schemes for the diffusion equation on complex shapes, J. Comput. Phys. 133 (1997) 279–288.
[38] M. Elghaoui, R. Pasquetti, A spectral embedding method applied to the advection–diffusion equation, J. Comput. Phys. 125 (1996) 464–476.
[39] B. Lombard, J. Piraux, C. Gelis, J. Virieux, Free and smooth boundaries in 2-d finite-difference schemes for transient elastic waves, Geophys. J. Int. 172 (2008) 252–261.
[40] B. Griffith, C. Peskin, On the order of accuracy of the immersed boundary method: higher order convergence rates for suffciently smooth problems, J. Comput. Phys. 208 (2005) 75–105.
[41] M. Lai, C. Peskin, An immersed boundary method with formal second-order accuracy and reduced numerical viscosity, J. Comput. Phys. 160 (2000) 705–719.
[42] C. Peskin, The immersed boundary method, Acta Numer. 11 (2002) 479–517.
[43] H. Johansen, P. Colella, A cartesian grid embedded boundary method for Poisson's equation on irregular domains, J. Comput. Phys. 147 (1998) 60–85.
[44] H. Kreiss, N. Petersson, A second order accurate embedded boundary method for the wave equation with dirichlet data, SIAM J. Sci. Comput. 27 (2006) 1141–1167.
[45] A. Ditkowski, K. Dridi, J. Hesthaven, Convergent Cartesian grid, methods for maxwell's equations in complex geometries, J. Comput. Phys. 170 (2001) 39–80.
[46] J. Li, L. Greengard, High order marching schemes for the wave equation in complex geometry, J. Comput. Phys. 198 (2004) 295–309.
[47] F. Sabetghadam, S. Sharafatmandjoor, F. Norouzi, Fourier spectral embedded boundary solution of the Poisson's and Laplace equations with dirichlet boundary conditions, J. Comput. Phys. 228 (2009) 55–74.
[48] K. Eckhoff, On a high order numerical method for solving Partial Differential Equations in complex geometries, J. Sci. Comput. 12 (2) (1997) 119–138.
[49] O. Næss, K. Eckhoff, A modified Fourier Galerkin method for the Poisson and Helmholtz equations, J. Sci. Comput. 17 (2002) 529–539.
[50] G. Zhao, Q. Liu, The unconditionally stable pseudospectral time-domain (PSTD) method, IEEE Microw. Wireless Comp. Lett. 13 (11) (2003) 475–477.
[51] B. Bialecki, R. Fernandes, An alternating-direction implicit orthogonal spline collocation scheme for nonlinear parabolic problems on rectangular polygons, SIAM J. Sci. Comput. 28 (3) (2006) 1054–1077.
[52] K. Cooper, P. Prenter, Alternating direction for separable elliptic Partial Differential Equations, SIAM J. Numer. Anal. 28 (3) (1991) 711–727.
[53] K. Cooper, K. McArthur, P. Prenter, Alternating direction collocation for irregular regions, Numer. Meth. Part. Differ. Equat. 12 (1996) 147–159.
[54] C. Canuto, M. Hussaini, A. Quarteroni, T. Zang, Spectral Methods Fundamentals in Single Domains, Scientific Computation, Springer, Berlin, 2006.
[55] C. Canuto, M. Hussaini, A. Quarteroni, T. Zang, Spectral Methods: Evolution to Complex Geometries and Applications to Fluid Dynamics, Scientific Computation, Springer, Berlin, 2007.
[56] D. Gottlieb, C.-W. Shu, On the Gibbs phenomenon and its resolution, SIAM Rev. 39 (4) (1997) 644–668.
[57] A. Gelb, J. Tanner, Robust reprojection methods for the resolution of the Gibbs phenomenon, Appl. Comput. Harmon. Anal. 20 (2006) 3–25.

[58] J.P. Boyd, J.R. Ong, Exponentially-convergent strategies for defeating the Runge Phenomenon for the approximation of non-periodic functions, part I: single-interval schemes, Commun. Comput. Phys. 5 (2-4) (2009) 484–497.
[59] M. Frigo, S. Johnson, The design and implementation of FFTW3, Proc. IEEE 93 (2) (2005) 216–231.
[60] Å. Björck, Numerical Methods for Least Squares Problems, Society for Industrial and Applied Mathematics, Philadelphia, 1996.
[61] R. Barnard, G. Dahlquist, K. Pearce, L. Reichel, K. Richards, Gram polynomials and the kummer function, J. Approx. Theory 94 (1998) 128–143.
[62] J. Demmel, Applied Numerical Linear Algebra, SIAM, Philadelphia, 1997.
[63] M. Lyon, High-order unconditionally-stable FC-AD PDE solvers for general domains, Ph.D. thesis, California Institute of Technology, 2009.
[64] J. Weideman, S. Reddy, A MATLAB differentiation matrix suite, ACM T. Math. Software 26(4) (2000) 465–519. url http://doi.acm.org/10.1145/365723.365727.
[65] B.R. StoerJ, Introduction to Numerical Analysis, second ed., Springer, New York, 2004.
[66] E. Wachspress, Iterative Solution of Elliptic Systems and Application to the Neutron Diffusion Equations of Reactor Physics, Prentice-Hall, Englewood Cliffs, NJ, 1971.
[67] D. Young, Iterative Solution of Large Linear Systems, Academic Press, New York, 1971.
[68] A. Quarteroni, S. Riccardo, F. Saleri, Numerical Mathematics, Texts in Applied Mathematics, Springer, New York, 2000.
[69] M. Benzi, Preconditioning techniques for large linear systems: a survey, J. Comput. Phys. 182 (2002) 418–477.