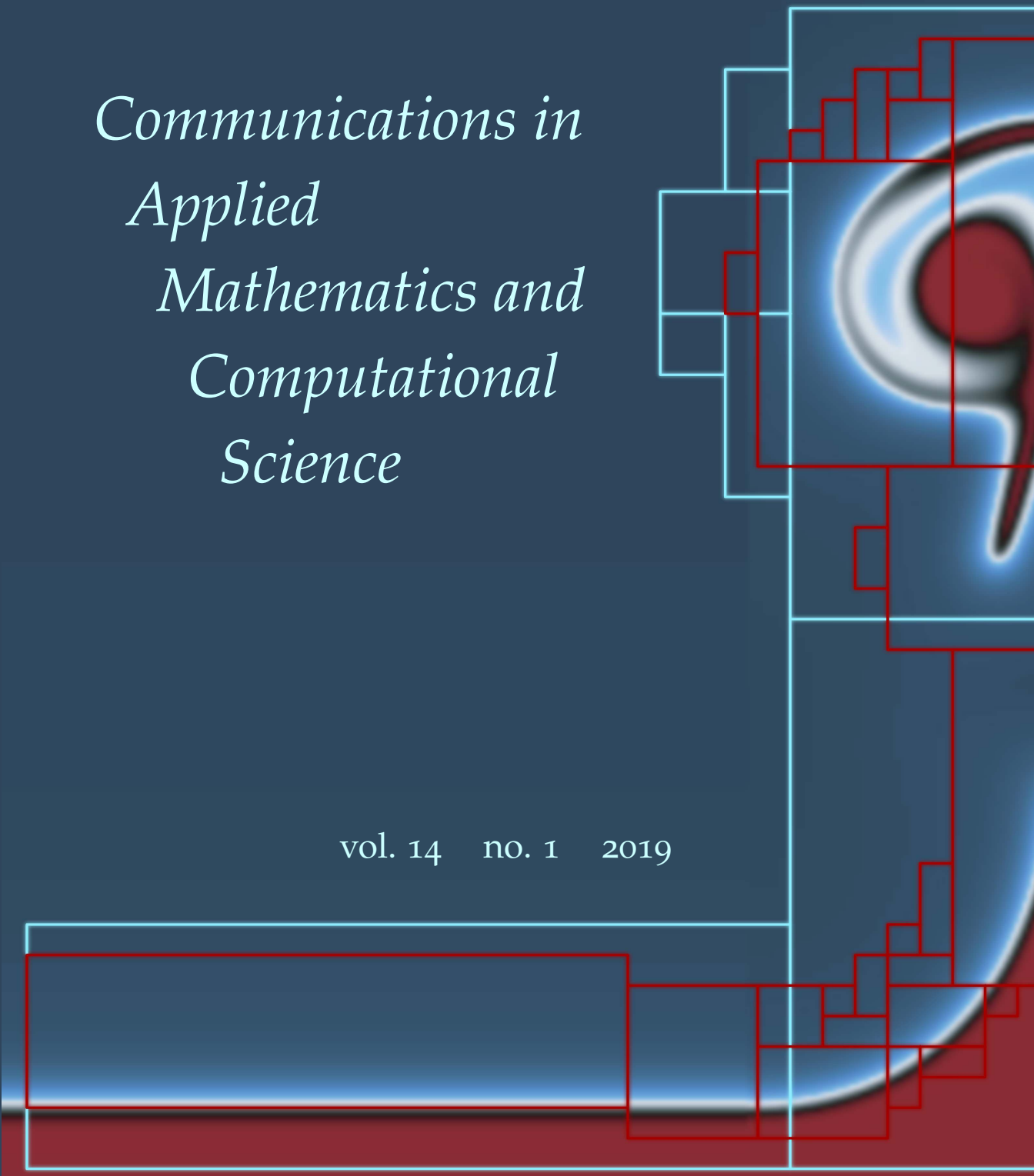
Communications in Applied Mathematics and Computational Science



vol. 14 no. 1 2019

msp

Communications in Applied Mathematics and Computational Science

msp.org/camcos

See inside back cover or msp.org/camcos for submission instructions.

The subscription price for 2019 is US $105/year for the electronic version, and $155/year (+$15, if shipping outside the US) for print and electronic. Subscriptions, requests for back issues from the last three years and changes of subscriber address should be sent to MSP.

Communications in Applied Mathematics and Computational Science (ISSN 2157-5452 electronic, 1559-3940 printed) at Mathematical Sciences Publishers, 798 Evans Hall #3840, c/o University of California, Berkeley, CA 94720-3840, is published continuously online. Periodical rate postage paid at Berkeley, CA 94704, and additional mailing offices.

CAMCoS peer review and production are managed by EditFLOW® from MSP.

PUBLISHED BY

mathematical sciences publishers

nonprofit scientific publishing

http://msp.org/

COMM. APP. MATH. AND COMP. SCI.
Vol. 14, No. 1, 2019
dx.doi.org/10.2140/camcos.2019.14.1

msp

COMPUTATION OF VOLUME POTENTIALS ON STRUCTURED GRIDS WITH THE METHOD OF LOCAL CORRECTIONS

CHRIS KAVOUKLIS AND PHILLIP COLELLA

We present a new version of the method of local corrections (MLC) of McCorquodale, Colella, Balls, and Baden (2007), a multilevel, low-communication, noniterative domain decomposition algorithm for the numerical solution of the free space Poisson's equation in three dimensions on locally structured grids. In this method, the field is computed as a linear superposition of local fields induced by charges on rectangular patches of size $O(1)$ mesh points, with the global coupling represented by a coarse-grid solution using a right-hand side computed from the local solutions. In the present method, the local convolutions are further decomposed into a short-range contribution computed by convolution with the discrete Green's function for a Q-th-order accurate finite difference approximation to the Laplacian with the full right-hand side on the patch, combined with a longer-range component that is the field induced by the terms up to order $P-1$ of the Legendre expansion of the charge over the patch. This leads to a method with a solution error that has an asymptotic bound of $O(h^P)+O(h^Q)+O(\epsilon h^2)+O(\epsilon)$, where h is the mesh spacing and ϵ is the max norm of the charge times a rapidly decaying function of the radius of the support of the local solutions scaled by h. The bound $O(\epsilon)$ is essentially the error of the global potential computed on the coarsest grid in the hierarchy. Thus, we have eliminated the low-order accuracy of the original method (which corresponds to $P=1$ in the present method) for smooth solutions, while keeping the computational cost per patch nearly the same as that of the original method. Specifically, in addition to the local solves of the original method we only have to compute and communicate the expansion coefficients of local expansions (that is, for instance, 20 scalars per patch for $P=4$). Several numerical examples are presented to illustrate the new method and demonstrate its convergence properties.

MSC2010: 65N06, 65N12, 65N15, 68W10.

Keywords: Poisson solver, method of local corrections, Mehrstellen stencils, domain decomposition, parallel solvers.

1. Introduction

We are interested in solving Poisson's equation with infinite domain boundary conditions in three dimensions, that is

$$\begin{aligned} \Delta\phi \equiv \frac{\partial^2\phi}{\partial x^2} + \frac{\partial^2\phi}{\partial y^2} + \frac{\partial^2\phi}{\partial z^2} &= f \qquad\qquad \text{in } \mathbb{R}^3, \\ \phi(\boldsymbol{x}) = -\frac{1}{4\pi\|\boldsymbol{x}\|}\int_{\mathbb{R}^3} f(\boldsymbol{y})\,d\boldsymbol{y} + o\left(\frac{1}{\|\boldsymbol{x}\|}\right), &\quad \|\boldsymbol{x}\| \to \infty, \end{aligned} \tag{1}$$

where f is a function with bounded support and by $\|\cdot\|$ we denote the Euclidean norm. It is well known that problem (1) has a solution if f is Hölder continuous and has compact support Ω [12]. Furthermore, the solution of (1) is unique by means of a maximum principle argument for harmonic functions and is given as a convolution of the data with the three-dimensional infinite domain Green's function [10]

$$\phi(\boldsymbol{x}) = \int_\Omega G(\boldsymbol{x}-\boldsymbol{y})f(\boldsymbol{y})\,d\boldsymbol{y} \equiv (G * f)(\boldsymbol{x}), \quad G(z) = -\frac{1}{4\pi\|z\|}. \tag{2}$$

In addition, if $\Omega \subset B(\boldsymbol{x}_0, R)$, where $B(\boldsymbol{x}_0, R)$ is the closed ball of radius R centered at point $\boldsymbol{x}_0$, then ϕ is harmonic in $\mathbb{R}^3 \setminus B(\boldsymbol{x}_0, R)$ and hence real analytic. By differentiating (2), we find that the derivatives of the potential are rapidly decaying functions of the form

$$(\nabla^{\boldsymbol{p}}\phi)(\boldsymbol{x}) = O\left(\left(\frac{1}{\|\boldsymbol{x}-\boldsymbol{x}_0\|}\right)^{\|\boldsymbol{p}\|_1+1} R^3\|f\|_\infty\right). \tag{3}$$

This suggests a domain decomposition strategy, in which the contribution to the fields on each local domain is computed independently and the nonlocal coupling is computed using a reduced number of computational degrees of freedom. This approach has been exploited for particle methods with the right-hand side in (1) given by $f(\boldsymbol{x}) = \sum_i q_i\delta(\boldsymbol{x}-\boldsymbol{x}_i)$. For instance, we mention the Barnes–Hut algorithm [6], the fast multipole method (FMM) [13; 7; 14], and the method of local corrections (MLC) [3; 1; 2]. The aforementioned particle algorithms have been modified to handle gridded data. In [20] the approximate solution of the Poisson problem is given as a discrete convolution of the discrete Green's function with the charge [15] and is computed efficiently by combining the fast Fourier transform (FFT) with interpolation of the kernel. This strategy is substantially accelerated within an FMM setting but has not been extended to support multiresolution calculations. A very attractive kernel-independent method is discussed in [27], for the case of a smooth charge distribution that is represented on a uniform mesh. The kernel is truncated on a sphere that encloses the charge so that its Fourier transform is C^∞. This allows for a fast and accurate computation of the potential via the trapezoid rule and the FFT transform; however, the method is not readily applicable to adaptive mesh refinement

(AMR) hierarchies. The first kernel-independent, adaptive volume-FMM method has been presented in [19]. The integral in (2) is computed directly with numerical quadrature, and local charges are approximated with polynomials. The analogs of the multipole and local expansions in the original FMM method are convolutions with equivalent source densities defined on auxiliary surfaces that encompass octree boxes. The method can handle nonuniform sources, and a Chebyshev grid may be required to achieve high-order accuracy. A highly optimized parallel implementation is discussed in [21; 22]. For a comprehensive review that includes benchmark studies of the FFT, FMM, and multigrid methods, we refer to [11]. The MLC method [25] relies upon a localization approach that takes advantage of the rapid decay in truncation error of compact finite difference Laplace operators. Further, it is more compatible with traditional AMR. As such, it can be coupled with numerical schemes that require solving Poisson's equation on nested locally refined grids, for instance adaptive projection methods for computational fluid dynamics. It should also be noted that MLC exhibits a good balance between computation and communication, which is essential for simulations on the emerging exascale platforms.

The present work is based on the extension of the method of local corrections to structured grid data described in [5; 4; 25]. In this approach, the support of the right-hand side is discretized with a rectangular grid, which is decomposed into a set of cubic patches. For two levels the method proceeds in three steps: (i) a loop over the fine disjoint patches and the computation of local potentials induced by the charge restricted to those patches on sufficiently large extensions of their support (downward pass), (ii) a global coarse-grid Poisson solve with a right-hand side computed by applying the coarse-grid Laplacian to the local potentials of step (i), and (iii) a correction of the local solutions computed in step (i) on the boundaries of the fine disjoint patches based on interpolating the global coarse solution from which the contributions from the local solutions have been subtracted (upward pass). These boundary conditions are propagated into the interior of the patches by performing Dirichlet solves on each patch. This can be generalized by replacing the global coarse solution in (ii) by a recursive call to MLC, or by replacing uniform grids at each level covering the entire domain by nested block-structured locally refined grids. The local volume potentials are computed using a high-order finite difference approximation to the Laplacian, combined with an extension to three dimensions of the James–Lackner algorithm [17; 18] for representing infinite domain boundary conditions. Furthermore, in order to make the nested refinement version of this algorithm practical, we require that $R = O(H) = O(h)$, where R is the radius (in max norm) of local patches, H the coarse mesh spacing, and h the fine-mesh spacing (i.e., a fixed number of points per patch and a fixed refinement ratio). In [25], the local field calculation in (i) was split into two contributions: one that represented the field induced by the complete charge distribution on a patch,

and a second corresponding to the monopole component of the charge. By using such a splitting, it is possible to obtain a convergent method by using a relatively large region for computing the monopole component only while keeping the overall computation and communications cost low. However, the convergence properties of the resulting method were erratic, and exhibited a large $O(h)$ solution error for smooth charge distributions that were well resolved on the fine grid.

The starting point for the present work is a new error analysis for the MLC algorithm that suggests a number of generalizations of the method that have better and more predictable convergence properties. For example, we replace the separate treatment of the monopole component of the charge on each patch by a similar treatment of a truncated expansion in Legendre polynomials of the charge distribution on each patch. Our error analysis predicts an $O(h^P)+O(h^Q)+O(\epsilon h^2)+O(\epsilon)$ solution error, where $P-1$ is the maximum degree of the polynomials in the Legendre expansions, and Q is the order of accuracy of the finite difference discretization used to compute the local potentials. This is consistent with the earlier results in [25] corresponding to $P=1$. The $O(\epsilon)$ term is a localization error, proportional to the max norm of the charge divided by a localization distance (measured in units of the number of coarse grid points across the patch) raised to the order of accuracy of the discretized Laplacian on harmonic functions. We also change the detailed approach to computing the local potentials, replacing the James–Lackner representation of the infinite domain boundary conditions in the calculation of the local potentials in step (i) with local discrete convolutions computed using FFTs via a variation on Hockney's domain-doubling method [16]. This leads to a conceptually simpler algorithm, and provides a compact numerical kernel on which to focus the effort of optimization.

In this paper, we focus on the design of the algorithm, including an error analysis of the method and calculations that demonstrate the error properties derived from that analysis. In a second paper [24], we will present performance and parallel scaling results on high-performance computing (HPC) platforms.

2. Mehrstellen discretization and finite difference localization

Notation. We denote by $D^h, \Omega^h, \ldots \subset \mathbb{Z}^3$ grids with grid spacing h of discrete points in physical space: $\{\boldsymbol{g}h : \boldsymbol{g} \in D^h\}$. Arrays of values defined over such sets will approximate functions on subsets of $\mathbb{R}^3$; i.e., if $\psi = \psi(\boldsymbol{x})$ is a function on $D \subset \mathbb{R}^3$, then $\psi^h[\boldsymbol{g}] \approx \psi(\boldsymbol{g}h)$. We denote operators on arrays over grids of mesh spacing h by $L^h, \Delta^h, \ldots$; $L^h(\phi^h) : D^h \to \mathbb{R}$. Such operators are also defined on functions of $\boldsymbol{x} \in \mathbb{R}^3$, and on arrays defined on finer grids $\phi^{h'}$, $h = Nh'$, $N \in \mathbb{N}_+$, by sampling: $L^h(\phi) \equiv L^h(\mathcal{I}^h(\phi))$, $\mathcal{I}^h(\phi)[\boldsymbol{g}] \equiv \phi(\boldsymbol{g}h)$, and $L^h(\phi^{h'}) \equiv L^h(\mathcal{I}^h(\phi^{h'}))$, $\mathcal{I}^h(\phi^{h'})[\boldsymbol{g}] \equiv \phi^{h'}[N\boldsymbol{g}]$.

For a rectangle $D = [\boldsymbol{l}, \boldsymbol{u}]$, defined by its lower-left and upper-right corners $\boldsymbol{l}, \boldsymbol{u} \in \mathbb{Z}^3$, we define two operators: a grid extension operator

$$\mathcal{G}(D, r) = [\boldsymbol{l} - (r, r, r), \boldsymbol{u} + (r, r, r)], \quad r \in \mathbb{Z},$$

and a grid coarsening operator

$$\mathcal{C}(D) = \left[\left\lfloor \frac{\boldsymbol{l}}{N_{\text{ref}}} \right\rfloor, \left\lceil \frac{\boldsymbol{u}}{N_{\text{ref}}} \right\rceil\right].$$

Throughout this paper, we will use $N_{\text{ref}} = 4$ for the refinement ratio between consecutive levels.

We begin our discussion presenting the finite difference discretizations of (1) that we will be using throughout this work and some of their properties that pertain to the method of local corrections. Specifically, we are employing Mehrstellen discretizations [8] (also referred to as compact finite difference discretizations) of the three-dimensional Laplace operator

$$(\Delta^h \phi^h)_{\boldsymbol{g}} = \sum_{\boldsymbol{s} \in [-s, s]^3} a_{\boldsymbol{s}} \phi^h_{\boldsymbol{g}+\boldsymbol{s}}, \quad a_{\boldsymbol{s}} \in \mathbb{R}. \tag{4}$$

If ϕ^h is defined on D^h, then $\Delta^h \phi^h$ is defined on $D^{h,s} \equiv \mathcal{G}(D^h, -s)$. The associated truncation error $\tau^h \equiv (\Delta^h - \Delta)(\phi) = -\Delta^h(\phi^h - \phi)$ for the Mehrstellen discrete Laplace operator is of the form

$$\tau^h(\phi) = C_2 h^2 \Delta(\Delta\phi) + \sum_{q'=2}^{q/2-1} h^{2q'} \mathcal{L}^{2q'}(\Delta\phi) + h^q L^{q+2}(\phi) + O(h^{q+2}), \tag{5}$$

where q is even and $\mathcal{L}^{2q'}$ and L^{q+2} are constant-coefficient differential operators that are homogeneous, i.e., for which all terms are derivatives of orders $2q'$ and $q+2$, respectively. For the two operators we will consider here, $C_2 = \frac{1}{12}$. In general, the truncation error is $O(h^2)$. However, if ϕ is harmonic in a neighborhood of $\boldsymbol{x}$,

$$\tau^h(\phi)(\boldsymbol{x}) = \Delta^h(\phi)(\boldsymbol{x}) = h^q L^{q+2}(\phi)(\boldsymbol{x}) + O(h^{q+2}). \tag{6}$$

In our numerical test cases we make use of the 19-point (L^h_{19}) and 27-point (L^h_{27}) Mehrstellen stencils [26] that are described in Appendix A, for which $q = 4$ and $q = 6$, respectively. In general, it is possible to define operators for which $s = \lfloor q/4 \rfloor$ for any even q, using higher-order Taylor expansions and repeated applications of the identity

$$\frac{\partial^{2r}\phi}{\partial x_d^{2r}} = \frac{\partial^{2r-2}}{\partial x_d^{2r-2}}(\Delta\phi) - \sum_{d' \neq d} \frac{\partial^{2r}}{x_{d'}^{2r-2} x_d^2}(\phi).$$

Since we are primarily concerned with solving the free-space problem, the corresponding discrete problem can be expressed formally as a discrete convolution

$$(G^h * f^h) = (\Delta^h)^{-1}(f^h), \quad (G^h * f^h)[\boldsymbol{g}] \equiv \sum_{\boldsymbol{g}' \in \mathbb{Z}^3} h^3 G^h[\boldsymbol{g} - \boldsymbol{g}'] f[\boldsymbol{g}']^h, \tag{7}$$

where the discrete Green's function $G^h[\boldsymbol{g}] = h^{-1} G^{h=1}[\boldsymbol{g}]$ satisfies

$$(\Delta^{h=1} G^{h=1})[\boldsymbol{g}] = \begin{cases} 1 & \text{if } \boldsymbol{g} = \boldsymbol{0}, \\ 0 & \text{otherwise} \end{cases} \tag{8}$$

and

$$G^{h=1}[\boldsymbol{g}] = \frac{1}{4\pi \|\boldsymbol{g}\|} + o\left(\frac{1}{\|\boldsymbol{g}\|}\right), \quad \|\boldsymbol{g}\| \to \infty.$$

We use these conditions to construct approximations to G^h numerically; see Appendix A. For any n, we have

$$\sum_{\boldsymbol{g} \in D} h^3 |G^h[\boldsymbol{g}]| \leq C, \quad C = C(nh), \ D \subseteq [-n, \ldots, n]^3,$$

from which it follows that convolution with G^h is max norm stable on bounded domains, i.e.,

$$\|G^h * f^h\|_\infty \leq C' \|f^h\|_\infty,$$
$$C' \text{ independent of } f, h, \ \operatorname{supp}(f^h) \subseteq \left[-\left\lfloor \frac{A}{h} \right\rfloor, \ldots, \left\lceil \frac{A}{h} \right\rceil \right]^3, \tag{9}$$

for any fixed $A > 0$.

The form of the truncation error (5) allows us to compute q-th-order accurate solutions to (1) by modifying the right-hand side, i.e.,

$$\Delta^h(\phi) = \tilde{f}^h + O(h^q), \tag{10}$$

$$\tilde{f}^h = f^h + \left(C_2 h^2 (\Delta(f))^h + \sum_{q'=2}^{q/2-1} h^{2q'} \mathcal{L}^{2q'}(f)^h \right), \tag{11}$$

and replacing the differential operators on the right-hand side with finite difference approximations. If only a fourth-order accurate solution is required, it suffices to use the first term, leading to a correction of a particularly simple form:

$$\phi = G^h * f^h + C_2 h^2 f^h + O(h^4). \tag{12}$$

In particular, the solution error $\epsilon^h = G^h * f^h - \phi = O(h^4)$ away from the support of f without any modification of f^h.

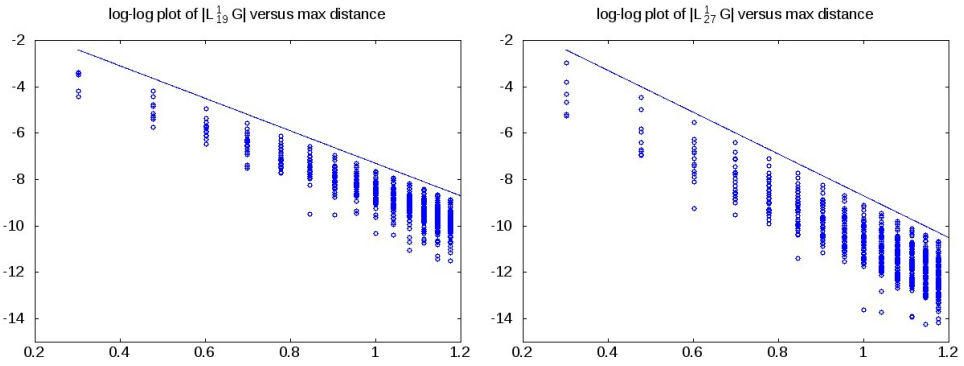


Figure 1. Scatter plots of $\log_{10}(|(\Delta^{h=1}\mathcal{S}^{h=1}(G))[\boldsymbol{g}]|)$ versus $\log_{10}(\|\boldsymbol{g}\|_\infty)$, $\boldsymbol{g} \in \mathbb{Z}^3$, at points away from the singularity of G for the L^h_{19} and L^h_{27} discrete Laplacians. The slopes of the lines depicted are -7 and -9 for the L^h_{19} and L^h_{27}, respectively.

Suppose that $\operatorname{supp}(f) \subset P_{\boldsymbol{c}}$, where $P_{\boldsymbol{c}} = \boldsymbol{c} + [-R, R]^3$ is a cube of radius R centered at point $\boldsymbol{c}$, and that the differential operator L^q is a linear combination of derivatives of order q. By differentiating (2), we have

$$[(L^q G) * f](\boldsymbol{x}) = O\left(\left(\frac{1}{R}\right)^{q-2} \frac{1}{\|\boldsymbol{x}/R - \boldsymbol{c}/R\|_\infty^{q+1}}\right)\|f\|_\infty. \tag{13}$$

In particular, away from the support of f, (5) becomes

$$\tau^h(f) = \Delta^h(G * f)(\boldsymbol{x}) = O\left(\left(\frac{h}{R}\right)^{q} \frac{1}{\|\boldsymbol{x}/R - \boldsymbol{c}/R\|_\infty^{q+3}}\right)\|f\|_\infty. \tag{14}$$

It is precisely this rapid decay of the truncation error, a consequence of the fact that the local potentials are harmonic away from the supports of the associated charges, that allows us to use a coarse mesh for the global coupling computation. In Figure 1, scatter plots of the truncation error for the case of a point charge located at the origin using the 19-point and 27-point Laplacians are depicted. The rapid decay of the truncation error in the far field and the faster decay with increasing q are evident. Using this localization property of the Mehrstellen operators, we can reduce the cost of computing the potential (2) induced by a localized charged distribution to the cost of computing the potential near the support of the charge, using the finite difference localization approach originally introduced in [23]. We assume that the support of f is contained in cube D of radius R centered at $\boldsymbol{c}$. First, let $\phi = G * f$ be the exact solution restricted on the extended cube D_β of radius βR, $\beta > 1$. Then we compute $\phi^H = G^H * F^H$ on Ω^H. The coarse right-hand side is defined by

$$F^H = \begin{cases} \Delta^H(\phi) & \text{on } D_\beta^{H,s} = \mathcal{G}(\mathcal{C}(D_\beta^h), -s), \\ 0 & \text{on } \Omega^H \setminus D_\beta^{H,s}. \end{cases} \tag{15}$$

Using (14), we have

$$\Delta^H(\phi^H - G*f) = \begin{cases} 0 & \text{on } D_\beta^{H,s}, \\ O((H/R)^q(1/(k+\beta)^{q+3})\|f\|_\infty) & \\ \qquad \text{on } \{\boldsymbol{g} : ((k+\beta)+1)R \geq \|\boldsymbol{g}H\|_1 \geq (k+\beta)R\}, \end{cases} \tag{16}$$

where $\boldsymbol{g} \in \mathbb{Z}^3$, $k \in \mathbb{N}$. One can decompose the annular region $\{\boldsymbol{g} : ((k+\beta)+1)R \geq \|\boldsymbol{g}H\|_1 \geq (k+\beta)R\}$ into $O((k+\beta)^2)$ rectangles, each of which has radius $\leq R$, leading to an analogous decomposition of the right-hand side of (16) into a sum of terms, each of which is supported on one such rectangle. Applying convolution with G^H to both sides of (16) represented in terms of such sums leads to a solution error given by

$$\begin{aligned} \phi^H - G*f &= \sum_{k=0}^{\infty} O\left(\left(\frac{H}{R}\right)^q \frac{1}{(k+\beta)^{q+3}}\|f\|_\infty\right) \\ &= O\left(\left(\frac{H}{R}\right)^q \frac{1}{\beta^q}\|f\|_\infty\right). \end{aligned} \tag{17}$$

Thus, the accuracy of the potential away from the support of the charge can be improved by decreasing the ratio H/R or, for fixed values of that ratio, by adjusting β or q. In any case, the error is only weakly dependent on f. In this context, we will refer to β as a *localization radius*. In addition, (17) is truly independent of whether the right-hand side is modified using the Mehrstellen correction (11). The MLC algorithm combines finite difference localization with domain decomposition into a collection of rectangular patches of size R to obtain a low-communication method for computing volume potentials. This is done in a way that generalizes to nested refinement on an arbitrary number of levels, with the domain at each level decomposed into patches having a fixed number of mesh points, independent of the level of refinement. This implies that H/R remains constant, which leads to (17) being an $O(1)$ error relative to the mesh spacing. Ultimately, that error is controlled by increasing β, combined with choosing a discretization with a larger q. However, the cost of computing the local convolution $G*f$ on $D_\beta^{H,s}$ scales like β^3. To reduce that cost, we introduce a second localization radius α, $\alpha < \beta$. On $D_\alpha^{H,s}$, we use the full convolution to compute F^H. In the remaining annular region, we use a reduced representation based on the field induced by the first few moments of the Legendre expansion of f, which is much less expensive to compute.

3. Method of local corrections: semidiscrete case

To clarify ideas, we discuss in this section a theoretical proxy for the fully discrete algorithm. We construct a function $\phi^{\mathrm{MLC}} : \Omega \to \mathbb{R}$ that approximates the potential ϕ by a linear superposition of local potentials, combined with data interpolated from

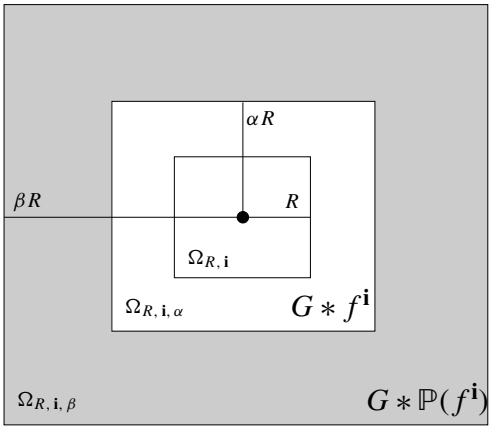


Figure 2. Regions associated with subdomain $\Omega_{R,\boldsymbol{i}}$. The potential in $\Omega_{R,\boldsymbol{i},\alpha}$ (white region) is given by $G * f^{\boldsymbol{i}}$. In the ring $\Omega_{R,\boldsymbol{i},\beta} \setminus \Omega_{R,\boldsymbol{i},\alpha}$ (shaded region) we use the field induced by a truncated Legendre expansion on $\Omega_{R,\boldsymbol{i}}$ of the local charge $f^{\boldsymbol{i}}$ to represent the potential.

a discrete global solution. The computational domain is a cube Ω that contains the support of f and is decomposed into a finite union of disjoint cubic subdomains of equal volume that are translations of $[-R, R]^3$, $R > 0$:

$$\operatorname{supp}(f) \subset \Omega = \bigcup_{\boldsymbol{i}} \Omega_{R,\boldsymbol{i}}, \quad \Omega_{R,\boldsymbol{i}} = c^{\boldsymbol{i}} + [-R, R]^3, \\ \boldsymbol{i} \in \mathbb{Z}^3,\ c^{\boldsymbol{i}} = (2\boldsymbol{i} + (1, 1, 1))R. \quad (18)$$

Then $f = \sum_{\boldsymbol{i}} f^{\boldsymbol{i}}$ where $f^{\boldsymbol{i}} = f\chi^{\boldsymbol{i}}$, where $\chi^{\boldsymbol{i}}$ is the characteristic function of $\Omega_{R,\boldsymbol{i}}$. As a consequence, the global potential may be written as

$$\phi(\boldsymbol{x}) = (G * f)(\boldsymbol{x}) = \sum_{\boldsymbol{i}} (G * f^{\boldsymbol{i}})(\boldsymbol{x}). \quad (19)$$

In other words, it is the linear superposition of the potentials induced by the local charges $f^{\boldsymbol{i}}$ which can be computed independently in parallel. The MLC algorithm replaces each of the summands in (19) with a solution truncated to zero outside of a localization radius βR, with the contribution to the solution outside the localization radius represented by interpolation from a single coarse-grid solution ϕ^H obtained by summing contributions of the form (15) over all the patches. At each point in space, the coarse-grid values used to interpolate the global contribution are corrected by subtracting off the contributions of the patches within the localization radius. Finally, to reduce the cost of computing the localized potentials, while keeping β large enough to make the $O(1)$ contribution to the error coming from localization be acceptably small, we introduce an inner radius $\alpha < \beta$ (see Figure 2). Within that inner radius, we compute the full convolution $G * f^{\boldsymbol{i}}$; in the annular region

$\Omega_{R,\boldsymbol{i},\beta} \setminus \Omega_{R,\boldsymbol{i},\alpha}$, the local solution is approximated by $G * \mathbb{P}(f^{\boldsymbol{i}})$, where $\mathbb{P}(f^{\boldsymbol{i}})$ is the orthogonal projection onto the Legendre polynomials on $\Omega_{R,\boldsymbol{i}}$ of some degree $P-1$:

$$\begin{aligned} \mathbb{P}(f^{\boldsymbol{i}}) &= \sum_{\boldsymbol{p} \in \mathbb{N}^3 : \|\boldsymbol{p}\|_1 < P} \langle Q^{\boldsymbol{p}}, f^{\boldsymbol{i}} \rangle Q^{\boldsymbol{p}}, \\ Q^{\boldsymbol{p}}(\boldsymbol{x}) &= R^{-3/2} \prod_{d=1}^{3} Q^{p_d}\left(\frac{x_d - c_d^{\boldsymbol{i}}}{R}\right), \quad \boldsymbol{x} \in \Omega_{R,\boldsymbol{i}},\ \boldsymbol{q} \in \mathbb{N}^3, \end{aligned} \tag{20}$$

where $\langle \cdot, \cdot \rangle$ is the inner product on $\Omega_{R,\boldsymbol{i}}$, and $Q^p : [-1, 1] \to \mathbb{R}$ is the classical Legendre polynomial of degree p.

3.1. ***The semidiscrete MLC algorithm.*** The semidiscrete MLC algorithm consists of three steps.

Step 1 (local convolutions). We perform local convolutions in regions around each subdomain $\Omega_{R,\boldsymbol{i}}$ that are used to compute local charges at points on the grid:

$$F^{\boldsymbol{i},H}[\boldsymbol{g}] = \begin{cases} \Delta^H(G * f^{\boldsymbol{i}})[\boldsymbol{g}] & \text{if } \boldsymbol{g} \in \Omega^H_{R,\boldsymbol{i},\alpha}, \\ \Delta^H(G * \mathbb{P}(f^{\boldsymbol{i}}))[\boldsymbol{g}] & \text{if } \boldsymbol{g} \in \Omega^H_{R,\boldsymbol{i},\beta} \setminus \Omega^H_{R,\boldsymbol{i},\alpha}, \\ 0 & \text{otherwise.} \end{cases}$$

Step 2 (global coarse solve). The global charge at coarse mesh points is constructed by assembling local contributions

$$F^H[\boldsymbol{g}] = \sum_{\boldsymbol{i}} F^{\boldsymbol{i},H}[\boldsymbol{g}],$$

and we obtain a global approximation ϕ^H of the potential, represented on the coarse mesh, by computing the discrete convolution over Ω^H:

$$\phi^H = G^H * F^H. \tag{21}$$

Step 3 (local interactions/local corrections). In the final step, we represent the solution on the boundary of each $\Omega_{R,\boldsymbol{i}}$ as the sum of local convolutions induced by charges on nearby patches and values interpolated from the grid calculation, from which the local convolution values have been subtracted:

$$\phi^{B,\boldsymbol{i}}(\boldsymbol{x}) = \phi^{\mathrm{loc},\boldsymbol{x}}(\boldsymbol{x}) + \mathcal{I}^H(\phi^H - \phi^{\mathrm{loc},\boldsymbol{x}})(\boldsymbol{x}). \tag{22}$$

Here $\mathcal{I}^H(\psi^H)(\boldsymbol{x})$ is an interpolation operator that takes as input values of $\psi^H : \mathcal{N}(\boldsymbol{x}) \to \mathbb{R}$, where $\mathcal{N}(\boldsymbol{x}) \subset \{\boldsymbol{g}H : \boldsymbol{g} \in \mathbb{Z}^3\}$, and returns a q_I-th-order accurate polynomial interpolant. In all of the algorithms described here, $\boldsymbol{x}$ and all of the points in $\mathcal{N}(\boldsymbol{x})$ are coplanar, so the interpolant is particularly easy to construct. Furthermore, $\phi^{\mathrm{loc},\boldsymbol{x}}(\boldsymbol{x})$ is the sum of all local convolutions the support of whose

charges is sufficiently close to $\boldsymbol{x}$ so that they contributed to the right-hand side for the grid solution near that point:

$$\phi^{\mathrm{loc},\boldsymbol{x}}(\boldsymbol{x}') = \sum_{\boldsymbol{i}:\boldsymbol{x}\in\Omega_{R,\boldsymbol{i},\alpha}} (G * f^{\boldsymbol{i}})(\boldsymbol{x}') + \sum_{\boldsymbol{i}:\boldsymbol{x}\in\Omega_{R,\boldsymbol{i},\beta}\setminus\Omega_{R,\boldsymbol{i},\alpha}} (G * \mathbb{P}(f^{\boldsymbol{i}}))(\boldsymbol{x}'). \tag{23}$$

Equation (22) can be interpreted as the decomposition of the potential at a point $\boldsymbol{x}$, into the sum of local contributions to the potential given by $\phi^{\mathrm{loc},\boldsymbol{x}}$ and corrections to include the global coupling by interpolating a corrected form of the coarse-mesh global solution ϕ^H. Specifically, the correction term in (22) is computed by evaluating $\phi^{\mathrm{loc},\boldsymbol{x}}$ at the points of the interpolation stencil $\mathcal{N}(\boldsymbol{x})$, subtracting these values from ϕ^H, and interpolating the result to $\boldsymbol{x}$. The MLC solution ϕ^{MLC} is specified in terms of solutions to Dirichlet problems on each $\Omega_{R,\boldsymbol{i}}$:

$$\Delta\phi^{\mathrm{MLC}} = f^{\boldsymbol{i}} \quad \text{in } \Omega_{R,\boldsymbol{i}}, \qquad \phi^{\mathrm{MLC}} = \phi^{B,\boldsymbol{i}} \quad \text{on } \partial\Omega_{R,\boldsymbol{i}}. \tag{24}$$

3.2. ***Error analysis.*** The error of the local corrections step for $\boldsymbol{x}\in\partial\Omega_{R,\boldsymbol{i}}$ is given by

$$\begin{aligned}(\phi^{B,\boldsymbol{i}} - \phi)(\boldsymbol{x}) &= \phi^{\mathrm{loc},\boldsymbol{x}}(\boldsymbol{x}) - \phi(\boldsymbol{x}) - \mathcal{I}^H(\phi^{\mathrm{loc},\boldsymbol{x}} - \phi)(\boldsymbol{x}) + \mathcal{I}^H(\phi^H - \phi)(\boldsymbol{x}) \\ &= \epsilon_I^H(\phi^{\mathrm{loc},\boldsymbol{x}} - \phi)(\boldsymbol{x}) + \mathcal{I}^H(\phi^H - \phi)(\boldsymbol{x})\end{aligned} \tag{25}$$

where $\epsilon_I^H(\psi)(\boldsymbol{x})$ is the error in applying the interpolation operator $\mathcal{I}^H$ to a smooth function ψ evaluated on the grid and evaluating it at $\boldsymbol{x}$. There are two sources of error for the semidiscrete algorithm: one from the calculation of ϕ^H in (21), and the other due to interpolation at the local corrections step (22). To estimate the former, i.e., the second term of (25), it suffices to bound the coarse mesh error $\phi^H - \phi$. To do so, we estimate the truncation error of the coarse solve (21) at points $\boldsymbol{g}$:

$$\begin{aligned}\tau_C^H &= \Delta^H(\phi^H - \phi)[\boldsymbol{g}] \\ &= -\Delta^H\Bigg(\sum_{\boldsymbol{i}:\boldsymbol{g}H\notin\Omega_{R,\boldsymbol{i},\beta}} G*f^{\boldsymbol{i}}\Bigg)[\boldsymbol{g}] - \Delta^H\Bigg(\sum_{\boldsymbol{i}:\boldsymbol{g}H\in\Omega_{R,\boldsymbol{i},\beta}\setminus\Omega_{R,\boldsymbol{i},\alpha}} G*((\mathbb{I}-\mathbb{P})(f^{\boldsymbol{i}}))\Bigg)[\boldsymbol{g}].\end{aligned} \tag{26}$$

To bound the first term of (26), we use (14) to find that

$$\begin{aligned}&\Delta^H\Bigg(\sum_{\boldsymbol{i}:\boldsymbol{g}H\notin\Omega_{R,\boldsymbol{i},\beta}} G*f^{\boldsymbol{i}}\Bigg)[\boldsymbol{g}] \\ &\qquad = O\Bigg(\Big(\frac{H}{R}\Big)^q \sum_{k=0}^{\infty} \sum_{\boldsymbol{i}:\boldsymbol{g}H\in\Omega_{R,\boldsymbol{i},\beta+k+1}\setminus\Omega_{R,\boldsymbol{i},\beta+k}} \frac{1}{(\beta+k)^{q+3}}\|f^{\boldsymbol{i}}\|_\infty\Bigg) \\ &\qquad = O\Bigg(\Big(\frac{H}{R}\Big)^q \frac{1}{\beta^q}\|f\|_\infty\Bigg).\end{aligned} \tag{27}$$

The second term of (26) is bounded in a similar fashion:

$$\Delta^H\left(\sum_{\boldsymbol{i}:\boldsymbol{g}H\in\Omega_{R,i,\beta}\setminus\Omega_{R,i,\alpha}} G*((\mathbb{I}-\mathbb{P})(f^{\boldsymbol{i}}))\right)[\boldsymbol{g}] = O\left(\left(\frac{H}{R}\right)^q \frac{1}{\alpha^q}\max_{\boldsymbol{i}}\|(\mathbb{I}-\mathbb{P})(f^{\boldsymbol{i}})\|_\infty\right)$$
$$= O\left(\left(\frac{H}{R}\right)^q \frac{1}{\alpha^q}H^P\right), \tag{28}$$

where we have used

$$\|(\mathbb{I}-\mathbb{P})(f^{\boldsymbol{i}})\|_\infty = O(R^P), \tag{29}$$

which follows directly from Taylor's theorem for $f^{\boldsymbol{i}}$ and the fact that $\pi = \mathbb{P}(\pi)$ for polynomials π of degree less than P. As a result, the estimate

$$\Delta^H(\phi^H - \phi) = O\left(\left(\frac{H}{R}\right)^q \frac{1}{\alpha^q}H^P\right) + O\left(\left(\frac{H}{R}\right)^q \frac{1}{\beta^q}\|f\|_\infty\right) \tag{30}$$

for the coarse mesh error holds uniformly on coarse mesh points. Since convolution with G^H and the interpolation operator $\mathcal{I}^H$ are max norm bounded, $\epsilon_C^H \equiv \phi^H - \phi$ is also bounded by an expression of the form of the right-hand side of (30).

To bound the first term in (25), it follows from the fact that the interpolation method is q_I-th-order accurate that

$$\epsilon_I^H(\phi^{\mathrm{loc},\boldsymbol{x}} - \phi)(\boldsymbol{x}) = H^{q_I}L_I^{q_I}(\phi^{\mathrm{loc},\boldsymbol{x}} - \phi)(\boldsymbol{\xi})$$
$$= -H^{q_I}\left(\sum_{\boldsymbol{i}:\boldsymbol{x}\in\Omega_{R,i,\beta}\setminus\Omega_{R,i,\alpha}} ((L_I^{q_I}G)*(\mathbb{I}-\mathbb{P})(f^{\boldsymbol{i}}))(\boldsymbol{\xi}) + \sum_{\boldsymbol{i}:\boldsymbol{x}\notin\Omega_{R,i,\beta}} ((L_I^{q_I}G)*f^{\boldsymbol{i}})(\boldsymbol{\xi})\right) \tag{31}$$

where $\boldsymbol{\xi}$ is in an $O(H)$ neighborhood of $\mathcal{N}(\boldsymbol{x})$ and $L_I^{q_I}$ is a linear differential operator with terms that are derivatives of order q_I. Using (13), a similar argument to that given in the proof of (30) leads to

$$\epsilon_I^H = H^{P+2}O\left(\left(\frac{H}{R}\right)^{q_I-2}\frac{1}{\alpha^{q_I-2}}\right) + H^2O\left(\left(\frac{H}{R}\right)^{q_I-2}\frac{1}{\beta^{q_I-2}}\|f\|_\infty\right)$$

so that (25) is estimated as

$$\epsilon^{\mathrm{SD}} \equiv \phi^{B,\boldsymbol{i}} - \phi = H^{P+2}O\left(\left(\frac{H}{R}\right)^{q_I-2}\frac{1}{\alpha^{q_I-2}}\right) + H^2O\left(\left(\frac{H}{R}\right)^{q_I-2}\frac{1}{\beta^{q_I-2}}\|f\|_\infty\right)$$
$$+ O\left(\left(\frac{H}{R}\right)^q\frac{1}{\alpha^q}H^P\right) + O\left(\left(\frac{H}{R}\right)^q\frac{\|f\|_\infty}{\beta^q}\right). \tag{32}$$

4. Method of local corrections: fully discrete case

In this section, we describe the two-level algorithm as it is actually implemented. Ω^h is a fine-grid discretization of a bounded domain Ω, the latter containing

the support of f. Ω^h is assumed to be a finite union of rectangles of the form $\Omega^h_{R,\boldsymbol{i}} = n\boldsymbol{i} + [0,n]^3$, $R = nh/2$. We also define discrete forms of $\Omega^h_{R,\boldsymbol{i},\alpha}$ and $\Omega^h_{R,\boldsymbol{i},\beta}$: $\Omega^h_{R,\boldsymbol{i},\alpha} = \mathcal{G}(\Omega^h_{R,\boldsymbol{i}}, \lceil(\alpha-1)n/2\rceil)$ and $\Omega^h_{R,\boldsymbol{i},\beta} = \mathcal{G}(\Omega^h_{R,\boldsymbol{i}}, \lceil(\beta-1)n/2\rceil)$. The coarse grid Ω^H is assumed to cover all of the fine patch data required for the algorithm described below: $\mathcal{G}(\mathcal{C}(\Omega^h_{R,\boldsymbol{i},\beta}), b) \subset \Omega^H$ where b is the radius of the stencil for the interpolation function $\mathcal{I}^H$. We also define a discretized form of the characteristic function of a rectangular patch $D \subset \mathbb{Z}^3$:

$$\chi_D(\boldsymbol{x}) = \begin{cases} \frac{1}{8} & \text{if } \boldsymbol{g} \text{ is a corner of } D, \\ \frac{1}{4} & \text{if } \boldsymbol{g} \text{ lies on an edge of } D, \\ \frac{1}{2} & \text{if } \boldsymbol{g} \text{ lies on a face of } D, \\ 1 & \text{if } \boldsymbol{g} \text{ lies in the interior of } D, \\ 0 & \text{elsewhere.} \end{cases}$$

In the fully discrete algorithm, we replace the local convolutions with local discrete convolutions, e.g., $G * f^{\boldsymbol{i}} \to G^h * f^{\boldsymbol{i},h}$ and $f^{\boldsymbol{i},h} = \chi_{\Omega^h_{R,\boldsymbol{i}}} f$, and we take $H = N_{\text{ref}} h$.

4.1. *The fully discrete two-level algorithm.*

Step 1 (local convolutions). For each $\Omega^h_{R,\boldsymbol{i}}$, we compute the potential induced by $f^{\boldsymbol{i},h} = \chi_{\Omega^h_{R,\boldsymbol{i}}} f^h$:

$$\phi^{\boldsymbol{i},h} = G^h * f^{\boldsymbol{i},h} \quad \text{on } \mathcal{G}(\Omega^h_{R,\boldsymbol{i},\alpha}, N_{\text{ref}} b). \tag{33}$$

The Legendre expansion coefficients of $f^{\boldsymbol{i},h}$ required to compute $\mathbb{P}(f^{\boldsymbol{i}})$ are computed with composite numerical integration. We employ Boole's rule if f is given only at points of Ω^h or Gauss integration if f is specified analytically. For each $\Omega^h_{R,\boldsymbol{i}}$ we also compute the associated local charges

$$F^{\boldsymbol{i},H}[\boldsymbol{g}] = \begin{cases} \Delta^H \phi^h_{\boldsymbol{i}}[\boldsymbol{g}], & \boldsymbol{g} \in \mathcal{C}(\Omega^h_{R,\boldsymbol{i},\alpha}), \\ \Delta^H (G^h * \mathbb{P}^h(f^{\boldsymbol{i},h}))[\boldsymbol{g}], & \boldsymbol{g} \in \mathcal{C}(\Omega^h_{R,\boldsymbol{i},\beta}) \setminus \mathcal{C}(\Omega^h_{R,\boldsymbol{i},\alpha}), \\ 0, & \boldsymbol{g} \notin \mathcal{C}(\Omega^h_{R,\boldsymbol{i},\beta}). \end{cases} \tag{34}$$

The values of $\Delta^H(G^h * Q^{\boldsymbol{p}})$ can be computed once and stored, reducing the calculation of $\Delta^H(G^h * \mathbb{P}^h(f^{\boldsymbol{i},h}))$ to computing linear combinations of the appropriate subset of those precomputed values.

Step 2 (global coarse solve). $\phi^H = G^H * F^H$ on Ω^H, $F^H = \sum_{\boldsymbol{i}} F^{\boldsymbol{i},H}$.

Step 3 (local interactions/local corrections). We define the local potentials at fine boundary points $\boldsymbol{g} \in \partial\Omega^h_{R,\boldsymbol{i}}$ as combinations of short-range and intermediate-range components

$$\phi^{\text{loc},\boldsymbol{g}}[\boldsymbol{g}'] = \sum_{\boldsymbol{i}' : \boldsymbol{g} \in \Omega^h_{R,\boldsymbol{i}',\alpha}} \phi^{\boldsymbol{i}',h}[\boldsymbol{g}'] + \sum_{\boldsymbol{i}' : \boldsymbol{g} \in \Omega^h_{R,\boldsymbol{i}',\beta} \setminus \Omega^h_{R,\boldsymbol{i}',\alpha}} (G^h * \mathbb{P}^h(f^{\boldsymbol{i}',h}))[\boldsymbol{g}'], \tag{35}$$

and we correct them by adding the far field effects as in (22):

$$\phi^{B,\boldsymbol{i},h}[\boldsymbol{g}] = \phi^{\mathrm{loc},\boldsymbol{g}}[\boldsymbol{g}] + \mathcal{I}^H(\phi^H - (\phi^{\mathrm{loc},\boldsymbol{g}}))(\boldsymbol{g}h), \quad \boldsymbol{g} \in \partial\Omega_{\boldsymbol{i}}^h. \tag{36}$$

The interpolation operator on coplanar points $\mathcal{I}^H$ that we are employing is the same as in [25]. Using these boundary conditions, we solve the following local Dirichlet problems on $\Omega_{\boldsymbol{i}}^h$ patches:

$$\begin{aligned} \Delta^h \tilde{\phi}^{\mathrm{MLC},\boldsymbol{i},h} &= f^{\boldsymbol{i},h} \quad \text{on } \Omega_{R,\boldsymbol{i}}^h \setminus \partial\Omega_{R,\boldsymbol{i}}^h, \\ \tilde{\phi}^{\mathrm{MLC},\boldsymbol{i},h} &= \phi^{B,\boldsymbol{i},h} \quad \text{on } \partial\Omega_{R,\boldsymbol{i}}^h. \end{aligned} \tag{37}$$

Finally, the fourth-order Mehrstellen correction (12) is applied to obtain the values of $\phi^{\mathrm{MLC},h}$

$$\phi^{\mathrm{MLC},h}[\boldsymbol{g}] = \tilde{\phi}^{\mathrm{MLC},\boldsymbol{i},h}[\boldsymbol{g}] + C_2 h^2 f^h[\boldsymbol{g}], \quad \boldsymbol{g} \in \Omega_{R,\boldsymbol{i}}^h. \tag{38}$$

If we want to go to higher than fourth-order accuracy in h, the algorithm is more complicated — the Mehrstellen correction must be applied earlier in the process. We will not discuss the details in this paper.

4.2. ***Error analysis.*** We proceed in this section with estimating the error for the fully discrete MLC algorithm. We want to get some idea of the impact of replacing the analytic continuous convolutions by the discretized convolutions. To do this, we use a modified equation approach, in which we assume that we can approximate the solution error by the action of the operator on the truncation error. In the present setting, this amounts to making the substitution

$$G^h * \psi^h \to G * (\psi + \delta\tau^h(\psi)) - C_2 h^2 \psi, \tag{39}$$

$$\delta\tau^h(\psi) = \Delta(G^h * \psi^h) - \psi + C_2 h^2 \Delta\psi = O(h^4). \tag{40}$$

As in the semidiscrete case, we want to estimate the error in the boundary conditions

$$\begin{aligned} \phi^{B,\boldsymbol{i},h}[\boldsymbol{g}] - \tilde{\phi}(\boldsymbol{g}h) &= \phi^{\mathrm{loc},\boldsymbol{g}}[\boldsymbol{g}] - \tilde{\phi}(\boldsymbol{g}h) + \mathcal{I}^H(\phi^H - \phi^{\mathrm{loc},\boldsymbol{g}})(\boldsymbol{g}h) \\ &= \mathcal{I}^H(\phi^H - \tilde{\phi})(\boldsymbol{g}h) \qquad (41) \\ &\quad + \phi^{\mathrm{loc},\boldsymbol{g}}[\boldsymbol{g}] - \tilde{\phi}(\boldsymbol{g}h) - \mathcal{I}^H(\phi^{\mathrm{loc},\boldsymbol{g}} - \tilde{\phi})(\boldsymbol{g}h), \quad \boldsymbol{g} \in \Omega_{R,\boldsymbol{i}}^h, \qquad (42) \end{aligned}$$

where

$$\tilde{\phi} \equiv \phi + C_2 h^2 f.$$

An estimate of the contribution from (41) is obtained by bounding $\Delta^H(\phi^H - \tilde{\phi})$, since $\mathcal{I}^H$ and convolution with G^H are both stable in max norm. We have, by (39),

$$\begin{aligned}
\Delta^H(\phi^H - \tilde{\phi})[\boldsymbol{g}] = &- \sum_{\boldsymbol{i}':\boldsymbol{g}\notin\Omega^H_{R,\boldsymbol{i}',\beta}} \Delta^H(G * f^{\boldsymbol{i}'})[\boldsymbol{g}] \\
&- \sum_{\boldsymbol{i}':\boldsymbol{g}\in\Omega^H_{R,\boldsymbol{i}',\beta}\setminus\Omega^H_{R,\boldsymbol{i}',\alpha}} \Delta^H(G * (\mathbb{I}-\mathbb{P})(f^{\boldsymbol{i}'}))[\boldsymbol{g}] - \sum_{\boldsymbol{i}':\boldsymbol{g}\notin\Omega^H_{R,\boldsymbol{i}',\beta}} \Delta^H(G * (\delta\tau^h(f^{\boldsymbol{i}',h})))[\boldsymbol{g}] \\
&- \sum_{\boldsymbol{i}':\boldsymbol{g}\in\Omega^H_{R,\boldsymbol{i}',\beta}\setminus\Omega^H_{R,\boldsymbol{i}',\alpha}} \Delta^H(G * \delta\tau^h((\mathbb{I}-\mathbb{P})(f^{\boldsymbol{i}'})))[\boldsymbol{g}] \\
&- \sum_{\boldsymbol{i}':\boldsymbol{g}\in\Omega^H_{R,\boldsymbol{i}',\beta}\setminus\Omega^H_{R,\boldsymbol{i}',\alpha}} \Delta^H(G^h * ((\mathbb{P}(f^{\boldsymbol{i}'}))^h - (\mathbb{P}^h(f^{\boldsymbol{i}',h}))))[\boldsymbol{g}] + O(h^4). \qquad (43)
\end{aligned}$$

The first two terms are identical to the ones that appear in the semidiscrete case, while (39) and the estimate $\|(\mathbb{P}-\mathbb{P}^h)(f^{\boldsymbol{i}'})\|_\infty = O(h^6)$ (which holds since our quadrature rules for computing the Legendre coefficients are at least sixth-order accurate) guarantee that the remaining terms are $O(h^4)$ or smaller. Using similar arguments to those in (43), we have

$$\phi^{\mathrm{loc},\boldsymbol{g}} - \tilde{\phi} = - \sum_{\boldsymbol{i}':\boldsymbol{g}\notin\Omega^h_{R,\boldsymbol{i}',\beta}} G * f^{\boldsymbol{i}'} - \sum_{\boldsymbol{i}':\boldsymbol{g}\in\Omega^h_{R,\boldsymbol{i}',\beta}\setminus\Omega^h_{R,\boldsymbol{i}',\alpha}} G * ((\mathbb{I}-\mathbb{P})(f^{\boldsymbol{i}'})) + O(h^4),$$

and therefore, following (31), we have

$$\begin{aligned}
&\epsilon_I^H(\phi^{\mathrm{loc},\boldsymbol{g}} - \tilde{\phi})(\boldsymbol{g}h) \\
&\qquad = H^{P+2} O\left(\left(\frac{H}{R}\right)^{q_I-2} \frac{1}{\alpha^{q_I-2}}\right) + H^2 O\left(\left(\frac{H}{R}\right)^{q_I-2} \frac{1}{\beta^{q_I-2}} \|f\|_\infty\right) + O(h^4),
\end{aligned}$$

Thus, we have

$$\phi^{B,\boldsymbol{i},h}[\boldsymbol{g}] - \tilde{\phi}(\boldsymbol{g}h) = \epsilon^{\mathrm{SD}} + O(h^4).$$

The stability of the discretized boundary value problem implies $\|\phi^{\mathrm{MLC},h} - \phi\|_\infty = O(\|\phi^{B,h} - \phi\|_\infty) + O(h^4)$, so we finally have the estimate

$$\begin{aligned}
\phi^{\mathrm{MLC},h} - \phi &= O(h^4) + \epsilon^{\mathrm{SD}} \\
&= O(h^4) + H^{P+2} O\left(\left(\frac{H}{R}\right)^{q_I-2} \frac{1}{\alpha^{q_I-2}}\right) + H^2 O\left(\left(\frac{H}{R}\right)^{q_I-2} \frac{\|f\|_\infty}{\beta^{q_I-2}}\right) \\
&\qquad + O\left(\left(\frac{H}{R}\right)^{q} \frac{1}{\alpha^q} H^P\right) + O\left(\left(\frac{H}{R}\right)^{q} \frac{\|f\|_\infty}{\beta^q}\right). \qquad (44)
\end{aligned}$$

at all fine grid points. This error can be written in the form

$$\phi^{\mathrm{MLC},h} = \phi + O(h^4) + O(h^P) + O\left(h^2 \|f\|_\infty \frac{1}{\beta^{q_I-2}}\right) + O\left(\|f\|_\infty \frac{1}{\beta^q}\right). \quad (45)$$

Thus, MLC differs from classical finite difference methods in that there is a contribution to the error that does not vanish as $h \to 0$, i.e., the right-most summand in (44). We refer to this contribution to the error as the *barrier error*. Note that, if we take $q_I = q + 2$, we obtain the form of the error given in the Introduction. We have specialized this algorithm to the case of fourth-order accuracy, primarily because it allows us the simplification of applying the Mehrstellen correction (38) at the end of the calculation. However, this analysis suggests that, even with this simplification, there might be an advantage to using discretizations of the Laplacian with larger q, i.e., ones that are higher-order accurate when applied to harmonic functions, since the barrier error is proportional to β^{-q}. We observe this to be the case in the results in Section 7.

5. Multilevel method of local corrections

Following [25], we generalize the method in Section 4 to the case of an arbitrary number of levels $l = 0, \ldots, l_{\max}$, where $l_{\max}$ is the finest level on which the solution is sought. We denote the discrete Laplacian with mesh size h_l by Δ^{h_l}, with $h_l = N_{\mathrm{ref}} h_{l+1}$. At each level we discretize the solution on a collection of node-centered cubic patches $\Omega_{R_l,\boldsymbol{i}}$, $R_l = N_{\mathrm{ref}} R_{l+1}$, and the corresponding discretized grids $\Omega^{h_l}_{R_l,\boldsymbol{i}}$; the combined level-$l$ grid is given by $\Omega^{l,h_l} \equiv \bigcup_{\boldsymbol{i}} \Omega^{h_l}_{R_l,\boldsymbol{i}}$. We also define, for each $\boldsymbol{i}$, localization regions $\Omega_{R_l,\boldsymbol{i},\alpha}$ and $\Omega_{R_l,\boldsymbol{i},\beta}$, and their discretizations $\Omega^{h_l}_{R_l,\boldsymbol{i},\alpha}$ and $\Omega^{h_l}_{R_l,\boldsymbol{i},\beta}$, $1 < \alpha < \beta$. At level 0 there is only one patch Ω^{0,h_0} at which the coarse solve of the method is performed, just as in the two-level algorithm. We also impose a proper nesting condition: for $l = 1, \ldots, l_{\max}$,

$$\mathcal{G}(\mathcal{C}(\Omega^{h_l}_{R_l,\boldsymbol{i},\beta}), b) \subset \Omega^{l-1,h_{l-1}}. \quad (46)$$

The multilevel MLC comprises the following steps.

Step 1 (downward pass: initial local convolutions). Local convolutions are computed at levels $l = l_{\max}, \ldots, 1$:

$$\phi^{\boldsymbol{i},h_l} = G^{h_l} * \tilde{f}^{\boldsymbol{i},h_l} \quad \text{on } \mathcal{G}(\Omega^{h_l}_{R_l,\boldsymbol{i},\alpha}, N_{\mathrm{ref}} b), \quad (47)$$

where the local right-hand sides are defined as

$$\tilde{f}^{\boldsymbol{i},h_l} = \sum_{\boldsymbol{i}'} \Delta^{h_l}(\phi^{\boldsymbol{i}',h_{l+1}})|_{\mathscr{C}(\Omega^{h_{l+1}}_{R_{l+1},\boldsymbol{i}',\alpha})} + \sum_{\boldsymbol{i}'} \Delta^{h_l}(G^{h_{l+1}} * \mathbb{P}(f^{\boldsymbol{i}',h_{l+1}}))|_{\mathscr{C}(\Omega^{h_{l+1}}_{R_{l+1},\boldsymbol{i}',\beta}\setminus\Omega^{h_{l+1}}_{R_{l+1},\boldsymbol{i}',\alpha})} + \tilde{\chi}_{\Omega^{h_l}_{R_l,\boldsymbol{i}}} f^{h_l},$$

$$\tilde{\chi}_{\Omega^{h_l}_{R_l,\boldsymbol{i}}}[\boldsymbol{g}] = \chi_{\Omega^{h_l}_{R_l,\boldsymbol{i}}}[\boldsymbol{g}] - \sum_{\substack{\boldsymbol{i}'=N_{\mathrm{ref}}\boldsymbol{i}+\boldsymbol{s}\\ 0\le \boldsymbol{s}_d\le N_{\mathrm{ref}}}} \chi_{\Omega^{h_{l+1}}_{R_{l+1},\boldsymbol{i}'}}[N_{\mathrm{ref}}\boldsymbol{g}].$$

Step 2 (global coarse solve). $\phi^{h_0} = G^{h_0} * \tilde{f}^{h_0}$ on Ω^{0,h_0}.

Step 3 (upward pass: local interactions/local corrections for $1,\ldots,l_{\max}$). Starting from level 1, the following local Dirichlet problems are solved at levels $l = 1,\ldots,l_{\max}$:

$$\begin{aligned} \Delta^{h_l}\tilde{\phi}^{\mathrm{MLC},\boldsymbol{i},h_l} &= \tilde{f}^{\boldsymbol{i},h_l} && \text{on } \Omega^{h_l}_{R_l,\boldsymbol{i}} \setminus \partial\Omega^{h_l}_{R_l,\boldsymbol{i}},\\ \tilde{\phi}^{\mathrm{MLC},\boldsymbol{i},h_l} &= \phi^{B,\boldsymbol{i},h_l} && \text{on } \partial\Omega^{h_l}_{R_l,\boldsymbol{i}},\\ \tilde{\phi}^{\mathrm{MLC},l} &= \tilde{\phi}^{\mathrm{MLC},\boldsymbol{i},h_l} && \text{on } \Omega^{h_l}_{R_l,\boldsymbol{i}}. \end{aligned} \tag{48}$$

The Dirichlet boundary conditions are given by

$$\phi^{B,\boldsymbol{i},h_l}[\boldsymbol{g}] = \phi^{\mathrm{loc},l,\boldsymbol{g}}[\boldsymbol{g}] + \mathcal{I}^{h_{l-1}}(\tilde{\phi}^{\mathrm{MLC},l-1} - \phi^{\mathrm{loc},l,\boldsymbol{g}})(\boldsymbol{g}h_l). \tag{49}$$

Here the local potentials $\phi^{\mathrm{loc},\boldsymbol{g},l}$ are given by

$$\phi^{\mathrm{loc},l,\boldsymbol{g}}[\boldsymbol{g}'] = \sum_{\boldsymbol{i}':\boldsymbol{g}\in\Omega^{h^l}_{R_l,\boldsymbol{i}',\alpha}} \phi^{\boldsymbol{i}',h_l}[\boldsymbol{g}'] + \sum_{\boldsymbol{i}':\boldsymbol{g}\in\Omega^{h_l}_{R_l,\boldsymbol{i}',\beta}\setminus\Omega^{h_l}_{R_l,\boldsymbol{i}',\alpha}} (G^{h_l} * \mathbb{P}(f^{\boldsymbol{i}'}))[\boldsymbol{g}']. \tag{50}$$

Finally, the Mehrstellen correction at all levels is applied as

$$\phi^{\mathrm{MLC},l}[\boldsymbol{g}] = \tilde{\phi}^{\mathrm{MLC},l}[\boldsymbol{g}] + C_2 h_l^2 f^{\boldsymbol{i},h_l}[\boldsymbol{g}], \quad \boldsymbol{g}\in\Omega^{l,h_l} \tag{51}$$

We do not have a complete error analysis for the above algorithm corresponding to that given in the two-level case. However, we can look at error analysis of the two-level algorithm, and determine the change in the error introduced there by replacing the coarse-grid convolution with G^H with an MLC calculation. We denote

- $G^{\mathrm{MLC},S}(r)$ the two-level semidiscrete method of local corrections approximation to $G * r$, with patch radius S,
- $N_1^S(r)(\boldsymbol{x}) \equiv \sum_{\boldsymbol{i}:\boldsymbol{x}\notin\Omega_{S,\boldsymbol{i},\beta}} h^q L^{q+2}(G * r^{\boldsymbol{i}})(\boldsymbol{x})$,
- $N_2^S(r)(\boldsymbol{x}) \equiv \sum_{\boldsymbol{i}:\boldsymbol{x}\in\Omega_{S,\boldsymbol{i},\beta}\setminus\Omega_{S,\boldsymbol{i},\alpha}} h^q L^{q+2}(G * ((\mathbb{I}-\mathbb{P})r^{\boldsymbol{i}}))(\boldsymbol{x})$, and
- $N^S(r) = N_1^S(r) + N_2^S(r)$.

By (25) and (26), $G^H * (N^R(f))^H = (G * f)^H - \phi^H$ is the only quantity in the error in which convolution with G^H appears. Given that, it is straightforward to assess the impact of replacing the convolution with G^H in this expression with applying the MLC algorithm for a patch size $N_{\text{ref}}R$. To estimate this effect, we use a modified equation approach, in which the difference is approximated by $G * (N^R(f)) - G^{\text{MLC},N_{\text{ref}}R}(N^R(f))$. Applying the error estimate (26), we obtain

$$\begin{aligned} G * (N^R(f)) - G^{\text{MLC},N_{\text{ref}}R}(N^R(f)) &= N^{N_{\text{ref}}R}(N^R(f)) \\ &= N_1^{N_{\text{ref}}R}(N_1^R(f)) + N_1^{N_{\text{ref}}R}(N_2^R(f)) + N_2^{N_{\text{ref}}R}(N_1^R(f)) + N_2^{N_{\text{ref}}R}(N_2^R(f)). \end{aligned}$$

For this substitution to have an appropriately small impact, it is sufficient for the error to be comparable to or less than the error in the two-level algorithm. The sum of the first three terms meet this criterion — the sum of the first two terms is bounded by the max norm of the two-level error multiplied by $O(\beta^{-q})$, and the third term is bounded by $O(\alpha^{-q})$ times the max norm of the barrier error of the two-level algorithm. The final term, however, is problematic. In particular, the impact on the error of multiple applications of $\mathbb{I} - \mathbb{P}$ at increasing mesh spacings is far from clear. We will see evidence of this in the numerical results in Section 7.2, and will suggest a remedy that allows the error to be controlled.

6. Computational issues

The analysis and demonstration of the performance of this algorithm will be the subject of a separate paper [24], so we will just make a few high-level observations to justify the pursuit of this line of research. The largest contribution to the floating point operation count in this method comes from the initial local discrete convolutions (33). To compute these convolutions, we use a generalization of Hockney's domain-doubling algorithm [16], which we describe in Appendix B. The floating point work per unknown for this step is $O(\alpha^3 \log(n))$, $\alpha > 1$, where n^3 is the number of points per patch. The next-largest computation is that of the final Dirichlet solutions (37), performed using sine transforms, which is $O(\log(n))$ per unknown. The floating point work associated with computing the Legendre expansions is small, with the convolutions of Legendre polynomials with the discrete Green's functions precomputed and stored. The memory overhead for storing these quantities scales like $O(\beta^3 n^3)$. However, there is one copy of these per processor, shared across multiple patches/cores. Furthermore, they are only stored either on a sampled grid coarsened by N_{ref}, or on planar subsets corresponding to boundaries of patches, which reduces the memory overhead further.

The parallel implementation of this algorithm is via domain decomposition, with patches distributed to processors. For the choices of α and β used in the results described below, this corresponds to a floating point operation count about

three times that of a corresponding multigrid algorithm for comparable accuracy. Roughly speaking, the communications cost, in terms of number of messages and overall volume of data moved, corresponds to that of a single multigrid V-cycle, plus the negligible costs of communicating a small number of Legendre expansion coefficients (20 per patch for the case $P = 4$). This is to be compared to the 8 multigrid V-cycles required to obtain a comparable level of accuracy. Current trends in the design of HPC processors based on low-power processor technologies indicate a rapid growth in the number of cores capable of performing floating point operations on a processor, while the communications bandwidth between processors, or between the processor and main memory, is growing much more slowly. In addition, most of the floating point work is performed using FFTs on small patches on a single node, for which there are multiple opportunities for performance optimization. Thus, the present algorithm is well positioned to take advantage of these trends.

7. Numerical test cases

We present in this section several examples that demonstrate the convergence properties of the MLC method described above. In all cases, we use as a measure of the solution error the max norm error of the potential, divided by max norm of the potential

$$\frac{\|\phi^{\mathrm{MLC},h} - \phi\|_\infty}{\|\phi\|_\infty}. \tag{52}$$

For all cases, we set $n = 32$, so that $H/R = \frac{1}{4}$. We refer to the special case $\beta = \alpha$ (i.e., if the long-range potentials induced by the truncated Legendre expansions of local charges are ignored) as the MLC-0 method and to the general case $\alpha < \beta$ as the MLC method. It is not difficult to see that for MLC-0, the estimate (44) reduces to

$$\phi^{\mathrm{MLC},h} - \phi = O(h^4) + O\left(h^2\left(\frac{H}{R}\right)^{q_I-2}\frac{\|f\|_\infty}{\beta^{q_I-2}}\right) + O\left(\left(\frac{H}{R}\right)^{q}\frac{\|f\|_\infty}{\beta^{q}}\right). \tag{53}$$

Increasing β to reduce the barrier error in (53) substantially increases the per patch computational cost of the discrete convolution in the downward pass of the method. This is, in fact, the reason we replaced the local long-range potential values with the convolutions of the local Legendre expansions in Section 3.1.

7.1. ***A smooth charge distribution.*** The first test case we are considering involves computing the potential induced by a smooth charge. The computational domain is the unit cube $\Omega = [0, 1]^3$. The charge density is given by

$$f(\boldsymbol{x}) = \begin{cases} (r - r^2)^4, & r < 1, \\ 0, & r \geq 1, \end{cases} \qquad r = \frac{1}{R_o}\|\boldsymbol{x} - \boldsymbol{x}_o\|,$$

N	$\beta = 1.5$	$\beta = 3.0$	$\beta = 6.0$
256	1.43756×10^{-5}	6.07186×10^{-7}	5.80288×10^{-8}
512	1.29572×10^{-5}	4.32691×10^{-7}	2.67372×10^{-8}
1024	1.27114×10^{-5}	4.01180×10^{-7}	2.44521×10^{-8}

Table 1. 2-level MLC-0: scaled fine-mesh maximum errors (52) using the L_{19}^h Mehrstellen Laplacian.

N	MLC-0 ($\beta = 1.5$)	$P = 1$	$P = 4$
256	1.43756×10^{-5}	4.35976×10^{-6}	1.63706×10^{-6}
512	1.29572×10^{-5}	1.43414×10^{-6}	4.58615×10^{-7}
1024	1.27114×10^{-5}	5.77475×10^{-7}	3.65246×10^{-7}

Table 2. 2-level MLC: scaled fine-mesh maximum errors (52) using L_{19}^h. For sufficiently small h and $P = 4$, nearly the same errors as the second column of Table 1 are obtained.

N	MLC-0 ($\beta = 1.5$)	$P = 1$	$P = 4$	$P = 5$
256	1.43756×10^{-5}	4.05752×10^{-6}	1.45072×10^{-6}	1.68422×10^{-6}
512	1.29572×10^{-5}	1.12630×10^{-6}	1.04191×10^{-7}	4.49529×10^{-8}
1024	1.27114×10^{-5}	2.37651×10^{-7}	2.55964×10^{-8}	2.44951×10^{-8}

Table 3. 2-level MLC: scaled fine-mesh maximum errors (52) using L_{19}^h. Here $\alpha = 1.5$ and $\beta = 6$. For sufficiently small h and high values of P, nearly the same errors as the third column of Table 1 are obtained.

and the support of the charge is a sphere of radius $R_o = \frac{1}{4}$, centered at the point $\boldsymbol{x}_o = (\frac{1}{2}, \frac{1}{2}, \frac{1}{2})$. The exact solution for this problem is given by

$$\phi(\boldsymbol{x}) = R_o^2 \begin{cases} \frac{1}{42} r^6 - \frac{1}{14} r^7 + \frac{1}{12} r^8 - \frac{2}{45} r^9 + \frac{1}{110} r^{10} - \frac{1}{1260}, & r < 1, \\ -1/(2310 r), & r \geq 1, \end{cases}$$

and reduces to a pure monopole field for $r \geq 1$.

7.1.1. *Two-level results.* In Table 1 we present the fine-mesh errors for the MLC-0 algorithm with two levels for mesh sizes $h = \frac{1}{256}, \frac{1}{512}, \frac{1}{1024}$ using the L_{19}^h Mehrstellen Laplacian ($q = 4$). We set $b = 2 \to q_I = 6$ so that dependence of the interpolation error as a function of α and β matches that of the other error terms. For this problem, the errors in all three cases are so small that they are the barrier errors; each time we double β, the error goes down by roughly a factor of 16, as predicted by (53). In Tables 2 and 3 we present fine-mesh errors for the MLC algorithm, with $\alpha = 1.5$, for $\beta = 3$ and $\beta = 6$, respectively, when refining both h and P. As $h \to 0$, the error in this case approaches a barrier error for both the $P = 1$ and $P = 4$ cases at a rate of $O(h^2)$–$O(h^4)$, and those barrier errors correspond to the errors for the MLC-0 calculations with same corresponding values of β. For comparison, we also include

N	$\beta = 2.0$	$\beta = 3.25$
256	1.25208×10^{-7}	4.11121×10^{-8}
512	1.14831×10^{-7}	4.92150×10^{-9}
1024	1.01073×10^{-7}	3.11897×10^{-9}

Table 4. 2-level MLC-0: scaled fine-mesh maximum error (52) using the L_{27}^h Mehrstellen Laplacian. Compare with the second and third columns of Table 1.

N	$P = 1$	$P = 4$
256	1.45293×10^{-6}	1.40270×10^{-6}
512	5.20885×10^{-7}	1.89409×10^{-7}
1024	1.77613×10^{-7}	1.02341×10^{-7}

Table 5. 2-level MLC: scaled fine-mesh maximum errors (52) using L_{27}^h. Here $\alpha = 1.5$ and $\beta = 2$. The $h \to 0$ errors are the same as the barrier errors in the first column of Table 4.

N	$P = 1$	$P = 4$	$P = 5$
256	1.40367×10^{-6}	1.47261×10^{-6}	1.63939×10^{-6}
512	4.32214×10^{-7}	8.68274×10^{-8}	5.91126×10^{-8}
1024	9.11841×10^{-8}	1.18905×10^{-8}	1.17441×10^{-8}

Table 6. 2-level MLC: scaled fine-mesh maximum errors (52) using L_{27}^h. Here $\alpha = 1.5$ and $\beta = 3.25$. The barrier errors are comparable with those using L_{19}^h with $\beta = 6$ (Table 3).

the values of the error for the MLC-0 calculations with comparable computational costs, i.e., for $\beta = 1.5$. It is clear that for the negligible cost of adding the Legendre expansion, we obtain a decrease in the error by 1–3 orders of magnitude.

Next, we present the errors obtained by performing similar runs using the L_{27}^h Mehrstellen Laplacian, for which $q = 6$. We set $b = 3 \to q_I = 8$ so that dependence of the interpolation error as a function of α and β matches that of the other error terms. In this case, the barrier error is $O(\beta^{-6})$; hence, we expect that smaller values of the β correction radius are required to obtain errors similar to those obtained with the L_{19}^h difference operator. Since $3^4 \approx 2^6$ and $6^4 \approx 3.25^6$, we set $\beta = 2, 3.25$. First, in order to estimate the barrier values, we present the fine-mesh errors for the MLC-0 method in Table 4 with $\beta = 2, 3.25$ using the L_{27}^h operator. With those values of β, we expect errors comparable to or smaller than those of the MLC-0 method with $\beta = 3, 6$ using the L_{19}^h operator. This is the case, as is evident from a comparison with the error values of Table 1. Furthermore, the barrier error as a function of β decreases by more than the factor of $18.4 = (3.25/2)^6$ predicted by the analysis.

N	$P=1$	$P=4$	$P=6$	$P=9$
256	1.91470×10^{-7}	1.96490×10^{-7}	6.39837×10^{-8}	4.90745×10^{-8}
512	5.42412×10^{-8}	9.16574×10^{-9}	5.99534×10^{-9}	6.02843×10^{-9}
1024	1.40428×10^{-8}	2.79547×10^{-9}		

Table 7. 2-level MLC: scaled fine-mesh maximum errors (52) using L^h_{27}. Here $\alpha = 1.75$ and $\beta = 3.25$. Compare with the second column of Table 4. A high polynomial degree is required to attain it for $h = \frac{1}{256}$.

N	$\beta=2.0$	$\beta=3.25$
512	1.30594×10^{-7}	4.86092×10^{-9}
1024	1.90632×10^{-7}	3.92874×10^{-9}

Table 8. 3-level MLC-0: scaled fine-mesh maximum errors (52) using the L^h_{27} Mehrstellen Laplacian. Compare with Table 4, which contains the two-level results.

In Tables 5 and 6, we present the errors for the MLC algorithm, for the cases $\beta = 2, 3.25$; $\alpha = 1.5$ for both cases. The $\beta = 2$ calculations reach the same barrier errors as h decreases. That is not the case for the $\beta = 3.25$ results in Table 4, but that is not surprising — the reduction of the barrier error by nearly an order of magnitude provides more headroom for h-convergence. However, we see that in Table 7 a slight increase of the inner correction radius to $\alpha = 1.75$ allows us to reach the barrier error more rapidly. This is consistent with the error analysis, in that increasing α reduces the coefficient in front of the $O(h^P)$ error from truncating the Legendre expansion, from which we infer that the error from that source, rather than the error from the inner local convolution, is the dominant h-dependent error for this smooth example.

7.1.2. *Three-level results.* We next present similar results using the multilevel MLC algorithm of Section 5 with three levels. Since we have demonstrated a clear advantage to using the 27-point stencil, in the remaining studies we will restrict our attention to that operator. In Table 8 we show the barrier fine-mesh errors obtained using the MLC-0 method for $\beta = 2, 3.25$. The errors for $\beta = 3.25$ are more than 18.4 times smaller than the errors for $\beta = 2$ and are nearly the same as the two-level method errors (Table 4). As predicted by the error analysis in Section 5, the error of MLC-0 is insensitive to the number of levels.

In Table 9 the errors obtained with the three-level MLC method are shown using $\alpha = 1.75$ and $\beta = 3.25$. Unlike the two-level results, the $P = 4$ errors are significantly poorer than the MLC-0 errors. For example, we recover the barrier errors only for $N = 4096$, as opposed to the $N = 512$ results for MLC-0. We can improve matters somewhat by increasing P, but even for this very smooth problem,

N	level	$P=1$	$P=4$	$P=6$	$P=8$
512	$l=0$	1.4509×10^{-7}	1.1379×10^{-7}	5.8886×10^{-8}	4.2602×10^{-8}
	$l=1$	4.9396×10^{-7}	1.0594×10^{-6}	1.0990×10^{-6}	3.0059×10^{-7}
	$l=2$	5.2600×10^{-7}	1.0782×10^{-6}	1.1101×10^{-6}	1.4926×10^{-7}
1024	$l=0$	1.1143×10^{-7}	1.9032×10^{-8}	9.5197×10^{-9}	4.1018×10^{-9}
	$l=1$	2.2539×10^{-7}	1.6461×10^{-7}	9.9491×10^{-8}	2.3381×10^{-8}
	$l=2$	2.3665×10^{-7}	1.6596×10^{-7}	9.9989×10^{-8}	2.3381×10^{-8}
2048	$l=0$	3.8485×10^{-8}	5.7487×10^{-9}	5.0311×10^{-9}	
	$l=1$	5.9923×10^{-8}	1.0143×10^{-8}	5.9864×10^{-9}	
	$l=2$	6.1989×10^{-8}	1.0276×10^{-8}	6.0168×10^{-9}	
4096	$l=0$	1.3028×10^{-8}	5.1364×10^{-9}		
	$l=1$	1.6487×10^{-8}	5.2147×10^{-9}		
	$l=2$	1.6861×10^{-8}	5.2621×10^{-9}		

Table 9. 3-level MLC: scaled maximum errors (52) at all levels using L_{27}^h. Here $\alpha = 1.75$ and $\beta = 3.25$. Compare with the second column of Table 8.

we do not get close to the barrier errors until $N = 2048$. This is consistent with the analysis in Section 5, and indicates that using higher values of P does not solve the problem. We will propose a different solution in Section 7.2.

7.2. ***An oscillatory charge test case.*** We further consider a case of three oscillatory charges that has been previously studied in [25]. The computational domain is again the unit cube $\Omega = [0,1]^3$. Here we define a local charge density, whose support is a sphere of radius R_o centered at point $\boldsymbol{x}_o$, by

$$f_{\boldsymbol{x}_o}(\boldsymbol{x}) = \begin{cases} (1/R_o^3)(r-r^2)^2\sin^2((\gamma/2)r), & r<1,\\ 0, & r\geq 1,\end{cases}$$
$$r = \frac{1}{R_o}\|\boldsymbol{x}-\boldsymbol{x}_o\|,\ \gamma = 4\mu\pi,\ \mu = 7. \quad (54)$$

The exact solution associated with this charge density is given by

$$\phi_{\boldsymbol{x}_o}(\boldsymbol{x}) = \frac{1}{R_o}\begin{cases} -\frac{1}{120}-\frac{6}{\gamma^4}, & r=0,\\ \frac{r^6}{84}-\frac{r^5}{30}+\frac{r^4}{40}+\frac{60}{\gamma^6}-\frac{9}{\gamma^4}-\frac{1}{120}+\frac{120}{\gamma^6 r} & \\ \quad+\big(-\frac{120}{\gamma^6 r}-\frac{9}{\gamma^4}+\frac{300}{\gamma^6}+\frac{36r}{\gamma^4}+\frac{r^2}{2\gamma^2}-\frac{30r^2}{\gamma^4}-\frac{r^3}{\gamma^2}+\frac{r^4}{2\gamma^2}\big)\cos(\gamma r) & \\ \quad+\big(\frac{12}{\gamma^5 r}-\frac{360}{\gamma^7 r}-\frac{96}{\gamma^5}+\frac{120r}{\gamma^5}-\frac{3r}{\gamma^3}+\frac{8r^2}{\gamma^3}-\frac{5r^3}{\gamma^3}\big)\sin(\gamma r), & r<1,\\ \big(-\frac{1}{210}-\frac{12}{\gamma^4}+\frac{360}{\gamma^6}\big)\frac{1}{r}, & r\geq 1,\end{cases}$$

and is a pure monopole for $r \geq 1$. For our test case we consider three charges of the form (54), of radius $R_o = \frac{5}{100}$, centered at points $\boldsymbol{c}_1 = (\frac{3}{16}, \frac{7}{16}, \frac{13}{16})$, $\boldsymbol{c}_2 = (\frac{7}{16}, \frac{13}{16}, \frac{3}{16})$, and $\boldsymbol{c}_3 = (\frac{13}{16}, \frac{3}{16}, \frac{7}{16})$. The total charge and total potential are given via linear

N	level	error
2048	$l=0$	9.59918×10^{-7}
	$l=1$	1.00600×10^{-6}
	$l=2$	1.04402×10^{-6}
4096	$l=0$	5.82005×10^{-8}
	$l=1$	6.47409×10^{-8}
	$l=2$	6.71067×10^{-8}
8192	$l=0$	8.42867×10^{-9}
	$l=1$	8.42867×10^{-9}
	$l=2$	8.44657×10^{-9}

Table 10. 3-level MLC-0: scaled maximum errors (52) using the L_{27}^h Mehrstellen Laplacian with $\beta = 3.25$.

N	level	error
2048	$l=0$	1.03645×10^{-7}
	$l=1$	9.59723×10^{-7}
	$l=2$	1.00621×10^{-6}
	$l=3$	1.04423×10^{-6}
4096	$l=0$	2.93837×10^{-8}
	$l=1$	5.84863×10^{-8}
	$l=2$	6.50247×10^{-8}
	$l=3$	6.73912×10^{-8}
8192	$l=0$	7.84890×10^{-9}
	$l=1$	8.78853×10^{-9}
	$l=2$	8.78853×10^{-9}
	$l=3$	8.79911×10^{-9}

Table 11. 4-level MLC-0: scaled maximum errors (52) using L_{27}^h with $\beta = 3.25$.

superposition by

$$f(\boldsymbol{x}) = f_{\boldsymbol{c}_1}(\boldsymbol{x}) + f_{\boldsymbol{c}_2}(\boldsymbol{x}) + f_{\boldsymbol{c}_3}(\boldsymbol{x}),$$
$$\phi(\boldsymbol{x}) = \phi_{\boldsymbol{c}_1}(\boldsymbol{x}) + \phi_{\boldsymbol{c}_2}(\boldsymbol{x}) + \phi_{\boldsymbol{c}_3}(\boldsymbol{x}).$$

We first present the results using three levels (Table 10) and four levels (Table 11) using MLC-0. The primary features of the convergence properties of the solution are that the errors are nearly uniform as a function of level, and are the same in both the three- and four-level cases. There is some indication of slowing down of the convergence rate on the finest two levels, but the convergence is still faster than $O(h^2)$.

N	level	$P=1$	$P=4$
2048	$l=0$	1.09448×10^{-7}	1.07739×10^{-7}
	$l=1$	9.60320×10^{-7}	9.57456×10^{-7}
	$l=2$	1.00544×10^{-6}	1.00767×10^{-6}
	$l=3$	1.04414×10^{-6}	1.04686×10^{-6}
4096	$l=0$	3.58565×10^{-8}	3.03039×10^{-8}
	$l=1$	5.81436×10^{-8}	7.22707×10^{-8}
	$l=2$	6.55356×10^{-8}	6.44269×10^{-8}
	$l=3$	6.74960×10^{-8}	6.95632×10^{-8}
8192	$l=0$	2.88555×10^{-8}	1.92320×10^{-8}
	$l=1$	1.61302×10^{-7}	7.46182×10^{-8}
	$l=2$	1.63846×10^{-7}	7.61412×10^{-8}
	$l=3$	1.64401×10^{-7}	7.61592×10^{-8}

Table 12. 4-level MLC: scaled maximum errors (52) using L_{27}^h. Here $\alpha=2.25$ and $\beta=3.25$.

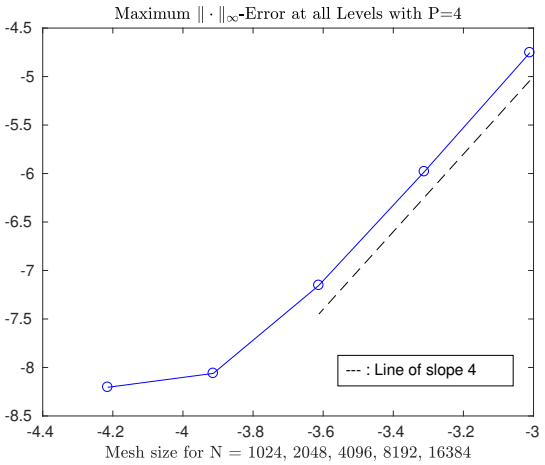


Figure 3. Log-log plot of greatest max norm error at all levels against mesh size using the L_{27}^h Mehrstellen Laplacian. Here fourth-order Legendre polynomials are employed at level 3. For levels 1 and 2, $\alpha=\beta=3.25$, and $\alpha=2.25$ at level 3.

In the MLC convergence results in Table 12, we see substantial deviations from the MLC-0 convergence results. The error shows no consistent behavior as a function of resolution, and in fact is worse at the finest resolution ($N=8192$) in Table 12 than it is at the $N=4096$ resolution in Table 11. We see no analogous problems in the MLC-0 calculations. Examining the error analysis in Section 5, we identified the terms in a three-level calculation that might lead to problems. Even in the smooth example above, it is clear that the increasing P does not have sufficient impact to solve this problem. A different approach, suggested by the form of the error, is to

N	level	α_l	$P=1$	$P=4$
1024	$l=0$		6.75815×10^{-8}	6.75467×10^{-8}
	$l=1$	3.25	9.06867×10^{-6}	9.07109×10^{-6}
	$l=2$	3.25	1.68260×10^{-5}	1.68313×10^{-5}
	$l=3$	2.25	1.76069×10^{-5}	1.76196×10^{-5}
2048	$l=0$		1.03740×10^{-7}	1.03725×10^{-7}
	$l=1$	3.25	9.59547×10^{-7}	9.59794×10^{-7}
	$l=2$	3.25	1.00638×10^{-6}	1.00564×10^{-6}
	$l=3$	2.25	1.04443×10^{-6}	1.04435×10^{-6}
4096	$l=0$		2.96831×10^{-8}	2.95118×10^{-8}
	$l=1$	3.25	5.81150×10^{-8}	5.83051×10^{-8}
	$l=2$	3.25	6.45808×10^{-8}	6.48388×10^{-8}
	$l=3$	2.25	6.86291×10^{-8}	7.02081×10^{-8}
8192	$l=0$		7.52964×10^{-9}	7.73906×10^{-9}
	$l=1$	3.25	8.60687×10^{-9}	8.68798×10^{-9}
	$l=2$	3.25	8.72734×10^{-9}	8.68798×10^{-9}
	$l=3$	2.25	8.64239×10^{-9}	8.68813×10^{-9}
16384	$l=0$		5.94183×10^{-9}	5.97157×10^{-9}
	$l=1$	3.25	6.16822×10^{-9}	6.20416×10^{-9}
	$l=2$	3.25	6.20010×10^{-9}	6.24059×10^{-9}
	$l=3$	2.25	6.21301×10^{-9}	6.24325×10^{-9}

Table 13. 4-level MLC: scaled maximum errors (52) using L^h_{27} with higher values of α at intermediate levels. Here $\beta=3.25$ and $\alpha=\beta$ at levels 1 and 2 and $\alpha=2.25$ at level 3. Compare with Table 11.

reduce the difference $\beta-\alpha$ at coarser levels. In fact, there is likely a mechanism for defining a systematic strategy for doing this, since $(\mathbb{I}-\mathbb{P})f^i$ is easily computed. We defer that to later work. For the moment, we demonstrate this by setting $\alpha=\beta$ at coarser levels, holding β fixed (Table 12). We see that we can recover exactly the errors in the MLC-0 calculation and moreover there is no appreciable difference in error by increasing P. In addition, the cost of increasing α at coarser levels has a small impact on the overall cost of a multiresolution calculation, since these are applied to calculations at the coarser resolutions, which remain a small fraction of the overall cost of the method, even with the increased values of α. In Figure 3 we present the error behavior for the case of Table 13 with $P=4$. For $N=1024$–4096 the error is fourth-order accurate as is expected from the error estimate (45) where term $O(h^4)$ dominates at coarser mesh resolutions. For $N\geq 8192$ the error reaches a plateau imposed by the barrier error term $O(\|f\|_\infty/\beta^q)$. This can be reduced

further by employing higher-order Mehrstellen discretizations of the Laplacian or larger values of parameter β.

8. Conclusions

We have presented a domain decomposition method for the numerical solution of Poisson's equation with infinite domain boundary conditions in three dimensions on a nested hierarchy of structured grids. The method is an extension of Anderson's method of local corrections for particles [3] to gridded data and generalizes the scheme of McCorquodale et al. [25]. In the present method, local potentials are computed as volume potentials of local charges up to an inner localization radius, combined with volume potentials induced by order-$(P-1)$ truncated Legendre expansions of the local charges up to an outer localization radius. The remaining global coupling is represented using a coarse-grid version of the same representation. This generalizes the method in [25], which corresponds to the $P = 1$ special case in the current method. Also, in [25] the local potentials were computed by means of the James–Lackner representation [17; 18] of infinite domain boundary conditions. In the present work, this is replaced by a representation using discrete convolution operators, which can be computed efficiently using FFTs via Hockney's algorithm. This approach eliminates the complicated quadratures that are necessary for the extension of the James–Lackner algorithm to three dimensions, while the FFT-based approach leads to compact compute kernels that can be highly optimized. The resulting algorithm is well suited for high performance on HPC computing platforms made up of multicore processors; in [24], we will present a systematic study of the performance and scaling of the algorithm on such systems.

In this paper, we have focused primarily on the analytical foundations of the MLC method and have provided a detailed error analysis. The errors are of the form $O(h^P) + O(h^4) + O(h^2\beta^{-q}) + O(\beta^{-q})$, where h is the mesh spacing, β is the nondimensionalized outer localization radius which is independent of h, and q is the order of accuracy of the Mehrstellen operator on harmonic functions. Numerical experiments indicate that the observed convergence behavior of the method is consistent with the analysis. For computationally practical values of the localization radius, and using the 27-point Mehrstellen operator (for which $q = 6$), the barrier error corresponds to relative solution error norms of 10^{-8}–10^{-9}. While the β^{-q} term looks like an $O(1)$ error relative to the mesh spacing h, it is better to think of it as a separate discretization parameter that governs the accuracy of the representation of the nonlocal coupling. Doubling β decreases the error by a factor of 2^{-q}, analogous to the impact of halving h.

For the two-level algorithm, the results indicate that, for a given choice of the Mehrstellen operator, the two localization radii, and $P = 4$, the method converges at

a rate in the range $O(h^4)$–$O(h^2)$, until the error reaches the barrier, i.e., consistent with the error analysis. We have also defined and implemented the extension to more than two levels, following the approach in [25]. A preliminary analysis of that algorithm indicates the need to control errors at coarser levels coming from the field induced between the inner and outer localization radii by the truncation of the Legendre expansion. The analysis suggests that these might be controlled by increasing the inner localization radius α at coarser levels. The numerical examples indicate that the problem is real, and that the proposed solution represents a viable approach. More generally, an important question that needs to be addressed is turning the error analysis in this work into practical strategies for choosing discretization parameters. For example, what are the tradeoffs between decreasing $\beta - \alpha$ and decreasing h in order to improve the accuracy of a calculation, versus the cost of doing each? We will address these issues in [24].

There are various possible ways to extend the present work. Perhaps most straightforward are extensions to finite volume discretizations and the implementation of other boundary conditions on rectangular domains (including periodic boundary conditions) using a method-of-images approach. Another possibility would be to apply even higher-order Mehrstellen discretizations of the Laplacian to see whether it results in smaller values of the barrier errors than those reported in this work. As was seen in Section 7, the L^h_{27} ($q = 6$) Mehrstellen Laplacian leads to comparable barrier errors to those obtained using the L^h_{19} ($q = 4$) stencil, but using smaller localization radii, in a manner consistent with the $O(\beta^{-q})$ scaling of that error. It is possible to derive Mehrstellen stencils for which $q = 10$, with the stencil contained in a $5 \times 5 \times 5$ block around the evaluation point. This leads to only a modest increase in the computational cost and complexity: for example, the per patch computational cost of the most computationally intensive component of the algorithm — the local discrete convolutions — does not depend on the size of the stencil. Finally, it would be interesting to investigate extensions of this method to other elliptic problems in mathematical physics employing different Green's functions and high-order discretizations of the associated differential operators. The error analysis of the method as extended to other kernels should be essentially the same as what is discussed in the present study. Moreover, Hockney's algorithm is kernel-independent and can be readily applied with minor modifications. More generally, the present work uses some detailed analytic tools for understanding the discrete potential theory on locally structured grids associated with the combination of finite difference localization in [23] and the local interactions/local corrections construction underlying [3]. It would be interesting to go back to the original MLC method for particles and to other particle-grid methods, such as particle-in-cell and immersed boundary methods, and apply these tools to better understand the error properties of these methods.

Appendix A: L_{19}^h and L_{27}^h Mehrstellen discretizations of the Laplacian

The stencil coefficients for the L_{19}^h and L_{27}^h Mehrstellen Laplacians are $a_{\boldsymbol{j}} = (1/h^2)b_{|\boldsymbol{j}|}$, where $|\boldsymbol{j}|$ is the number of nonzero components of $\boldsymbol{j}$ and b_k are defined as

$$b_0 = -4, \quad b_1 = \tfrac{1}{3}, \quad b_2 = \tfrac{1}{6}, \quad b_3 = 0 \quad \text{(19-point stencil)},$$
$$b_0 = -\tfrac{64}{15}, \quad b_1 = \tfrac{7}{15}, \quad b_2 = \tfrac{1}{10}, \quad b_3 = \tfrac{1}{30} \quad \text{(27-point stencil)}.$$

The corresponding expressions for the truncation errors τ_{19}^h and τ_{27}^h for L_{19}^h and L_{27}^h, are given by

$$\tau_{19}^h(\phi) = \frac{h^2}{12}(\Delta(\Delta\phi)) + h^4 L^{(6)}(\phi) + O(h^6)$$

and

$$\tau_{27}^h(\phi) = \frac{h^2}{12}(\Delta(\Delta\phi)) + \frac{h^4}{360}\left(\left(\Delta^2 + 2\left(\frac{\partial^4}{\partial x^2 \partial y^2} + \frac{\partial^4}{\partial y^2 \partial z^2} + \frac{\partial^4}{\partial z^2 \partial x^2}\right)\right)(\Delta\phi)\right) + h^6 L^{(8)}(\phi) + O(h^8)$$

where $L^{(q)}$ are homogeneous constant-coefficient q-th-order differential operators.

We need to compute an approximation to the discrete Green's function (8) for the 19-point and 27-point operators, restricted to a domain of the form $D = [-n, n]^3$. We do this by solving the following inhomogeneous Dirichlet problem on a larger domain $D_\zeta = [-\zeta n, \zeta n]^3$:

$$(L^{h=1}G^{h=1})[\boldsymbol{g}] = \delta_{\boldsymbol{0}}[\boldsymbol{g}] \quad \text{for } \boldsymbol{g} \in \mathcal{G}(D_\zeta, -1),$$
$$G^{h=1}[\boldsymbol{g}] = G(\boldsymbol{g}) \quad \text{for } \boldsymbol{g} \in D_\zeta - \mathcal{G}(D_\zeta, -1),$$

where $G = G(\boldsymbol{x})$ is the Green's function (2) and L^h is either the 19-point or 27-point operator. Then our approximation to $G^{h=1}$ on D is the solution computed on D_ζ, restricted to D. To compute this solution, we put the inhomogeneous boundary condition into residual-correction form and solve the resulting homogeneous Dirichlet problem using the discrete sine transform. The error estimate (12) applied here implies that the error in replacing the correct discrete boundary conditions with those of the exact Green's function scales like $O((\zeta n)^{-4})$ in max norm. In the calculations presented here, we computed $G^{h=1}$ using $n \geq 128$ and $\zeta = 2$, leading to at least 10 digits of accuracy for $G^{h=1}$.

Appendix B: Hockney's method for fast evaluation of discrete convolutions

Hockney [16, pp. 180–181] (see also [9]) observed that discrete convolutions with one of the functions having support on a bounded domain in $\mathbb{Z}^{\boldsymbol{D}}$, and evaluated on a bounded domain, can be computed exactly in terms of discrete Fourier transforms.

For completeness, we describe that method. We show this first for the case $\boldsymbol{D} = 1$, and state the general result for any number of dimensions. Given Ψ, $f : \mathbb{Z} \to \mathbb{R}$, $\text{supp}(f) \subseteq [0, b]$, we want to compute

$$(\Psi * f)[i] = (f * \Psi)[i] = \sum_{j \in \mathbb{Z}} f[i-j]\Psi[j], \quad i \in [0, n]. \tag{55}$$

First, we observe that the infinite sum can be replaced by a finite sum.

$$\sum_{j \in \mathbb{Z}} f[i-j]\Psi[j] = \sum_{j=-b'}^{n} f[i-j]\Psi[j], \quad i \in [0, n], \tag{56}$$

for any $b' \geq b$. Second, we observe that Ψ and f can be replaced in (56) by periodic extensions of those functions restricted to the interval $[-b', n]$:

$$\sum_{j=-b'}^{n} f[i-j]\Psi[j] = \sum_{j=-b'}^{n} \tilde{f}[i-j]\tilde{\Psi}[j], \quad i \in [0, n],$$

$$\tilde{f}[l], \tilde{\Psi}[l] \equiv f[l_{\text{mod}}], \Psi[l_{\text{mod}}], \quad l_{\text{mod}} = \text{mod}(l + b', (n + b' + 1)) - b'. \tag{57}$$

Finally, we express the periodic convolution in (57) in terms of discrete Fourier transforms:

$$\sum_{j=-b'}^{n} \tilde{f}[i-j]\tilde{\Psi}[j] = \mathscr{F}^{-1}(\mathscr{F}(\tilde{\Psi}) \cdot \mathscr{F}(\tilde{f}))[i], \tag{58}$$

where $\mathscr{F}$ and $\mathscr{F}^{-1}$ are the discrete complex Fourier transform and its inverse on the interval $[-b', n] \subset \mathbb{Z}$.

This generalizes to rectangular domains in any number of dimensions. For example, for cubic domains, given Ψ, $f : \mathbb{Z}^{\boldsymbol{D}} \to \mathbb{R}^{\boldsymbol{D}}$, $\text{supp}(f) \subseteq [0, b]^{\boldsymbol{D}}$,

$$\sum_{\boldsymbol{j} \in \mathbb{Z}^{\boldsymbol{D}}} \Psi[\boldsymbol{i} - \boldsymbol{j}] f[\boldsymbol{j}] = \mathscr{F}^{-1}(\mathscr{F}(\tilde{\Psi}) \cdot \mathscr{F}(\tilde{f}))[\boldsymbol{i}], \boldsymbol{i} \in [0, n]^{\boldsymbol{D}}, \tag{59}$$

$$\tilde{f}[\boldsymbol{l}], \tilde{\Psi}[\boldsymbol{l}] \equiv f[\boldsymbol{l}_{\text{mod}}], \Psi[\boldsymbol{l}_{\text{mod}}], \tag{60}$$

$$(\boldsymbol{l}_{\text{mod}})_d = \text{mod}((\boldsymbol{l})_d + b', (n + b' + 1)) - b', \quad d = 0, \ldots \boldsymbol{D} - 1, \tag{61}$$

where $b' \geq b$ and $\mathscr{F}$ and $\mathscr{F}^{-1}$ are the complex discrete Fourier transform and its inverse on the cube $[-b', n]^{\boldsymbol{D}} \subset \mathbb{Z}^{\boldsymbol{D}}$. In practice, this is efficient for a broad range of (b, n) since we can choose b' so that the radices of the FFTs are highly composite, with the size of the problem changing by only a small amount. In the case where $b = n$, the length of the domain doubles in each direction; hence, this is often referred to as Hockney's domain-doubling algorithm. However, in the present application, we want to use the more general case, since the size of the support of the localized charge distributions and the size of the grid on which the local fields are defined differ by a significant amount.

Acknowledgments

The authors would like to thank Brian Van Straalen and Peter McCorquodale for a number of helpful discussions. This research was supported at the Lawrence Berkeley National Laboratory by the Office of Advanced Scientific Computing Research of the U.S. Department of Energy (DOE) under Contract Number DE-AC02-05CH11231 and at the National Energy Research Scientific Computing Center by the DOE Petascale Initiative in Computational Science and Engineering.

References

[1] A. S. Almgren, *A fast adaptive vortex method using local corrections*, Ph.D. thesis, University of California, Berkeley, 1991.

[2] A. S. Almgren, T. Buttke, and P. Colella, *A fast adaptive vortex method in three dimensions*, J. Comput. Phys. **113** (1994), no. 2, 177–200. MR Zbl

[3] C. R. Anderson, *A method of local corrections for computing the velocity field due to a distribution of vortex blobs*, J. Comput. Phys. **62** (1986), no. 1, 111–123. MR Zbl

[4] G. T. Balls and P. Colella, *A finite difference domain decomposition method using local corrections for the solution of Poisson's equation*, J. Comput. Phys. **180** (2002), no. 1, 25–53. MR Zbl

[5] G. T. Balls, *A finite-difference domain decomposition method using local corrections for the solution of Poisson's equation*, Ph.D. thesis, University of California, Berkeley, 1999. MR Zbl

[6] J. Barnes and P. Hut, *A hierarchical $O(N \log N)$ force-calculation algorithm*, Nature **324** (1986), 446–449.

[7] J. Carrier, L. Greengard, and V. Rokhlin, *A fast adaptive multipole algorithm for particle simulations*, SIAM J. Sci. Statist. Comput. **9** (1988), no. 4, 669–686. MR Zbl

[8] L. Collatz, *The numerical treatment of differential equations*, 3rd ed., Die Grundlehren der mathematischen Wissenschaften, no. 60, Springer, 1960. MR Zbl

[9] J. W. Eastwood and D. R. K. Brownrigg, *Remarks on the solution of Poisson's equation for isolated systems*, J. Comput. Phys. **32** (1979), no. 1, 24–38. MR Zbl

[10] L. C. Evans, *Partial differential equations*, 2nd ed., Graduate Studies in Mathematics, no. 19, American Mathematical Society, 2010. MR Zbl

[11] A. Gholami, D. Malhotra, H. Sundar, and G. Biros, *FFT, FMM, or multigrid? A comparative study of state-of-the-art Poisson solvers for uniform and nonuniform grids in the unit cube*, SIAM J. Sci. Comput. **38** (2016), no. 3, C280–C306. MR Zbl

[12] D. Gilbarg and N. S. Trudinger, *Elliptic partial differential equations of second order*, Springer, 2001. MR Zbl

[13] L. Greengard and V. Rokhlin, *A fast algorithm for particle simulations*, J. Comput. Phys. **73** (1987), no. 2, 325–348. MR Zbl

[14] ———, *A new version of the fast multipole method for the Laplace equation in three dimensions*, Acta Numer. **6** (1997), 229–269. MR Zbl

[15] P. Henrici, *Fast Fourier methods in computational complex analysis*, SIAM Rev. **21** (1979), no. 4, 481–527. MR Zbl

[16] R. Hockney, *The potential calculation and some applications*, Method. Comput. Phys. **9** (1970), 135–211.

[17] R. A. James, *The solution of Poisson's equation for isolated source distributions*, J. Computational Phys. **25** (1977), no. 2, 71–93. MR Zbl

[18] K. Lackner, *Computation of ideal MHD equilibria*, Comput. Phys. Commun. **12** (1976), no. 1, 33–44.

[19] M. H. Langston, L. Greengard, and D. Zorin, *A free-space adaptive FMM-based PDE solver in three dimensions*, Commun. Appl. Math. Comput. Sci. **6** (2011), no. 1, 79–122. MR Zbl

[20] S. Liska and T. Colonius, *A parallel fast multipole method for elliptic difference equations*, J. Comput. Phys. **278** (2014), 76–91. MR Zbl

[21] D. Malhotra and G. Biros, *PVFMM: a parallel kernel independent FMM for particle and volume potentials*, Commun. Comput. Phys. **18** (2015), no. 3, 808–830. MR Zbl

[22] ———, *Algorithm 967: a distributed-memory fast multipole method for volume potentials*, ACM Trans. Math. Software **43** (2016), no. 2, 17. MR Zbl

[23] A. Mayo, *Fast high order accurate solution of Laplace's equation on irregular regions*, SIAM J. Sci. Statist. Comput. **6** (1985), no. 1, 144–157. MR Zbl

[24] P. McCorquodale, P. Colella, B. V. Straalen, and C. Kavouklis, *High-performance implementations of the method of local corrections on parallel computers*, preprint, 2018.

[25] P. McCorquodale, P. Colella, G. T. Balls, and S. B. Baden, *A local corrections algorithm for solving Poisson's equation in three dimensions*, Commun. Appl. Math. Comput. Sci. **2** (2007), 57–81. MR Zbl

[26] W. F. Spotz and G. F. Carey, *A high-order compact formulation for the* 3*D Poisson equation*, Numer. Methods Partial Differential Equations **12** (1996), no. 2, 235–243. MR Zbl

[27] F. Vico, L. Greengard, and M. Ferrando, *Fast convolution with free-space Green's functions*, J. Comput. Phys. **323** (2016), 191–203. MR

Received October 5, 2016. Revised July 14, 2018.

CHRIS KAVOUKLIS: kavouklis1@llnl.gov
Computational Engineering Division, Lawrence Livermore National Laboratory, Livermore, CA, United States

PHILLIP COLELLA: pcolella@lbl.gov
Applied Numerical Algorithms Group, Computational Research Division, Lawrence Berkeley National Laboratory, Berkeley, CA, United States

COMM. APP. MATH. AND COMP. SCI.
Vol. 14, No. 1, 2019
dx.doi.org/10.2140/camcos.2019.14.33

msp

ON THE CONVERGENCE OF SPECTRAL DEFERRED CORRECTION METHODS

MATHEW F. CAUSLEY AND DAVID C. SEAL

In this work we analyze the convergence properties of the spectral deferred correction (SDC) method originally proposed by Dutt et al. (BIT **40** (2000), no. 2, 241–266). The framework for this high-order ordinary differential equation (ODE) solver is typically described as a low-order approximation (such as forward or backward Euler) lifted to higher-order accuracy by applying the *same* low-order method to an error equation and then adding in the resulting defect to correct the solution. Our focus is not on solving the error equation to increase the order of accuracy, but on rewriting the solver as an iterative Picard integral equation solver. In doing so, our chief finding is that it is not the low-order solver that picks up the order of accuracy with each correction, but it is the underlying quadrature rule of the right-hand-side function that is solely responsible for picking up additional orders of accuracy. Our proofs point to a total of three sources of errors that SDC methods carry: the error at the current time point, the error from the previous iterate, and the numerical integration error that comes from the total number of quadrature nodes used for integration. The second of these two sources of errors is what separates SDC methods from Picard integral equation methods; our findings indicate that as long as the difference between the current and previous iterates always gets multiplied by at least a constant multiple of the time step size, then high-order accuracy can be found even if the underlying ODE "solver" is inconsistent. From this vantage, we solidify the prospects of extending spectral deferred correction methods to a larger class of solvers, of which we present some examples.

1. Introduction

The spectral deferred correction (SDC) method defines a large class of ordinary differential equation (ODE) solvers that were originally introduced in 2000 by Dutt, Greengard, and Rokhlin [12]. These types of methods are typically introduced by

We would like to thank the anonymous referees for their thoughtful comments, suggestions that improved the quality of this manuscript, and recommendations for future research. The work of Seal was supported by the Naval Academy Research Council.

MSC2010: 65L05, 65L20.

Keywords: initial-value problems, spectral deferred correction, Picard integral, semi-implicit methods.

defining an *error equation*, and then repeatedly applying the same low-order solver to the error equation and adding the solution back into the current approximation in order to pick up an order of accuracy. This idea can be traced back to the work of Zadunaisky in 1976 [40], who sought out high-order solvers in order to reduce numerical roundoff errors for astronomical applications. Before introducing the classical SDC methods defined in [12], we stop here to point out some of the recent work that has been happening over the past two decades including [30; 28; 6; 26; 7; 4; 22]. We refer the interested reader to [32] for a nice list of references for the first of these last two decades. Here, we provide a sampling of some of the current topics of interest to the community.

Many variations of the original SDC method are being studied as part of an effort to expedite the convergence of the solver. The chief goal here is to reduce the total number of iterations required to obtain the same high-order accuracy of the original method. These methods include the option of using Krylov deferred correction methods [20; 21] as well as the *multilevel* SDC methods [27; 36]. The multilevel approach starts with a lower-order interpolant and then successively increases the degree of the interpolant with each future sweep of the method. This has the primary advantage of decreasing the overall number of function evaluations that need to be conducted, but introduces additional complications involving the need to evaluate interpolating polynomials. In the same vein, higher-order embedded integrators have been explored within the so-called integral deferred correction (IDC) framework [6; 7], where a moderate order solver (such as second- or fourth-order Runge–Kutta method) is embedded inside a very high-order SDC solver. With this framework, each successive correction increases the order by the same amount as that of the base solver. In addition, parallel in time solvers [9; 31; 5; 13] are being investigated as a mechanism to address the needs of modern high-performance computing architectures, and adaptive time stepping options have been more recently investigated in [10]. This work is based upon the nice property that SDC methods naturally embed a lower-order solver inside a higher-order solver.

In addition to the above mentioned extensions, various semi-implicit formulations have been, and are currently being, explored. While the original solver was meant for classical nonlinear ODEs, semi-implicit formulations have been derived as early as 2003 [30] and are still an ongoing topic of research [29; 4]. The effect of the choice of correctors including second-order semi-implicit solvers for the error equation has been researched in [25], and an investigation into the efficiency of semi-implicit and multi-implicit spectral deferred correction methods for problems with varying temporal scales has been conducted in [26]. Related high-order operator splitting methods have been proposed in [15; 3; 8], where the focus is not on an implicit-explicit splitting, but rather on splitting the right-hand side of the ODE into smaller systems that can be more readily inverted with each sweep of the solver.

Very recent work includes applications of the SDC framework to generate exponential integrators of arbitrary orders [2], exploring interesting LU decompositions of the implicit Butcher tableau on nonequispaced grids [38], further investigation into high-order operator splitting [11], a comparison of essentially nonoscillatory (ENO) versus piecewise parabolic methods (PPM) coupled with SDC time integrators [23], and additional implicit-explicit (IMEX) splittings for fast-wave slow-wave splitting constructed from within the SDC framework [35].

It is not our aim to conduct a comprehensive review and comparison of all of these methods; rather it is our goal to present rigorous analysis of the original method that can be extended to these more complicated solvers. With that in mind, we now turn to a brief introduction of the spectral deferred correction framework, and in the process of doing so, we seek to directly compare this method with that of the Picard integral formulation of a numerical ODE solver.

1A. ***Picard iteration and the SDC framework.*** We begin by giving a brief description of classical SDC methods. In doing so, we explain the differences between SDC and Picard iteration, which defines the cornerstone of the present work.

Classical SDC solvers are designed to solve initial value problems of the form

$$y' = \frac{dy}{dt} = f(y), \quad t > 0, \ y(0) = y_0, \tag{1}$$

where y can be taken to be a vector of unknowns. The solution $y(t)$ can be expressed as an integral through formal integration:

$$y(t) = y_0 + \int_0^t f(y(s))\, ds, \quad t > 0. \tag{2}$$

In this work, we assume that f is Lipshitz continuous. That is, we assume

$$|f(z) - f(w)| \le L|z - w|, \tag{3}$$

for some constant $L \ge 0$ and all $z, w \in \mathbb{R}$. This is sufficient to guarantee existence and uniqueness for solutions of IVP (1), and produce rigorous numerical error bounds for SDC methods.

Consider a set of M quadrature points $0 \le \xi_1 < \cdots < \xi_M \le 1$ that partition the unit interval into a total of N disjoint subintervals, defined by

$$N = \begin{cases} M - 1 & \text{if both endpoints are used,} \\ M & \text{if only one endpoint is used,} \\ M + 1 & \text{if neither endpoint is used.} \end{cases}$$

We make this choice because a given quadrature rule may or may not include the endpoints of the interval, and this convention allows us to study Gaussian quadrature rules, uniformly spaced quadrature rules, Radau II quadrature rules, and others

all within the same context. With that in mind, we define the right endpoints ξ_n^R, for $n = 0, 1, \ldots N-1$, of each of the N subintervals as

$$\xi_n^R = \begin{cases} \xi_{n+1} & \text{if the left endpoint is included,} \\ \xi_n & \text{if the left endpoint not included,} \end{cases}$$

and $\xi_0^R = 0$ and $\xi_N^R = 1$ for the two boundary edge cases. Next, we define quadrature weights by

$$w_{n,m} = \int_{\xi_{n-1}^R}^{\xi_n^R} \ell_m(x)\,dx, \quad n = 1, 2, \ldots N,\ m = 1, 2, \ldots M, \tag{4}$$

where $\ell_m(x)$ is the Lagrange interpolating polynomial of degree at most $M-1$ corresponding to the quadrature point ξ_m:

$$\ell_m(x) = \frac{1}{c_m} \prod_{k=1,\, k\neq m}^{M} (x - \xi_k), \qquad c_m = \prod_{k=1,\, k\neq m}^{M} (\xi_m - \xi_k). \tag{5}$$

Once these weights are obtained, approximate integral solutions, say $\eta_m \approx y(\xi_m^R h)$ for $h > 0$ and $m = 0, 1, \ldots, N$, can be formed via

$$\text{(fully implicit collocation)} \quad \eta_n = \eta_{n-1} + h \sum_{m=1}^{M} w_{n,m} f(\eta_m), \quad n = 1, 2, \ldots, N, \tag{6}$$

whereas the exact solution $y_m := y(\xi_m^R h)$ satisfies the exact integral

$$y_n = y_{n-1} + \int_{t_{n-1}}^{t_n} f(y(t))\,dt, \quad t_n = \xi_n^R h,\ n = 1, 2, \ldots, N. \tag{7}$$

By convention, $\eta_0 := y_0$ is known to high order (because it comes from the previous time step), and $\eta_N \approx y(h)$ constitutes one "full" time step. Since each substep uses information from all substeps to construct the right-hand side, the solution is higher order, but also requires the solution of a nonlinear system of M unknowns (one for each quadrature point) at each time step. Although this integrator has some very nice properties (e.g., it can be made to be symplectic and L-stable for suitably chosen quadrature points), it is not typically used in practice given the additional storage requirements and the larger matrices that need to be inverted for each time step. This is particularly relevant when it is used as the base solver for a partial differential equation, but even these bounds are being explored as a viable option for PDE solvers such as the discontinuous Galerkin method [33].

In place of the fully implicit collocation method, Picard iteration (with numerical quadrature) defines a solver by iterating on a current solution $\eta_n^{[p]}$, $p \in \mathbb{Z}_{\geq 0}$, and

then creates a better approximation through

$$\text{(Picard iteration)} \qquad \eta_n^{[p+1]} = \eta_{n-1}^{[p+1]} + h \sum_{m=1}^{M} w_{n,m} f\left(\eta_m^{[p]}\right), \quad n = 1, 2, \ldots N. \quad (8)$$

Note that the current value $\eta_0^{[p+1]} := \eta_0 \approx y_0$ is a known value that is equal to the exact solution up to high order. While this solver picks up a single order of accuracy with each correction, it has the unfortunate consequence of having a finite region of absolute stability.

The explicit spectral deferred correction framework is

$$\text{(explicit SDC)} \ \eta_n^{[p+1]} = \eta_{n-1}^{[p+1]} + h_n\left[f\left(\eta_{n-1}^{[p+1]}\right) - f\left(\eta_{n-1}^{[p]}\right)\right] + h\sum_{m=1}^{M} w_{n,m} f\left(\eta_m^{[p]}\right), \ (9)$$

where $h_n = (\xi_n^R - \xi_{n-1}^R)h$ is the length of the n-th subinterval. This solver also has a finite region of absolute stability.

Remark. Although traditional SDC methods were originally cast as a method that corrects a provisional solution by solving an error equation, some modern descriptions of the same solver identify (9) as the base solver, which has the added benefit of pointing out a solid link between SDC methods and iterative Picard integral equation solvers.

In order to construct methods that have more favorable regions of absolute stability for stiff problems, the implicit SDC framework exacts multiple backward Euler time steps through each iteration with

$$\text{(implicit SDC)} \ \eta_n^{[p+1]} = \eta_{n-1}^{[p+1]} + h_n\left[f\left(\eta_n^{[p+1]}\right) - f\left(\eta_n^{[p]}\right)\right] + h\sum_{m=1}^{M} w_{n,m} f\left(\eta_m^{[p]}\right). \ (10)$$

Note that this framework allows for implicit and high-order solutions to be constructed with greater computational efficiency when compared to the fully implicit collocation solver defined in (6) because smaller systems need to be inverted in order to take a single time step.

Remark. It has been noted that the scaling in front of the h_n term does not have impact on the order of accuracy [39]. It is our aim with this work to solidify that claim with rigorous numerical bounds, which we do for both the explicit and implicit solvers.

Before doing so, we point out an aside that is in common with all SDC solvers.

Remark. If $\lim_{p\to\infty} \eta_n^{[p]} = \eta_n$ converges, then solutions to (8), (9), and (10) converge to that of the fully implicit collocation method defined in (6).

While some SDC methods work with a fixed number of iterates in order to obtain a desired order of accuracy, there are many examples in the literature where convergence of the SDC iterations to the fully implicit scheme is considered. For example, the work in [38] is wholly concerned with this convergence, and not the accuracy of the underlying method for fixed iterations. Moreover, for stiff problems, it is well understood that using a fixed number of iterations can lead to order reduction which negates the advantage of using SDC in the first place. In addition, the multilevel spectral deferred correction (MLSDC) methods also are typically iterated to a residual tolerance, since one cannot be sure that coarse level sweeps will provide enough increase in accuracy (or decrease in the residual) [36].

One key advantage of iterating an SDC method to convergence is that when this is done, the method inherits well known and desirable properties that the fully implicit collocation method enjoys. For example, Kuntzmann [24] and Butcher [1] separately point out that if a total of M Gaussian quadrature points are used, then the fully implicit collocation method will have superconvergence order $\mathbb{O}(h^{2M})$. (For more details, we refer the interested reader to the excellent tomes of Hairer, Wanner et al. [16; 17; 18]. For example, see [16, §II.7], [17, Theorem 5.2], or [18, Theorem 1.5].) In general, the maximum order of accuracy for the underlying solver with M quadrature points is $\mathbb{O}(h^{2M})$ if they are Gauss–Legendre points, $\mathbb{O}(h^{2M-1})$ for the RadauIIA points, and $\mathbb{O}(h^{2M-2})$ for Gauss–Lobatto points. (The local truncation error is one order higher.) Uniform points have $\mathbb{O}(h^M)$ order of convergence if M is even, and $\mathbb{O}(h^{M+1})$ if M is odd. The extra pickup in the order of accuracy is due to symmetry of the quadrature rule. (For example, $M = 1$ points reproduces the so-called "midpoint" rule, $M = 3$ reproduces Simpson's rule, and $M = 5$ yields Boole's rule, each of which pick up an extra order of accuracy.)

1B. ***An outline of the present work.*** Despite the increasing popularity of spectral deferred correction solvers, very little work has been performed on convergence results for this large class of methods. The results that are currently in the literature [15; 6; 7; 19; 37] typically proceed via induction on the current order of the approximate solution, and they all hinge on solving the *error equation*, wherein the same low-order solver is applied and then a defect, or correction, is added back into the current solution in order to increase its overall order of accuracy. In other recent work [34], the authors consider SDC methods as fixed-point iterations on a Neumann series expansion. There, the low-order method is viewed as an efficient preconditioner (in numerical linear algebra language), and the SDC iterations are thought of as simplified Newton iterations. Additionally, the work in [38] makes use of linear algebra techniques in order to optimize coefficients so that the method converges faster to the collocation solution for stiff problems.

In this work, we do not require the use of the error equation, nor do we work with any sort of defect such as that defined in [40]; rather we instead focus on the

Picard integral underpinnings inherent to all SDC methods. Our work solely uses fundamental numerical analysis tools: error estimates for numerical interpolation and integration. While these tools do rely on quadrature rules, our proofs are generic enough to accommodate any set of quadrature points, which are an ongoing discussion in terms of how to construct base solvers.

In this work, we prove rigorous error bounds for both implicit and explicit SDC methods, and in doing so, we expect the reader will find that these methods can be thought of as being built upon classical Picard iteration. Our results are applicable for general quadrature rules, but unlike the findings found in [37], where convergence is proven using the error equation, our work relies on the fundamental mechanics behind why the solver works. That is, we point out that the primary contributor to the order of accuracy of the solver lies within the integral of the residual, and not necessarily the application of any base solver to an error equation.

Indeed, our proofs follow in a manner similar to the proof of the Picard–Lindelöf theorem, but our proofs take into account numerical quadrature errors and do not rely on exact integration of the right-hand-side function $f(y)$. The primary differences between our proofs and that of the Picard–Lindelöf theorem are the following:

- Spectral deferred correction methods require the use of *numerical quadrature* to approximate the integrals presented in the Picard–Lindelöf theorem. Our error estimates take into account any errors resulting from quadrature rules.
- Each correction step in the *implicit scheme* defined in (10) requires a nonlinear inversion, whereas the Picard–Lindelöf theorem is typically proven using exact integration.

There are two main results in this work, one for explicit SDC methods and one for implicit SDC methods. These are both found as corollaries to a single theorem on semi-implicit SDC. In each case, we produce rigorous error bounds that are applicable for generic quadrature rules. Furthermore, we find that there are a total of three sources of error that SDC methods carry: the error from the previous time step, the error from the previous iterate, and the error from the quadrature rule being used.

The outline of this paper is as follows. In Section 2, we present some necessary lemmas concerning error estimates for integrals of interpolants as well as some error estimates for sequences of inequalities that show up in our proofs. In Section 3 we present a convergence proof for the more general case of a semi-implicit SDC solver, and then immediately point out two corollaries that prove implicit and explicit SDC methods converge. In Section 4, we present results for an SDC method that makes use of a higher-order base solver, the trapezoidal rule. In Section 5 we present some numerical results, where we compare explicit SDC methods with Picard iterative methods, we investigate modified implicit SDC methods, and we experiment with different semi-implicit formulations of SDC methods. Error estimates for all of

these variants come from direct extensions of the proofs found in this work. Finally, some conclusions and suggestions for future work are drawn up in Section 6.

2. Preliminaries

We now point out a couple of important tools that we use to show that SDC solvers converge. Our aim is to focus on a single time step. Without loss of generality, from here on out we will focus on constructing a solution over the interval $[0, h]$, where h is the time step size and we will assume that $\eta_0 \approx y_0$ is a high-order approximation to the exact solution.

2A. ***Error estimates for integrals of interpolants.*** If $\boldsymbol{\eta} = (\eta_1, \eta_2, \ldots, \eta_M)$ is a set of discrete values and $t \in [0, h]$ is a time interval we are interested in studying, we define the *interpolation operator* I to be the projection onto the space of polynomials of degree at most $M - 1$ via

$$I[f(\boldsymbol{\eta})](t) := \sum_{m=1}^{M} f(\eta_m) l_m(t/h), \quad f(\boldsymbol{\eta}) := (f(\eta_1), f(\eta_2), \ldots, f(\eta_M)). \tag{11}$$

Note that this produces the integration identity

$$\int_{t_{n-1}}^{t_n} I[f(\boldsymbol{\eta})](t)\, dt = h \sum_{m=1}^{M} w_{n,m} f(\eta_m) \tag{12}$$

after integrating (11) over a subinterval $[t_{n-1}, t_n] := [h\xi_{n-1}^R, h\xi_n^R]$, and the weights are defined as in (4).

Convergence results for both the explicit and the implicit SDC method (as well as Picard iteration) require the use of the following lemma.

Lemma 2.1. *Suppose that* $f \circ y \in C^M([0, h])$, $\left\| \frac{d^M}{dt^M}(f \circ y) \right\|_\infty \le F$, *and* f *is Lipschitz continuous with Lipshitz constant* L. *Then we have the estimate*

$$\left| \int_{t_{n-1}}^{t_n} I[f(\boldsymbol{\eta})](t) - f(y(t))\, dt \right| \le h \|\boldsymbol{\eta} - \boldsymbol{y}\| W_n L + \frac{F}{M!} h^{M+1}, \tag{13}$$

where the discrete norm is defined by

$$\|\boldsymbol{e}\| := \max_{1 \le n \le M} |e_n|, \quad \boldsymbol{e} = (e_1, e_2, \ldots, e_M), \tag{14}$$

and the constant W_n *is defined by*

$$W_n := \sum_{m=1}^{M} \int_{\xi_{n-1}^R}^{\xi_n^R} |l_m(\xi)|\, d\xi. \tag{15}$$

For a fixed quadrature rule, this constant is finite and independent of the function.

Proof. Add and subtract the Lagrange interpolant $I[f(\boldsymbol{y})](t)$ for $f \circ y$ inside the left-hand side of (13) and apply the triangle inequality:

$$\left|\int_{t_{n-1}}^{t_n} I[f(\boldsymbol{\eta})](t) - f(y(t))\,dt\right| \le \left|\int_{t_{n-1}}^{t_n} I[f(\boldsymbol{\eta})](t) - I[f(\boldsymbol{y})](t)\,dt\right| + \left|\int_{t_{n-1}}^{t_n} I[f(\boldsymbol{y})](t) - f(y(t))\,dt\right|. \quad (16)$$

An estimate for the first of these two terms follows by linearity of the interpolation operator:

$$\begin{aligned}\left|\int_{t_{n-1}}^{t_n} I[f(\boldsymbol{\eta})](t) - I[f(\boldsymbol{y})](t)\,dt\right| &= \left|h\sum_{m=1}^{M}\omega_{n,m}(f(\eta_m) - f(y_m))\right| \\ &\le h\sum_{m=1}^{M}|\omega_{n,m}||f(\eta_m) - f(y_m)| \\ &\le hL\sum_{m=1}^{M}|\omega_{n,m}||\eta_m - y_m| \\ &\le hL\|\boldsymbol{\eta} - \boldsymbol{y}\|\sum_{m=1}^{M}|\omega_{n,m}|. \end{aligned} \quad (17)$$

The quadrature weights in this estimate are bounded above by

$$|\omega_{n,m}| \le \int_{\xi_{n-1}^R}^{\xi_n^R} |\ell_m(\xi)|\,d\xi$$

and then summed over all m to produce the constant W_n.

The second of the two integrals in (16) is a function solely of the smoothness of f and the choice of the quadrature rule. That is, classical interpolation error estimates result in a bound on the M-th derivative of $f \circ y$ through a single point $z(t) \in [0, h]$ that yields

$$|I[f(\boldsymbol{y})](t) - f(y(t))| = \left|\frac{(f\circ y)^{(M)}(z(t))}{M!}\prod_{m=1}^{M}(t - t_m)\,dt\right| \le \frac{F}{M!}\prod_{m=1}^{M}|t - t_m|. \quad (18)$$

Because $|t - t_m| \le h$ for each m, the result follows after integration. □

We stop to point out that due to the Runge phenomenon, the coefficient W_n defined in (15) can become quite large if a large number of quadrature points are chosen for constructing the polynomial interpolants required for the SDC method. In Table 1, we demonstrate a few sample values when uniform, Chebyshev, Gauss–Legendre, Gauss–Radau, and Gauss–Lobatto quadrature nodes are used to construct

	type of quadrature points				
M	uniform	Chebyshev	Legendre	Gauss–Radau	Gauss–Lobatto
2	1.000	1.207	1.366	1.500	1.000
3	1.000	1.244	1.479	1.558	1.000
4	1.056	1.257	1.527	1.578	1.000
5	1.152	1.263	1.551	1.586	1.000
6	1.257	1.266	1.566	1.591	1.000
7	1.362	1.268	1.575	1.594	1.000
8	1.663	1.269	1.581	1.596	1.000
9	2.550	1.270	1.585	1.597	1.000
10	4.028	1.271	1.588	1.598	1.000
11	6.506	1.271	1.590	1.599	1.000
12	10.963	1.271	1.592	1.599	1.000
13	18.340	1.272	1.594	1.600	1.000
14	32.060	1.272	1.595	1.600	1.000
15	54.998	1.272	1.596	1.600	1.000
16	98.531	1.272	1.596	1.600	1.000
17	172.176	1.272	1.597	1.601	1.000
18	313.675	1.272	1.597	1.601	1.000
19	556.491	1.273	1.598	1.601	1.000
20	1026.313	1.273	1.598	1.601	1.000
30	496210.554	1.273	1.600	1.602	1.000
50	208948162475.383	1.273	1.601	1.602	1.000

Table 1. Maximum size of the Lagrange polynomials $\max_{1\le n\le M}\max_{\xi\in[0,1]}|\ell_n(\xi)|$ for different quadrature points. The Gauss–Legendre, Gauss–Radau, and Gauss–Lobatto quadrature rules with M points have degrees of precision $2M+1$, $2M$, and $2M-1$, respectively.

the polynomial interpolants. Uniform quadrature points tend to start performing quite poorly in the teens; however, even a small amount of points, say five or six, produces a high-order numerical method compared to other ODE solvers because in this regime the error constant is reasonable. The selection of quadrature points that minimizes this portion of the error constant is the Gauss–Lobatto nodes, but because convergence is found through refinement in h rather than p, any of these points will produce a method that converges, provided the exact solution has a suitable degree of regularity.

2B. ***Error estimates for sequences of inequalities.*** Finally, we require a second lemma as well as a simple corollary. Both of these are stated in [14], and their proofs are elementary.

Lemma 2.2. *If $\{a_n\}_{n\in\mathbb{Z}_{\geq 0}}$ is a sequence that satisfies $|a_n| \leq A|a_{n-1}| + B$ with $A \neq 1$, then*

$$|a_n| \leq A^n|a_0| + \frac{A^n - 1}{A - 1}B. \tag{19}$$

Proof. Recursively apply the inequality, and sum the remaining finite geometric series. □

Corollary 2.3. *If $A > 1$ and $\{a_n\}_{n\in\mathbb{Z}}$ is a sequence that satisfies $|a_n| \leq A|a_{n-1}| + B$, then*

$$|a_n| \leq A^n|a_0| + nA^{n-1}B \tag{20}$$

for every n.

Proof. By Lemma 2.2, the sequence satisfies (19). We estimate the (finite) geometric series by

$$\frac{A^n - 1}{A - 1} = 1 + A + \cdots + A^{n-1} \leq nA^{n-1} \tag{21}$$

because there are a total of n terms and each $A^l \leq A^{n-1}$ for $l = 0, 1, \ldots, n-1$. □

With these preliminaries out of the way, we are now ready to state and prove our main result.

3. Convergence results

In place of separately proving explicit and implicit results for (1), we instead consider an umbrella class of ODEs, defined through a semi-implicit formulation:

$$y' = f(y), \quad f(y) = f_I(y) + f_E(y), \ y(0) = y_0, \tag{22}$$

where f_I is to be treated implicitly and f_E is to be treated explicitly. We assume that both f_I and f_E have Lipshitz constants L_I and L_E, respectively. In turn, this implies that f has a Lipshitz constant of $L := L_I + L_E$. In the case where $f_I \equiv 0$, we set $L_I = 0$, and in the case where $f_E \equiv 0$, we set $L_E = 0$.

The classical semi-implicit SDC (SISDC) method for (22) begins with a *provisional solution*, or initial guess $\eta_n^{[0]} \approx y(\xi_n h)$, that is typically defined with

$$\eta_n^{[0]} = \eta_{n-1}^{[0]} + h_n f_I\big(\eta_n^{[0]}\big) + h_n f_E\big(\eta_{n-1}^{[0]}\big), \quad n = 1, 2, \ldots, N, \tag{23}$$

where $h_n = (\xi_n^R - \xi_{n-1}^R)h$. This yields a first-order implicit-explicit (IMEX) predictor for the solution based upon a forward-backward Euler method. Our numerical (and analytical) results indicate the "predictor" step has little bearing on the overall order of accuracy of the solver. For example, it is possible to hold the solution constant for the initial iteration and still obtain high-order accuracy, albeit with one additional iteration.

The classical SISDC method [30] iterates on the provisional solution through

$$\begin{aligned}(\text{SISDC}) \quad \eta_n^{[p+1]} &= \eta_{n-1}^{[p+1]} + h_n\big[f_I\big(\eta_n^{[p+1]}\big) - f_I\big(\eta_n^{[p]}\big)\big] \\ &\quad + h_n\big[f_E\big(\eta_{n-1}^{[p+1]}\big) - f_E\big(\eta_{n-1}^{[p]}\big)\big] + h\sum_{m=1}^{M} w_{n,m} f\big(\eta_m^{[p]}\big), \quad (24)\end{aligned}$$

where $\eta_0^{[p+1]} = \eta_0$ is a known quantity. This value is typically taken to be the result from the previous time step, and we assume that it is known to high-order accuracy. Our focus is on the local truncation error, to which end we assume that the error at time zero is nonzero. That is, we assume $e_0 = \eta_0 - y_0 \neq 0$. Once the single step error is established, a global error can be directly found using textbook techniques. In the event where $f_I \equiv 0$, we end up with the explicit SDC method defined in (9), and when $f_E \equiv 0$, we end up the implicit SDC defined in (10).

We repeat that the collocation method defined in (6) requires simultaneously solving for each η_n and is clearly more expensive than multiple applications of the backward Euler method found in (24), either on part of or the entire right-hand side. In the event where $f_I \equiv 0$, then the method should be less expensive to run for a single time step, but the regions of absolute stability suffer [28; 29]. We also repeat that, provided $\boldsymbol{\eta}^{[p]}$ converges as $p \to \infty$, then (24) defines a solution to (6). Proving which initial guesses converge to the fully implicit solver is beyond the scope of this work. Currently, our aim is to show that each correction step in the SDC framework picks up at least a single order of accuracy to the order predetermined by the quadrature rule.

3A. *Statement of the main result.*

Theorem 3.1. *The errors for a single step of the semi-implicit SDC method satisfy*

$$\big|e_n^{[p+1]}\big| \leq e^{Nh(2L_I+L_E)}|e_0| + C_1 h\big\|\boldsymbol{e}^{[p]}\big\| + C_2 h^{M+1}, \quad (25)$$

provided $hL_I < \frac{1}{2}$, where N is the number of intervals under consideration, $L := L_I + L_E$ is the Lipschitz constant of f, and

$$C_1 = 2Ne^{Nh(2L_I+L_E)}W \quad \text{and} \quad C_2 = 2Ne^{Nh(2L_I+L_E)}\frac{F}{M!}$$

are constants that depend only on f, the exact solution y, and the selection of quadrature points.

In Section 3C we point out two corollaries to this result, one for implicit and one for explicit SDC, but before proving this theorem, we stop to point out an important observation that is applicable to any of the aforementioned methods.

Remark. The statement of this theorem highlights that there are a total of three sources of error that SDC methods admit, which are ordered by appearance in the right-hand side of (25):

(1) the error at the current time step $e_0 = \eta_0 - y_0$,

(2) the error from the previous iterate (or predictor) $\boldsymbol{e}^{[p]} = \boldsymbol{\eta}^{[p]} - \boldsymbol{y}$, and

(3) the number of quadrature points M.

The most important takeaway is that *because the error from the previous iterate, $\|\boldsymbol{e}^{[p]}\|$, gets multiplied by a factor of h, the error gets improved by one order of accuracy with each correction*. Of course this order reaches a maximum order based upon the number of the quadrature points chosen, which can be seen in the third source of error. This can be improved by selecting quadrature points with superconvergence properties such as the Gaussian or Gauss–Lobatto quadrature points. Finally, please note that we make no comment about how the "previous" function values were found. This is intentional because we would like to focus our attention on the impact of what a single correction does to the solution. Doing so permits the analysis to apply to parallel implementations of SDC methods where synchronizations between different correctors (threads) are seldom seen [9; 13].

3B. *Proof of the main result.*

Proof. We subtract the exact equation (7) from (24) and find that the discrete error evolution equation is

$$e_n^{[p+1]} = e_{n-1}^{[p+1]} + h_n\big[f_I\big(\eta_n^{[p+1]}\big) - f_I\big(\eta_n^{[p]}\big)\big] + h_n\big[f_E\big(\eta_{n-1}^{[p+1]}\big) - f_E\big(\eta_{n-1}^{[p]}\big)\big] + \int_{t_{n-1}}^{t_n} I\big[f\big(\boldsymbol{\eta}^{[p]}\big)\big](t) - f(y(t))\,dt. \quad (26)$$

The last term in this summand can be estimated by appealing to Lemma 2.1 and observing

$$|I_n| := \left|\int_{t_{n-1}}^{t_n} I\big[f\big(\boldsymbol{\eta}^{[p]}\big)\big](t) - f(y(t))\,dt\right| \le h\big\|\boldsymbol{e}^{[p]}\big\| W_n L + \frac{F}{M!}h^{M+1}. \quad (27)$$

We estimate the other terms by making use of their respective Lipshitz constants:

$$\begin{aligned}\big|e_n^{[p+1]}\big| &\le \big|e_{n-1}^{[p+1]}\big| + h_n\big|f_I\big(\eta_n^{[p+1]}\big) - f_I\big(\eta_n^{[p]}\big)\big| + h_n\big|f_E\big(\eta_{n-1}^{[p+1]}\big) - f_E\big(\eta_{n-1}^{[p]}\big)\big| + |I_n| \\ &\le \big|e_{n-1}^{[p+1]}\big| + hL_I\big(\big|e_n^{[p+1]}\big| + \big|e_n^{[p]}\big|\big) + hL_E\big(\big|e_{n-1}^{[p+1]}\big| + \big|e_{n-1}^{[p]}\big|\big) + |I_n|. \quad (28)\end{aligned}$$

The second line follows from the first by adding and subtracting $f_I(y_n)$ and $f_E(y_{n-1})$ to the inside of each of the absolute values containing $\big|f\big(\eta_n^{[p+1]}\big) - f\big(\eta_n^{[p]}\big)\big|$ and $\big|f\big(\eta_{n-1}^{[p+1]}\big) - f\big(\eta_{n-1}^{[p]}\big)\big|$, respectively. Note that we also make use of the fact that $h_n \le h$, although this too can be relaxed.

We continue by subtracting $hL_I\big|e_n^{[p+1]}\big|$ from both sides, dividing by $1 - hL_I > 0$, recognizing that $h_n < h$, and collecting the remaining terms involving the "explicit"

portions:

$$\begin{aligned}\left|e_n^{[p+1]}\right| &\le \frac{1}{1-hL_I}\left[(1+hL_E)\left|e_{n-1}^{[p+1]}\right| + hL_I\left|e_n^{[p]}\right| + hL_E\left|e_{n-1}^{[p]}\right| + |I_n|\right] \\ &\le \frac{1}{1-hL_I}\left[(1+hL_E)\left|e_{n-1}^{[p+1]}\right| + hL\left\|\boldsymbol{e}^{[p]}\right\| + |I_n|\right] \\ &\le \frac{1}{1-hL_I}\left[(1+hL_E)\left|e_{n-1}^{[p+1]}\right| + hL(1+W_n)\left\|\boldsymbol{e}^{[p]}\right\| + \frac{F}{M!}h^{M+1}\right] \\ &\le \frac{1+hL_E}{1-hL_I}\left|e_{n-1}^{[p+1]}\right| + \frac{1}{1-hL_I}\left[hW\left\|e^{[p]}\right\| + \frac{F}{M!}h^{M+1}\right], \end{aligned} \tag{29}$$

where we define $W := \max_{1\le n\le N}(1+W_nL)$.

We make use of two separate estimates for $1/(1-hL_I)$ to estimate the two terms found in the right-hand side of (29). For the first term, we expand the geometric series and keep the first two terms:

$$\frac{1}{1-hL_I} = 1 + (hL_I) + (hL_I)^2 + \cdots = 1 + (hL_I) + (hL_I)^2\frac{1}{1-hL_I}. \tag{30}$$

This is valid because $hL_I < 1$. Additionally, $hL_I < \frac{1}{2}$, and therefore,

$$hL_I < 1 - hL_I \quad\Longrightarrow\quad \frac{(hL)^2}{1-hL_I} < hL_I. \tag{31}$$

Together, these estimates imply that the first term can be estimated with

$$\frac{1}{1-hL_I} \le 1 + 2hL_I \le e^{2hL_I}. \tag{32}$$

For the second term, we have $1/(1-hL_I) \le 2$ for all $hL_I \in \left[0, \frac{1}{2}\right]$. This leads us to observe that

$$\left|e_n^{[p+1]}\right| \le e^{2hL_I}(1+hL_E)\left|e_{n-1}^{[p+1]}\right| + 2\left(hW\left\|\boldsymbol{e}^{[p]}\right\| + \frac{F}{M!}h^{M+1}\right). \tag{33}$$

Next, we appeal to Corollary 2.3 and make use of $A = e^{2hL_I}(1+hL_E) > 1$ and $B = 2\left(hW\left\|\boldsymbol{e}^{[p]}\right\| + (F/M!)h^{M+1}\right)$ to conclude that

$$\begin{aligned}\left|e_n^{[p+1]}\right| \le{}& e^{2hnL_I}(1+hL_E)^n|e_0| \\ &+ ne^{2h(n-1)L_I}(1+hL_E)^{n-1}2\left(hW\left\|\boldsymbol{e}^{[p]}\right\| + \frac{F}{M!}h^{M+1}\right). \end{aligned} \tag{34}$$

Since $1 + hL_E \le e^{hL_E}$ and $n \le N$, we have the desired result. □

3C. ***Corollaries of main result: implicit and explicit error estimates.*** With the general case proven in Theorem 3.1, we find results for both implicit as well as explicit SDC solvers. An immediate corollary to Theorem 3.1 can be found by setting $f_E \equiv 0$, in which case L_I becomes the Lipshitz constant for f, and the SISDC solver reduces to classical SDC with backward Euler defined in (10).

Corollary 3.2. *The errors for a single step of the implicit SDC method defined in* (10) *satisfy*

$$\left|e_n^{[p+1]}\right| \leq e^{2NhL}|e_0| + C_1 h \left\|\boldsymbol{e}^{[p]}\right\| + C_2 h^{M+1} \tag{35}$$

provided $h < 1/(2L)$. *The constants* C_1 *and* C_2 *depend only on the smoothness of* f, *the exact solution* y, *and the choice of quadrature points.*

It is worth noting that the error estimate provided here is an *asymptotic* error estimate. That is, one key assumption that we have to make is that $h < 1/(2L)$, which we do not have to make for the explicit case. Unfortunately, one key benefit of implicit solvers is that large time steps can be taken, in which case it is certainly possible that the solver does not obey this assumption. For these cases, a rigorous error estimate and analysis when $h > 1/(2L)$ would make for an interesting result, which would be especially important for multiscale problems that contain large time scale separations. This observation is beyond the scope of the present work.

A related corollary for explicit solvers with tighter error bounds can be found. The result is the following:

Corollary 3.3. *The errors for a single step of the explicit SDC method defined in* (9) *satisfy*

$$\left|e_n^{[p+1]}\right| \leq e^{NhL}|e_0| + C_1 h \left\|\boldsymbol{e}^{[p]}\right\| + C_2 h^{M+1}, \tag{36}$$

where N *is the number of intervals under consideration,* L *is the Lipschitz constant of* f *for the ODE* $y' = f(y)$, *and* C_1 *and* C_2 *are constants that depend only on* f, *the exact solution* y, *and the selection of quadrature points.*

Proof. Revisit the proof of Theorem 3.1, and replace the error estimate for $1/(1 - hL_I) \leq 2$ with 1 instead of 2. □

4. Convergence proofs for higher-order base solvers

We now consider the spectral deferred correction method with the implicit trapezoidal rule as its base solver:

$$\eta_n^{[p+1]} = \eta_{n-1}^{[p+1]} + \frac{h_n}{2}\left[f\left(\eta_n^{[p+1]}\right) + f\left(\eta_{n-1}^{[p+1]}\right) - f\left(\eta_n^{[p]}\right) - f\left(\eta_{n-1}^{[p]}\right)\right] + h\sum_{m=1}^{M} w_{n,m} f\left(\eta_m^{[p]}\right). \tag{37}$$

What makes this method interesting is that it picks up a total of two orders of accuracy with each correction. Note again that, in the absence of the terms that the factor $h_n/2$ multiplies, this method reduces to explicit Picard iteration, which picks up a single additional order of accuracy with each correction.

The result we focus on is the impact of each correction step, in which case any provisional solution may be used. In order to retain large regions of absolute stability reasonable methods include low-order implicit solvers such as backward Euler, or the second-order implicit trapezoidal (Crank–Nicholson) rule

$$\text{(trapezoidal rule)} \quad \eta_n^{[0]} = \eta_{n-1}^{[0]} + \frac{h_n}{2}\left(f\left(\eta_{n-1}^{[0]}\right) + f\left(\eta_n^{[0]}\right)\right), \quad n = 1, 2, \ldots, N. \tag{38}$$

In this section, we examine the interplay between the integral over the entire time interval, and the addition of extra integral terms that allows this solver to pick up additional orders of accuracy.

Let us define the exact value of the right-hand-side function as $f_n = f(y(t_n))$, the approximate value of the right-hand-side function as $f_n^{[p]} = f\left(\eta_n^{[p]}\right)$, and the local and global quadrature rules for integration over the subinterval $[t_{n-1}, t_n]$ as

$$T_n = \frac{h_n}{2}[f_{n-1} + f_n], \qquad T_n^{[p]} = \frac{h_n}{2}\left[f_{n-1}^{[p]} + f_n^{[p]}\right],$$
$$H_n = h\sum_{m=1}^{M} \omega_{n,m} f_m, \qquad H_n^{[p]} = h\sum_{m=1}^{M} \omega_{n,m} f_m^{[p]}.$$

These definitions allow us to compactly write the SDC method with the trapezoidal rule defined in (37) to read

$$\eta_n^{[p+1]} = \eta_{n-1}^{[p+1]} + \left(T_n^{[p+1]} - T_n^{[p]}\right) + H_n^{[p]}. \tag{39}$$

Recall that the exact solution satisfies the integral (7), which we repeat:

$$y_n = y_{n-1} + \int_{t_{n-1}}^{t_n} f(y(t))\, dt.$$

Theorem 4.1. *When coupled with the implicit trapezoidal rule, the errors for a single step of the spectral deferred correction method satisfy*

$$\left|e_n^{[p+1]}\right| \le e^{2NhL}|e_0| + 2Ne^{2(N-1)hL}\left(\frac{1}{12}\left\|\frac{d^2E^{[p]}}{dt^2}(\cdot)\right\|h^3 + \frac{F}{M!}h^{M+1}\right), \tag{40}$$

provided $hL < 1$, where N is the number of intervals under consideration, M is the number of points involved, L is the Lipschitz constant of f, F is an upper bound for the M-th derivative of f, and the function $E^{[p]}(t)$ is the polynomial interpolant for the error in the approximation of the right-hand-side function during the p-th

iterate defined by

$$E^{[p]}(t) := I\big[f\big(\boldsymbol{\eta}^{[p]}\big)\big](t) - I[f(\mathbf{y})](t) = \sum_{m=1}^{M} \Delta f_m^{[p]} \ell_m(t/h), \tag{41}$$

where $\Delta f_m^{[p]} := f_m^{[p]} - f_m$ *for each* $m = 1, 2, \ldots, M$. *The norm defined in* (40) *is the maximum absolute value of the second derivative of* $E^{[p]}$:

$$\left\| \frac{d^2 E^{[p]}}{dt^2}(\cdot) \right\| := \max_{t\in[0,h]} \left| \big(E^{[p]}\big)''(t) \right|. \tag{42}$$

Proof. We subtract the exact solution defined in (7) from the SDC method based upon the trapezoidal rule defined in (39) to end up with

$$\begin{aligned} e_n^{[p+1]} &= e_{n-1}^{[p+1]} + \big(T_n^{[p+1]} - T_n^{[p]}\big) + H_n^{[p]} - \int_{t_{n-1}}^{t_n} f(y(t))\,dt \\ &= e_{n-1}^{[p+1]} + T_n^{[p+1]} - T_n + T_n - T_n^{[p]} + H_n^{[p]} - H_n + H_n - \int_{t_{n-1}}^{t_n} f(y(t))\,dt \\ &= e_{n-1}^{[p+1]} + \underbrace{\big(T_n^{[p+1]} - T_n\big)}_{\text{I}} + \underbrace{\big(H_n^{[p]} - T_n^{[p]} + T_n - H_n\big)}_{\text{II}} + \underbrace{I_n}_{\text{III}}, \end{aligned} \tag{43}$$

where $I_n := H_n - \int_{t_{n-1}}^{t_n} f(y(t))\,dt$ is the difference between the high-order (discrete) integral and the exact integral of the right-hand side.

We now estimate each of the three terms to the right of $e_{n-1}^{[p+1]}$ in (43) separately, starting with the first term:

$$|\text{I}| = \frac{h_n}{2}\big|f_{n-1}^{[p+1]} + f_n^{[p+1]} - f_{n-1} - f_n\big| \le \frac{Lh_n}{2}\big(\big|e_n^{[p+1]}\big| + \big|e_{n-1}^{[p+1]}\big|\big), \tag{44}$$

which follows from the Lipshitz continuity of f. The third term can be estimated by first recognizing that

$$H_n := h\sum_{m=1}^{M} \omega_{n,m} f_m = \int_{t_{n-1}}^{t_n} I[f(\mathbf{y})](t)\,dt,$$

and then using (18) (which requires assuming that $f \circ y \in C^M$) in order to yield

$$|\text{III}| = \left| H_n - \int_{t_{n-1}}^{t_n} f(y(t))\,dt \right| = |H_n - I_n| \le \frac{F}{M!} h^{M+1}, \tag{45}$$

where F is any number that satisfies $\left\| \frac{d^M}{dt^M}(f \circ y) \right\|_\infty \le F$.

Finally, we address the second, and most interesting, term on the right-hand side of (43). The key observation comes from recognizing this term as the difference between a low-order (local) quadrature, T_n, and a high-order (global) quadrature H_n.

Note that the exact integral of the polynomial interpolant $E^{[p]}$ over the subinterval $[t_{n-1}, t_n]$ is

$$\int_{t_{n-1}}^{t_n} E^{[p]}(t)\,dt = H_n^{[p]} - H_n, \tag{46}$$

and that

$$T_n^{[p]} - T_n = \frac{h_n}{2}\big(E^{[p]}(t_n) + E^{[p]}(t_{n-1})\big) \tag{47}$$

is a low-order approximation to this integral. By textbook results, we have

$$\mathrm{II} = \big(H_n^{[p]} - H_n - T_n^{[p]} + T_n\big) = -\frac{h_n^3}{12}\frac{d^2E^{[p]}}{dt^2}(\xi_n), \tag{48}$$

where ξ_n is some number between t_{n-1} and t_n. Together, this implies

$$|\mathrm{II}| \le \frac{h_n^3}{12}\left\|\frac{d^2E^{[p]}}{dt^2}(\,\cdot\,)\right\|. \tag{49}$$

All together, inserting (44), (49), and (45) into (43), we have

$$\begin{aligned}\big|e_n^{[p+1]}\big| &\le \big|e_{n-1}^{[p+1]}\big| + |\mathrm{I}| + |\mathrm{II}| + |\mathrm{III}| \\ &\le \left(1 + \frac{Lh_n}{2}\right)e_{n-1}^{[p+1]} + \frac{Lh_n}{2}e_n^{[p+1]} + \frac{h_n^3}{12}\left\|\frac{d^2E^{[p]}}{dt^2}(\,\cdot\,)\right\| + \frac{F}{M!}h^{M+1}.\end{aligned} \tag{50}$$

After replacing each $h_n \le h$, rearranging, and assuming that $hL/2 < 1$, we have

$$\big|e_n^{[p+1]}\big| \le \underbrace{\frac{1 + hL/2}{1 - hL/2}}_{\le e^{2hL}} e_{n-1}^{[p+1]} + \underbrace{\frac{1}{1 - hL/2}}_{\le 2}\left(\frac{h^3}{12}\left\|\frac{d^2E^{[p]}}{dt^2}(\cdot)\right\| + \frac{F}{M!}h^{M+1}\right). \tag{51}$$

We now verify the two underscored inequalities involving the $1 \pm hL/2$ terms in (51). Identical to (32), we have

$$\frac{1}{1 - hL/2} \le 1 + 2(hL/2) = 1 + hL,$$

after expanding the rational expression in terms of a geometric series, and assuming that $hL/2 < 1$ in order to retain convergence. Because $1 + hL/2 \le 1 + hL$, we have

$$\frac{1 + hL/2}{1 - hL/2} \le (1 + hL)^2 \le e^{2hL},$$

which verifies the first of the two underscored inequalities. For the second one, we need only assume that $hL < 1$, which yields $1/(1 - hL/2) < 2$.

All together, we have

$$\big|e_n^{[p+1]}\big| \le e^{2hL}e_{n-1}^{[p+1]} + 2\left(\frac{h^3}{12}\left\|\frac{d^2E^{[p]}}{dt^2}(\,\cdot\,)\right\| + \frac{F}{M!}h^{M+1}\right), \tag{52}$$

which yields the desired result after appealing to Corollary 2.3 and using $n \le N$. □

Remark. The same decomposition for the error can be used for SDC coupled with forward (or backward) Euler. That is, identical to the decomposition found in (43), we can decompose the error as

$$\begin{aligned} e_n^{[p+1]} &= e_{n-1}^{[p+1]} + L_n^{[p+1]} - L_n^{[p]} + H_n^{[p]} - \int_{t_{n-1}}^{t_n} f(y(t))\,dt \\ &= e_{n-1}^{[p+1]} + L_n^{[p+1]} - L_n + L_n - L_n^{[p]} + H_n^{[p]} - H_n + H_n - \int_{t_{n-1}}^{t_n} f(y(t))\,dt \\ &= e_{n-1}^{[p+1]} + \underbrace{\left(L_n^{[p+1]} - L_n\right)}_{\text{I}} + \underbrace{\left(H_n^{[p]} - L_n^{[p]} + L_n - H_n\right)}_{\text{II}} + \underbrace{I_n}_{\text{III}}, \end{aligned} \tag{53}$$

where L_n and $L_n^{[p]}$ denote a "low-order" integral of the right-hand side, but this time we use

$$L_n := h_n f_{n-1}, \qquad L_n^{[p]} := h_n f_{n-1}^{[p]}, \tag{54}$$

for forward Euler, or instead

$$L_n := h_n f_n, \qquad L_n^{[p]} := h_n f_n^{[p]} \tag{55}$$

for backward Euler. The third term III is again $\mathbb{O}(h^{M+1})$, and the first term I can be bounded by a constant times $\left|e_n^{[p]}\right| + \left|e_{n-1}^{[p+1]}\right|$. The lack of additional order pickup can be found by observing that the second source of error instead satisfies

$$\text{II} = \left(H_n^{[p]} - H_n - L_n^{[p]} + L_n\right) = -\frac{h_n^2}{2}\frac{dE^{[p]}}{dt}(\xi_n), \tag{56}$$

where ξ_n is some number between t_{n-1} and t_n. In the following examples, we compare this term to that found from the trapezoidal rule.

4A. ***Examples.*** To illustrate the results of the theorem presented in this section, we consider the linear test case

$$y'(t) = y(t), \quad t > 0, \;\; y(0) = 1, \tag{57}$$

and examine the errors produced by the SDC method when coupled with a higher-order base solver. The order pickup for the SDC method can be found by examining the size of the second source of error, defined in (48) and (56), given by $h_n^3/12\left(E^{[p]}\right)''(\xi_n)$ for the trapezoidal rule and $h_n^2/2\left(E^{[p]}\right)'(\xi_n)$ for the forward (or backward) Euler method. Note that the form and size of this error is identical for either the forward or backward Euler base solver.

In the following examples, we work out the size of this term for a few different case studies. Taylor expansions are found by making use of the Maple software package.

	trapezoidal rule		forward Euler	
interval	error	$p=0$	error	$p=0$
$[0, \frac{h}{2}]$	$\frac{h}{24}(e_1^{[p]} - 2e_2^{[p]} + e_3^{[p]})$	$\mathbb{O}(h^4)$	$\frac{h}{24}(7e_1^{[p]} - 8e_2^{[p]} + e_3^{[p]})$	$\mathbb{O}(h^3)$
$[\frac{h}{2}, h]$	$\frac{h}{24}(e_1^{[p]} - 2e_2^{[p]} + e_3^{[p]})$	$\mathbb{O}(h^4)$	$\frac{h}{24}(e_1^{[p]} + 4e_2^{[p]} - 5e_3^{[p]})$	$\mathbb{O}(h^3)$

Table 2. Correction errors with $M = 3$ uniformly spaced points.

	trapezoidal rule		forward Euler	
interval	error	$p=0$	error	$p=0$
$[0, \frac{h}{3}]$	$\frac{h}{72}(3e_1^{[p]} - 7e_2^{[p]} + 5e_3^{[p]} - e_4^{[p]})$	$\mathbb{O}(h^4)$	$\frac{h}{72}(15e_1^{[p]} - 19e_2^{[p]} + 5e_3^{[p]} - e_4^{[p]})$	$\mathbb{O}(h^3)$
$[\frac{h}{3}, \frac{2h}{3}]$	$\frac{h}{72}(e_1^{[p]} - e_2^{[p]} - e_3^{[p]} + e_4^{[p]})$	$\mathbb{O}(h^4)$	$\frac{h}{72}(e_1^{[p]} + 11e_2^{[p]} - 13e_3^{[p]} + e_4^{[p]})$	$\mathbb{O}(h^3)$
$[\frac{2h}{3}, h]$	$\frac{h}{72}(e_1^{[p]} - 5e_2^{[p]} + 7e_3^{[p]} - 3e_4^{[p]})$	$\mathbb{O}(h^4)$	$\frac{h}{72}(e_1^{[p]} - 5e_2^{[p]} - 5e_3^{[p]} + 9e_4^{[p]})$	$\mathbb{O}(h^3)$

Table 3. Correction errors with $M = 4$ uniformly spaced points.

$M = 3$ *uniformly spaced points.* We first consider the results of using a total of $M = 3$ equispaced quadrature points and look at the local error over the subinterval $[t_{n-1}, t_n]$ produced by the second term II defined in (48) and (56). In the second set of columns in Table 2 look at the size of this term by writing out Taylor expansions for the first $p = 0$ SDC iteration of a solver constructed by taking a forward Euler provisional solution for the first time step. Note that in this case, each point satisfies $e_n^{[0]} = \mathbb{O}(h^2)$ for each n, because the local truncation error (LTE) for Euler's method is second-order accurate. Therefore, the jump from $\mathbb{O}(h^2)$ to $\mathbb{O}(h^4)$ indicates that the trapezoidal correction picks up an additional two orders of accuracy, whereas the jump from $\mathbb{O}(h^2)$ to $\mathbb{O}(h^3)$ picks up a single additional order of accuracy for the Euler base solver, which is consistent with the theory.

$M = 4$ *uniformly spaced points.* We next consider the same problem with a total of $M = 4$ equispaced quadrature points. Again, we look at the errors for the trapezoidal method compared to the forward Euler method after first constructing a provisional solution with the forward Euler method. We again observe that the trapezoidal rule improves the order of accuracy of the provisional solution by two factors, whereas forward Euler (or likewise backward Euler) only improves the order by one. Results for these quadrature points are presented in Table 3.

$M = 4$ *nonequispaced spaced points.* Finally, we consider a case with a total of $M = 4$ nonequispaced points. As an illustrative example, we consider the quadrature points $\xi_1 = 0$, $\xi_2 = \frac{1}{3}$, $\xi_3 = \frac{1}{2}$, and $\xi_4 = 1$. We find that the trapezoidal error only increases the order of the solver by one degree, which is consistent with the findings in [6], where the authors show that when the second-order Runge–Kutta

	trapezoidal rule	
interval	error	$p = 0$
$\left[0, \frac{h}{3}\right]$	$\frac{h}{162}\left(8e_1^{[p]} - 27e_2^{[p]} + 20e_3^{[p]} - e_4^{[p]}\right)$	$\mathbb{O}(h^3)$
$\left[\frac{h}{3}, \frac{h}{2}\right]$	$\frac{h}{5184}\left(14e_1^{[p]} - 27e_2^{[p]} + 8e_3^{[p]} + 5e_4^{[p]}\right)$	$\mathbb{O}(h^3)$
$\left[\frac{h}{2}, h\right]$	$\frac{h}{192}\left(10e_1^{[p]} - 81e_2^{[p]} + 88e_3^{[p]} - 17e_4^{[p]}\right)$	$\mathbb{O}(h^3)$
	forward Euler	
interval	error	$p = 0$
$\left[0, \frac{h}{3}\right]$	$\frac{h}{162}\left(35e_1^{[p]} - 54e_2^{[p]} + 20e_3^{[p]} - e_4^{[p]}\right)$	$\mathbb{O}(h^3)$
$\left[\frac{h}{3}, \frac{h}{2}\right]$	$\frac{h}{5184}\left(14e_1^{[p]} + 405e_2^{[p]} - 424e_3^{[p]} + 5e_4^{[p]}\right)$	$\mathbb{O}(h^3)$
$\left[\frac{h}{2}, h\right]$	$\frac{h}{192}\left(10e_1^{[p]} - 81e_2^{[p]} + 40e_3^{[p]} + 31e_4^{[p]}\right)$	$\mathbb{O}(h^3)$

Table 4. Correction errors with $M = 4$ nonequispaced points. In this case, we find that both methods only pick up a single additional order of accuracy.

method is used as a corrector, then the solver does not always pick up two orders of accuracy with each correction loop. Results for this problem are presented in Table 4.

5. Numerical results

The primary contribution of this work is to construct rigorous error estimates for classical SDC methods, and therefore, we only include a couple of numerical results. An abundance of SDC examples applied to ordinary and partial differential equations can be found in the literature. One of our key goals here is to promulgate the fact that the primary source of high-order accuracy inherent in all SDC methods comes from its underlying Picard integral formulation, and not necessarily the "base solver", and therefore, we focus our results on nearby variations of classical SDC methods and demonstrate how classical SDC methods can be extended to produce related high-order solvers.

First we introduce a comparison of errors (and stability regions) for explicit SDC versus Picard iteration, second we explore modifications of the constant in front of an implicit SDC method, and finally we compare semi-implicit SDC and modified semi-implicit SDC solvers. For the sake of brevity the proposed modifications to SDC methods are not formally analyzed but straightforward extensions of the theorems presented in this work can be constructed to present formal error bounds for these methods. The numerical evidence presented here supports this claim.

5A. ***A comparison of explicit SDC and Picard iteration.*** In this numerical example, we compare the errors and stability regions by applying the Picard iterative

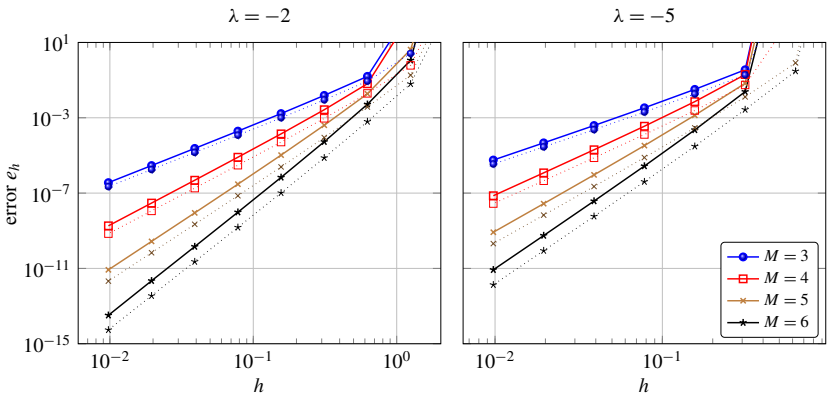


Figure 1. Linear test case. Here, we compare explicit SDC (solid lines) with Picard iteration (dashed lines) of various orders. Each method attains the desired order of accuracy with the minimum number of corrections. In each case, Picard iteration has slightly smaller errors when compared to the equivalent explicit SDC method of the same order and same quadrature rule.

method defined in (8) to that of the explicit SDC defined in (9). In order to present an equal comparison of these two solvers, we consider identical initial guesses, or provisional solutions $\boldsymbol{\eta}^{[0]}$ based upon forward Euler time stepping and we work with uniform quadrature points for all of our test cases.

Errors for a linear test case. In Figure 1, we compare errors for the linear equation

$$y' = \lambda y, \quad y(0) = 1, \tag{58}$$

at a final time of $T = 10$ with $\lambda = -2$ and $\lambda = -5$. Other orders and values of λ show similar results where we observe slightly smaller error constants when using the Picard iterative method compared to the equivalent SDC method. This is consistent with the findings of Corollary 3.3, because the first error estimate

$$\left|e_n^{[p+1]}\right| \leq \left|e_{n-1}^{[p+1]}\right| + h_n \left| f\left(\eta_{n-1}^{[p+1]}\right) - f\left(\eta_{n-1}^{[p]}\right)\right| + |I_n|$$

could be tightened up to read

$$\left|e_n^{[p+1]}\right| \leq \left|e_{n-1}^{[p+1]}\right| + |I_n|,$$

which produces a smaller (provable) overall error for the Picard method when compared to the SDC method.

A comparison of regions of absolute stability for explicit methods. Next, we seek to compare regions of absolute stability for explicit SDC methods and their Picard iterative cousins. Here we observe that the stability regions are slightly improved when the "Euler term" in the time stepping is dropped from the SDC method. That

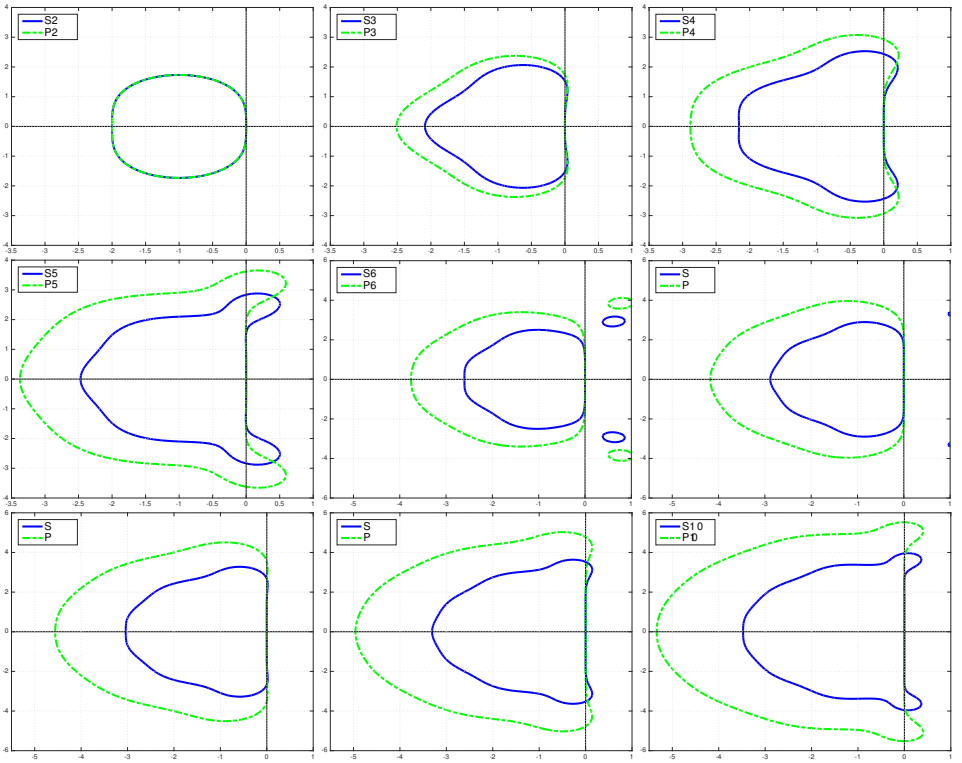


Figure 2. Stability regions for explicit methods. Here, we compare SDC methods to that of Picard iterative methods where the explicit "Euler term" is dropped from the iterative process. In each case save one, we observe that the Picard iterative methods have slightly larger regions of absolute stability. The second-order SDC and Picard methods that use two quadrature points are identical because the forward Euler time step vanishes in the SDC method. The scaling on the axes for the methods of orders 6 through 10 is different than the scaling for the methods of orders two through five.

is to say, we find that the Picard iterative methods generally have larger regions of absolute stability when compared to their SDC counterparts.

In order to demonstrate this, in Figure 2, we include a comparison of plots of the regions of absolute stability, defined by

$$\mathbb{D} := \{z \in \mathbb{C} : |\rho(z)| < 1\} \tag{59}$$

where $\rho(z)$ is the amplification factor for various quadrature rules for both of these methods, $z := \lambda h$, and λ is defined as in (58). There, we compare methods of orders two through ten, all based on equispaced quadrature points, forward Euler time stepping for the provisional solution, and the minimum number of corrections required to reach the desired order of accuracy. (For example, the third-order method uses two corrections and the fifth-order method uses four corrections.) Similar to

most explicit Runge–Kutta methods, we find that the regions of absolute stability increase as the order is increased, but there are also more function evaluations per time step.

5B. ***Implicit SDC methods with modified backward Euler time steps.*** Here, we consider implicit SDC methods with a variable constant in front of the vanishing term:

$$\eta_n^{[p+1]} = \eta_{n-1}^{[p+1]} + \theta h\big[f\big(\eta_n^{[p+1]}\big) - f\big(\eta_n^{[p]}\big)\big] + h\sum_{m=1}^{M} w_{n,m} f\big(\eta_m^{[p]}\big), \qquad n = 1, 2, \ldots, N. \quad (60)$$

This same scaling has already been explored in [39] for SDC methods, but there the authors only consider the case where $\frac{1}{2} \leq \theta \leq 1$. With $\theta = 0$, we have (explicit) Picard iteration (provided the provisional solution is modified), with $\theta = 1$, we have the classical implicit SDC method, and with negative values of θ we have backward Euler solves on negative time steps; none of the these changes affect the overall order of accuracy, only the size of the error constant and the regions of absolute stability. Following the proof of the main theorem in this work, we find the following result under a modified estimate for the time step size.

Theorem 5.1. *The errors for a single step of the modified implicit SDC method defined in* (60) *satisfy*

$$\big|e_n^{[p+1]}\big| \leq e^{2N|\theta|hL}|e_0| + C_1 h\big\|\boldsymbol{e}^{[p]}\big\| + C_2 h^{M+1} \quad (61)$$

provided $|\theta|h < 1/(2L)$. *The constants*

$$C_1 = 2Ne^{2N|\theta|hL}\Big(|\theta| + L\max_n W_n\Big) \quad \textit{and} \quad C_2 = 2Ne^{2N|\theta|hL}\frac{F}{M!}$$

again depend only on the smoothness of f, *the exact solution* y, *and the choice of quadrature points*, *but this time they also depend on* θ.

Note that the value of $\theta = 0$ minimizes the size of these constants, but it also produces poor regions of absolute stability. With $\theta \gg 1$ we have a method that is heavy handed on multiple backward Euler solves, and therefore, it has a very large region of absolute stability; however, these methods unfortunately introduce larger error constants. Small values of θ decrease these error constants, but they modify the regions of absolute stability to the point where they become finite and therefore undesirable as these are implicit methods.

A verification of high-order accuracy: the nonlinear pendulum problem. As a verification of the high-order accuracy of the solvers, we consider the equations of motion for a nonlinear pendulum:

$$x''(t) + \sin x(t) = 0 \quad (62)$$

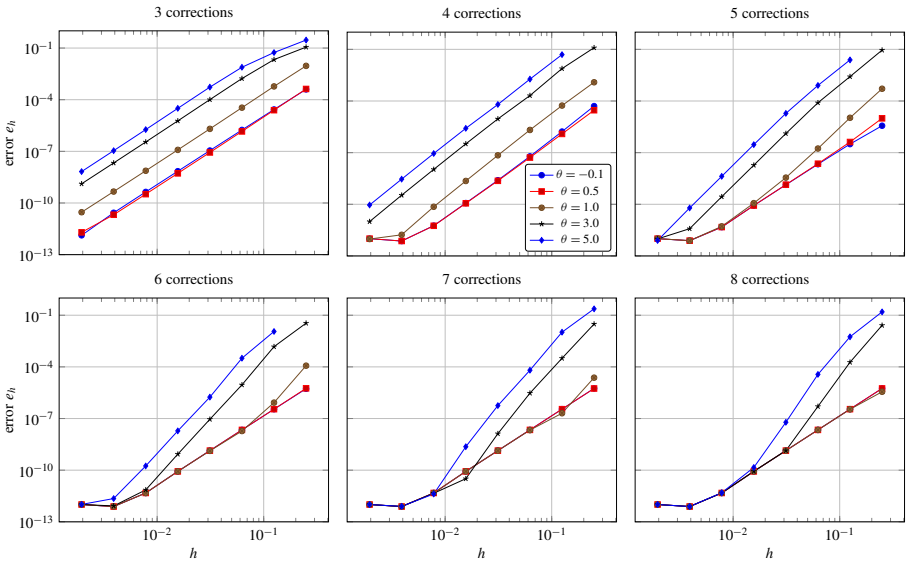


Figure 3. Nonlinear pendulum problem. Here, we compare SDC methods with different scalings on the backward Euler term as defined in (60). The case with $\theta = 1$ is classical implicit SDC. We observe the expected result that all methods have high-order accuracy and that $\theta > 1$ produces larger error constants. The case with $\theta = -0.1$ is not a useful method because it has a finite region of absolute stability. While all methods here are fourth-order accurate after three corrections, the methods with large values of θ stand to gain the most through additional corrections. This can be attributed to the large error constants found in the backward Euler term that vanish at the number of iterations increase (provided the iterates converge).

with appropriate initial conditions. If we perform the change of variables $y_1(t) = x(t)$ and $y_2(t) = x'(t)$, we end up with the following first-order nonlinear system of equations that is equivalent to (62):

$$(y_1, y_2)' = (y_2, -\sin y_1). \tag{63}$$

We consider initial conditions defined by $(y_1(0), y_2(0)) = (0, 1)$, and we integrate this problem to a final time of $T = 10$. To compute a reference solution, we use MATLAB's built-in ode45 with a relative tolerance of 10^{-12} and an absolute tolerance of 10^{-14}.

We present convergence results for this problem in Figure 3. These results indicate that each method is indeed high-order independent of the value of θ. For the sake of brevity, we only report results for methods with a total of $M = 4$ equispaced quadrature points, but we also compare results for different number of corrections. (This underlying quadrature rule is also known as Simpson's $\frac{3}{8}$ rule, which has a smaller error constant than Simpson's rule that uses $M = 3$ equispaced points, but it comes at the cost of an additional function evaluation.) We not only

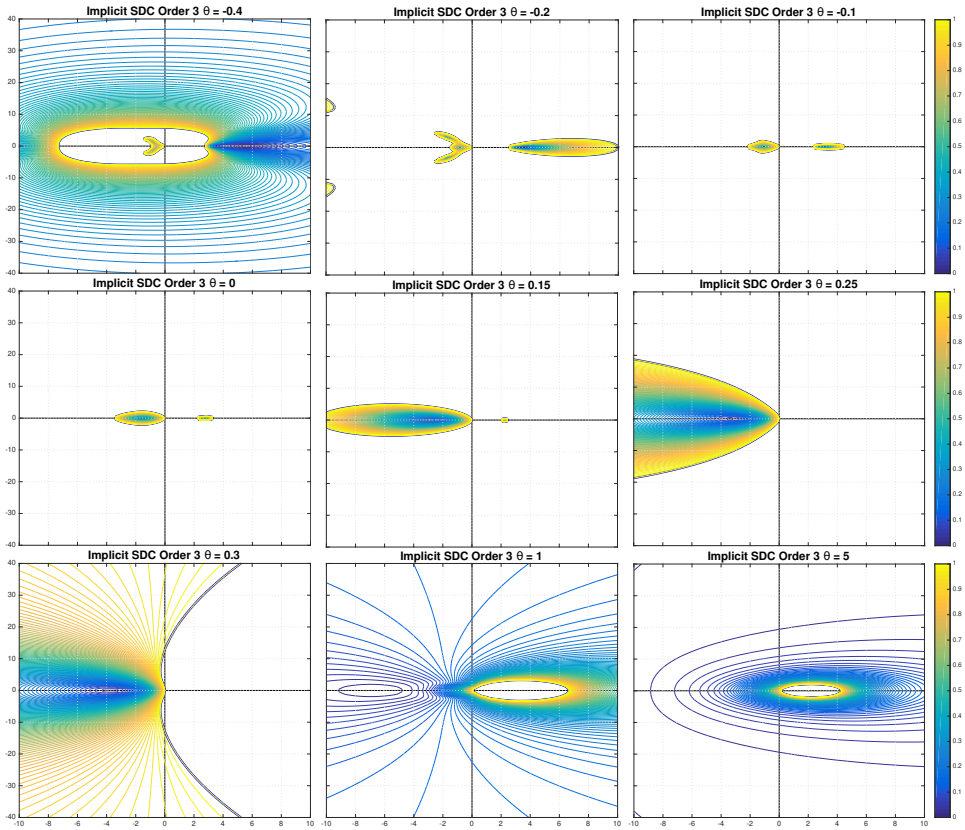


Figure 4. Stability regions for modified implicit methods. Here, we perform a parameter study to the modified implicit SDC method introduced in (60); the case with $\theta = 0$ is explicit for each of the correction steps because there is no backward Euler time step to be solved for; however, the stability region is different than that of the same-order Picard method because the initial guess (provisional solution) is computed using backward Euler time steps, whereas the Picard method makes use of forward Euler steps for its initial guess. Somewhere in the interval $\theta \in [0, 1]$ there is a transition between finite and infinite regions of absolute stability. Infinite and large regions are typically desirable when performing implicit solves. Large values of θ increase the regions of absolute stability but at the cost of stiff inversions and larger error constants.

find that smaller values of θ produce smaller errors, which the theory supports, but we also demonstrate that more corrections for the methods with large θ values can help to decrease the errors.

A parameter study of regions of absolute stability for implicit methods. Given the results of the previous section, it would be tempting to set $\theta = 0$ in order to reduce the total error. What is missing from this observation is an understanding of the regions of absolute stability. We now address this question. We find that small values of θ produce finite regions of absolute stability, and that large values of θ

increase the regions of absolute stability (when compared to classical SDC methods) but they also increase the stiffness of each implicit solve. With that being said, larger time steps should be able to be taken, but as is pointed out in the previous section, larger errors are introduced. This reproduces the usual tradeoff between being able to take large time steps with large errors or being forced to take smaller time steps but at an increased computational cost.

In Figure 4 we present results for a third-order method with various values of θ. There we plot contour plots of the modulus of the amplification factor $|\rho(z)|$ in place of the boundary defined by $|\rho(z)| = 1$ because if we were to plot the boundary, then it would not be clear what parts are stable. In this sequence of images, we present results for various values of $\theta \in [-0.4, 5]$. Larger values of θ such as $\theta = 100$ look very similar to $\theta = 5$. Tests on methods of other orders produce similar results involving transitions between finite and infinite regions of absolute stability as θ increases from 0. The tradeoff between the size and shape of the stability regions for various quadrature rules is left for future work.

Before continuing, we point out that diagonally implicit Runge–Kutta methods (on nonequispaced points) can be constructed from this very same framework. The point here is that because the term involving the difference $f\left(\eta_n^{[p+1]}\right) - f\left(\eta_n^{[p]}\right)$ in (60) does not contribute to the overall order of accuracy, the scaling in front of this term can be modified so that each implicit solve uses the exact same time step, which would result in a singly diagonally implicit Runge–Kutta (SDIRK) method. This could be advantageous for easing the implementation of SDC methods in large-scale code bases. In such a case the provisional solution would have to be modified in order to retain constant time steps for each stage in the solver. This would mean an extra correction or a (low-order) polynomial interpolation step would be necessary to not lose the starting accuracy found in the provisional solution, which would again modify the regions of absolute stability for solvers of various orders.

Similar modifications have recently been explored on nonequispaced points from a linear algebra perspective. In [38], the author makes use of this vantage and optimizes their solvers by modifying the coefficients in the fixed-point iteration matrices. It is pointed out that there are a number of items that could be optimized, such as the spectral radius of the solver (in order to optimize the convergence rate of the sweeps), the matrix norm (for the purpose of reducing the error under the assumption of a small number of sweeps), the error at the final time, the average reduction factor in each sweep block (for the purpose of adaptively choosing the number of sweeps, which could include flexible or greedy sweeps), and so on. Even though SDC methods are a subset of Runge–Kutta methods, and all of these options can be found by looking at this more general class of methods, one key advantage SDC methods enjoy is they do not typically sacrifice the difficult order conditions

that a more generic RK method would have to address. At the same time, SDC methods still have ample levers to tune for optimization purposes.

5C. ***Modified semi-implicit SDC methods.*** Recall the semi-implicit SDC (SISDC) method begins with a partition of the right-hand side into two functions f_I and f_E via

$$y' = f(y), \quad f(y) = f_I(y) + f_E(y), \ y(0) = y_0, \tag{64}$$

and then they apply a forward Euler/backward Euler (FE/BE) pair to the right-hand side defined in (24):

$$\begin{aligned}\eta_n^{[p+1]} = \eta_{n-1}^{[p+1]} &+ h_n\big[f_I\big(\eta_n^{[p+1]}\big) - f_I\big(\eta_n^{[p]}\big)\big] \\ &+ h_n\big[f_E\big(\eta_{n-1}^{[p+1]}\big) - f_E\big(\eta_{n-1}^{[p]}\big)\big] + h\sum_{m=1}^{M} w_{n,m} f\big(\eta_m^{[p]}\big).\end{aligned} \tag{65}$$

With a straightforward extension the results from the present work point out that high-order accuracy can be achieved where there are no forward Euler time steps on the explicit term $f_E(y)$. That is, we propose examining the modified SISDC method

$$\eta_n^{[p+1]} = \eta_{n-1}^{[p+1]} + h_n\big[f_I\big(\eta_n^{[p+1]}\big) - f_I\big(\eta_n^{[p]}\big)\big] + h\sum_{m=1}^{M} w_{n,m} f\big(\eta_m^{[p]}\big). \tag{66}$$

Note that *the "base solver" for this method is not even consistent with the underlying ODE*. This serves as another example of how the findings from this work permit modifications to classical SDC methods in order to produce nearby variations. As an additional benefit, this reduces the computational coding complexity by asking the user to only define f and f_I as opposed to f, f_I, and f_E. This type of modification can be readily found after understanding the source of high-order accuracy inherent to the SDC framework.

Given that the iterative Picard methods demonstrate smaller errors by dropping the forward Euler terms in the right-hand side of the iterations from the SDC solver, one might expect that this method has better accuracy than its SISDC parent. In the event when $f_I = 0$ (or is small), this would be true because in that case we would be comparing explicit SDC to Picard iteration, and we have already shown that those errors are smaller for some problems. However, we will shortly see that this is not necessarily the case. Even though this modification does not affect the overall order of the solver, we will show that for the following test case it does not improve the total overall error of the solver. With that being said, we believe it is still important to understand the source of the overall order of the SDC solvers, because only then can new methods be developed from the existing framework.

mesh	SISDC	order	modified SISDC	order
4	2.24×10^{-02}		6.45×10^{-02}	
8	6.06×10^{-04}	5.21	2.84×10^{-03}	4.51
16	4.11×10^{-05}	3.88	1.91×10^{-04}	3.89
32	3.44×10^{-06}	3.58	1.46×10^{-05}	3.71
64	2.56×10^{-07}	3.75	1.01×10^{-06}	3.85
128	1.78×10^{-08}	3.85	6.68×10^{-08}	3.92
256	1.17×10^{-09}	3.93	4.29×10^{-09}	3.96
512	7.26×10^{-11}	4.01	2.69×10^{-10}	3.99

Table 5. Van der Pol oscillator. Here we present numerical results where we compare the implicit classical method defined in (24) as well as the modified semi-implicit SDC method defined in (66) against each other. Despite the fact that theory can show that the errors could be smaller for the modified method that relies solely on backward Euler time stepping embedded within Picard iteration, in this case the classical SISDC method based forward/backward Euler time stepping outperforms the other solver with its smaller error constants.

Van der Pol's equation. As a prototypical IMEX example, we include results for Van der Pol's equation

$$x''(t) = -x(t) + \mu(1 - x(t))^2 x'(t) \tag{67}$$

with appropriate initial conditions. After the usual transformation of $y_1(t) = x(t)$ and $y_2(t) = \mu x'(t)$, and rescaling time through $t \to t/\mu$, we have the system of differential equations [30; 25]

$$y_1' = y_2, \qquad y_2' = \frac{-y_1 + (1 - y_1^2)y_2}{\epsilon}, \qquad \epsilon = \frac{1}{\mu^2}. \tag{68}$$

In an IMEX setting, this problem is typically split into $f_E(y) = (y_2, 0)$ and $f_I = (0, (-y_1 + (1 - y_1^2)y_2)/\epsilon)$ as an effort to account for the stiffness as $\epsilon \to 0$. For this problem we only seek to verify the high-order accuracy of the classical semi-implicit SDC method defined in (24), denoted by SISDC, as well as the modified solver defined in (66), denoted by "modified SISDC". With this aim in mind, we set $\epsilon = 1$ so that the equations remain nonstiff, and we integrate to a long final time of $T = 4$. The initial conditions are the same as those found in an example in [25], which are $y_1(0) = 2$ and $y_2(0) = -0.666666654321$. In Table 5, we compare a convergence study for the fourth-order versions of these two methods where we use a total of four equispaced quadrature points, a provisional solution defined by the split forward/backward Euler method, as well as three corrections in the solver. For this problem, we find that the classical SISDC method has slightly smaller errors, despite what theory might otherwise predict we could observe. For problems where f_I is negligible or small, the modified method should outperform the SISDC solver.

Other values for ϵ produce similar findings, where the usual order reduction can be found as ϵ approaches zero. In other cases, the two solvers have similar behavior, and both show high-order accuracy for large values of ϵ. For brevity, these other results are omitted.

6. Conclusions

In this work we present rigorous error bounds for both explicit and implicit spectral deferred correction methods. Unlike most presentations that introduce SDC methods as methods that iteratively correct provisional solutions by solving an error equation, our work hinges on the fact that the basic solver can be recast as a variation on Picard iteration. This observation allows new SDC methods to be developed through modifications of the (forward or backward) Euler part of the iterative procedure. In addition, we present some analysis for SDC methods constructed with higher-order base solvers. In the numerical results section we present some sample variations that serve to indicate that the choice of the base solver need not be consistent with the underlying ODE in order to obtain a method that converges. That is, our findings indicate that it is not important to use the same low-order solver for each correction step because the desired high-order accuracy can be found in the integral of the residual. However, the choice of the base solver certainly has an impact on the overall scheme. For example, up to the degree of precision of the underlying quadrature rule, the choice of the forward or backward Euler method or even an inconsistent base solver leads to a single pickup on the order of accuracy of the solver with each correction step, whereas the choice of a trapezoidal rule for a base solver yields two orders of pickup with each correction step. For stiff problems, an implicit method constructed with the backward Euler method (or some variation of it) is certainly preferable due to the larger regions of absolute stability. Future work involves further analysis of embedded high-order base solvers as well as exploring further modifications of the solver to modify regions of absolute stability of existing solvers for explicit, implicit, and semi-implicit SDC methods.

References

[1] J. C. Butcher, *Implicit Runge–Kutta processes*, Math. Comp. **18** (1964), 50–64. MR Zbl

[2] T. Buvoli, *A class of exponential integrators based on spectral deferred correction*, preprint, 2015. arXiv

[3] A. Christlieb, W. Guo, M. Morton, and J.-M. Qiu, *A high order time splitting method based on integral deferred correction for semi-Lagrangian Vlasov simulations*, J. Comput. Phys. **267** (2014), 7–27. MR Zbl

[4] A. Christlieb, M. Morton, B. Ong, and J.-M. Qiu, *Semi-implicit integral deferred correction constructed with additive Runge–Kutta methods*, Commun. Math. Sci. **9** (2011), no. 3, 879–902. MR Zbl

[5] A. Christlieb and B. Ong, *Implicit parallel time integrators*, J. Sci. Comput. **49** (2011), no. 2, 167–179. MR Zbl

[6] A. Christlieb, B. Ong, and J.-M. Qiu, *Comments on high-order integrators embedded within integral deferred correction methods*, Commun. Appl. Math. Comput. Sci. **4** (2009), 27–56. MR Zbl

[7] ———, *Integral deferred correction methods constructed with high order Runge–Kutta integrators*, Math. Comp. **79** (2010), no. 270, 761–783. MR Zbl

[8] A. J. Christlieb, Y. Liu, and Z. Xu, *High order operator splitting methods based on an integral deferred correction framework*, J. Comput. Phys. **294** (2015), 224–242. MR Zbl

[9] A. J. Christlieb, C. B. Macdonald, and B. W. Ong, *Parallel high-order integrators*, SIAM J. Sci. Comput. **32** (2010), no. 2, 818–835. MR Zbl

[10] A. J. Christlieb, C. B. Macdonald, B. W. Ong, and R. J. Spiteri, *Revisionist integral deferred correction with adaptive step-size control*, Commun. Appl. Math. Comput. Sci. **10** (2015), no. 1, 1–25. MR Zbl

[11] M. Duarte and M. Emmett, *High order schemes based on operator splitting and deferred corrections for stiff time dependent PDEs*, preprint, 2014. arXiv

[12] A. Dutt, L. Greengard, and V. Rokhlin, *Spectral deferred correction methods for ordinary differential equations*, BIT **40** (2000), no. 2, 241–266. MR Zbl

[13] M. Emmett and M. L. Minion, *Toward an efficient parallel in time method for partial differential equations*, Commun. Appl. Math. Comput. Sci. **7** (2012), no. 1, 105–132. MR Zbl

[14] I. Faragó, *Note on the convergence of the implicit Euler method*, Numerical analysis and its applications (I. Dimov, I. Faragó, and L. Vulkov, eds.), Lecture Notes in Comput. Sci., no. 8236, Springer, 2013, pp. 1–11. MR Zbl

[15] T. Hagstrom and R. Zhou, *On the spectral deferred correction of splitting methods for initial value problems*, Commun. Appl. Math. Comput. Sci. **1** (2006), 169–205. MR Zbl

[16] E. Hairer, S. P. Nørsett, and G. Wanner, *Solving ordinary differential equations, I: Nonstiff problems*, 2nd ed., Springer Series in Computational Mathematics, no. 8, Springer, 1993. MR Zbl

[17] E. Hairer and G. Wanner, *Solving ordinary differential equations, II: Stiff and differential-algebraic problems*, 2nd ed., Springer Series in Computational Mathematics, no. 14, Springer, 1996. MR Zbl

[18] E. Hairer, C. Lubich, and G. Wanner, *Geometric numerical integration: structure-preserving algorithms for ordinary differential equations*, 2nd ed., Springer Series in Computational Mathematics, no. 31, Springer, 2006. MR Zbl

[19] A. C. Hansen and J. Strain, *On the order of deferred correction*, Appl. Numer. Math. **61** (2011), no. 8, 961–973. MR Zbl

[20] J. Huang, J. Jia, and M. Minion, *Accelerating the convergence of spectral deferred correction methods*, J. Comput. Phys. **214** (2006), no. 2, 633–656. MR Zbl

[21] ———, *Arbitrary order Krylov deferred correction methods for differential algebraic equations*, J. Comput. Phys. **221** (2007), no. 2, 739–760. MR Zbl

[22] J. Jia, J. C. Hill, K. J. Evans, G. I. Fann, and M. A. Taylor, *A spectral deferred correction method applied to the shallow water equations on a sphere*, Mon. Weather Rev. **141** (2013), 3435–3449.

[23] S. Y. Kadioglu and V. Colak, *An essentially non-oscillatory spectral deferred correction method for conservation laws*, Int. J. Comput. Methods **13** (2016), no. 5, art. id. 1650027. MR Zbl

[24] J. Kuntzmann, *Neuere Entwicklungen der Methode von Runge und Kutta*, Z. Angew. Math. Mech. **41** (1961), no. S1, T28–T31. Zbl

[25] A. T. Layton, *On the choice of correctors for semi-implicit Picard deferred correction methods*, Appl. Numer. Math. **58** (2008), no. 6, 845–858. MR Zbl

[26] ———, *On the efficiency of spectral deferred correction methods for time-dependent partial differential equations*, Appl. Numer. Math. **59** (2009), no. 7, 1629–1643. MR Zbl

[27] A. T. Layton and M. L. Minion, *Conservative multi-implicit spectral deferred correction methods for reacting gas dynamics*, J. Comput. Phys. **194** (2004), no. 2, 697–715. MR Zbl

[28] ———, *Implications of the choice of quadrature nodes for Picard integral deferred corrections methods for ordinary differential equations*, BIT **45** (2005), no. 2, 341–373. MR Zbl

[29] ———, *Implications of the choice of predictors for semi-implicit Picard integral deferred correction methods*, Commun. Appl. Math. Comput. Sci. **2** (2007), 1–34. MR Zbl

[30] M. L. Minion, *Semi-implicit spectral deferred correction methods for ordinary differential equations*, Commun. Math. Sci. **1** (2003), no. 3, 471–500. MR Zbl

[31] ———, *A hybrid parareal spectral deferred corrections method*, Commun. Appl. Math. Comput. Sci. **5** (2010), no. 2, 265–301. MR Zbl

[32] M. M. Morton, *Integral Deferred Correction methods for scientific computing*, Ph.D. thesis, Michigan State University, 2010. MR

[33] W. Pazner and P.-O. Persson, *Stage-parallel fully implicit Runge–Kutta solvers for discontinuous Galerkin fluid simulations*, J. Comput. Phys. **335** (2017), 700–717. MR Zbl

[34] W. Qu, N. Brandon, D. Chen, J. Huang, and T. Kress, *A numerical framework for integrating deferred correction methods to solve high order collocation formulations of ODEs*, J. Sci. Comput. **68** (2016), no. 2, 484–520. MR Zbl

[35] D. Ruprecht and R. Speck, *Spectral deferred corrections with fast-wave slow-wave splitting*, SIAM J. Sci. Comput. **38** (2016), no. 4, A2535–A2557. MR Zbl

[36] R. Speck, D. Ruprecht, M. Emmett, M. Minion, M. Bolten, and R. Krause, *A multi-level spectral deferred correction method*, BIT **55** (2015), no. 3, 843–867. MR Zbl

[37] T. Tang, H. Xie, and X. Yin, *High-order convergence of spectral deferred correction methods on general quadrature nodes*, J. Sci. Comput. **56** (2013), no. 1, 1–13. MR Zbl

[38] M. Weiser, *Faster SDC convergence on non-equidistant grids by DIRK sweeps*, BIT **55** (2015), no. 4, 1219–1241. MR Zbl

[39] Y. Xia, Y. Xu, and C.-W. Shu, *Efficient time discretization for local discontinuous Galerkin methods*, Discrete Contin. Dyn. Syst. Ser. B **8** (2007), no. 3, 677–693. MR Zbl

[40] P. E. Zadunaisky, *On the estimation of errors propagated in the numerical integration of ordinary differential equations*, Numer. Math. **27** (1976), no. 1, 21–39. MR Zbl

Received June 19, 2017. Revised November 7, 2018.

MATHEW F. CAUSLEY: mcausley@kettering.edu
Mathematics Department, Kettering University, Flint, MI, United States

DAVID C. SEAL: seal@usna.edu
Department of Mathematics, United States Naval Academy, Annapolis, MD, United States

COMM. APP. MATH. AND COMP. SCI.
Vol. 14, No. 1, 2019
dx.doi.org/10.2140/camcos.2019.14.65

A THEORETICAL STUDY OF AQUEOUS HUMOR SECRETION BASED ON A CONTINUUM MODEL COUPLING ELECTROCHEMICAL AND FLUID-DYNAMICAL TRANSMEMBRANE MECHANISMS

LORENZO SALA, AURELIO GIANCARLO MAURI, RICCARDO SACCO, DARIO MESSENIO, GIOVANNA GUIDOBONI AND ALON HARRIS

Intraocular pressure, resulting from the balance of aqueous humor (AH) production and drainage, is the only approved treatable risk factor in glaucoma. AH production is determined by the concurrent function of ion pumps and aquaporins in the ciliary processes, but their individual contribution is difficult to characterize experimentally. In this work, we propose a novel unified modeling and computational framework for the finite element simulation of the role of the main ion pumps and exchangers involved in AH secretion, namely, the sodium-potassium pump, the calcium-sodium exchanger, the chloride-bicarbonate exchanger, and the sodium-proton exchanger. The theoretical model is developed at the cellular scale and is based on the coupling between electrochemical and fluid-dynamical transmembrane mechanisms characterized by a novel description of the electric pressure exerted by the ions on the intrapore fluid that includes electrochemical and osmotic corrections. Considering a realistic geometry of the ion pumps, the proposed model is demonstrated to correctly predict their functionality as a function of (1) the permanent electric charge density over the pore surface, (2) the osmotic gradient coefficient, and (3) the stoichiometric ratio between the ion pump currents enforced at the inlet and outlet sections of the pore. In particular, theoretical predictions of the transepithelial membrane potential for each simulated pump/exchanger allow us to perform a first significant model comparison with experimental data for monkeys. This is a significant step for future multidisciplinary studies on the action of molecules on AH production.

1. Introduction

The flow of aqueous humor (AH) and its regulation play an important role in ocular physiology by contributing to control the level of intraocular pressure (IOP) [36; 28]. Elevated IOP is the only approved treatable risk factor in glaucoma, an

MSC2010: primary 35K59, 65M60, 76D07, 92C35, 92C37; secondary 76Z05.
Keywords: ion exchangers, eye, ion pumps, aqueous humor, mathematical modeling, simulation, finite element method.

optic neuropathy characterized by a multifactorial aetiology with a progressive degeneration of retinal ganglion cells that ultimately leads to irreversible vision loss [18; 59]. Currently, glaucoma affects more than 60 million people worldwide and is estimated to reach almost 80 million by 2020 [38]. IOP can be lowered via hypotonizing eye drops and/or surgical treatment, and it can be shown that reducing IOP by 1 mmHg has the effect of reducing the risk of glaucoma progression and subsequent vision loss by 10% [19].

Several classes of IOP-lowering medications are available for use in patients with glaucoma, including prostaglandin analogues, beta-blockers, carbonic anhydrase inhibitors, and alpha-2-adrenergic agonists, in fixed and variable associations, while newer classes are still in clinical trials, such as the rho kinase inhibitors [17; 21; 39]. All currently available IOP-lowering agents function by altering AH production or drainage. However, differences in drug efficacy have been observed among patients that cannot be completely explained without a clear understanding of the mechanisms regulating AH flow. Motivated by this need, in this work we focus on AH production and we propose a mathematical approach to model and simulate the contribution of ion pumps and exchanger to determine AH flow.

The production of AH takes places in the ciliary processes within the ciliary body, where clear liquid flows across the ciliary epithelium, a two-layered structure composed of an inner nonpigmented layer, representing the continuation of retinal pigmented epithelium, and an external nonpigmented layer, representing the continuation of the retina [16], as illustrated in Figure 1.

Three main mechanisms are involved in the production of AH: (i) convective delivery of fluid and metabolic components via the ciliary circulation, (ii) ultrafiltration and diffusion of fluid and metabolic components across the epithelial cells driven by gradients in hydrostatic pressure, oncotic pressure, and metabolite concentrations, and (iii) active secretion into the posterior chamber driven by increased ion concentrations within the basolateral space between nonpigmented epithelial cells.

In this work we focus on the third mechanism, henceforth referred to as *AH secretion*, which is responsible for approximately 80–90% of the whole AH production process [15; 32]. More precisely, we aim to model the selective movement of anions and cations across the membrane of the nonpigmented epithelial cells, the resulting gradient of ion and solute concentrations across the membrane, and the induced fluid egression into the posterior chamber.

The proposed simulation of AH secretion presents many challenges from the modeling viewpoint. Existing references concerning AH secretion are primarily based on lumped parameter models that provide a systemic view of AH flow [7; 29; 52], but do not reproduce the detailed phenomena occurring at the level of single ion pumps and exchangers. Detailed models based on the velocity-extended Poisson–Nernst–Planck (VE-PNP) system have been utilized to simulate electrokinetic flows

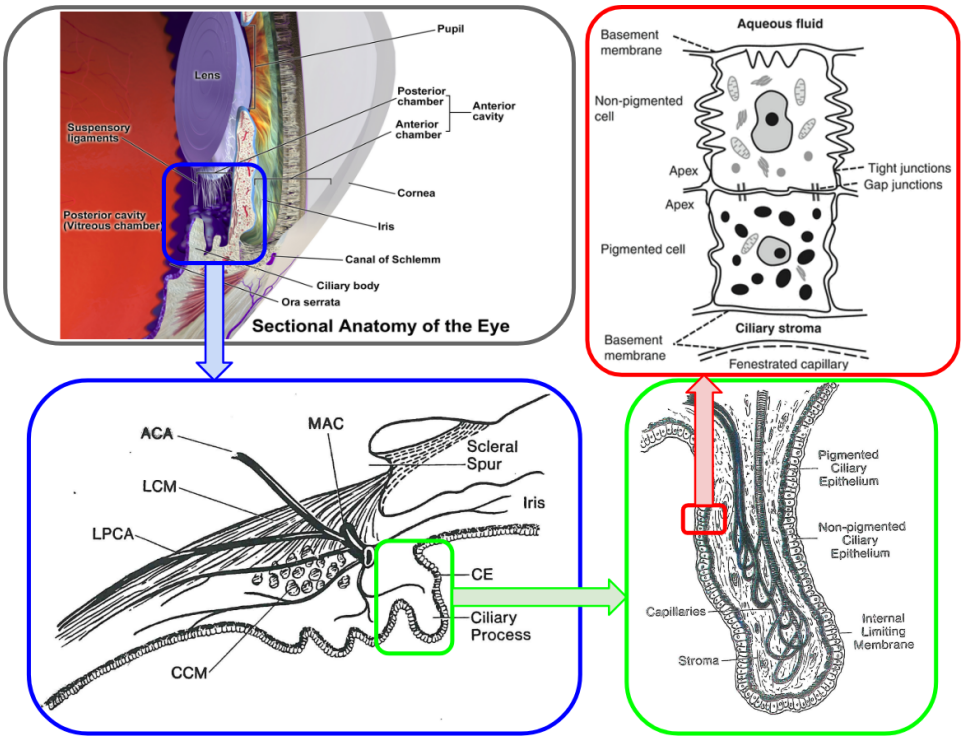


Figure 1. Left top: anatomy of the eye and of the structures involved in aqueous humor production and regulation [5]. Left bottom: MAC: major arterial circle; ACA: anterior ciliary arteries; LPCA: long posterior ciliary artery; LCM: longitudinal (fibers) ciliary muscle; CCM: circular (fibers) ciliary muscle; CE: ciliary epithelium (figure reproduced from [49]). Right bottom: a ciliary process is composed of capillaries, stroma, and two layers of epithelium (inner, pigmented and outer, nonpigmented) (figure reproduced from [49]). Right top: the two-layer structure of the ciliary epithelium (figured reproduced from [56]).

[26; 3; 4], but different models for the volumetric force coupling electrochemical and fluid-dynamical mechanisms have been proposed [51; 41; 42; 13], thereby raising the question of which one, if any, is the most appropriate for the application at hand. In addition, some of the most important parameters in the VE-PNP model, such as the concentration of ions within the pore, the fixed charge on the pore lateral surface, and the osmotic diffusive parameter, cannot be easily accessed experimentally and so are not readily available in the literature. In the pilot investigation conducted in [35], we explored the feasibility of utilizing a VE-PNP model to simulate the sodium-potassium pump (Na^+-K^+) within the nonpigmented epithelial cells of the eye. However, several other ion pumps and exchangers are involved in AH secretion [28], and have not yet been modeled in the context of AH flow.

The present work aims at extending the modeling and simulation treatment of [35] through the development of a unified framework capable of simulating

AH secretion by including the four main ion pumps and exchangers involved in the process, namely, the calcium-sodium exchanger (Ca^{++}-Na^{+}), the chloride-bicarbonate exchanger (Cl^{-}-HCO_3^{-}), and sodium-proton exchanger (Na^{+}-H^{+}). The computational structure used in the numerical simulations is based on the adoption of (1) a temporal semidiscretization with the backward Euler method, (2) a Picard iteration to successively solve the equation system at each discrete time level, and (3) a spatial discretization of each differential subproblem obtained from system decoupling using the Galerkin finite element method.

Remark. The present work focuses on the study of the system when steady-state conditions are reached.

Numerical simulations are utilized to (i) compare how and to what extent different modeling choices for the volumetric coupling force affect the resulting transmembrane potential, stoichiometric ratio, and intrapore fluid velocity and (ii) characterize the correct boundary conditions and the value of the permanent electric charge density on the pore lateral surface that allow us to predict a biophysically reasonable behavior of ion pumps and exchangers in realistic geometries. Overall, this work provides the first systematic investigation of VE-PNP models in the context of AH secretion and paves the way to future studies on biochemical, pharmacological, and therapeutic aspects of AH flow regulation.

The paper is organized as follows. Section 2 provides a brief functional description of the main features pertaining to the Na^{+}-K^{+} pump and the Ca^{++}-Na^{+}, Cl^{-}-HCO_3^{-}, and Na^{+}-H^{+} exchangers. The VE-PNP system is described in Section 3, and the mathematical model for the volume force density in the right-hand side of the linear momentum balance equation for the intrapore fluid is described in Section 4. The numerical discretization of the VE-PNP model equations is discussed in Section 5, whereas simulation results of the effect of volumetric forces and permanent electric charge density are presented in Sections 6 and 7, respectively. Model limitations, conclusions, and future perspectives are outlined in Section 8.

2. Ion pumps and exchangers in AH secretion

In this section we provide a short description of the main ion pumps and exchangers that are involved in the process of AH secretion. A schematic representation of these ion pumps and exchangers is illustrated in Figure 2. We refer to [20] for an overview of ion pumps and exchangers in cellular biology and to [27] for their mathematical treatment.

The sodium-potassium pump. This pump plays a fundamental role in cellular biology as it is present in the membrane of every cell in the human body. The enzyme ATPase, located either in pigmented or nonpigmented ciliary epithelium,

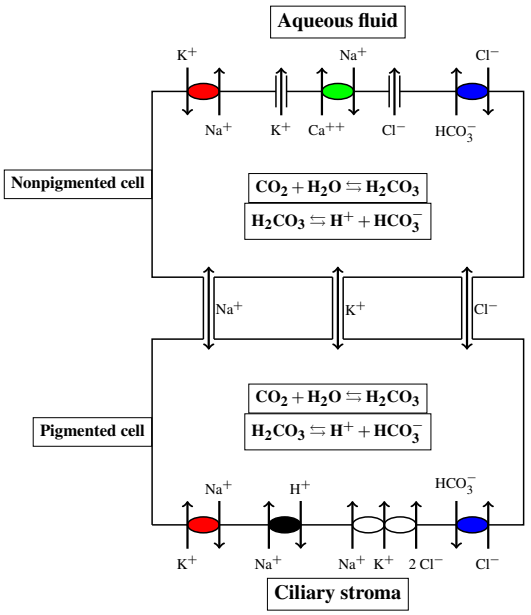


Figure 2. Schematic diagram of ion pumps and exchangers located on the lipid membrane in the nonpigmented epithelial cells of the ciliary body of the eye. Aqueous humor is produced by the active secretion of fluid through the ion pumps and exchangers during their activity. This figure is inspired by Figure 9 of [48].

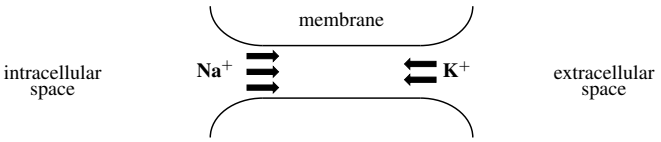


Figure 3. Schematic representation of the Na^+-K^+ pump. The stoichiometric ratio is 3 : 2, since there is an outflux of three Na^+ ions and an influx of two K^+ ions.

causes the hydrolysis of one molecule of ATP and produces the necessary energy to expel three Na^+ ions, while allowing two K^+ ions to enter. This process is not electrically neutral as it entails an outflux of three positive charged particles of sodium and an influx of only two positive charged particles of potassium. The ion outflux and influx are schematically represented in Figure 3.

The calcium-sodium exchanger. This exchanger is activated when calcium accumulates inside the cell above a certain threshold that is usually around 1 mM. It entails the influx of three Na^+ ions and an outflux of one Ca^{++} ion. As in the previous case, this process is not electrically neutral as three positive sodium ions enter the cell whereas only one positive calcium ion exits the cell. The ion outflux and influx are schematically represented in Figure 4.

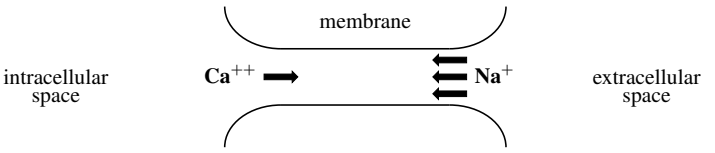


Figure 4. Schematic representation of the Ca^{++}-Na^{+} pump. The stoichiometric ratio is 3 : 1, since there is an influx of three Na^{+} ions and an outflux of one Ca^{++} ion.

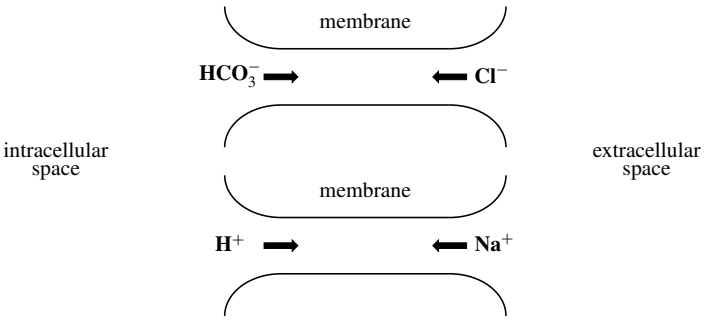


Figure 5. Schematic representation of the chloride-bicarbonate (top) and the sodium-proton exchangers (bottom). The stoichiometric ratio of the chloride-bicarbonate exchanger is 1 : 1, since there is an influx of one Cl^- ion and an outflux of one HCO_3^- ion. The stoichiometric ratio of the sodium-proton exchanger is also 1:1, since there is an influx of one Na^+ ion and an outflux of one H^+ ion.

The chloride-bicarbonate exchanger. This exchanger involves the movement of negative ions. Carbonic anhydrase is an enzyme that mediates the transport of bicarbonate across the ciliary epithelium to maintain the homeostatic balance of carbonate across the cell membrane. More precisely, carbonic anhydrase favors the splitting of one molecule of H_2CO_3 into a positive H^+ ion and a negative HCO_3^- ion. Then, the HCO_3^- ion exits the cell through the pore with an exchange of a chlorine ion Cl^- that enters the cell. The balance of this exchanger is electrically neutral because for every negative charged HCO_3^- leaving the cell there is a negative charged Cl^- ion entering the cell [58]. The ion outflux and influx are schematically represented in the top panel of Figure 5.

The sodium-proton exchanger. This exchanger is strictly correlated with the activity of the chloride-bicarbonate exchanger previously described. A positive H^+ ion resulting from the splitting reaction of one molecule of H_2CO_3 exits the cell with an exchange of one Na^+ ion entering the cell. Thus, this exchanger is electrically neutral. The ion outflux and influx are schematically represented in the bottom panel of Figure 5.

3. The mathematical model

In this section we illustrate the system of partial differential equations (PDEs) constituting the mathematical model at the cellular scale level of ion pumps and exchangers that activate the AH secretion. We refer to [45] for a detailed discussion of the analytical properties of the model equations and of the numerical methods used for their discretization, and to [35] for preliminary results on the adoption of the model in the study of the role of bicarbonate ion to correctly determine the electrostatic potential drop across the cellular membrane at the level of eye transepithelium.

The geometrical setting that we consider henceforth is the computational domain Ω illustrated in Figure 6 representing a cross-section of a simplified pore geometry.

Remark. The pore geometry for real ion pumps is much more complex than the simple cylinder depicted in Figure 6. It has also been observed that the pore geometry might vary during the activity of the pump [60]. Even though a more complicated geometry is considered in Section 7, the present work has to be considered as a first step towards more realistic geometric settings.

Such representation includes any of the ion pumps and exchangers described in Section 2, which are located on the lipid bilayer constituting the membrane of the nonpigmented epithelial cells of the ciliary body of the eye schematically depicted in Figure 1 (right top). We indicate by $\partial\Omega$ the boundary of Ω, by $\underline{n}$ the outward unit normal vector on $\partial\Omega$, and by sideA, sideB, and Γ_{LAT} the intracellular surface, the extracellular surface, and the lateral boundary, respectively, in such a way that $\partial\Omega = \mathrm{sideA} \cup \mathrm{sideB} \cup \Gamma_{\mathrm{LAT}}$. For a given starting time t_0 and a given observational time window T_{obs}, we set $I_T := (t_0, t_0 + T_{\mathrm{obs}})$ and we denote by $Q_T := \Omega \times I_T$ the space-time cylinder in which we study the spatial and temporal evolution of the process of AH secretion at the cellular scale level. To clarify the physical foundation of the cellular scale model object of the present article, we assume that the following strongly coupled mechanisms concur to determine AH secretion:

electric field formation. This mechanism is determined by the mutual interaction among ions in the intrapore fluid and their interaction with the permanent electric charge density distributed on the surface of the pore structure. Mathematically, the mechanism is described by the Poisson equation, supplied by appropriate boundary conditions at the inlet and outlet sections of the pore and on its external surface.

ion motion. This mechanism is determined by the superposition of a diffusion process driven by ion concentration gradients along the pore and of a drift process driven by the force exerted by the electric field on each ion charged particle. Mathematically, the mechanism is described by the Nernst–Planck

equations, supplied by appropriate initial conditions inside the pore and by appropriate boundary conditions at the inlet and outlet sections of the pore and on its external surface.

fluid motion. This mechanism is determined by the volume force density that is exerted by the charged ion particles because of their motion inside the pore. Mathematically, the mechanism is described by the time-dependent Stokes equations, supplied by appropriate initial conditions inside the pore and by appropriate boundary conditions at the inlet and outlet sections of the pore and on its external surface.

We refer to the resulting mathematical model as *velocity-extended Poisson–Nernst–Planck* system (VE-PNP) [44; 24; 47; 25]. The VE-PNP system can be derived by the application of the following physical laws [45]: (i) mass balance for each of the M chemical species included in the system (1a), (ii) linear momentum balance for each chemical species (1b), (iii) electric charge conservation (1c), (iv) mass balance for intrapore fluid (1e), and (v) linear momentum balance for the mixture of intrapore fluid and ion species (1f). Ultimately, the system consists of the following set of PDEs to be solved in Q_T:

$$\frac{\partial n_i}{\partial t} + \operatorname{div} \underline{f}_i = 0 \qquad \text{for all } i = 1, \dots, M, \tag{1a}$$

$$\underline{f}_i = \frac{z_i}{|z_i|}\mu_i n_i \underline{E} - D_i \nabla n_i + \boxed{n_i \underline{u}} \qquad \text{for all } i = 1, \dots, M, \tag{1b}$$

$$\operatorname{div}(-\epsilon_f \nabla \varphi) = q \sum_{i=1}^{M} z_i n_i + q\rho_{\text{fixed}}, \tag{1c}$$

$$\underline{E} = -\nabla \varphi, \tag{1d}$$

$$\operatorname{div} \underline{u} = 0, \tag{1e}$$

$$\rho_f \frac{\partial \underline{u}}{\partial t} = \operatorname{div} \underline{\underline{\sigma}}(\underline{u}, p) + \boxed{\underline{F}_{\text{ion}}}, \tag{1f}$$

$$\underline{\underline{\sigma}}(\underline{u}, p) = 2\mu_f \underline{\underline{S}}(\underline{u}) - p\,\underline{\underline{\delta}}, \tag{1g}$$

$$\underline{\underline{S}}(\underline{u}) = \underline{\underline{\nabla}}_s \underline{u} = \tfrac{1}{2}(\nabla \underline{u} + (\nabla \underline{u})^T). \tag{1h}$$

The equation set (1) comprises two main blocks. Equations (1a)–(1d) constitute the Poisson–Nernst–Planck (PNP) block whereas (1e)–(1h) constitute the Stokes block. As far as the PNP block is concerned, M is the number of ion species, $\underline{f}_i$ denotes the ion particle flux [$\mathrm{cm^{-2}\,s^{-1}}$], n_i is the ion concentration [$\mathrm{cm^{-3}}$], and μ_i and D_i are the ion electric mobility [$\mathrm{cm^2\,V\,s^{-1}}$] and diffusivity [$\mathrm{cm^2\,s^{-1}}$], respectively. The mobility μ_i is proportional to the diffusivity D_i through the

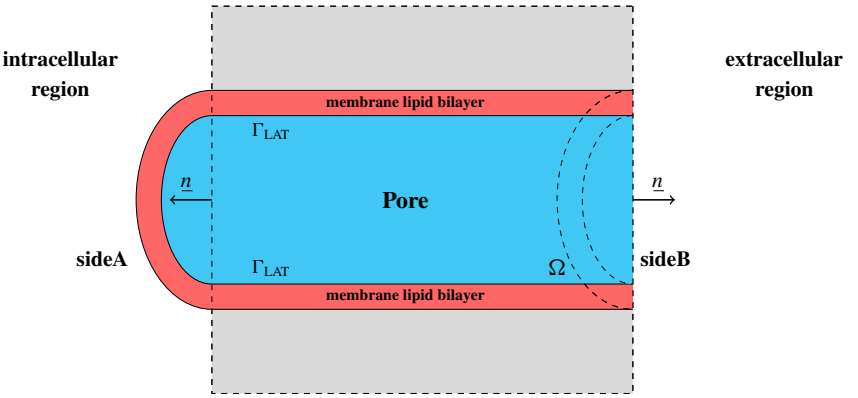


Figure 6. Cross-section of a simplified pore geometry. The lipid membrane bilayer is represented in brown color. The pore is represented in cyan color. The portions of the boundary $\partial\Omega$ are labeled as sideA (intracellular side), sideB (extracellular side), and Γ_{LAT} (lateral surface).

Einstein relation

$$D_i = \frac{K_B T}{q|z_i|}\mu_i, \quad i = 1, \ldots, M, \tag{2}$$

where q is the electron charge [C], K_B is the Boltzmann constant [$\mathrm{cm^2\,g\,s^{-2}\,K^{-1}}$], and T is the absolute temperature [K]. In (1c) and (1d), $\underline{E}$ is the electric field [$\mathrm{V\,cm^{-1}}$], φ is the electric potential [V], and ϵ_f is the electrolyte fluid dielectric permittivity [$\mathrm{F\,cm^{-1}}$]. The quantity z_i is the valence of the i-th ion ($z_i > 0$ for cations, $z_i < 0$ for anions) whereas ρ_{fixed} is the permanent electric charge density [$\mathrm{C\,cm^{-3}}$]. We define the ion current density $\underline{J}_i$ [$\mathrm{A\,cm^{-2}}$] of each ion species as

$$\underline{J}_i = q z_i \underline{f}_i, \quad i = 1, \ldots, M. \tag{3}$$

As far as the Stokes block is concerned, $\underline{u}$ is the mixture velocity [$\mathrm{cm\,s^{-1}}$] and mixture incompressibility is expressed by (1e). Equations (1g) and (1h) are the constitutive laws for the stress tensor [$\mathrm{dyne\,cm^{-2}}$] and the strain rate [$\mathrm{s^{-1}}$], respectively, where p denotes the mixture pressure [$\mathrm{dyne\,cm^{-2}}$], μ_f is the mixture viscosity [$\mathrm{g\,cm^{-1}\,s^{-1}}$], ρ_f is the mixture mass density [$\mathrm{g\,cm^{-3}}$], and $\underline{\underline{\delta}}$ the second-order identity tensor of dimension 3. Since the focus of the article is the investigation of the role of electrochemical forces on the secretion of AH across the nonpigmented epithelial cells in the ciliary body of the eye, we assume that the temperature T of the intrapore mixture is constant and equal to the value $T_0 = 293.75$ K and we will refer to the mixture of water and ion species as AH fluid. The boxed terms in (1b) and (1f) are the contributions that introduce the coupling between the PNP block of

system (1) and the Stokes block of system (1). In particular,

$$\underline{F}_{\text{ion}} = \sum_{i=1}^{M} \underline{F}_i \tag{4}$$

expresses the volume force density [dyne cm^{-3}] exerted by the ion charges on the intrapore AH fluid, the quantities $\underline{F}_i$ representing the contribution to the volume force density given by each ion species, $i = 1, \ldots, M$.

The equation system (1) is equipped with the initial conditions

$$n_i(\underline{x}, 0) = n_i^0(\underline{x}), \quad i = 1, \ldots, M, \quad \underline{x} \in \Omega, \tag{5a}$$

$$\underline{u}(\underline{x}, 0) = \underline{u}^0(\underline{x}), \qquad \underline{x} \in \Omega, \tag{5b}$$

where n_i^0 are given positive functions and $\underline{u}^0$ is a given function. The initial condition $\varphi^0 = \varphi^0(\underline{x})$, $\underline{x} \in \Omega$, is the solution of the equation set (1c)–(1d), under appropriate boundary conditions on $\partial\Omega$, having set $n_i = n_i^0$, $i = 1, \ldots, M$.

The boundary conditions associated with system (1) that are considered in the present article are of mixed Dirichlet–Neumann type. Their characterization for each equation in the system is specified in each simulation illustrated in Sections 6 and 7.

4. Model for the volume force density

Many approaches have been adopted in the literature to model volume force density dictated by different needs in various contexts. One of the most used force models is the Stratton model [51] for both its simplicity and efficacy. However, more sophisticated modeling approaches are needed to account for microscopic phenomena such as drift and diffusion effects in semiconductor devices [37] or the effect of particle size exclusion that is well described by the hard sphere theory [43; 40]. Our idea is to unify the various volume force models proposed in the literature, compare their different impact in our problem, and select the more appropriate one.

In this section, therefore, we discuss a general approach to the modeling of the volume force density $\underline{F}_i$ on the right-hand side in the linear momentum balance equation (1f). $\underline{F}_i$ expresses the contribution from the i-th ion species, $i = 1, \ldots, M$, to the total volume force density exerted by the ion charged particles on the intrapore fluid and is assumed henceforth to be characterized by the relation

$$\underline{F}_i = q z_i n_i \underline{E}_i^{\text{echs}} - k_{\text{osm}} \nabla n_i, \quad i = 1, \ldots, M. \tag{6}$$

The first term on the right-hand side of (6) represents the volume force density due to a generalized electrochemical field $\underline{E}_i^{\text{echs}}$ whereas the second term represents the volume force density due to an osmotic concentration gradient according to

the parameter k_{osm} [N m] [23]. The generalized electrochemical field is the result of the superposed effect of passive drift due to the electric field (e), the diffusion mechanism associated with a chemical concentration gradient (c), and the particle size exclusion effect associated with the hard sphere (hs) theory [43; 40]. Relation (6) is referred to henceforth as an electrochemical model including osmotic and size exclusion mechanisms (echsk).

Assuming that $\underline{E}_i^{\mathrm{echs}}$ is a gradient field, we have

$$\underline{E}_i^{\mathrm{echs}} = -\nabla\varphi_i^{\mathrm{echs}}, \quad i = 1, \ldots, M, \tag{7a}$$

$$\varphi_i^{\mathrm{echs}} = \varphi_i^{\mathrm{ec}} + \mu_i^{\mathrm{ex}}, \quad i = 1, \ldots, M, \tag{7b}$$

where φ_i^{ec} is the generalized electrochemical potential of the i-th species

$$\varphi_i^{\mathrm{ec}} = \varphi + \frac{V_{\mathrm{th}}}{z_i} \ln\left(\frac{n_i}{n_{\mathrm{ref}}}\right), \quad i = 1, \ldots, M, \tag{7c}$$

and μ_i^{ex} is the exclusion effect potential of the i-th species [6]

$$\begin{aligned}\mu_i^{\mathrm{ex}} = -V_{\mathrm{th}}\Bigg[& \ln\left(1 - \frac{4\pi}{3}\sum_{k=1}^{M} n_k R_k^3\right) \\ & + 4\pi\,\frac{R_i\left(\sum_{k=1}^{M} n_k R_k^2\right) + R_i^2\left(\sum_{k=1}^{M} n_k R_k\right) + \frac{1}{3}R_i^3\left(\sum_{k=1}^{M} n_k\right)}{1 - (4\pi/3)\sum_{k=1}^{M} n_k R_k^3} \\ & + \frac{16\pi^2}{3}\,\frac{R_i^3\left(\sum_{k=1}^{M} n_k R_k\right)\left(\sum_{k=1}^{M} n_k R_k^2\right) + \frac{3}{2}R_i^2\left(\sum_{k=1}^{M} n_k R_k^2\right)^2}{\left(1 - (4\pi/3)\sum_{k=1}^{M} n_k R_k^3\right)^2} \\ & + \frac{64\pi^3}{9}\,\frac{R_i^3\left(\sum_{k=1}^{M} n_k R_k^2\right)^3}{\left(1 - (4\pi/3)\sum_{k=1}^{M} n_k R_k^3\right)^3}\Bigg], \quad i = 1, \ldots, M, \end{aligned} \tag{7d}$$

n_{ref} and R_i being positive constants [cm^{-3}] representing reference concentration and radius of the i-th ion species, respectively. Relation (6) is indeed a general view of the volume force density from which simpler expressions of $\underline{F}_i$ can be derived. These models constitute a hierarchy characterized by an increasing number of approximations and a consequent decreasing level of physical complexity.

echs (*electrochemical model including hard sphere theory*). This model is derived from (6) by neglecting the contribution of the osmotic gradient. This includes the Coulomb electric force associated with a charge density, the chemical gradient, and size exclusion mechanisms. It is mathematically expressed by

$$\underline{F}_i = q z_i n_i \underline{E}_i^{\mathrm{echs}}, \quad i = 1, \ldots, M. \tag{8}$$

ec (*electrochemical model*). This is derived from (8) neglecting the size exclusion phenomena. This includes the Coulomb electric force associated with a charge density and the chemical gradient mechanism. It is mathematically expressed by

$$\underline{F}_i = q z_i n_i \underline{E}_i^{\mathrm{ec}}, \quad i = 1, \ldots, M, \tag{9a}$$

$$\underline{E}_i^{\mathrm{ec}} = -\nabla \varphi_i^{\mathrm{ec}}, \qquad i = 1, \ldots, M. \tag{9b}$$

Stratton (*Stratton model*). This model is derived from (9a) by neglecting the contribution induced by the chemical gradient. This was originally proposed in [51], and it is widely adopted in the modeling description of electrokinetic phenomena [26]. It is mathematically expressed by

$$\underline{F}_i = q z_i n_i \underline{E}, \quad i = 1, \ldots, M. \tag{10}$$

eck (*electrochemical model including osmotic force*). This model is derived from (6) by neglecting the contribution of the size exclusion phenomena. This model includes the Coulomb electric force associated with a charge density and the chemical and osmotic gradient mechanisms. It is mathematically expressed by

$$\underline{F}_i = q z_i n_i \underline{E}_i^{\mathrm{ec}} - k_{\mathrm{osm}} \nabla n_i, \quad i = 1, \ldots, M. \tag{11}$$

The effect on intrapore AH fluid motion induced by the above force models is analyzed, in the context of the Na^+-K^- pump, in Section 6 where the predicted electrolyte fluid velocity is compared with the volumetric force density $\underline{F}_i$ exerted by the ions on it.

5. Time advancing, functional iteration, and numerical discretization

In this section we provide a short description of the algorithm that is used to numerically solve system (1). We refer to [45] for more details on the stability and convergence analysis of the adopted methods as well as their implementation in the general-purpose C++ modular numerical code MP-FEMOS (Multi-Physics Finite Element Modeling Oriented Simulator) that has been developed by some of the authors [34; 33; 1].

The VE-PNP model mathematically represents a nonlinearly coupled system of PDEs of incomplete parabolic type because of the fact that at each time level the electrostatic potential φ and the electric field $\underline{E}$ must be updated as a function of the ion concentrations and of the fixed permanent charge by solving the elliptic Poisson equation (1c). In turn, the electric field $\underline{E}$ contributes to determine ion motion through the Nernst–Planck relation (1b) and fluid motion through the relation (6) for the volume force density.

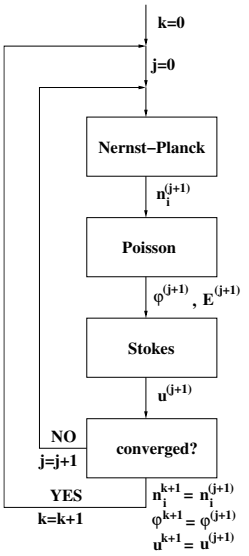


Figure 7. Flowchart of the computational algorithm to solve the VE-PNP system. The nonnegative integer k is the temporal discretization counter. The nonnegative integer j is the Picard iteration counter.

To disentangle the various coupling levels that are present in the VE-PNP system, we proceed as follows:

(1) We perform a temporal semidiscretization of the problem by resorting to the backward Euler (BE) method.

(2) We introduce a Picard iteration to successively solve the equation system at each discrete time level.

(3) We perform a spatial discretization of each differential subproblem obtained from system decoupling using the Galerkin finite element method (GFEM).

The use of the BE method has the twofold advantage that (a) it is unconditionally absolute stable, (b) it is monotone.

Property (a) allows us to take relatively large time steps, thus *reducing the computational effort to reach steady-state conditions*. Property (b) combined with an analogous one for the spatial discretization scheme of the Nernst–Planck equations ensures that the *computed ion concentrations are positive for all discrete time levels*.

The use of a Picard iteration has the twofold advantage that (c) it is a decoupled algorithm and (d) a maximum principle is satisfied by the solutions of two of the boundary value problems (BVPs) obtained from decoupling. Property (c) amounts to transforming the nonlinearly coupled system (1) into the successive solution of three sets of linear BVPs of reduced size. Property (d) implies the existence of an invariant region for the electric potential depending only on the boundary

data and on the fixed permanent charge and the positivity of the computed ion concentrations.

The use of the GFEM has the twofold advantage that (e) it can easily handle complex geometries and (f) provides an accurate and stable numerical approximation of the solution of each BVP obtained from system decoupling. Property (e) is implemented through the partition of the domain of biophysical interest into the union of tetrahedral elements of variable size. Property (f) is implemented through the use of piecewise linear finite elements for the Poisson equation, piecewise linear finite elements with exponential fitting stabilization along of mesh edges for the Nernst–Planck equations, and the inf-sup stable Taylor–Hood finite element pair for the Stokes equations.

Figure 7 illustrates the flowchart of the temporal semidiscretization for time advancing with the BE method and the Picard iteration used to successively solve the three linear equation subsystems.

6. Comparison of volumetric force models on an idealized sodium-potassium pump

The aim of this section is to compare the different descriptions introduced in Section 4 of the volume force density $\underline{F}_{\text{ion}}$ exerted on the intrapore fluid in the linear momentum balance equation (1f). In particular we investigate the biophysical reliability of the various models to describe the functionality of an idealized version of the Na^+-K^+ pump illustrated in Section 2 in which the simultaneous presence of Na^+, K^+, Cl^-, and HCO_3^- ion species is considered. The analysis criteria are based on the comparison of simulation results with (i) the electrostatic potential drop measured across the transepithelial membrane, (ii) the theoretical stoichiometric ratio 3 : 2, and (iii) the direction of the AH flow from the cell into the basolateral space. The ideality of the Na^+-K^+ pump is represented by the geometry adopted for numerical simulation, consisting of the cylinder with axial length $L_{\text{ch}} = 5\,\text{nm}$ and radius $R_{\text{ch}} = 0.4\,\text{nm}$ shown in Figure 8 together with its partition into 37075 tetrahedral finite elements. We point out that the above geometrical setting, despite being a simplified approximation of the real structure, has been successfully employed for biological investigations in [1; 45; 35].

Boundary and initial conditions. Because of the complexity of the boundary conditions (BCs) and initial conditions (ICs) involved in the simulation of the pump, it is useful to accurately describe them for each equation in (1).

Poisson equation. For all $t \in I_T$, the BCs for the Poisson equation (1c)–(1d) are

$$\varphi = 0 \qquad \text{on sideA}, \tag{12a}$$

$$\underline{D} \cdot \underline{n} = 0 \qquad \text{on sideB} \cup \Gamma_{\text{LAT}}. \tag{12b}$$

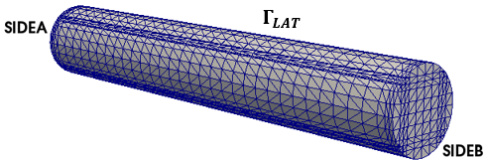


Figure 8. Computational domain for the simulation of the Na^+-K^+ pump. The finite element triangulation consists of 37075 tetrahedra.

Condition (12a) has the scope of introducing a reference value for the calculation of the electrostatic potential drop across the cellular membrane. Condition (12b) expresses the biological fact that the aforementioned potential drop is caused solely by the ion charge distribution within the pore because no external bias is applied.

Nernst–Planck equations. The Nernst–Planck equation system allows us to determine the spatial concentration of the various ion species inside the pore. The connection between the pore region and the intra- and extracellular sides is made possible by enforcing nonhomogeneous Neumann boundary conditions that preserve the correct input/output biophysical pump functionality. The boundary and initial conditions for each simulated ion species read as follows:

$$
\begin{aligned}
&K^+: && \underline{f}_{K^+} \cdot \underline{n} = g_{K^+} && \text{on sideA}, && (13a)\\
&&& n_{K^+} = K^+_{out} && \text{on sideB}, && (13b)\\
&&& \underline{f}_{K^+} \cdot \underline{n} = 0 && \text{on } \Gamma_{LAT}, && (13c)\\
&&& n_{K^+}(\underline{x}, 0) = K^+_0 && \text{for all } \underline{x} \in \Omega, && (13d)\\
&Na^+: && n_{Na^+} = Na^+_{in} && \text{on sideA}, && (13e)\\
&&& \underline{f}_{Na^+} \cdot \underline{n} = g_{Na^+} && \text{on sideB}, && (13f)\\
&&& \underline{f}_{Na^+} \cdot \underline{n} = 0 && \text{on } \Gamma_{LAT}, && (13g)\\
&&& n_{Na^+}(\underline{x}, 0) = Na^+_0 && \text{for all } \underline{x} \in \Omega, && (13h)\\
&Cl^-: && n_{Cl^-} = Cl^-_{in} && \text{on sideA}, && (13i)\\
&&& n_{Cl^-} = Cl^-_{out} && \text{on sideB}, && (13j)\\
&&& \underline{f}_{Cl^-} \cdot \underline{n} = 0 && \text{on } \Gamma_{LAT}, && (13k)\\
&&& n_{Cl^-}(\underline{x}, 0) = Cl^-_0 && \text{for all } \underline{x} \in \Omega, && (13l)\\
&HCO_3^-: && n_{HCO_3^-} = HCO^-_{3,in} && \text{on sideA}, && (13m)\\
&&& n_{HCO_3^-} = HCO^-_{3,out} && \text{on sideB}, && (13n)\\
&&& \underline{f}_{HCO_3^-} \cdot \underline{n} = 0 && \text{on } \Gamma_{LAT}, && (13o)\\
&&& n_{HCO_3^-}(\underline{x}, 0) = HCO^-_{3,0} && \text{for all } \underline{x} \in \Omega. && (13p)
\end{aligned}
$$

The values of the boundary data for the cations are specified in Table 12 whereas the values of the boundary data for the anions are specified in Table 13. The boundary values for the ions agree with the experimental values of the ion concentrations measured in the intracellular and extracellular sides of the nonpigmented epithelial cells [53].

Stokes system. The calculation of AH fluid velocity is made possible by solving the Stokes system (1e)–(1f). To prescribe a correct biophysical condition of the intrapore AH fluid we need to mathematically express that (1) the fluid is adherent to the pore wall, (2) no external pressure drop is applied across the pore, and (3) the AH fluid is at the rest when the pump is not active. To this purpose, the appropriate BCs and ICs read

$$\underline{u} = \underline{0} \quad \text{on } \Gamma_{\text{LAT}}, \tag{14a}$$

$$\underline{\underline{\sigma}}\,\underline{n} = \underline{0} \quad \text{on sideA} \cup \text{sideB}, \tag{14b}$$

$$\underline{u}(\underline{x}, 0) = \underline{0} \quad \text{for all } \underline{x} \in \Omega. \tag{14c}$$

Remark. In this work, we are not explicitly describing the active role of pumps in exchanging ions across the cell membrane. This would require, for example, including in the model the contribution of chemical processes such as ATP consumption. This contribution is implicitly taken into account in our model by means of the ion flux densities $\underline{f}$ that mathematically translate in the boundary conditions the result of these processes.

Simulation results. To ease the interpretation of the reported results, we point out that the Z axis coincides with the axial direction of the pore and it is positively oriented towards the extracellular region. Reported data for the vector-valued variables (such as electric field, current densities, and velocity) are the Z components of the vectors because the other two computed components were comparably negligible. We set $\rho_{\text{fixed}} = 0\,[\text{C m}^{-3}]$, $t_0 = 0\,[\text{s}]$, and $T_{\text{obs}} = 50\,[\text{ns}]$, a sufficiently large value to ensure that the simulated system has reached steady-state conditions at $t = T_{\text{obs}}$.

Remark. In this test case we set to zero the surface charge density in order to single out the influence of each force on model predictions.

The values of the dielectric permittivity of the intrapore water fluid ϵ_f, of the AH fluid shear viscosity μ_f, of the AH fluid mass density ρ_f, and of the diffusion coefficients D_i of each i-th ion species involved in the computational tests are reported in Table 1. All the figures in the remainder of the section illustrate computed results at $t = T_{\text{obs}}$.

model parameter	value	units
ϵ_f	$708.32 \cdot 10^{-10}$	$[\mathrm{F\,cm^{-1}}]$
μ_f	10^{-2}	$[\mathrm{g\,cm^{-1}\,s^{-1}}]$
ρ_f	1	$[\mathrm{g\,cm^{-3}}]$
D_{K^+}	$1.957 \cdot 10^{-5}$	$[\mathrm{cm^2\,s^{-1}}]$
D_{Na^+}	$1.334 \cdot 10^{-5}$	$[\mathrm{cm^2\,s^{-1}}]$
D_{Cl^-}	$2.033 \cdot 10^{-5}$	$[\mathrm{cm^2\,s^{-1}}]$
$D_{\mathrm{HCO}_3^-}$	$1.185 \cdot 10^{-5}$	$[\mathrm{cm^2\,s^{-1}}]$
$D_{\mathrm{Ca}^{++}}$	$7.92 \cdot 10^{-6}$	$[\mathrm{cm^2\,s^{-1}}]$
D_{H^+}	$9.315 \cdot 10^{-5}$	$[\mathrm{cm^2\,s^{-1}}]$

Table 1. Values of model parameters.

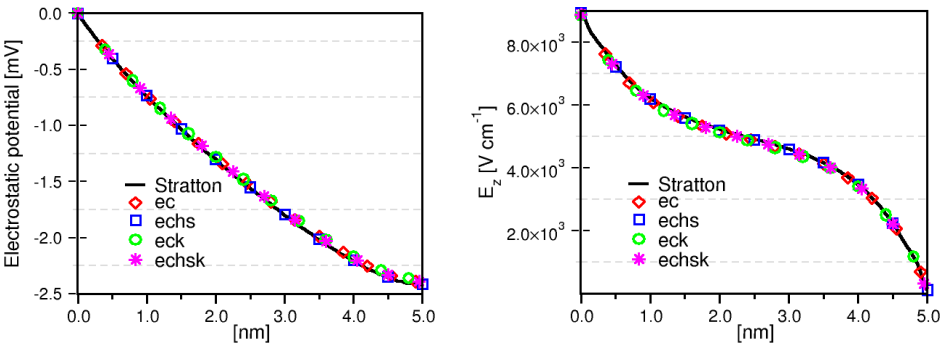


Figure 9. Electric variables along the axis of the pore. Left: electrostatic potential. Right: electric field.

Electric variables. Figure 9 shows the spatial distributions of the electric potential and of the electric field inside the pore. Since no permanent charge is included in the present simulation, the electric potential (and therefore also the electric field) is determined only by the Coulomb interaction among the ions in the pore. This is the reason why electric field and electric potential distributions are scarcely affected by the different models of Section 4. Figure 9, right, also shows that the electric field profile is monotonic inside the pore. This means, on the one hand, that ions are transported with a constant direction depending on their sign (from the intracellular to the extracellular sides in the case of cations, from the extracellular to the intracellular sides in the case of anions) and, on the other hand, that ions cannot be trapped inside the pore; rather they are helped travel throughout the pore. The electrostatic potential shown in Figure 9, left, allows us to perform a *first significant model comparison with experimental data* reported in Table 2 where the measured value of the transepithelial membrane potential $V_m := \varphi(0) - \varphi(L_{\mathrm{ch}})$ is reported for various animals. Results indicate that for all model choices of Section 4, the

V_m [mV]	animal	reference
3.80 ± 0.26	ox	[11]
5.53 ± 0.41	ox	[12]
3.83 ± 0.16	rabbit	[12]
-3.7 ± 0.3	toad	[57]
-1.2 ± 0.1	rabbit	[31]
-1.35 ± 0.08	dog	[22]
$\boxed{-2.5 \pm 0.2}$	monkey	[10]

Table 2. Experimental measurements for the transepithelial membrane potential. The boxed value is the measured data for monkeys and is considered as the reference for comparison with our model simulations.

simulated potential difference is in very good agreement with the data for monkeys [10] which can be considered as the animal species most similar to humans.

Remark. Simulations have been conducted using model parameters taken from human subjects [53]. However, no data are available for transepithelial membrane potential measured in humans, so results were compared to measurements for monkeys because of their physiological and structural similarity to humans.

Remark. The significant variability in the experimental measurements of V_m reported in Table 2 is the result of several factors, possibly depending on the different biophysical structure of the ciliary epithelium in the various animal species (for example, the difference in size and/or ion concentrations in the intra- and extracellular sites), and leads to a different electric equilibrium at steady-state. However, the order of magnitude of all the measured data is in the range of mV, which demonstrates the existence of a common mechanistic framework regulating the formation of the transepithelial membrane potential.

Chemical variables. Figure 10, top, shows the spatial distributions of cations inside the pore whereas Figure 10, bottom, shows the spatial distributions of anions inside the pore. Consistently with the simulated electric field and potential distributions, results indicate that the different models of the volume force density scarcely affect the ion concentrations indicating that the differences in AH fluid velocity, shown in the next section, are not strong enough to modify the ion profiles. As a second comment, we see that the spatial distribution of each ion concentration is not linear inside the pore because of the presence of the electric field that is responsible for the drift contribution in the ion flux constitutive relation (1b). The data reported in Table 3 allow us to perform a *second significant model comparison with experimental data*. The data include the computed value of the axial component of the potassium current density at the extracellular side of the pore $Z = L_{\text{ch}}$ and the computed value of the axial component of the sodium current density at the intracellular side of

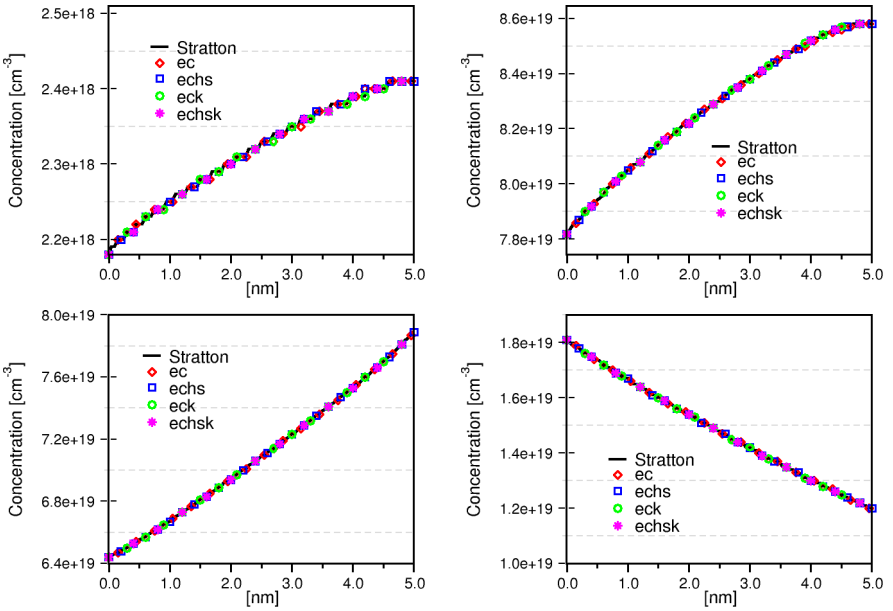


Figure 10. Ion concentration spatial distribution. Top left: K^+. Top right: Na^+. Bottom left: Cl^-. Bottom right: HCO_3^-.

the pore $Z = 0$ for the various choices for the model of the volume force density $\underline{F}_{\text{ion}}$ illustrated in Section 4. To allow a quantitative verification of the biophysical correctness of the predicted exchange of potassium and sodium ions across the pore, we introduce the parameter

$$r := \frac{g_{K^+}}{g_{Na^+}}, \tag{15}$$

where g_{K^+} denotes the boundary value of the flux density of potassium ions that enter into the cell and g_{Na^+} denotes the boundary value of the flux density of sodium ions that flow out of the cell. According to the data of Table 12, we have $r = 2 : 3$. The above parameter expresses the biophysical consistency of the boundary data adopted in the numerical simulation because it coincides with the theoretically expected stoichiometric ratio of the K^+ and Na^+ ions exchanged $(2 : 3)$ by the sodium-potassium pump as represented in the schematic picture of Figure 3. Because of the continuum approach employed in our model, we are going to check the correct functionality of the simulated pump by computing the parameter

$$\mathcal{R} := \left| \frac{J_{Z,K^+}}{J_{Z,Na^+}} \right|, \tag{16}$$

where J_{Z,K^+} is the potassium ion current density at the extracellular side of the pore and J_{Z,Na^+} is the sodium ion current density at the intracellular side of the pore.

model for $\underline{F}_{\text{ion}}$	$J_{Z,K^+}[\text{Acm}^{-2}]$	$J_{Z,Na^+}[\text{Acm}^{-2}]$	$\mathcal{R}$
Stratton	-0.064	0.098	0.653
ec	-0.054	0.44	0.123
echs	-0.046	0.74	0.062
eck	-0.064	0.085	0.753
echsk	-0.056	0.38	0.147

Table 3. Computed values of the axial component of the current density for sodium and potassium. The value J_{Z,K^+} is computed at $Z = L_{\text{ch}}$ whereas the value J_{Z,Na^+} is computed at $Z = 0$. We set $\mathcal{R} := |J_{Z,K^+}/J_{Z,Na^+}|$. The boxed values indicate the best model predictions to be compared with the theoretical expected ratio 2:3.

Remark. The parameter $\mathcal{R}$ is the counterpart of the quantity r defined in (15) and is quite sensitive to the choice of the volumetric force. The aim of our investigation is to quantify the impact of this choice (if any) on the ion current behavior, as we would like to capture the physiological function of the pumps/exchangers. The correct predicted ratio, moreover, does not only ensure the expected physiological effect, but also confirms the self-consistency of the model, which is not guaranteed a priori for every volume force density model.

As a first comment, the results of Table 3 show that for each considered model of $\underline{F}_{\text{ion}}$ the computed potassium current density is negative whereas the computed sodium current density is positive. This is consistent with the physiological function of the sodium-potassium pump because sodium ions flow out of the cell and potassium ions flow into the cell. As a second comment, the computed values of $\mathcal{R}$ indicate that agreement with the theoretical expected ratio $2:3$ is achieved only in the case of the Stratton model and of the eck model, whereas the values of $\mathcal{R}$ computed with the other models are not in a feasible range. This allows us to conclude that the VE-PNP model predicts a correct direction of ion flow for the sodium-potassium pump in a good quantitative agreement with the stoichiometric ratio of the pump only if the *Stratton* or the *eck* model is adopted to mathematically represent the volume force density in the linear momentum balance equation for the aqueous intracellular fluid.

Remark. This outcome is physically significant considering the fact that imposing on two different boundaries the ratio of the ion currents is no guarantee that such a ratio will be respected in the interior of the three-dimensional channel domain, whereas the value $\mathcal{R}$ is reached only at steady-state.

AH fluid variables. Figure 11 shows the spatial distributions of the component of the AH fluid velocity and of the volumetric force density along the Z axis. Predicted velocities are all negative except in the case of the eck model; similarly, the computed volumetric force densities are all positive except in the case of the

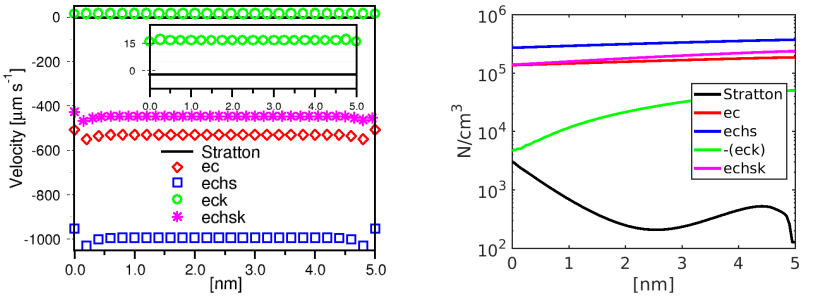


Figure 11. AH fluid variables along the axis of the pore. Left: aqueous humor fluid velocity. Right: volumetric force density.

model for $\underline{F}_{\text{ion}}$	$\bar{v}_Z[\mu\text{m s}^{-1}]$
Stratton	-2.34
ec	-529.5
echs	-992
eck	$\boxed{16.54}$
echsk	-445.6

Table 4. Computed mean values of the axial component of the intrapore AH fluid velocity. The boxed value indicates the sole model results that are in agreement with an outflux of aqueous humor from the cell into the extracellular side.

eck model (in Figure 11, right, we have reported the absolute value of the force density). Moreover, it is easy to check that the variation of the velocity is one-to-one correlated with the variation of the volumetric force density because of the homogeneous initial and boundary conditions that are applied to the Stokes system.

Table 4 reports the values of the AH fluid velocity computed at the center of the pore for each model considered in Section 4. Results allow us to perform a *third significant model comparison with experimental data*: only by describing the volume force density through the *eck* model is the VE-PNP formulation able to predict a positive AH fluid velocity which corresponds to the production of AH from the cell into the basolateral space. More specifically, if we assume a value of $2.5\,[\mu\text{l s}^{-1}]$ for a normal AH flow through the eye pupil [36] and an equivalent radius of 1 [mm] for the eye pupil of an adult, we see that a physiological value of v_Z is of about $14\,[\mu\text{m s}^{-1}]$, which agrees well with the value of $16.54\,[\mu\text{m s}^{-1}]$ predicted by the eck model. The other results from Table 4 (negative velocities) indicate that the magnitude of the predicted AH flow is nonphysically large, except in the case of the Stratton model, thus justifying its wide adoption in the literature.

Conclusions. The study of the interaction between the ion component (Na^+-K^+ pump) and AH production through a mathematical continuum approach based on

the VE-PNP model, under the condition of adopting the *eck* formulation of the volume force density that constitutes the source term in the fluid momentum balance equation, shows that simulation results are in agreement with

(1) the experimentally measured value of transepithelial membrane potential,
(2) the physiological stoichiometric rate of 2 : 3 that characterizes the sodium-potassium pump,
(3) the direction of current densities of sodium (flowing out of the cell) and potassium (flowing into the cell),
(4) the direction of AH flow (outward the cell), and
(5) the magnitude of AH fluid velocity.

The aforementioned results support the mathematical and biophysical motivation to adopt the *eck* model in the remainder of the article where we introduce a more realistic geometrical description of the ion pore and we include the main ion pumps and exchangers to describe the electric pressure exerted by the ions on the intrapore AH fluid.

7. Cellular scale simulation of ion pumps and exchangers in AH production

In this section we use the VE-PNP model to carry out an extensive quantitative investigation on the active role of the ion exchanges that are identified in [30; 29] as important determinants in AH secretion. To this purpose, we adopt the VE-PNP formulation in which the volumetric force exerted from ions onto the fluid is described by the eck model illustrated in Section 4. In addition, we employ in the numerical simulations a more realistic ion pore geometry than that shown in Figure 8, obtained by including in the computational domain a small amount of cell membrane as well as the presence of the antichambers.

Remark. We emphasize that only the Poisson equation (1c) is solved in the large rectilinear domain (represented in gray color in Figure 12). The dimensions of this domain are set in such a way that boundary effects have no influence on the solution computed in the pore domain.

The unified modeling and computational framework is here applied to study the function of each ion pump and exchanger illustrated in Section 2 with the goal of examining the output results, such as electrostatic potential, stoichiometric ratios, and AH velocity, as functions of the input parameters, such as the osmotic coefficient, the value of the permanent electric charge density, and the nonhomogeneous Neumann boundary conditions for ion flux densities.

Figure 12 shows the geometrical structure constituting the computational domain. The parallelepiped containing the cylinder represents the regions in which the

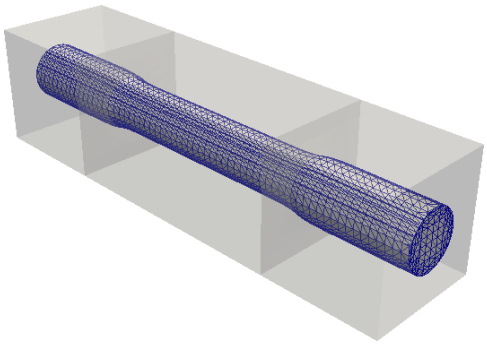


Figure 12. Computational domain for the simulation of the ion pumps and exchangers involved in AH production. The two external cylinders represent the pore antichambers whereas the central cylinder is the pore region. The partition of the pore domain into about 110391 tetrahedral elements is illustrated, whereas the mesh partition of the region surrounding the pore is not shown for visual clarity.

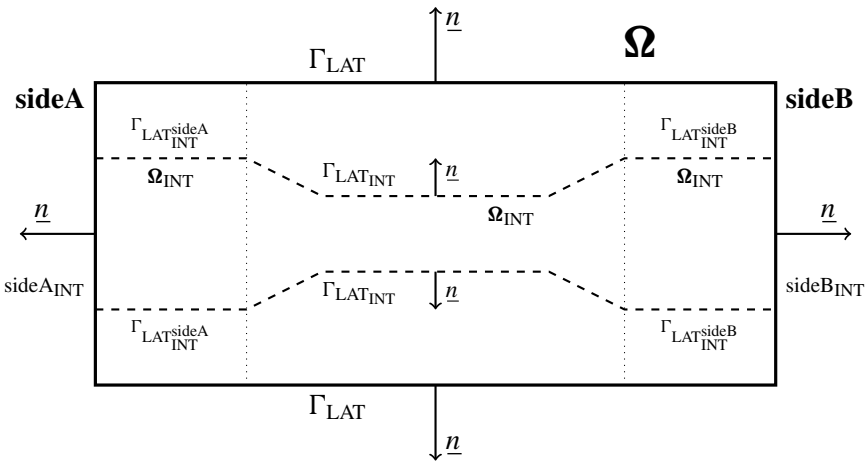


Figure 13. Two-dimensional cross-section of the pore geometry and boundary labels for the simulation of AH production.

transmembrane pore is divided, namely two external cylinders representing the pore antichambers and a central cylinder representing the ion pore. The parallelepiped is composed by the union of two cubes of side equal to 2.5 [nm] and by a central parallelepiped of length equal to 5 [nm] in such a way that the total length is equal to 10 [nm]. The portion of the cylindrical structure inside the two external cubes has a radius of 0.6 [nm] whereas the portion inside the central parallelepiped has a radius of 0.4 [nm]. The adopted geometrical representation is based on the biophysical setting analyzed in [9; 8] and aims at reproducing the morphology of a realistic protein membrane pore, where the two external cylinders play the role of pore

antichambers and the cylinder at the center plays the role of the pore region in which the main electrochemical and fluid processes take place. The full domain has been partitioned in tetrahedra as reported in Figure 12 where the discretization of the parallelepiped surrounding the cylinder is not shown for sake of visual clarity. Referring to the notation of Figure 13, in the remainder of the section, the VE-PNP equations (1a)–(1b) and the Stokes system (1e)–(1h) are solved only inside the cylinder Ω_{INT} whereas the Poisson equation (1c)–(1d) is solved in the whole domain Ω.

Boundary and initial conditions. In Section 6 we have highlighted the fundamental role played by the BCs in the simulation of the sodium-potassium pump. Because here we are treating a wider variety of ion exchangers and we have a more complex computational domain as well as the presence of a larger number of ion species, to help the clarity of the discussion we report in Figure 13 a two-dimensional cross-section of the pore geometry and identify the various regions of the domain with the corresponding labels for further reference.

Poisson equation. Because of the presence of the interface between the internal cylinder and the surrounding parallelepiped, we need to treat the jump of the electric displacement across that interface as well as the possible presence of electric charge on this interface. Let $\gamma := \Gamma_{\text{LAT}_{\text{INT}}^{\text{sideA}}} \cup \Gamma_{\text{LAT}_{\text{INT}}^{\text{sideB}}} \cup \Gamma_{\text{LAT}_{\text{INT}}}$ denote the two-dimensional surface separating the pore region from the surrounding membrane region as depicted in Figure 13. For a given vector-valued function $\underline{\tau} : \Omega \to \mathbb{R}^3$ we define the jump of $\underline{\tau}$ across the surface γ as

$$[\![\underline{\tau}]\!]_\gamma := (\underline{\tau}|_{\Omega\setminus\Omega_{\text{INT}}}|_\gamma - \underline{\tau}|_{\Omega_{\text{INT}}}|_\gamma) \cdot \underline{n},$$

whereas for a given scalar-valued function $\phi : \Omega \to \mathbb{R}$ we define the jump of ϕ across the surface γ as

$$[\![\phi]\!]_\gamma := \phi|_{\Omega\setminus\Omega_{\text{INT}}}|_\gamma \underline{n} - \phi|_{\Omega_{\text{INT}}}|_\gamma \underline{n}.$$

We notice that the jump of a vector-valued function is a scalar function whereas the jump of a scalar-valued function is a vector function. For all $t \in I_T$, the BCs for the Poisson equation (1c)–(1d) are

$$\varphi = 0 \quad \text{on sideA}_{\text{int}}, \tag{17a}$$

$$\underline{D} \cdot \underline{n} = 0 \quad \text{on } \Gamma_{\text{LAT}} \cup \text{sideB} \cup \text{sideA} \cup \text{sideB}_{\text{int}}, \tag{17b}$$

$$[\![\underline{D}]\!]_\gamma = h_\gamma \quad \text{on } \gamma, \tag{17c}$$

$$[\![\varphi]\!]_\gamma = 0 \quad \text{on } \gamma, \tag{17d}$$

where

$$h_\gamma = \begin{cases} \sigma_{\text{fixed}} & \text{on } \Gamma_{\text{LAT}_{\text{INT}}}, \\ 0 & \text{on } \Gamma_{\text{LAT}_{\text{INT}}^{\text{sideA}}} \cup \Gamma_{\text{LAT}_{\text{INT}}^{\text{sideB}}}, \end{cases}$$

pump/exchanger	σ_{fixed} [C cm^{-2}]
sodium-potassium pump	$-1 \cdot 10^{10}$
calcium-sodium exchanger	$-1.2 \cdot 10^{12}$
chloride-bicarbonate exchanger	$+3.9 \cdot 10^{11}$
sodium-proton exchanger	$-2.65 \cdot 10^{12}$

Table 5. Values of the fixed charge density σ_{fixed}.

$\text{Na}^+ = \text{Na}^+_{\text{in}}$	$\underline{f}_{\text{K}^+} \cdot \underline{n} = g_{\text{K}^+}$	on sideA_{int}
$\text{K}^+ = \text{K}^+_{\text{out}}$	$\underline{f}_{\text{Na}^+} \cdot \underline{n} = g_{\text{Na}^+}$	on sideB_{int}
$\underline{f}_{\text{Na}^+} \cdot \underline{n} = 0$	$\underline{f}_{\text{K}^+} \cdot \underline{n} = 0$	on γ
$\text{K}^{+^0}(x) = \text{K}^+_0$	$\text{Na}^{+^0}(x) = \text{Na}^+_0$	in Ω_{INT}
$\text{Cl}^- = \text{Cl}^-_{\text{in}}$	$\text{HCO}_3^- = \text{HCO}^-_{3_{\text{in}}}$	on sideA_{int}
$\text{Cl}^- = \text{Cl}^-_{\text{out}}$	$\text{HCO}_3^- = \text{HCO}^-_{3_{\text{out}}}$	on sideB_{int}
$\underline{f}_{\text{Cl}^-} \cdot \underline{n} = 0$	$\underline{f}_{\text{HCO}_3^-} \cdot \underline{n} = 0$	on γ
$\text{Cl}^{-^0}(x) = \text{Cl}^-_0$	$\text{HCO}_3^{-^0}(x) = \text{HCO}^-_{3_0}$	in Ω_{INT}

Table 6. BCs and ICs for the sodium-potassium pump.

σ_{fixed} [C m^{-2}] being a given distribution of superficial permanent charge density that mathematically represents the electric charge contained in the amino-acid structure of the protein surrounding the ion pore. We notice that the interface condition (17d) expresses the physical fact that the electric potential is a continuous function across γ, whereas the interface condition (17c) expresses the physical fact that the normal component of the displacement vector is discontinuous across the surface separating the pore region and the lipid membrane bilayer because of the presence of amino-acid fixed charge density σ_{fixed}. The value of σ_{fixed} needs be determined in order to reproduce the correct functionality of the several ion pumps/exchangers. To this purpose, a simulation campaign has to be performed to heuristically tune-up the values of σ_{fixed} (see [45] for the sodium-potassium pump). The results of this procedure in the present context are reported in Table 5.

Nernst–Planck equations. The several ion pumps/exchangers involved in AH production are simulated by considering the contribution of different ions in order to produce the correct electrostatic potential drop across the cell membrane. The list of these ions is reported below. In complete analogy with what was done in Section 6 for the BCs and ICs of the Nernst–Planck equations (1a)–(1b), we report in Tables 6–9 the BCs and ICs adopted to reproduce the correct biophysical functionality of each ion pump/exchanger:

$K^+ = K^+_{in}$	$Na^+ = Na^+_{in}$	$\underline{f}_{Ca^{++}} \cdot \underline{n} = g_{Ca^{++}}$	on $sideA_{int}$
$K^+ = K^+_{out}$	$Ca^{++} = Ca^{++}_{out}$	$\underline{f}_{Na^+} \cdot \underline{n} = g_{Na^+}$	on $sideB_{int}$
$\underline{f}_{Ca^{++}} \cdot \underline{n} = 0$	$\underline{f}_{Na^+} \cdot \underline{n} = 0$	$\underline{f}_{K^+} \cdot \underline{n} = 0$	on γ
$K^{+^0}(x) = K^+_0$	$Na^{+^0}(x) = Na^+_0$	$Ca^{++^0}(x) = Ca^{++}_0$	in Ω_{INT}
$Cl^- = Cl^-_{in}$	$HCO_3^- = HCO_{3_{in}}^-$		on $sideA_{int}$
$Cl^- = Cl^-_{out}$	$HCO_3^- = HCO_{3_{out}}^-$		on $sideB_{int}$
$\underline{f}_{Cl^-} \cdot \underline{n} = 0$	$\underline{f}_{HCO_3^-} \cdot \underline{n} = 0$		on γ
$Cl^{-^0}(x) = Cl^-_0$	$HCO_3^{-^0}(x) = HCO_{3_0}^-$		in Ω_{INT}

Table 7. BCs and ICs for the calcium-sodium exchanger.

$K^+ = K^+_{in}$	$Na^+ = Na^+_{in}$	on $sideA_{int}$
$K^+ = K^+_{out}$	$Na^+ = Na^+_{out}$	on $sideB_{int}$
$\underline{f}_{Na^+} \cdot \underline{n} = 0$	$\underline{f}_{K^+} \cdot \underline{n} = 0$	on γ
$K^{+^0}(x) = K^+_0$	$Na^{+^0}(x) = Na^+_0$	in Ω_{INT}
$\underline{f}_{Cl^-} \cdot \underline{n} = g_{Cl^-}$	$\underline{f}_{HCO_3^-} \cdot \underline{n} = g_{HCO_3^-}$	on $sideA_{int}$
$Cl^- = Cl^-_{out}$	$HCO_3^- = HCO_{3_{out}}^-$	on $sideB_{int}$
$\underline{f}_{Cl^-} \cdot \underline{n} = 0$	$\underline{f}_{HCO_3^-} \cdot \underline{n} = 0$	on γ
$Cl^{-^0}(x) = Cl^-_0$	$HCO_3^{-^0}(x) = HCO_{3_0}^-$	in Ω_{INT}

Table 8. BCs and ICs for the chloride-bicarbonate exchanger.

Na^+-K^+ pump. Na^+, K^+, Cl^-, and HCO_3^- are included.

Ca^{++}-Na^+ exchanger. Na^+, K^+, Cl^-, HCO_3^-, and Ca^{++} are included.

Cl^--HCO_3^- exchanger. Na^+, K^+, Cl^-, and HCO_3^- are included.

Na^+-H^+ exchanger. Na^+, K^+, Cl^-, HCO_3^-, and H^+ are included.

The values of the boundary data for the ion pumps and exchangers are specified in Tables 14–17.

Stokes system. We adopt the same BCs and ICs as in Section 6:

$$\underline{u} = \underline{0} \quad \text{on } \gamma, \tag{18a}$$

$$\underline{\underline{\sigma}}\,\underline{n} = \underline{0} \quad \text{on } sideA_{int} \cup sideB_{int}, \tag{18b}$$

$$\underline{u}(\underline{x}, 0) = \underline{0} \quad \text{for all } \underline{x} \in \Omega_{INT}. \tag{18c}$$

As already pointed out, to describe the volumetric force on the right-hand side of the linear momentum balance equation in the Stokes system, we use the *eck*

$K^+ = K^+_{in}$	$Na^+ = Na^+_{in}$	$\underline{f}_{H^+} \cdot \underline{n} = g_{H^+}$	on sideA_{int}
$K^+ = K^+_{out}$	$H^+ = H^+_{out}$	$\underline{f}_{Na^+} \cdot \underline{n} = g_{Na^+}$	on sideB_{int}
$\underline{f}_{H^+} \cdot \underline{n} = 0$	$\underline{f}_{Na^+} \cdot \underline{n} = 0$	$\underline{f}_{K^+} \cdot \underline{n} = 0$	on γ
$K^{+^0}(x) = K^+_0$	$Na^{+^0}(x) = Na^+_0$	$H^{+^0}(x) = H^+_0$	in Ω_{INT}
$Cl^- = Cl^-_{in}$	$HCO_3^- = HCO_{3_{in}}^-$		on sideA_{int}
$Cl^- = Cl^-_{out}$	$HCO_3^- = HCO_{3_{out}}^-$		on sideB_{int}
$\underline{f}_{Cl^-} \cdot \underline{n} = 0$	$\underline{f}_{HCO_3^-} \cdot \underline{n} = 0$		on γ
$Cl^{-^0}(x) = Cl^-_0$	$HCO_3^{-^0}(x) = HCO_{3_0}^-$		in Ω_{INT}

Table 9. BCs and ICs for the sodium-proton exchanger.

Pump/exchanger	k [N m]
sodium-potassium pump	$4.1 \cdot 10^{-19}$
calcium-sodium exchanger	$24 \cdot 10^{-19}$
chloride-bicarbonate exchanger	$4.1 \cdot 10^{-19}$
sodium-proton exchanger	$4 \cdot 10^{-19}$

Table 10. Values of the electrochemical osmotic parameter k for each pump/exchanger involved in the process of AH production.

model. The value of k is considered a characteristic property of the single pump and exchanger, and it is reported in Table 10.

Simulation results. Reported data for the vector-valued variables (such as electric field, current densities, and AH fluid velocity) are the Z component of the vectors because the other two computed components were comparably negligible. We set $t_0 = 0$ [s] and $T_{obs} = 50$ [ns], a sufficiently large value to ensure that the simulated system has reached steady-state conditions at $t = T_{obs}$: all the figures in the remainder of the section illustrate computed results at this time. The values of the dielectric permittivity of the intrapore fluid ϵ_f, of the AH fluid shear viscosity μ_f, of the AH fluid mass density ρ_f, and of the diffusion coefficients D_i of each i-th ion species involved in the computational tests are reported in Table 1.

Electric variables. Figure 14, left, shows the transepithelial electrostatic potential as calculated by the simulations. We note how the electric potential is strongly influenced by the presence of the fixed surface charge density σ_{fixed} in the central region of the domain (see Table 5), with particular emphasis on the case of the sodium-proton exchanger. It is remarkable to notice that, as in the case of the simulation of the sodium-potassium pump illustrated in Section 6, also in this more complex biophysical setting, for each simulated exchanger, the computed

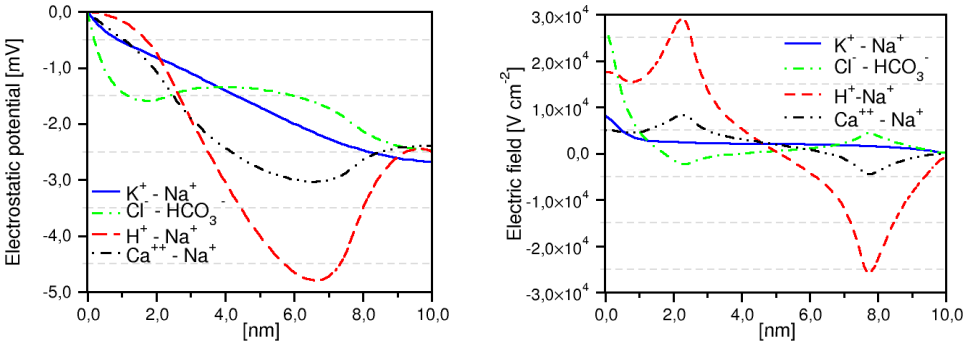


Figure 14. Electric variables along the axis of the pore. Left: electrostatic potential. Right: electric field.

value of *the transepithelial membrane potential is in very good agreement with the experimental data* for monkeys reported in Table 2.

Figure 14, right, shows the computed spatial behavior of the axial component E_Z of the electric field for each considered pump/exchanger. Consistently with electrostatic potential, we see that in all simulations E_Z is a monotonic function of position in the central region of the pore. Then, coming closer to the outlet section at $Z = L_{\text{ch}}$, all the simulated profiles become flat in accordance with the homogeneous Neumann boundary condition (17b). Specifically, in the case of cation pump/exchanger, the electric field is decreasing along the central part of the pore whereas in the case of the chloride-bicarbonate exchanger the electric field is increasing. These two opposite behaviors are related to the presence of surface charge on $\Gamma_{\text{LAT}_{\text{INT}}}$ of opposite sign (negative for cation pumps/exchangers, positive for the anion exchanger). In the case of the sodium-proton exchanger, the electric field profile experiences a large increase in magnitude moving along the pore axis from the intracellular side towards the extracellular side because of the elevated negative fixed charge density distributed on the lateral surface on the pore region (see Table 5).

Chemical variables. The computed profiles of the ion concentrations for each simulated ion pump and exchanger are reported in Figure 15. Results show the onset of a concentration gradient for each simulated ion species which appears not to be spatially constant for all ion species because of the action of the electric drift force which displaces the ion profile from the linear equilibrium distribution corresponding to a null electric field. Particularly worth noticing is the occurrence of significant variations for the concentration of the sodium ion in the simulation of the sodium-proton exchanger shown in Figure 15, bottom right. These variations are the result of the attractive electrostatic force exerted on the sodium ions by the elevated negative fixed charge density distributed on the lateral surface on the pore region (see Table 5).

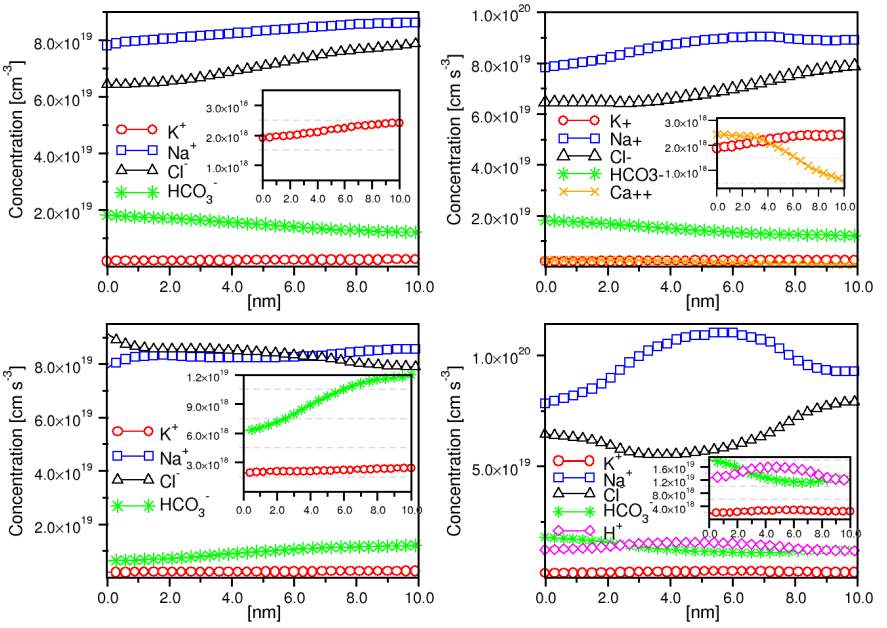


Figure 15. Computed ion concentration along the Z axis. Top left: sodium-potassium pump. Top right: calcium-sodium exchanger. Bottom left: chloride-bicarbonate exchanger. Bottom right: sodium-proton exchanger.

Figure 16 shows the spatial distributions of the computed axial component of the ion current density for each pump and exchanger. For sake of clarity, in these figures we report only the current density related to the pump/exchanger functionality. We notice that for each ion, the value of the current density along the Z axis is not constant because the cross-section varies along the pore axis.

First, the sign of the computed current density for each simulated pump/exchanger agrees with the theoretically expected direction. As a second comment, *the predicted value of the stoichiometric ratio $\mathcal{R}$ reasonably agrees with the corresponding theoretical value r*. Specifically, in the case of the sodium-potassium pump shown in Figure 16, top left, $\mathcal{R} = 0.83$ whereas $r = 2:3 = 0.67$, in the case of the calcium-sodium exchanger shown in Figure 16, top right, $\mathcal{R} = 0.43$ whereas $r = 1:3 = 0.33$, in the case of the chloride-bicarbonate exchanger shown in Figure 16, bottom left, we have $\mathcal{R} = 1.23$ to be compared with $r = 1:1$, and in the case of the sodium-proton exchanger shown in Figure 16, bottom right, the value $\mathcal{R} = 0.92$ agrees fairly well with the theoretical value $r = 1:1$.

AH fluid variables. Figure 17 shows the spatial distribution of the axial component of the intrapore AH fluid velocity predicted by the simulation with the VE-PNP model of each ion pump and exchanger involved in the process of AH production. Results exhibit a significant difference among the various pumps/exchangers mainly

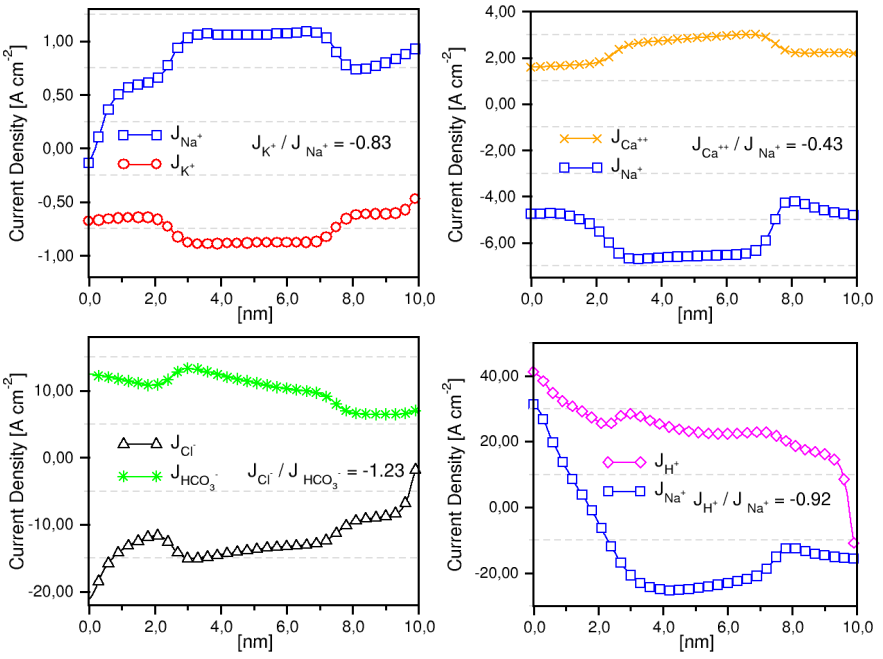


Figure 16. Computed ion current density along the Z axis. Top left: sodium-potassium pump. Top right: calcium-sodium exchanger. Bottom left: chloride-bicarbonate exchanger. Bottom right: sodium-proton exchanger.

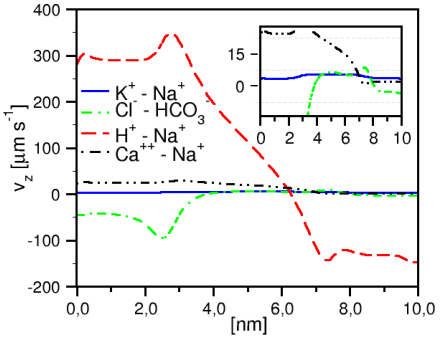


Figure 17. Computed spatial distribution of the axial component of AH fluid velocity for each pump/exchanger involved in AH production.

due to the presence of the fixed surface charge density σ_{fixed} in the central region of the domain that needed to be included to reproduce the correct pump functionalities. Specifically, in the case of the cation-based ion pumps/exchangers, model simulation predicts a positive value of the AH velocity in the whole computational domain whereas in the case of the chloride-bicarbonate exchanger the computed AH fluid velocity is strictly positive only in the central region of the domain. In the case of

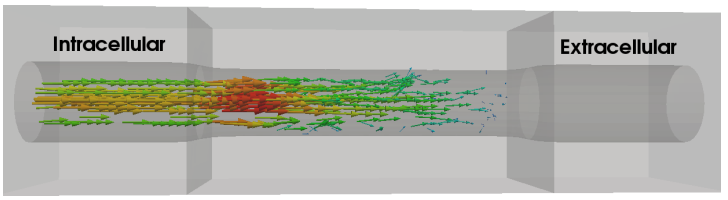


Figure 18. Computed three-dimensional spatial distribution of AH fluid velocity in the case of the calcium-sodium exchanger.

the sodium-proton exchanger, the elevated fixed surface charge density on $\Gamma_{\mathrm{LAT}_{\mathrm{INT}}}$ gives rise to a change of sign of the axial component of the electric field at about the center of the domain. This, in turn, gives rise to a change of sign in the volume force density $\underline{F}_{\mathrm{ion}}$ in the momentum balance equation of the AH fluid, causing an inversion of the intrapore AH fluid flow at $Z = 7.5\,[\mathrm{nm}]$ where the direction of the axial velocity changes sign, from positive to negative. These results seem to suggest that the main ion pumps/exchangers contributing to AH secretion are those that actively involve Na^+.

Figure 18 shows an example of three-dimensional computed spatial distribution of the AH fluid velocity in the case of the calcium-sodium exchanger. Results clearly show that AH flows from the intracellular space towards the extracellular space.

Further remarks on pump/exchanger functionality. In this section we briefly address a series of further considerations on the analysis of the simulation of the various ion pumps and exchangers involved in the process of AH production, in particular, those related to pump/exchanger functionality. The first consideration concerns the temporal evolution of calcium in the Ca^{++}-Na^+ exchanger. To this purpose, Figure 19, top left, illustrates the spatial calcium concentration at $t = 0$ (dashed line) and that at $t = T_{\mathrm{obs}}$ (solid line). Results show a decrease of the level of calcium in the intracellular side of the domain.

Remark. This result is interesting from a physiological viewpoint due to the fact that the intracellular initial concentration of Ca^{++} is imposed at $3.011 \cdot 10^{18}\,\mathrm{cm}^{-3} = 5\,\mathrm{mM}$, which corresponds to a pathological condition of calcium excess within the cell [2; 14]. Such a decrease is slow because of the relatively low value $D_{Ca^{++}} = 7.92 \cdot 10^{-6}\,[\mathrm{cm}^2\,\mathrm{s}^{-1}]$ of the diffusion coefficient adopted in the numerical simulation, but nonetheless, it is compatible with a theoretical expectation of a value of $10^{-4}\,\mathrm{mM}$ for healthy intracellular calcium level [2; 14].

The second consideration concerns the time behavior of carbonate (HCO_3^-) in the chloride-bicarbonate exchanger. To this purpose, Figure 19, top right, illustrates the spatial bicarbonate concentration at $t = 0$ (dashed line) and that at $t = T_{\mathrm{obs}}$

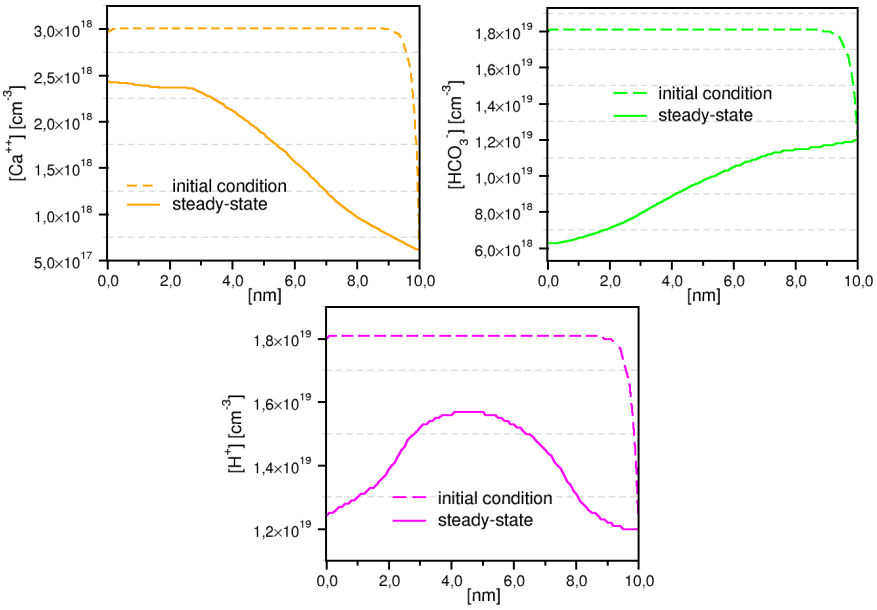


Figure 19. Computed ion current density along the Z axis at various times. Dashed line: distribution at $t = 0$. Solid line: distribution at $t = T_{\text{obs}}$. Top left: $[Ca^{++}]$ in the calcium-sodium exchanger. Top right: $[HCO_3^-]$ for the chloride-bicarbonate exchanger. Bottom: $[H^+]$ for the sodium-proton exchanger.

(solid line). Results show a significant decrease of the level of bicarbonate in the intracellular region. This behavior agrees with the fact that the simulated electric field profile for the chloride-bicarbonate exchanger is positive in the intracellular region of the domain (see Figure 14, right) and therefore bicarbonate ions are swept away from left to right. For further analysis of the importance of the bicarbonate ion in AH production, we refer to [35].

The third consideration concerns the spatial distribution of the proton (H^+) in the simulation of the sodium-proton exchanger. To this purpose, Figure 19, bottom, illustrates the spatial proton profile at $t = 0$ (dashed line) and that at $t = T_{\text{obs}}$ (solid line). Similarly to the previous case of carbonate, results show a significant reduction with simulation time of proton concentration in all the domain. However, unlike the previous case, concentration reduction equally occurs in both intracellular and extracellular sides whereas proton accumulation occurs in the pore region. This behavior is due to the direction of the electric field (see Figure 14, right) which pushes H^+ from left to right in the intracellular side (where $E_Z > 0$) and from right to left in the extracellular side (where $E_Z < 0$).

Summary of the simulation results. In Table 11 we report the main outcomes of the simulation of ion pumps and exchangers carried out in the context of AH

Pump/exchanger	k [N cm]	σ_{fixed} [C cm^{-2}]	φ_m [mV]	$\mathcal{R}$	v_Z [μm s^{-1}]
K^+-Na^+	$4.1 \cdot 10^{-19}$	$-1.0 \cdot 10^{10}$	-2.67	0.83 (0.67)	[4, 5.56]
Cl^--HCO_3^-	$24 \cdot 10^{-19}$	$+3.9 \cdot 10^{11}$	-2.46	1.23 (1)	[-100, $+6.56$]
Na^+-H^+	$4.0 \cdot 10^{-19}$	$-2.65 \cdot 10^{12}$	-2.49	0.92 (1)	[-150, 300]
Ca^{++}-Na^+	$3.95 \cdot 10^{-19}$	$-6 \cdot 10^{11}$	-2.39	0.43 (0.33)	[2, 28]

Table 11. A summary of the simulation results for ion pumps and exchangers involved in AH production. Aside the predicted value of $\mathcal{R}$ we report in parentheses the theoretically expected value. The column v_Z reports for each row the predicted range of the AH fluid velocity.

secretion induced at the cellular scale level by the effect of ion pressure exerted on transmembrane fluid. To summarize, *a unified modeling and computational framework allowed us to successfully simulate the functionality of several ion pump/exchangers* while preserving at the same time the features of each single pump/exchanger, by a proper selection of the ion flux density BCs, of the osmotic gradient coefficient, and of the amount of amino-acid charge in the pore protein folder. These conclusions are significant outcomes of our computational model because osmotic gradient coefficient and permanent electric surface charge do not yet have a quantitative comparison with experimental data, though they have been shown to be essential parts of the biophysical description of the pore and to play a relevant role in determining AH flow direction.

8. Conclusions, model limitations, and future objectives

A unified modeling and computational framework with electrochemical osmotic correction and with a realistic geometry to represent the computational domain has been proposed to investigate the main functional principles of the sodium-potassium pump and the calcium-sodium, chloride-bicarbonate, and sodium-proton exchangers that are involved in the production of aqueous humor in the ciliary body of the eye.

The theoretical model has been demonstrated to correctly reproduce, for each simulated ion pump and exchanger, existing experimental data of transepithelial membrane potential in animal models. The model has also allowed, for the first time to the best of our knowledge in the study of AH production, the quantitative analysis of novel biophysical mechanisms such as the physiological stoichiometric rate, the direction of AH flow, and the magnitude of AH fluid velocity. Thus, the present study motivates the further development of this modeling approach to (i) simulate the simultaneous presence and action of the several ion pump/exchangers considered in this work, with the aim of quantitatively estimating their reciprocal influence, (ii) include the presence of other molecules actively transported through the cell membrane, including ascorbic acid, which is secreted by a transporter

(sodium-dependent vitamin C transporter 2 (SVCT2)) [55], and (iii) the simulation of the effect of administration of a drug in the regulation of AH secretion.

It is expected that such theoretical advancement of the frontier of knowledge in this branch of human sciences may significantly help design new molecules for drug synthesis and, as a consequence, considerably reduce time and costs for clinical availability of new pharmacological therapies.

It is important to emphasize, though, that a number of biophysical limitations still affect the proposed mathematical model of aqueous humor dynamics. Among them, we mention that our model does not account for (i) autonomic system pathways, specifically the sympathetic and parasympathetic pathways [54], (ii) variations due to the circadian rhythm [50], which would require a higher-order temporal discretization in order to account for the temporal transients, or (iii) the role of aquaporins in the exchange of fluid across the cell membrane [46]. A research effort to address these limitations is currently in progress.

Acknowledgements

PhD candidate Sala is supported by a scholarship of the Ministère de l'Enseignement supérieur et de la Recherche (France). Doctor Sacco has been partially supported by Micron Semiconductor Italia S.r.l., SOW number 4505462139. Doctor Guidoboni has been partially supported by the awards NSF DMS-1224195 and NSF DMS-1853222, the Chair Gutenberg funds of the Cercle Gutenberg (France), and the LabEx IRMIA (University of Strasbourg, France). Doctor Harris has been partially supported by Research to Prevent Blindness (New York) and the award NSF DMS-1853303.

Data tables

K_0^+ $2.41 \cdot 10^{18}$ [cm^{-3}]	g_{K^+} $+4 \cdot 10^{17}$ [cm^{-2}s^{-1}]	K_{out}^+ $2.41 \cdot 10^{18}$ [cm^{-3}]
Na_0^+ $8.19 \cdot 10^{19}$ [cm^{-3}]	Na_{in}^+ $7.82 \cdot 10^{19}$ [cm^{-3}]	g_{Na^+} $+6 \cdot 10^{17}$ [cm^{-2}s^{-1}]

Table 12. Boundary and initial data for the cations in Section 6.

Cl_0^- $7.17 \cdot 10^{19}$ [cm^{-3}]	Cl_{in}^- $6.44 \cdot 10^{19}$ [cm^{-3}]	Cl_{out}^- $7.89 \cdot 10^{19}$ [cm^{-3}]
$HCO_{3,0}^-$ $1.51 \cdot 10^{19}$ [cm^{-3}]	$HCO_{3,in}^-$ $1.81 \cdot 10^{19}$ [cm^{-3}]	$HCO_{3,out}^-$ $1.2 \cdot 10^{19}$ [cm^{-3}]

Table 13. Boundary and initial data for the anions in Section 6.

K_0^+ [cm^{-3}]	g_{K^+} [cm^{-2}s^{-1}]	K_{out}^+ [cm^{-3}]
$2.41 \cdot 10^{18}$	$+4 \cdot 10^{19}$	$2.41 \cdot 10^{18}$
Na_0^+ [cm^{-3}]	Na_{in}^+ [cm^{-3}]	g_{Na^+} [cm^{-2}s^{-1}]
$8.19 \cdot 10^{19}$	$7.82 \cdot 10^{19}$	$+6 \cdot 10^{19}$
Cl_0^- [cm^{-3}]	Cl_{in}^- [cm^{-3}]	Cl_{out}^- [cm^{-3}]
$7.17 \cdot 10^{19}$	$6.44 \cdot 10^{19}$	$7.89 \cdot 10^{19}$
$HCO_{3\,0}^-$ [cm^{-3}]	$HCO_{3\,in}^-$ [cm^{-3}]	$HCO_{3\,out}^-$ [cm^{-3}]
$1.51 \cdot 10^{19}$	$1.81 \cdot 10^{19}$	$1.20 \cdot 10^{19}$

Table 14. Data for the sodium-potassium pump in Section 7.

K_0^+ [cm^{-3}]	K_{in}^+ [cm^{-3}]	K_{out}^+ [cm^{-3}]
$2.41 \cdot 10^{18}$	$1.90 \cdot 10^{18}$	$2.41 \cdot 10^{18}$
Na_0^+ [cm^{-3}]	Na_{in}^+ [cm^{-3}]	g_{Na^+} [cm^{-2}s^{-1}]
$8.19 \cdot 10^{19}$	$7.82 \cdot 10^{19}$	$-6 \cdot 10^{19}$
Ca_0^{++} [cm^{-3}]	$g_{Ca^{++}}$ [cm^{-2}s^{-1}]	Ca_{out}^{++} [cm^{-3}]
$3.011 \cdot 10^{18}$	$-2 \cdot 10^{19}$	$6.022 \cdot 10^{17}$
Cl_0^- [cm^{-3}]	Cl_{in}^- [cm^{-3}]	Cl_{out}^- [cm^{-3}]
$7.17 \cdot 10^{19}$	$6.44 \cdot 10^{19}$	$7.89 \cdot 10^{19}$
$HCO_{3\,0}^-$ [cm^{-3}]	$HCO_{3\,in}^-$ [cm^{-3}]	$HCO_{3\,out}^-$ [cm^{-3}]
$1.51 \cdot 10^{19}$	$1.81 \cdot 10^{19}$	$1.2 \cdot 10^{19}$

Table 15. Data for the calcium-sodium exchanger in Section 7.

K_0^+ [cm^{-3}]	$K^+{}_{in}$ [cm^{-3}]	$K^+{}_{out}$ [cm^{-3}]
$2.41 \cdot 10^{18}$	$1.90 \cdot 10^{18}$	$2.41 \cdot 10^{18}$
Na_0^+ [cm^{-3}]	Na_{in}^+ [cm^{-3}]	$Na^+{}_{out}$ [cm^{-3}]
$8.19 \cdot 10^{19}$	$7.82 \cdot 10^{19}$	$8.55 \cdot 10^{19}$
$Cl^-{}_0$ [cm^{-3}]	g_{Cl^-} [cm^{-2}s^{-1}]	$Cl^-{}_{out}$ [cm^{-3}]
$7.17 \cdot 10^{19}$	$+8 \cdot 10^{19}$	$7.89 \cdot 10^{19}$
$HCO_{3\,0}^-$ [cm^{-3}]	$g_{HCO_3^-}$ [cm^{-2}s^{-1}]	$HCO_{3\,out}^-$ [cm^{-3}]
$1.81 \cdot 10^{19}$	$-8 \cdot 10^{19}$	$1.20 \cdot 10^{19}$

Table 16. Data for the chloride-bicarbonate exchanger in Section 7.

K_0^+ [cm^{-3}] $2.41 \cdot 10^{18}$	K_{in}^+ [cm^{-3}] $1.90 \cdot 10^{18}$	K_{out}^+ [cm^{-3}] $2.41 \cdot 10^{18}$
Na_0^+ [cm^{-3}] $8.19 \cdot 10^{19}$	Na_{in}^+ [cm^{-3}] $7.82 \cdot 10^{19}$	g_{Na^+} [cm^{-2}s^{-1}] $-1.0 \cdot 10^{20}$
H_0^+ [cm^{-3}] $1.81 \cdot 10^{19}$	g_{H^+} [cm^{-2}s^{-1}] $-1.0 \cdot 10^{20}$	H_{out}^+ [cm^{-3}] $1.20 \cdot 10^{19}$
Cl_0^- [cm^{-3}] $7.17 \cdot 10^{19}$	Cl_{in}^- [cm^{-3}] $6.44 \cdot 10^{19}$	Cl_{out}^- [cm^{-3}] $7.89 \cdot 10^{19}$
$HCO_{3\,0}^-$ [cm^{-3}] $1.81 \cdot 10^{19}$	$HCO_{3\,in}^-$ [cm^{-3}] $1.81 \cdot 10^{19}$	$HCO_{3\,out}^-$ [cm^{-3}] $1.20 \cdot 10^{19}$

Table 17. Data for the sodium-proton exchanger in Section 7.

References

[1] P. Airoldi, A. G. Mauri, R. Sacco, and J. W. Jerome, *Three-dimensional numerical simulation of ion nanochannels*, J. Coupl. Sys. Multiscale Dyn. **3** (2015), no. 1, 57–65.

[2] B. Alberts, A. Johnson, J. Lewis, M. Raff, K. Roberts, and P. Walter, *Molecular biology of the cell*, 5th ed., Garland Science, New York, 2007.

[3] M. Z. Bazant, *Nonlinear electrokinetic phenomena*, Encyclopedia of microfluidics and nanofluidics (D. Li, ed.), Springer, 2008, pp. 1461–1470.

[4] _____, *Electrokinetics meets electrohydrodynamics*, J. Fluid Mech. **782** (2015), 1–4. MR Zbl

[5] Blausen Medical, *Medical gallery of Blausen Medical 2014*, WikiJ. Medicine **1** (2014), no. 2, art. id. 10.

[6] C. Breitkopf and K. Swider-Lyons (eds.), *Springer Handbook of electrochemical energy*, Springer, 2017.

[7] R. F. Brubaker, *Measurement of aqueous flow by fluorophotometry*, The glaucomas (R. Ritch, M. B. Shields, and T. Krupin, eds.), Mosby, St. Louis, 1989, pp. 337–344.

[8] D. Chen and R. Eisenberg, *Charges, currents, and potentials in ionic channels of one conformation*, Biophys. J. **64** (1993), no. 5, 1405–1421.

[9] _____, *Flux, coupling, and selectivity in ionic channels of one conformation*, Biophys. J. **65** (1993), no. 2, 727–746.

[10] T. C. Chu, O. A. Candia, and S. M. Podos, *Electrical parameters of the isolated monkey ciliary epithelium and effects of pharmacological agents*, Invest. Ophth. Vis. Sci. **28** (1987), no. 10, 1644–1648.

[11] D. F. Cole, *Electrical potential across the isolated ciliary body observed in vitro*, Brit. J. Ophthalmol. **45** (1961), 641–653.

[12] _____, *Transport across the isolated ciliary body of ox and rabbit*, Brit. J. Ophthalmol. **46** (1962), 577–591.

[13] B. Eisenberg, Y. Hyon, and C. Liu, *Energy variational analysis of ions in water and channels: field theory for primitive models of complex ionic fluids*, J. Chem. Phys. **133** (2010), no. 10, art. id. 104104.

[14] G. B. Ermentrout and D. H. Terman, *Mathematical foundations of neuroscience*, Interdisciplinary Applied Mathematics, no. 35, Springer, 2010. MR Zbl

[15] B. T. Gabelt and P. L. Kaufman, *Aqueous humor hydrodynamics*, Adler's physiology of the eye (P. L. Kaufman and A. Alm, eds.), vol. 8, Mosby, St. Louis, 1995, pp. 237–289.

[16] M. Goel, R. G. Picciani, R. K. Lee, and S. K. Bhattacharya, *Aqueous humor dynamics: a review*, Open Ophthalmol. J. **4** (2010), 52–59.

[17] B. Goldhagen, A. D. Proia, D. L. Epstein, and P. V. Rao, *Elevated levels of RhoA in the optic nerve head of human eyes with glaucoma*, J. Glaucoma **21** (2012), no. 8, 530–538.

[18] G. Guidoboni, A. Harris, J. C. Arciero, B. A. Siesky, A. Amireskandari, A. L. Gerber, A. H. Huck, N. J. Kim, S. Cassani, and L. Carichino, *Mathematical modeling approaches in the study of glaucoma disparities among people of African and European descents*, J. Coupl. Sys. Multiscale Dyn. **1** (2013), no. 1, 1–21.

[19] A. Heijl, M. C. Leske, B. Bengtsson, L. Hyman, B. Bengtsson, M. Hussein, and Early Manifest Glaucoma Trial Group, *Reduction of intraocular pressure and glaucoma progression: results from the Early Manifest Glaucoma Trial*, Arch. Opthalmol. **120** (2002), no. 10, 1268–1279.

[20] B. Hille, *Ion channels of excitable membranes*, 3rd ed., Sinauer, Sunderland, MA, 2001.

[21] M. Honjo, H. Tanihara, M. Inatani, N. Kido, T. Sawamura, B. Y. J. T. Yue, S. Narumiya, and Y. Honda, *Effects of rho-associated protein kinase inhibitor Y-27632 on intraocular pressure and outflow facility*, Invest. Ophth. Vis. Sci. **42** (2001), no. 1, 137–144.

[22] S. Iizuka, K. Kishida, S. Tsuboi, K. Emi, and R. Manabe, *Electrical characteristics of the isolated dog ciliary body*, Curr. Eye. Res. **3** (1984), no. 3, 417–421.

[23] M. Jaeger, M. Carin, M. Medale, and G. Tryggvason, *The osmotic migration of cells in a solute gradient*, Biophys. J. **77** (1999), no. 3, 1257–1267.

[24] J. W. Jerome, *Analytical approaches to charge transport in a moving medium*, Transport Theory Statist. Phys. **31** (2002), no. 4-6, 333–366. MR Zbl

[25] J. W. Jerome and R. Sacco, *Global weak solutions for an incompressible charged fluid with multi-scale couplings: initial-boundary-value problem*, Nonlinear Anal. **71** (2009), no. 12, e2487–e2497. MR Zbl

[26] G. Karniadakis, A. Beskok, and N. Aluru, *Microflows and nanoflows: fundamentals and simulation*, Interdisciplinary Applied Mathematics, no. 29, Springer, 2005. MR Zbl

[27] J. Keener and J. Sneyd, *Mathematical physiology*, Interdisciplinary Applied Mathematics, no. 8, Springer, 1998. MR Zbl

[28] J. W. Kiel, *Physiology of the intraocular pressure*, Pathophysiology of the eye: Glaucoma (J. Feher, ed.), Akademiai Kiadò, Budapest, 1998, pp. 109–144.

[29] J. W. Kiel, M. Hollingsworth, R. Rao, M. Chen, and H. A. Reitsamer, *Ciliary blood flow and aqueous humor production*, Prog. Retin. Eye Res. **30** (2011), no. 1, 1–17.

[30] J. W. Kiel, H. A. Reitsamer, J. S. Walker, and F. W. Kiel, *Effects of nitric oxide synthase inhibition on ciliary blood flow, aqueous production and intraocular pressure*, Exp. Eye Res. **73** (2001), no. 3, 355–364.

[31] T. Krupin, P. S. Reinach, O. A. Candia, and S. M. Podos, *Transepithelial electrical measurements on the isolated rabbit iris-ciliary body*, Exp. Eye Res. **38** (1984), no. 2, 115–123.

[32] H. H. Mark, *Aqueous humor dynamics in historical perspective*, Surv. Ophthalmol. **55** (2010), no. 1, 89–100.

[33] A. Mauri, A. Bortolossi, G. Novielli, and R. Sacco, *3D finite element modeling and simulation of industrial semiconductor devices including impact ionization*, J. Math. Ind. **5** (2015), art. id. 1. MR

[34] A. Mauri, R. Sacco, and M. Verri, *Electro-thermo-chemical computational models for 3D heterogeneous semiconductor device simulation*, Appl. Math. Model. **39** (2015), no. 14, 4057–4074. MR

[35] A. G. Mauri, L. Sala, P. Airoldi, G. Novielli, R. Sacco, S. Cassani, G. Guidoboni, B. Siesky, and A. Harris, *Electro-fluid dynamics of aqueous humor production: simulations and new directions*, J. Model. Ophthalmol. **1** (2016), no. 2, 48–58.

[36] R. A. Moses, *Intraocular pressure*, Adler's physiology of the eye: clinical application (R. A. Moses and W. M. Hart, eds.), Mosby, St. Louis, 1987, pp. 223–245.

[37] D. A. Neamen, *Semiconductor physics and devices*, McGraw-Hill, New York, 1997.

[38] H. A. Quigley and A. T. Broman, *The number of people with glaucoma worldwide in 2010 and 2020*, Brit. J. Ophthalmol. **90** (2006), 262–267.

[39] V. P. Rao and D. L. Epstein, *Rho GTPase/Rho kinase inhibition as a novel target for the treatment of glaucoma*, BioDrugs **21** (2007), no. 3, 167–177.

[40] Y. Rosenfeld, *Free-energy model for the inhomogeneous hard-sphere fluid mixture and density-functional theory of freezing*, Phys. Rev. Lett. **63** (1989), no. 9, 980–983.

[41] Y. Rosenfeld, M. Schmidt, H. Löwen, and P. Tarazona, *Fundamental-measure free-energy density functional for hard spheres: dimensional crossover and freezing*, Phys. Rev. E **55** (1997), no. 4, 4245–4263.

[42] R. Roth, *Introduction to density functional theory of classical systems: theory and applications*, lecture notes, Universität Stuttgart, 2006.

[43] R. Roth, R. Evans, A. Lang, and G. Kahl, *Fundamental measure theory for hard-sphere mixtures revisited: the White Bear version*, J. Phys. Condens. Mat. **14** (2002), no. 46, art. id. 12063.

[44] I. Rubinstein, *Electro-diffusion of ions*, SIAM Studies in Applied Mathematics, no. 11, SIAM, Philadelphia, 1990. MR

[45] R. Sacco, P. Airoldi, A. G. Mauri, and J. W. Jerome, *Three-dimensional simulation of biological ion channels under mechanical, thermal and fluid forces*, Appl. Math. Model. **43** (2017), 221–251. MR

[46] K. L. Schey, Z. Wang, J. L. Wenke, and Y. Qi, *Aquaporins in the eye: expression, function, and roles in ocular disease*, Biochim. Biophys. Acta **1840** (2014), no. 5, 1513–1523.

[47] M. Schmuck, *Analysis of the Navier–Stokes–Nernst–Planck–Poisson system*, Math. Models Methods Appl. Sci. **19** (2009), no. 6, 993–1015. MR Zbl

[48] M. Shahidullah, W. H. Al-Malki, and N. A. Delamere, *Mechanism of aqueous humor secretion, its regulation and relevance to glaucoma*, Glaucoma: basic and clinical concepts (S. Rumelt, ed.), InTech, London, 2011, pp. 3–32.

[49] M. B. Shields, *Study guide for glaucoma*, Williams & Wilkins, Baltimore, 1982.

[50] A. J. Sit, C. B. Nau, J. W. McLaren, D. H. Johnson, and D. Hodge, *Circadian variation of aqueous dynamics in young healthy adults*, Invest. Ophth. Vis. Sci. **49** (2008), no. 4, 1473–1479.

[51] J. A. Stratton, *Electromagnetic theory*, IEEE, Piscataway, NJ, 2007. Zbl

[52] M. Szopos, S. Cassani, G. Guidoboni, C. Prud'homme, R. Sacco, B. Siesky, and A. Harris, *Mathematical modeling of aqueous humor flow and intraocular pressure under uncertainty: towards individualized glaucoma management*, J. Model. Ophthalmol. **1** (2016), no. 2, 29–39.

[53] C. H. To, C. W. Kong, C. Y. Chan, M. Shahidullah, and C. W. Do, *The mechanism of aqueous humour formation*, Clin. Exp. Optom. **85** (2002), no. 6, 335–349.

[54] C. B. Toris, *Pharmacology of aqueous humor formation*, Encyclopedia of the eye (D. A. Dartt, ed.), Academic, Cambridge, MA, 2010, pp. 312–315.

[55] H. Tsukaguchi, T. Tokui, B. Mackenzie, U. V. Berger, X. Z. Chen, Y. Wang, R. F. Brubaker, and M. A. Hediger, *A family of mammalian Na+-dependent L-ascorbic acid transporters*, Nature **399** (1999), no. 6731, 70–75.

[56] A. Viswanathan, *Ocular biochemistry: tears, cornea, lens, aqueous, vitreous, retina and rhodopsin*, presentation, 2017.

[57] T. Watanabe and Y. Saito, *Characteristics of ion transport across the isolated ciliary epithelium of the toad as studied by electrical measurements*, Exp. Eye Res. **27** (1978), no. 2, 215–226.

[58] P. J. Wistrand, *Carbonic anhydrase in the anterior uvea of the rabbit*, Acta Physiol. Scand. **24** (1951), no. 2–3, 145–148.

[59] A. Wójcik-Gryciuk, M. Skup, and W. J. Waleszczyk, *Glaucoma: state of the art and perspectives on treatment*, Restor. Neurol. Neuros. **34** (2016), no. 1, 107–123.

[60] S. Yao, D. E. Hertzog, S. Zeng, J. C. Mikkelsen, Jr., and J. G. Santiago, *Porous glass electroosmotic pumps: design and experiments*, J. Colloid Interf. Sci. **268** (2003), no. 1, 143–153.

Received December 20, 2017. Revised December 6, 2018.

LORENZO SALA: sala@unistra.fr
Institut de Recherche Mathématique Avancée, Université de Strasbourg, CNRS, Strasbourg, France

AURELIO GIANCARLO MAURI: aureliogiancarlo.mauri@polimi.it
Dipartimento di Matematica, Politecnico di Milano, Milano, Italy

RICCARDO SACCO: riccardo.sacco@polimi.it
Dipartimento di Matematica, Politecnico di Milano, Milano, Italy

DARIO MESSENIO: dmessenio@virgilio.it
Eye Clinic, Department of Clinical Science, ASST Fatebenefratelli Sacco, University of Milan, Milan, Italy

GIOVANNA GUIDOBONI: guidobonig@missouri.edu
Department of Electrical Engineering and Computer Science, University of Missouri, Columbia, MO, United States

ALON HARRIS: alharris@indiana.edu
Eugene and Marilyn Glick Eye Institute, Indiana University School of Medicine, Indianapolis, IN, United States

COMM. APP. MATH. AND COMP. SCI.
Vol. 14, No. 1, 2019
dx.doi.org/10.2140/camcos.2019.14.105

msp

AN ADAPTIVE LOCAL DISCRETE CONVOLUTION METHOD FOR THE NUMERICAL SOLUTION OF MAXWELL'S EQUATIONS

BORIS LO AND PHILLIP COLELLA

We present a numerical method for solving the free-space Maxwell's equations in three dimensions using compact convolution kernels on a rectangular grid. We first rewrite Maxwell's equations as a system of wave equations with auxiliary variables and discretize its solution from the method of spherical means. The algorithm has been extended to be used on a locally refined nested hierarchy of rectangular grids.

1. Introduction

We want to solve the free-space three-dimensional Maxwell's equations

$$\frac{\partial \boldsymbol{E}}{\partial t} = c\nabla \times \boldsymbol{B} - 4\pi \boldsymbol{J}, \tag{1}$$

$$\frac{\partial \boldsymbol{B}}{\partial t} = -c\nabla \times \boldsymbol{E}, \tag{2}$$

$$\nabla \cdot \boldsymbol{E} = 4\pi\rho, \tag{3}$$

$$\nabla \cdot \boldsymbol{B} = 0. \tag{4}$$

In our previous work [7], we considered Maxwell's equations in Fourier space, derived a real-space propagator for the system, and discretized the exact solution from Duhamel's formula. This propagator includes Helmholtz decomposition operators. The Helmholtz decomposition operators require global Poisson solves at every time step, which offsets the computational advantages of the local convolution kernel parts of the propagator.

In the present work, we get around this difficulty by applying a similar technique to an auxiliary system of equations instead of directly to Maxwell's equations. This auxiliary system is a system of wave equations for $\boldsymbol{E}, \boldsymbol{B}$ combined with constraints which, if satisfied initially, are satisfied for all time, such that the solutions of the auxiliary system are solutions to Maxwell's equations. We then apply Kirchhoff's formula to this system and discretize the resulting convolution

MSC2010: primary 65M55, 65M80; secondary 78-04.
Keywords: electromagnetics, Green's function, propagator method, adaptive mesh refinement.

equations. The convolution kernels from this propagator are the same as the local kernels for the transverse Maxwell's equations' propagator in [7], and thus, the same discretization techniques and domain decomposition can be applied. The locality of the convolution kernels allows us to naturally incorporate adaptive mesh refinement (AMR), where the domain is divided up into a nested hierarchy of rectangular grids at each refinement level.

In Section 2 we introduce the auxiliary system and show the analytic solution for Maxwell's equations in terms of a propagator with specified charges and currents. In Section 3, we describe the discretization process briefly, and discuss in detail the local discrete convolution method (LDCM) Maxwell solver for a single level and its extension to multiple levels. In Section 4 we present a number of numerical tests that show an implementation of our algorithm. Finally, in Section 5 we make some concluding remarks.

2. Problem statement and derivation of propagators

2.1. ***Maxwell's equations.*** Introducing $\boldsymbol{\Phi} \equiv \nabla \times \boldsymbol{B}$ and $\boldsymbol{\Psi} \equiv \nabla \times \boldsymbol{E}$, we rewrite Maxwell's equations, with ρ, $\boldsymbol{J}$ specified, as the auxiliary system of wave equations

$$\frac{\partial \boldsymbol{E}}{\partial t} = c\boldsymbol{\Phi} - 4\pi \boldsymbol{J}, \tag{5}$$

$$\frac{\partial \boldsymbol{\Phi}}{\partial t} = c\nabla^2 \boldsymbol{E} - 4\pi c \nabla \rho, \tag{6}$$

$$\frac{\partial \boldsymbol{B}}{\partial t} = -c\boldsymbol{\Psi}, \tag{7}$$

$$\frac{\partial \boldsymbol{\Psi}}{\partial t} = -c\nabla^2 \boldsymbol{B} - 4\pi \nabla \times \boldsymbol{J}. \tag{8}$$

If the initial conditions satisfy

$$\boldsymbol{\Psi} = \nabla \times \boldsymbol{E}, \tag{9}$$

$$\boldsymbol{\Phi} = \nabla \times \boldsymbol{B}, \tag{10}$$

$$\nabla \cdot \boldsymbol{E} = 4\pi\rho, \tag{11}$$

$$\nabla \cdot \boldsymbol{B} = 0, \tag{12}$$

then the auxiliary system is equivalent to the original Maxwell system. To show this, consider the four error quantities associated with the initial value constraints at $t = 0$:

$$\boldsymbol{K}_B = \boldsymbol{\Phi} - \nabla \times \boldsymbol{B}, \tag{13}$$

$$\boldsymbol{K}_E = \boldsymbol{\Psi} - \nabla \times \boldsymbol{E}, \tag{14}$$

$$D_B = \nabla \cdot \boldsymbol{B}, \tag{15}$$

$$D_E = \nabla \cdot \boldsymbol{E} - 4\pi\rho. \tag{16}$$

Using the auxiliary system (5)–(8), the four evolution equations associated with these quantities are given by

$$\frac{\partial \boldsymbol{K}_B}{\partial t} = c\nabla \times \boldsymbol{K}_E + c\nabla D_E, \tag{17}$$

$$\frac{\partial \boldsymbol{K}_E}{\partial t} = -c\nabla \times \boldsymbol{K}_B - c\nabla D_B, \tag{18}$$

$$\frac{\partial D_B}{\partial t} = -c\nabla \cdot \boldsymbol{K}_E, \tag{19}$$

$$\frac{\partial D_E}{\partial t} = c\nabla \cdot \boldsymbol{K}_B. \tag{20}$$

It is clear that if $\boldsymbol{K}_B$, $\boldsymbol{K}_E$, D_B, D_E vanish at $t = 0$, then they remain zero for all time after. In particular, the symbol of the linear operator associated with these eight evolution equations has the eigenvalues $\pm ic|\boldsymbol{k}|$ each with a multiplicity of four. Since errors propagate away with the same wave speed, any error will not accumulate at a fixed location and be a potential source of numerical instability. The initial value problem (5)–(8) is well posed even if the initial-value constraints (13)–(16) are not satisfied. The constraints are required only so that the solution is equivalent to the solution to Maxwell's equations. Since the two systems are equivalent, the solutions for $\boldsymbol{E}$, $\boldsymbol{B}$ obtained from the auxiliary system will also be the solution to the original Maxwell system.

The solutions to (5)–(8) are given by Kirchhoff's formula using the method of spherical means [13, p. 231]. Defining the kernels $G^{\Delta t}$ and $H^{\Delta t}$ as

$$G^{\Delta t}(\boldsymbol{z}) \equiv \frac{\delta(|\boldsymbol{z}| - c\Delta t)}{4\pi c\Delta t}, \tag{21}$$

$$H^{\Delta t}(\boldsymbol{z}) \equiv \frac{1}{c}\frac{\partial}{\partial s}\left(\frac{\delta(|\boldsymbol{z}| - cs)}{4\pi cs}\right)\bigg|_{s=\Delta t}, \tag{22}$$

$G^{\Delta t}$ is a spherical delta distribution with radius $c\Delta t$. The action of the propagator on an arbitrary state vector $\boldsymbol{h}(\boldsymbol{x}) \equiv [\boldsymbol{f}(\boldsymbol{x}), \boldsymbol{g}(\boldsymbol{x})]^T$ with $\boldsymbol{f}, \boldsymbol{g} \in \mathbb{R}^3$ is given by

$$\mathscr{P}^{\Delta t}[\boldsymbol{h}] = \begin{bmatrix} H^{\Delta t} * \boldsymbol{f} + G^{\Delta t} * \boldsymbol{g} \\ G^{\Delta t} * \nabla^2 \boldsymbol{f} + H^{\Delta t} * \boldsymbol{g} \end{bmatrix}, \tag{23}$$

where the scalar convolution kernel with vector quantity is defined as convolution with each component and convolutions are defined spatially as

$$(K * f)(\boldsymbol{x}) \equiv \int_{\mathbb{R}^3} K(\boldsymbol{y}) f(\boldsymbol{x} - \boldsymbol{y})\, d\boldsymbol{y}. \tag{24}$$

In particular, the solution to (5)–(6) is then given by

$$\begin{pmatrix} \boldsymbol{E}(\boldsymbol{x}, t+\Delta t) \\ \boldsymbol{\Phi}(\boldsymbol{x}, t+\Delta t) \end{pmatrix} = \mathscr{P}^{\Delta t}\left[\begin{pmatrix} \boldsymbol{E}(\boldsymbol{x}, t) \\ \boldsymbol{\Phi}(\boldsymbol{x}, t) \end{pmatrix}\right] - 4\pi \int_t^{t+\Delta t} \mathscr{P}^{t+\Delta t-s}\left[\begin{pmatrix} \boldsymbol{J}(\boldsymbol{x}, s) \\ c\nabla\rho(\boldsymbol{x}, s) \end{pmatrix}\right] ds. \tag{25}$$

The propagator for (7)–(8) is the same as that for (5)–(6), with the substitution $\Delta t \to -\Delta t$. Thus, the solution is given by

$$\begin{pmatrix} \boldsymbol{B}(\boldsymbol{x}, t+\Delta t) \\ \boldsymbol{\Psi}(\boldsymbol{x}, t+\Delta t) \end{pmatrix} = \mathcal{P}^{-\Delta t}\left[\begin{pmatrix} \boldsymbol{B}(\boldsymbol{x}, t) \\ \boldsymbol{\Psi}(\boldsymbol{x}, t) \end{pmatrix}\right] - 4\pi \int_t^{t+\Delta t} \mathcal{P}^{-(t+\Delta t-s)}\left[\begin{pmatrix} 0 \\ \nabla \times \boldsymbol{J}(\boldsymbol{x}, s) \end{pmatrix}\right] ds. \quad (26)$$

It can be seen from the Fourier transforms of the convolution kernels that

$$G^{-\Delta t} * f = -G^{\Delta t} * f, \quad (27)$$

$$H^{-\Delta t} * f = H^{\Delta t} * f. \quad (28)$$

In addition

$$H^{\Delta t} * f = \frac{1}{ct} G^{\Delta t} * f - \sum_{i=1}^{3} G_i^{\Delta t} * \frac{\partial f}{\partial z_i}, \quad (29)$$

$$G_i^{\Delta t}(\boldsymbol{z}) = \frac{z_i \delta(|\boldsymbol{z}| - c\Delta t)}{4\pi c \Delta t}. \quad (30)$$

With these, we have fully specified the solutions, (25) and (26), in terms of convolution with weighted spherical delta distributions. We note that it can be shown directly that $\boldsymbol{\Psi}(\boldsymbol{x}, t+\Delta t) = \nabla \times \boldsymbol{E}(\boldsymbol{x}, t+\Delta t)$ and $\boldsymbol{\Phi}(\boldsymbol{x}, t+\Delta t) = \nabla \times \boldsymbol{B}(\boldsymbol{x}, t+\Delta t)$ given the constraints are satisfied at t. When ρ, $\boldsymbol{J}$ are not specified but functions of field variables, instead of using Kirchhoff's formula and a quadrature scheme one can use Lawson's method [6] for time integration.

3. Discretization approach

3.1. ***Single-level algorithm.*** We consider a rectangular domain discretized with a Cartesian grid with grid spacing h with open boundary conditions. The convolutions in (25)–(26) are approximated with discrete convolutions on the grid. This requires a discretized representation of the convolution kernels, $G^{\Delta t,h} \approx G^{\Delta t}(\boldsymbol{z})$ and $H^{\Delta t,h} \approx H^{\Delta t}(\boldsymbol{z})$, on the grid. $H^{\Delta t,h}$ is obtained by (29), so that the problem reduces to only creating discrete representations of (weighted) spherical delta distributions. We refer the reader to [7] for a detailed treatment of the discretization of the convolution kernels. The resulting discrete convolution kernels have compact support just like their continuous counterparts. Thus, the discrete convolutions can be computed exactly using Hockney's method [5].

The overall time-stepping algorithm is given in Algorithm 1. This defines the discrete evolution for $\boldsymbol{E}$, $\boldsymbol{B}$, since $\boldsymbol{\Phi}$, $\boldsymbol{\Psi}$ are computed at the beginning of every time step. The source term integrals are discretized using a closed Newton–Cotes quadrature scheme with step size $\Delta s = \Delta t/(M-1)$ where M is the number of

Initialize Newton–Cotes quadrature weights $\{w_m\}_{m=0}^{M}$
/* Create the convolution kernels with quadrature step size Δs and spacing h */
Compute $G^{\Delta s,h}$, and $H^{\Delta s,h}$
/* Begin time-stepping loop */
for $n = 1, 2, \ldots$ **do**
  /* Initialize the fields for this time step */
  /* Let $U^{(n),h} \approx U(n\Delta t, \boldsymbol{x})$ */
  $\boldsymbol{E}^{(n),h} \leftarrow \boldsymbol{E}^{(n-1),h}$, $\boldsymbol{B}^{(n),h} \leftarrow \boldsymbol{B}^{(n-1),h}$, $\boldsymbol{\Phi}^{(n),h} \leftarrow \nabla \times \boldsymbol{E}^{(n),h}$, $\boldsymbol{\Psi}^{(n),h} \leftarrow \nabla \times \boldsymbol{B}^{(n),h}$
  /* Begin quadrature loop */
  for $m = 1, 2, \ldots, M$ **do**
    /* Add in source terms evaluated at $t = (n-1)\Delta t + (m-1)\Delta s$ */
    $\boldsymbol{E}^{(n),h} \leftarrow \boldsymbol{E}^{(n),h} - w_m 4\pi \boldsymbol{J}^h$
    $\boldsymbol{\Phi}^{(n),h} \leftarrow \boldsymbol{\Phi}^{(n),h} - w_m 4\pi c \nabla \rho^h$
    $\boldsymbol{\Psi}^{(n),h} \leftarrow \boldsymbol{\Psi}^{(n),h} - w_m 4\pi \nabla \times \boldsymbol{J}^h$
    /* Apply propagator to the fields except final quadrature point */
    if $m < M$ **then**

$$\begin{bmatrix} \boldsymbol{E}^{(n),h} \\ \boldsymbol{\Phi}^{(n),h} \end{bmatrix} \leftarrow \begin{bmatrix} H^{\Delta s,h} * \boldsymbol{E}^{(n),h} + G^{\Delta s,h} * \boldsymbol{\Phi}^{(n),h} \\ (G^{\Delta s,h} * \nabla^2) * \boldsymbol{E}^{(n),h} + H^{\Delta s,h} * \boldsymbol{\Phi}^{(n),h} \end{bmatrix}$$

$$\begin{bmatrix} \boldsymbol{B}^{(n),h} \\ \boldsymbol{\Psi}^{(n),h} \end{bmatrix} \leftarrow \begin{bmatrix} H^{\Delta s,h} * \boldsymbol{B}^{(n),h} - G^{\Delta s,h} * \boldsymbol{\Psi}^{(n),h} \\ -(G^{\Delta s,h} * \nabla^2) * \boldsymbol{B}^{(n),h} + H^{\Delta s,h} * \boldsymbol{\Psi}^{(n),h} \end{bmatrix}$$

    end if
  end for
  /* Enforcing constraints */
  $\boldsymbol{E}^{(n),h} \leftarrow \boldsymbol{E}^{(n),h} + \eta(\mathcal{L}\boldsymbol{E}^{(n),h} - 4\pi \nabla \rho^h)$
  $\boldsymbol{B}^{(n),h} \leftarrow \boldsymbol{B}^{(n),h} + \eta \mathcal{L} \boldsymbol{B}^{(n),h}$
end for

Algorithm 1. Single-level LDCM for Maxwell's equations.

quadrature points. We choose a fixed step size quadrature because $\mathcal{P}^{t_1}[\mathcal{P}^{t_2}[\boldsymbol{U}]] = \mathcal{P}^{t_1+t_2}[\boldsymbol{U}]$, and therefore, we only need to create one propagator with step size Δs during initial setup.

Even though the divergence constraints are preserved by the continuous time evolution, deviations from (11)–(12) may be generated by discretization error. To help remedy this, we apply local filters [8] of the form

$$\boldsymbol{E} := \boldsymbol{E} + \eta(\mathcal{L}\boldsymbol{E} - 4\pi \nabla \rho), \tag{31}$$

$$\boldsymbol{B} := \boldsymbol{B} + \eta \mathcal{L} \boldsymbol{B}, \tag{32}$$

$$\mathcal{L}_{ij} = \partial_{x_i} \partial_{x_j}, \tag{33}$$

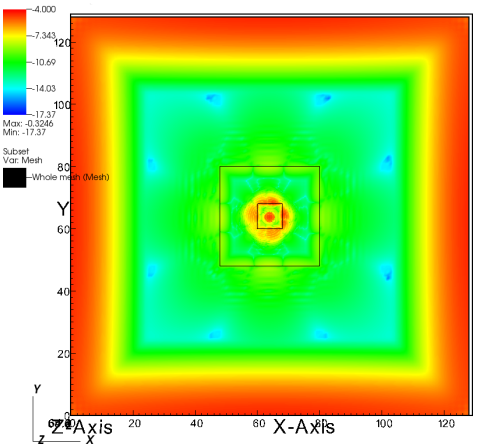


Figure 1. $\log_{10}(|\nabla \cdot \boldsymbol{E} - 4\pi\rho| / \max_{\boldsymbol{x}} 4\pi\rho)$ at $z = 0.5$ for the stopped translating spherical charge distribution problem at $t = \frac{200}{2048}$ for $N = 129$ showing that there are no reflected waves at the refinement boundaries.

where $\eta \sim \mathbb{O}(h^2)$ is a constant and $\mathcal{L}$ is a matrix-valued operator with the diagonal terms discretized with centered-difference approximations to the second derivative while the off-diagonal terms are products of centered-difference approximations to the first derivatives. This filtering step corresponds to applying an explicit diffusion step to the error in the longitudinal fields. Note that we do not have to do this for the curl constraints (9)–(10), since $\boldsymbol{\Phi}$, $\boldsymbol{\Psi}$ are reinitialized at the beginning of each time step.

3.2. ***Domain decomposition.*** Since the discretized version of the propagator involves only local operators, we can use standard domain decomposition to parallelize this algorithm. Consider a single-level domain, Ω_h, partitioned into rectangular patches. For each patch,

(1) at the beginning of each quadrature step, copy field values in ghost region from neighboring processors, and

(2) apply propagator to update local field values, invalidating values in ghost region.

The minimum width of the ghost region is determined by the size of the quadrature, Δs, and the order of the method because the size of the support of the spherical delta distributions is dependent on how far in time the fields are to be advanced.

For a point, $\boldsymbol{x}_k$, near the boundary, when applying the discrete convolutions we replace the field values outside the computational domain with the current field value at $\boldsymbol{x}_k$. This approximation leads to waves reflecting back into the computational domain. We could employ standard techniques for simulating infinite domain such as perfectly matched layer (PML) [2]. However, we wanted to focus on the

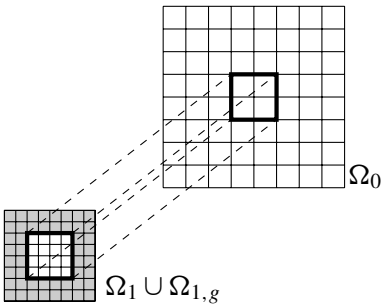


Figure 2. Example schematic of a two-level nested domain with factor of 2 refinement. The unshaded region is Ω_1 and the shaded region is the ghost region $\Omega_{1,g}$.

propagator method and not the boundary conditions. Therefore, in this work, we mitigate this reflection error with mesh refinement, by placing the boundary of the computational domain far away from the sources. This is possible because our method does not generate significant internal reflection at refinement boundaries as shown in Figure 1. The amplitude of the waves reaching the domain boundary will thus be weaker and the reflected error waves will also be smaller.

3.3. ***Multilevel algorithm.*** Consider now a hierarchy of nested rectangular grids, Ω_j, $j = 0, \ldots, J-1$, where the grid spacing for Ω^j is h/r^j for some refinement factor, $r \in \mathbb{Z}^+$, with $\Omega_j \cup \Omega_{j+1} = \Omega_{j+1}$, $j = 0, \ldots, J-2$. We introduce sampling and interpolation operators, $\mathcal{S}$ and $\mathcal{I}$, respectively, to communicate field values with the next immediate lower and upper levels. Similar to the ghost regions for each patch in parallelizing the single-level algorithm, we define a ghost region for each level, $\Omega_{j,g}$, where the width of the ghost region is determined by how far in time the fields are to be advanced. At the beginning of each quadrature step, except on the first level, for all nodes in $\Omega_{j,g}$ we interpolate $\boldsymbol{E}, \boldsymbol{B}$ from level $j-1$. After interpolating, except on the finest level, we replace $\boldsymbol{E}, \boldsymbol{B}$ at level j with field values from level $j+1$ on the nodes that are in $\Omega_j \cap \Omega_{j+1}$. A sample schematic of two levels with $r = 2$ is shown in Figure 2. After interpolating and sampling, each level is evolved independently with the propagator.

Let $f_j^{(n)}$ denote discretized f on level j and at time $t_n = n\Delta t$; the multilevel algorithm is outlined in Algorithm 2. Since (5)–(8) is a system of linear differential equations, we can use linear superposition to generate the overall solution to the problem in this multilevel setup; the solution is given by a composite where it takes the finest level values for any subdomain. For example, in the two-level case, let $\boldsymbol{U} = (\boldsymbol{E}, \boldsymbol{B}, \boldsymbol{\Psi}, \boldsymbol{\Phi})^T$; then the solution is given by

$$\boldsymbol{U}^{(n)} = \begin{cases} \boldsymbol{U}_1^{(n)} & \text{on } \Omega_1, \\ \boldsymbol{U}_0^{(n)} & \text{on } \Omega_0 \setminus \Omega_1. \end{cases} \tag{34}$$

Initialize Newton–Cotes quadrature weights $\{w_m\}_{m=0}^{M}$
for all levels Ω_j, $j = 0, \ldots, J-1$ **do**
    Initialize $\boldsymbol{U}_j^{(0)}$
    Compute $G^{\Delta s, h/r^j}$, and $H^{\Delta s, h/r^j}$
end for
`/* Begin time-stepping loop */`
for all $n = 1, 2, \ldots$ **do**
    for all levels Ω_j, $j = 0, \ldots, J-1$ **do**
        `/* Initialize the fields for this time step */`
        $\boldsymbol{U}_j^{(n)} \leftarrow \boldsymbol{U}_j^{(n-1)}$
        for quadrature step s **do**
            `/* Apply sampling operator except for level 0 */`
            $\boldsymbol{U}_{j-1}^{(n)} \leftarrow \mathcal{I}[\boldsymbol{U}_j^{(n)}]$ on Ω_j
            `/* Apply interpolation operator except for level` $J-1$ `*/`
            $\boldsymbol{U}_{j+1}^{(n)} \leftarrow \mathcal{I}[\boldsymbol{U}_j^{(n)}]$ on $\Omega_{j+1,g}$
            Apply single-level operations (add in source term and apply propagator)
        end for
    end for
    Sample and interpolate $\boldsymbol{E}$, $\boldsymbol{B}$ so that $\mathcal{L}$ can be applied on the refinement levels
    Enforce the constraints independently for each level
end for

Algorithm 2. Multilevel LDCM for Maxwell's equations.

Since we interpolate once every quadrature step, the width of $\Omega_{i,g}$ for level i has the same width as the ghost region required for domain decomposition.

Interpolation. We use high-order B-splines (see the Appendix) to interpolate the fields between levels similar to the ones used to regularize the delta distributions in the propagator. However, the choice of interpolant is more restrictive than the one used to regularize the delta distribution. The convergence of spherical quadrature when regularizing the delta distribution depends on the smoothness of the integrand [1]. However, we are interested in the regularized delta distribution as a discrete convolution kernel with some discretized function f. Numerically, the spherical quadrature and discrete convolution commute, and therefore, we relied on the smoothness of f for the convergence of the spherical quadrature. This allows us to use a C^0 high-order B-spline as a regularizer with the advantage that it has minimal support.

In this method, f is a field component or a component of the source terms. Since the field components must be sufficiently smooth for the spherical quadrature and the accuracy of the high-order finite difference operators applied to the field components also depends on smoothness, these translate into a smoothness requirement for the

for levels Ω_j, $j = 1, \ldots, J-1$ **do**
  if regrid **do**
    `/* Sample down starting from topmost level */`
    for $k = J-1, \ldots, j$ **do**
      $\boldsymbol{U}_{k-1}^{(n)} \leftarrow \mathcal{S}[\boldsymbol{U}_k^{(n)}]$ on $\Omega_k \cap \Omega_{j,\text{discard}}$
      `/* Discard part of domain that has been sampled from */`
      $\Omega_k \leftarrow \Omega_k \setminus (\Omega_k \cap \Omega_{j,\text{discard}})$
    end for
    $\Omega_j \leftarrow \Omega_j \cup \Omega_{j,\text{new}}$
    `/* Interpolate from level` $j-1$ `*/`
    $\boldsymbol{U}_j^{(n)} \leftarrow \mathcal{I}[\boldsymbol{U}_{j-1}^{(n)}]$ on $\Omega_{j,\text{new}}$
    Enforce the constraints
  end if
end for

Algorithm 3. Regridding algorithm.

interpolants. For a q-th-order method, we would need the error from the spherical quadrature to be at least $\mathbb{O}(h^q)$, which requires $f \in C^q$. Therefore, the interpolant must also be at least q-th-order accurate and C^q.

Regridding. For an adaptive version of this method, instead of a fixed hierarchy of rectangular grids, we regrid at the beginning of any time step as needed. Suppose we wish to regrid level j, $j > 0$; let $\Omega_j = \Omega_{j,\text{discard}} \cup \Omega_{j,\text{keep}}$ before regridding and $\Omega_j = \Omega_{j,\text{keep}} \cup \Omega_{j,\text{new}}$ after regridding. First sample down on $\Omega_{j,\text{discard}}$; then interpolate on $\Omega_{j,\text{new}}$ using the same sampling and interpolating operators. The regridding algorithm is outlined in Algorithm 3.

4. Numerical results

We implemented a fourth-order version of our Maxwell solver with $c = 1$; the one-step error for the solver is $\mathbb{O}(h^q)$, but after some number of time steps the total error will be $\mathbb{O}(h^{q-1})$ for a method that has a one-step error of $\mathbb{O}(h^q)$ and $\Delta t = \mathbb{O}(h)$. We used sixth-order centered differences for the spatial derivatives, the fifth-order $\frac{3}{8}$ Simpson's rule for the source integration, $W_{6,0}$ for the discrete delta distribution, and $W_{6,6}$ for the interpolation operator. The discrete convolutions are performed via Hockney's method extending the domain equal to the support of the discrete convolution kernels and using the FFTW library [4]. The domain at the coarsest level is a unit cube and each level is divided into 33^3 node patches with factor of 4 refinement; every level has the same number of nodes, N. The filter parameter at level j is $\eta_j = \frac{45}{544} h_j^2$. For each test, Δt is the same across refinement levels.

4.1. ***Translating spherical charge distribution.*** For the first numerical test, we used a C^6 spherical-support charge distribution with a spatially constant $\boldsymbol{v}(t)$.

$$\rho(\boldsymbol{x},t) = \begin{cases} a(r(t)-r(t)^2)^6, & r<1, \\ 0, & r\geq 1, \end{cases} \qquad r = \frac{1}{R_0}\|\boldsymbol{x}-\boldsymbol{x}_0\|, \tag{35}$$

$$\boldsymbol{J}(\boldsymbol{x},t) = \boldsymbol{v}(t)\rho(\boldsymbol{x},t), \tag{36}$$

$$\boldsymbol{v}(t) = \nu d\pi \tfrac{35}{16}\sin^7(2\pi\nu t)\hat{\boldsymbol{v}}. \tag{37}$$

The electrostatic solution is given by

$$\boldsymbol{E}(\boldsymbol{x}) = 4\pi R_0 a\hat{r}\begin{cases} \frac{1}{9}r^7 - \frac{3}{5}r^8 + \frac{15}{11}r^9 - \frac{5}{3}r^{10} + \frac{15}{13}r^{11} - \frac{3}{7}r^{12} + \frac{1}{15}r^{13}, & r<1 \\ \frac{1}{45045}r^{-2}, & r\geq 1, \end{cases} \tag{38}$$

$$\boldsymbol{B}(\boldsymbol{x}) = 0. \tag{39}$$

Here $\hat{r}$ is with respect to $\boldsymbol{x}_0$, and we use this as the initial condition for this test problem. We perform this test on fixed grids with two refinement levels, $\Omega_1 = \left[\frac{3}{8}, \frac{5}{8}\right]^3$ and $\Omega_2 = \left[\frac{15}{32}, \frac{17}{32}\right]^3$, with parameters $a = 10^4$, $d = \frac{1}{256}$, $\nu = \frac{1024}{80}$, $R_0 = \frac{1}{72}$, $\boldsymbol{x}_0 = \left(\frac{127}{256}, \frac{127}{256}, \frac{127}{256}\right)$, $\hat{\boldsymbol{v}} = \left(\cos\frac{\sqrt{3}}{3}\cos\frac{\sqrt{2}}{3}, \sin\frac{\sqrt{3}}{3}\cos\frac{\sqrt{2}}{3}, \sin\frac{\sqrt{2}}{3}\right)$, and $N = (65, 129, 257)$ with $\Delta t = \left(\frac{1}{1024}, \frac{1}{2048}, \frac{1}{4096}\right)$, respectively; this corresponds to CFL $= 1$ at the finest level, out to $t_{\text{final}} = \frac{200}{1024}$. Figure 3 shows the E_x Richardson convergence rate estimate and the associated ℓ_∞ error as well as the absolute convergence rate and associated ℓ_∞ errors for $\nabla\cdot\boldsymbol{E} - 4\pi\rho$ on the three grids in Ω_2 as a function of time step, and as expected our solution shows fourth-order convergence.

Electrostatic test. We performed another test with same discretization and parameters but stopped the charge distribution after $t = \frac{40}{1024}$ and then ran out to $t_{\text{final}} = \frac{100}{1024}$

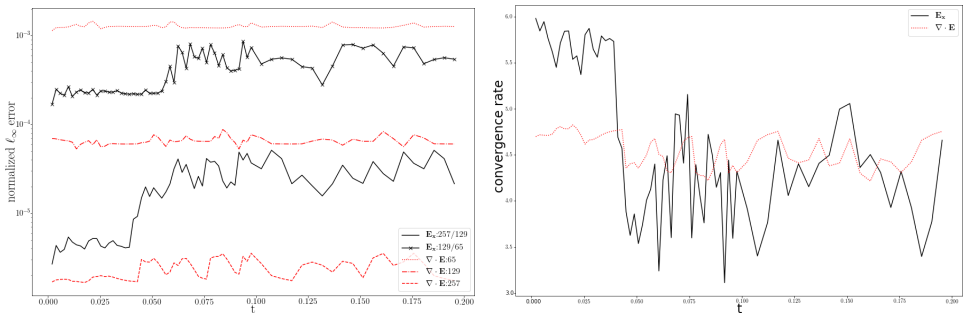


Figure 3. ℓ_∞ error values and convergence results for E_x and $\nabla\cdot\boldsymbol{E} - 4\pi\rho$ for the translating spherical charge distribution problem as a function of time in Ω_2. On the left are the normalized ℓ_∞ errors for E_x and $\nabla\cdot\boldsymbol{E} - 4\pi\rho$. The errors for E_x are obtained from the difference of sampled field values from $N = 257$ with $N = 129$ and also from sampled $N = 129$ with $N = 65$ test case. The E_x error is normalized by the max norm of the electrostatic solution (≈ 0.0694795), and $\nabla\cdot\boldsymbol{E} - 4\pi\rho$ error is normalized by $\max_{\boldsymbol{x}} 4\pi\rho \approx 30.6796$. On the right are the associated convergence rates.

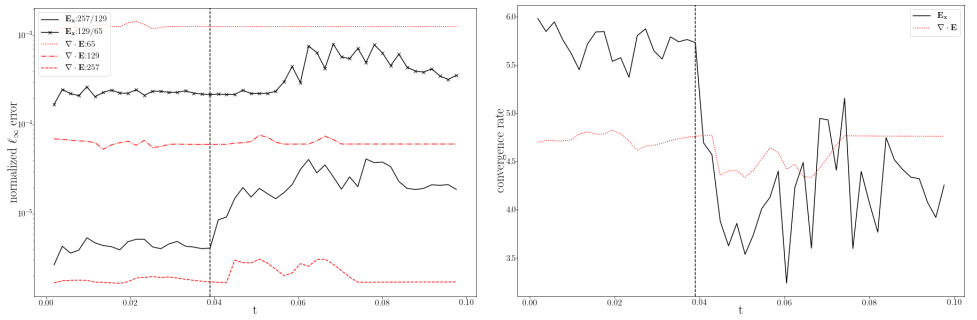


Figure 4. ℓ_∞ error values and convergence results for E_x and $\nabla \cdot \boldsymbol{E} - 4\pi\rho$ for the stopped spherical charge distribution problem as a function of time in Ω_2. On the left are the normalized ℓ_∞ errors for E_x and $\nabla \cdot \boldsymbol{E} - 4\pi\rho$. The errors for E_x are obtained from the difference of sampled field values from $N = 257$ with $N = 129$ and also from sampled $N = 129$ with $N = 65$ test case. The E_x error is normalized by the max norm of the electrostatic solution (≈ 0.0694795) and $\nabla \cdot \boldsymbol{E} - 4\pi\rho$ error is normalized by $\max_{\boldsymbol{x}} 4\pi\rho \approx 30.6796$. On the right are the associated convergence rates. The vertical line indicates the time at which the charge distribution stops moving.

to show that the solver recovers the electrostatic solution. Figure 4 shows the E_x Richardson convergence rate estimate and associated ℓ_∞ error as well as the absolute convergence rate and associated ℓ_∞ errors for $\nabla \cdot \boldsymbol{E} - 4\pi\rho$ on the three grids in Ω_2 as a function of time step, and as expected our solution shows fourth-order convergence.

Regridding test. We tested our regridding algorithm with the translating charge distribution with $\boldsymbol{v} = vd\pi \sin(2vt)\hat{\boldsymbol{x}}$, $a = \frac{1}{160}$, $d = \frac{1}{64}$, $\boldsymbol{x}_0 = \left(\frac{31}{64}, \frac{1}{2}, \frac{1}{2}\right)$, $v = \frac{1024}{80}$, $t_{\text{final}} = \frac{800}{1024}$, and other parameters being the same. We kept Ω_1 the same and fixed, but regridded Ω_2 starts with $\Omega_{2,a}$ and changes between $\Omega_{2,a}$ and $\Omega_{2,b}$ whenever the x coordinate of the center of the charge distribution crosses $\frac{63}{128}$, where $\Omega_{2,a}$ is the rectangular prism defined by the corner points $\left(\frac{29}{64}, \frac{17}{32}, \frac{17}{32}\right)$ and $\left(\frac{33}{64}, \frac{17}{32}, \frac{17}{32}\right)$, and $\Omega_{2,b} = \left[\frac{15}{32}, \frac{17}{32}\right]^3$; effectively Ω_2 oscillates in the x direction with amplitude $\frac{1}{64}$ in the direction of the charge motion. Figure 5 shows E_x and the regridding domains for $N = 129$. Figure 6 shows the E_x Richardson convergence rate estimate and the associated ℓ_∞ error as well as the absolute convergence rate and associated ℓ_∞ errors for $\nabla \cdot \boldsymbol{E} - 4\pi\rho$ on the three grids in Ω_2 as a function of time step and our solution shows fifth-order convergence.

4.2. ***Divergence-free current source.*** We've also tested with a divergence-free current source of the form

$$J_x(x, y, z, t) = -100\frac{y - y_0}{r} \sin\frac{\pi r}{2a} \cos^{10}\frac{\pi r}{2a} \cos^{11}\frac{\pi(z - z_0)}{d} \sin(2\pi vt), \quad (40)$$

$$J_y(x, y, z, t) = \quad 100\frac{x - x_0}{r} \sin\frac{\pi r}{2a} \cos^{10}\frac{\pi r}{2a} \cos^{11}\frac{\pi(z - z_0)}{d} \sin(2\pi vt), \quad (41)$$

$$J_z(x, y, z, t) = 0, \quad (42)$$

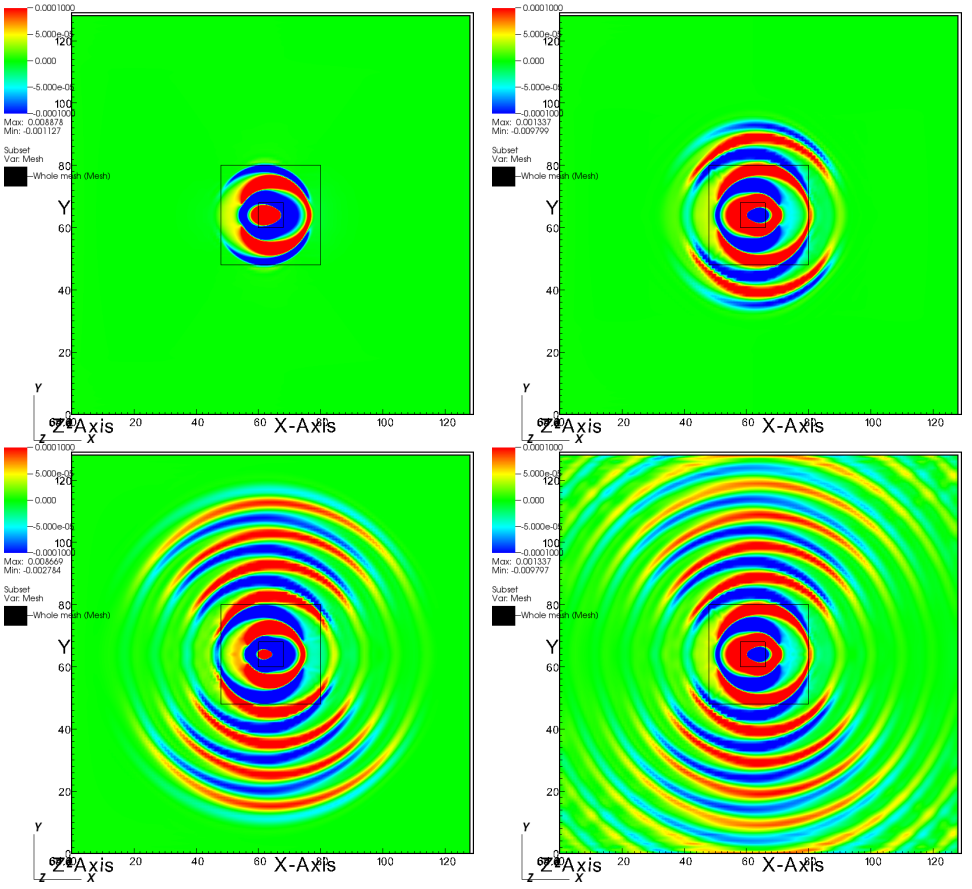


Figure 5. E_x minus the instantaneous electrostatic solution, at $z = \frac{1}{2}$, for the spherical charge distribution problem with regridding for $N = 129$. Top left: $t = \frac{256}{2048}$, charge distribution moving to the right, has almost reached its rightmost position, $\Omega_2 = \Omega_{2,b}$. Top right: $t = \frac{480}{2048}$, charge distribution is at its leftmost position, $\Omega_2 = \Omega_{2,a}$. Bottom left: $t = \frac{864}{2048}$, charge distribution moving to the left, $\Omega_2 = \Omega_{2,b}$. Bottom right: $t = \frac{1600}{2048}$, final time step, $\Omega_2 = \Omega_{2,a}$.

where $r = \sqrt{(x - x_0)^2 + (y - y_0)^2}$ with parameters $a = \frac{3}{160}$, $d = \frac{13}{320}$, $x_0 = y_0 = z_0 = 0.5$, and $\nu = 20$, and using the same refinement levels, discretization, and t_{final} as the fixed-grids translating-charge problem. Figure 7 shows the E_x Richardson convergence rate estimate and the associated ℓ_∞ error as well as the absolute convergence rate and associated ℓ_∞ errors for $\nabla \cdot \boldsymbol{E}$ on the three grids in Ω_2 as a function of time step, and as expected our solution shows fourth-order convergence.

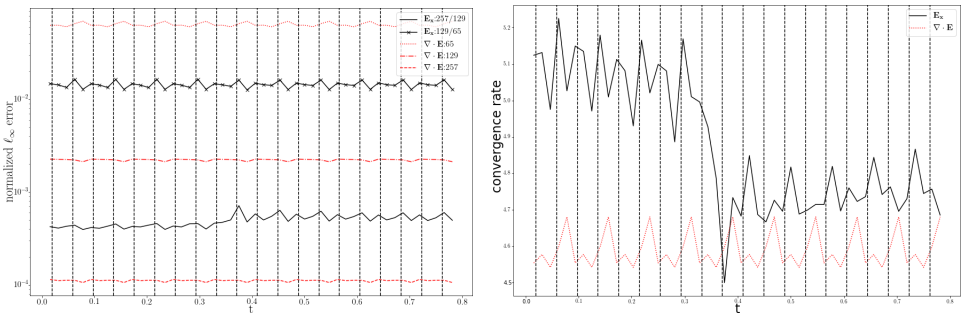


Figure 6. ℓ_∞ error values and convergence results for E_x and $\nabla \cdot \boldsymbol{E} - 4\pi\rho$ for the regridding spherical charge distribution problem as a function of time in Ω_2. On the left are the normalized ℓ_∞ errors for E_x and $\nabla \cdot \boldsymbol{E} - 4\pi\rho$. The errors for E_x are obtained from the difference of sampled field values from $N = 257$ with $N = 129$ and also from sampled $N = 129$ with $N = 65$ test case. The E_x error is normalized by the max norm of the electrostatic solution (≈ 0.0312658), and $\nabla \cdot \boldsymbol{E} - 4\pi\rho$ error is normalized by $\max_{\boldsymbol{x}} 4\pi\rho \approx 30.6796$. On the right are the associated convergence rates. The vertical lines are the times at which regridding occurs.

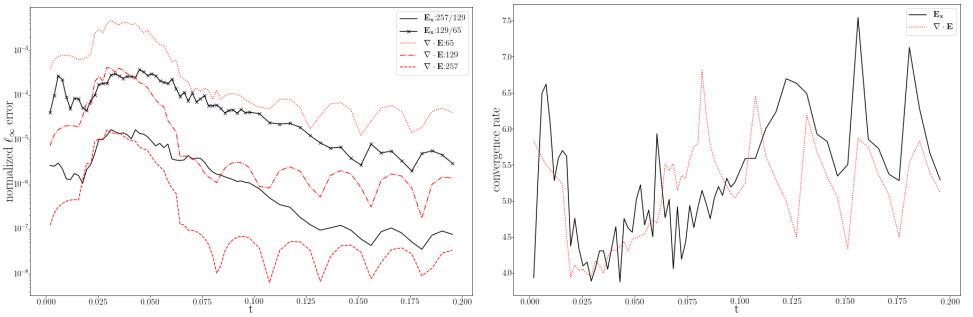


Figure 7. ℓ_∞ error values and convergence results for E_x and $\nabla \cdot \boldsymbol{E} - 4\pi\rho$ for the divergence-free current problem as a function of time in Ω_2. On the left are the normalized ℓ_∞ errors for E_x and $\nabla \cdot \boldsymbol{E} - 4\pi\rho$. The errors for E_x are obtained from the difference of sampled field values from $N = 257$ with $N = 129$ and also from sampled $N = 129$ with $N = 65$ test case. The E_x error is normalized by $|(4\pi/\nu)\max_{r,z} J_x| \approx |10.2341 \sin(2\pi\nu t)|$ and $\nabla \cdot \boldsymbol{E}$ is normalized by $|(4\pi/\nu a)\max_{r,z} J_x| \approx |545.8187 \sin(2\pi\nu t)|$. On the right are the associated convergence rates.

5. Conclusion

We have presented a new version of our Green's function numerical method for Maxwell's equations. This new formulation results in a completely local propagator that does not require Helmholtz decomposition. In principle, the method can choose any CFL but at the cost of larger ghost regions. We have demonstrated a high-order adaptive version of the solver in some test examples. In the future, we are interested in incorporating this method in EM PIC using Lawson's method where the fields and

particles are evolved together with a Runge–Kutta scheme with an extra propagator step for the fields.

Appendix: High-order B-splines

For completeness, we give the B-splines used in our implementation for the delta approximants and interpolants. Detailed discussions on creating high-order B-splines are given in [7; 3; 12; 11; 10; 9]. $W_{q,p}$ denotes a q-th-order accurate, C^p B-spline:

$$W_{6,0}(x) = \begin{cases} -\frac{1}{12}|x|^5+\frac{1}{4}|x|^4+\frac{5}{12}|x|^3-\frac{5}{4}|x|^2-\frac{1}{3}|x|+1, & |x|\in[0,1], \\ \frac{1}{24}|x|^5-\frac{3}{8}|x|^4+\frac{25}{24}|x|^3-\frac{5}{8}|x|^2-\frac{13}{12}|x|+1, & |x|\in[1,2], \\ -\frac{1}{120}|x|^5+\frac{1}{8}|x|^4-\frac{17}{24}|x|^3+\frac{15}{8}|x|^2-\frac{137}{60}|x|+1, & |x|\in[2,3], \\ 0, & |x|>3, \end{cases} \tag{43}$$

$$W_{6,6}(x) = \begin{cases} -\frac{665}{12048}|x|^9+\frac{665}{3012}|x|^8-\frac{2419}{12048}|x|^7-\frac{2437}{12048}|x|^6 \\ \quad +\frac{2723}{3012}|x|^4-\frac{4543}{3012}|x|^2+\frac{19177}{21084}, & |x|\in[0,1], \\ \frac{133}{4016}|x|^9-\frac{399}{1004}|x|^8+\frac{39659}{20080}|x|^7-\frac{104409}{20080}|x|^6+\frac{23443}{3012}|x|^5 \\ \quad -\frac{14175}{2008}|x|^4+\frac{7553}{1506}|x|^3-\frac{32207}{10040}|x|^2+\frac{2933}{15060}|x|+\frac{13081}{14056}, & |x|\in[1,2], \\ -\frac{133}{12048}|x|^9+\frac{665}{3012}|x|^8-\frac{114139}{60240}|x|^7+\frac{109283}{12048}|x|^6-\frac{79303}{3012}|x|^5 \\ \quad +\frac{283423}{6024}|x|^4-\frac{75215}{1506}|x|^3+\frac{170023}{6024}|x|^2-\frac{90923}{15060}|x|-\frac{17653}{42168}, & |x|\in[2,3], \\ \frac{19}{12048}|x|^9-\frac{133}{3012}|x|^8+\frac{225859}{421680}|x|^7-\frac{221003}{60240}|x|^6+\frac{23299}{1506}|x|^5 \\ \quad -\frac{30793}{753}|x|^4+\frac{49184}{753}|x|^3-\frac{208208}{3765}|x|^2\frac{53632}{3765}|x|+\frac{32512}{5271}, & |x|\in[3,4], \\ 0, & |x|>4. \end{cases} \tag{44}$$

Acknowledgments

This research is supported by the Office of Advanced Scientific Computing Research of the U.S. Department of Energy under contract number DE-AC02-05CH11231. This research used resources of the National Energy Research Scientific Computing Center (NERSC), a Department of Energy Office of Science User Facility.

References

[1] K. Atkinson, *Numerical integration on the sphere*, J. Austral. Math. Soc. B **23** (1982), no. 3, 332–347. MR Zbl

[2] J.-P. Berenger, *A perfectly matched layer for the absorption of electromagnetic waves*, J. Comput. Phys. **114** (1994), no. 2, 185–200. MR Zbl

[3] A. K. Chaniotis and D. Poulikakos, *High order interpolation and differentiation using B-splines*, J. Comput. Phys. **197** (2004), no. 1, 253–274. MR Zbl

[4] M. Frigo and S. G. Johnson, *FFTW: an adaptive software architecture for the FFT*, Proceedings of the 1998 IEEE International Conference on Acoustics, Speech and Signal Processing (Piscataway, NJ), vol. III: Digital signal processing, IEEE, 1998, pp. 1381–1384.

[5] R. Hockney, *Potential calculation and some applications*, Method. Comput. Phys. **9** (1970), 136–212.

[6] J. D. Lawson, *Generalized Runge–Kutta processes for stable systems with large Lipschitz constants*, SIAM J. Numer. Anal. **4** (1967), 372–380. MR Zbl

[7] B. Lo, V. Minden, and P. Colella, *A real-space Green's function method for the numerical solution of Maxwell's equations*, Commun. Appl. Math. Comput. Sci. **11** (2016), no. 2, 143–170. MR Zbl

[8] B. Marder, *A method for incorporating Gauss' law into electromagnetic PIC codes*, J. Comput. Phys. **68** (1987), no. 1, 48–55. Zbl

[9] J. J. Monaghan, *Extrapolating B-splines for interpolation*, J. Comput. Phys. **60** (1985), no. 2, 253–262. MR Zbl

[10] I. J. Schoenberg, *Contributions to the problem of approximation of equidistant data by analytic functions, A: On the problem of smoothing or graduation, a first class of analytic approximation formulae*, Quart. Appl. Math. **4** (1946), 45–99. MR Zbl

[11] A.-K. Tornberg and B. Engquist, *Numerical approximations of singular source terms in differential equations*, J. Comput. Phys. **200** (2004), no. 2, 462–488. MR Zbl

[12] J. Waldén, *On the approximation of singular source terms in differential equations*, Numer. Methods Partial Differential Equations **15** (1999), no. 4, 503–520. MR Zbl

[13] G. B. Whitham, *Linear and nonlinear waves*, Wiley-Interscience, New York, 1974. MR Zbl

Received April 29, 2018. Revised February 4, 2019.

BORIS LO: bt.lo@berkeley.edu
Applied Science and Technology, University of California, Berkeley, Berkeley, CA, United States
and
Lawrence Berkeley National Laboratory, Berkeley, CA, United States

PHILLIP COLELLA: pcolella@lbl.gov
Applied Numerical Algorithms Group, Computational Research Division, Lawrence Berkeley National Laboratory, Berkeley, CA, United States

Guidelines for Authors

Authors may submit manuscripts in PDF format on-line at the Submission page at msp.org/camcos.

Originality. Submission of a manuscript acknowledges that the manuscript is original and and is not, in whole or in part, published or under consideration for publication elsewhere. It is understood also that the manuscript will not be submitted elsewhere while under consideration for publication in this journal.

Language. Articles in CAMCoS are usually in English, but articles written in other languages are welcome.

Required items. A brief abstract of about 150 words or less must be included. It should be self-contained and not make any reference to the bibliography. If the article is not in English, two versions of the abstract must be included, one in the language of the article and one in English. Also required are keywords and subject classifications for the article, and, for each author, postal address, affiliation (if appropriate), and email address.

Format. Authors are encouraged to use LaTeX but submissions in other varieties of TeX, and exceptionally in other formats, are acceptable. Initial uploads should be in PDF format; after the refereeing process we will ask you to submit all source material.

References. Bibliographical references should be complete, including article titles and page ranges. All references in the bibliography should be cited in the text. The use of BibTeX is preferred but not required. Tags will be converted to the house format, however, for submission you may use the format of your choice. Links will be provided to all literature with known web locations and authors are encouraged to provide their own links in addition to those supplied in the editorial process.

Figures. Figures must be of publication quality. After acceptance, you will need to submit the original source files in vector graphics format for all diagrams in your manuscript: vector EPS or vector PDF files are the most useful.

Most drawing and graphing packages (Mathematica, Adobe Illustrator, Corel Draw, MATLAB, etc.) allow the user to save files in one of these formats. Make sure that what you are saving is vector graphics and not a bitmap. If you need help, please write to graphics@msp.org with details about how your graphics were generated.

White space. Forced line breaks or page breaks should not be inserted in the document. There is no point in your trying to optimize line and page breaks in the original manuscript. The manuscript will be reformatted to use the journal's preferred fonts and layout.

Proofs. Page proofs will be made available to authors (or to the designated corresponding author) at a Web site in PDF format. Failure to acknowledge the receipt of proofs or to return corrections within the requested deadline may cause publication to be postponed.

Communications in Applied Mathematics and Computational Science

vol. 14 no. 1 2019

1559-3940(2019)14:1;1-8