Journal of Computational Physics 507 (2024) 112979

Contents lists available at ScienceDirect

Journal of Computational Physics

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

A high-order finite difference method for moving immersed domain boundaries and material interfaces

James Gabbard, Wim M. van Rees *

Department of Mechanical Engineering, Massachusetts Institute of Technology, 77 Masschusetts Ave., Cambridge, 02139, MA, United States

ARTICLE INFO

Keywords:
Immersed method
High-order methods
Advection-diffusion
Moving boundaries
Runge-Kutta

ABSTRACT

We present a high-order sharp treatment of immersed moving domain boundaries and material interfaces, and apply it to the advection-diffusion equation in two and three dimensions. The spatial discretization combines dimension-split finite difference schemes with an immersed boundary treatment based on a weighted least-squares reconstruction of the solution, providing stable discretizations with up to sixth order accuracy for diffusion terms and third order accuracy for advection terms. The temporal discretization relies on a novel strategy for maintaining high-order temporal accuracy in problems with moving boundaries that minimizes implementation complexity and allows arbitrary explicit or diagonally-implicit Runge-Kutta schemes. The approach is broadly compatible with popular PDE-specialized Runge-Kutta time integrators, including low-storage, strong stability preserving, and diagonally implicit schemes. Through numerical experiments we demonstrate that the full discretization maintains high-order spatial and temporal accuracy in the presence of complex 3D geometries and for a range of boundary conditions, including Dirichlet, Neumann, and flux conditions with large jumps in coefficients.

1. Introduction

Moving domain boundaries and material interfaces are a hallmark of many single- and multiphysics problems, including multi-phase flows and fluid-structure interaction. For stationary surfaces it is feasible to generate a mesh that conforms to the problem geometry. However, for moving surfaces the cost of mesh adaptation and remeshing can become prohibitively high. This is particularly relevant in large-scale parallel simulations, which may require a significant communication overhead to maintain mesh quality and load balance.

Immersed methods offer an alternative where the surface geometry is incorporated within the discretization scheme, so that regular, structured grids can be used. In these schemes the surface is typically superimposed on a background Cartesian grid and the discretization is locally altered to account for the boundary or interface conditions prescribed on the immersed surface. The first immersed methods were developed with low order (second or lower) spatial accuracy [1–3], but recent progress has generated a variety of immersed methods for stationary geometries that treat sharply defined boundaries with third order spatial accuracy or higher. Notable examples include explicit dimension-split finite difference schemes in 2D [4–6] and in 3D [7,8]; compact finite difference schemes in 2D [9–13] and 3D [14,15]; inverse Lax-Wendroff (ILW) finite difference schemes for 2D hyperbolic PDEs [16–22]; cut-cell finite volume schemes in 2D [23,24] and 3D [25]; and implicit-mesh discontinuous Galerkin methods in 2D [26]

* Corresponding author.
E-mail address: wvanrees@mit.edu (W.M. van Rees).

https://doi.org/10.1016/j.jcp.2024.112979
Received 26 October 2023; Received in revised form 27 March 2024; Accepted 27 March 2024

Available online 2 April 2024

and 3D [27,28], enabled by the construction of high-order quadrature rules on implicitly defined cut cells [29,30]. Together these methods cover a variety of PDEs relevant in fluid and solid mechanics, including reactive gas dynamics [18], linear elastodynamics [28], the compressible Navier-Stokes equations [7,21,6], and the incompressible Navier-Stokes equations [9,11,13].

Despite rapid progress in the development of high-order immersed methods, very few of these methods allow for moving geometries. This is due in part to the difficulty of achieving high order temporal convergence near moving surfaces, which requires sharp immersed methods to address the issue of "freshly cleared" computational elements [31]. As an immersed domain boundary moves across a stationary Cartesian grid, grid points continuously enter and exit the problem domain. Exiting or "freshly covered" points can typically be removed from the discretization without complication. However, entering or "freshly cleared" points enter the domain with no predefined value or time history, making it difficult to apply a multistep or multistage time integrator to these systems. Strategies to address this issue fall into two broad categories, which are discussed separately in the following paragraphs: altering the temporal discretization to account for a discontinuity at each cell crossing, or constructing artificial values of the solution and its history that are compatible with standard time integrators.

One of the earliest sharp immersed methods to incorporate moving boundaries through direct alteration of the temporal discretization is the immersed interface method (IIM) for a prototypical 1D nonlinear PDE developed in [32]. The IIM is applied both to the spatial discontinuities at the interface and to temporal discontinuities in the solution when grid points cross the immersed interface. The authors estimate each of these crossing times and the magnitude of the discontinuity based on the interface motion and boundary conditions, and this information is used to construct a semi-implicit Adams-Bashforth/Crank-Nicolson (ABCN) time integrator that is modified to maintain first order accuracy at freshly cleared cells. The approach is extended to the 2D incompressible Navier-Stokes equations in [33] and applied to fluid-structure interaction with both elastic membranes and rigid solid boundaries. Brehm and Fasel [34] replace the Adams-Bashforth method with a specialized semi-implicit discretization of nonlinear convective terms. By using the interface boundary condition to obtain higher derivatives of the boundary motion from higher derivatives of the solution, the authors are able to maintain second order accuracy at freshly-cleared cells. For explicit time integrators, Xu and Wang [35] apply the IIM to treat freshly cleared cells in each stage of a four stage Runge-Kutta scheme, treating each stage as its own Euler-like time step. The method exhibits first order temporal convergence at freshly cleared cells, with marginally lower error than time integration that completely ignores the temporal discontinuities. While these approaches successfully allow for sharp moving interfaces, they are all specialized to a particular integration scheme and are difficult to extend to higher-order accuracy or PDEs with more complex boundary conditions.

An alternative strategy for freshly cleared cells is to generate either an initial value or an artificial time history for uncovered points, which can be achieved through spatial interpolation, spatial extrapolation, or a combination of the two. Here we categorize these methods as interpolation-based if initial values are prescribed to points after they enter the problem domain, and extrapolation-based if a time history is assigned to points while they lie outside of the problem domain. The interpolation-based approach was developed in [31,36] for moving-interface diffusion simulations and the 2D incompressible Navier-Stokes equations, in which the authors determine solution values at freshly cleared points based on a second-order spatial interpolation that incorporates an interface boundary condition. Similar interpolation-based approaches for the 3D Navier-Stokes equations are developed in [37,38] and applied to biologically-inspired unsteady 3D flows in [39,40], as well as to wind-turbine simulations with adaptive mesh refinement in [41]. For 3D compressible flows the interpolation-based freshly cleared cell treatment is extended to higher-order interpolants and higher-order RK integrators in [42], and an analysis presented therein indicates that the order of accuracy of the freshly cleared cell treatment is largely determined by the order of the spatial interpolation. The method is applied to large-scale 3D aeroacoustics, and to simulations to fluid-structure interaction with thin compliant structures in [43]. A variation of this method that mixes spatial interpolation and extrapolation is introduced in [44] for simulations of incompressible 3D turbulence with moving boundaries. In this method all cells inside the fluid domain and adjacent to the boundary are considered "forcing points". They are not included in the three-stage low-storage Runge-Kutta (LSRK) scheme used in the interior, and they are assigned spatially interpolated field values at each stage. However, they also receive right hand side values calculated with a ghost-fluid approach, even though these values are not required for time integration with stationary boundaries. When a forcing point enters further into the fluid domain due to boundary motion, these extra right hand side values form a time history compatible with the LSRK scheme. A related approach based on spatial extrapolation alone is developed in [17] for the compressible Euler equations, with the state variables being extrapolated outside of the computational domain at each stage of a three-stage strong stability preserving Runge-Kutta scheme. In a similar spirit, [16] discretizes a variety of linear time-dependent PDEs by extrapolating the state variables to multiple layers of ghost points at the start of each step, and solving the governing equations on an extended domain that shrinks by one ghost layer at each Runge-Kutta stage. Boundary motion is accounted for through additional ghost layers, so that the dynamics extend to any point which may enter the domain during a given time step.

While previous efforts have led to a variety of successful treatments of freshly cleared cells, none of these have demonstrated greater than second order accuracy in both space and time for the 3D advection-diffusion equation with moving boundaries. We attribute this to the relatively small number of high-order sharp immersed methods that have been applied in 3D simulations, even with stationary boundaries. Additionally, there is little work that quantifies the temporal errors associated with high order freshly cleared cell treatments, and few proposed treatments that are independent of a specific choice of time integrator. In this work we extend previous freshly cleared cell treatments based on extrapolation to high-order spatial discretizations and arbitrary high-order explicit RK schemes, and quantify the magnitude of the associated errors through systematic numerical experimentation. Specifically, we introduce a high-order sharp immersed spatial discretization that allows for third order accurate advection terms and up to sixth order accurate diffusion terms in domains with complex immersed bodies. These discretizations are based on high-order weighted least squares reconstructions near the immersed body, which can also be used to extrapolate the PDE solution to

points outside of the computational domain. We use this strategy to extrapolate both the solution and its time derivative during each stage of a Runge-Kutta time integration scheme, providing an artificial time history for freshly cleared cells. This history is sufficient to maintain the accuracy of simulations with arbitrary explicit or diagonally implicit RK schemes, including low storage (LS) and strong stability preserving (SSP) integrators, so long as the time step is chosen to satisfy a CFL constraint based on the interface velocity. Through 2D and 3D simulations of the advection-diffusion equation with moving boundaries, we demonstrate that our sharp immersed discretizations converge at third order or higher in the L_∞ norm, accurately predict surface quantities, and can resolve thin boundary layers on moving interfaces, all of which have traditionally been challenging for immersed methods.

The remainder of this work is organized as follows. Section 2 reviews the high order sharp immersed spatial discretizations developed in [8] and their application to model hyperbolic and parabolic PDEs with stationary boundaries. Section 3 outlines a novel method for the time integration of immersed discretizations with moving boundaries that is compatible with general explicit Runge-Kutta schemes and diagonally implicit Runge-Kutta schemes. Section 4.1 quantifies the error introduced by this moving boundary treatment through extensive numerical experiments with 1D systems, while the remainder of section 4 focuses on convergence results and applications involving 2D and 3D geometries. We draw conclusions in section 5.

2. Spatial discretizations

2.1. Sharp immersed finite difference schemes

In this work spatial discretizations are based on the high-order sharp immersed method outlined in [8], which we summarize here. We denote the computational domain $\Omega \subset \mathbb{R}^d$, with $d \in [1,2,3]$ the dimensionality of the problem. Let $\Gamma(t)$ be a closed moving surface immersed in Ω at time t, defined by the zero level set of a smooth function $\phi(\mathbf{x},t)$. Here and below we label points on the surface by their Cartesian coordinates $\mathbf{s} \in \Gamma(t) \subset \mathbb{R}^d$, and use the more general label $\mathbf{x} \in \Omega \in \mathbb{R}^d$ to refer any point in the computational domain. We divide Ω into two regions based on the sign of the level-set function,

$$\Omega^+(t) = \{\mathbf{x} \in \Omega \mid \phi(\mathbf{x},t) > 0\} \quad \text{and} \quad \Omega^-(t) = \{\mathbf{x} \in \Omega \mid \phi(\mathbf{x},t) \leq 0\}. \tag{1}$$

For $\mathbf{s} \in \Gamma(t)$ let $\mathbf{n}(\mathbf{s},t) = \nabla\phi(\mathbf{s},t)/\|\nabla\phi(\mathbf{s},t)\|$ be the normal vector pointing into $\Omega^+(t)$ and away from $\Omega^-(t)$. Likewise, for any function $u(\mathbf{x},t)$ defined in a neighborhood of $\mathbf{s} \in \Gamma(t)$, let $[u](\mathbf{s},t) = u^+(\mathbf{s},t) - u^-(\mathbf{s},t)$ indicate the jump in u across the interface at $\mathbf{s}$; here superscripts indicate quantities defined on the side of the interface adjacent to Ω^+ or Ω^-. While the level set formulation does not explicitly identify material points on the surface, the normal component of the surface velocity $\mathbf{v}_b(\mathbf{s},t)$ can be computed directly from the level set via $\mathbf{n}(\mathbf{s},t) \cdot \mathbf{v}_b(\mathbf{s},t) = -\partial_t\phi(\mathbf{s},t)/\|\nabla\phi(\mathbf{s},t)\|$.

We discretize partial derivative operators on a d-dimensional Cartesian grid with coordinates $\{x_i\}_{i=1}^d$, unit vectors $\{\hat{\mathbf{e}}_i\}_{i=1}^d$, and uniform grid spacing Δx along each dimension. Spatial derivatives are discretized with non-conservative dimension-split finite difference schemes of the form

$$\frac{\partial u(\mathbf{x})}{\partial x_i} \approx \sum_{j=-\ell}^{r} a_j u(\mathbf{x} + j\Delta x \hat{\mathbf{e}}_i), \tag{2}$$

where $\{a_j\}_{j=-\ell}^r$ are 1D finite difference coefficients with ℓ points to the left of the evaluation point and r points to the right of the evaluation point. On the outer boundaries of the computational domain, which are assumed to align with the Cartesian grid, we supplement Eq. (2) with standard boundary treatments that are appropriate for the given PDE. For grid points near the immersed surface, one or more of these 1D finite difference stencils will cross Γ at a control point $\mathbf{x}_c$, defined to be the intersection between the grid line and the immersed surface. When this occurs the regular interior stencil becomes invalid, and ghost points must be constructed for the difference scheme.

To do so, we extrapolate field values across the surface using a weighted least squares approach. Let $\mathcal{X} = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n\} \subset \mathbb{R}^d$ be a set of n interpolation points from a Cartesian grid with grid spacing Δx, and let $u(\mathbf{x})$ be a scalar function defined at the grid points. For a given set of positive weights $\{w_i\}_{i=1}^n$, the weighted least squares interpolant $p(\mathbf{x})$ of degree k is defined by the minimization

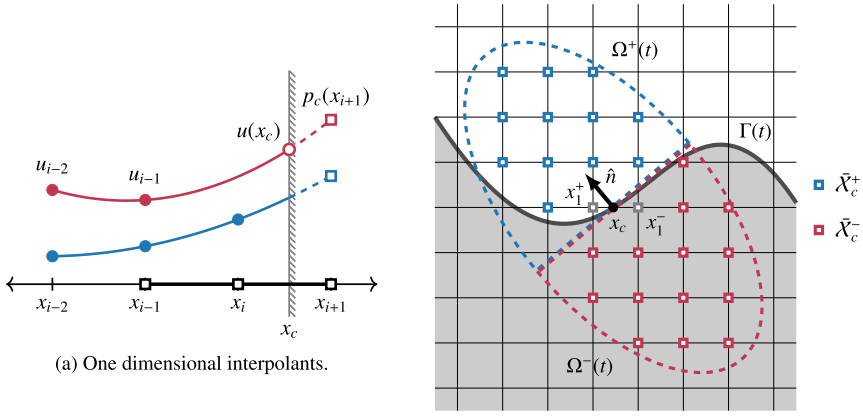
$$p(x) = \underset{q(\mathbf{x}) \in \mathcal{P}_k}{\arg\min} \sum_{\mathbf{x}_i \in \mathcal{X}} w_i \left[q(\mathbf{x}_i) - u(\mathbf{x}_i)\right]^2, \tag{3}$$

where $\mathcal{P}_k$ is the set of polynomials in d variables with degree less than or equal to k. So long as any basis for $\mathcal{P}_k$ is linearly independent over $\mathcal{X}$, the interpolant is unique and its m-th derivative with respect to coordinate x_j satisfies

$$\partial_{x_j}^m p(\mathbf{x}_e) = \partial_{x_j}^m u(\mathbf{x}_e) + \mathcal{O}\left(\Delta x^{k+1-m}\right) \tag{4}$$

for any evaluation point $\mathbf{x}_e$. Because the interpolant is linearly related to the input data, there exists a set of stencil coefficients $\{s_i\}_{i=1}^n$ associated with each derivative $\partial_{x_j}^m$ such that $\partial_{x_j}^m p(\mathbf{x}_e) = \sum_{i=1}^n s_i u(\mathbf{x}_i)$.

To construct this weighted least squares interpolant in a sharp immersed PDE discretization, we assign two sets of interpolation points to each control point on the immersed surface. For 1D discretizations, we construct the set $\mathcal{X}_c^+$ containing the n closest points to the interface in Ω^+, and the equivalent set $\mathcal{X}_c^-$ containing the n closest points in Ω^-, as shown in Fig. 1a. We note that a degree $n-1$ interpolant can interpolate these n data points exactly, and no weights $\{w_i\}$ are required. For multidimensional discretizations, we construct the set $\mathcal{X}_c^+$ containing all available grid points in Ω^+ that fall within a half ellipsoid centered on the control point $\mathbf{x}_c$



(a) One dimensional interpolants.

(b) Multidimensional interpolants.

Fig. 1. (a) The 1D IIM boundary treatment outlined in section 2.1. For a Dirichlet boundary condition (red), a quadratic polynomial $p_c(x)$ interpolates a solution $u(x)$ at $\bar{\mathcal{X}}_c = \{x_{i-2}, x_{i-1}\}$ and at x_c, and the value $p(x_{i+1})$ is used when applying a three-point centered finite difference (black). When no boundary condition is prescribed (blue), $p_c(x)$ interpolates the solution at $\mathcal{X}_c = \{x_{i-2}, x_{i-1}, x_i\}$ instead. (b) The half-elliptical stencil used to construct multidimensional polynomial interpolants in this work. (For interpretation of the colors in the figure(s), the reader is referred to the web version of this article.)

with a major radius r_n along the surface normal $\mathbf{n}_c$ and a minor radius r_t along any tangential direction, as shown in Fig. 1b. For the negative side of the surface we construct the equivalent set $\mathcal{X}_c^-$ using points from Ω^- and a half ellipsoid with the same normal and tangential radii. For 3D geometries, the radii r_n and r_t are selected so that the resulting least-squares fit is well-posed whenever the surface is smooth and the principal curvatures κ_1, κ_2 of the surface satisfy $|\kappa_i \Delta x| < 1/4$, indicating a radius of curvature greater than four grid points for any nonconvex feature. For 2D geometries, the constraint reduces to $|\kappa \Delta x| < 1/4$ for the scalar curvature κ. For a degree $k-1$ interpolant the choices generally correspond to $r_n \approx k\Delta x$ and $r_t \approx k\Delta x/2$, so that the resulting interpolation stencils resemble a one-sided stencil of order k in the normal direction and a centered stencil of order k in the tangential direction(s). If the control point $\mathbf{x}_c$ is included in the interpolation stencil, we omit the grid point in $\mathcal{X}_c^-$ or $\mathcal{X}_c^+$ located closest to $\mathbf{x}_c$, as these two points can become arbitrarily close. In 1D discretizations this is necessary for a well-conditioned interpolation procedure, and in multiple dimensions we have found that the omission is necessary to ensure stability for PDEs with an advection terms, as discussed in section 2.2. Letting $\mathbf{x}_1^-$ and $\mathbf{x}_1^+$ be the closest interpolation points in $\mathcal{X}_c^-$ and $\mathcal{X}_c^+$ respectively, the resulting sets of grid points will be notated $\bar{\mathcal{X}}_c^+ \equiv \mathcal{X}_c^+ \setminus \{\mathbf{x}_1^+\}$ and $\bar{\mathcal{X}}_c^- \equiv \mathcal{X}_c^- \setminus \{\mathbf{x}_1^-\}$ below.

For an sharp immersed PDE discretization, the high-order polynomial fits constructed at each control point are used to maintain the accuracy of the regular interior difference scheme across an immersed boundary or interface. For example, assuming that a Dirichlet boundary condition for $u(\mathbf{x})$ is prescribed on the boundary, a polynomial approximation $p_c(\mathbf{x})$ to the function $u(\mathbf{x})$ can be constructed using a least squares fit that incorporates the boundary condition at a single control point $\mathbf{x}_c$ and the domain values $\{u(\mathbf{x}_i) \mid \mathbf{x}_i \in \bar{\mathcal{X}}_c^+\}$. Each 1D finite difference stencil that intersects the boundary at $\mathbf{x}_c$ can then be applied to the extended function

$$u_c(\mathbf{x}) = \begin{cases} u(\mathbf{x}), & \mathbf{x} \in \Omega^+ \\ p_c(\mathbf{x}), & \mathbf{x} \in \Omega^- \end{cases}. \tag{5}$$

Provided that $p_c(\mathbf{x})$ is a sufficiently high order interpolant, this procedure maintains the accuracy of the interior scheme for stencils that cross the boundary at $\mathbf{x}_c$. In general the least-squares interpolant does not exactly interpolate the prescribed boundary condition at $\mathbf{x}_c$, but still enforces this boundary condition with high-order accuracy. Throughout this work, when the interior scheme has order of accuracy P and each $p_c(\mathbf{x})$ has degree $k-1$, we refer to the full IIM discretization as an (P, k) scheme for brevity.

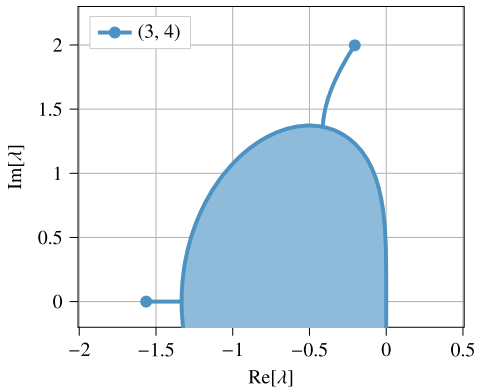
2.2. Boundary conditions for advection and diffusion terms

In this work we focus on model initial-boundary value problems that form the building blocks for many common fluid systems: linear hyperbolic PDEs with Dirichlet boundary conditions, parabolic PDEs with Dirichlet or Neumann boundary conditions, and parabolic PDEs with discontinuous coefficients. Each requires a distinct algorithm for the proper enforcement of boundary or interface conditions.

For hyperbolic PDEs, we consider a (3, 4) IIM discretization of the linear advection equation $\partial_t u + \mathbf{v} \cdot \nabla u = 0$ on the irregular domain $\Omega^+(t)$ with interface $\Gamma(t)$. The interior scheme is a third order upwind discretization applied dimension-by-dimension, with each dimension using the 1D stencil

$$\left.\frac{\partial u}{\partial x}\right|_i = \begin{cases} \frac{1}{6\Delta x}\left(u_{i-2} - 6u_{i-1} + 3u_i + 2u_{i+1}\right), & v_i \geq 0, \\ \frac{1}{6\Delta x}\left(-2u_{i-1} - 3u_i + 6u_{i+1} - u_{i+2}\right), & v_i < 0. \end{cases} \tag{6}$$

A Dirichlet boundary condition value is only required at control points where the relative velocity on the boundary acts as an inflow, so that $\mathbf{n}(\mathbf{x}_c) \cdot (\mathbf{v}(\mathbf{x}_c) - \mathbf{v}_b(\mathbf{x}_c)) > 0$. At these points we construct fourth order polynomial interpolants using data from the points in $\bar{\mathcal{X}}_c$



(a) Spectrum of the (3,4) 1D IIM advection discretization.

Integrator	Reference	C
RK2	(Heun's Method)	0.52
LSRK(3,3)	[45, Eq. 11]	1.07
SSPRK(3,3)	[46, Eq. 2.18]	1.07
LSRK(5,4)	[47, Solution 3]	1.69
SSPRK(5,3)	[48, Table 1]	1.57
Vern7	[49, Table 6]	2.36

(b) CFL constraints for the 1D (3,4) IIM advection discretization.

Fig. 2. (a) The spectrum of the (3,4) IIM advection spatial discretization presented in section 2.2, for unit grid spacing and unit velocity on the semi-infinite domain $x \in [\alpha, \infty]$. The domain is discretized using grid points $x_j = j\Delta x$ and an IIM boundary treatment at the inflow boundary $x = \alpha$. The eigenvalues of the third order upwind finite difference scheme used away from the inflow boundary fall on the border of the shaded region. The "whiskers" protruding from this region are the loci of all additional eigenvalues introduced by the boundary treatment as the position of the boundary is varied over the interval $0 \leq \alpha \leq \Delta x$. (b) A list of CFL criteria that must be satisfied to guarantee stability when the (3,4) IIM advection discretization is paired with each of the integrators referenced in this work. At this CFL the entire spectrum shown in the left panel is contained within the integrator's linear stability region.

and the prescribed boundary condition at $\mathbf{x}_c$. For outflow control points satisfying $\mathbf{n}(\mathbf{x}_c) \cdot (\mathbf{v}(\mathbf{x}_c) - \mathbf{v}_b(\mathbf{x}_c)) \leq 0$ there is no prescribed boundary condition, and the interpolants are constructed using data from the set $\mathcal{X}_c$ only. For 1D discretizations, the least squares fitting procedure reduces to 1D polynomial interpolation, and this third order boundary treatment is provably stable when paired with explicit RK schemes and a constraint on the CFL number $C_v = |v\Delta t/\Delta x|$. The analysis is presented in [8] and builds on the methodology established in [20] for stability analyses of immersed methods for hyperbolic PDEs. The spectrum of the 1D difference scheme and boundary treatment is illustrated in Fig. 2, and is accompanied by precise CFL limits for the time integrators used in this work.

To ensure stability in 2D and 3D advection discretizations, we limit the multidimensional CFL number $C_v = \max_\Omega \|\mathbf{v}(\mathbf{x})\|_1 \Delta t/\Delta x$, and choose weights for each least squares fit that decay rapidly away from the control point as suggested in [23]. For the half-elliptical interpolation stencils used in this work, the weight function $w(\mathbf{x}) = \max[d(\mathbf{x}, \mathbf{x}_c), 0.7]^{-6}$ with elliptical distance function

$$d(\mathbf{x}, \mathbf{x}_c) = \sqrt{(\mathbf{x} - \mathbf{x}_c)^T \boldsymbol{\Sigma}_c (\mathbf{x} - \mathbf{x}_c)} \quad \text{with} \quad \boldsymbol{\Sigma}_c = \frac{\mathbf{n}_c \mathbf{n}_c^T}{r_n^2} + \frac{\mathbf{I} - \mathbf{n}_c \mathbf{n}_c^T}{r_t^2} \tag{7}$$

has been shown to produce stable 2D and 3D boundary treatments for third-order discretizations of the advection equation with Dirichlet boundary conditions [8]. For higher order interior schemes, including the standard fourth order centered and fifth order upwind discretizations, similar boundary treatments based on least squares fits with higher than third order accuracy become unstable. While there are existing stable high-order boundary treatments for these schemes in 2D, they are limited to convex geometries [5] or require considerable complexity when extended to 3D [18,20]. As a result, we limit our attention to third order upwind advection schemes in this work.

As a model parabolic PDE, we consider the scalar diffusion equation $\partial_t u = \beta \nabla^2 u$ defined on $\Omega^+(t)$ with bounding surface $\Gamma(t)$. The Laplacian operator is discretized with standard centered finite difference stencils of fourth or sixth order. For simulations with Dirichlet boundary conditions, the correction procedure described in the previous section is applied to all stencils that cross the immersed surface. For the case of diffusion on the full domain $\Omega(t)$ with a diffusivity field $\beta(\mathbf{x})$ that is piecewise constant and discontinuous across the immersed interface $\Gamma(t)$, the following jump conditions hold

$$\begin{aligned} [u(\mathbf{s}, t)] &= j_0(\mathbf{s}, t) \quad \text{on} \quad \Gamma, \\ \left[\beta \partial_n u(\mathbf{s}, t)\right] &= j_1(\mathbf{s}, t) \quad \text{on} \quad \Gamma. \end{aligned} \tag{8}$$

These jump conditions are enforced by determining equivalent boundary solution values $u^\pm(\mathbf{x}_c)$ at each control point $\mathbf{x}_c$, and subsequently treating the two domains $\Omega^+(t)$ and $\Omega^-(t)$ as separate domains with prescribed Dirichlet boundary conditions given by $u^\pm(\mathbf{x}_c)$. The equivalent boundary solution values are determined from a direct discretization of Eq. (8) using two sets of stencil coefficients. The coefficients s_c^+ and $\{s_i^+\}$ map the boundary value $u^+(\mathbf{x}_c)$ and domain values $\left\{u(\mathbf{x}_i) \,|\, \mathbf{x}_i \in \tilde{\mathcal{X}}_c^+\right\}$ to the normal derivative $\partial_n u^+(\mathbf{x}_c)$, approximated by the normal derivative of a least squares polynomial fit. The second set of coefficients s_c^- and $\{s_i^-\}$ is designed analogously to map data from Ω^- to the normal derivative $\partial_n u^-(\mathbf{x}_c)$. The boundary values $u^\pm(\mathbf{x}_c)$ can then be determined from the linear system

$$u^+(\mathbf{x}_c) - u^-(\mathbf{x}_c) = j_0(\mathbf{x}_c) \quad \text{and} \quad \beta^+ \left(s_c^+ u^+(\mathbf{x}_c) + \sum_{\mathbf{x}_i \in \tilde{\mathcal{X}}_c^+} s_i^+ u^+(\mathbf{x}_i) \right) - \beta^- \left(s_c^- u^-(\mathbf{x}_c) + \sum_{\mathbf{x}_i \in \tilde{\mathcal{X}}_c^-} s_i^- u^-(\mathbf{x}_i) \right) = j_1(\mathbf{x}_c). \tag{9}$$

For consistency, a single pair of polynomial fits is used both to approximate the normal derivatives and to construct ghost values for difference stencils that cross the boundary at each control point.

For the scalar diffusion equation on $\Omega^+(t)$ with Neumann boundary conditions $\beta\partial_n u(\mathbf{s},t) = g(\mathbf{s},t)$ our approach is analogous to Eq. (9) with vanishing diffusivity β^-. After dropping superscripts the boundary value $u(\mathbf{x}_c)$ at each control point can be approximated by

$$u(\mathbf{x}_c) = \frac{1}{s_c}\left(\frac{g(\mathbf{x}_c)}{\beta} - \sum_{x_i \in \tilde{\mathcal{X}}_c} s_i u(\mathbf{x}_i)\right). \tag{10}$$

Once the boundary value at each control point is computed, $\Omega^+(t)$ is treated as a domain with a prescribed Dirichlet boundary condition. As with jump conditions, a single polynomial fit is used both to construct ghost points for the difference stencils crossing the boundary at $\mathbf{x}_c$ and to determine the stencil coefficients s_c and $\{s_i\}$ that approximate the normal derivative on the boundary.

For all three type of boundary conditions for the diffusion equation described above, the weights used to construct each weighted least squares interpolant are chosen to be uniform ($w_i = 1$). For a (P,k) IIM discretization of a parabolic PDE, the truncation error at the boundary is of order $k-2$ for all three types of boundary conditions. However, due to elliptic regularity effects [50], spatial convergence of order P for the solution in the L_∞ norm requires only a $(P, P+1)$ discretization for jump conditions or Neumann boundary conditions [8]. For Dirichlet boundary conditions, a (P,P) discretization provides P-th order convergence [8], but we have observed that a $(P,P+1)$ discretization often leads to lower error magnitudes at the same resolution. As a result, we use $(P,P+1)$ discretizations for all parabolic terms in this work, leading to truncation errors of order P in the interior and $P-1$ on the boundary.

3. Temporal discretizations

3.1. Runge-Kutta time integration with stationary boundaries

Before discussing moving boundaries and interfaces, we briefly review the formulation and notation of Runge-Kutta time integration for systems with stationary boundaries. After discretizing a time-dependent PDE in space we obtain a system of the form

$$\dot{u} = f(u,t), \quad u|_{t=0} = u_0, \tag{11}$$

where u represents the state of the discretized system, u_0 is an initial condition, and $f(u,t)$ is the discretized right hand side of the PDE. For PDEs with stationary boundaries we take $u \in \mathbb{R}^n$, where n is a constant number of discrete degrees of freedom, so that Eq. (11) is a finite dimensional ODE that can be integrated with standard multistep or multistage time integration schemes. In this work our focus is on Runge-Kutta schemes, which can be expressed in the modified Shu-Osher form [51,52]

$$\begin{aligned} u^{(i)} &= v_i u^n + \textstyle\sum_{j=1}^{s} \alpha_{ij} u^{(j)} + \Delta t \sum_{j=1}^{s} \beta_{ij} f(u^{(j)}, t^{(j)}) \quad \text{for} \quad 1 \le i \le s+1, \\ u^{n+1} &= u^{(s+1)}. \end{aligned} \tag{12}$$

Here Δt is the length of the time step, s is the number of stages, u^n represents the state at the n-th time step, $u^{(j)}$ represents the state at the j-th stage, v_i is an $s+1$ element vector, and $\{\alpha_{ij}, \beta_{ij}\}$ are $(s+1)\times s$ matrices. The stage times are given by $t^{(i)} \equiv t^n + c_i \Delta t$, where the abscissae c_i are obtained by integrating the auxiliary ODE $\dot{t} = 1$. This representation is not unique, and any implicit or explicit Runge-Kutta scheme in this form can be re-expressed uniquely as Butcher tableau under mild assumptions on irreducibility [53]. However, the modified Shu-Osher form is useful for expressing PDE-specialized Runge-Kutta schemes such as strong stability preserving (SSP) or low-storage (LS) schemes. Any RK scheme presented as a Butcher tableau with coefficients $\{a_{ij}, b_i\}$ for $1 \le i,j \le s$ can be placed in modified Shu-Osher form by defining

$$v_i = 1, \quad \alpha_{ij} = 0, \quad \text{and} \quad \beta_{ij} = \begin{cases} a_{ij} & 1 \le i \le s \\ b_j, & i = s+1 \end{cases}, \tag{13}$$

with stage times $c_i = \sum_{j=1}^{s} a_{ij}$. For diagonally implicit schemes α_{ij} and β_{ij} are lower triangular, and for explicit schemes they are strictly lower triangular.

3.2. Runge-Kutta time integration with moving boundaries

For systems with moving boundaries, time integration becomes more complex. As computational elements enter or exit the problem domain, both the dimension of the state vector u and the right hand side $f(u,t)$ are constantly changing, and the discretized system cannot be viewed simply as a system of ODEs to be integrated with the method of lines. “Freshly-covered” points which exit the problem domain are excluded from the Runge-Kutta update while they lie outside of the domain, and do not influence the temporal discretization. However, “freshly-cleared” points which enter the problem domain do so without an initial value or time history, and some special treatment is necessary to provide these. Here we present an algorithm that is designed to preserve as much as possible the flexibility of the method of lines, avoiding intrusive alterations to the discretization of $f(u,t)$ and retaining compatibility with a wide range of established RK integrators. We begin by discussing explicit integration schemes for PDEs posed

on an irregular domain, then expand this algorithm to diagonally implicit integrators as well as PDEs defined on both sides of an immersed interface and coupled through interface boundary conditions.

Elements in the state and right hand side evaluated at grid point $\mathbf{x}_k$ at the i-th stage are denoted as $u^{(i)}(\mathbf{x}_k)$ and $f^{(i)}(\mathbf{x}_k)$, respectively. We will use a zero superscript (for example, $u^{(0)}$) to refer to quantities at time t^n whenever convenient. Following the notation of section 2, grid points in $\Omega^+(t^{(i)})$ are referred to as *active* during the i-th stage, while those in $\Omega^-(t^{(i)})$ are referred to as *inactive*. The complete vectors $u^{(i)}$ and $f^{(i)}$ will be used to indicate a collection of values from all points on the grid, both active and inactive, with the understanding that $f(u,t)$ ignores the input values of inactive points and assigns a zero value to inactive output points. During the i-th stage of any time step a standard sharp immersed discretization as discussed in section 2 can be used to evaluate the right hand side $f^{(i)}$ at all active points, provided that the state value $u^{(i)}$ is known at all active points. Applying the standard Runge-Kutta update defined in Eq. (12) to calculate $u^{(i)}$ at all active points in the domain leads to two possibilities. For currently active points $\mathbf{x}_k$ that have also been active during all prior stages the Runge-Kutta update coincides with the standard method of lines procedure, and no further modification is necessary. For points $\mathbf{x}_k$ that were inactive during a prior stage j, the values $u^{(j)}(\mathbf{x}_k)$ and $f^{(j)}(\mathbf{x}_k)$ needed for the Runge-Kutta update are missing. This is the freshly cleared cell problem, which occurs in the stage that a cell is uncovered and in all subsequent stages due to the time history needed by multistage Runge-Kutta integrators.

We circumvent the freshly-cleared cell problem by calculating both the missing state $u^{(j)}(\mathbf{x}_k)$ and the missing right hand side $f^{(j)}(\mathbf{x}_k)$ at the inactive point $\mathbf{x}_k$ through a spatial extrapolation from active point values at the associated stage time $t^{(j)}$. We note that many existing methods use a similar extrapolation of the state $u^{(j)}$ to allow moving boundaries, but the added extrapolation of the right hand side $f^{(j)}$ allows this procedure to be applied to any explicit multistage integration scheme that can be expressed in Shu-Osher form. More formally, we define the set

$$\mathcal{N}^{(i)} = \{\mathbf{x}_k \in \Omega^-(t^{(i)}) \mid \mathbf{x}_k \text{ has a neighbor in } \Omega^+(t^{(i)})\} \tag{14}$$

which contains all inactive points that have an active neighbor during the i-th stage. Here "neighbor" indicates two points that fall on the same grid line and are separated by distance Δx. While we do not do so here, the set $\mathcal{N}^{(i)}$ can be broadened to include points that fall further from the active domain, which may ease some of the restrictions on boundary motions that are discussed in section 3.5. We also define a zeroing operator $Z^{(i)}[\cdot]$ for the i-th stage which sets the value of the argument on all points in $\mathcal{N}^{(i)}$ to zero, as well as an extension operator $E^{(i)}[\cdot]$ that overwrites the value of the argument on all points in $\mathcal{N}^{(i)}$ with an extrapolation from the solution on $\Omega(t^{(i)})$. Each step begins with an extrapolation of the initial state, so that $u^{(0)} = E^{(0)}[u^n]$. In subsequent stages the state $u^{(i)}$ is assembled via the standard Runge-Kutta update at all points, and the zeroing operator is applied to remove any extraneous information at inactive points. The right hand side $f^{(j)}$ is then calculated as usual, and the extension operator is applied to both the state $u^{(i)}$ and the right hand side $f^{(i)}$. When the time step is restricted by a body CFL criterion, as discussed below, the extrapolation ensures that any point which crosses from Ω^- to Ω^+ does so with a complete time history that can be used in subsequent Runge-Kutta updates. The complete moving-boundary explicit Runge-Kutta algorithm is presented in Algorithm 1 and illustrated in Fig. 3a.

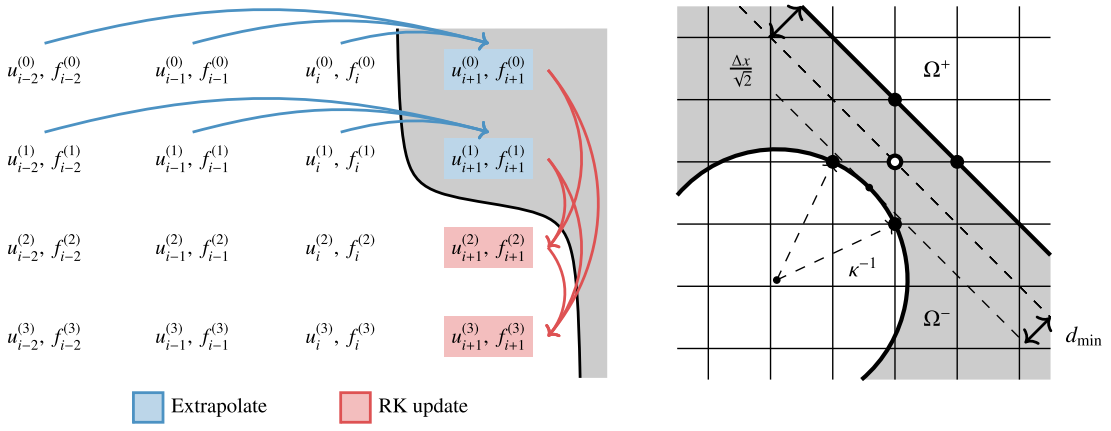
Algorithm 1 Explicit Runge-Kutta integration with moving boundaries.

$u^{(0)} = E^{(0)}[u^n]$
for $1 \le i \le s$ **do**
  $\tilde{u}^{(i)} = Z^{(i)}\left[v_i u^{(0)} + \sum_{j=1}^{s} \alpha_{ij} u^{(j)} + \Delta t \sum_{j=1}^{s} \beta_{ij} f^{(j)}\right]$
  $f^{(i)} = E^{(i)}\left[f(\tilde{u}^{(i)}, t^{(i)})\right]$
  $u^{(i)} = E^{(i)}\left[\tilde{u}^{(i)}\right]$
end for
$u^{n+1} = Z^{(s+1)}\left[v_{s+1} u^{(0)} + \sum_{j=1}^{s} \alpha_{s+1,j} u^{(j)} + \Delta t \sum_{j=1}^{s} \beta_{s+1,j} f^{(j)}\right]$

We note that the precise form of the extrapolation operators $E^{(i)}[\cdot]$ is not fixed, and that many sharp immersed methods for stationary bodies already involve a ghost-point extrapolation procedure that can be reused in a moving boundary time integration algorithm. In this work we choose to extrapolate by reusing the existing interpolants, and further incorporate boundary conditions as much as possible. For a pure diffusion problem, this means we reuse the uniformly weighted least-squares interpolant that incorporates the boundary condition to extrapolate the solution u, and compute a new interpolant with the same weighting scheme and order to extrapolate the right hand side f without a boundary condition. For advection and advection-diffusion problems, we extrapolate using the weighted least-squares interpolant from the advection term, again using any imposed boundary conditions on u and without boundary conditions for f. For outflow boundaries, where no boundary condition on u is imposed, we use the advection-based interpolant to extrapolate both the solution and right hand side with no boundary information. For points with multiple neighbors in Ω^+, each neighbor contributes an extrapolated value and the results are averaged as in [54]. While this averaging strategy may fail near sharp corners or thin, under-resolved features, it is an appropriate choice for smooth and well-resolved geometries. From a software perspective, these extrapolations can be implemented as a callback which runs at the end of each integration stage in an established time integration package.

3.3. Diagonally-implicit integrators with moving boundaries

The Shu-Osher form given in Eq. (12) also encompasses standard diagonally implicit Runge-Kutta (DIRK) schemes and low-storage DIRK schemes such as those presented in [55]. For these diagonally implicit schemes, we begin each step by extrapolating the state u^n, and thereafter each stage consists of the update



(a) Moving boundary time integration algorithm for immersed domain boundaries. (b) Geometry of the boundary CFL constraints in Eq. 20.

Fig. 3. (a) From section 3.2: an illustration of Algorithm 1 applied to a 1D domain with four grid points and a time integrator with three stages. For the initial state and first stage, the state and right hand side are extrapolated to point x_{i+1} which lies outside of the problem domain. When this point enters the domain during the second stage, its value is calculated using the standard RK update and the extrapolated time history. For the third stage the RK update relies on a mix of extrapolated and calculated values. (b) From section 3.5: constraints on the body CFL can be derived from the minimum distance between a boundary with curvature κ and a grid point with all of its nearest neighbors on the same side of the interface. For a planar boundary the closest approach is $\Delta x/\sqrt{2}$ (upper right), while for a curved boundary the closest approach becomes smaller with increasing boundary curvature (lower left). The 3D problem can be solved analogously for both planar and spherical boundaries.

$$(1-\alpha_{ii})u^{(i)} - \Delta t\beta_{ii} f(u^{(i)},t^{(i)}) = r^{(i)}, \quad \text{with} \quad r^{(i)} \equiv v_i u^n + \sum_{j=1}^{i-1} \alpha_{ij} u^{(j)} + \Delta t \sum_{j=1}^{i-1} \beta_{ij} f(u^{(j)},t^{(j)}). \tag{15}$$

Here the residual $r^{(i)}$ is a linear combination of previous stages and right hand side evaluations, which is assembled in a way analogous to the state $u^{(i)}$ for an explicit integrator. Once $r^{(i)}$ is assembled for all point inside the problem domain, the system $(1-\alpha_{ii})u^{(i)} - \Delta t\beta_{ii} f(u^{(i)},t^{(i)}) = r^{(i)}$ can be solved using a sharp immersed spatial discretization designed for stationary domains. The state $u^{(i)}$ and right hand side $f(u^{(i)},t^{(i)})$ are then extrapolated, and the integration proceeds to the next stage. This procedure is outlined in Algorithm 2. The issue of artificial time history arises only in the assembly of the residual $r^{(i)}$, which can be calculated under the same conditions that apply to the state $u^{(i)}$ in an explicit integrator.

For parabolic PDEs with moving boundaries, the use of a DIRK integrator can eliminate the severe time step restriction $\Delta t \sim \mathcal{O}(\Delta x^2)$ required by explicit integrators. However, as discussed further below in section 3.5, our proposed moving boundary treatment always comes with a boundary CFL restriction of $\Delta t \sim \mathcal{O}(\Delta x)$. Consequently, there is typically little incentive to use a DIRK scheme for advection-dominant moving boundary simulations. The numerical results using DIRK schemes presented in section 4.2 focus exclusively on the parabolic case.

Algorithm 2 Diagonally implicit Runge-Kutta integration with moving boundaries.

$u^{(0)} = E^{(0)}[u^n]$
for $1 \le i \le s+1$ **do**
  $r^{(i)} = Z^{(i)}\left[v_i u^{(0)} + \sum_{j=1}^{s} \alpha_{ij} u^{(j)} + \Delta t \sum_{j=1}^{s} \beta_{ij} f^{(j)}\right]$
  Solve $(1-\alpha_{ii})\tilde{u}^{(i)} - \Delta t\beta_{ii} f(\tilde{u}^{(i)},t^{(i)}) = r^{(i)}$
  $f^{(i)} = E^{(i)}\left[f(\tilde{u}^{(i)},t^{(i)})\right]$
  $u^{(i)} = E^{(i)}\left[\tilde{u}^{(i)}\right]$
end for
$u^{n+1} = Z^{(s+1)}\left[u^{(s+1)}\right]$

3.4. Modifications for high-order Runge-Kutta schemes in Butcher form

In the moving boundary treatment discussed above, we propose to extend the solution u at each stage using spatial extrapolations that incorporate any prescribed boundary conditions. On the other hand, generally no boundary condition is available for the right-hand side f. It is therefore preferable to choose time integrators that incorporate as much as possible the state values $u^{(i)}$ in their update, in order to increase the opportunity to enforce boundary conditions at freshly cleared points. This is the case for many common integrators that have nonzero α_{ij} when expressed in the modified Shu-Osher form of Eq. (12), but not for high-order Runge-Kutta schemes in Butcher form. In the standard Butcher-form Runge-Kutta update

$$\begin{aligned} u^{(i)} &= u^n + \Delta t \sum_{j=1}^{i-1} a_{ij} f(u^{(j)},t^{(j)}) \quad \text{for} \quad 1 \le i \le s, \\ u^{n+1} &= u^n + \Delta t \sum_{i=1}^{s} b_i f(u^{(i)},t^{(i)}), \end{aligned} \tag{16}$$

freshly cleared cells at the i-th stage are filled using only the original state u^n and the right hand sides $f^{(j)}$ for $1 \leq j < i$. When expressed directly in modified Shu-Osher form, as in Eq. (13), this update leads to an empty α_{ij} coefficient matrix, indicating that extensions of the stage value $u^{(i)}$ with boundary condition are not used in later updates. This stands in contrast with many SSP or low-storage RK schemes that have nonzero α_{ij}.

To maximize the opportunity for boundary condition enforcement, Runge-Kutta schemes in Butcher form can be re-expressed in a different but equivalent modified Shu-Osher form that maximizes the number of nonzero coefficients in α_{ij} without increasing the storage requirements of the scheme. While the two forms are equivalent for problems with stationary boundaries, they lead to different initial values for points that enter the computational domain, which are constructed using a different linear combination of previous stage values and stage boundary conditions. We propose an explicit form in which α_{ij} is strictly lower triangular and β_{ij} is nonzero only on the first subdiagonal, leading to the update

$$\begin{aligned} u^{(1)} &= u^n, \\ u^{(i)} &= \textstyle\sum_{j=1}^{i-1} \alpha_{ij} u^{(j)} + \Delta t \beta_{i,i-1} f(u^{(i-1)}, t^{(i-1)}) \quad \text{for} \quad 2 \leq i \leq s+1, \\ u^{n+1} &= u^{(s+1)}. \end{aligned} \tag{17}$$

Because each $f^{(j)}$ is referenced only once, this algorithm can be implemented in a way that requires the same amount of storage as the standard Butcher implementation. The coefficients $(\alpha_{ij}, \beta_{ij})$ can be obtained from the Butcher tableau (a_{ij}, b_i) as follows. The coefficients of α_{ij} below the main diagonal (which form a lower-triangular matrix $\alpha_{s\times s} \in \mathbb{R}^{s\times s}$) can be written as $\alpha_{s\times s} = M_2 M_1^{-1}$, where the lower-triangular matrices M_1, $M_2 \in \mathbb{R}^{s\times s}$ include lower triangular blocks of a_{ij} and are given by

$$M_1 = \begin{bmatrix} 1 & 0 & \dots & 0 \\ 1 & a_{21} & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 1 & a_{s,1} & \dots & a_{s,s-1} \end{bmatrix} \quad \text{and} \quad M_2 = \begin{bmatrix} 1 & 0 & 0 & \dots & 0 & 0 \\ 1 & a_{31} & 0 & \dots & 0 & 0 \\ 1 & a_{41} & a_{42} & \dots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 1 & a_{s,1} & s_{s,2} & \dots & a_{s,s-2} & 0 \\ 1 & b_1 & b_2 & \dots & b_{s-2} & b_{s-1} \end{bmatrix}. \tag{18}$$

This transformation is possible whenever the first $s-1$ columns of a_{ij} are linearly independent. The s nonzero coefficients in β_{ij} are given by

$$\beta_{i,i-1} = a_{i,i-1} \quad \text{for} \quad 2 \leq i \leq s, \quad \text{and} \quad \beta_{s+1,s} = b_s. \tag{19}$$

For all results presented in this work, Runge-Kutta schemes of order higher than four are implemented using the update in Eq. (17) along with the moving boundary treatment in Algorithm 1.

3.5. Restrictions on boundary motion

For simplicity Algorithms 1 and 2 rely on an extrapolation to nearest neighbors only. To ensure that this generates a complete time history for the active points at each stage, there can be no points that are active at stage i and both inactive and outside of $\mathcal{N}^{(j)}$ in any stage $j < i$. This condition is necessary and sufficient; a complete time history at $\mathbf{x}_k$ implies that for each stage $j < i$, $\mathbf{x}_k$ was either active and received $u^{(j)}$, $f^{(j)}$ through the standard path, or inactive but included in $\mathcal{N}^{(j)}$ so that $u^{(j)}$ and $f^{(j)}$ are extrapolated values. The opposite direction follows immediately. While this condition is easy to check for a given boundary motion and time step, it does not directly offer guidance on the proper choice of time step Δt.

A more convenient sufficient condition is a constraint on the boundary CFL $C_b \equiv \max \left\| \mathbf{v}_b(\mathbf{s},t) \cdot \mathbf{n}(\mathbf{s},t) \right\| \Delta t / \Delta x$. Here the maximum velocity is taken over all $\mathbf{s} \in \Gamma(t)$ and all $t \in [t_{\min}, t_{\max}]$, where $t_{\min}$ and $t_{\max}$ are the minimum and maximum of the stage times $\{t^{(j)}\}_{j=1}^{s+1}$ used during a single Runge-Kutta time step. This bracketing interval is necessary to accommodate Runge-Kutta schemes with abscissae c_i that fall outside the interval $[0,1]$. To arrive at a bound of this form, we focus on the case where $\mathcal{N}^{(i)}$ contains nearest neighbors only and consider the more restrictive condition that no point can be active in one stage and inactive along with all of its nearest neighbors during any other stage of the same time step. This places a restriction on both the rate that the boundary can advance and on the rate at which it can recede, although the former is not strictly necessary in one-sided problems. From there we determine the minimum possible distance $d_{\min}$ between an inactive point with inactive neighbors and a boundary segment with given curvature κ. A boundary CFL constraint follows by requiring that no point on the boundary travels further than $d_{\min}$ along the normal direction during a given time step, which is guaranteed when $\max \left\| \mathbf{v}_b(\mathbf{s},t) \cdot \mathbf{n}(\mathbf{s},t) \right\| (t_{\max} - t_{\min}) < d_{\min}$.

To determine the distance $d_{\min}$, we consider the purely geometric problem illustrated in Fig. 3b of finding the minimum distance between a grid point with four neighbors on the same side of the boundary and a circular boundary with curvature κ. For a straight segment ($\kappa = 0$) the minimum distance is $\Delta x/\sqrt{2}$ (Fig. 3b upper right), while for a curved boundary the minimum distance becomes smaller with increasing κ (Fig. 3b lower left). The analysis is similar in 3D: the minimum distance between a point with six neighbors on the same side of the boundary and a spherical shell of radius κ^{-1} is at most $1/\sqrt{3}$ for planar boundaries and decreases for curved boundaries. Taking the circle and sphere as the worst-case geometry with a prescribed maximum curvature in 2D or prescribed maximum principal curvature in 3D, this geometric analysis leads to the bounds

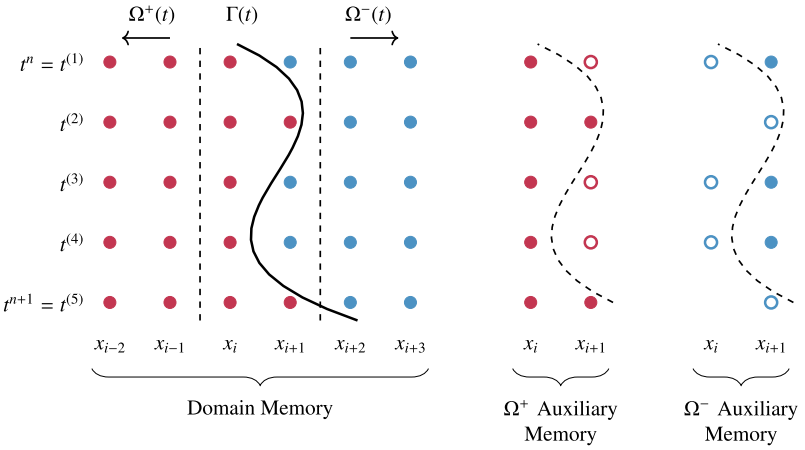


Fig. 4. The moving body implementation for two-sided interfaces described in section 3.6, using a four stage time integrator on a domain with six grid points. At the beginning of the step, four additional memory registers per stage are allocated to hold the values $\{u^{(j,+)}, f^{(j,+)}, u^{(j,-)}, f^{(j,-)}\}$ at both x_i and x_{i+1}, as these points each have a neighbor across the interface. At each stage the Runge-Kutta update is performed on both the domain and the auxiliary memory. After the RK update, the domain values at x_i and x_{i+1} are overwritten with values from one of the auxiliary memory registers; the choice of register is made based on the sign of the level set at that stage. This arrangement maintains a time history for the positive and negative side dynamics at x_i and x_{i+1}, with each history containing both calculated values (solid points) and extension values (open points). When no extension value is available to fill one of the registers, a missing value is assigned (indicated blank space), and that grid point is no longer eligible to cross the interface. This occurs for the Ω^- auxiliary register at x_i during the second stage, and an error would be thrown if x_i crossed into Ω^- during later stages.

$$
\begin{aligned}
C_{b,2D} &< \left(\frac{1}{c_{\max} - c_{\min}}\right)\left(\sqrt{\frac{1}{2}} - \frac{1}{|\kappa\Delta x|} + \sqrt{\frac{1}{|\kappa\Delta x|^2} - \frac{1}{2}}\right) = \left(\frac{1}{c_{\max} - c_{\min}}\right)\left(\frac{1}{\sqrt{2}} - \frac{|\kappa\Delta x|}{4} - \frac{|\kappa\Delta x|^3}{32}\right) + \mathcal{O}\left(|\kappa\Delta x|^5\right), \\
C_{b,3D} &< \left(\frac{1}{c_{\max} - c_{\min}}\right)\left(\sqrt{\frac{1}{3}} - \frac{1}{|\kappa\Delta x|} + \sqrt{\frac{1}{|\kappa\Delta x|^2} - \frac{2}{3}}\right) = \left(\frac{1}{c_{\max} - c_{\min}}\right)\left(\frac{1}{\sqrt{3}} - \frac{|\kappa\Delta x|}{3} - \frac{|\kappa\Delta x|^3}{18}\right) + \mathcal{O}\left(|\kappa\Delta x|^5\right),
\end{aligned}
\tag{20}
$$

where $c_{\max}$ and $c_{\min}$ are the maximum and minimum of the abscissae $\{c^{(j)}\}_{j=1}^{s+1}$ for a given Runge-Kutta scheme. The 2D bound in Eq. (20) is presented in [56] for low storage Runge-Kutta schemes with all abscissae satisfying $0 \leq c_i \leq 1$. In this work we treat geometries that satisfy the curvature constraint $|\kappa\Delta x| < 1/4$, as discussed in section 2.1, and the corresponding CFL restrictions are $C_{b,2D} < 0.644$ and $C_{b,3D} < 0.493$. While these upper bounds are tighter than the CFL restrictions required for stability in advection-dominant problems, the boundary CFL constraint may or may not reduce the maximum time step in practice, as the boundary CFL number C_b depends on the L_2 norm of the body velocity while the flow CFL number depends on the L_1 norm of the flow velocity.

3.6. Two-sided interfaces

So far our analysis has focused on simulations where all dynamics occur on $\Omega^+(t)$, leaving $\Omega^-(t)$ as an empty domain on which extension values are stored. A modified implementation is necessary for simulations in which there are nontrivial dynamics on both $\Omega^+(t)$ and $\Omega^-(t)$ coupled by interface conditions on $\Gamma(t)$. Based on the kinematic analysis of the prior section, a grid point can only cross from $\Omega^+(t)$ to $\Omega^-(t)$ or vice versa during a time step if they have a neighbor in the opposite domain at time t^n. At the start of the time step we allocate $4s$ additional memory locations at each of these points to hold the both the time history $(u^{(j,+)}, f^{(j,+)})$ for the positive-side dynamics and the time history $(u^{(j,-)}, f^{(j,-)})$ for the negative-side dynamics at each stage $1 \leq j \leq s$. After this we perform Algorithm 1, applying the Runge-Kutta updates to both the buffers and the computational domain. The extension operator $E(t^{(j)})[\cdot]$ is re-interpreted as an operator that copies solution and extension values into the buffers, and the zero operator $Z(t^{(j)})[\cdot]$ is re-interpreted as an operator that copies either the positive or negative side buffer values back into the computational domain.

In addition to allowing two-sided dynamics, this implementation also allows for the direct verification of the restrictions on boundary motion. We do so by requiring that the extension operator assigns a flag value to the buffer when a point and all of its neighbors fall on the same side of the interface. This point should not cross the interface in future stages, and the zero operator can verify that values are not copied back into the computational domain from a flagged point. The verification procedure and buffer implementation are illustrated together in Fig. 4.

3.7. Time integrators

In the remainder of this work we focus on seven explicit Runge-Kutta schemes, each referred to by a standard abbreviation: RK2, a second order integrator with two stages; RK4, the classic fourth order integrator with four stages; LSRK(3,3), a third order low-storage integrator with three stages [45]; LSRK(5,4), a fourth order low-storage integrator with five stages that is optimized for stability when applied to advection-diffusion equations [47]; SSPRK(3,3) and SSPRK(5,3), both third order strong-stability-preserving

integrators with three and five stages, respectively [46,48]; and Vern7, Verner's "most efficient" seventh order integrator with a sixth order embedded method and ten stages [49].

In our numerical experiments we apply Runge-Kutta schemes to PDEs with time-dependent boundary conditions, which can lead to an reduction in temporal order due to a mismatch in truncation error between the boundary and interior points [57]. For explicit integrators we do not observe this effect except in the most highly resolved convergence tests, indicating that these errors are typically dominated by existing spatial and temporal errors in our IIM simulations. Consequently, we do not explore any modification of the boundary conditions that could eliminate this effect, such as those proposed in [57–59]. For IIM simulations with implicit time integration this order reduction is more pronounced, and we rely on DIRK schemes with high weak stage order [59] that have been designed specifically to eliminate this effect. Here we adopt the notation DIRK(s,p,q) to indicate a scheme of order p with s stages and weak stage order q, and consider the L-stable schemes DIRK$(4,3,2)$ and DIRK$(6,4,3)$ from [60] as well as DIRK$(7,4,4)$ from [61].

4. Results

4.1. Quantifying moving boundary errors in 1D

In this section we present error analyses of moving boundary treatment proposed above for a range of different physical problems and discretization schemes. For context, we note that in a previous work [56] the authors conjecture that a specialization of the moving boundary treatment presented in section 3, which was developed specifically for explicit low-storage Runge-Kutta schemes, introduces an $\mathcal{O}(\Delta t \Delta x^N)$ error term when applied to 1D IIM discretizations with N-th order spatial accuracy and Dirichlet boundary conditions. This result is confirmed in [56] via numerical experiments for third order 1D advection schemes and second order 1D diffusion schemes. In this section, we extend those results to include higher-order spatial discretizations, more general Runge-Kutta time integrators, and a wider range of boundary conditions.

4.1.1. 1D diffusion with Dirichlet boundary conditions

To begin, we consider a diffusion equation on a periodic 1D domain with an immersed body,

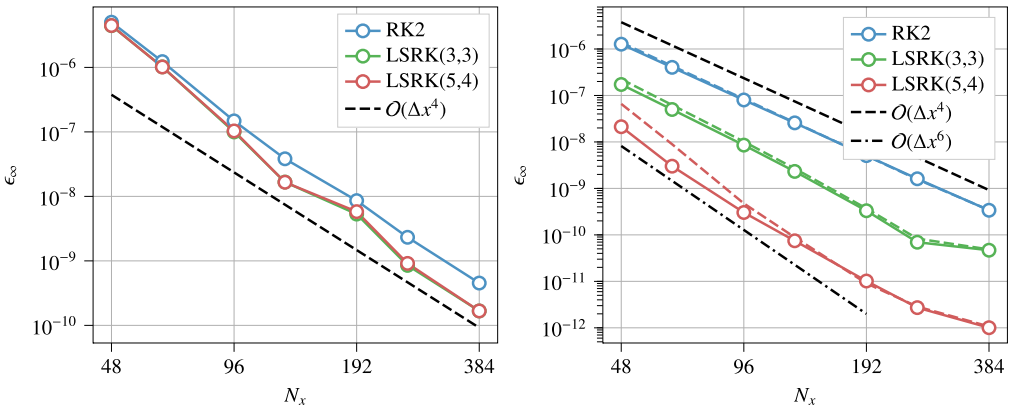
$$\partial_t u + \beta \partial_{xx} u = 0 \quad \text{for} \quad x \in \left[0, x_\ell(t)\right] \cup \left[x_r(t), 1\right). \tag{21}$$

Here the body occupies the moving region $[x_\ell(t), x_r(t)]$. Following the method of manufactured solutions, we define a periodic solution $g(x,t) = \exp(-\beta k^2 t)\sin(kx)$ with $k = 2\pi$ and $\beta = 0.01$, which obeys the diffusion equation on the interval $x \in [0,1]$. Here and in all following diffusion test cases, the diffusivity β is chosen so that the solution magnitude $\exp(-\beta k^2 t)$ at the final time is a reasonable fraction of the initial magnitude (typically between 0.3 and 0.7). We discretize Eq. (21) with periodic domain boundaries, initial condition $u(x,0) = g(x,0)$, and Dirichlet boundary conditions $u(x,t) = g(x,t)$ at the immersed boundary points $\{x_\ell(t), x_r(t)\}$. For all tests here we choose an immersed body in the form of a uniformly translating interval $[x_{\ell,0} + v_b t, x_{r,0} + v_b t]$ with $x_{\ell,0} = 0.261$ and $x_{r,0} = 0.447$. The body translates with velocity $v_b = 0.25$ from initial time $t = 0$ to final time $T = 0.7$.

This 1D system is discretized in space using centered finite differences of order four and six, with a 1D extension-based IIM applied at the immersed boundaries. For an (P,k) IIM discretization, polynomial interpolants of degree $k-1$ are constructed to interpolate the solution at the boundary point and the closest $k-1$ points in the domain, excluding the point closest to the boundary, as illustrated in Fig. 1a (red); these polynomials are extended into the body to construct ghost points needed by the difference scheme. The temporal discretization follows the technique outlined in section 3, using polynomial extensions of the state $u(x,t)$ and the discretized right hand side $f(u,t)$. The extensions of the state are constructed as mentioned above; for the right hand side, polynomials of degree $k-1$ are constructed using the k points closest to the boundary, including the closest point, as illustrated in Fig. 1a (blue).

To measure the spatial convergence of each IIM diffusion discretization, simulations are performed with varying spatial resolution and a time step of $\Delta t = r\Delta x^2/\beta$, where $r = 0.2$ is a constant Fourier number. For each simulation we report the maximum error $\epsilon_\infty = \|u(x,T) - g(x,T)\|_\infty$ in the solution at the final time $t = T$. The results for $(4,5)$ and $(6,7)$ IIM spatial discretizations paired with second, third, and fourth order RK time integrators are presented in Fig. 5. The $(4,5)$ spatial discretization converges at fourth order or higher for any of the chosen time integrators, which is consistent with a fourth order spatial error and a second order or higher temporal error combined with the constant Fourier number constraint $\Delta t \sim \Delta x^2$. Results for the sixth order discretization are more complex, exhibiting fourth order convergence with RK2, fourth order convergence with a lower prefactor for LSRK$(3,3)$, and approximately sixth order convergence with LSRK$(5,4)$. To explain this convergence behavior, we also show the results of a set of simulations that replace the moving interval with a stationary interval $[x_\ell(T), x_r(T)]$ that matches the moving geometry at the final time $t = T$. The error magnitudes for these cases are shown in Fig. 5 as dashed lines without markers and are nearly identical to the moving cases. This indicates that the convergence behavior is not driven primarily by the moving boundary treatment. For the simulations with higher temporal resolution, it appears that the dominant error is due to the order reduction phenomenon that affects Runge-Kutta schemes when applied to method-of-lines discretizations with time-varying boundary conditions. A reduction of each of the Runge-Kutta schemes to second order temporal accuracy, as suggested in [57,58], would lead to fourth order spatial convergence due to the time step restriction $\Delta t \sim \mathcal{O}(\Delta x^2)$. This is consistent with the fourth order spatial convergence rates observed for simulations with LSRK$(3,3)$ and LSRK$(5,4)$.

To more thoroughly explore the temporal convergence of each discretization, we fix a single spatial resolution and perform simulations with varying Δt, beginning at the maximum stable time step for the discretization and decreasing over several orders of



(a) Spatial convergence for (4,5) diffusion.

(b) Spatial convergence for (6,7) diffusion.

Fig. 5. Convergence results for 1D diffusion with Dirichlet boundaries at fixed Fourier number (section 4.1.1). Here ϵ_∞ represents L_∞ norm of the error in the final time solution. Results are shown for both (a) a $(4,5)$ IIM discretization and (b) a $(6,7)$ IIM discretization, which exhibit roughly fourth or sixth order convergence respectively when integrated with a Runge-Kutta scheme of sufficiently high order. Dashed colored lines indicate the error in the equivalent stationary cases on both plots, though these are indistinguishable from the moving cases for the $(4,5)$ discretization.

Table 1
Spatial exponents of the mixed error terms $A\Delta t\Delta x^N$ observed in diffusion simulations with moving boundaries and Dirichlet boundary conditions (Dir, see Fig. 6), Neumann boundary conditions (Neu, see Fig. A.19), or discontinuous coefficients with jump conditions (Jump, see Fig. A.20). The data used to generate each fit is indicated by black dashed lines on the corresponding figures. The $(6,7)$ Dirichlet datapoint is omitted due to a lack of sufficient data for the fit.

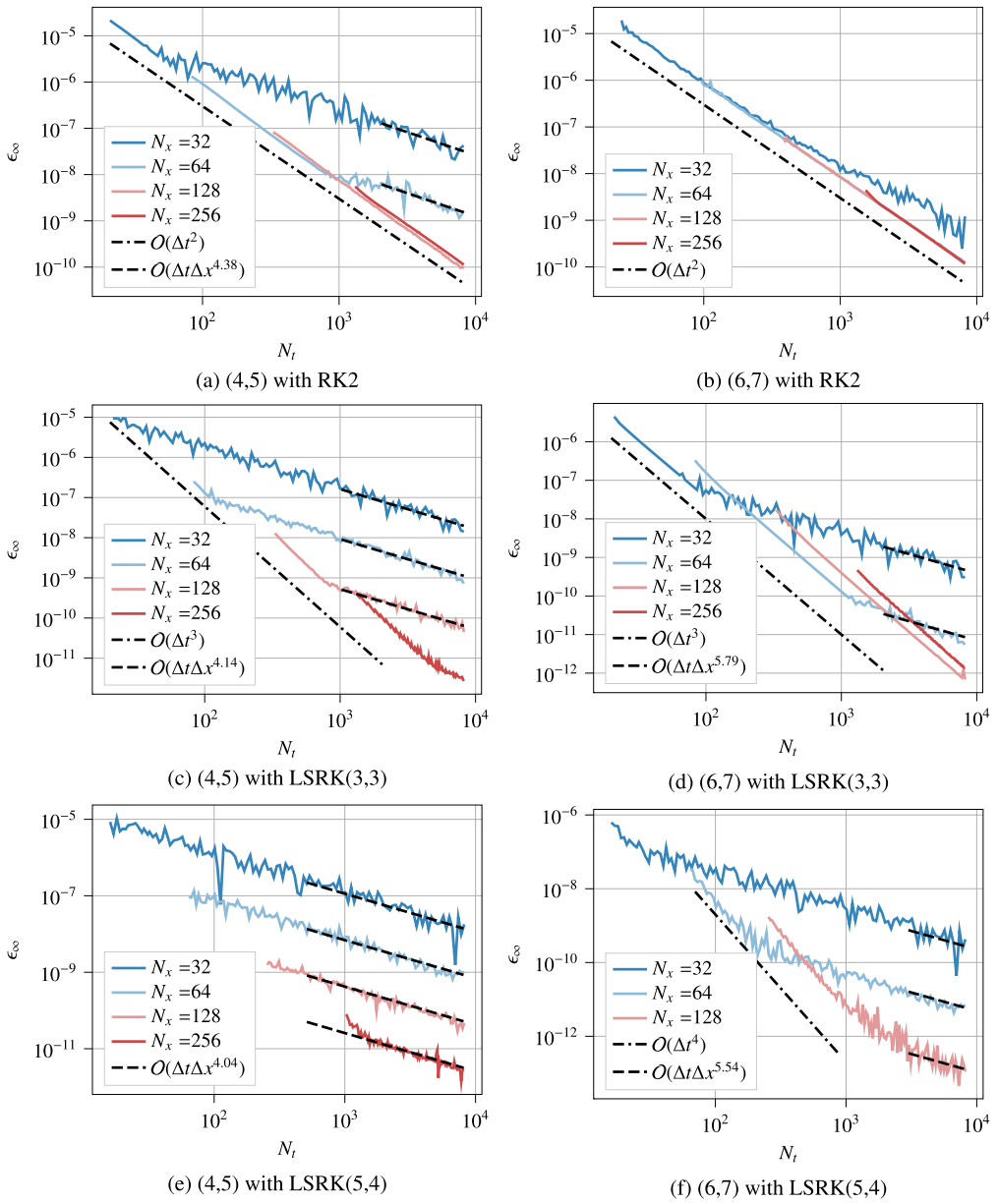
Integrator	(4, 5) Dir	(6, 7) Dir	(4, 5) Neu	(6, 7) Neu	(4, 5) Jump	(6, 7) Jump
RK2	4.38	N/A	2.89	4.94	3.51	5.09
LSRK(3,3)	4.14	5.79	2.94	4.89	3.51	5.19
LSRK(5,4)	4.04	5.54	2.93	4.74	3.40	5.17

magnitude. The results are compared to a solution with the same resolution and a much smaller time step, so that any purely spatial error is removed from the comparison. Fig. 6 displays the results for $(4,5)$ and $(6,7)$ IIM discretizations with Dirichlet boundary conditions at resolutions $N_x = [32, 64, 128, 256]$. Time integration is performed with either the RK2, LSRK(3,3), and LSRK(5,4) time integrators. Referring to Fig. 6a, corresponding to a $(4,5)$ scheme with RK2 time integration, we observe that for each fixed spatial resolution the temporal convergence order is initially second order, consistent with the order of the time integrator. When the number of time steps increases beyond a given threshold, the convergence diminishes to first order, indicating that the error introduced by moving boundaries is dominant. By varying the spatial resolution we can deduce that the magnitude of the $\mathcal{O}(\Delta t^2)$ error term is independent of Δx, while the $\mathcal{O}(\Delta t)$ error term has a prefactor that scales with Δx^N for some positive exponent N. Similar behavior can be observed for the other combinations of time integrator and spatial discretization shown in Figs. 6b through 6f: for integrators of order M, the convergence is initially $\mathcal{O}(\Delta t^M)$, with an $\mathcal{O}(\Delta t)$ error term that depends strongly on spatial resolution dominating at smaller time steps. This is consistent with the existence of an $\mathcal{O}(\Delta t\Delta x^N)$ moving boundary error term, as proposed in [56].

To determine the exponent N for a given discretization, we discard convergence data for which the $\mathcal{O}(\Delta t^M)$ error term is dominant, and perform a linear least squares fit between $\log \epsilon_\infty$ and $\log(A\Delta t\Delta x^N)$ to determine the prefactor A and exponent N of the moving boundary error term. The results for each combination of spatial discretization and time integrator are listed in Table 1, which indicates that the exponent is primarily determined by the order of the spatial discretization: a $(4,5)$ scheme leads to $N \approx 4$, while a $(6,7)$ scheme leads to $N \approx 6$. Importantly, when the time step obeys the stability constraint $\Delta t \sim \Delta x^2$ that is necessary for diffusion simulations with explicit time integration, the resulting moving boundary error terms have magnitude $\mathcal{O}(\Delta x^{N+2})$ and do not dominate the existing spatial error (Fig. 5).

4.1.2. 1D diffusion with Neumann boundary conditions

To evaluate the magnitude of the mixed error term introduced by Neumann boundary conditions, the temporal convergence tests described above are repeated with the same simulation parameters, discretizations, and resolutions, changing only the boundary condition prescribed in each simulation to the Neumann condition $\partial_x u(x,t) = \partial_x g(x,t)$ for $x \in \{x_\ell(t), x_r(t)\}$. The results mirror the Dirichlet case, displaying regions with a dominant $\mathcal{O}(\Delta t^M)$ error term and regions with a dominant $\mathcal{O}(\Delta t\Delta x^N)$ error term; for full results see Fig. A.19 in the Appendix. To determine the exponent N in the mixed error term, we repeat the least squares fitting procedure described above for each combination of time integrator and spatial discretization. The results are given in Table 1, which indicates that $N \approx 3$ for the $(4,5)$ IIM discretizations and $N \approx 5$ for the $(6,7)$ discretizations independent of the choice of time integrator. These exponents are one order lower than those observed with Dirichlet boundary conditions, likely due to a boundary



(a) (4,5) with RK2 (b) (6,7) with RK2

(c) (4,5) with LSRK(3,3) (d) (6,7) with LSRK(3,3)

(e) (4,5) with LSRK(5,4) (f) (6,7) with LSRK(5,4)

Fig. 6. Temporal convergence of the final-time L_∞ error norm for the 1D diffusion equation with Dirichlet boundary conditions and a fourth order (left column) or sixth order (right column) discretization; see section 4.1.1. N_t represents the number of time steps in each simulation. Time integration is performed with the RK2, LSRK(3,3), or LSRK(5,4) time integrators, and the leftmost point of each series represents the maximum stable time step for each discretization. For all cases the temporal convergence is dominated by either an $\mathcal{O}(\Delta t^M)$ error term with a prefactor that is largely independent of spatial resolution or by an $\mathcal{O}(\Delta t \Delta x^N)$ term attributed to the presence of moving boundaries, where M is the order of the time integrator and N is the order of the spatial discretization. Dashed lines indicate the data on each plot used to determine the corresponding exponent N via a least squares fit.

condition that depends on the first derivative of the solution. As in the previous section, the mixed error term does not asymptotically dominate the spatial error when the time step is determined by the stability constraint $\Delta t \sim \Delta x^2$.

4.1.3. 1D diffusion with discontinuous coefficients

To investigate PDEs with dynamics on both sides of an immersed interface, we repeat the analysis of the previous section for the diffusion equation with a piecewise constant diffusivity

$$\partial_t u = \beta(x)\partial_{xx} u, \quad \beta(x) = \begin{cases} \beta^-, & x \in [x_\ell(t), x_r(t)] \\ \beta^+, & \text{otherwise} \end{cases}. \tag{22}$$

On the interface $\Gamma(t) = \{x_\ell(t), x_r(t)\}$ boundary conditions are replaced by the jump conditions on the solution and the diffusive flux defined in Eq. (8). The magnitude of the jumps is set to match the manufactured solution

$$g(x,t) = \begin{cases} \exp\left(-\beta^- k^2 t\right)\sin(kx), & x \in \left[x_\ell(t), x_r(t)\right] \\ \exp\left(-\beta^+ k^2 t\right)\sin(k(x-\delta)), & \text{otherwise} \end{cases} \tag{23}$$

with $\beta^- = 0.01$, $\beta^+ = 0.005$, $k = 2\pi$, and $\delta = 0.2$. The moving interface begins at $x_{\ell,0} = 0.251$, $x_{r,0} = 0.507$ at time $t = 0$ and translates at a constant velocity $v_b = 0.25$ until final time $T = 1$.

Repeating the temporal convergence tests of the previous two sections produces behavior that is consistent with the Dirichlet and Neumann cases; see Fig. A.20 in the Appendix for a full listing of the results. We observe both an $\mathcal{O}(\Delta t^M)$ temporal error term and an $\mathcal{O}(\Delta t \Delta x^N)$ mixed error term attributed to moving boundaries, with the exponent N for each integrator and spatial discretization listed in Table 1. For each spatial discretization the measured exponent is larger than that observed with Neumann boundary conditions and smaller than that observed with Dirichlet boundary conditions: the $(4,5)$ discretization leads to $3.4 \leq N \leq 3.51$, while the $(6,7)$ discretizations leads to $5.09 \leq N \leq 5.19$.

4.1.4. 1D advection with Dirichlet boundary conditions

As a model hyperbolic system we consider the 1D linear advection equation $\partial_t u + v\partial_x u = 0$. The problem domain is identical to the previous sections, an interval with initial position $x_{\ell,0} = 0.261$ and $x_{r,0} = 0.447$ translating at speed v_b. A Dirichlet boundary condition is prescribed on boundaries with $n(v - v_b) > 0$, where the normal $n = \pm 1$ points into the problem domain. Convergence is measured with the method of manufactured solutions using the translating wave $g(x,t) = \sin(k(x - vt))$ for $(x, t) \in [0,1] \times [0,T]$ with $k = 6\pi$ and $T = 0.7$. Errors are measured in the L_∞ norm at the final time $t = T$. The IIM discretization uses the $(3,4)$ upwind finite difference scheme presented in section 2.2, and moving boundaries are treated using fourth-order extrapolations. Where a Dirichlet boundary condition is available, it is used both in the difference scheme and in the moving boundary extrapolations; otherwise all extrapolation is performed using the interior solution values only.

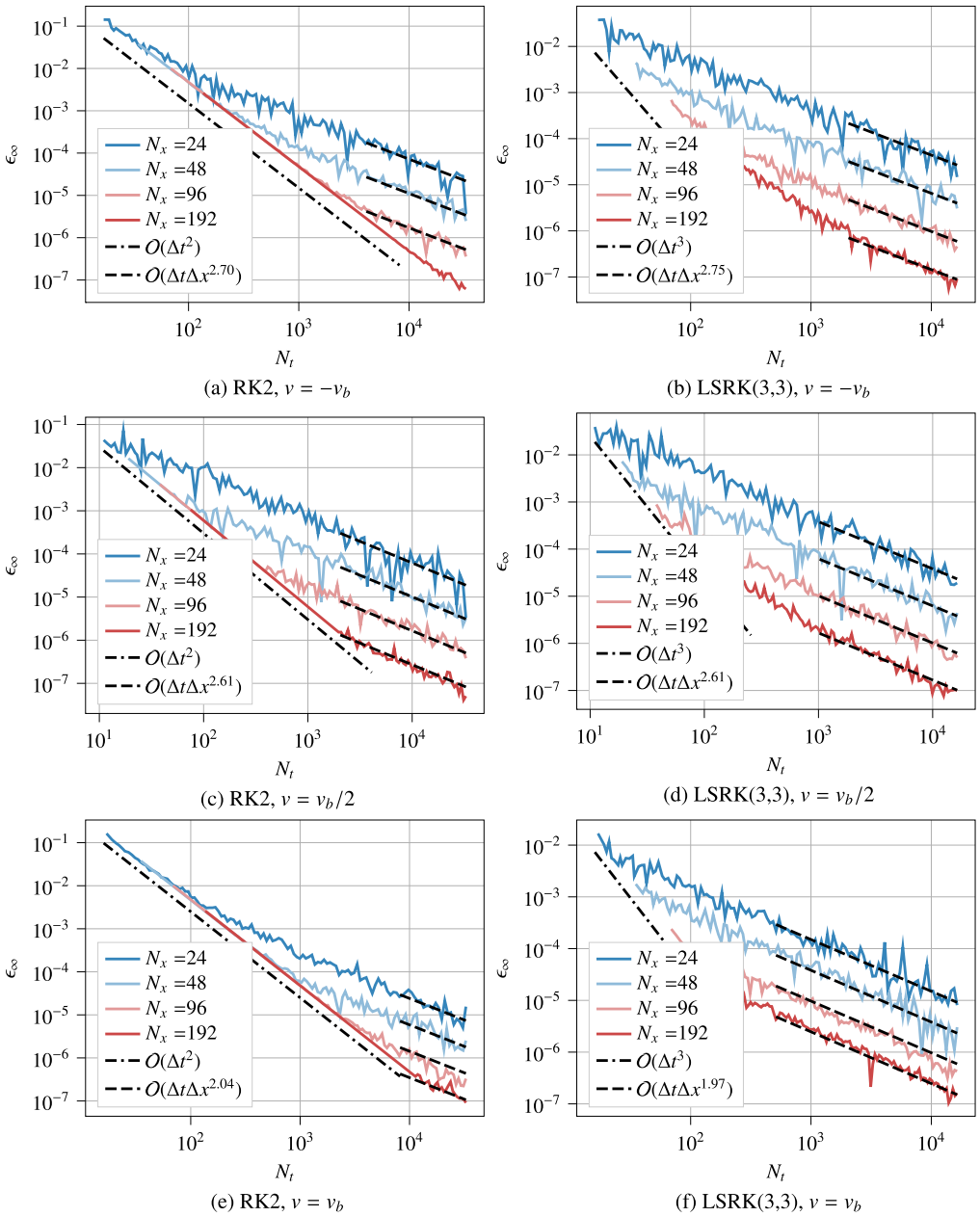
Throughout our test cases we set the boundary velocity to the constant value $v_b = 0.5$ so that the boundaries move left-to-right, while varying the flow velocity v in three different ways. For case one, the flow velocity $v = -v_b$, so that the flow moves right-to-left, opposing the boundary motion. In case two, $v = v_b/2$ so that the flow and boundary are aligned, but the body outpaces the flow. Finally, in case three the flow and boundary translate at the same velocity $v = v_b$, so that there is no through flow and no Dirichlet boundary condition is required. Fig. 7 presents temporal convergence results obtained with either RK2 or LSRK$(3,3)$ time integration for these three cases. As with the diffusion test cases, an $\mathcal{O}(\Delta t \Delta x^N)$ moving boundary error term dominates at fine temporal resolutions, and a least squares fit is used to determine the spatial exponent N for each time integrator and velocity combination. For the first two cases $v = -v_b$ and $v = v_b/2$, the exponent is $2.6 \leq N \leq 2.75$, or slightly lower than the spatial convergence order of 3, which is consistent with the convergence order of the mixed term in the diffusion test cases. However, for the no-through flow case (case three) with either RK2 or LSRK$(3,3)$ integration we observe $N \approx 2$. To explain these values of N we can postulate a straightforward error model, starting from the empirical observation that a pointwise error of magnitude $\mathcal{O}(\Delta t \Delta x^3)$ is introduced at each grid point that enters the domain. For linear PDEs the dynamics of the error mirror the dynamics of the solution, which means that any errors introduced into the problem domain are advected by the flow. When $v - v_b \neq 0$ each of the pointwise $\mathcal{O}(\Delta t \Delta x^3)$ error contributions is advected away from the boundary, leaving an error field of rough magnitude $\mathcal{O}(\Delta t \Delta x^3)$ dispersed throughout the domain. When $v - v_b = 0$, however, all error contributions are advected alongside the body, so that the error adjacent to the immersed boundary is an accumulation of the error from all grid point crossings. Under a CFL constraint, the number of grid point crossings is $\mathcal{O}(\Delta x^{-1})$ at any given final time, explaining why the spatial exponent in the mixed error term is reduced by one for this case.

In all cases we observe that $N \geq 2$, so that when the time step is determined by a CFL criterion, the overall order of the moving boundary error term is equal to or higher than the third order spatial error. This means in practical simulations using explicit time integrators, the moving boundary error term will not dominate the convergence of the algorithm.

4.2. 1D simulations with additional time integrators

While the results of the previous section focused on low storage RK schemes, the moving boundary treatment outline in section 3 is equally applicable to standard Butcher-form RK schemes, strong stability preserving RK schemes, and diagonally implicit schemes. As a representative example, Fig. 8a revisits the spatial convergence of 1D advection testcase with $v = -0.5$, $v_b = 0.5$, comparing LSRK$(3,3)$ time integration with the RK4, Vern7, SSPRK$(3,3)$ and SSPRK$(5,3)$ time integrators. Both the RK4 and Vern7 integrators are implemented in the modified Shu-Osher form outlined in section 3.4, and all time steps are chosen to satisfy the CFL constraint $C = C_b = 0.7$. At lower resolutions all four integrators achieve third order convergence with nearly identical error magnitudes, indicating that the $\mathcal{O}(\Delta x^3)$ error of the spatial discretization is dominant. As the resolution increases, an $\mathcal{O}(\Delta x^2)$ error emerges for the third and fourth order time integrators, which is consistent with an order reduction due to time-dependent boundary conditions as discussed in section 3.7. Notably, the Vern7 integrator is able to maintain third order convergence up to the maximum resolution $\Delta x = 1/4096$, indicating that the prefactor on the $\mathcal{O}(\Delta x^2)$ error is significantly smaller than that of the lower order integrators.

For diffusion-dominated PDEs, explicit time integrators suffer from restrictive time step limits due to stability constraints. Diagonally implicit schemes offer an attractive alternative. Fig. 8b revisits the 1D diffusion test case from section 4.1.1, measuring the



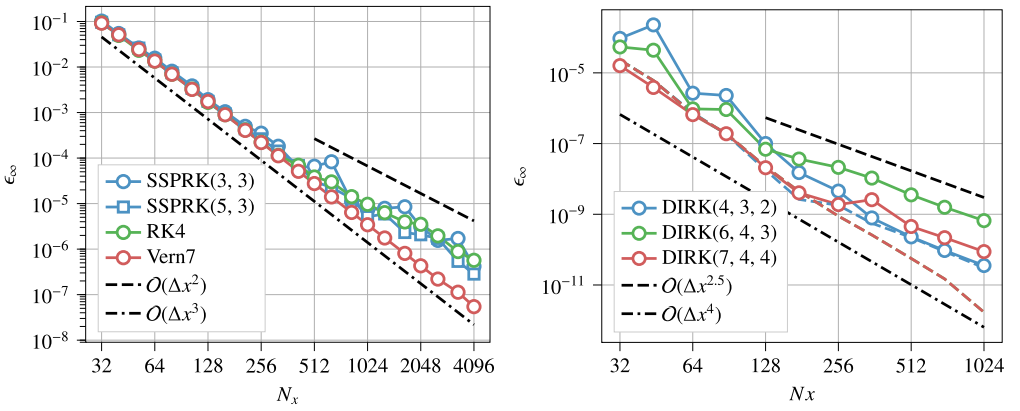
(a) RK2, $v = -v_b$ (b) LSRK(3,3), $v = -v_b$

(c) RK2, $v = v_b/2$ (d) LSRK(3,3), $v = v_b/2$

(e) RK2, $v = v_b$ (f) LSRK(3,3), $v = v_b$

Fig. 7. Temporal convergence of the final-time L_∞ error norm for the 1D advection equation with Dirichlet boundary conditions and either RK2 (left column) or LSRK(3, 3) (right column) time integration. See section 4.1.4 for interpretation.

convergence of a (4, 5) IIM diffusion discretization when integrated with three different DIRK schemes of weak stage order higher than one [60,61]. For each integrator results are also shown for a similar case with the same parameters and a stationary body that occupies the interval $[x_\ell(T), x_r(T)]$, which matches the geometry of the moving cases at the final time $t = T$. In each simulation the time step is chosen to yield a body CFL number that is 80% of the maximum acceptable value, which varies between integrators due to the fact that DIRK(6, 4, 3) and DIRK(7, 4, 4) have abscissae outside of the interval $[0, 1]$. There is no constraint on the Fourier number, so that $\Delta t \sim \Delta x$ for these convergence tests. For each integrator the convergence of each moving test case is approximately fourth order at low resolutions, with an asymptotically larger $\mathcal{O}(\Delta x^{2.5})$ error dominating at finer resolutions. For the fourth order integrators DIRK(6, 4, 3) and DIRK(7, 4, 4), the equivalent stationary test cases do not exhibit order reduction due to time-dependent boundary conditions, indicating that in general DIRK schemes with a high weak stage order do not mitigate order reduction when combined with our moving boundary treatment. Interestingly, the third order DIRK(4, 3, 2) integrator also exhibits order reduction in the stationary test case, but provides the most accurate results for moving test cases at fine resolutions, outperforming both of the



(a) (3, 4) IIM advection with various explicit integrators. (b) (4, 5) IIM diffusion with various DIRK integrators.

Fig. 8. Results from section 4.2. (a) 1D convergence results for a (3,4) IIM advection discretization integrated with a constant body CFL and four different explicit Runge-Kutta schemes. At lower resolutions the $\mathcal{O}(\Delta x^3)$ spatial error dominates. At higher resolutions the $\mathcal{O}(\Delta t^2)$ error attributed to the presence of time-dependent boundary conditions becomes dominant for all integrators except Vern7. (b) 1D convergence results for a (4,5) IIM diffusion discretization integrated with a constant body CFL and various DIRK schemes of weak stage order greater than one. For each integrator the results of an equivalent stationary test case are shown with a dashed line of the same color (though the stationary results for DIRK(7,4,4) nearly overlap DIRK(6,4,3)). In all moving test cases the convergence begins at approximately fourth order and degenerates to $\mathcal{O}(\Delta x^{2.5})$ at fine resolutions, at which point the third order integrator DIRK(4,3,2) achieves the lowest error.

fourth order integrators with higher weak stage orders while requiring the fewest linear system solves. We leave a full investigation of these order reduction phenomena in moving boundary simulations to future work.

4.3. 2D convergence results

For all 2D results we consider PDEs posed on the unit square $\Omega = [0,1]^2$ with periodic boundaries. The immersed surface $\Gamma(t)$ is a smooth five-pointed star with initial center $\mathbf{x}_0$ that translates uniformly with velocity $\mathbf{v}_b$. In a polar coordinate system centered on the point $\mathbf{x}_b(t) = \mathbf{x}_0 + \mathbf{v}_b t$, the star is defined by the level set

$$\phi(r,\theta) = r - r_0 - \Delta r \cos(5\theta), \tag{24}$$

where r_0 is the average radius and Δr is the maximum radial perturbation. For all cases we use the method of manufactured solutions and report the L_∞ norm of the error in the final time solution.

4.3.1. 2D diffusion with Dirichlet or Neumann boundary conditions

For this test case we consider the 2D diffusion equation $\partial_t u = \beta \nabla^2 u$ on an irregular domain, with either Dirichlet or Neumann boundary conditions on the immersed surface. The domain is defined by the star level set from Eq. (24) with average radius $r = 0.2$, radial perturbation $\Delta r = 0.03$, initial center $\mathbf{x}_0 = [0.41, 0.41]$, and constant velocity $\mathbf{v}_b = [0.4, 0.4]$. The manufactured solution is a decaying sinusoid

$$g(\mathbf{x},t) = \exp\left(-\beta(k_1^2 + k_2^2)t\right)\sin\left(k_1 x_1\right)\sin\left(k_2 x_2\right) \tag{25}$$

with diffusivity $\beta = 0.004$ and wavenumbers $k_1 = k_2 = 4\pi$. Time integration begins at $t = 0$ and runs to a final time $T = 0.5$, using the LSRK(3,3) integrator and a fixed Fourier number $r = \Delta t \beta / \Delta x^2 = 0.1$. This ensures that as the spatial grid is refined, any purely temporal errors have a magnitude that reduces with $\mathcal{O}(\Delta x^6)$ at most.

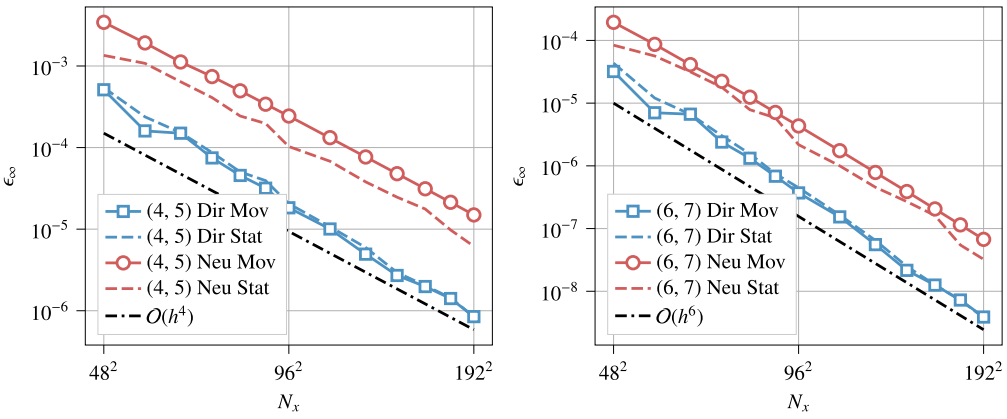
Fig. 9 plots the L_∞ error in the final time solution as a function of the grid resolution, using either a (4,5) or (6,7) IIM discretization. Results for a moving domain are shown alongside a set of similar stationary cases, which use the same manufactured solution but a constant body center $\mathbf{x}_b = \mathbf{x}_0 + \mathbf{v}_b T$ that agrees with the moving domain cases at the final time. For both Dirichlet and Neumann boundary conditions the IIM discretizations achieve the expected fourth or sixth order convergence in the L_∞ norm, with an error magnitude that is comparable to the stationary cases.

4.3.2. 2D diffusion with discontinuous coefficients

For this test case we consider the 2D diffusion equation with an immersed interface that separates two regions with different diffusivities,

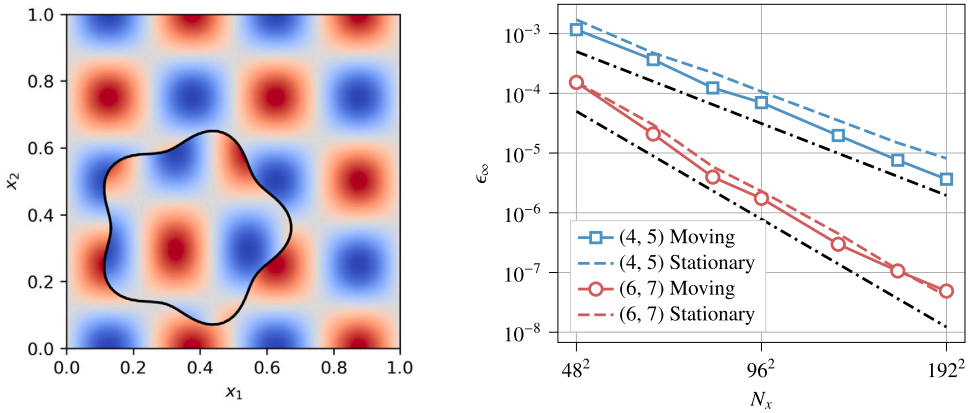
$$\partial_t u = \beta(x)\nabla^2 u, \quad \beta(x) = \begin{cases} \beta^+, & \phi(x,t) > 0 \\ \beta^-, & \phi(x,t) < 0 \end{cases}. \tag{26}$$

On the interface jump conditions on the solution and diffusive flux are prescribed as in Eq. (8), with the jump values that are set to match the manufactured solution. The domain is defined by the star level set from Eq. (24) with average radius $r = 0.27$,



(a) (4, 5) IIM discretization. (b) (6, 7) IIM discretization.

Fig. 9. Convergence results for 2D diffusion with Dirichlet or Neumann boundary conditions (section 4.3.1). Each simulation is performed with LSRK(3, 3) time integration using (a) a fourth order spatial discretization or (b) a sixth order spatial discretization. Both the fourth and sixth order discretizations exhibit the expected order of accuracy, and the magnitude of the error is comparable to the equivalent stationary cases (dashed lines).



(a) Initial condition and interface geometry. (b) Convergence results, $\beta^-/\beta^+ = 2$.

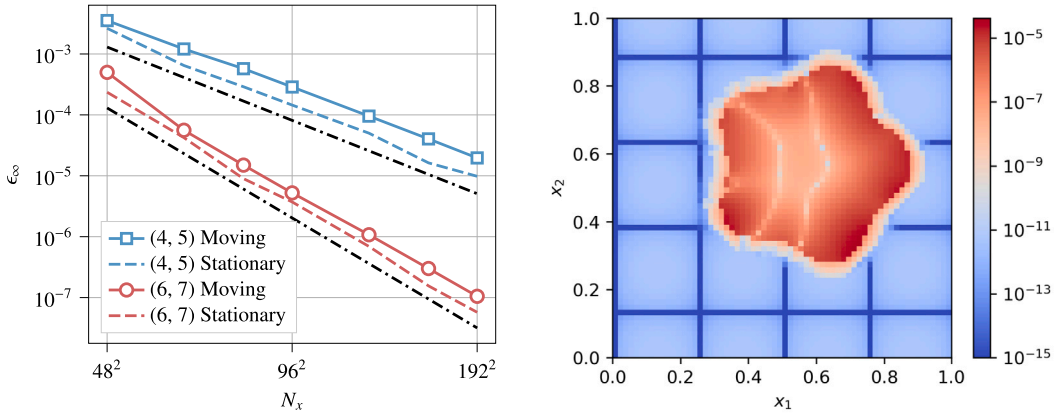
Fig. 10. Convergence results for 2D diffusion with discontinuous coefficients and a moving boundary (section 4.3.2). The final time manufactured solution is shown in (a), along with the L_∞ error as a function of grid resolution in (b). Both the fourth and sixth order discretizations exhibit the expected order of accuracy, and the magnitude of the error is comparable to the equivalent stationary cases (dashed lines).

radial perturbation $\Delta r = 0.03$, initial center $\mathbf{x}_0 = [0.372, 0.360]$, and constant velocity $\mathbf{v}_b = [0.4, 0.4]$. The manufactured solution is a decaying sinusoid in each domain with differing wavenumbers,

$$g(\mathbf{x}, t) = \begin{cases} +\exp\left(-\beta^+ \|\mathbf{k}^+\|^2 t\right) \sin\left(k_1^+ x_1\right) \sin\left(k_2^+ x_2\right), & \phi(x,t) > 0 \\ -\exp\left(-\beta^- \|\mathbf{k}^-\|^2 t\right) \sin\left(k_1^- x_1\right) \sin\left(k_2^- x_2\right), & \phi(x,t) < 0 \end{cases}, \tag{27}$$

with diffusivities $\beta^+ = 0.002$ and $\beta^- = 0.004$. We choose wavenumbers $\mathbf{k}^+ = [4\pi, 4\pi]$ and $\mathbf{k}^- = [4.6\pi, 3.4\pi]$ to ensure that the jump conditions on the interface vary in both space and time. Time integration is performed as in the previous section with fixed Fourier number $r = \max(\beta^+, \beta^-)\Delta t/\Delta x^2 = 0.15$.

Fig. 10 provides convergence results for both (4, 5) and (6, 7) IIM discretizations. Both achieve their nominal order of accuracy in the L_∞ norm with error magnitudes that are equal to or less than the equivalent stationary cases. To demonstrate robustness to large jumps in coefficients, we repeat the tests above with the same β^- and a larger diffusivity ratio $\beta^-/\beta^+ = 10^4$. Fig. 11a displays the results, which show roughly the same convergence behavior and error magnitudes as with $\beta^-/\beta^+ = 2$. This indicates that the error is dominated by the domain with the larger diffusivity (as shown in Fig. 11b), and that the stability of the discretization is not affected by large jumps in coefficients. Notably, the (6, 7) IIM discretization is able to converge at sixth order for a two-sided diffusion problem with a moving nonconvex interface, coupling conditions that involve surface gradients of the solution, and jumps in coefficients by up to four orders of magnitude.



(a) Convergence results, $\beta^-/\beta^+ = 10^4$.

(b) Final time error for the (6, 7) discretization with $N_x = 64$.

Fig. 11. Results for 2D diffusion with discontinuous coefficients and a large diffusivity ratio of $\beta^-/\beta^+ = 10^4$ (section 4.3.2). Convergence results in (a) indicate that both fourth and sixth order discretizations remain stable and achieve their nominal order of convergence despite the large diffusivity ratio. Additionally, the error magnitudes for these moving cases are comparable to those of the equivalent stationary cases (dashed lines). The final time error field for the (6, 7) discretization with $N_x = 64$ is shown in (b) and indicates that the error is dominated by the region with larger diffusivity. This explains the similarity in error magnitude between these results and those shown in Fig. 10, which use the same β^- and a smaller diffusivity ratio $\beta^-/\beta^+ = 2$.

4.3.3. 2D advection with Dirichlet boundary conditions and constant velocity

To test the stability and convergence of the IIM advection discretization developed in section 2, we consider the scalar advection equation $\partial_t u + \mathbf{v} \cdot \nabla u = 0$. For hyperbolic PDEs a boundary condition is required only where characteristics enter the domain, so that a Dirichlet boundary condition on a moving body takes the form

$$u(\mathbf{s},t) = g(\mathbf{s},t) \quad \text{for all} \quad \left\{\mathbf{s} \in \Gamma(t) \,|\, (\mathbf{v}(\mathbf{s},t) - \mathbf{v}_b(\mathbf{s},t)) \cdot \mathbf{n}(\mathbf{s},t) > 0\right\}. \tag{28}$$

The above PDE and Dirichlet boundary condition are discretized using a $(3,4)$ IIM advection discretization, which has a third order spatial truncation error. The problem domain is periodic and consists of the exterior of the star defined by the level set given in Eq. (24), with radius $r = 0.2$, radial perturbation $\Delta r = 0.03$, initial center $\mathbf{x}_0 = [0.351, 0.351]$, and constant body velocity $\mathbf{v}_b = [0.6, 0.6]$. Convergence is assessed using the manufactured solution

$$g(\mathbf{x},t) = \sin\left(4\pi(x_1 - v_1 t)\right)\cos\left(4\pi(x_2 - v_2 t)\right). \tag{29}$$

To test the selective enforcement of inflow boundary conditions, we consider four distinct cases for the flow velocity: (i) a velocity field opposed to the body motion, $\mathbf{v} = -\mathbf{v}_b$; (ii) a velocity field perpendicular to the body motion, $\mathbf{v} = \mathbf{v}_b \times \hat{\mathbf{e}}_3$; (iii) a velocity field $\mathbf{v} = \mathbf{v}_b/2$ that is parallel to the flow but slower; and (iv) a no through-flow velocity field with $\mathbf{v} = \mathbf{v}_b$, for which no boundary condition is enforced on any part of the immersed surface. Time integration runs from $t = 0$ to $T = 0.5$ using the LSRK(3,3) time integrator, with the time step determined by the body CFL constraint $C_b = 0.64$. This results in a constant flow CFL of $C_v = 0.45$ for case (iii) and $C_v = 0.9$ for the other three cases.

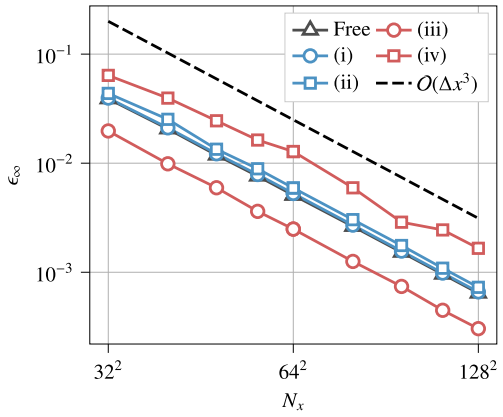
Fig. 12a presents convergence results for all four test cases, as well as a free-space test case that uses the same manufactured solution with no solid body and $\mathbf{v} = [0.6, 0.6]$. All of the moving boundary cases converge at third order in the L_∞ norm, and all but the no through-flow case show a similar or smaller error magnitude compared to the free space case. The final-time error field for the no through-flow case with $N_x = 32$, shown in Fig. 12b, indicates that the IIM boundary treatment leads to an accumulation of error near the surface for this test case. This is possible because of the lack of relative motion between the flow and the body, which for other test cases allows some accumulated error to be covered by the boundary motion or advected into the problem domain. Consistent with the 1D convergence results presented in section 4.1.4, this accumulation of error does not affect the third order spatial convergence rate of the method.

4.3.4. 2D advection with Dirichlet boundary conditions and variable velocity

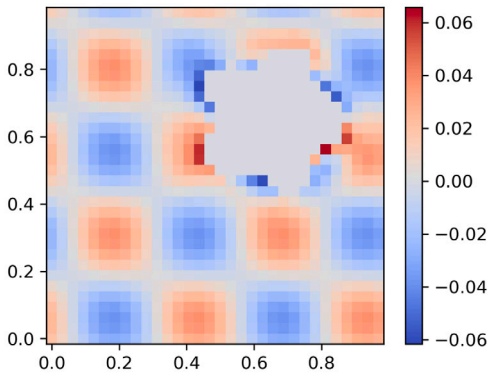
To test the utility of our advection discretization when applied to more realistic flow fields, we consider a test case with body and flow velocities that vary in both space and time. Manufactured solutions with variable velocity can be constructed through the use of a reference map $\boldsymbol{\xi}(\mathbf{x},t)$ that obeys the advection equation $\partial_t \boldsymbol{\xi} + \mathbf{v} \cdot \nabla \boldsymbol{\xi} = 0$ with initial condition $\xi(\mathbf{x},0) = \mathbf{x}$. For any initial condition $g_0(\mathbf{x})$, the time-varying field $g(\mathbf{x},t) = g_0(\xi(\mathbf{x},t))$ solves the advection equation. Here use the sinusoidal reference map and corresponding velocity field

$$\boldsymbol{\xi}(\mathbf{x},t) = \mathbf{x} - \delta \sin(\omega t)[\sin\left(2\pi x_2\right), \sin\left(2\pi x_1\right)], \quad \mathbf{v}(\mathbf{x},t) = [\nabla \boldsymbol{\xi}(\mathbf{x},t)]^{-1} \partial_t \boldsymbol{\xi}(\mathbf{x},t), \tag{30}$$

with frequency $\omega = 7\pi/2$, length scale $\delta = 0.1$ and initial condition $g_0(\mathbf{x}) = \sin\left(4\pi x_1\right)\sin\left(4\pi x_2\right)$. This velocity field is not incompressible, and the reference map has a Jacobian $J = \det(\nabla \boldsymbol{\xi})$ in the range $1 \pm 4\pi^2\delta^2$. The domain is defined by the star level set from Eq. (24) with average radius $r = 0.2$, radial perturbation $r = 0.03$, and initial center $\mathbf{x}_0 = [0.321, 0.321]$. To create a time-varying body

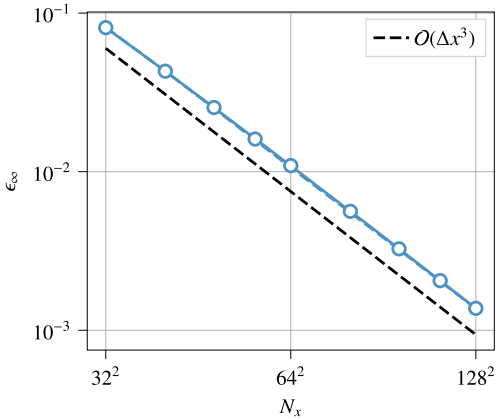


(a) Convergence results for 2D advection.

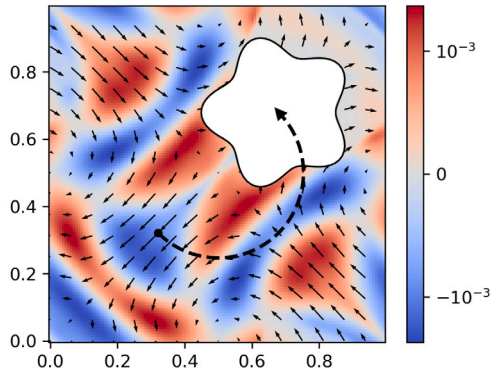


(b) No through-flow error field (case iv), $N_x = 32$.

Fig. 12. (a) Convergence results for 2D advection with Dirichlet boundary conditions (section 4.3.3). Cases (i) through (iv) and the free-space case are described in the text, and all cases converge at third order in the L_∞ norm. (b) The final-time error field for the no through-flow case (case iv) with $N_x = 32$. The maximum error is concentrated on the immersed surface, while the error in the rest of the domain is consistent with that of the free-space finite difference scheme.



(a) Convergence results for 2D advection with variable velocity.



(b) Final time velocity and error fields, $N_x = 128$.

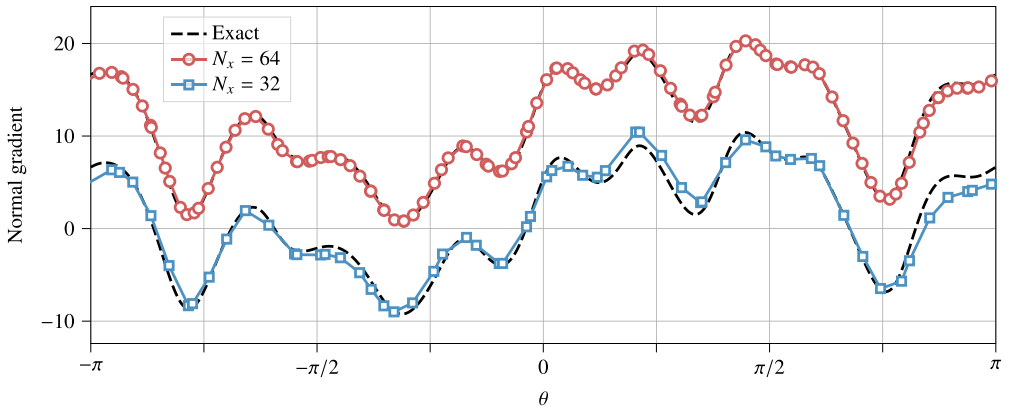
Fig. 13. Results from section 4.3.4. (a) Third order L_∞ norm convergence for the 2D variable velocity advection test case. The blue dashed line lying nearly on top of the series indicates the error for the equivalent stationary boundary test case. (b) The final time velocity field and error field for the variable velocity test case with $N_x = 128$. The path of the body center is superimposed as a dashed line.

velocity at each boundary point, the star rotates about an axis passing through $\mathbf{x}_c = [0.5, 0.5]$ with constant angular velocity $\omega_b = \pi$. Time integration begins at time $t = 0$ and proceeds time $T = 1.0$ using the LSRK(3, 3) time integrator and a constant time step chosen to satisfy the body CFL constraint $C_b < 0.64$, which leads to a maximum flow CFL of $C = 0.93$.

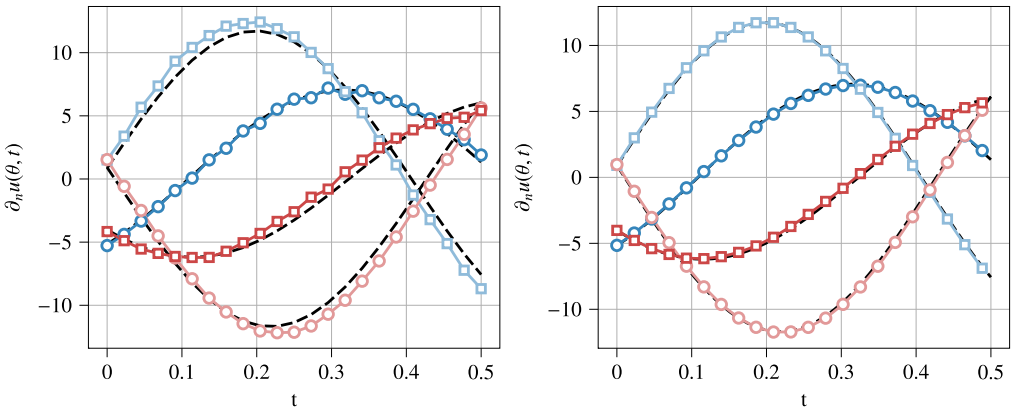
Fig. 13a plots the maximum error in the final time solution as a function of the spatial resolution N_x, demonstrating third-order spatial convergence with no order reduction in the presence of a moving boundary. The error magnitudes are nearly identical to those of the equivalent stationary boundary cases, shown as a dashed line. Fig. 13b illustrates the final time velocity field and error field for $N_x = 128$, with the path of the body center superimposed. The maximum error occurs in the interior of the domain, and the effect of the Dirichlet boundary condition can be observed as a region of diminished error in the upper right hand corner. This is consistent with the behavior of stationary cases, in which the error has a minimum at Dirichlet inflow boundaries and grows as the solution is transported along characteristics into the interior of the domain. For this test case the moving boundary treatment acts as a source of error with magnitude well below that of the interior truncation error, and has a minimal effect on the final time error magnitude.

4.4. Surface quantities

One of the advantages of sharp immersed methods over diffuse immersed methods is the ability to accurately reconstruct quantities defined on an immersed surface. To illustrate this capability, we return to the 2D moving advection test case presented in section 4.3.3 with flow velocity $\mathbf{v} = [0.6, -0.6]$ perpendicular to the boundary motion, and extract the normal gradient of the solution on the boundary using the same fourth-order accurate weighted least squares interpolants that are used to discretize the advection term. The results at two different spatial resolutions are presented in Fig. 14a. For $\Delta x = 1/32$ the surface gradient closely follows



(a) Spatial convergence of the normal gradient distribution at $t = T$



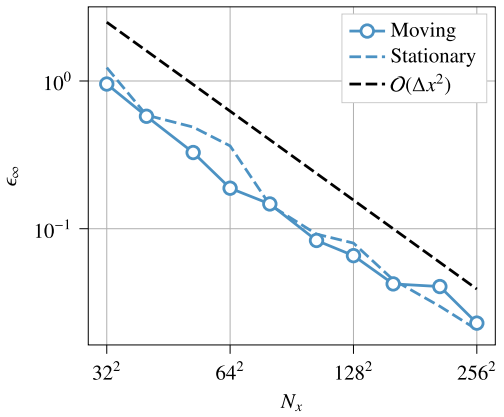
(b) Time history of the normal gradient at $N_x = 32$ at four control points (c) Time history of the normal gradient at $N_x = 64$ at four control points

Fig. 14. Results from section 4.4. (a) The surface normal gradient at $t = T$ for the 2D advection test case from section 4.3.3. Data for $N_x = 64$ is offset for readability. (b) Time history of the surface normal gradient with $N_x = 1/32$. The four data series correspond to control points with $x_2 = 3/8$ (—○— and —□—) and $x_2 = 5/8$ (—○— and —□—), with the first point in each pair satisfying $x_1 < x_{b,1}(t)$ (on the 'trailing' side) and the second point satisfying $x_1 > x_{b,1}(t)$ (on the 'leading' side). (c) Time history of the surface normal gradient with $N_x = 1/64$ for the same four control points. All data for this spatial resolution is downsampled by a factor of two for readability.

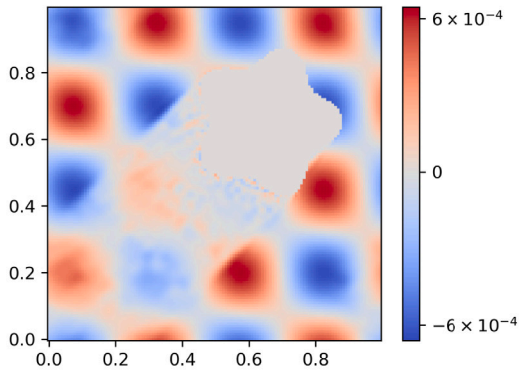
the exact value, even though at this resolution the star shaped body has only approximately five grid points across each of its lobes (see Fig. 12b). For $\Delta x = 1/64$, a more moderate resolution, the exact and numerical values are nearly indistinguishable. To examine the temporal history of surface gradients we consider a second test case with the same velocity field and a star-shaped body initially centered at $\mathbf{x}_0 = [0.351, 0.503]$ that translates with body velocity $\mathbf{v}_b = 0.6\hat{\mathbf{e}}_1$. For this case the control points that lie on x_1 direction grid lines coincide with the same material point on the boundary at each time step, so that it is possible to extract the time history of the normal gradient at a single material point. Fig. 14b provides this time history for material points located on the grid lines $x_2 = 3/8$ and $x_2 = 5/8$ using a spatial resolution of $\Delta x = 1/32$; the equivalent results for $\Delta x = 1/64$ are shown in Fig. 14c. Neither case exhibits large temporal oscillations in the surface gradient, and with a resolution of $\Delta x = 1/64$ the maximum error at any of the four sample points is less than 5% of the maximum normal gradient observed at that point.

To assess the convergence of the surface normal gradient, we return to the advection test case from section 4.3.3 with velocity $\mathbf{v} = [-0.6, -0.6]$ that opposes the motion of the body. Fig. 15a plots the maximum error in the surface normal gradient at the final time as a function of the spatial resolution N_x. The results indicate second order convergence for moving boundary simulations, with error magnitudes that closely follow those of the equivalent stationary boundary simulations with geometry taken from the final time $t = T$. This is a reduction in order compared to the solution error, which is third order in the L_∞ norm. Fig. 15b provides the final time error field for this test case with $N_x = 128$, which indicates that the moving boundary in this test case acts as a source of error that is not smooth either in space or in time, but with magnitude approximately equal to that of the interior scheme. The result is a "wake" region behind the body in which the error has a magnitude that is $\mathcal{O}(\Delta x^3)$ and varies over a length scale that is $\mathcal{O}(\Delta x)$, leading to $\mathcal{O}(\Delta x^2)$ convergence for the normal gradient.

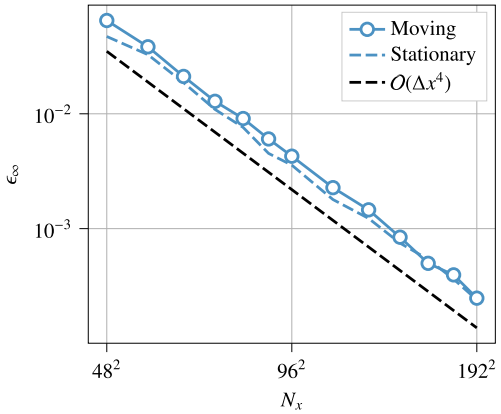
We also consider the convergence of surface quantities for parabolic PDEs, returning to the 2D diffusion test case with Dirichlet boundary conditions presented in section 4.3.1. Fig. 15c plots the maximum error in the surface normal gradient at the final time, using a (4, 5) IIM discretization and the same spatial resolutions and parameters as the convergence results in Fig. 9a. The convergence of the normal gradient is fourth order, and error magnitudes agree well with those of the equivalent stationary cases. Fig. 15d provides the magnitude of the final time error on a logarithmic scale for $N_x = 72$, which indicates that the errors are smooth



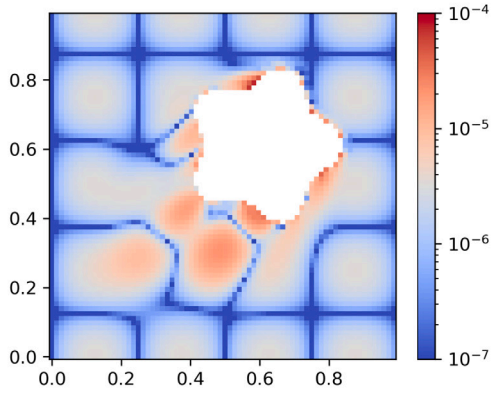
(a) Convergence of the normal gradient for a 2D advection test case.



(b) Final time error field for $N_x = 128$



(c) Convergence of the normal gradient for a 2D diffusion test case.



(d) Final time error field for $N_x = 72$

Fig. 15. Convergence results for the surface normal gradient (section 4.4). (a) Second order convergence for the normal gradient in a 2D advection test case with Dirichlet boundaries. The error magnitudes agree well with those of the equivalent stationary cases, shown as a dashed line. (b) The final time error field for the (3, 4) advection discretization with $N_x = 128$, which indicates a non-smooth error. (c) Fourth order convergence for the normal gradient in a 2D diffusion test case with Dirichlet boundary conditions. (d) The final time error field for the (4, 5) diffusion discretization with $N_x = 72$, shown on a logarithmic scale. The solution error is smooth and concentrated on the boundaries.

and concentrated on the boundaries. This is consistent with the increased truncation error in that region. As this boundary error source moves across the domain, a "wake" of increased error magnitude forms behind the body and diffuses into adjacent regions of the domain. Because the error field is smooth, the surface normal gradient achieves the same fourth order convergence rate as the solution error.

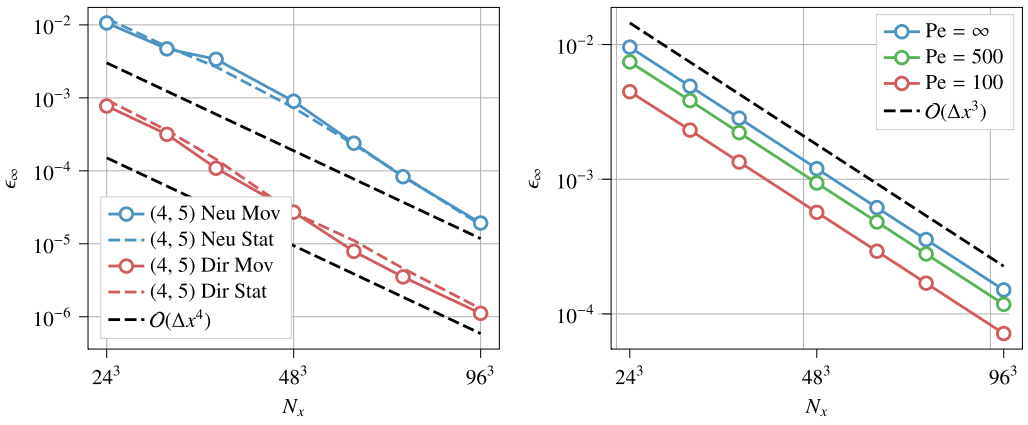
4.5. 3D convergence results

For 3D simulations, the sharp immersed method presented in sections 2 and 3 has been implemented within MURPHY [62], an open source software framework for PDE simulations on block-structured multiresolution grids. Following the methodology used for 2D simulations, we measure the convergence of 3D IIM discretizations via the method of manufactured solutions on the computational domain $[0,1]^3$ with periodic domain boundaries. The problem domain is the exterior of a sphere that is radially perturbed by a distance proportional to a spherical harmonic. In a spherical coordinate system (ρ, θ, φ) that is aligned with the x_3 axis and centered on the moving point $\mathbf{x}_b(t) = \mathbf{x}_0 + \mathbf{v}_b t$, this region is defined by the level set

$$\begin{aligned} \phi(\rho,\theta,\varphi) &= -(\rho-\rho_0) + \tilde{\rho} Y_5^{10}(\theta,\varphi), \\ Y_5^{10}(\theta,\varphi) &= -\frac{3}{256}\sqrt{\frac{1001}{\pi}}\cos(5\varphi)\sin^5(\theta)\left(323\cos^5(\theta) - 170\cos^3(\theta) + 15\cos(\theta)\right). \end{aligned} \tag{31}$$

In this section we use a uniform Cartesian grid and a body with average radius $\rho_0 = 0.25$, perturbation radius $\tilde{\rho} = 0.02$, initial center $\mathbf{x}_0 = [0.355, 0.501, 0.509]$, and body velocity $\mathbf{v}_b = 0.301\hat{\mathbf{e}}_1$.

For 3D convergence tests we solve the advection-diffusion equation $\partial_t u + \mathbf{v} \cdot \nabla u = \beta \nabla^2 u$, which reduces to pure diffusion if $\mathbf{v} = 0$ and to pure advection if $\beta = 0$. In all cases, we use a manufactured solution



(a) Diffusion with Dirichlet or Neumann boundary conditions. (b) Advection-diffusion with Dirichlet boundary conditions.

Fig. 16. (a) Spatial convergence results for 3D diffusion with Dirichlet or Neumann boundary conditions (section 4.5). Dashed lines represent equivalent stationary cases. For both boundary conditions the convergence is fourth order or higher in the L_∞ norm, and the error magnitudes for the moving and stationary simulations are nearly identical. (b) Spatial convergence for the advection-diffusion equation with varying Peclet number and Dirichlet boundary conditions. Convergence is third order in the L_∞ norm.

$$g(\mathbf{x},t) = g_0(\mathbf{x}-\mathbf{v}t,t), \qquad g_0(\mathbf{x},t) = \exp(-\beta\|\mathbf{k}\|^2 t)\sin\left(k_1 x_1\right)\sin\left(k_2 x_2\right)\sin\left(k_3 x_3\right). \tag{32}$$

All time integration runs from $t=0$ to $T=1$, with the time step constrained by both a maximum body CFL of $C_b = 0.4$ and a stability criterion chosen for each finite difference scheme. For pure diffusion we use diffusivity $\beta = 5.85\times10^{-3}$ and wavenumbers $k_i = 2\pi$. The time step is constrained by a maximum Fourier number of $r=0.39$ for the $(4,5)$ discretization and $r=0.34$ for the $(6,7)$ discretization, which ensures stability when paired with the low-storage RK4()6[2S] integrator from [63] that is optimized for real-line stability. For pure advection we take $\beta=0$ and use a constant velocity field $\mathbf{v}=[0.31,0.32,0.28]$, integrating in time with LSRK$(3,3)$ and a maximum flow CFL number of $C_v = 1.0$. Finally, for the advection-diffusion equation we use the same velocity and define the diffusivity in terms of the Peclet number $\mathrm{Pe}_L = \|\mathbf{v}\|L/\beta$ based on the domain length $L=1$. For these cases the advection term is computed using a boundary condition only at inflow boundaries, while both the diffusion term and the moving boundary extrapolations use a boundary condition at all boundary points. Time integration is performed with LSRK$(5,4)$ and a time step that is chosen to enforce a maximum flow CFL of $C_v = 0.8$ and maximum Fourier number of $r=0.2$. For all cases we report the L_∞ norm of the error in the final time solution.

Fig. 16a provides spatial convergence results for a $(4,5)$ diffusion discretization with either Dirichlet or Neumann boundary conditions, as well as for the equivalent stationary cases using the same manufactured solution and the final time geometry. In both cases the convergence rates are fourth order or better in the L_∞ norm, and the magnitude of the errors is nearly identical to the stationary cases. Fig. 16b provides spatial convergence results for pure advection ($\mathrm{Pe}_L = \infty$) as well as for the advection-diffusion equation with $\mathrm{Pe}_L = 500$ and $\mathrm{Pe}_L = 100$. All cases use a Dirichlet boundary condition, prescribed at inflows for the advection equation and on all boundaries for advection-diffusion. Advection terms are discretized with the $(3,4)$ advection scheme, while diffusion terms are discretized with the $(4,5)$ discretization, leading to a scheme which is formally third order accurate. For all three of these cases the dominant error comes from the advection term, and as a result the convergence is third order in the L_∞ norm.

4.6. 3D multiresolution advection-diffusion

As a final test case, we consider the 3D advection-diffusion equation with moving boundaries and simulation parameters that lead to the formation of thin boundary layers. The fluid domain is a box with side lengths $L_1 = L_2 = 2$ and $L_3 = 4$, with an immersed spherical body of radius $R=0.51$ that is initially centered at $\mathbf{x}_0 = [1,\,1,\,3]$ and translates with constant velocity $v_b\hat{\mathbf{e}}_3$ with $v_b = -2$. The flow field $\mathbf{v}(\mathbf{x})$ is the potential flow about a moving sphere with no free stream velocity; in a spherical coordinate system (ρ,θ,φ) centered on $\mathbf{x}_b(t)$ and aligned with the x_3 axis,

$$\mathbf{v}(\mathbf{x}) = -v_b\left(1-\frac{R^3}{\rho^3}\right)\cos(\varphi)\hat{\boldsymbol{\rho}} + v_b\left(1+\frac{1}{2}\frac{R^3}{\rho^3}\right)\sin(\varphi)\hat{\boldsymbol{\varphi}} + v_b\hat{\mathbf{e}}_3. \tag{33}$$

The initial condition is a compact Gaussian

$$u(\mathbf{x},0) = u_0\exp\left(\frac{-z^2}{1-z^2}\right), \quad z \equiv \frac{\left\|\mathbf{x}-\mathbf{x}_g\right\|}{\sigma_g}, \tag{34}$$

with center $\mathbf{x}_g = [1.0,\,0.8,\,2.0]$, width $\sigma_g = 0.3$, and magnitude $u_0 = -1$. In order to produce a boundary layer on the sphere surface, a constant Dirichlet boundary condition $u_b = -u_0$ or Neumann boundary condition $\partial_n u = 2.6u_0/R$ is prescribed on the immersed solid boundary, and the simulation is run with Peclet number $\mathrm{Pe}_D = 2v_bR/\beta = 1000$. Anticipating that the support of $u(\mathbf{x},t)$ will be

(a) $t = 0$ (b) $t = 0.25$ (c) $t = 0.5$ (d) $t = 0.75$

Fig. 17. A rendering of the 3D advection-diffusion test case with Neumann boundary conditions (section 4.6). The sphere surface is colored based on surface values of the scalar field u, while contours indicate the level sets for $u = \{-0.4, -0.3, -0.2, -0.1, -0.01, 0.01, 0.1, 0.2, 0.3\}$. The body motion is from left to right, with a negative Gaussian distribution that begins to the right of the sphere and passes below the sphere as time progresses. Contours indicate the formation of a thin boundary layer on the sphere surface.

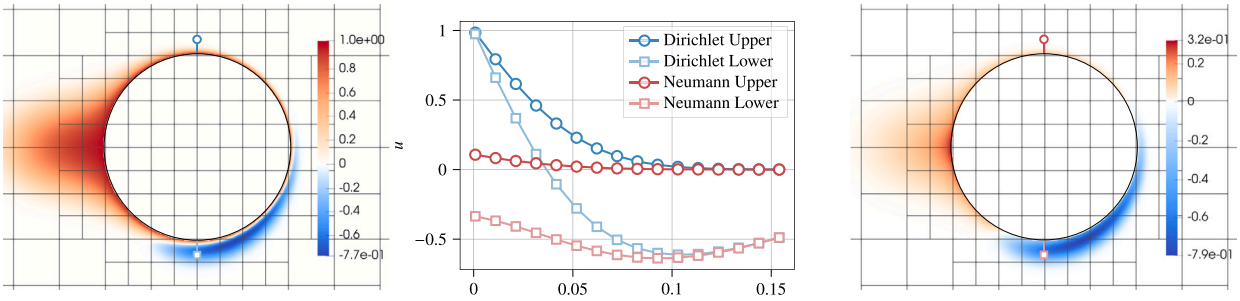


Fig. 18. The scalar distribution $u(\mathbf{x}, t)$ from the 3D advection-diffusion test with Dirichlet (left) or Neumann (right) boundary conditions; see section 4.6. Samples are take at time $t = 0.5$ from the plane $x_0 = 1$. Gray lines indicate the multiresolution grid, with each block containing 24^2 points. Samples from the upper and lower boundary layer over the lines indicated on each slice are shown in the central plot, with each marker representing data from individual grid points that are separated by a distance $\Delta x = 1/768$.

contained within the computational domain for the duration of the simulation, we enforce domain boundary conditions by prescribing zero values at two layers of ghost points adjacent to the computational domain. The spatial discretization of the advection-diffusion equation is the same as in section 4.5, while time stepping is performed with LSRK(3, 3) and a body CFL of $C_b = 0.32$. To increase the accuracy of the simulation near the immersed surface, the simulation is conducted on a block-based multiresolution grid that is continually adapted to both the body motion and the scalar $u(\mathbf{x}, t)$. Both grid adaptation and ghost point reconstruction for each block are performed with sixth order interpolating wavelets [62], so that the fourth order accuracy of the diffusion term is maintained across resolution boundaries.

Fig. 17 provides renderings of the scalar field $u(\mathbf{x}, t)$ at times $t = \{0, 0.25, 0.5, 0.75\}$ for the case of Neumann boundary conditions. As the spherical body translates, the Neumann boundary condition acts as a source of scalar on the body surface, which forms a thin boundary layer that is transported over the surface and into the domain. Simultaneously, the initial Gaussian distribution passes over the surface of the sphere and interacts with the boundary layer. Fig. 18 displays slices in the plane $x_1 = 1$ taken from both the Dirichlet and Neumann cases at $t = 0.5$, which illustrate both the solution and the multiresolution grid used to concentrate computational elements near the surface of the sphere. Also shown are samples taken at the same time along a line passing through the sphere center and perpendicular to the direction of travel. These samples indicate that the boundary layer profile is smooth and well resolved, with seven to eight grid points across the boundary layer thickness. While this resolution would require $768 \times 768 \times 1536 \approx 9.1 \times 10^8$ points on a uniform grid, the total number of grid points on the multiresolution grid is on average 2.9×10^7 during the simulation, giving an average compression ratio of roughly 31.

5. Conclusion

We have presented a high-order sharp immersed discretization of the advection-diffusion equation for simulations with moving domain boundaries or material interfaces. Spatial operators are discretized with a combination of high-order dimension-split finite difference schemes and high-order weighted least-squares interpolants used to construct ghost values near immersed surfaces. For moving surfaces, the issue of freshly cleared cells is addressed by extrapolating both the state field and its time derivative beyond the domain boundary, which provides an artificial time history that is compatible with a wide range of explicit and diagonally-implicit Runge-Kutta time integrators. This strategy introduces a mixed spatial-temporal error term that is first order in time and higher order in space, which can dominate the existing temporal error term in purely temporal convergence tests and is most easily observable in simulations with low spatial resolution. However, extensive numerical experiments indicate that this mixed error term rarely dominates the existing spatial error terms, and does not affect the convergence order of the free-space difference scheme when the time step is chosen to satisfy a flow CFL, Fourier number, or body CFL constraint. This allows for the construction of immersed schemes with up to third order convergence for hyperbolic PDEs and up to sixth order convergence for parabolic PDEs in

both 2D and 3D domains with moving boundaries and interfaces. We demonstrate that these schemes can accurately predict surface quantities on a moving immersed surface without oscillations even at low spatial resolutions, and that thin boundary layers on immersed surfaces can be effectively captured by combining a high-order immersed discretization with high-order multiresolution grid adaptation.

Our results present for the first time a quantitative description of the errors associated with sharply discretized moving surfaces that holds for multiple combinations of PDEs, boundary conditions, spatial discretizations, and time integrators. Compared to previous moving boundary treatments, our extrapolation-based approach is applicable to a wider range time integrators, both in terms of the type of integrator as well as the order of accuracy. The spatial accuracy of our immersed method matches the previously demonstrated third order convergence rate for 2D hyperbolic PDEs with moving boundaries [17], and outperforms the previously demonstrated fourth-order convergence rate for 2D diffusion problems with moving boundaries [4]. Finally, we extend these levels of accuracy for the first time to 2D diffusion simulations with moving two-sided interfaces and 3D advection-diffusion simulations with a moving nonconvex immersed body.

While the algorithms presented here are described and analyzed for simulations with prescribed surface motion, we expect that they will also be useful for interface-coupled multiphysics simulations with physics-driven surface motion such as ablation or fluid-structure interaction. For these systems a discretization must allow for physics on both sides of an immersed interface while accurately estimating the surface quantities that drive interface motion, both of which are capabilities demonstrated in our results. The errors introduced by our treatment of freshly cleared cells are likely to appear in any simulation with surface motion, and as a result we view the convergence results presented in section 4 as an upper bound on what is possible when applying our sharp immersed method to interface-coupled multiphysics systems. However, further work is necessary to determine if the proposed freshly-cleared cell treatment maintains its accuracy in the presence of weakly or strongly coupled boundary motion.

CRediT authorship contribution statement

James Gabbard: Conceptualization, Formal analysis, Investigation, Methodology, Software, Validation, Visualization, Writing – original draft, Writing – review & editing. **Wim M. van Rees:** Conceptualization, Funding acquisition, Investigation, Methodology, Project administration, Resources, Supervision, Validation, Writing – original draft, Writing – review & editing.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

Data will be made available on request.

Acknowledgements

We wish to acknowledge financial support from an Early Career Award from the Department of Energy, Program Manager Dr. Steven Lee, award number DE-SC0020998.

This research used resources of the National Energy Research Scientific Computing Center (NERSC), a U.S. Department of Energy Office of Science User Facility located at Lawrence Berkeley National Laboratory, operated under Contract No. DE-AC02-05CH11231 using National Energy Research Scientific Computing Center award ASCR-ERCAP0023392.

Appendix A. Additional temporal convergence data for 1D diffusion

Figs. A.19 and A.20 provide the full convergence data for the 1D diffusion test cases presented in sections 4.1.2 and 4.1.3, respectively, which are used to generate the spatial convergence exponents listed in Table 1.

References

[1] C.S. Peskin, Flow patterns around heart valves: a numerical method, J. Comput. Phys. 10 (1972) 252–271, https://doi.org/10.1016/0021-9991(72)90065-4.

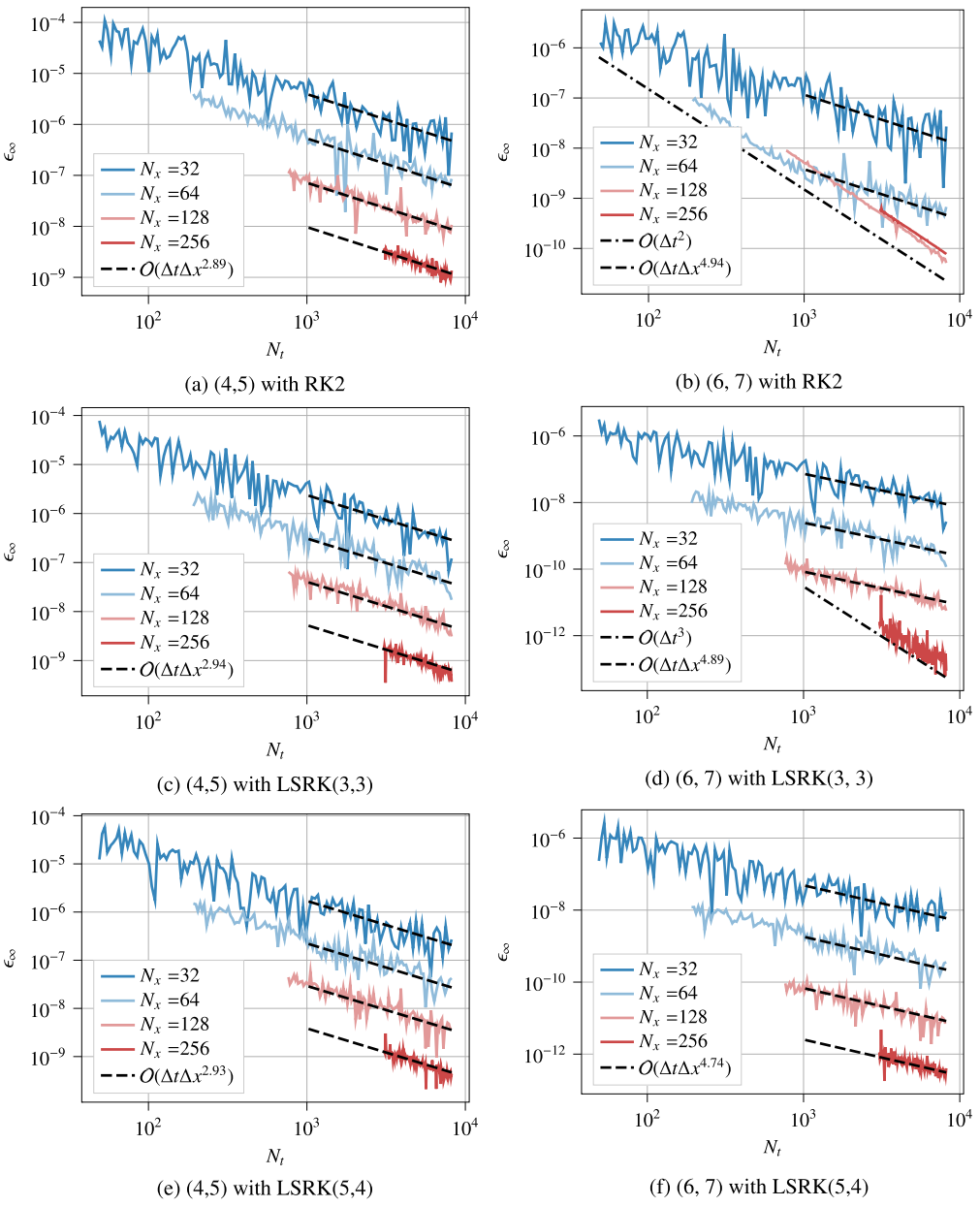
[2] R.J. LeVeque, Z. Li, The immersed interface method for elliptic equations with discontinuous coefficients and singular sources, SIAM J. Numer. Anal. 31 (1994) 1019–1044, https://doi.org/10.1137/0731054.

[3] R.P. Fedkiw, T. Aslam, B. Merriman, S. Osher, A non-oscillatory Eulerian approach to interfaces in multimaterial flows (the ghost fluid method), J. Comput. Phys. 152 (1999) 457–492, https://doi.org/10.1006/jcph.1999.6236.

[4] F. Gibou, R. Fedkiw, A fourth order accurate discretization for the Laplace and heat equations on arbitrary domains, with applications to the Stefan problem, J. Comput. Phys. 202 (2005) 577–601, https://doi.org/10.1016/j.jcp.2004.07.018.

[5] P.T. Brady, D. Livescu, Foundations for high-order, conservative cut-cell methods: stable discretizations on degenerate meshes, J. Comput. Phys. 426 (2021) 109794, https://doi.org/10.1016/j.jcp.2020.109794.

[6] N. Sharan, P.T. Brady, D. Livescu, High-order dimensionally-split Cartesian embedded boundary method for non-dissipative schemes, J. Comput. Phys. 464 (2022) 111341, https://doi.org/10.1016/j.jcp.2022.111341.



(a) (4,5) with RK2

(b) (6, 7) with RK2

(c) (4,5) with LSRK(3,3)

(d) (6, 7) with LSRK(3, 3)

(e) (4,5) with LSRK(5,4)

(f) (6, 7) with LSRK(5,4)

Fig. A.19. Temporal convergence for the 1D diffusion equation with moving boundaries and Neumann boundary conditions (section 4.1.2). The spatial discretization is fourth order (left column) or sixth order (right column) and time integration is performed with RK2, LSRK(3, 3), or LSRK(5, 4) time integration. An $\mathcal{O}(\Delta t x^N)$ moving boundary error term is observed for each discretization, and a least squares fit is constructed through selected convergence data to determine the exponent N. The fit and the data used to construct it are indicated one each plot with a set of black dashed lines.

[7] C. Brehm, M.F. Barad, C. Hader, A high-order immersed interface method for compressible flows, in: 7th AIAA Theoretical Fluid Mechanics Conference, 2014, p. 2093.

[8] J. Gabbard, W.M. van Rees, A high-order 3D immersed interface finite difference method for the advection-diffusion equation, in: AIAA SCITECH 2023 Forum, 2023, p. 2480.

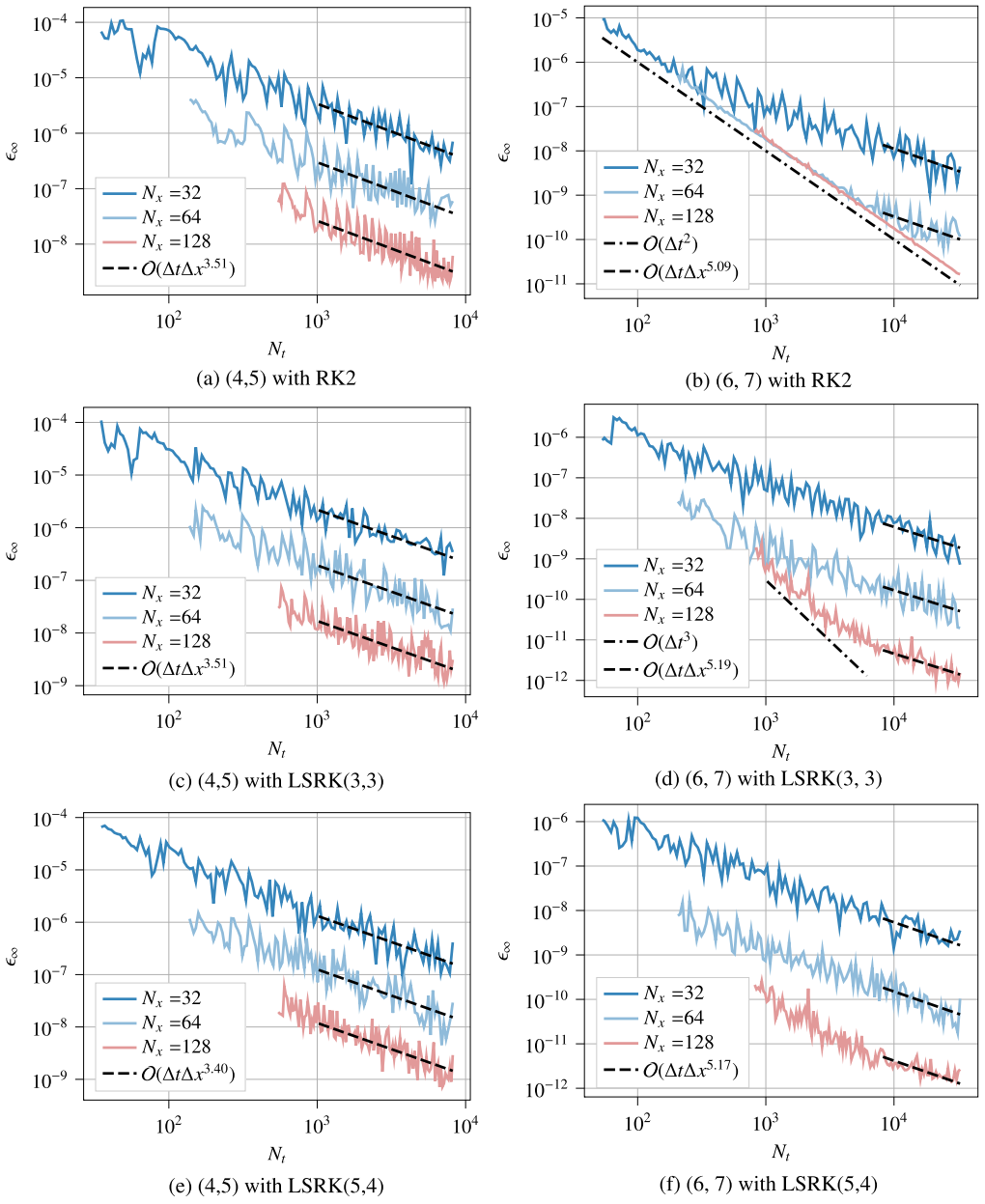
[9] M.N. Linnick, H.F. Fasel, A high-order immersed interface method for simulating unsteady incompressible flows on irregular domains, J. Comput. Phys. 204 (2005) 157–192, https://doi.org/10.1016/j.jcp.2004.09.017.

[10] C. Zhu, H. Luo, G. Li, High-order immersed-boundary method for incompressible flows, AIAA J. 54 (2016) 2734–2741, https://doi.org/10.2514/1.J054628.

[11] S. Hosseinverdi, H.F. Fasel, Very high-order accurate sharp immersed interface method: application to direct numerical simulations of incompressible flows, in: 23rd AIAA Computational Fluid Dynamics Conference, 2017, p. 3624.

[12] S. Hosseinverdi, H.F. Fasel, An efficient, high-order method for solving Poisson equation for immersed boundaries: combination of compact difference and multiscale multigrid methods, J. Comput. Phys. 374 (2018) 912–940, https://doi.org/10.1016/j.jcp.2018.08.006.

[13] R. Singhal, J.C. Kalita, An efficient explicit jump high-order compact immersed interface approach for transient incompressible viscous flows, Phys. Fluids 34 (2022), https://doi.org/10.1063/5.0107308.



(a) (4,5) with RK2

(b) (6, 7) with RK2

(c) (4,5) with LSRK(3,3)

(d) (6, 7) with LSRK(3, 3)

(e) (4,5) with LSRK(5,4)

(f) (6, 7) with LSRK(5,4)

Fig. A.20. Temporal convergence for the 1D diffusion equation with moving boundaries and discontinuous diffusivity (section 4.1.3). The spatial discretization is fourth order (left column) or sixth order (right column) and time integration is performed with RK2, LSRK(3, 3), or LSRK(5, 4) time integration. An $\mathcal{O}(\Delta t x^N)$ moving boundary error term is observed for each discretization, and a least squares fit is constructed through selected convergence data to determine the exponent N. The fit and the data used to construct it are indicated one each plot with a set of black dashed lines.

[14] J.H. Seo, R. Mittal, A high-order immersed boundary method for acoustic wave scattering and low-Mach number flow-induced sound in complex geometries, J. Comput. Phys. 230 (2011) 1000–1019, https://doi.org/10.1016/j.jcp.2010.10.017.

[15] S. Hosseinverdi, H.F. Fasel, A fourth-order accurate compact difference scheme for solving the three-dimensional Poisson equation with arbitrary boundaries, in: AIAA Scitech 2020 Forum, 2020, p. 0805.

[16] X. Zeng, C. Farhat, A systematic approach for constructing higher-order immersed boundary and ghost fluid methods for fluid–structure interaction problems, J. Comput. Phys. 231 (2012) 2892–2923, https://doi.org/10.1016/j.jcp.2011.12.027.

[17] S. Tan, C.-W. Shu, A high order moving boundary treatment for compressible inviscid flows, J. Comput. Phys. 230 (2011) 6023–6036, https://doi.org/10.1016/j.jcp.2011.04.011.

[18] S. Tan, C. Wang, C.-W. Shu, J. Ning, Efficient implementation of high order inverse Lax–Wendroff boundary treatment for conservation laws, J. Comput. Phys. 231 (2012) 2510–2527, https://doi.org/10.1016/j.jcp.2011.11.037.

[19] S. Tan, C.-W. Shu, Inverse Lax–Wendroff procedure for numerical boundary conditions of hyperbolic equations: survey and new developments, in: Advances in Applied Mathematics, Modeling, and Computational Science, Springer, 2013, pp. 41–63.

[20] T. Li, C.-W. Shu, M. Zhang, Stability analysis of the inverse Lax–Wendroff boundary treatment for high order upwind-biased finite difference schemes, J. Comput. Appl. Math. 299 (2016) 140–158, https://doi.org/10.1016/j.cam.2015.11.038.

[21] J. Lu, J. Fang, S. Tan, C.-W. Shu, M. Zhang, Inverse Lax–Wendroff procedure for numerical boundary conditions of convection–diffusion equations, J. Comput. Phys. 317 (2016) 276–300, https://doi.org/10.1016/j.jcp.2016.04.059.
[22] J. Lu, C.-W. Shu, S. Tan, M. Zhang, An inverse Lax-Wendroff procedure for hyperbolic conservation laws with changing wind direction on the boundary, J. Comput. Phys. 426 (2021) 109940, https://doi.org/10.1016/j.jcp.2020.109940.
[23] D. Devendran, D. Graves, H. Johansen, T. Ligocki, A fourth-order Cartesian grid embedded boundary method for Poisson's equation, Commun. Appl. Math. Comput. Sci. 12 (2017) 51–79, https://doi.org/10.2140/camcos.2017.12.51.
[24] N. Overton-Katz, X. Gao, S.M. Guzik, O. Antepara, D. Graves, H. Johansen, Towards a high-order embedded boundary finite volume method for the incompressible Navier–Stokes equations with complex geometries, in: AIAA SCITECH 2022 Forum, 2022, p. 2202.
[25] N. Overton-Katz, X. Gao, S. Guzik, O. Antepara, D.T. Graves, H. Johansen, A fourth-order embedded boundary finite volume method for the unsteady Stokes equations with complex geometries, SIAM J. Sci. Comput. 45 (2023) A2409–A2430, https://doi.org/10.1137/22M1532019.
[26] G. Guarino, V. Gulizzi, A. Milazzo, Accurate multilayered shell buckling analysis via the implicit-mesh discontinuous Galerkin method, AIAA J. 60 (2022) 6854–6868, https://doi.org/10.2514/1.J061933.
[27] V. Gulizzi, A.S. Almgren, J.B. Bell, A coupled discontinuous Galerkin-finite volume framework for solving gas dynamics over embedded geometries, J. Comput. Phys. 450 (2022) 110861, https://doi.org/10.1016/j.jcp.2021.110861.
[28] V. Gulizzi, R. Saye, Modeling wave propagation in elastic solids via high-order accurate implicit-mesh discontinuous Galerkin methods, Comput. Methods Appl. Mech. Eng. 395 (2022) 114971, https://doi.org/10.1016/j.cma.2022.114971.
[29] R.I. Saye, High-order quadrature methods for implicitly defined surfaces and volumes in hyperrectangles, SIAM J. Sci. Comput. 37 (2015) A993–A1019, https://doi.org/10.1137/140966290.
[30] R.I. Saye, High-order quadrature on multi-component domains implicitly defined by multivariate polynomials, J. Comput. Phys. 448 (2022) 110720, https://doi.org/10.1016/j.jcp.2021.110720.
[31] H. Udaykumar, R. Mittal, W. Shyy, Computation of solid–liquid phase fronts in the sharp interface limit on fixed grids, J. Comput. Phys. 153 (1999) 535–574, https://doi.org/10.1006/jcph.1999.6294.
[32] Z. Li, Immersed interface methods for moving interface problems, Numer. Algorithms 14 (1997) 269–293, https://doi.org/10.1023/A:1019173215885.
[33] D.-V. Le, B.C. Khoo, J. Peraire, An immersed interface method for viscous incompressible flows involving rigid and flexible boundaries, J. Comput. Phys. 220 (2006) 109–138, https://doi.org/10.1016/j.jcp.2006.05.004.
[34] C. Brehm, H. Fasel, Immersed interface method for solving the incompressible Navier–Stokes equations with moving boundaries, in: 49th AIAA Aerospace Sciences Meeting Including the New Horizons Forum and Aerospace Exposition, 2011, p. 758.
[35] S. Xu, Z.J. Wang, An immersed interface method for simulating the interaction of a fluid with moving boundaries, J. Comput. Phys. 216 (2006) 454–493, https://doi.org/10.1016/j.jcp.2005.12.016.
[36] H. Udaykumar, R. Mittal, P. Rampunggoon, A. Khanna, A sharp interface Cartesian grid method for simulating flows with complex moving boundaries, J. Comput. Phys. 174 (2001) 345–380, https://doi.org/10.1006/jcph.2001.6916.
[37] A. Gilmanov, F. Sotiropoulos, A hybrid Cartesian/immersed boundary method for simulating flows with 3D, geometrically complex, moving bodies, J. Comput. Phys. 207 (2005) 457–492, https://doi.org/10.1016/j.jcp.2005.01.020.
[38] R. Mittal, H. Dong, M. Bozkurttas, F. Najjar, A. Vargas, A. Von Loebbecke, A versatile sharp interface immersed boundary method for incompressible flows with complex boundaries, J. Comput. Phys. 227 (2008) 4825–4852, https://doi.org/10.1016/j.jcp.2008.01.028.
[39] L. Ge, F. Sotiropoulos, A numerical method for solving the 3D unsteady incompressible Navier–Stokes equations in curvilinear domains with complex immersed boundaries, J. Comput. Phys. 225 (2007) 1782–1809, https://doi.org/10.1016/j.jcp.2007.02.017.
[40] I. Borazjani, L. Ge, F. Sotiropoulos, Curvilinear immersed boundary method for simulating fluid structure interaction with complex 3D rigid bodies, J. Comput. Phys. 227 (2008) 7587–7620, https://doi.org/10.1016/j.jcp.2008.04.028.
[41] D. Angelidis, S. Chawdhary, F. Sotiropoulos, Unstructured Cartesian refinement with sharp interface immersed boundary method for 3D unsteady incompressible flows, J. Comput. Phys. 325 (2016) 272–300, https://doi.org/10.1016/j.jcp.2016.08.028.
[42] C. Brehm, M.F. Barad, C.C. Kiris, Development of immersed boundary computational aeroacoustic prediction capabilities for open-rotor noise, J. Comput. Phys. 388 (2019) 690–716, https://doi.org/10.1016/j.jcp.2019.02.011.
[43] J. Boustani, M.F. Barad, C.C. Kiris, C. Brehm, An immersed boundary fluid–structure interaction method for thin, highly compliant shell structures, J. Comput. Phys. 438 (2021) 110369, https://doi.org/10.1016/j.jcp.2021.110369.
[44] J. Yang, E. Balaras, An embedded-boundary formulation for large-eddy simulation of turbulent flows interacting with moving boundaries, J. Comput. Phys. 215 (2006) 12–40, https://doi.org/10.1016/j.jcp.2005.10.035.
[45] J.H. Williamson, Low-storage Runge–Kutta schemes, J. Comput. Phys. 35 (1980) 48–56, https://doi.org/10.1016/0021-9991(80)90033-9.
[46] C.-W. Shu, S. Osher, Efficient implementation of essentially non-oscillatory shock-capturing schemes, J. Comput. Phys. 77 (1988) 439–471, https://doi.org/10.1016/0021-9991(88)90177-5.
[47] M.H. Carpenter, C.A. Kennedy, Fourth-order 2N-storage Runge–Kutta schemes, NASA Technical Memorandum 109112, 1994.
[48] S. Ruuth, Global optimization of explicit strong-stability-preserving Runge–Kutta methods, Math. Comput. 75 (2006) 183–207.
[49] J.H. Verner, Explicit Runge–Kutta methods with estimates of the local truncation error, SIAM J. Numer. Anal. 15 (1978) 772–790, https://doi.org/10.1137/0715051.
[50] H. Johansen, P. Colella, A Cartesian grid embedded boundary method for Poisson's equation on irregular domains, J. Comput. Phys. 147 (1998) 60–85, https://doi.org/10.1006/jcph.1998.5965.
[51] L. Ferracina, M. Spijker, An extension and analysis of the Shu-Osher representation of Runge–Kutta methods, Math. Comput. 74 (2005) 201–219.
[52] S. Gottlieb, D. Ketcheson, C.-W. Shu, Strong Stability Preserving Runge–Kutta and Multistep Time Discretizations, World Scientific, 2011.
[53] I. Higueras, Representations of Runge–Kutta methods and strong stability preserving methods, SIAM J. Numer. Anal. 43 (2005) 924–948, https://doi.org/10.1137/S0036142903427068.
[54] J. Gabbard, T. Gillis, P. Chatelain, W.M. van Rees, An immersed interface method for the 2D vorticity-velocity Navier–Stokes equations with multiple bodies, J. Comput. Phys. 464 (2022) 111339, https://doi.org/10.1016/j.jcp.2022.111339.
[55] D. Cavaglieri, T. Bewley, Low-storage implicit/explicit Runge–Kutta schemes for the simulation of stiff high-dimensional ODE systems, J. Comput. Phys. 286 (2015) 172–193, https://doi.org/10.1016/j.jcp.2015.01.031.
[56] X. Ji, J. Gabbard, W.M. van Rees, A sharp immersed method for 2D flow-body interactions using the vorticity-velocity Navier-Stokes equations, J. Comput. Phys. 494 (2023) 112513, https://doi.org/10.1016/j.jcp.2023.112513.
[57] M.H. Carpenter, D. Gottlieb, S. Abarbanel, W.-S. Don, The theoretical accuracy of Runge–Kutta time discretizations for the initial boundary value problem: a study of the boundary error, SIAM J. Sci. Comput. 16 (1995) 1241–1252, https://doi.org/10.1137/0916072.
[58] D. Pathria, The correct formulation of intermediate boundary conditions for Runge–Kutta time integration of initial boundary value problems, SIAM J. Sci. Comput. 18 (1997) 1255–1266, https://doi.org/10.1137/S1064827594273948.
[59] R.R. Rosales, B. Seibold, D. Shirokoff, D. Zhou, Spatial manifestations of order reduction in Runge–Kutta methods for initial boundary value problems, arXiv preprint, arXiv:1712.00897, 2017, https://doi.org/10.48550/arXiv.1712.00897.
[60] D.I. Ketcheson, B. Seibold, D. Shirokoff, D. Zhou, DIRK schemes with high weak stage order, in: Spectral and High Order Methods for Partial Differential Equations, 2020, p. 453.

[61] A. Biswas, D.I. Ketcheson, B. Seibold, D. Shirokoff, Design of DIRK schemes with high weak stage order, Commun. Appl. Math. Comput. Sci. 18 (2023) 1–28, https://doi.org/10.2140/camcos.2023.18.1.
[62] T. Gillis, W.M. van Rees, MURPHY—a scalable multiresolution framework for scientific computing on 3D block-structured collocated grids, SIAM J. Sci. Comput. 44 (2022) C367–C398, https://doi.org/10.1137/21M141676X.
[63] D.I. Ketcheson, Runge–Kutta methods with minimum storage implementations, J. Comput. Phys. 229 (2010) 1763–1773, https://doi.org/10.1016/j.jcp.2009.11.006.