Seismic modeling with RBF-FD – a simplified treatment of interfaces

Authors:

Bradley Martin (University of Colorado-Boulder) Email: brma7253@colorado.edu

Bengt Fornberg (University of Colorado-Boulder) Email: Fornberg@colorado.edu

Submitted to the Journal of Computational Physics for peer review on ???

ABSTRACT

In a previous study of seismic modeling with RBF-FD, we outlined a numerical method for solving 2-D wave equations in domains with discontinuous interfaces between different regions. The method was applicable on a mesh-free set of data nodes. It included all information about interfaces within the weights of a stencil (allowing the use of traditional time integrators), and was shown to solve problems of the 2-D elastic wave equation to 3rd-order accuracy. In the present paper, we discuss a refinement of that method that makes it simpler to implement. It can also improve accuracy for the case of smoothly-variable model parameter values near interfaces. We give several test cases that demonstrate the method solving 2-D elastic wave equation problems to 4th-order accuracy, even in the presence of smoothly-curved interfaces with jump discontinuities in the model parameters.

INTRODUCTION

Numerical methods for solving wave equation problems are typically only 1st-order accurate in the presence of interfaces unless special correction techniques are incorporated ([1],[2]). In [3], we introduced a new radial basis function-derived finite difference (RBF-FD) method that achieved global 3rd-order accuracy in the presence of material interfaces. While the method shared some fundamental principles with other techniques developed for use in domains with interfaces, it also features a unique combination of advantages:

- as an RBF-FD method, it is applicable on a mesh-free set of data nodes that can be placed in an advantageous arrangement around interfaces, as explored in [4];
- we need not modify any material parameters near the interface (as done in [5] and [6]);
- stencils that cross an interface contain all relevant information about that interface within the stencil weights themselves, and any explicit integrator with an appropriate stability domain may be used to evolve a solution in time (e.g. RK4, AB3) without any further consideration of the interfaces during the time stepping (as was necessary in [7], [8]);
- these explicit stencils can be directly applied to data on both sides of an interface, requiring no formation of fictitious data extensions as in [9] and avoiding the necessary meshing inherent in the FEM approaches of [10] and [11];
- for each stencil that crosses the interface, all polynomial basis functions that support RBFs in determining the stencil's collocation weights are determined via a relatively simple matrix problem; and
- the method is designed, at least in theory, to support an arbitrary degree of spatial accuracy (though it may be practically limited by stability considerations).

In this paper, we will introduce a simplification of the method in [3]. Our current approach avoids having to form the null spaces of rectangular matrices in creating the piecewise smooth polynomial basis functions that support stencils that cross an interface. Instead, the refinement of our method requires only the inversion of small, square matrices. The present paper explains a simple way to explicitly determine the structure of every single data field in the 2-D elastic wave equation as that data crosses an arbitrarily-oriented interface, to any order of accuracy. This alteration of the method allows for the use of centered, symmetric stencils throughout the computational domain. We will also share a simple method for accommodating smoothly-variable model parameters on either side of an interface. We conclude with numerical examples which show that the method solves the 2-D EWE to 4th-order accuracy in domains with both smoothly-curved interfaces and smoothly-variable model parameters present.

RBF-FD METHODOLOGY

In this section, we will introduce our method for solving a 2-D elastic wave equation. We have reproduced **Figure 1** from [3] to review the different cases our method must handle. In **Figure 1**, the unit square is divided into two regions by a mildly curved interface. Across this interface, the model parameters ρ (density), μ and λ (Lamé parameters) may be discontinuous. **Figure 1** shows a data node structure that conforms to the interface but then transitions to a Cartesian lattice to enable standard FD application a short distance away from it. This type of hybrid node layout can be generated very quickly, as special structure is only imparted to the node set in a small zone near the interface. Also, using standard FD in the majority of the domain allows us to retain much of the economy of memory bandwidth and footprint that are characteristic of Cartesian-grid methods, especially on highly-parallel computing hardware (GPUs).

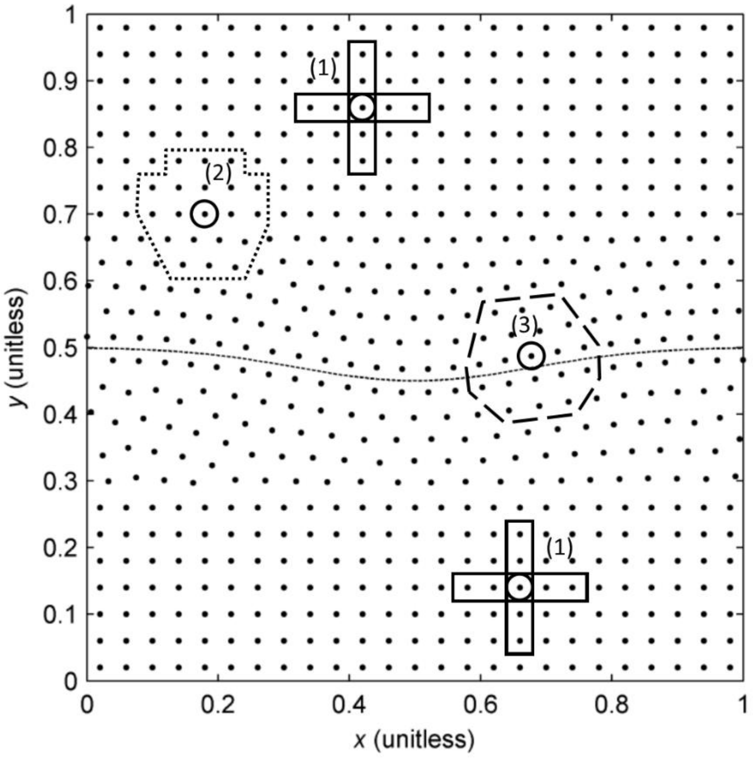

Figure 1. Layout of data nodes near a curved interface. The enclosed regions show *Type 1*, *Type 2*, and *Type 3* stencils. The stencils approximate spatial derivatives at the circled nodes.

The three types of stencils we need to handle are:

- *Type 1*: Stencils are far enough away from the interface that they only contain Cartesian grid nodes
- *Type 2*: Stencils that do not cross the interface, but do include at least some nodes that are not in a regular lattice
- *Type 3*: Stencils that cross the interface

In *Type 1* stencils, we can use traditional FD methods for spatial differentiation and standard integration techniques to update data at each discrete time step. In *Type 2* stencils, we obtain stencil weights through RBF-FD, allowing us to accommodate the transitional structure of stencils that reside between the interface and regions of perfect Cartesian regularity. In *Type 3* stencils, we use continuity of velocity and traction to create a special set of RBF-FD weights that preserve high-order accuracy when used on data interacting with the interface.

***Type 2*: RBF-FD stencils in regions with smoothly variable medium properties, but without completely Cartesian node structure**

When 2-D nodes are not located on a regular grid, the weights for approximating an operator L (such as $L = \partial / \partial x$ or $L = \partial^2 / \partial x^2 + \partial^2 / \partial y^2$) can no longer be obtained by means of 1-D polynomial procedures. Generalizing the 1-D approach for obtaining FD weights to instead use 2-D polynomials gives rise to linear systems that become prone to instabilities and singularities. However, these difficulties can be overcome by supplementing bivariate polynomials with radial basis functions $\phi(\| \underline{x} - \underline{x}_k \|_2)$, with one such centered at each stencil point $\underline{x}_k = (x_k, y_k)$, and then introducing matching constraints. The weights for scattered node RBF-FD stencils can then be obtained by solving a linear system of the form indicated in (1) in the case of using up through linear polynomials in x and y:

$$\left[\begin{array}{ccc|ccc} & & & 1 & x_1 & y_1 \\ & A & & \vdots & \vdots & \vdots \\ & & & 1 & x_n & y_n \\ - & - & - & + & - & - \\ 1 & \cdots & 1 & & & \\ x_1 & \cdots & x_n & & 0 & \\ y_1 & \cdots & y_n & & & \end{array}\right]\left[\begin{array}{c} w_1 \\ \vdots \\ w_n \\ - \\ w_{n+1} \\ w_{n+2} \\ w_{n+3} \end{array}\right] = \left[\begin{array}{c} L\phi(\| \underline{x} - \underline{x}_1 \|)\,|_{\underline{x}=\underline{x}_c} \\ \vdots \\ L\phi(\| \underline{x} - \underline{x}_n \|)\,|_{\underline{x}=\underline{x}_c} \\ - \\ L1\,|_{\underline{x}=\underline{x}_c} \\ Lx\,|_{\underline{x}=\underline{x}_c} \\ Ly\,|_{\underline{x}=\underline{x}_c} \end{array}\right] \tag{1}$$

The entries in the matrix A are $a_{i,j} = \phi(\| \underline{x}_i - \underline{x}_j \|)$. All the terms in the RHS should be evaluated at the stencil's 'center point' $\underline{x}_c$. Non-singularity will generally hold when the radial function $\phi(r)$ is any one of a number of commonly used choices, such as multiquadrics (MQ): $\phi(r) = \sqrt{1+(\varepsilon r)^2}$, Gaussians (GA): $\phi(r) = e^{-(\varepsilon r)^2}$, or polyharmonic splines (PHS) $\phi(r) = \begin{cases} r^m & , m \text{ odd} \\ r^m \log r & , m \text{ even} \end{cases}$. In the solution vector, $w_1, \ldots, w_n$ provides the weights to be used at nodes $\underline{x}_k, k = 1, 2, \ldots, n$, while the remaining w-entries should be ignored. For an overview of the above observations, see [12] and, for a derivation of (1), its Section 5.1.4.

In the present work, we use for stencils of *Type 2* (as shown in **Figure 1**) the approach above, with {... describe here the stencils used, such as RBF type, etc.}. The main theme of the present study is how to improve on the handling of material interfaces, i.e. to create stencils of *Type 3*, by means of altering the form of the supporting polynomials in (1).

***Type 3*: RBF-FD stencils across interfaces**

Although RBF-FD allows us to optimally place nodes around interfaces, we still need to combine this with a strategy that allows accurate differentiation across the interface. In [3], we used our knowledge of interface continuity conditions along with the governing PDE to create a customized set of piecewise polynomial functions that accurately represents data that passes through an interface. We then added these functions to the set of RBF basis functions that allow determination of weights within RBF-FD stencils that cross an interface. It was observed in [13] that, under node refinement, such polynomial functions "take over" from the RBF part of *Type 2* stencils. Hence, in determining weights for *Type 3* stencils, we focus on the polynomial rather than RBF part of their basis when incorporating interface continuity conditions.

Our present approach avoids analysis of matrix null spaces, and instead relies solely on the inversion of small matrices to explicitly determine how the polynomial expansion coefficients of all data (even possibly discontinuous data) change across an interface. We also describe a simple way to account for smoothly variable model parameters present on either side of an interface.

In 2-D, we will be using the first-order formulation of the isotropic elastic wave equation:

$$\begin{cases} \rho u_t = f_x + g_y \\ \rho v_t = g_x + h_y \\ f_t = (\lambda + 2\mu) u_x + \lambda v_y \\ g_t = \mu (u_x + v_y) \\ h_t = (\lambda + 2\mu) v_y + \lambda u_x \end{cases} \tag{2}$$

Above, u, v are local medium velocities in the x and y directions, respectively, f, g, h are the elements of the 2-D stress tensor, ρ represents density, and the Lamé parameters λ, μ describe the material properties with regard to pressure and shear. At the interface, one or more of ρ, λ, μ will be discontinuous or non-smooth. Still, velocity and traction (components of stress perpendicular to the interface) must remain continuous. Away from interfaces, density and the Lamé parameters may be smoothly variable functions of x and y.

Before showing how these equations and continuity conditions lead to our present, 2-D RBF-FD approach, we will as an introduction describe an analogous approach to a 1-D simplification of the problem.

Determining Type 3 polynomial basis functions: 1-D simplification

There are three main reasons why we will present the method as applied to a 1-D case before moving on to 2-D:

- we can more easily visualize the explicit way in which Taylor coefficients of data change across an interface;
- the structure of matrix operators and representations used in specific examples are much smaller and easy to display; and
- the 1-D case requires no use of RBF-FD methodology. For a 1-D problem, we can find standard finite difference weights that maintain high-order accuracy, even in stencils that cross an interface.

In the 1-D case, (2) reduces to a first-order form of the 2-way wave equation:

$$\begin{bmatrix} u_t \\ f_t \end{bmatrix} = \begin{bmatrix} 0 & \frac{1}{\rho}\frac{\partial}{\partial x} \\ \rho c_p^2 \frac{\partial}{\partial x} & 0 \end{bmatrix} \begin{bmatrix} u \\ f \end{bmatrix} \equiv D \begin{bmatrix} u \\ f \end{bmatrix} \tag{3}$$

In (3), c_p is the local speed of waves within the medium. Higher-order time derivatives and their equivalences in terms of spatial derivatives can be obtained by applying the differential operator *D* multiple times:

$$\begin{bmatrix} u_{(k)t} \\ f_{(k)t} \end{bmatrix} = \begin{bmatrix} 0 & \frac{1}{\rho}\frac{\partial}{\partial x} \\ \rho c_p^2 \frac{\partial}{\partial x} & 0 \end{bmatrix}^k \begin{bmatrix} u \\ f \end{bmatrix} \equiv D^k \begin{bmatrix} u \\ f \end{bmatrix} \tag{4}$$

In (4), the subscript *k* on the left hand side of the equation denotes the *k*th time derivative of that data field. If data is continuous at a point on an interface, all of its time derivatives must also be continuous at that point (otherwise, a discontinuity would appear in the data at that point as time progresses).

In 1-D, continuity of velocity and traction mandate that both *u* and *f* must be continuous across an interface. Therefore, $u_{(k)t}, f_{(k)t}$ are also continuous there for all values of *k*. If we refer to the left and right sides of the interface as sides *L* and *R*, respectively, then for $k = 0,1,2,\ldots$, the following must hold as we approach the interface location from the two sides:

$$D_L^k \begin{bmatrix} u_L \\ f_L \end{bmatrix} = D_R^k \begin{bmatrix} u_R \\ f_R \end{bmatrix} \tag{5}$$

We now describe how we form 1-D stencils that uphold these conditions.

Suppose an interface is located at $x = 0$. For our examples, we will form 4th-order stencils for differentiation across the interface, and thus we consider up to 4th-degree Taylor expansion terms of both data fields about the interface location:

$$\mathbf{u} \approx \begin{cases} \mathbf{u}_L = u_{L,1}1 + u_{L,x}x + u_{L,x^2}x^2 + u_{L,x^3}x^3 + u_{L,x^4}x^4 & \text{if } x \le 0 \\ \mathbf{u}_R = u_{R,1}1 + u_{R,x}x + u_{R,x^2}x^2 + u_{R,x^3}x^3 + u_{R,x^4}x^4 & \text{if } x \ge 0 \end{cases} \tag{6}$$

$$\mathbf{f} \approx \begin{cases} \mathbf{f}_L = f_{L,1}1 + f_{L,x}x + f_{L,x^2}x^2 + f_{L,x^3}x^3 + f_{L,x^4}x^4 & \text{if } x \le 0 \\ \mathbf{f}_R = f_{R,1}1 + f_{R,x}x + f_{R,x^2}x^2 + f_{R,x^3}x^3 + f_{R,x^4}x^4 & \text{if } x \ge 0 \end{cases} \tag{7}$$

In (6) and (7), the boldface notation indicates that we are now interested in vectors of polynomial coefficients for each data field in $\mathbb{P}_4(\mathbb{R})$.

$$\mathbf{u}_L = \begin{bmatrix} u_{L,1} \\ u_{L,x} \\ u_{L,x^2} \\ u_{L,x^3} \\ u_{L,x^4} \end{bmatrix}, \ \mathbf{u}_R = \begin{bmatrix} u_{R,1} \\ u_{R,x} \\ u_{R,x^2} \\ u_{R,x^3} \\ u_{R,x^4} \end{bmatrix}, \ \mathbf{f}_L = \begin{bmatrix} f_{L,1} \\ f_{L,x} \\ f_{L,x^2} \\ f_{L,x^3} \\ f_{L,x^4} \end{bmatrix}, \ \mathbf{f}_R = \begin{bmatrix} f_{R,1} \\ f_{R,x} \\ f_{R,x^2} \\ f_{R,x^3} \\ f_{R,x^4} \end{bmatrix} \tag{8}$$

We now define the discretized PDE differential operators D_L and D_R by their action on these monomial basis functions:

$$\begin{bmatrix} \mathbf{u}_{L,(k)t} \\ \mathbf{f}_{L,(k)t} \end{bmatrix} = \begin{bmatrix} 0 & \frac{1}{\rho_L}\frac{\partial}{\partial x} \\ \rho_L c_{p,L}^2 \frac{\partial}{\partial x} & 0 \end{bmatrix}^k \begin{bmatrix} \mathbf{u}_L \\ \mathbf{f}_L \end{bmatrix} \equiv D_L{}^k \begin{bmatrix} \mathbf{u}_L \\ \mathbf{f}_L \end{bmatrix} \tag{9}$$

$$\begin{bmatrix} \mathbf{u}_{R,(k)t} \\ \mathbf{f}_{R,(k)t} \end{bmatrix} = \begin{bmatrix} 0 & \frac{1}{\rho_R}\frac{\partial}{\partial x} \\ \rho_R c_{p,R}^2 \frac{\partial}{\partial x} & 0 \end{bmatrix}^k \begin{bmatrix} \mathbf{u}_R \\ \mathbf{f}_R \end{bmatrix} \equiv D_R{}^k \begin{bmatrix} \mathbf{u}_R \\ \mathbf{f}_R \end{bmatrix} \tag{10}$$

In this context, we define the discrete differential block operator $\partial / \partial x$ by its action on the standard monomial basis set in $\mathbb{P}_{\mathbf{4}}(\mathbb{R})$:

$$\frac{\partial}{\partial x}\begin{bmatrix} 1 \\ x \\ x^2 \\ x^3 \\ x^4 \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 2 & 0 & 0 \\ 0 & 0 & 0 & 3 & 0 \\ 0 & 0 & 0 & 0 & 4 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}\begin{bmatrix} 1 \\ x \\ x^2 \\ x^3 \\ x^4 \end{bmatrix} \tag{11}$$

Suppose that the wave speed c_p is smoothly variable on each side of the interface at $x = 0$, with Taylor expansion (through 4th degree):

$$\mathbf{c}_p \approx \begin{cases} \mathbf{c}_L = c_{L,1}1 + c_{L,x}x + c_{L,x^2}x^2 + c_{L,x^3}x^3 + c_{L,x^4}x^4 & if\ x < 0 \\ \mathbf{c}_R = c_{R,1}1 + c_{R,x}x + c_{R,x^2}x^2 + c_{R,x^3}x^3 + c_{R,x^4}x^4 & if\ x \geq 0 \end{cases} \tag{12}$$

We can use this information to replace the wave speed parameter in (9) and (10) with a matrix representation of how multiplication by the (variable) wave speed affects data expansions (up through 4th-degree terms) on both sides of the interface:

$$c_{p,L}\begin{bmatrix} 1 \\ x \\ x^2 \\ x^3 \\ x^4 \end{bmatrix} = \begin{bmatrix} c_{L,1} & 0 & 0 & 0 & 0 \\ c_{L,x} & c_{L,1} & 0 & 0 & 0 \\ c_{L,x^2} & c_{L,x} & c_{L,1} & 0 & 0 \\ c_{L,x^3} & c_{L,x^2} & c_{L,x} & c_{L,1} & 0 \\ c_{L,x^4} & c_{L,x^3} & c_{L,x^2} & c_{L,x} & c_{L,1} \end{bmatrix}\begin{bmatrix} 1 \\ x \\ x^2 \\ x^3 \\ x^4 \end{bmatrix} \tag{13}$$

$$c_{p,R}\begin{bmatrix} 1 \\ x \\ x^2 \\ x^3 \\ x^4 \end{bmatrix} = \begin{bmatrix} c_{R,1} & 0 & 0 & 0 & 0 \\ c_{R,x} & c_{R,1} & 0 & 0 & 0 \\ c_{R,x^2} & c_{R,x} & c_{R,1} & 0 & 0 \\ c_{R,x^3} & c_{R,x^2} & c_{R,x} & c_{R,1} & 0 \\ c_{R,x^4} & c_{R,x^3} & c_{R,x^2} & c_{R,x} & c_{R,1} \end{bmatrix}\begin{bmatrix} 1 \\ x \\ x^2 \\ x^3 \\ x^4 \end{bmatrix} \tag{14}$$

In the case of constant wave speeds on one side (or both sides) of the interface, the representations in (13) and/or (14) reduce to diagonal (scalar) matrices that are equivalent to multiplication by the (constant) wave speed on that side of the interface.

We next need to use our discrete formulations in (9) and (10) to make sure that (5) holds, up through the desired degree of accuracy (4th degree, in the example we're currently constructing). We'll illustrate how we do this with a particular example.

Determining Type 3 polynomial basis functions for a specific 1-D problem

Suppose we're interested in solving a wave equation problem on the closed 1-D interval $[-1,1]$, with the wave speed profile shown in **Figure 2**.

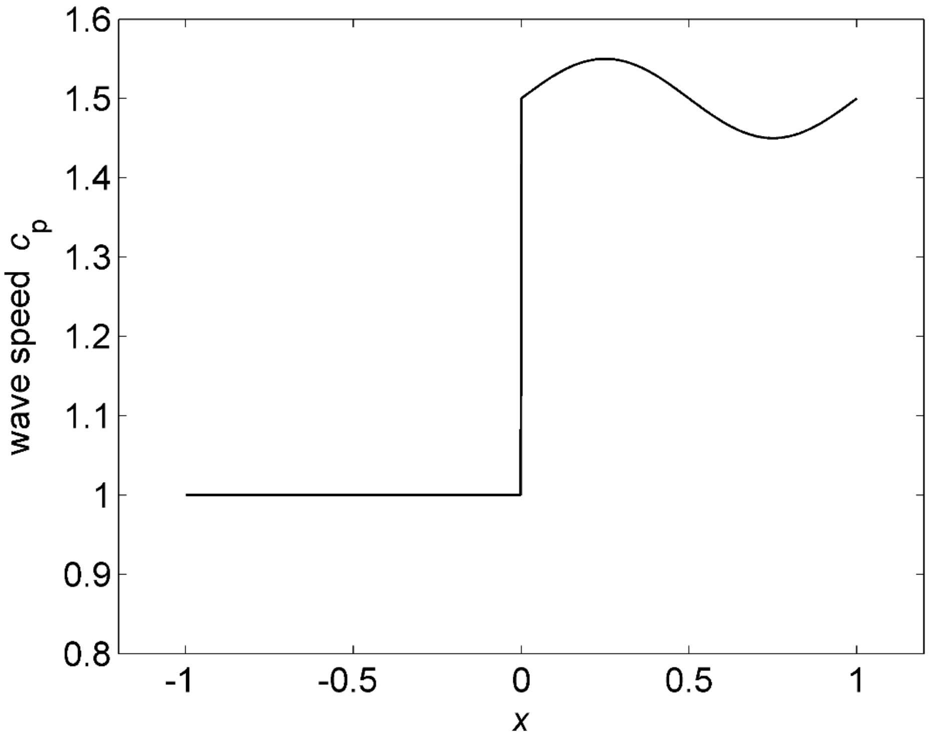

Figure 2. Wave speed profile for our specific 1-D, non-dimensionalized example.

In our example, wave speed and density throughout the domain are defined as follows:

$$c_p = \begin{cases} 1, & x \in [-1,0) \\ 1.5 + 0.05\sin(2\pi x), & x \in [0,1.0] \end{cases} \tag{15}$$

$$\rho = \begin{cases} 1, & x \in [-1,0) \\ 1.5, & x \in [0,1.0] \end{cases} \tag{16}$$

We will walk through the steps of enforcing (5) and determining weights for stencils that cross the interface at $x = 0$. We have the following matrix representations for multiplication of polynomials by wave speed, using the Taylor expansion of (15) about $x = 0$:

$$c_{p,L}\begin{bmatrix} 1 \\ x \\ x^2 \\ x^3 \\ x^4 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}\begin{bmatrix} 1 \\ x \\ x^2 \\ x^3 \\ x^4 \end{bmatrix} \tag{17}$$

$$c_{p,R}\begin{bmatrix} 1 \\ x \\ x^2 \\ x^3 \\ x^4 \end{bmatrix} = \begin{bmatrix} 1.5 & 0 & 0 & 0 & 0 \\ 0.314 & 1.5 & 0 & 0 & 0 \\ 0 & 0.314 & 1.5 & 0 & 0 \\ -2.07 & 0 & 0.314 & 1.5 & 0 \\ 0 & -2.07 & 0 & 0.314 & 1.5 \end{bmatrix}\begin{bmatrix} 1 \\ x \\ x^2 \\ x^3 \\ x^4 \end{bmatrix} \tag{18}$$

First, we'll consider characterization of data field u at the interface. If we set $k = 0$ and evaluate (5) with our discrete forms of the left- and right side differential operators, we find that the following must be true, from the upper 5 rows and leftmost 5 columns of either side of the block-diagonal matrix equation:

$$\mathbf{u}_{L,(0)t}\Big|_{x=0} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} (1)u_{L,1} \\ (x)u_{L,x} \\ (x^2)u_{L,x^2} \\ (x^3)u_{L,x^3} \\ (x^4)u_{L,x^4} \end{bmatrix}\Bigg|_{x=0} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} (1)u_{R,1} \\ (x)u_{R,x} \\ (x^2)u_{R,x^2} \\ (x^3)u_{R,x^3} \\ (x^4)u_{R,x^4} \end{bmatrix}\Bigg|_{x=0} = \mathbf{u}_{R,(0)t}\Big|_{x=0}$$

(19)

With the interface conveniently located at $x = 0$, (19) simply indicates that the constant expansion terms of u across the interface are equal:

$$\begin{bmatrix} 1 & 0 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} u_{L,1} \\ u_{L,x} \\ u_{L,x^2} \\ u_{L,x^3} \\ u_{L,x^4} \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} u_{R,1} \\ u_{R,x} \\ u_{R,x^2} \\ u_{R,x^3} \\ u_{R,x^4} \end{bmatrix} \tag{20}$$

We next set $k = 1$ and reevaluate (5). This time, we will enforce continuity of the 1st time derivative of f across the interface, which equates to spatial derivatives of u through the PDE. From the lower 5 rows and leftmost 5 columns of the result, we have the following:

$$\mathbf{f}_{L,(1)t}\Big|_{x=0} = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 2 & 0 & 0 \\ 0 & 0 & 0 & 3 & 0 \\ 0 & 0 & 0 & 0 & 4 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} (1)u_{L,1} \\ (x)u_{L,x} \\ (x^2)u_{L,x^2} \\ (x^3)u_{L,x^3} \\ (x^4)u_{L,x^4} \end{bmatrix}\Bigg|_{x=0} = \begin{bmatrix} 0 & 3.4 & 0 & 0 & 0 \\ 0 & 1.4 & 6.8 & 0 & 0 \\ 0 & 0.4 & 2.8 & 10 & 0 \\ 0 & -9.3 & 0.3 & 4.2 & 14 \\ 0 & -2.0 & -19 & 0.4 & 5.7 \end{bmatrix} \begin{bmatrix} (1)u_{R,1} \\ (x)u_{R,x} \\ (x^2)u_{R,x^2} \\ (x^3)u_{R,x^3} \\ (x^4)u_{R,x^4} \end{bmatrix}\Bigg|_{x=0} = \mathbf{f}_{R,(1)t}\Big|_{x=0}$$

(21)

Since f must always be continuous as we approach $x = 0$, the time derivative of its constant coefficient must be equal across the interface. Therefore, we need to enforce the first row of linear relationships between Taylor coefficients of u. We can add this first row of (21) to the condition we already have established in (20):

$$\begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} u_{L,1} \\ u_{L,x} \\ u_{L,x^2} \\ u_{L,x^3} \\ u_{L,x^4} \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 3.4 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} u_{R,1} \\ u_{R,x} \\ u_{R,x^2} \\ u_{R,x^3} \\ u_{R,x^4} \end{bmatrix} \tag{22}$$

To find more relationships between the Taylor coefficients of u on either side of the interface, we continue evaluating (5) for higher and higher values of k, each time collecting rows of coefficients that correspond to time derivatives of the constant expansion terms of u or f: these must be equal to uphold continuity at $x = 0$. After proceeding through 4th-degree time derivatives, we have accumulated the following relationships between the expansion coefficients of u:

$$\begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 2 & 0 & 0 \\ 0 & 0 & 0 & 6 & 0 \\ 0 & 0 & 0 & 0 & 24 \end{bmatrix} \begin{bmatrix} u_{L,1} \\ u_{L,x} \\ u_{L,x^2} \\ u_{L,x^3} \\ u_{L,x^4} \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 3.4 & 0 & 0 & 0 \\ 0 & 0.9 & 4.5 & 0 & 0 \\ 0 & 0.7 & 13 & 46 & 0 \\ 0 & -84 & 6.2 & 51 & 122 \end{bmatrix} \begin{bmatrix} u_{R,1} \\ u_{R,x} \\ u_{R,x^2} \\ u_{R,x^3} \\ u_{R,x^4} \end{bmatrix} \tag{23}$$

For the duration of this paper, we will refer to these square, well-conditioned matrices that relate data expansion coefficients on one side of an interface to those on the other as “continuity matrices”, indicated as an upper-case C. In this notation, we can express (23) more succinctly:

$$C_L \mathbf{u}_L = C_R \mathbf{u}_R \tag{24}$$

We can use this relationship to explicitly determine Taylor coefficients of u on the left side of the interface if we know those coefficients on the right side, or vice versa. For example:

$$\mathbf{u}_L = C_L^{-1} C_R \mathbf{u}_R \tag{25}$$

If we assume the standard basis set of polynomial coefficients for $\mathbf{u}_R$, we can express an entire polynomial expansion for u, on both sides of the interface, as a concatenated matrix U of expansion terms on the left side of the interface (U_L; top 5 rows) and right side of the interface (U_R; bottom 5 rows), as shown in (26).

$$U = \begin{bmatrix} U_L \\ U_R \end{bmatrix} = \begin{bmatrix} C_L^{-1} C_R \\ I_5 \end{bmatrix} = \left[\begin{array}{ccccc} 1 & 0 & 0 & 0 & 0 \\ 0 & 3.4 & 0 & 0 & 0 \\ 0 & 0.47 & 2.3 & 0 & 0 \\ 0 & 0.11 & 2.1 & 7.6 & 0 \\ 0 & -3.5 & 0.26 & 2.1 & 5.1 \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{array}\right] \begin{array}{l} \leftarrow u_{L,1} \\ \leftarrow u_{L,x} \\ \leftarrow u_{L,x^2} \\ \leftarrow u_{L,x^3} \\ \leftarrow u_{L,x^4} \\ \leftarrow u_{R,1} \\ \leftarrow u_{R,x} \\ \leftarrow u_{R,x^2} \\ \leftarrow u_{R,x^3} \\ \leftarrow u_{R,x^4} \end{array} \tag{26}$$

Each column of the matrix represents a single expansion function defined on both sides of the interface. As stated above, we have assumed the standard basis set in $\mathbb{P}_4(\mathbb{R})$ for $\mathbf{u}_R$, making U_R equal to the 5-by-5 identity matrix U_5. This designation is arbitrary – we could have similarly assumed the standard basis set for $\mathbf{u}_L$, and obtained a companion set of basis functions for $\mathbf{u}_R$ through similar use of the continuity matrices. We need only specify *a* basis for $\mathbf{u}$ on either side of the interface to explicitly determine its structure on the other side. As constructed in (26), all 5 expansion functions for u are plotted in **Figure 3**.

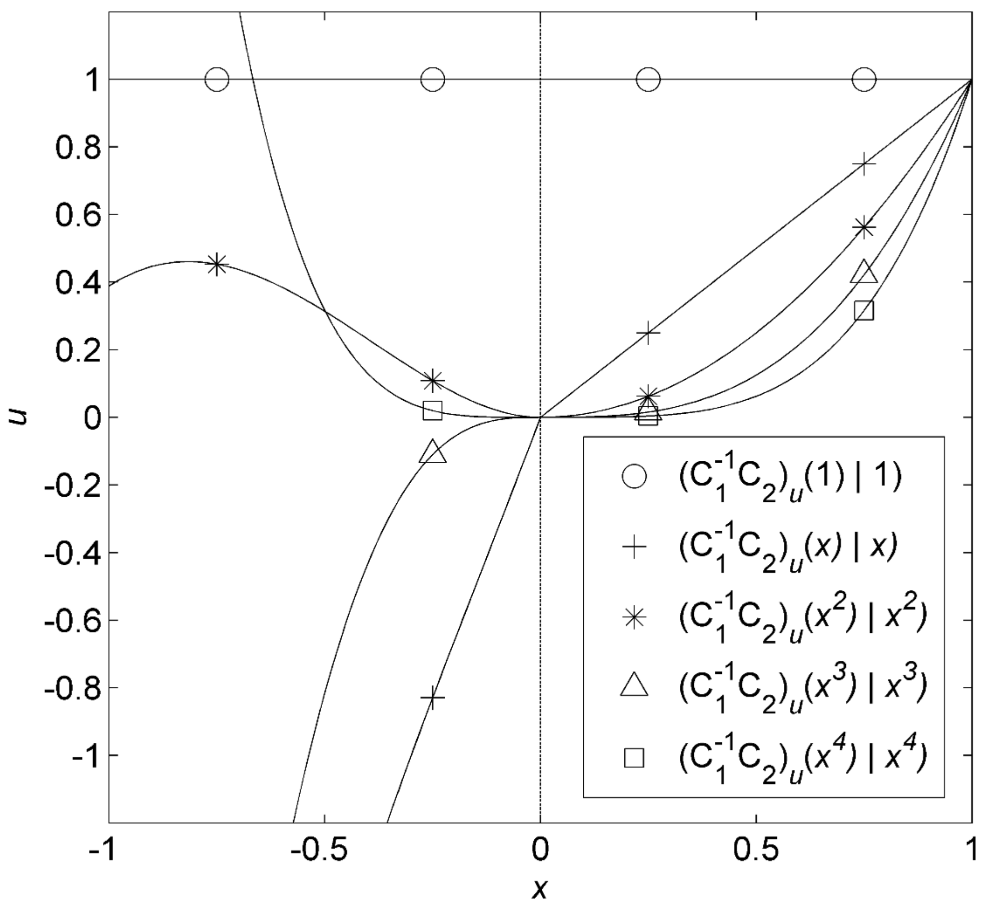

Figure 3. Standard basis functions for u in $\mathbb{P}_4(\mathbb{R})$ (on the right side of the interface at $x = 0$) and how they must change (on the left side) to uphold interface continuity conditions.

This completes our characterization of expansion terms for data field u near the interface. We must also go through an equivalent process for data field f, as its expansion terms may change in different ways as they cross the interface. To do so, we can follow the same procedure as we did for u. We can work our way up through powers k of the differential operator D in (5), collecting linear relationships between the expansion coefficients of f at each step by enforcing that constant expansion terms of both u and f must be equal in all their time derivatives. However, with expansion terms of u already determined near the interface, there is another very simple method to determine expansion terms for f. An analog of this method will be very important when we return to 2-D.

From (9) and (10), recall that how the expansion coefficients of f change in time is directly related to expansion coefficients of u currently present in the data on either side of the interface:

$$\mathbf{f}_{L,t} = \rho_L c_{p,L}^2 \frac{\partial}{\partial x}\mathbf{u}_L = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 2 & 0 & 0 \\ 0 & 0 & 0 & 3 & 0 \\ 0 & 0 & 0 & 0 & 4 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix} \mathbf{u}_L \tag{27}$$

$$\mathbf{f}_{R,t} = \rho_R c_{p,R}^2 \frac{\partial}{\partial x}\mathbf{u}_R = \begin{bmatrix} 0 & 3.4 & 0 & 0 & 0 \\ 0 & 1.4 & 6.8 & 0 & 0 \\ 0 & 0.4 & 2.8 & 10 & 0 \\ 0 & -9.3 & 0.3 & 4.2 & 14 \\ 0 & -2.0 & -19 & 0.4 & 5.7 \end{bmatrix} \mathbf{u}_R \tag{28}$$

If we have stored our expansion functions for u as square matrices U_L and U_R as in (26), we can determine all possibilities of how coefficients of f can change in time by examining the column space of the following (rectangular) matrix:

$$F_t = \begin{bmatrix} F_{L,t} \\ F_{R,t} \end{bmatrix} = \begin{bmatrix} \begin{bmatrix} 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 2 & 0 & 0 \\ 0 & 0 & 0 & 3 & 0 \\ 0 & 0 & 0 & 0 & 4 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix} U_L \\ \begin{bmatrix} 0 & 3.4 & 0 & 0 & 0 \\ 0 & 1.4 & 6.8 & 0 & 0 \\ 0 & 0.4 & 2.8 & 10 & 0 \\ 0 & -9.3 & 0.3 & 4.2 & 14 \\ 0 & -2.0 & -19 & 0.4 & 5.7 \end{bmatrix} U_R \end{bmatrix} = \begin{bmatrix} 0 & 3.4 & 0 & 0 & 0 \\ 0 & 0.94 & 4.5 & 0 & 0 \\ 0 & 0.33 & 6.4 & 23 & 0 \\ 0 & -14 & 1.0 & 8.5 & 20 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 3.4 & 0 & 0 & 0 \\ 0 & 1.4 & 6.8 & 0 & 0 \\ 0 & 0.14 & 2.8 & 10 & 0 \\ 0 & -9.3 & 0.30 & 4.2 & 14 \\ 0 & -1.9 & -19 & 0.44 & 5.7 \end{bmatrix}$$

(29)

Here, possibilities for expansion terms of f_t are also expressed as a concatenated matrix F_t of column vectors holding expansion coefficients of f_t on the left side of the interface (top 5 rows) and on the right side of the interface (bottom 5 rows).

Since the linearly independent column vectors of F_t form a basis for the space of all possible (coupled) *time derivatives of* expansion coefficients for f on both sides of the interface, these column vectors also form a basis for all possible *data* expansion coefficients for f on both sides of the interface. Starting from data (at least locally) at value zero, any allowable combination of nonzero *data* expansion coefficients must arise over time from an allowable combination of *time derivatives* of those very same coefficients.

The only drawback in characterizing expansion coefficients of data field f in this manner is that one degree of accuracy is lost. For example, figuring out a basis for f through the PDE's action on a 4th-order basis for u will result in a 3rd-order accurate basis for f (some of the 4th-degree terms may not satisfy all continuity conditions). For instance, our example exercise has given us a 3rd-order accurate basis for the expansion of f near the interface, which we can find by truncating the expansion after 3rd-degree terms and omitting the column vector of zeros from (29) .

$$F = \begin{array}{l} \left[\begin{array}{cccc} 3.4 & 0 & 0 & 0 \\ 0.94 & 4.5 & 0 & 0 \\ 0.33 & 6.4 & 23 & 0 \\ -14 & 1.0 & 8.5 & 20 \\ 3.4 & 0 & 0 & 0 \\ 1.4 & 6.8 & 0 & 0 \\ 0.14 & 2.8 & 10 & 0 \\ -9.3 & 0.30 & 4.2 & 14 \end{array}\right] \end{array} \begin{array}{l} \leftarrow f_{L,1} \\ \leftarrow f_{L,x} \\ \leftarrow f_{L,x^2} \\ \leftarrow f_{L,x^3} \\ \leftarrow f_{R,1} \\ \leftarrow f_{R,x} \\ \leftarrow f_{R,x^2} \\ \leftarrow f_{R,x^3} \end{array} \tag{30}$$

If we need a 4th-order basis for the expansion coefficients of f and we want to gain that basis through this method, all we need to do is compute a 5th-order basis for u first, and carry out the process of (27) through (29) on this 5th-order basis. We then obtain a 4th-order basis for f by culling the resultant 5th-order terms and the column vector of zeros that results from differentiating constant expansion terms of u.

Finally, we need to find finite difference weights that accurately approximate u_x and f_x to our desired order of accuracy. In our 4th-order example, that means we must enforce that u_x weights for (centered) stencils that cross the interface must accurately reproduce derivatives of all 5 functions plotted in **Figure 3**, and f_x weights do the same for an appropriate set of 5 4th-order basis functions in f near the interface (obtained through the methods described above). **Figure 4** shows a plot of several stencil weights for u_x in our example problem, where the spacing h between data nodes is 0.01, and those nodes are offset from the interface by a distance $h/2$.

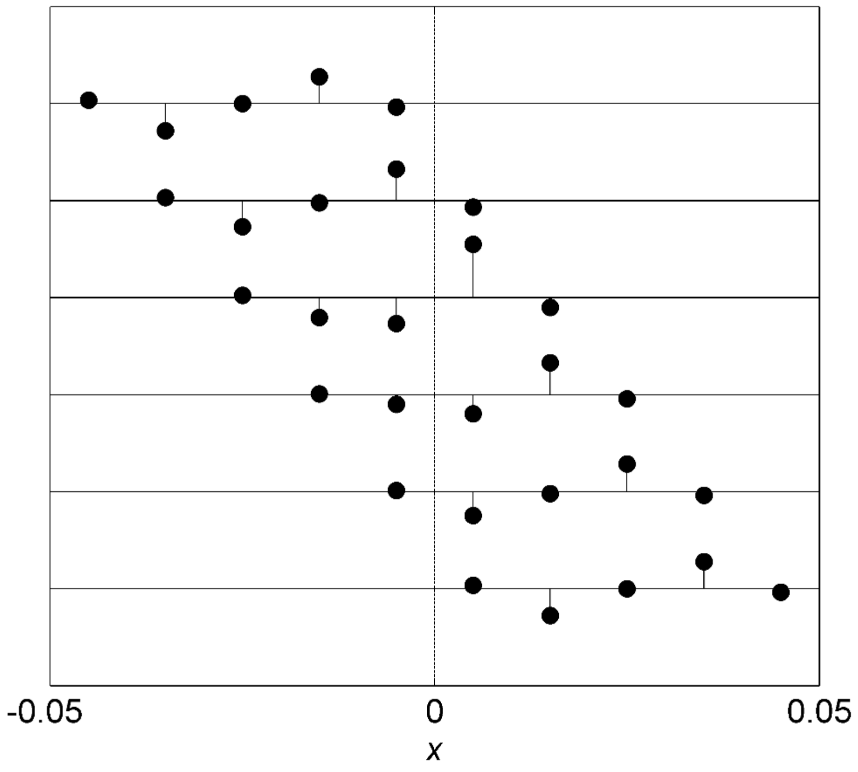

Figure 4. Plot of weights for u_x stencils near the interface in our test problem, with h = 0.01. Relative vertical scale is the same for all stencils, and height of dots above or below each stencil indicates magnitude of a stencil weight above or below zero, respectively. The top and bottom stencil are standard 4th-order finite difference stencils (with weights of $1/12h$, $-2/3h$, 0, $2/3h$, and $-1/12h$, respectively), since they do not overlap the interface. The perturbation of the weights of stencils that cross the interface is a result of their containing all important information about the interface: not only the exact jump in parameter values right at the interface, but also how the parameters may smoothly vary on either side.

Determining Type 3 polynomial basis functions: 2-D example

In 1-D, we were able to find a complete basis for piecewise Taylor expansion functions that upheld continuity of traction and motion through an arbitrary order of accuracy, and we were able to find traditional finite difference weights that accurately differentiated these non-smooth functions. In 2-D, attempting to create traditional FD stencils from an explicit list of individual Taylor basis functions gives rise to very ill-conditioned and singular systems of equations to solve. On the other hand, the RBF-FD methodology allows inclusion of such functions (along with RBFs) in a basis used to compute stencil weights, and we can extend our method to 2-D (and higher-D, if we wish).

For illustration, assume that we wish to add polynomial support up to and including second degree terms to an RBF-FD stencil that crosses an interface. We pick a point on the interface that lies near the data node where the stencil approximates a linear operator. As in the 1-D example, we designate the nearby point on the interface as the origin for purposes of evaluating support polynomials, and we initially assume that all expansion coefficients of all data fields may vary from one side of the interface to another.

We have the following data expansions for which we want to determine coefficients:

$$\begin{cases} \mathbf{u} \approx \begin{cases} \mathbf{u}_L = u_{L,1}1 + u_{L,x}x + u_{L,y}y + u_{L,x^2}x^2 + u_{L,xy}xy + u_{L,y^2}y^2 & \textit{if } y < c(x) \\ \mathbf{u}_R = u_{R,1}1 + u_{R,x}x + u_{R,y}y + u_{R,x^2}x^2 + u_{R,xy}xy + u_{R,y^2}y^2 & \textit{otherwise} \end{cases} \\ \mathbf{v} \approx \begin{cases} \mathbf{v}_L = v_{L,1}1 + v_{L,x}x + v_{L,y}y + v_{L,x^2}x^2 + v_{L,xy}xy + v_{L,y^2}y^2 & \textit{if } y < c(x) \\ \mathbf{v}_R = v_{R,1}1 + v_{R,x}x + v_{R,y}y + v_{R,x^2}x^2 + v_{R,xy}xy + v_{R,y^2}y^2 & \textit{otherwise} \end{cases} \\ \qquad\qquad\vdots \\ \mathbf{h} \approx \begin{cases} \mathbf{h}_L = h_{L,1}1 + h_{L,x}x + h_{L,y}y + h_{L,x^2}x^2 + h_{L,xy}xy + h_{L,y^2}y^2 & \textit{if } y < c(x) \\ \mathbf{h}_R = h_{R,1}1 + h_{R,x}x + h_{R,y}y + h_{R,x^2}x^2 + h_{R,xy}xy + h_{R,y^2}y^2 & \textit{otherwise} \end{cases} \end{cases} \tag{31}$$

Although left (*L*) and right (*R*) don't have absolute meaning in 2-D, we've kept the notation from our 1-D illustration for consistency: *L* and *R* subscripts indicate data values or expansion coefficients as we approach an interface from one side or the other. The differential operator *D* is a bit more complicated in 2-D:

$$\begin{bmatrix} u_{(k)t} \\ v_{(k)t} \\ f_{(k)t} \\ g_{(k)t} \\ h_{(k)t} \end{bmatrix} = \begin{bmatrix} 0 & 0 & \left(\frac{1}{\rho}\right)\frac{\partial}{\partial x} & \left(\frac{1}{\rho}\right)\frac{\partial}{\partial y} & 0 \\ 0 & 0 & 0 & \left(\frac{1}{\rho}\right)\frac{\partial}{\partial x} & \left(\frac{1}{\rho}\right)\frac{\partial}{\partial y} \\ (\lambda+2\mu)\frac{\partial}{\partial x} & \lambda\frac{\partial}{\partial y} & 0 & 0 & 0 \\ \mu\frac{\partial}{\partial y} & \mu\frac{\partial}{\partial x} & 0 & 0 & 0 \\ \lambda\frac{\partial}{\partial x} & (\lambda+2\mu)\frac{\partial}{\partial y} & 0 & 0 & 0 \end{bmatrix}^k \begin{bmatrix} u \\ v \\ f \\ g \\ h \end{bmatrix} \equiv D^k \begin{bmatrix} u \\ v \\ f \\ g \\ h \end{bmatrix} \quad (32)$$

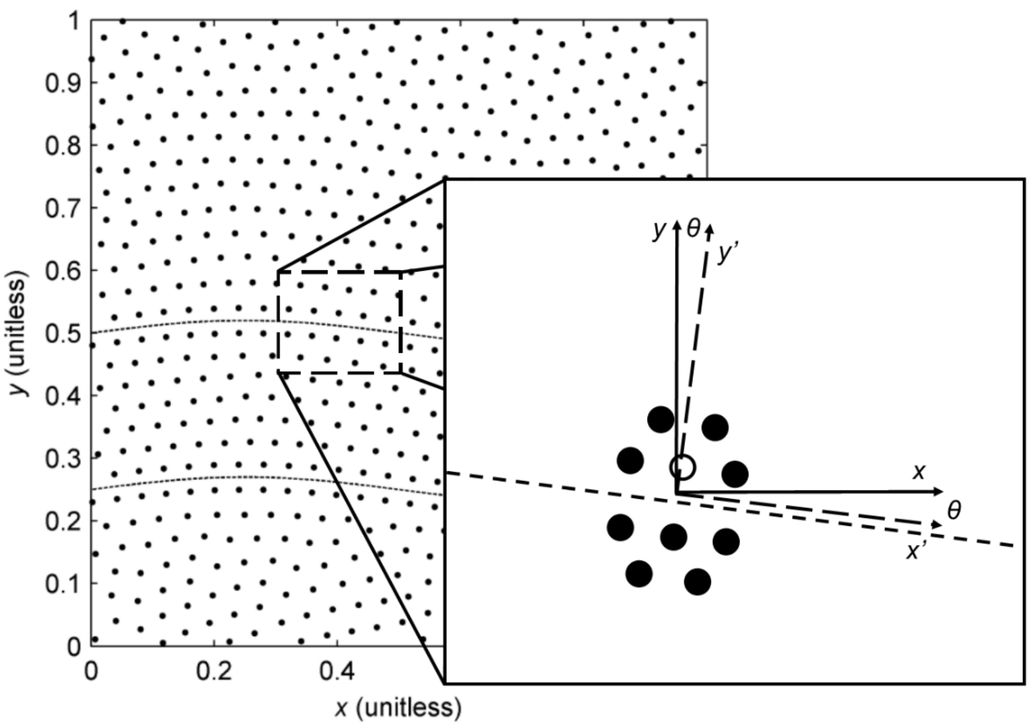

Figure 5. RBF nodes surround curved interfaces, shown by dashed lines. (Zoomed-in view) A very small stencil is shown here for evaluating the spatial derivatives at the RBF node indicated by the empty circle. Weights are applied to data at all RBF node (circle) locations. The stencil is created using RBF-FD for a horizontal interface in the x', y' coordinate system.

Consider an RBF-FD stencil near a curved interface, as shown in **Figure 5**. At this location, the following must be true for continuity of traction and motion to hold:

$$\begin{cases} u'_L\big|_{(y'=0)} = u'_R\big|_{(y'=0)} \\ v'_L\big|_{(y'=0)} = v'_R\big|_{(y'=0)} \\ g'_L\big|_{(y'=0)} = g'_R\big|_{(y'=0)} \\ h'_L\big|_{(y'=0)} = h'_R\big|_{(y'=0)} \end{cases} \qquad (33)$$

In (33), the primed data values are the same physical quantities we have been discussing already (horizontal particle velocity, vertical particle velocity, etc.), but expressed in the rotated coordinate system shown in **Figure 5**. Note that there is no continuity relationship for f' (stress parallel to the interface).

In **Figure 5**, also note how two rows of nodes orthogonally straddle the interface. We began using this strategy by observing the results of [4], thinking we might see an accuracy advantage when arranging nodes this way. This indeed appears to be the case. More importantly, though, this type of node layout (as compared to nodes scattered without regard to the interface, or placed upon the interface itself) has been key in maintaining stability of the current method in many trial problems solved thus far, especially when the contrast between model parameters across an interface is significant.

The rest of the nodes in the domain are placed via a simulated static repulsion routine, resulting in a hex-like, quasi-uniform arrangement as seen in **Figure 5**. Based on the analysis in [14] and tests in [15], we argue that such a structure is at least as inherently resistant to dispersion error as a Cartesian grid of nodes with the same spatial resolution.

We can express each of the 5 data fields in the rotated coordinate system:

$$\begin{bmatrix} \cos(\theta) & \sin(\theta) \\ -\sin(\theta) & \cos(\theta) \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix} \equiv R_{uv} \begin{bmatrix} u \\ v \end{bmatrix} = \begin{bmatrix} u' \\ v' \end{bmatrix} \tag{34}$$

$$\begin{bmatrix} \cos^2(\theta) & 2\sin(\theta)\cos(\theta) & \sin^2(\theta) \\ -\sin(\theta)\cos(\theta) & \cos^2(\theta)-\sin^2(\theta) & \sin(\theta)\cos(\theta) \\ \sin^2(\theta) & -2\sin(\theta)\cos(\theta) & \cos^2(\theta) \end{bmatrix} \begin{bmatrix} f \\ g \\ h \end{bmatrix} \equiv R_{fgh} \begin{bmatrix} f \\ g \\ h \end{bmatrix} = \begin{bmatrix} f' \\ g' \\ h' \end{bmatrix} \tag{35}$$

We can also express time derivatives in the rotated coordinate system:

$$\begin{bmatrix} u'_{(k)t} \\ v'_{(k)t} \\ f'_{(k)t} \\ g'_{(k)t} \\ h'_{(k)t} \end{bmatrix} = \begin{bmatrix} R_{uv} & 0 \\ 0 & R_{fgh} \end{bmatrix} \left(D^k \begin{bmatrix} u \\ v \\ f \\ g \\ h \end{bmatrix} \right). \tag{36}$$

As in 1-D, we can uphold continuity of motion and traction by enforcing a form of (37) that acts on 2-D expansion terms of data such as those found in (31).

$$D_L^k \left. \begin{bmatrix} u'_L \\ v'_L \\ g'_L \\ h'_L \end{bmatrix} \right|_{y'=0} = D_R^k \left. \begin{bmatrix} u'_R \\ v'_R \\ g'_R \\ h'_R \end{bmatrix} \right|_{y'=0} \tag{37}$$

In 1-D, we determined each support monomial for an FD stencil that crosses an interface by examining every row of (37) associated with time derivatives of constant expansion terms after k applications of the differential operator D. For (both) continuous data fields, all time derivatives of their constant expansion terms had to be equal across the interface for continuity of traction and motion to hold as time passes.

In 2-D, we similarly uphold continuity of motion and traction by gathering linear relationships between their expansion coefficients. We obtain these by observing time derivatives of the first (p-k) monomial terms in the coordinate x' (that is, 1, $x', (x')^2$, etc.) of the continuous fields u', v', g', and h' after k applications of the discrete form of (37). Here, p is the maximum degree of polynomial support we wish to add across the interface. We can use the rotational transformation in (36) to observe how these continuity conditions translate to linear relationships between u and v expansion terms in the original (traditional x and y) coordinate system; we will use these to form our RBF-FD stencil weights. This ensures that continuity of motion and traction are upheld to $O(h^p)$ accuracy at $y'=0$ for a mildly-curved interface that is well-represented locally by a linear approximation. Just as in 1-D, we place these gathered linear relationships into rows in continuity matrices that relate expansion coefficients of u and v on one side of the interface to those on the other:

$$C_L \begin{bmatrix} \mathbf{u}_L \\ \mathbf{v}_L \end{bmatrix} = C_R \begin{bmatrix} \mathbf{u}_R \\ \mathbf{v}_R \end{bmatrix} \tag{38}$$

Again, given an expansion for u and v on one side of the interface, we can determine the shape of data in u and v to our desired order of accuracy on the other side of the interface by inverting either of the matrices in (38). Or, for example, if we have a matrix of column vectors U_R that form a basis for all possible Taylor expansion coefficients of u and v together on the "R" side of the interface, we can form a basis for data on both sides of the interface with a similar construction that we made in (26) for our 1-D example:

$$U = \begin{bmatrix} U_L \\ U_R \end{bmatrix} = \begin{bmatrix} C_L{}^{-1} C_R U_R \\ U_R \end{bmatrix} \tag{39}$$

Here, an easy choice for U_R might be the (12,12) identity matrix if we're staying consistent with our 2-D expansions up through degree 2 (recall that in 2-D, there are 6 polynomial expansion coefficients for both u and v on both sides of the interface, making U a (24,12) matrix). We complete our treatment of u and v by determining RBF-FD stencil weights to evaluate the three operators on u and v data that are relevant to the elastic wave equation:

$$\begin{cases} (\lambda+2\mu)u_x+\lambda v_y & (\mathit{for}\ f_t) \\ \mu(u_x+v_y) & (\mathit{for}\ g_t) \\ (\lambda+2\mu)v_y+\lambda u_x & (\mathit{for}\ h_t) \end{cases} \tag{40}$$

Determining stencils that act on stress tensor data (f, g, and h) is slightly problematic if we try to approach the task in the same way we did for the particle velocity fields. Since normal stress perpendicular to the interface may be discontinuous, we end up with too few relationships between the expansion coefficients of stress tensor data to exactly and explicitly determine its shape on one side of the interface based on Taylor coefficients present on the other side.

What we can do, though, is follow a procedure similar to the second method presented for characterizing data field f on both sides of the interface in 1-D ((27) through (30)). A simple and consistent way to begin is to find a matrix of basis function coefficients for u and v as in (39), with U_R equal to the appropriate identity matrix. All that remains, then, is to observe what Taylor coefficients in data fields f, g, and h emerge when columns of the basis coefficient matrix U are acted on by the PDE. In other words, time derivatives of the expansion coefficients of stress tensor components depend solely on expansion coefficients of particle velocity fields that are currently present in the data. Since we've already found a basis for the Taylor coefficients of particle velocity fields on both sides of the interface, all we need to do is observe what Taylor coefficients in f, g, and h emerge over time when our basis functions for u and v are acted on by the discrete form of the PDE (for the appropriate side of the interface). These combinations of f, g, and h Taylor coefficients (coupled together and across the interface) *contain* a basis for all *possible* combinations for f, g, and h Taylor coefficients on both sides of the interface, because they (together) are all the possibilities for f, g, and h coefficients that arise over time from the basis of valid combinations of u and v coefficients determined already. As in the 1-D case, in the matrix of time derivatives of concatenated f, g, and h expansion coefficients that arise from this process, we have to eliminate column vectors of coefficients that may create linear dependence relationships among the collection before we have a basis for the subspace of possible f, g, and h coefficient combinations.

Luckily, if we started by assuming the standard basis for coupled coefficients of u and v on one side of the interface, we can use this information to easily decide what column vectors of f, g, and h expansion coefficients we can eliminate from the collection. As in 1-D, we first eliminate the columns resulting from the PDE's effect on an isolated constant term present in either u or v on the side of the interface where the standard basis was assumed. A trivial result (all zeros) occurs in either case, and it is easy to see that we do not need the zero function in our basis. There is one more column vector of f, g, and h expansion coefficients that must be eliminated before a basis is obtained, and we have a choice: either the column vector of coefficients associated with the PDE's action on an isolated, linear "y" term in data field u or a linear "x" term in data field v need to be removed. As in the case of constant terms, we refer here to the PDE's action on these isolated expansion coefficients where they are present on the side of the interface where the standard basis was assumed. Once these three column vectors of f, g, and h expansion coefficients are removed, and those vectors of coefficients all culled to one degree less than the relationships between particle velocity fields are correct, one has a basis for all possible combinations of f, g, and h expansion coefficients on both sides of the interface. This basis of coupled piecewise polynomial functions can then be used to form RBF-FD stencils for the two spatial operators in the PDE that depend on spatial derivatives of the stress tensor fields:

$$\begin{cases} (f_x + g_y)/\rho & (for\, u_t) \\ (g_x + h_y)/\rho & (for\, v_t) \end{cases} \tag{41}$$

As performed above, the method for determining RBF-FD stencil weights works well if the interface is locally well-represented by a linear approximation. In practice, cases featuring very gradually curving, smooth interfaces and a relatively low resolution tend to perform quite well under this assumption. If desired, though, one can also account more rigorously for curvature of a smooth interface. The trigonometric functions involved in (34) and (35) may be replaced by local expansions of those functions in terms of the local horizontal coordinate x'. After application of a given power *k* of the differential operator in (37), a local expansion for the interface shape itself (again, in terms of x') can be inserted into entries for y'. Then, we can add together all rows of coefficients that express equivalences in continuous data field time derivatives in like powers of x'. Finally, enforcing that expansion coefficients associated with the first (*p*-*k*) monomial terms of x' sum to zero then upholds continuity of motion and traction to $O(h^p)$ accuracy not only for a locally flat interface, but also for any smooth interface. Note that this is technically identical to the method we use under the assumption of a flat interface. The only difference is that the method requires far less arithmetic when we assume that y' is equal to zero. We will examine the tradeoff between simplicity and accuracy of the local linear interface approximation in our first test case.

RBF-FD TEST CASES

In all test problems in this paper, we use dimensionless units of x and y (L), particle velocities u and v (L^1T^{-1}), Lamé parameters λ and μ $(M^1L^{-1}T^{-2})$, stress tensor components f, g, and h $(M^1L^{-1}T^{-2})$, and density ρ (ML^{-3}). Here, the upper-case letters L, T, and M denote a characteristic length, time, and mass scale of some system. Although our simulations are all carried out in 2-D, we retain 3-D units and assume we are simulating a 2-D slice of a 3-D system that does not vary in its third dimension.

All RBF-FD stencils in both test cases used Gaussian RBFs with a shape parameter (ε) calculated as follows:

$$\varepsilon = \frac{0.4}{d} \tag{42}$$

Here, d is the distance from the evaluation node of a stencil to its nearest neighbor. We have found that normalizing the shape parameter this way maintains a good compromise between stencil accuracy and conditioning of the linear systems solved to obtain stencil weights. One could attempt to gain additional accuracy by decreasing the constant in the numerator of (42), as long as the resulting weights produce an acceptably large and stable discrete time step.

All RBF-FD stencils that do not cross an interface contain the evaluation node and its 29 nearest neighbors for a total of 30 nodes. These stencils are supported by all polynomials up through 4th degree ($x^4, x^3y, \ldots, y^4$). RBF-FD stencils that cross an interface contain 19 nodes and are supported by all polynomials up through 3rd degree.

A small amount of Δ^3-type "hyperviscosity" is added to each data field to assure that RBF-FD operator eigenvalues fall in the left half of the complex plane, as described in detail in [16]. Stencil properties for RBF-FD hyperviscosity operators are exactly the same as those described above for the RBF-FD differential operators. To maintain a high degree of accuracy, it is important that hyperviscosity stencils that cross interfaces are supported by the same space of piecewise polynomials (determined by interface continuity conditions) that support stencils of the differential operators.

Fourth-order Runge-Kutta (RK4) time integration was used for all test cases presented in this paper. All RBF-FD stencils that cross an interface incorporate curvature information into calculation of stencil weights, with the sole exception of one illustrative example.

Test Case 1. Simple p-wave problem: two curved interfaces

Our first test problem involves two mildly curved interfaces within a doubly periodic unit square (**Figure 6**). Between the two interfaces, $\rho = 2$ and both λ and μ are defined by the following equation:

$$\lambda, \mu = 4 + \sin(2\pi x)\sin(2\pi y) \tag{43}$$

Elsewhere in the domain, all three model parameters (ρ, λ, μ) are equal to 1.

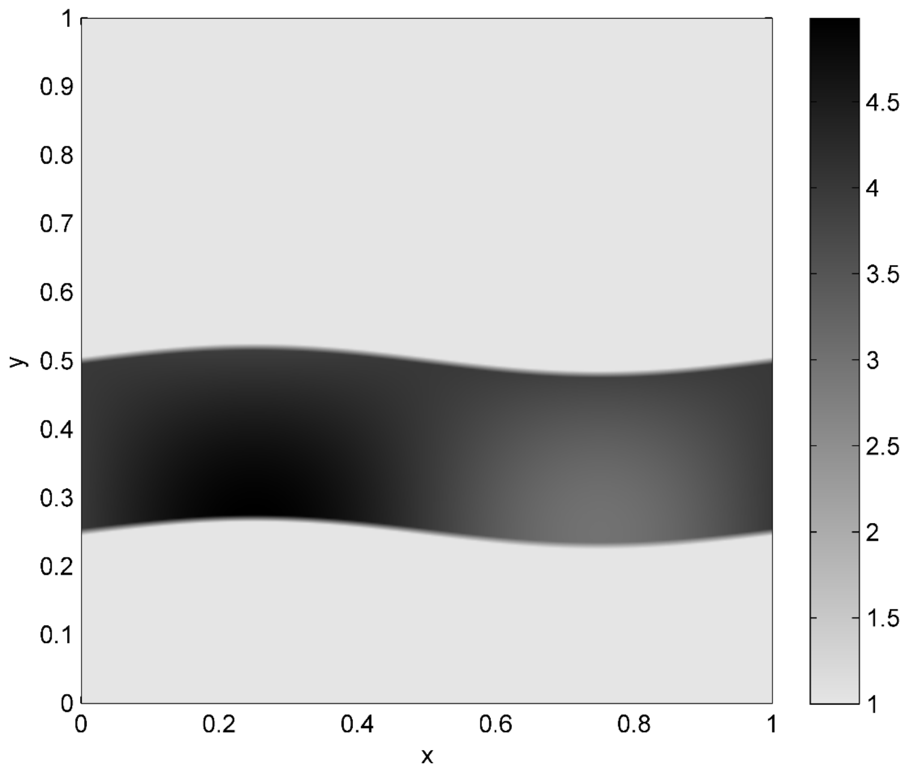

Figure 6. Greyscale map of Lamé parameters λ and μ for test case 1. These parameters are equal to each other everywhere in our test domain. Density has a value of 2.0 in the middle band and 1.0 elsewhere.

In this domain, A horizontal p-wave front begins at $y = 0.75$ and travels in the negative y-direction for 0.3 time units, encountering the mild sinusoidal interfaces near $y = 0.5$ and $y = 0.25$. **Figure 7(a)** shows a color map of data field v at $t = 0.0$, and **Figure 7(b)** shows the same data at $t = 0.3$.

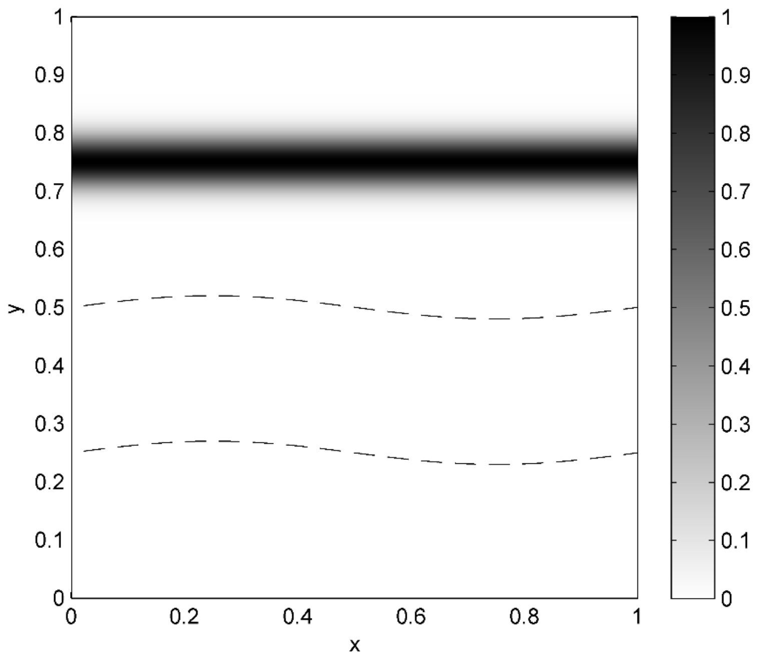

Figure 7a. Data field v from the first test case, at $t = 0.0$.

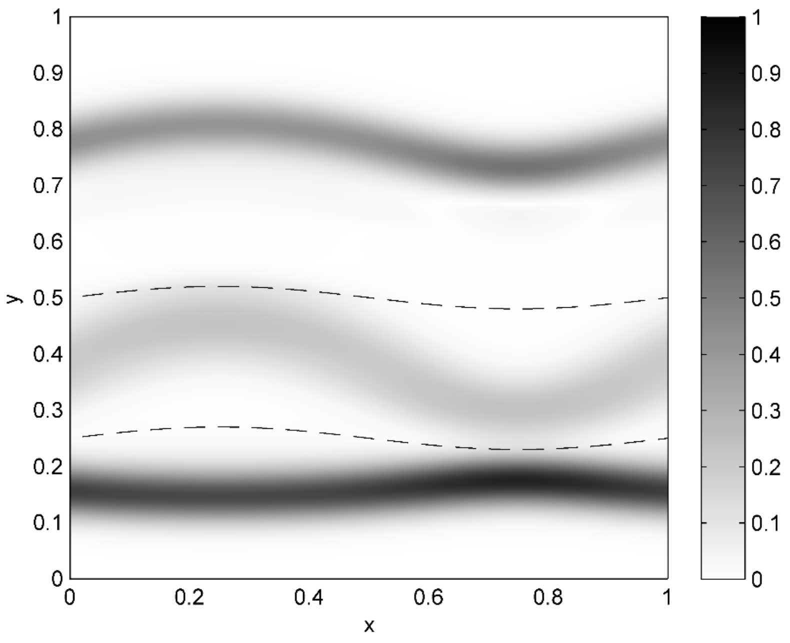

Figure 7b. Data field v from the first test case, at $t = 0.3$.

Figure 8 shows errors in data field *v* at *t* = 0.3 for different solution methods to test case 1 carried out at different resolutions. The methods differ in their spatial differentiation strategy. Represented here are fourth-order finite difference (FD4), pseudospectral (PS), RBF-FD, and hybrid RBF-FD/FD6 methods. Hybrid solutions used a node layout similar to that in **Figure 1**, where irregular node structure and RBF-FD methodology are only used in a small region near and across interfaces. Errors are calculated using a 640,000-node RBF-FD/FD6 hybrid solution as a reference.

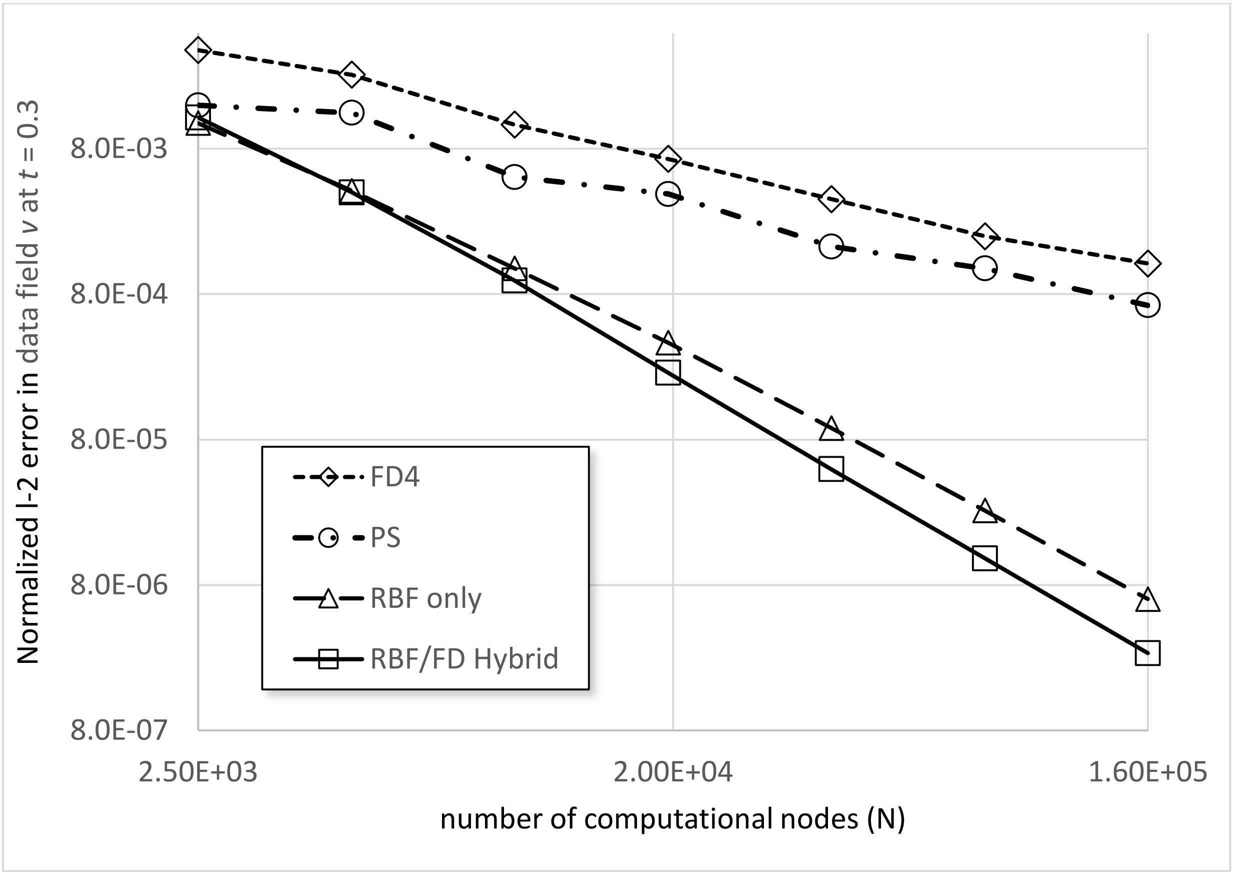

Figure 8. Convergence plot for different methods used to solve test case 1.

Figure 9 shows error vs. elapsed time expended in solving test case 1 at different resolutions with the various methods described above. The elapsed time displayed here is for time integration only, and does not include any time expended in preprocessing steps.

In **Figure 10**, convergence data are displayed for two different hybrid RBF-FD/FD6 approaches to test case 1. The first data series is simply the hybrid data reproduced from **Figure 8**. The second data series is from a sequence of solutions that are identical to the first in every way except one: weights for stencils that cross interfaces are calculated using a local, linear approximation to interface shape.

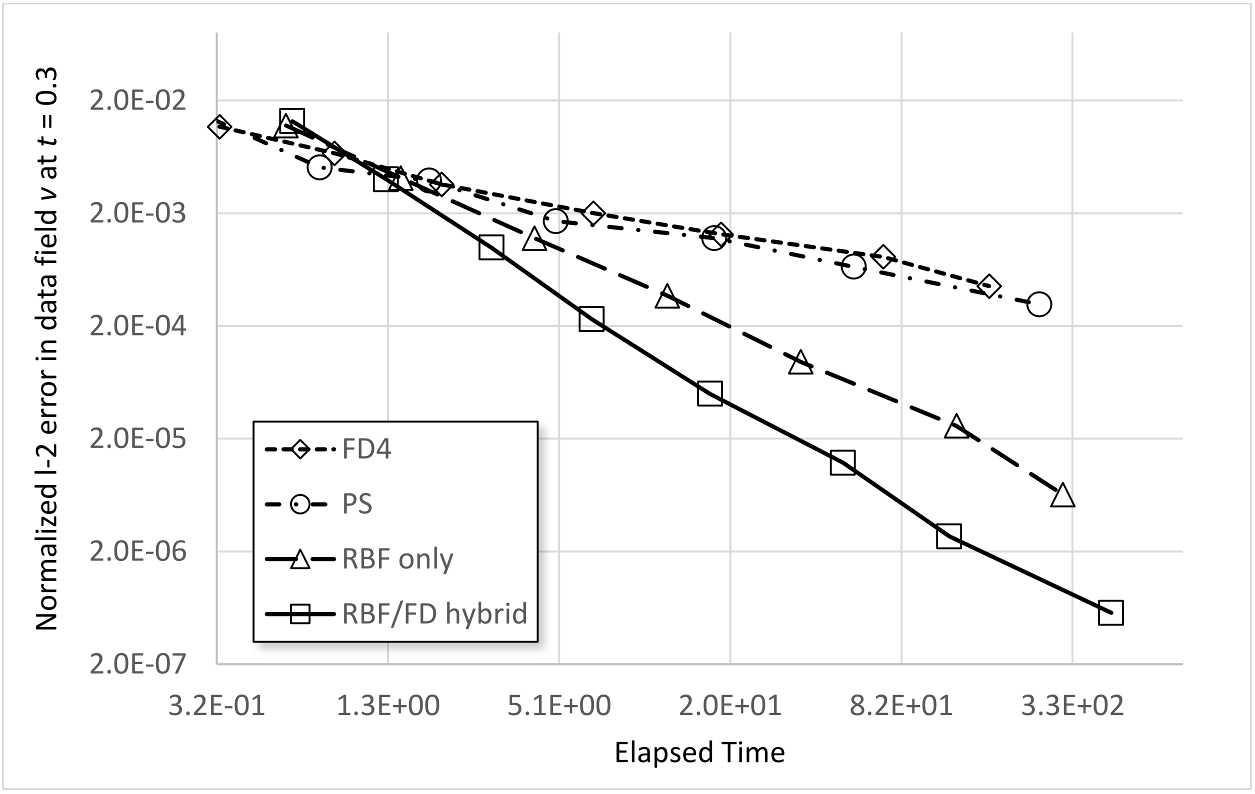

Figure 9. Performance plot for the 4 methods: error vs. wall clock time expended during RK4 time integration from t = 0.0 to t = 0.3 (on a 2.7 GHz 4-core i7 processor). Computation was performed in MATLAB with sparse and highly vectorized data structures, where possible.

In **Figure 10**, it can be seen that the locally linear approximation to interface shape begins to penalize solution quality (relative to a solution that incorporates curvature information into stencil weights) as we increase the number of nodes in the domain above about 40,000. Also shown in **Figure 10** are plots of perfect, synthetic 2nd-order and 4th-order convergence data. Comparing these with the slopes of our actual data, we see that the linear approximation to interface shape results in apparent 4th-order convergence on relatively coarse node sets. This convergence falters to around 2nd-order once the node set becomes more refined. On the other hand, inclusion of curvature information in stencil weights appears to result in 4th-order convergence throughout all RBF-FD solutions of test case 1 evaluated for this paper.

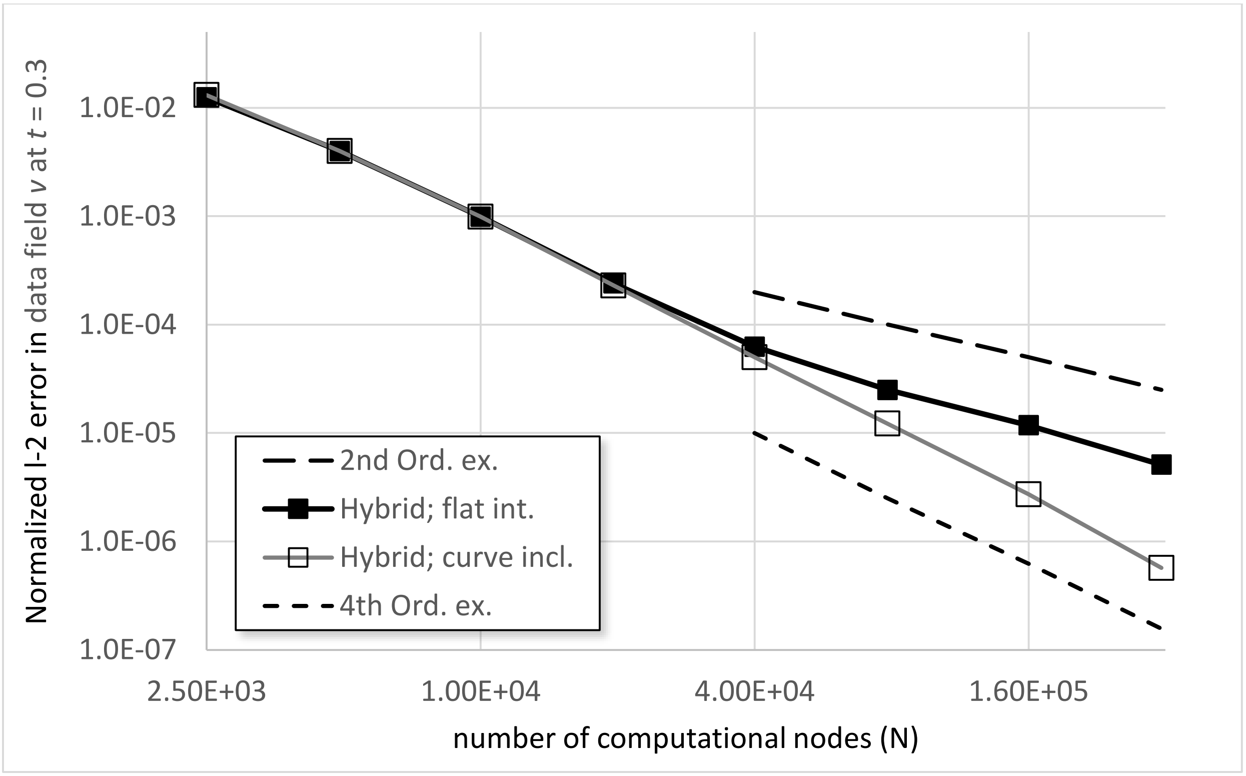

Figure 10. Convergence plot for hybrid method, curvature incl. vs. flat int. assumption

Example 2. Point source in a "mini-Marmousi" domain

Figure 11 shows the "mini-Marmousi" relative velocity model used in our second test case, and **Figure 12** displays data from a point source simulation in the region at t = 0.3.

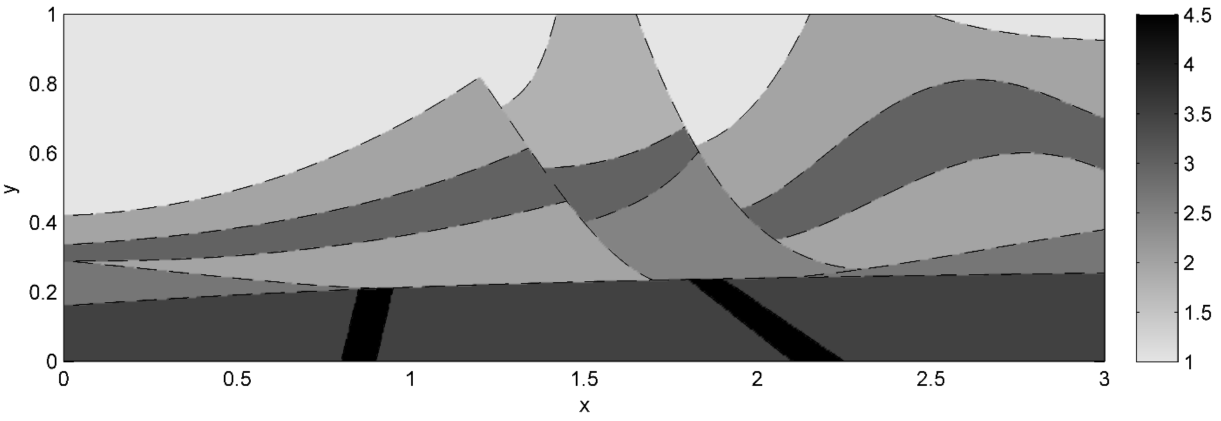

Figure 11. Relative velocities within the mini-Marmousi domain.

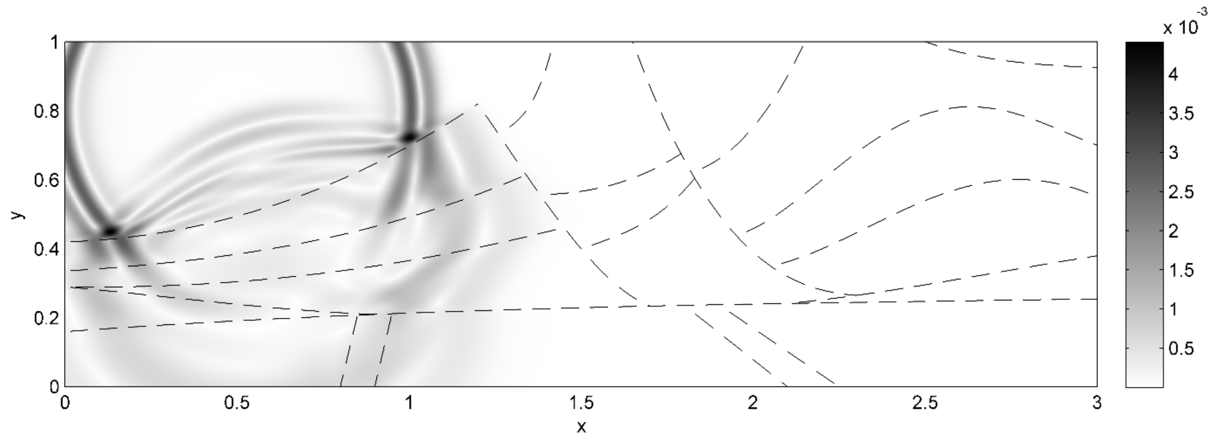

Figure 12. Absolute particle velocity $(u^2 + v^2)^{0.5}$ in test case 2 at t = 0.3.

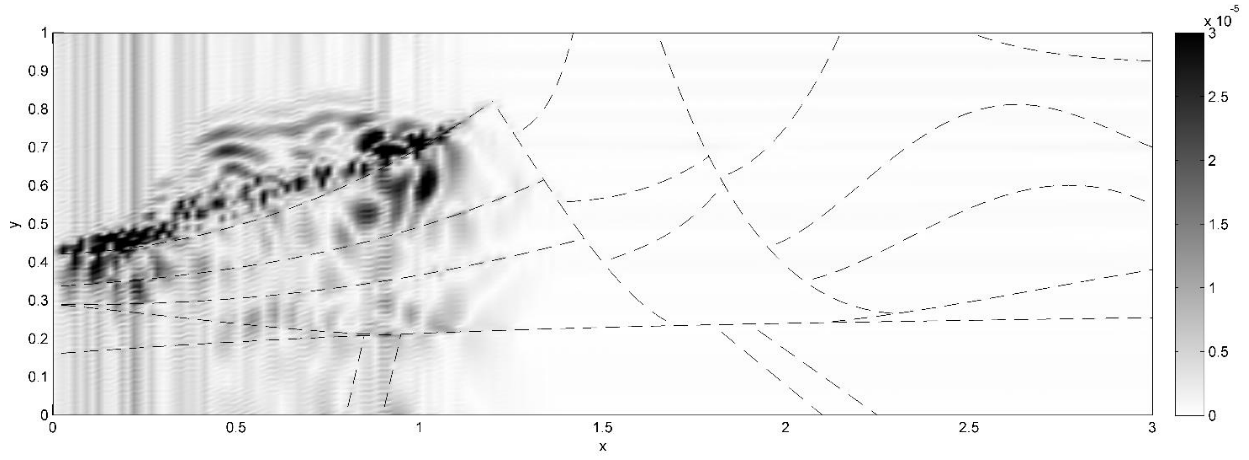
Figure 13a. Error in v at t = 0.3 of 240 x 720 PS solution (153,600 nodes)

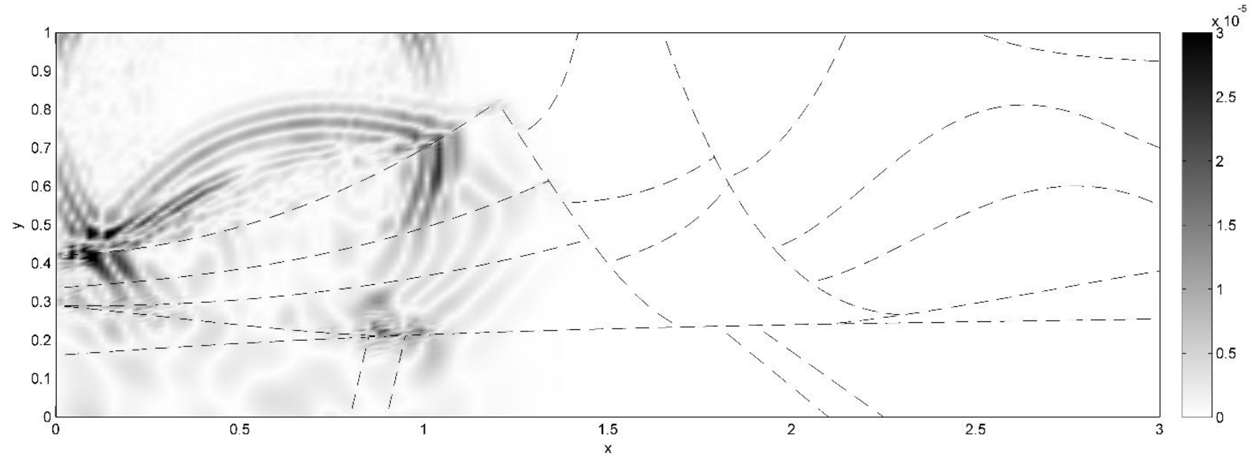
Figure 13b. Error in v at t = 0.3 of 153,600-node RBF-FD/AC solution (old method, from [3])

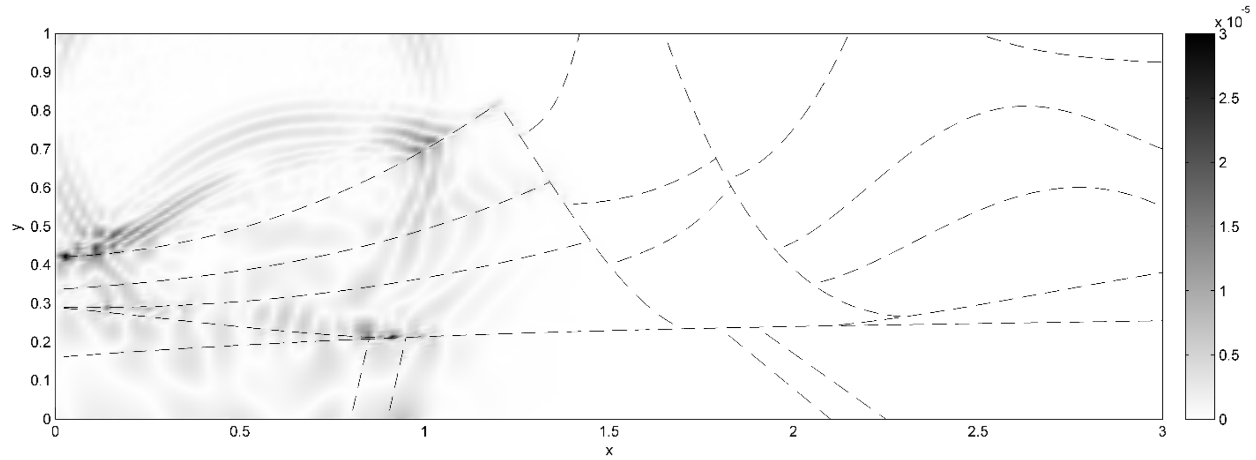
Figure 13c. Error in v at t = 0.3 of 153,600-node RBF-FD/AC solution (current method)

Figure 13 shows errors of three different 153,600-node RBF-FD solutions for the variable v at $t = 0.3$. The first of these solutions was obtained via pseudospectral spatial differentiation. The second and third solutions in **Figure 13** were obtained using the 3rd-order method we presented in [3] and the 4th-order method we have discussed in the present paper, respectively. All three solutions use a 614,400-node RBF-FD solution as a reference, and all three solutions employ split-field perfectly-matched layer (PML) absorbing boundary conditions as described in [17]. Here, the PML region is 0.1 units thick on each side of the domain. The PML boundary condition appears to be stable and compatible with the RBF-FD approach, even when interfaces extend into the PML region.

RBF-FD stencils that cross more than one interface are given no special treatment, and their weights are computed in the same manner as *Type 2* stencils. Even without upholding interface continuity conditions in the small zones around corners, cusps, and faults, we see that our current approach improves error significantly over the method presented in [3]. Both RBF-FD-based solution show much less error than the extremely high-order (pseudospectral) finite difference solution.

CONCLUSIONS

We presented and tested a simplified method for RBF-FD-based seismic modeling that can achieve fourth-order convergence even in the presence of curved interfaces. Especially when RBF-FD is hybridized with a traditional finite difference method, we have shown that our MATLAB prototypes function in a very cost-competitive manner compared to a finite difference approach applied everywhere in a heterogeneous domain. This localization of the interface treatment could be even more important in 3-D wave transport simulations, where storage and access of specialized weights for every single data node in a domain would create a very large computational burden both in terms of memory footprint and memory bandwidth demand.

In our second test case, RBF-FD stencils that crossed interface cusps, corners, or fault zones were determined without the special treatment given to stencils that crossed a single, smoothly-curved interface. Although RBF-FD solutions of the second test problem showed significant improvement over a very high-order finite difference approach, we will in the future pursue a more rigorous analysis of how the naïve treatment of small zones surrounding cusps, corners, and faults affects the accuracy of our approach.

ACKNOWLEDGEMENTS

We would like to thank Amik St-Cyr and Shell International Exploration and Production for valuable mentoring and funding during the development and implementation of techniques described in this paper. We would also like to thank the Graduate School of the University of Colorado at Boulder for financial support during final preparation of the manuscript. And finally, we would like to thank Natasha Flyer (University Corporation for Atmospheric Research) and the rest of the RBF research group at CU-Boulder for frequent and very helpful feedback on the mathematical techniques in this paper.

REFERENCES

[1] W.W. Symes, T. Vdovina, Interface error analysis for numerical wave propagation, Comput. Geosci. 13 (2009) 363–370. doi:10.1007/s10596-008-9124-8.

[2] D. Vishnevsky, V. Lisitsa, V. Tcheverda, G. Reshetova, Numerical study of the interface errors of finite difference-type simulations of seismic waves, Geophysics. 79 (2014) T219–T232. doi:10.1190/geo2013-0299.1.

[3] B. Martin, B. Fornberg, A. St-Cyr, Seismic modeling with radial basis function-generated finite differences, Geophysics. 80 (2015) T137–T146. doi:10.1190/geo2014-0492.1.

[4] B. Fornberg, The pseudospectral method: accurate representation of interfaces in elastic wave calculations, Geophysics. 53 (1988) 625–637. doi:10.1190/1.1442497.

[5] F. Muir, J. Dellinger, J. Etgen, D. Nichols, Modeling elastic fields across irregular boundaries, Geophysics. 57 (1992) 1189–1193. doi:10.1190/1.1443332.

[6] W.W. Symes, I.S. Terentyev, Subgrid modeling via mass lumping in constant density acoustics, in: SEG, 2009.

[7] C. Zhang, R.J. LeVeque, The immersed interface method for acoustic wave equations with discontinuous coefficients, Wave Motion. 25 (1997) 237–263. doi:10.1016/s0165-2125(97)00046-2.

[8] Y. Yu, Z. Chen, Implementation of material interface conditions in the radial point interpolation meshless method, IEEE Trans. Antennas Propag. 59 (2011) 2916–2923. doi:10.1109/tap.2011.2158969.

[9] B. Lombard, J. Piraux, Numerical treatment of two-dimensional interfaces for acoustic and elastic waves, J. Comput. Phys. 195 (2004) 90–116. doi:10.1016/j.jcp.2003.09.024.

[10] Z. Li, The immersed interface method using a finite element formulation, Appl. Numer. Math. 27 (1998) 253–267. doi:10.1016/s0168-9274(98)00015-4.

[11] E. Zhebel, S. Minisini, A. Kononov, W.A. Mulder, A comparison of continuous mass-lumped finite elements with finite differences for 3-D wave propagation, Geophys. Prospect. 62 (2014) 1111–1125. doi:10.1111/1365-2478.12138.

[12] B. Fornberg, N. Flyer, A primer on radial basis functions with applications to the geosciences, SIAM, Philadelphia, 2015.

[13] N. Flyer, B. Fornberg, G.A. Barnett, V. Bayona, On the role of polynomials in RBF-FD approximations: I. Interpolation and accuracy, J. Comput. Phys. (n.d.).

[14] Y. Liu, Fourier analysis of numerical algorithms for the Maxwell equations, J. Comput. Phys. 124 (1996) 396–416. doi:10.1006/jcph.1996.0068.

[15] N. Flyer, G.A. Barnett, L.J. Wicker, Enhancing finite differences with radial basis functions: experiments on the Navier-Stokes equations, J. Comput. Phys. 316 (2016) 39–62. doi:10.1016/j.jcp.2016.02.078.

[16] B. Fornberg, E. Lehto, Stabilization of RBF-generated finite difference methods for convective PDEs, J. Comput. Phys. 230 (2011) 2270–2285. doi:10.1016/j.jcp.2010.12.014.

[17] M. Zhou, Perfectly matched layers for the 2D elastic wave equation, (2003). http://utam.gg.utah.edu/tomo03/03_ann/pdf/mzhou_pmle.pdf (accessed April 18, 2016).