

SCATTERED NODE COMPACT FINITE DIFFERENCE-TYPE FORMULAS GENERATED FROM RADIAL BASIS FUNCTIONS

GRADY WRIGHT * AND BENGT FORNBERG †

Abstract. In standard equispaced finite difference (FD) formulas, symmetries can make the order of accuracy relatively high compared to the number of nodes in the FD stencil. With *scattered* nodes, such symmetries are no longer available. The generalization of compact FD (CFD) formulas that we propose for scattered nodes and RBFs achieves the goal of still keeping the number of stencil nodes small without a similar reduction in accuracy. We analyze the accuracy of these new compact RBF-FD formulas by applying them to some model problems, and study the effects of the shape parameter that arises in, for example, the multiquadric radial function.

Key words. Radial basis functions, RBF, partial differential equations, PDE, finite difference method, FDM, mesh-free, compact, *Mehrstellenverfahren*

AMS subject classifications. 41A21, 41A30, 41A63, 65D25, 65N06

1. Introduction. Radial basis functions (RBFs) are a primary tool for interpolating multidimensional scattered data. In the past decade or so they have also received increased attention as a “mesh-free” method for numerically solving partial differential equations (PDEs) on irregular domains by a global collocation approach (see for example [1, 2, 3, 4, 5, 6]). While these methods can be spectrally accurate (when proper attention is paid to boundaries), they generally result in having to solve a large, ill-conditioned, dense linear system. Some attempts have been made to resolve this issue [4, 7, 8] (and the references therein). However, stability issues have limited the use of RBFs for time dependent problems and adapting the methods for non-linear equations has proven to be difficult. To combat *all* of these problems, a “local” method has recently been proposed that gives up spectral accuracy for a sparse, better-conditioned linear system and more flexibility for handling non-linearities. This new “mesh-free” method is essentially a generalization of the classical finite difference (FD) method to scattered node layouts and appears to have been investigated independently by Shu *et al.* [9, 10], Tolstykh *et al.* [11], Cecil *et al.* [12], and the present authors [13].

In its most general form, the FD method is based on approximating some derivative of a function u at a given point based on a linear combination of the value of u at some surrounding points. In the classical 1-D case, $u(x)$ is given at n equispaced nodes:



and the k^{th} derivative of u at say $x = x_j$ ($1 \leq j \leq n$) is approximated by

$$\left. \frac{d^k u}{dx^k} \right|_{x=x_j} = u^{(k)}(x_j) \approx \sum_{i=1}^n c_{j,i}^k u(x_i). \quad (1.1)$$

*University of Utah, Department of Mathematics, 155 South 1400 East, JWB 233, Salt Lake City, UT 84112-0090, USA (wright@math.utah.edu). The work was supported by NSF VIGRE grant DMS-0091675.

†University of Colorado, Department of Applied Mathematics, 526 UCB, Boulder, CO 80309, USA (fornberg@colorado.edu). The work was supported by NSF grants DMS-9810751 (VIGRE) and DMS-0309803.

For example, for $n = 3$ nodes, i.e.



$k = 1$, and $j = 2$, we have the standard second order accurate approximation:

$$u'(x_2) \approx -\frac{1}{2h}u(x_1) + \frac{1}{2h}u(x_3), \quad (1.2)$$

where h is the spacing between the nodes. We refer to (1.1) as a FD formula, $c_{i,j}^k$ are called weights, and the collection of nodes is referred to as a stencil. It is common to represent FD formulas using a *computational molecule* containing the weights at the node points. The computational molecule for (1.2) is written

$$u'(x_2) = \boxed{-\frac{1}{2}} \text{---} \boxed{0} \text{---} \boxed{\frac{1}{2}} \frac{u}{h}. \quad (1.3)$$

For the 1-D case, the weights $c_{i,j}^k$ in (1.1) are usually computed using polynomial interpolation [14, 15]. These 1-D formulas can be combined to create FD formulas for partial derivatives in two and higher dimensions. This strategy, however, requires that the nodes of the stencils are situated on some kind of structured grid (or collection of structured grids), which severely limits the geometric flexibility of the FD method. One obvious approach for bypassing this problem is to instead allow the nodes of the FD stencil to be placed freely, so that a good discretization of the physical domain can be obtained. However, this approach also raises the question of how the weights of the scattered node FD formulas should be computed. Abgrall [16] and Schönauer and Adolph [17], for example, propose extending the classical polynomial interpolation technique. However, this idea has not become widely used, partly because it leads to several ambiguities about how to generate the FD formulas. For example, what mixed terms should be included in the multivariate polynomial interpolant to the nodes, and how should a set of nodes that leads to a singular polynomial interpolation problem be handled (polynomial interpolation in > 1 -D is not well-posed [18]).

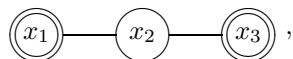
All of these ambiguities can be resolved if RBF interpolation is instead used to generate the weights in the FD formula [9, 11, 12, 13]. An RBF interpolant is formed by taking a linear combination of translates of a single univariate basis function $\phi(r)$ that is radially symmetric about its center. Thus, no decisions need to be made about what mixed terms to include in the interpolant. Moreover, for the appropriate choice of $\phi(r)$, the RBF interpolation method is well-posed in all dimensions. The following are some other properties that make RBF interpolants an attractive choice:

- Since the only geometric property used by an RBF interpolant is the pairwise distances between points, higher dimensions do not increase the coding complexity associated with computing the interpolants.
- RBF interpolants can be very accurate at approximating derivatives [19, 20].
- Certain types of radial functions $\phi(r)$ feature a “shape” parameter that allows them to vary from being nearly flat to sharply peaked. Recently it has been shown [21, 22, 23, 24] that, under some mild restrictions on $\phi(r)$, the RBF interpolants converge to polynomial interpolants in the limit of $\phi(r)$ becoming

entirely flat. Thus, all classical FD formulas can be recovered by the limiting RBF interpolant (when the nodes are arranged accordingly).

We refer to this method of using RBF interpolants to generate the weights of a FD formula as the RBF-FD method.

One of the issues with using scattered node FD formulas (whether the weights are generated by polynomials or RBFs) is that symmetries cannot be exploited to increase the accuracy of the formulas. Thus, the number of nodes in the stencils tend to be relatively large compared to the resulting accuracy. To circumvent this problem, we propose a generalization of a method introduced by Collatz [25] under the name *Mehrstellenverfahren*, and later developed into compact FD (CFD) formulas by Lele [26]. The basic idea behind this method is to keep the stencil size fixed and to also include in the FD formula a linear combination of derivatives of u at surrounding nodes. For example, the accuracy of (1.2) can be increased to fourth order by including the derivative of u at x_1 and x_3 . We represent this inclusion graphically with the stencil:



where a single circle indicates that the value of u is used at that node, and a double circle indicates that the value of u and u' are used. By including the derivative values at these node points, we arrive at the fourth order accurate approximation [25, p. 538]

$$u'(x_2) \approx -\frac{1}{4}u'(x_1) - \frac{1}{4}u'(x_3) - \frac{3}{4h}u(x_1) + \frac{3}{4h}u(x_3), \quad (1.4)$$

or, in computational molecule form

$$u'(x_2) \approx \boxed{-\frac{1}{4}} \text{---} \boxed{-\frac{1}{4}} u' + \boxed{-\frac{3}{4}} \text{---} \boxed{0} \text{---} \boxed{\frac{3}{4}} \frac{u}{h}. \quad (1.5)$$

In the case of 1-D and equispaced nodes, the weights for these CFD schemes can be conveniently derived using Padé approximants [15]. For scattered nodes, and for RBFs, this Padé approach is no longer available. Instead, we propose a method based on Hermite RBF interpolation.

The remainder of the paper is structured as follows: In Section 2, we review some properties of the standard and Hermite RBF interpolation methods. We also introduce some closed form expressions for the cardinal RBF interpolants which prove valuable for deriving the standard RBF-FD formulas and the compact RBF-FD (RBF-CFD) formulas in Section 3. In Section 4, we study the effect the shape parameter has on the RBF-FD and RBF-CFD formulas. In particular, we are interested in how the formulas behave in the limit of flat radial functions. We also review the Contour-Padé algorithm [27], which is an indispensable tool for studying formulas in the flat limit. In Section 5, we discuss some implementation details, including a procedure for selecting the stencils for the FD formulas. In Section 6, we present some examples that illustrate the effectiveness of the RBF-CFD formulas. Section 7 contains some concluding remarks.

Type of basis function	$\phi(r)$
Piecewise smooth RBFs	
Generalized Duchon spline	$r^{2k} \log r$, $k \in \mathbb{N}$
	$r^{2\nu}$, $\nu > 0$ and $\nu \notin \mathbb{N}$
Wendland	$(1-r)_+^k p(r)$, p a polynomial, $k \in \mathbb{N}$
Infinitely smooth RBFs	
Gaussian (GA)	$e^{-(\varepsilon r)^2}$
Generalized multiquadric	$(1 + (\varepsilon r)^2)^{\nu/2}$, $\nu \neq 0$ and $\nu \notin 2\mathbb{N}$
• Multiquadric (MQ)	$(1 + (\varepsilon r)^2)^{1/2}$
• Inverse multiquadric (IMQ)	$(1 + (\varepsilon r)^2)^{-1/2}$
• Inverse quadratic (IQ)	$(1 + (\varepsilon r)^2)^{-1}$

TABLE 2.1

Some commonly used radial basis functions. Note: in all cases, $\varepsilon > 0$.

2. RBF interpolation. Since the RBF-FD formulas are obtained from RBF interpolants, we review in this section some properties of RBF interpolation. We discuss standard RBF interpolation in which only function values are specified. This forms the background for Hermite RBF interpolation, in which function and derivative values are specified. In both cases, we derive some closed-form expressions for the cardinal interpolants which will be useful when deriving the RBF-FD formulas (Section 3) and when analytically studying the effect of the shape parameter ε (Section 4).

2.1. Standard RBF interpolation. Given a set of distinct data points $\underline{x}_i \in \mathbb{R}^d$, $i = 1, \dots, n$, and corresponding (scalar) function values $u(\underline{x}_i)$, $i = 1, \dots, n$, the standard RBF interpolation problem, is to find an interpolant of the form

$$s(\underline{x}) = \sum_{i=1}^n \lambda_i \phi(\|\underline{x} - \underline{x}_i\|) + \sum_{j=1}^m \beta_j p_j(\underline{x}), \quad (2.1)$$

where $\phi(r)$ is some radial function, $\|\cdot\|$ is the standard Euclidean norm, and $\{p_k(\underline{x})\}_{k=1}^m$ is a basis for the space of all d -variate polynomials that have degree $\leq Q$. The expansion coefficients λ_i and β_j are determined by enforcing the conditions

$$s(\underline{x}_i) = u(\underline{x}_i), \quad i = 1, \dots, n, \quad \text{and} \quad (2.2)$$

$$\sum_{i=1}^n \lambda_i p_j(\underline{x}_i) = 0, \quad j = 1, \dots, m. \quad (2.3)$$

The degree of the augmented polynomial in (2.1) depends on $\phi(r)$ and its inclusion may be necessary to guarantee a well-posed interpolation problem [28]. Table 2.1 lists a few of the many available choices for $\phi(r)$.

In this study, we are primarily interested in the infinitely smooth radial functions since, by suitable choices of the shape parameter ε , they can provide more accurate interpolants than the piecewise smooth case [29, 30]. We postpone the discussion on the effect of ε until Section 4. For now, we assume it is fixed at some non-zero value. Although (2.1) is well-posed without any polynomial augmentation for the GA, MQ,

IMQ, and IQ RBFs in Table 2.1, we let $Q = 0$ (i.e. $m = 1$) in order impose some desirable properties on the RBF-FD formulas. Thus, the standard RBF interpolant that we consider has the form

$$s(\underline{x}) = \sum_{i=1}^n \lambda_i \phi(\|\underline{x} - \underline{x}_i\|) + \beta, \quad (2.4)$$

Imposing the conditions (2.2) and (2.3) leads to the symmetric linear system of equations

$$\underbrace{\begin{bmatrix} \Phi & | & 1 \\ \hline 1 & | & 0 \end{bmatrix}}_A \begin{bmatrix} \lambda \\ \beta \end{bmatrix} = \begin{bmatrix} u \\ 0 \end{bmatrix}, \quad (2.5)$$

where $\Phi_{i,j} = \phi(\|\underline{x}_i - \underline{x}_j\|)$, $i, j = 1, \dots, n$.

When deriving classical FD formulas, it is often convenient to use the Lagrange form of the underlying interpolating polynomial (see for example [14, Chapter 3]). This idea will also be useful when deriving the RBF-FD formulas. Thus, the goal is to find an interpolant of the form

$$s(\underline{x}) = \sum_{i=1}^n \psi_i(\underline{x})u(\underline{x}_i), \quad (2.6)$$

where $\psi_i(\underline{x})$ are of the form (2.4) and satisfy the cardinal conditions

$$\psi_i(\underline{x}_k) = \begin{cases} 1 & \text{if } k = i \\ 0 & \text{if } k \neq i \end{cases}, \quad k = 1, \dots, n, \quad (2.7)$$

There turns out to be a nice closed-form expression for $\psi_i(\underline{x})$ that first appeared in [22]. Denote $A_k(\underline{x})$, $k = 1, \dots, n$, as the matrix A in (2.5) with the k^{th} row replaced by the vector

$$B(\underline{x}) = [\phi(\|\underline{x} - \underline{x}_1\|) \quad \phi(\|\underline{x} - \underline{x}_2\|) \quad \cdots \quad \phi(\|\underline{x} - \underline{x}_n\|) \quad | \quad 1]. \quad (2.8)$$

Note that with this notation $A = A_k(\underline{x}_k)$.

THEOREM 2.1 ([22]). *The cardinal RBF interpolant of the form (2.4) that satisfies (2.7) is given by*

$$\psi_i(\underline{x}) = \frac{\det(A_i(\underline{x}))}{\det(A)} \quad (2.9)$$

Proof. Expanding the determinant in the numerator along the i^{th} row shows that $\psi_i(\underline{x})$ is of the form (2.4). Additionally, a simple inspection of (2.9) reveals that it satisfies (2.7). The final step is to note that A is non-singular (for the appropriate choice of $\phi(r)$), which gives us existence and uniqueness. \square

While (2.9) is not recommended for computational work, it is useful for analytically studying the effects of the shape parameter ε in Section 4.

2.2. Hermite RBF interpolation. Let \mathcal{L} be some linear differential operator (e.g., the Laplacian, $\mathcal{L} = \Delta$) and let σ be a vector containing some combination of $m \leq n$ distinct numbers from the set $\{1, \dots, n\}$. Then the Hermite interpolation problem we consider is as follows:

PROBLEM 2.2 (Hermite Interpolation Problem). *Given a set of distinct data points $\underline{x}_i \in \mathbb{R}^d$, $i = 1, \dots, n$, and corresponding (scalar) data values $u(\underline{x}_i)$, $i = 1, \dots, n$, and $\mathcal{L}u(\underline{x}_{\sigma_i})$, $i = 1, \dots, m$, find a function that interpolates u and $\mathcal{L}u$ at the given data points. Here we define*

$$\mathcal{L}u(\underline{x}_j) := \mathcal{L}u(\underline{x})|_{\underline{x}=\underline{x}_j}.$$

To clarify the notation of the example, suppose u is given at the nodes $\underline{x}_1, \underline{x}_2, \dots, \underline{x}_{10}$ in Figure 3.1 and $\mathcal{L}u$ is given at the nodes $\underline{x}_2, \underline{x}_4, \underline{x}_5, \underline{x}_6, \underline{x}_{10}$. Then using the notation of the problem $n = 10$, $m = 5$, and $\sigma = \{2, 4, 5, 6, 10\}$.

Solutions to the above Hermite problem with RBFs have been around since Hardy's introduction of the MQ method [31] (see also [32]). Other expositions on this and related Hermite-type RBF problems can be found, for example, in [33, 34, 35]. We review the standard RBF solution to this problem.

The standard method for solving the above problem is similar to the Hermite-Birkhoff method proposed by Wu [35]. The goal is to introduce m more unknowns to (2.4) in order to satisfy the m new interpolation conditions. Formally, the interpolant has the form

$$s(\underline{x}) = \sum_{i=1}^n \lambda_i \phi(\|\underline{x} - \underline{x}_i\|) + \sum_{j=1}^m \alpha_j \mathcal{L}_2 \phi(\|\underline{x} - \underline{x}_{\sigma_j}\|) + \beta. \quad (2.10)$$

where

$$\mathcal{L}_2 \phi(\|\underline{x} - \underline{x}_j\|) := \mathcal{L} \phi(\|\underline{x} - \underline{y}\|)|_{\underline{y}=\underline{x}_j},$$

i.e. \mathcal{L} acts on ϕ as a function of \underline{y} . For the remainder of this article we will also use the notation

$$\mathcal{L} \phi(\|\underline{x}_j - \underline{y}\|) = \mathcal{L}_1 \phi(\|\underline{x}_j - \underline{y}\|) := \mathcal{L} \phi(\|\underline{x} - \underline{y}\|)|_{\underline{x}=\underline{x}_j},$$

i.e. if \mathcal{L} or \mathcal{L}_1 act on ϕ , then the operator applies to the first variable. For example, if $d = 1$ and $\mathcal{L} = \frac{d}{dx}$ then $\mathcal{L}_2 \phi(\|x - x_j\|) = -\mathcal{L}_1 \phi(\|x - x_j\|) = -\mathcal{L} \phi(\|x - x_j\|)$.

Imposing the interpolation conditions (2.2), (2.3), and

$$\mathcal{L}s(\underline{x}_{\sigma_i}) = \mathcal{L}u(\underline{x}_{\sigma_i}), \quad i = 1, \dots, m. \quad (2.11)$$

leads to the following linear system of equations

$$\underbrace{\begin{bmatrix} \Phi & \Phi \mathcal{L}_2 & 1 \\ \Phi \mathcal{L}_1 & \Phi \mathcal{L}_1 \mathcal{L}_2 & 0 \\ 1 & 0 & 0 \end{bmatrix}}_{A^H} \begin{bmatrix} \lambda \\ \alpha \\ \beta \end{bmatrix} = \begin{bmatrix} u \\ \mathcal{L}u \\ 0 \end{bmatrix}, \quad (2.12)$$

where Φ is the matrix from (2.5) and

$$\begin{aligned}\Phi_{i,j}^{\mathcal{L}_2} &= \mathcal{L}_2\phi(\|\underline{x}_i - \underline{x}_{\sigma_j}\|), & i = 1, \dots, n, j = 1, \dots, m, \\ \Phi_{i,j}^{\mathcal{L}_1} &= \mathcal{L}_1\phi(\|\underline{x}_{\sigma_i} - \underline{x}_j\|), & i = 1, \dots, m, j = 1, \dots, n, \\ \Phi_{i,j}^{\mathcal{L}_1\mathcal{L}_2} &= \mathcal{L}_1[\mathcal{L}_2\phi(\|\underline{x}_{\sigma_i} - \underline{x}_{\sigma_j}\|)], & i = 1, \dots, m, j = 1, \dots, m,\end{aligned}$$

Note that by using \mathcal{L}_1 and \mathcal{L}_2 in defining (2.12), we have the relationship $\Phi^{\mathcal{L}_1} = (\Phi^{\mathcal{L}_2})^T$, making A^H symmetric. For the appropriate choice of ϕ , A^H is also guaranteed to be non-singular [34].

We can also express (2.10) in the following Lagrange form:

$$s(\underline{x}) = \sum_{i=1}^n \psi_i(\underline{x})u(\underline{x}_i) + \sum_{j=1}^m \tilde{\psi}_{\sigma_j}(\underline{x})\mathcal{L}u(\underline{x}_{\sigma_j}), \quad (2.13)$$

where $\psi_i(\underline{x})$ and $\tilde{\psi}_{\sigma_j}(\underline{x})$ are of the form (2.10) and satisfy the cardinal conditions

$$\psi_i(\underline{x}_k) = \begin{cases} 1 & \text{if } k = i \\ 0 & \text{if } k \neq i \end{cases}, \quad k = 1, \dots, n, \quad (2.14)$$

$$\mathcal{L}\psi_i(\underline{x}_{\sigma_k}) = 0, \quad k = 1, \dots, m, \quad (2.15)$$

and

$$\tilde{\psi}_{\sigma_j}(\underline{x}_k) = 0, \quad k = 1, \dots, n, \quad (2.16)$$

$$\mathcal{L}\tilde{\psi}_{\sigma_j}(\underline{x}_{\sigma_k}) = \begin{cases} 1 & \text{if } k = j \\ 0 & \text{if } k \neq j \end{cases}, \quad k = 1, \dots, m. \quad (2.17)$$

Like the standard RBF interpolation case, there turns out to be a nice closed-form expression for $\psi_i(\underline{x})$ and $\tilde{\psi}_{\sigma_j}(\underline{x})$. Denote $A_k^H(\underline{x})$, $k = 1, \dots, n + m$, as the matrix A^H in (2.12) with the k^{th} row replaced by the vector

$$B^H(\underline{x}) = [B(\underline{x}) \mid \mathcal{L}_2\phi(\|\underline{x} - \underline{x}_{\sigma_1}\|) \quad \cdots \quad \mathcal{L}_2\phi(\|\underline{x} - \underline{x}_{\sigma_m}\|) \mid 1], \quad (2.18)$$

where $B(\underline{x})$ is given by (2.8), but without the last entry.

THEOREM 2.3. *Let $\psi_i(\underline{x})$ and $\tilde{\psi}_{\sigma_j}(\underline{x})$ be of the form (2.10) and satisfy the conditions (2.14), (2.15) and (2.16), (2.17). Then, provided the matrix A^H in (2.12) is non-singular,*

$$\psi_i(\underline{x}) = \frac{\det(A_i^H(\underline{x}))}{\det(A^H)}, \quad (2.19)$$

and

$$\tilde{\psi}_{\sigma_i}(\underline{x}) = \frac{\det(A_{n+i}^H(\underline{x}))}{\det(A^H)}, \quad (2.20)$$

Proof. The result follows by using the same inspection argument as used in the proof of theorem 2.1. \square

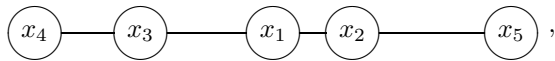
We postpone the discussion of the consequences of this formula as it relates to the shape parameter ε until Section 4.

3. RBF-FD Formulation. In this section we describe how the RBF-FD and RBF-CFD formulas are generated. With out loss of generality, we consider a stencil consisting of n (scattered) nodes $\underline{x}_1, \dots, \underline{x}_n$ and are interested in approximating $\mathcal{L}u(\underline{x}_1)$, for some linear differential operator \mathcal{L} . In Section 5, we discuss a procedure for selecting the points of a stencil from the much larger data set and how to select ε for each stencil.

3.1. RBF-FD method. The goal is to find weights c_i such that,

$$\mathcal{L}u(\underline{x}_1) \approx \sum_{i=1}^n c_i u(\underline{x}_i). \quad (3.1)$$

For example, given the 1-D scattered node stencil



we look for an approximation of the form

$$\mathcal{L}u(x_1) \approx \boxed{c_4} - \boxed{c_3} - \boxed{c_1} - \boxed{c_2} - \boxed{c_5} u.$$

See Figure 3.1 for a possible scattered node stencil in 2-D.

Using the Lagrange form of the standard RBF interpolant (2.6) to approximate $u(\underline{x})$ and then applying \mathcal{L} gives

$$\mathcal{L}u(\underline{x}_1) \approx \mathcal{L}s(\underline{x}_1) = \sum_{i=1}^n \mathcal{L}\psi_i(\underline{x}_1) u(\underline{x}_i).$$

Thus, the weights in RBF-FD formula (3.1) are formally given by

$$c_i = \mathcal{L}\psi_i(\underline{x}_1).$$

In practice, the weights are computed by solving the linear system

$$A[c|\mu]^T = (\mathcal{L}B(\underline{x}_1))^T \quad (3.2)$$

where A is the matrix in (2.5), $B(\underline{x})$ is the row vector (2.8), and μ is a dummy value related to the constant β in (2.4). Note that the constraint (2.3) enforces the condition

$$\sum_{i=1}^n c_i = 0,$$

i.e. the RBF-FD stencil is exact for all constants.

3.2. RBF-CFD method. The goal now is to increase the accuracy of the approximation (3.1) without increasing the stencil size by using nodes where u and $\mathcal{L}u$ are given exactly. Let σ be a vector containing some combination of $m < n$ distinct numbers from the set $\{2, \dots, n\}$, then we seek to find weights c_i and \tilde{c}_{σ_j} such that

$$\mathcal{L}u(\underline{x}_1) \approx \sum_{j=1}^m \tilde{c}_{\sigma_j} \mathcal{L}u(\underline{x}_{\sigma_j}) + \sum_{i=1}^n c_i u(\underline{x}_i). \quad (3.3)$$

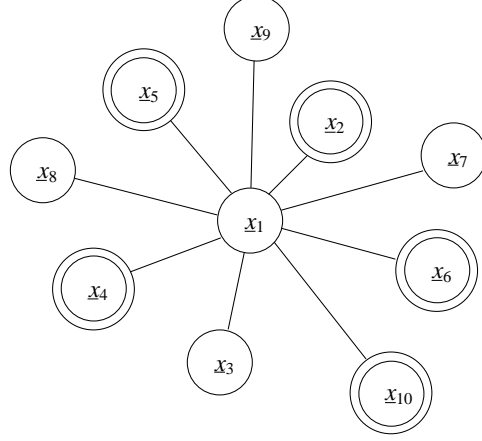
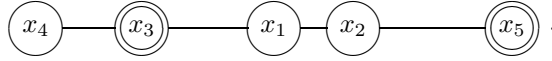


FIG. 3.1. Example of a scattered node stencil for use with the RBF-FD method.

For example, suppose we wish to increase the accuracy of the 1-D example from the previous section by including values of $\mathcal{L}u(x_3)$ and $\mathcal{L}u(x_5)$ in (3.1), i.e. using the stencil



Then we let $m = 2$, $\sigma = \{3, 5\}$, and look for an approximation of the form

$$\mathcal{L}u(x_1) \approx \begin{array}{c} \boxed{\tilde{c}_3} \text{---} \boxed{\tilde{c}_5} \mathcal{L}u + \\ \boxed{c_4} \text{---} \boxed{c_3} \text{---} \boxed{c_1} \text{---} \boxed{c_2} \text{---} \boxed{c_5} u . \end{array}$$

Figure 3.1 shows an example of a 2-D scattered node stencil. For this example, $m = 5$ and $\sigma = \{2, 4, 5, 6, 10\}$.

Using the Lagrange form of the Hermite RBF interpolant (2.13) to approximate $u(\underline{x})$ and then applying \mathcal{L} gives

$$\mathcal{L}u(\underline{x}_1) \approx \mathcal{L}s(\underline{x}_1) = \sum_{i=1}^n \mathcal{L}\psi_i(\underline{x}_1)u(\underline{x}_i) + \sum_{j=1}^m \mathcal{L}\tilde{\psi}_{\sigma_j}(\underline{x}_1)\mathcal{L}u(\underline{x}_{\sigma_j}) .$$

Thus, the weights in the RBF-CFD formula (3.3) are formally given by

$$c_i = \mathcal{L}\psi_i(\underline{x}_1) \quad \text{and} \quad \tilde{c}_{\sigma_j} = \mathcal{L}\tilde{\psi}_{\sigma_j}(\underline{x}_1) .$$

In practice, the weights are computed by solving the linear system

$$A^H[c|\tilde{c}|\mu]^T = (\mathcal{L}B^H(\underline{x}_1))^T \quad (3.4)$$

where A^H is the matrix in (2.12), $B^H(\underline{x})$ is the row vector (2.18), and μ is a dummy value related to the constant β in (2.10). Like the standard RBF-FD formula, the constraint (2.3) enforces that (3.3) is exact for all constants.

Note that the above formulation for computing CFD formulas differs from the standard polynomial-based formulation. For polynomials, the point at which we are constructing the approximation about does not matter; the only thing that does matter is that the formulas are exact for as high a degree polynomial as possible [15].

4. Observations on the effect of the shape parameter ε . In this section, we make some observations with regard to ε on the RBF-FD formulas. We are primarily interested in studying the resulting RBF-FD formulas as we let $\varepsilon \rightarrow 0$.

4.1. Some theoretical results on the $\varepsilon \rightarrow 0$ limit. For data points in 1-D, it was first shown in [21] that, under some mild restrictions on the infinitely smooth radial functions $\phi(r)$ (e.g. the multiquadric), the standard RBF interpolant converges to the Lagrange interpolating polynomial as $\varepsilon \rightarrow 0$. This result means that all “classical” polynomial-based FD stencils are reproduced by the standard RBF-FD stencils (with the appropriate choice of $\phi(r)$) in the limit of $\varepsilon \rightarrow 0$. A few examples of this result is given in the next section.

For data points in two and higher dimensions, the situation is complicated by the fact that multivariate polynomial interpolation is not well-posed [18]. While the standard RBF interpolant typically converges to a low degree multivariate polynomial as $\varepsilon \rightarrow 0$, there are circumstances where the interpolant may instead diverge; see [22, 23, 24, 27, 36] for more details.

The authors are unaware of any similar study of limiting ($\varepsilon \rightarrow 0$) Hermite RBF interpolants. However, we expect similar results to those of the standard RBF interpolant. For example, the following theorem indicates that polynomial type results may be expected in the $\varepsilon \rightarrow 0$ limit.

THEOREM 4.1. *Consider the Hermite RBF interpolant (2.10). If $\lim_{\varepsilon \rightarrow 0} s(\underline{x})$ exists, it will be a (multivariate) finite degree polynomial in \underline{x} .*

Proof. Consider the Lagrange form of $s(\underline{x})$ (2.13). Note that we can expand $\phi(\|\underline{x} - \underline{x}_j\|)$ and $\mathcal{L}_2\phi(\|\underline{x} - \underline{x}_j\|)$ in powers of ε^2 so that the coefficients in front of these powers will be some polynomial in \underline{x} . The same will, therefore, hold for the determinant in the numerator of (2.19) and (2.20). Thus, the ratios in (2.19) and (2.20) will be of the respective forms

$$\psi_i(\underline{x}) = \frac{\varepsilon^{2p_i} \{\text{poly in } \underline{x}\} + \varepsilon^{2p_i+2} \{\text{poly in } \underline{x}\} + \dots}{\varepsilon^{2q_i} \{\text{constant}\} + \varepsilon^{2q_i+2} \{\text{constant}\} + \dots}$$

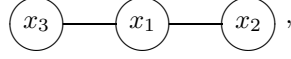
and

$$\tilde{\psi}_{\sigma_j}(\underline{x}) = \frac{\varepsilon^{2\tilde{p}_j} \{\text{poly in } \underline{x}\} + \varepsilon^{2\tilde{p}_j+2} \{\text{poly in } \underline{x}\} + \dots}{\varepsilon^{2\tilde{q}_j} \{\text{constant}\} + \varepsilon^{2\tilde{q}_j+2} \{\text{constant}\} + \dots},$$

where p_i , q_i , \tilde{p}_j , and \tilde{q}_j are positive integers. Since $\psi_i(\underline{x}_i) = 1$ and $\mathcal{L}\tilde{\psi}_{\sigma_j}(\underline{x}_{\sigma_j}) = 1$, it is impossible to have $p_i > q_i$ and $\tilde{p}_j > \tilde{q}_j$. If $p_i < q_i$ or $\tilde{p}_j < \tilde{q}_j$, the limit fails to exist. Thus, when $p_i = q_i$ and $\tilde{p}_j = \tilde{q}_j$, $s(\underline{x})$ is some polynomial in \underline{x} . \square

Like the polynomial result for the standard RBF interpolant, we can use Theorem 4.1 to conclude that when the underlying Hermite interpolant exists, the RBF-CFD formulas will be exact for some polynomials. In the next section and Section 4.4, we give some examples demonstrate this result.

4.2. A few examples with closed-form solutions for the $\varepsilon \rightarrow 0$ limit. We first present two examples based on an $n = 3$ node, equispaced 1-D stencil with the nodes numbered



where $\{x_1, x_2, x_3\} = \{0, h, -h\}$. Since the nodes are in 1-D, we drop the underline on the x -values. In all cases, we use a general radial function with an expansion

$$\phi(r) = a_0 + a_1(\varepsilon r)^2 + a_2(\varepsilon r)^4 + a_3(\varepsilon r)^6 \dots ,$$

and consider the resulting RBF-FD formulas as $\varepsilon \rightarrow 0$. We make use of the cardinal interpolants from Theorems 2.1 and 2.3.

EXAMPLE 4.2. Approximate $u'(x_1)$, i.e let $\mathcal{L} = \frac{d}{dx}$

RBF-FD formula: The determinant in the denominators of the cardinal interpolants (2.9) is given by

$$\det(A) = 6bh^6 F(a)\varepsilon^6 + O(\varepsilon^8) ,$$

while the determinants in the numerators are given by

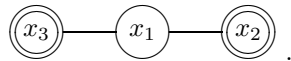
$$\begin{aligned} \det(A_1(x)) &= -2bh^4(4x^2 - 3h^2)F(a)\varepsilon^6 + O(\varepsilon^8), \\ \det(A_2(x)) &= bh^4x(4x + 3h)F(a)\varepsilon^6 + O(\varepsilon^8), \\ \det(A_3(x)) &= bh^4x(4x - 3h)F(a)\varepsilon^6 + O(\varepsilon^8) , \end{aligned}$$

where $b = 8$ and

$$F(a) = a_1 a_2 . \tag{4.1}$$

As expected from [21], provided $a_1 a_2 \neq 0$, $\psi_i(x)$, $i = 1, 2, 3$, converge to the standard cardinal interpolating polynomials in the limit as $\varepsilon \rightarrow 0$. Thus, the weights in the limiting RBF-FD scheme for $u'(x_1)$ (3.1) are the same as the classical, centered, second order FD stencil given by (1.3).

RBF-CFD formula: Using the notation from Section 2.2 and 3.2, we let $m = 2$ and $\sigma = \{2, 3\}$, i.e.



The determinant in the denominators of the cardinal interpolants (2.19) and (2.20) is given by

$$\det(A^H) = 4bh^{16} F(a)\varepsilon^{20} + O(\varepsilon^{22}) , \tag{4.2}$$

while the determinants in the numerators are given by

$$\det(A_1^H(x)) = -4bh^{12}(x^2 - h^2)^2 F(a)\varepsilon^{20} + O(\varepsilon^{22}), \quad (4.3)$$

$$\det(A_2^H(x)) = bh^{12}(2x - 3h)x(x + h)^2 F(a)\varepsilon^{20} + O(\varepsilon^{22}), \quad (4.4)$$

$$\det(A_3^H(x)) = bh^{12}(2x + 3h)x(x - h)^2 F(a)\varepsilon^{20} + O(\varepsilon^{22}), \quad (4.5)$$

$$\det(A_4^H(x)) = bh^{13}(x - h)x(x + h)^2 F(a)\varepsilon^{20} + O(\varepsilon^{22}), \quad (4.6)$$

$$\det(A_5^H(x)) = bh^{13}(x - h)^2 x(x + h) F(a)\varepsilon^{20} + O(\varepsilon^{22}), \quad (4.7)$$

where $b = 7680$ and

$$F(a) = (2a_2^2 - 5a_1a_3)(15a_3^2 - 28a_2a_4). \quad (4.8)$$

Provided $(2a_2^2 - 5a_1a_3) \neq 0$ and $(15a_3^2 - 28a_2a_4) \neq 0$, $\psi_i(x)$, $i = 1, 2, 3$, and $\tilde{\psi}_j(x)$, $j = 2, 3$, converge to the standard Hermite cardinal interpolating polynomials as $\varepsilon \rightarrow 0$. The weights in the limiting formula for $u'(x_1)$ (3.3) are the same as the classical, centered, fourth order CFD scheme given in (1.5). \square

EXAMPLE 4.3. Approximate $u''(x_1)$, i.e let $\mathcal{L} = \frac{d^2}{dx^2}$

RBF-FD formula: The cardinal interpolants for this example are the same as the ones from Example 4.2. Thus again, provided $a_1a_2 \neq 0$, the weights in the limiting RBF-FD stencil for $u''(x_1)$ (3.1) are same as the classical centered second order FD scheme:

$$u'(x_1) \approx \boxed{1} \text{---} \boxed{-2} \text{---} \boxed{1} \frac{u}{h^2}.$$

RBF-CFD formula: We again let $m = 2$ and $\sigma = \{2, 3\}$. The determinant in denominators of the cardinal interpolants (2.19) and (2.20) is given by

$$\det(A^H) = 60bh^{12}F(a)\varepsilon^{20} + O(\varepsilon^{22}), \quad (4.9)$$

while the determinants in the numerators are given by

$$\det(A_1^H(x)) = -12bh^8(x - h)(x + h)(x^2 - 5h^2)F(a)\varepsilon^{20} + O(\varepsilon^{22}), \quad (4.10)$$

$$\det(A_2^H(x)) = 6bh^8x(x + h)(x^2 - hx - 5h^2)F(a)\varepsilon^{20} + O(\varepsilon^{22}), \quad (4.11)$$

$$\det(A_3^H(x)) = 6bh^8(x - h)x(x^2 + hx - 5h^2)F(a)\varepsilon^{20} + O(\varepsilon^{22}), \quad (4.12)$$

$$\det(A_4^H(x)) = bh^{10}(x - h)x(x + h)(3x + 5h)F(a)\varepsilon^{20} + O(\varepsilon^{22}), \quad (4.13)$$

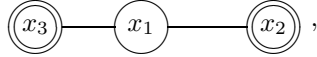
$$\det(A_5^H(x)) = bh^{10}(3x - 5h)(x - h)x(x + h)F(a)\varepsilon^{20} + O(\varepsilon^{22}), \quad (4.14)$$

where $b = 115200$ and $F(a)$ is given by (4.8). Provided again $F(a) \neq 0$, we recover the standard cardinal Hermite interpolating polynomials in the limit of $\varepsilon \rightarrow 0$. The weights in the limiting RBF-CFD scheme for $u''(x_1)$ (3.3) are

$$u''(x_1) \approx \boxed{-\frac{1}{10}} \text{---} \boxed{-\frac{1}{10}} u'' + \boxed{\frac{6}{5}} \text{---} \boxed{-\frac{12}{5}} \text{---} \boxed{\frac{6}{5}} \frac{u}{h^2}.$$

This is identical to the classical, centered, fourth order CFD scheme of Collatz' [25, p. 538]. \square

The last example illustrates how symmetries in the stencil can make the order of accuracy relatively high compared to the number of nodes. For the regular three node stencil, the weights come from the second degree interpolating polynomial. This may lead us to conclude that, after differentiating twice, the order of accuracy of the formula is only one (i.e. it is exact for all polynomials of degree ≤ 2). However, due to the symmetry around x_1 , the formula is in fact second order accurate (i.e. it is exact for all polynomials of degree ≤ 3). Similarly, for the compact three node stencil, we gain one order of accuracy from what is expected (fourth instead of third). In the next example, we study the limiting ($\varepsilon \rightarrow 0$) RBF-FD and RBF-CFD formulas for the "scattered" node stencil,



where $\{x_1, x_2, x_3\} = \{0, 3h/2, -h\}$, and show how the accuracy is reduced.

EXAMPLE 4.4. *Approximate $u''(x_1)$ using the above stencil.*

RBF-FD formula: Interpolating to only the function values in the above stencil yields the following results for the denominators and numerators of (2.9):

$$\begin{aligned} \det(A) &= 15bh^6F(a)\varepsilon^6 + O(\varepsilon^8), \\ \det(A_1(x)) &= 5bh^4(2x - 3h)(x + h)F(a)\varepsilon^6 + O(\varepsilon^8), \\ \det(A_2(x)) &= 4bh^4x(x + h)F(a)\varepsilon^6 + O(\varepsilon^8), \\ \det(A_3(x)) &= 3bh^4x(2x - 3h)F(a)\varepsilon^6 + O(\varepsilon^8), \end{aligned}$$

where $b = 45/4$ and $F(a)$ is given by (4.1). Again, provided $F(a) \neq 0$, the cardinal RBF interpolants converge to the standard Lagrange interpolating polynomials as $\varepsilon \rightarrow 0$. The limiting RBF-FD formula is given by

$$u''(x_1) \approx \boxed{\frac{4}{5}} - \boxed{\frac{4}{3}} - \boxed{\frac{8}{15}} \frac{u}{h^2}.$$

This FD scheme is exact for all polynomials of degree ≤ 2 , leading us to conclude that it is only first order accurate.

RBF-CFD formula: We again let $m = 2$ and $\sigma = \{2, 3\}$. The determinant in the denominators of the cardinal interpolants (2.19) and (2.20) is given by

$$\det(A^H) = -32768bh^{12}F(a)\varepsilon^{20} + O(\varepsilon^{22}), \tag{4.15}$$

while the determinants in the numerators are given by

$$\det(A_1^H(x)) = -4650bh^8(2x - 3h)(x + h)(4x^2 - 2hx - 31h^2)F(a)\varepsilon^{20} + O(\varepsilon^{22}), \quad (4.16)$$

$$\det(A_2^H(x)) = 14880bh^8x(x + h)(x^2 - 2hx + 7h^2)F(a)\varepsilon^{20} + O(\varepsilon^{22}), \quad (4.17)$$

$$\det(A_3^H(x)) = 2790bh^8(2x - 3h)x(4x^2 + 2hx - 33h^2)F(a)\varepsilon^{20} + O(\varepsilon^{22}), \quad (4.18)$$

$$\det(A_4^H(x)) = -930bh^{10}(2x - 3h)x(x + h)(7x + 12h)F(a)\varepsilon^{20} + O(\varepsilon^{22}), \quad (4.19)$$

$$\det(A_5^H(x)) = -465bh^{10}(2x - 3h)x(x + h)(16x - 39h)F(a)\varepsilon^{20} + O(\varepsilon^{22}), \quad (4.20)$$

where $b = 3375/16$ and $F(a)$ is given by (4.8). Provided again $F(a) \neq 0$, we recover the standard cardinal Hermite interpolating polynomials in the limit of $\varepsilon \rightarrow 0$. The weights in the limiting RBF-CFD scheme for $u''(x_1)$ (3.3) are

$$u''(x_1) \approx \boxed{\frac{-3}{155}} \text{---} \boxed{\frac{-22}{155}} u'' + \boxed{\frac{144}{155}} \text{---} \boxed{\frac{-48}{31}} \text{---} \boxed{\frac{96}{155}} \frac{u}{h^2}.$$

This FD formula is exact for all polynomials of degree ≤ 4 , leading us to conclude that it is third order accurate. If the application requires that a three node stencil is used, then to increase the accuracy it is imperative that we use a compact stencil. \square

From the examples above, we see that for each of the RBF-FD formulations we obtain a set of requirements that the Taylor coefficients of the radial function must satisfy in order for the limiting interpolant to exist. We note that each of the infinitely smooth functions in Table 2.1 satisfies the requirements from each example. For larger stencil sizes (i.e. more nodes), the requirements that the Taylor coefficients must satisfy become more and more intricate. For the standard method in 1-D, the full set of requirements for any number of nodes was proven in [21] and extended to >1-D in [23] (in both cases, however, a slightly different formulation of the interpolant is considered). The full set of requirements for convergence in the $\varepsilon \rightarrow 0$ limit of the Hermite RBF interpolation method are still unknown. However, experiments suggest they are similar to the standard case.

The results for the RBF-CFD formulas in the above examples required quite extensive symbolic manipulation with *Mathematica*. Extending these results to more nodes is, at the present time, intractable. We can, however, numerically study the $\varepsilon \rightarrow 0$ formulas for larger stencils with the Contour-Padé algorithm [27].

4.3. Contour-Padé algorithm. The condition number of the A matrix in (2.5) grows like ε^{-P} as $\varepsilon \rightarrow 0$, where P is a positive integer that depends on the number and dimension of the nodes [21, 23]. This also appears to be true of the A^H matrix in (2.12). A clear illustration of this is given in the examples from the previous section. We see that for only 3 nodes and 2 derivative points, the small ε expansion of the determinants of A^H (4.2) and (4.9) begin with ε^{20} . This leading power also increases with the number of nodes and Hermite points. Thus, the coefficients λ_i and α_j in (2.10) must grow very rapidly as ε decreases in magnitude. Even though the coefficients grow rapidly, we typically find that the sum (2.10) is bounded. Thus, the sum must feature extreme cancellation of large quantities which makes the direct solution of the Hermite RBF interpolants via (2.10) and (2.12) ill-conditioned. This is also true for the standard RBF interpolant (2.4).

The Contour-Padé algorithm [27] largely bypasses this ill-conditioning problem by using a contour integral technique together with Padé expansions. Briefly, the central ideas of the algorithm are as follows:

- Consider RBF interpolants (2.4) and (2.10) as functions of the complex variable ε . To emphasize this dependence, we mark variables of the different interpolation methods as explicitly depending on ε , e.g. the interpolant in (2.4) is written $s(\underline{x}, \varepsilon)$, the matrix in (2.5) is written $A(\varepsilon)$ and the row vector in (2.8) is written $B(\underline{x}, \varepsilon)$. Typically, $\varepsilon = 0$ is simply a removable singularity (or, at worst, a low-order pole) of $s(\underline{x}, \varepsilon)$.
- For a fixed \underline{x} , compute $s(\underline{x}, \varepsilon)$ at equispaced ε -values around a circle centered at the origin in the complex ε -plane, where the radius is chosen large enough that the associated linear system can be solved in a stable manner.
- Take the inverse Fourier transform of the values of $s(\underline{x}, \varepsilon)$ around the circle.
- If the circle does not enclose any of the poles of $s(\underline{x}, \varepsilon)$, then the values from the inverse Fourier transform correspond to the coefficients in the Taylor expansion

$$s(\underline{x}, \varepsilon) = s_0(\underline{x}) + \varepsilon^2 s_2(\underline{x}) + \varepsilon^4 s_4(\underline{x}) + \dots .$$

We can then use this expansion to compute the interpolant for any value of ε inside the circle. If the circle encloses any poles of $s(\underline{x}, \varepsilon)$, then the Fourier coefficients correspond to the coefficients in the Laurent expansion

$$s(\underline{x}, \varepsilon) = \dots \varepsilon^{-4} s_{-4}(\underline{x}) + \varepsilon^{-2} s_{-2}(\underline{x}) + s_0(\underline{x}) + \varepsilon^2 s_2(\underline{x}) + \varepsilon^4 s_4(\underline{x}) + \dots .$$

Using Padé approximants, we convert the coefficients corresponding to the negative powers into a rational function (see [27] for more details). Combining the Taylor part of the Laurent series with the Padé rational function, we can compute $s(\underline{x}, \varepsilon)$ for any value of ε inside the circle.

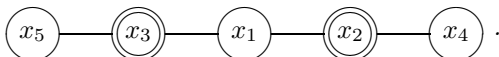
Although originally developed for the RBF interpolation problem, the above algorithm can be easily extended for computing the weights for both the standard and compact RBF-FD formulas. For example, to compute the weights for the standard RBF-FD formula (3.1), we use the algorithm on the function

$$[c(\varepsilon)|\mu(\varepsilon)] = (\mathcal{L}B(\underline{x}_1, \varepsilon)) (A(\varepsilon))^{-1} , \quad (4.21)$$

where $c(\varepsilon)$ is a row vector containing the weights, $\mu(\varepsilon)$ is a dummy value, and $B(\underline{x}, \varepsilon)$ and $A(\varepsilon)$ are given by given by (2.8) and (2.5), respectively.

4.4. Numerical results using the Contour-Padé algorithm. In the first couple of examples, we further explore the connection between limiting ($\varepsilon \rightarrow 0$) RBF-CFD schemes and the classical CFD schemes. We follow this with some results based on scattered node stencils. The results of each of the examples in this section are based on the MQ and GA radial functions.

EXAMPLE 4.5. *Approximate $u'(x_1)$ and $u''(x_1)$ using the five-node equispaced stencil:*



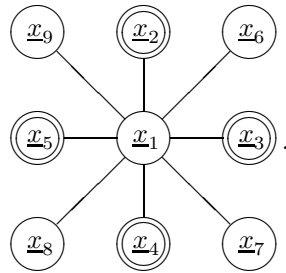
The parameters for this example are $n = 5$, $m = 2$, and $\sigma = \{2, 3\}$. Computations with the Contour-Padé algorithm suggest that the RBF-CFD formula in the $\varepsilon \rightarrow 0$ limit converges to the standard, sixth order accurate CFD formulas [25, p. 538]:

$$u'(x_1) \approx \boxed{-\frac{1}{3}} \text{---} \boxed{-\frac{1}{3}} u' + \boxed{-\frac{1}{36}} \text{---} \boxed{-\frac{7}{9}} \text{---} \boxed{0} \text{---} \boxed{\frac{7}{9}} \text{---} \boxed{\frac{1}{36}} \frac{u}{h}$$

and

$$u''(x_1) \approx \boxed{-\frac{2}{11}} \text{---} \boxed{-\frac{2}{11}} u'' + \boxed{\frac{3}{44}} \text{---} \boxed{\frac{48}{44}} \text{---} \boxed{\frac{102}{-44}} \text{---} \boxed{\frac{48}{44}} \text{---} \boxed{\frac{3}{44}} \frac{u}{h^2}. \quad \square$$

EXAMPLE 4.6. Approximate $\Delta u(\underline{x}_1)$ using the nine-node equispaced stencil:



The parameters for this example are $n = 9$, $m = 5$, and $\sigma = \{2, 3, 4, 5\}$. Computations with the Contour-Padé algorithm suggest that the RBF-CFD formula in the $\varepsilon \rightarrow 0$ limit converges to the classical, fourth order accurate CFD formula [25, p. 542]:

$$\Delta u(x_1) \approx \boxed{-\frac{1}{8}} \text{---} \boxed{-\frac{1}{8}} \Delta u + \begin{matrix} \boxed{\frac{1}{4}} & \boxed{1} & \boxed{\frac{1}{4}} \\ \boxed{1} & \boxed{-5} & \boxed{1} \\ \boxed{\frac{1}{4}} & \boxed{1} & \boxed{\frac{1}{4}} \end{matrix} \frac{u}{h^2}. \quad \square$$

The results for $u''(x_1)$ and $\Delta u(\underline{x}_1)$ (i.e the Laplacian in one and two dimensions) in the previous two examples illustrate an important observation. Had we not chosen to include the derivative information in the stencils then we would have been left with the standard fourth order FD formula

$$u'' = \boxed{-\frac{1}{12}} \text{---} \boxed{\frac{4}{3}} \text{---} \boxed{-\frac{5}{2}} \text{---} \boxed{\frac{4}{3}} \text{---} \boxed{-\frac{1}{12}} \frac{u}{h^2}$$

		m										
		0	1	2	3	4	5	6	7	8	9	10
n	1	0	0	1	1	1	2	2	2	2	3	3
	2	0	1	1	1	2	2	2	2	3	3	3
	3	1	1	1	2	2	2	2	3	3	3	3
	4	1	1	2	2	2	2	3	3	3	3	3
	5	1	2	2	2	2	3	3	3	3	3	4
	6	2	2	2	2	3	3	3	3	3	4	4
	7	2	2	2	3	3	3	3	3	4	4	4
	8	2	2	3	3	3	3	3	4	4	4	4
	9	2	3	3	3	3	3	4	4	4	4	4
	10	3	3	3	3	3	4	4	4	4	4	4
	11	3	3	3	3	4	4	4	4	4	4	5

TABLE 4.1

Numerical results for the maximum degree polynomial the limiting ($\epsilon \rightarrow 0$) RBF-CFD formulas for Δu are exact for using a scattered node stencil. Here n and m are the the number of values of u and Δu used in the formulas, respectively.

and the second order FD formula

$$\Delta u = \begin{array}{c} \boxed{\frac{-1}{18}} \\ \boxed{\frac{10}{9}} \\ \boxed{\frac{-1}{18}} \\ \boxed{\frac{10}{9}} \\ \boxed{\frac{-1}{18}} \end{array} \begin{array}{c} \boxed{\frac{10}{9}} \\ \boxed{\frac{-38}{9}} \\ \boxed{\frac{10}{9}} \\ \boxed{\frac{10}{9}} \\ \boxed{\frac{-1}{18}} \end{array} \frac{u}{h^2}.$$

Unlike their compact counterpart, these FD formulas are not diagonally dominant as we would expect from a discrete approximation of the Laplacian. In Section 6, we note that for scattered nodes it is also more likely to find a diagonally dominant RBF-FD formula when we make it compact.

We now focus on the case of a scattered node stencil (cf. Figure 3.1). As an experiment, we considered 11 scattered nodes in 2-D and computed the limiting ($\epsilon \rightarrow 0$) RBF-CFD formulas for approximating Δu using n values of u and m values of Δu such that $1 \leq n \leq 11$ and $0 \leq m \leq 10$. We then tested the resulting formulas for all 2-D polynomials of degree ≤ 6 . Table 4.1 displays the maximum degree polynomial the formulas were observed to become exact for. If we let Q equal the maximum degree polynomial, then the results from the table seem to indicate that when $\frac{1}{2}(Q+2)(Q+1) \leq n+m < \frac{1}{2}(Q+3)(Q+2)$, the limiting RBF-CFD formula is exact for all polynomials of degree Q . These are the same numbers we would expect if using regular (2-D) polynomial interpolation (with the standard basis) to generate the FD formulas.

5. Implementation details. Given a PDE on some simply connected domain Ω and an (unstructured) discretization of Ω , the goal of the RBF-FD and RBF-CFD

methods is to construct a discrete approximation to the spatial derivative operators of the PDE by approximating the operators locally on the nodes of the discretization. This discrete approximation is represented by a matrix consisting of the local approximations and is often called a *differentiation matrix*. We have described how to compute these local approximations in the previous sections with RBF-FD and RBF-CFD formulas; in this section we discuss various other implementation details of the method.

5.1. Application considerations. Typically, the application will dictate that the RBF-FD formulas satisfy some desirable properties. For example, in Section 6, we consider solving Poisson’s equation, which means $\mathcal{L} = \Delta$. Since the Laplacian is a negative definite operator, our discrete approximation to this operator using RBF-FD formulas should also be negative definite. This can be accomplished by enforcing that the weights c_j in the n -node stencil (3.1) or (3.3) satisfy

$$c_1 < 0, c_j > 0, j = 2, \dots, n, \text{ and } \sum_{j=1}^n c_j = 0, \quad (5.1)$$

or what is called the *diagonal dominance* property.

As another example, Cecil *et al.* [12] consider numerically solving the Hamilton-Jacobi equations with standard RBF-FD method. In their method, conditions are imposed to enforce the RBF-FD formulas are monotone so that convergence to the correct viscosity solution can be obtained.

5.2. Choosing the nodes. Suppose we are given an (unstructured) discretization which contains N points at which to compute all the n -node RBF-FD formulas. We denote each point as $\underline{x}_{i,1}$, $i = 1, \dots, N$, and the nodes that make up the stencil at each point as $\underline{x}_{i,j}$, $j = 1, \dots, n$ (note that $\underline{x}_{i,1}$ is always part of the stencil). Using the notation from Section 3, we let $0 \leq m < n$ be the number of nodes that contain both function and derivative information. We use the following *greedy* algorithm for selecting the nodes for each of the stencils:

1. For each $\underline{x}_{i,1}$, we first find its η nearest points, where η is on the order of n . The nodes of the stencil at $\underline{x}_{i,1}$ will be chosen from this set of points.
2. We compute all possible combinations of choosing $n - 1$ nodes from these η nearest points and sort them according to their average distance from node $\underline{x}_{i,1}$. We disregard the sets whose average distance is greater than some maximum, user defined value. These nodes will be where the function values of the stencil are given.
3. For each set of nodes from the previous step, we compute all possible combinations of choosing m nodes and again sort them by their average distance from $\underline{x}_{i,1}$. We use a similar test as the previous step to disregard some of the sets. These nodes will be where the derivative values of the function are also given.
4. Looping first over the sorted set of nodes from step 2 followed by the sorted set from step 3, we compute the RBF-FD formulas. When the direct computation becomes unstable for small ε , we use the Contour-Padé algorithm from Section 4.3. We terminate the nested loops when we have found an “acceptable” stencil. Here acceptable will depend on the application. For example, for solving Poisson’s equation, acceptable may mean the stencil satisfies (5.1). If an acceptable stencil is not found, one may choose to either

increase or decrease the size of the stencil n or the number of points that contain derivative information m . Based on the numerical results from Section 4.4, we recommend trying to keep $n + m$ constant for all the stencils.

This algorithm requires that for each $\underline{x}_{i,1}$ we know the points that are close to it. Thus, if we have a large number of nodes N in our discretization, we will need an efficient search algorithm. Liu [37, Chap. 15] describes a *binning* algorithm that is appropriate for this type of problem. Additionally, determining the stencil from the η closest points may not result in the best stencil choice, especially if there are large discrepancies in the discretization points. A possible improvement may be to compute a local Delaunay triangulation about the points surrounding $\underline{x}_{i,1}$ to determine the points natural neighbors. This idea is also mentioned in [12].

While the above procedure can be time consuming, for problems in which the discretization does not change, it can simply be done as a preprocessing step. In the case of a changing discretization, a more efficient method is necessary.

5.3. Normalization of ε for each stencil. Since there are no requirements on how the points making up the discretization of the domain are distributed, it is possible for the scales associated with each stencil to be widely different. To combat this scale problem, we may normalize the free parameter ε for each stencil. Thus far, the following two normalization methods have been proposed:

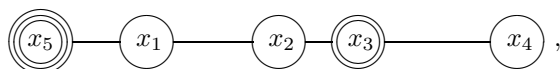
1. For each stencil, compute the maximum distance ρ_i of each of the nodes from the center node $\underline{x}_{i,1}$. Use $\varepsilon_i = \bar{\varepsilon}\rho_i$ to compute the weights, where $\bar{\varepsilon}$ is fixed for all the stencils. This method was proposed by Shu *et al.* [9] and is motivated from the finite element method.
2. Choose ε for each stencil such that the condition number of the linear system (3.2) or (3.4) for computing the RBF-FD weights is approximately the same for all the stencils. This idea is proposed by Cecil *et al.* [12] and is motivated from [29], where it is shown that the accuracy of the RBF interpolant is related to the conditioning of the associated linear system (2.5). It is obviously more computationally expensive than the previous, since we essentially have to use a numerical rootfinder on the condition number of A or A^H for each stencil.

In the numerical experiments that follow in the next section, we only explore the cases of no normalization of ε and the second method above.

5.4. Boundary conditions. In many applications, the boundary conditions of the PDE we are approximating may involve derivatives of the solution, i.e Neumann boundary conditions. When generating the RBF-CFD formulas for stencils near the boundaries where these derivative conditions are specified, it may be useful to include the derivative information in the resulting formulas. For example, consider approximating the 1-D Poisson equation

$$u'' = f \quad a \leq x \leq b, \quad u'(a) = \delta_1, \quad u(b) = \delta_2,$$

and suppose we have the following stencil near the boundary $x = a$:



where $x_5 = a$. The three concentric circles at x_5 are used to indicate that we are going to use $u(x_5)$, $u'(x_5)$, and $u''(x_5)$ to create a RBF-CFD of the form

$$u''(x_1) = \tilde{c}_5 u''(x_5) + \tilde{c}_3 u''(x_3) + d_5 u'(x_5) + c_5 u(x_5) + c_1 u(x_1) + c_2 u(x_2) + c_3 u(x_3) + c_4 u(x_4),$$

or using the computational molecule

$$\begin{array}{c}
 u''(x_1) \approx \boxed{\tilde{c}_5} \text{-----} \boxed{\tilde{c}_3} u'' + \\
 \boxed{d_5} u' + \\
 \boxed{c_5} \text{---} \boxed{c_1} \text{---} \boxed{c_2} \text{---} \boxed{c_3} \text{---} \boxed{c_4} u .
 \end{array} \tag{5.2}$$

We can use the same technique to compute these types of formulas as described in Section 3.2, however the basic form of the cardinal RBF interpolants needs to be modified. For this example, the cardinal interpolants would be of the form

$$\sum_{i=1}^5 \lambda_i \phi(\|x - x_i\|) + \sum_{j=1}^2 \alpha_j \frac{d^2}{dx^2} \phi(\|x - x_{\sigma_j}\|) - \gamma_1 \frac{d}{dx} \phi(\|x - x_5\|) + \beta ,$$

where $\sigma = \{3, 5\}$. Note that the negative sign in front of the $\frac{d}{dx}$ term appears since this term should really be $\left. \frac{d}{dy} \phi(\|x - y\|) \right|_{y=x_5}$ so that we will have a symmetric interpolation matrix (cf. Section 2.2). If we let $\bar{\psi}_5(x)$ be the cardinal interpolant that satisfies

$$\bar{\psi}_5(x_i) = 0, \quad i = 1, 2, 3, 4, \quad \bar{\psi}_5''(x_j) = 0, \quad j = 3, 5, \quad \text{and} \quad \bar{\psi}_5'(x_5) = 1,$$

then the weight d_5 in (5.2) is given by $\bar{\psi}_5'(x_1)$. The remaining weights in (5.2) are given by a similar procedure to that described in Section 3.2. The generalization of this idea to higher dimensions and stencils involving more boundary points should be straightforward.

5.5. Solving the linear systems. Depending on the PDE we are approximating, it may be necessary to invert the differentiation matrices. For example, consider approximating Poisson's equation with zero Dirichlet boundary conditions, i.e.

$$\Delta u = f \text{ in } \Omega, \quad u = 0 \text{ on } \partial\Omega ,$$

on some N node (unstructured) discretization of Ω using the RBF-CFD method. We can write the approximation to Δu in matrix form as

$$L^{\tilde{c}} \Delta \underline{u} \approx L^c \underline{u} ,$$

where \underline{u} contains the values of u at the node points, $L^{\tilde{c}}$ contains the weights of the RBF-CFD formulas for Δu , and L^c contains the weights for u . Since $\Delta u = f$, the solution to the following linear system gives an approximate solution \underline{u}^* to \underline{u} :

$$L^c \underline{u}^* = L^{\tilde{c}} \underline{f} , \tag{5.3}$$

where \underline{f} contains the values of f at the node points. Typically, L^c will be a sparse, non-symmetric matrix. Furthermore, if we impose extra conditions like (5.1) on the weights, then L^c will also be diagonally dominant. In this case classical iterative methods such as Jacobi, Gauss-Seidel and successive overrelaxation (SOR) can be applied. In the case where L^c is only assumed to be sparse and non-symmetric, several Krylov subspace methods may be applied [38, p. 321].

6. Examples: Poisson equation. In this section we apply the RBF-FD method to some model problems in order to illustrate the improved accuracy of the compact formulas. We use the MQ radial function in all the experiments because of its popularity in applications. All of the model problems involve the Laplace linear differential operator, i.e. $\mathcal{L} = \Delta$. For any radial function $\phi(r)$, the following formulas involving the d -dimensional Laplacian are extremely useful:

$$\Delta\phi = (d-1)\frac{1}{r}\frac{d\phi}{dr} + \frac{d^2\phi}{dr^2},$$

$$\Delta^2\phi = \frac{d^2 - 4d + 3}{r^2} \left[\frac{d^2\phi}{dr^2} - \frac{1}{r}\frac{d\phi}{dr} \right] + 2(d-1)\frac{1}{r}\frac{d^3\phi}{dr^3} + \frac{d^4\phi}{dr^4}.$$

Note also that $\mathcal{L}_1\phi(\|\underline{x} - \underline{y}\|) = \mathcal{L}_2\phi(\|\underline{x} - \underline{y}\|)$ for $\mathcal{L} = \Delta$.

We use the method discussed in Section 5 for selecting the stencils. In addition to enforcing the diagonal dominance property (5.1), we also enforce that

$$\frac{1}{h^2} \leq c_1 \leq \frac{2(n-1)}{h^2}, \quad h = \max_{i=1,\dots,n} \left[\min_{\substack{j=1,\dots,n \\ i \neq j}} \|\underline{x}_i - \underline{x}_j\| \right],$$

for each of the RBF-FD and RBF-CFD formulas, where \underline{x}_i , $i = 1, 2, \dots, n$ are the nodes in the stencil. This additional condition is meant to balance the weight of the approximation point with the other weights of the stencil and would be satisfied for both the standard FD and CFD formulas on an equispaced grid.

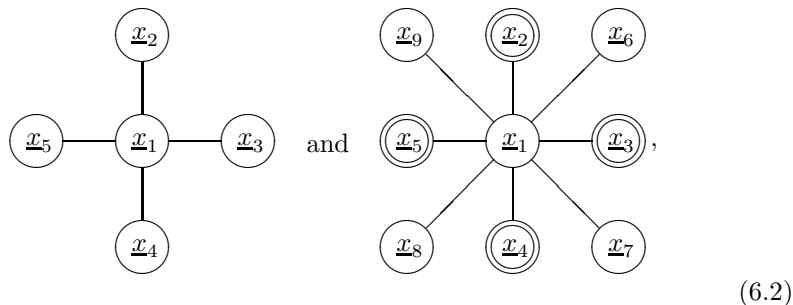
6.1. Poisson equation on the unit square. In this first example, we illustrate the rate of convergence of the RBF-FD method for the standard and compact formulas as the grid is refined. To do this, we consider the model problem

$$\Delta u = f \text{ in } \Omega = \{(x, y) \mid 0 \leq x, y \leq 1\}, \quad u = g \text{ on } \partial\Omega, \quad (6.1)$$

where f and g are computed from the known solution

$$u(\underline{x}) = u(x, y) = e^{-(x-1/4)^2 - (y-1/2)^2} \cos(2\pi y) \sin(\pi x).$$

We approximate the solution on an equispaced grid of spacing h using both an RBF-FD formula with $n = 5$ nodes and an RBF-CFD formula with $n = 9$ and $m = 5$ nodes. In this case, it is only necessary to compute the standard and compact formulas for the stencils



and apply them all over the discretization. Thus, it is also not necessary to use a node selection algorithm or any normalization of ε . From the results of Section 4.4,

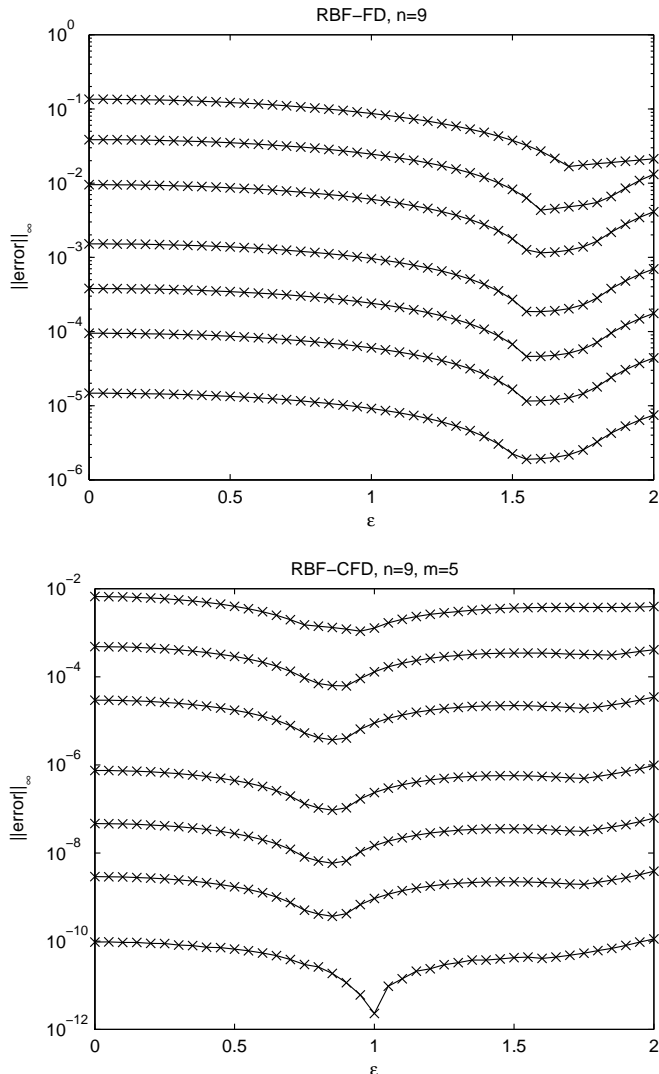


FIG. 6.1. The error as a function of ε for the solution of (6.1) using the RBF-FD and RBF-CFD formulas based on the stencils in (6.2).

we expect the RBF-FD and RBF-CFD formulas for these stencils to converge in the $\varepsilon \rightarrow 0$ limit to the classical, second order accurate and fourth order accurate FD formulas (FD2 and CFD4), respectively. We are interested to see how the $h \rightarrow 0$ convergence rate of the approximate solutions is affected by letting ε be a non-zero value in the formulas. Figure 6.1 displays the results for the max norm of error in the approximate solution based on these formulas for different values of ε and h . Moving from top to bottom in each of the plots, the error curves correspond to the solutions with grid spacing $h = 0.2, 0.1, 0.05, 0.02, 0.01, 0.005, 0.002$. We can see from the figure that for each h , the solution with the minimum error occurs at a non-zero value of ε . Furthermore, for each ε the error appears to be decreasing at a fairly constant rate as h decreases.

	h -grid spacing						
	$2.0 \cdot 10^{-1}$	$1.0 \cdot 10^{-1}$	$5.0 \cdot 10^{-2}$	$2.0 \cdot 10^{-2}$	$1.0 \cdot 10^{-2}$	$5.0 \cdot 10^{-3}$	$2.0 \cdot 10^{-3}$
$\varepsilon = 0.0$	$1.3 \cdot 10^{-1}$	$3.8 \cdot 10^{-2}$	$9.5 \cdot 10^{-3}$	$1.5 \cdot 10^{-3}$	$3.8 \cdot 10^{-4}$	$9.5 \cdot 10^{-5}$	$1.5 \cdot 10^{-5}$
rate	—	1.8	2.0	2.0	2.0	2.0	2.0
$\varepsilon = 0.25$	$1.3 \cdot 10^{-1}$	$3.8 \cdot 10^{-2}$	$9.3 \cdot 10^{-3}$	$1.5 \cdot 10^{-3}$	$3.7 \cdot 10^{-4}$	$9.2 \cdot 10^{-5}$	$1.4 \cdot 10^{-5}$
rate	—	1.8	2.0	2.0	2.0	2.0	2.0
$\varepsilon = 1.0$	$8.7 \cdot 10^{-2}$	$2.5 \cdot 10^{-2}$	$6.0 \cdot 10^{-3}$	$9.6 \cdot 10^{-4}$	$2.4 \cdot 10^{-4}$	$6.0 \cdot 10^{-5}$	$9.2 \cdot 10^{-6}$
rate	—	1.8	2.0	2.0	2.0	2.0	2.1
$\varepsilon = 1.6$	$2.7 \cdot 10^{-2}$	$4.3 \cdot 10^{-3}$	$1.1 \cdot 10^{-3}$	$1.8 \cdot 10^{-4}$	$4.6 \cdot 10^{-5}$	$1.2 \cdot 10^{-5}$	$1.9 \cdot 10^{-6}$
rate	—	2.6	1.9	2.0	2.0	2.0	2.0
$\varepsilon = 2.0$	$2.1 \cdot 10^{-2}$	$1.3 \cdot 10^{-2}$	$4.1 \cdot 10^{-3}$	$7.0 \cdot 10^{-4}$	$1.8 \cdot 10^{-4}$	$4.4 \cdot 10^{-5}$	$7.5 \cdot 10^{-6}$
rate	—	0.7	1.7	1.9	2.0	2.0	1.9

TABLE 6.1

Max norm of the error and observed convergence rate for the approximate solution of (6.1) using the $n = 5$ node RBF-FD formula.

	h -grid spacing						
	$2.0 \cdot 10^{-1}$	$1.0 \cdot 10^{-1}$	$5.0 \cdot 10^{-2}$	$2.0 \cdot 10^{-2}$	$1.0 \cdot 10^{-2}$	$5.0 \cdot 10^{-3}$	$2.0 \cdot 10^{-3}$
$\varepsilon = 0.0$	$6.6 \cdot 10^{-3}$	$4.8 \cdot 10^{-4}$	$2.9 \cdot 10^{-5}$	$7.5 \cdot 10^{-7}$	$4.7 \cdot 10^{-8}$	$2.9 \cdot 10^{-9}$	$9.5 \cdot 10^{-11}$
rate	—	3.8	4.0	4.0	4.0	4.0	3.7
$\varepsilon = 0.25$	$5.9 \cdot 10^{-3}$	$4.3 \cdot 10^{-4}$	$2.6 \cdot 10^{-5}$	$6.7 \cdot 10^{-7}$	$4.2 \cdot 10^{-8}$	$2.6 \cdot 10^{-9}$	$8.9 \cdot 10^{-11}$
rate	—	3.8	4.0	4.0	4.0	4.0	3.7
$\varepsilon = 0.85$	$1.3 \cdot 10^{-3}$	$6.3 \cdot 10^{-5}$	$3.7 \cdot 10^{-6}$	$9.4 \cdot 10^{-8}$	$5.9 \cdot 10^{-9}$	$3.7 \cdot 10^{-10}$	$1.8 \cdot 10^{-11}$
rate	—	4.4	4.1	4.0	4.0	4.0	3.3
$\varepsilon = 1.6$	$3.7 \cdot 10^{-3}$	$3.4 \cdot 10^{-4}$	$2.1 \cdot 10^{-5}$	$5.5 \cdot 10^{-7}$	$3.4 \cdot 10^{-8}$	$2.1 \cdot 10^{-9}$	$4.1 \cdot 10^{-11}$
rate	—	3.4	4.0	4.0	4.0	4.0	4.3
$\varepsilon = 2.0$	$3.9 \cdot 10^{-3}$	$4.1 \cdot 10^{-4}$	$3.5 \cdot 10^{-5}$	$9.8 \cdot 10^{-7}$	$6.2 \cdot 10^{-8}$	$3.9 \cdot 10^{-9}$	$1.1 \cdot 10^{-10}$
rate	—	3.3	3.6	3.9	4.0	4.0	3.9

TABLE 6.2

Max norm of the error and observed convergence rate for the approximate solution of (6.1) using the $n = 9, m = 5$ node RBF-CFD formula.

In Table 6.1 and 6.2 we display the max norm of the error for different values of ε and h and the observed rate of convergence for the RBF-FD and RBF-CFD formulas. We can see from the table that even for non-zero values of ε , both of the formulas demonstrate the same convergence rate as the standard FD2 and CFD4 formulas (i.e. the results for $\varepsilon = 0$). However, for larger values of ε , the results indicate that h is required to be much smaller before the standard rates of convergence are observed.

To obtain the approximate solutions to this model problem with the RBF-FD method it was necessary to solve a linear system similar to the one given in (5.3). Since we used a Cartesian grid to discretize the domain, the resulting linear system was symmetric and banded (it also happens to be negative definite). To solve this system, we used a standard banded Gaussian elimination solver. However, any iterative method that could be applied to the standard FD2 and CFD4 methods for this problem (e.g. multigrid) could also be applied RBF-FD and RBF-CFD methods.

6.2. Poisson equation on the unit disk. We consider the model problem

$$\Delta u = f \text{ in } \Omega = \{(x, y) \mid x^2 + y^2 \leq 1\}, \quad u = g \text{ on } \partial\Omega, \quad (6.3)$$

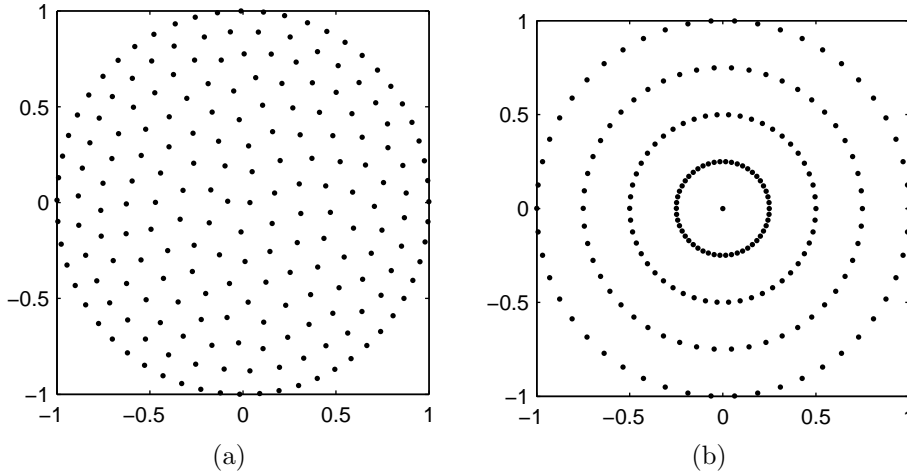


FIG. 6.2. (a) 200 point unstructured discretization of the unit disk (b) 201 point structured discretization of the unit disk.

where f and g are computed from the known solution

$$u(\underline{x}) = u(x, y) = \frac{25}{25 + (x - 0.2)^2 + 2y^2}.$$

The domain is discretized using the $N = 200$ points shown in Figure 6.2 (a). The purpose of this example is to compare the RBF-FD method for the standard and compact stencils on an unstructured grid as ε and the number of nodes in the stencil are varied.

For the first part of this example, we compute the approximate solutions without normalizing ε . Figure 6.3 contains these results using $n = 9$ scattered node RBF-FD formulas, and RBF-CFD formulas using $n = 9$, $m = 5$, and $n = 10$, $m = 9$ scattered nodes. Comparing the error for the standard $n = 9$ solution and the compact $n = 9$, $m = 5$ solution, we see that the accuracy is vastly improved by including the derivative information. Except for very small values of ε , the compact solution is at least an order of magnitude more accurate. As we should expect, the accuracy can be further improved by increasing n to 10 and m to 9. However, for this example, any improvements appear to be lost for ε approximately > 0.55 . The figure also illustrates that the optimal value of ε (the ε where the error reaches a minimum) is small (in magnitude), and nonzero, as is often the case for the RBF interpolation problem. For small values of ε that lead to an ill-conditioned linear system for computing the weights of the standard and compact formulas, we use the Contour-Padé algorithm described in Section 4.3.

One very important observation we make from this example is that no RBF-FD formula satisfying the diagonal dominance property (5.1) could be found for $n \geq 10$ and $m = 0$ using the stencil selection procedure of Section 5. However, if we choose $2 \leq m \leq 9$, such stencils could be found.

Since the domain for the problem is the unit disk, it is possible to compare the RBF-FD solutions to the standard FD solutions based on a uniform polar mesh. For the comparison, we require that the methods contain approximately the same number of boundary and interior nodes. The unstructured mesh in Figure 6.3 (a) contains 56

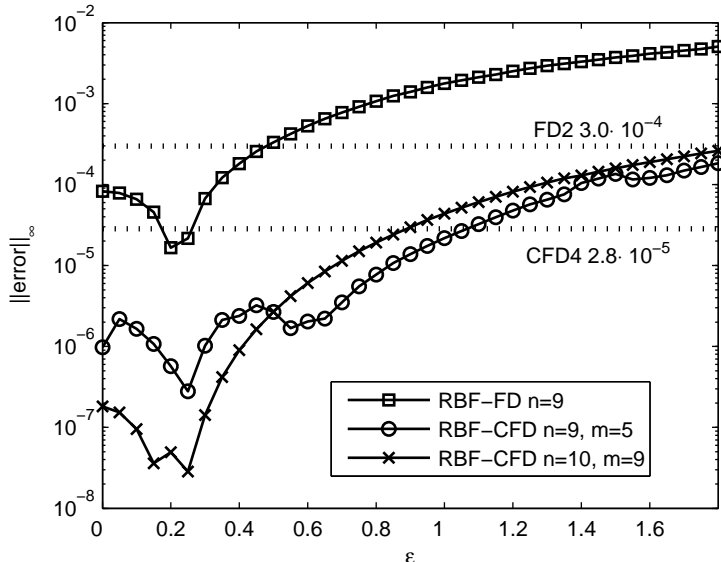


FIG. 6.3. The error as a function of a non-normalized ε for the approximate solution of (6.3) using the standard and compact RBF-FD schemes with the MQ RBF.

boundary and 144 interior nodes, while the numbers for the structured polar mesh (b) are 50 and 151, respectively. We compare the results to the standard, five node, second order FD scheme (FD2) and the standard, compact, nine node, fourth order CFD scheme (CFD4) (see, for example, [39]). These solutions are included in Figure 6.3 as dotted lines. Comparing the FD2 and standard $n = 9$ RBF-FD solution, we see that for approximately $\varepsilon < 0.6$, the RBF solution is the clear winner. In fact, at the optimal value of ε , the RBF solution is over one order of magnitude more accurate. Comparing the CFD4 solution, which uses nine function values and four derivative values, and the $n = 9$, $m = 5$ RBF-CFD solution, we see that the RBF solution is better for all values of ε approximately > 1.45 . Again, at the optimal ε , the RBF solution is over one order of magnitude more accurate.

Since we require that both the standard and compact formulas satisfy the diagonal dominance property (5.1), we use SOR for solving the linear systems associated with this example. The SOR method has a free relaxation parameter ω that can dramatically affect the number of iterations required for the solution to converge. In Figure 6.4 (a)–(c) we display how the number of iterations necessary for convergence of the different RBF-FD methods depends on ω and ε for this model problem. Using the notation of (5.3), we consider the approximate solution \underline{u}_i^* has converged when

$$\|L^c \underline{u}_i^* - L^{\tilde{c}} \underline{f}\|_{\infty} \leq 10^{-9} \left(\|L^c\|_{\infty} \|\underline{u}_i^*\|_{\infty} + \|L^{\tilde{c}} \underline{f}\|_{\infty} \right).$$

We can see from the figure that the optimal relaxation parameter depends quite significantly on the value of ε for the $n = 9$ RBF-FD solution. However, for both compact solutions, the optimal ω appears to be rather stable with respect to ε . The choice of $\omega \approx 1.6$ seems to be good for both compact methods. Figure 6.4 (d) shows the number of iterations necessary for convergence at the optimal value of ω for different values of ε . We can see from the figure that the minimum number of iterations is quite

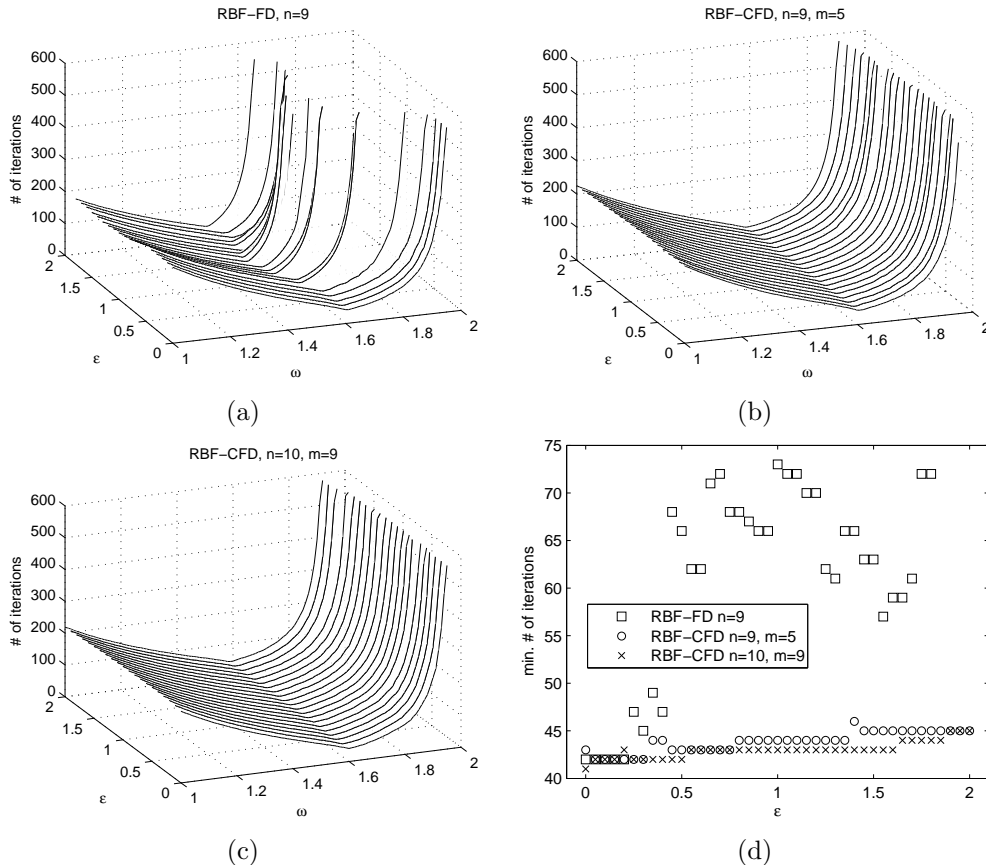


FIG. 6.4. (a)–(c) Convergence results of the SOR method for solving the linear systems associated with the Poisson model problem as a function of the shape parameter ϵ and SOR parameter ω . (d) Minimum number of iterations required for different values of ϵ .

consistent for the compact methods but jumps around for the non-compact method. There also appears to be a slight increase in the number of iterations as ϵ increases. Efficient direct solvers based on the FFT may be used for the standard FD2 and CFD4 methods applied to this model problem [40]. However, we note that if we had used SOR then the minimum number of iterations would be 162 for FD2 and 328 for CFD4, which are significantly higher than the standard and compact RBF-FD methods.

For the second part of this example, we compute the approximate solutions using the condition number normalization described in Section 5.3. When the stencils are such that the associated condition number of the linear system is $\geq 10^{10}$ we use the Contour-Padé algorithm to compute the weights of the stencils. Figure 6.5 contains these results. We see similar results to the non-normalized case. This is most likely due to the uniform (unstructured) discretization we have used for approximating the model problem. For a discretization with less uniformity, we expect the results to be different. Although data is not included here, we note that the results of applying SOR for solving the linear systems were similar to the non-normalized case.

From the numerical results for limiting ($\epsilon \rightarrow 0$) scattered node formulas in Table 4.1 we expect that the $n = 9$ node RBF-FD solution would be first order accurate,

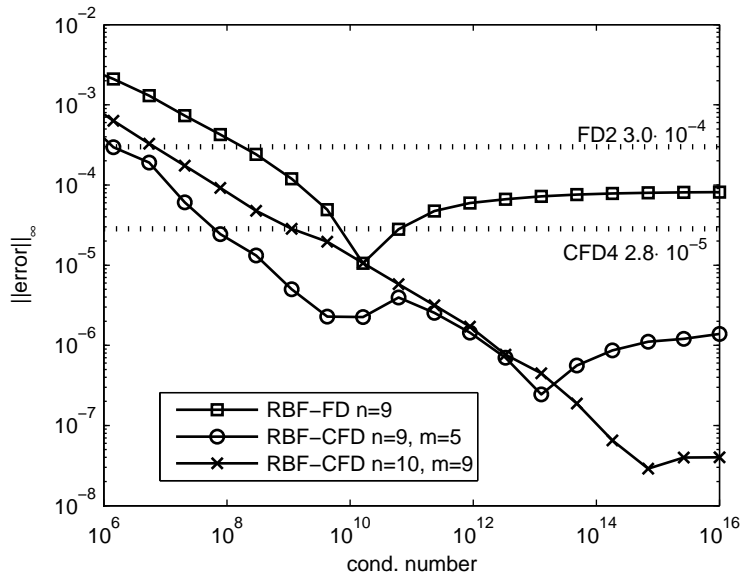


FIG. 6.5. The error as a function of the normalized condition number for the approximate solution of (6.3) using the standard and compact RBF-FD schemes with the MQ RBF. We use the Contour-Padé algorithm to bypass any numerical ill-conditioning problems with computing the weights for small ε .

the $n = 9$, $m = 5$ node RBF-CFD solution would be second order accurate, and the $n = 10$, $m = 9$ node RBF-CFD solution would be third order accurate.

7. Conclusions. In a similar style to how polynomials are used to generate FD and CFD stencils for 1-D, we have here shown how RBFs can be used to create analogous formulas also for multidimensional scattered node layouts. We have also demonstrated that, when the stencil nodes are arranged accordingly, RBF-FD and RBF-CFD formulas in the $\varepsilon \rightarrow 0$ limit are equivalent to standard FD and CFD formulas. In contrast to many methods, such as finite elements, the RBF-FD or RBF-CFD methods do not require the generation global meshes. Furthermore, the number of space dimensions and the geometric complexity of the methods can all be arbitrary without adversely affecting either computational speed or algorithmic complexity. Tests with Poisson's equation show that accuracy can be improved dramatically by using RBF-CFD formulas, and that it is imperative to use these formulas to preserve diagonal dominance. The latter property permits the use of fast iterative methods for computing the numerical solution. We believe—but it is yet to be fully explored—that the RBF-CFD approach will prove successful also for many further classes of PDEs.

REFERENCES

- [1] E. J. Kansa. Multiquadrics – a scattered data approximation scheme with applications to computational fluid-dynamics – II: Solutions to parabolic, hyperbolic and elliptic partial differential equations. *Comput. Math. Appl.*, 19:147–161, 1990.
- [2] Y. C. Hon and X. Z. Mao. An efficient numerical scheme for Burgers' equation. *Appl. Math. Comput.*, 95:37–50, 1998.
- [3] C. Franke and R. Schaback. Solving partial differential equations by collocation using radial basis functions. *Appl. Math. Comput.*, 93:73–82, 1998.

- [4] G. E. Fasshauer. Solving partial differential equations with radial basis functions: multilevel methods and smoothing. *Adv. Comput. Math.*, 11:139–159, 1999.
- [5] A. I. Fedoseyev, M. J. Friedman, and E. J. Kansa. Improved multiquadric method for elliptic partial differential equations via PDE collocation on the boundary. *Comput. Math. Appl.*, 43:439–455, 2001.
- [6] E. Larsson and B. Fornberg. A numerical study of some radial basis function based solution methods for elliptic PDEs. *Comput. Math. Appl.*, 46:891–902, 2003.
- [7] E. J. Kansa and Y. C. Hon. Circumventing the ill-conditioning problem with multiquadric radial basis functions: Applications to elliptic partial differential equations. *Comput. Math. Appl.*, 39:123–137, 2000.
- [8] L. Ling and E. J. Kansa. A least-squares preconditioner for radial basis functions collocation methods. *Adv. Comput. Math.*, 2004. In press.
- [9] C. Shu, H. Ding, and K. S. Yeo. Local radial basis function-based differential quadrature method and its application to solve two-dimensional incompressible Navier-Stokes equations. *Comput. Methods Appl. Mech. Engrg.*, 192:941–954, 2003.
- [10] C. Shu and H. Ding. Numerical comparison of least square-based finite difference (LSFD) and local multiquadrature-differential quadrature (LMQDQ) methods. *Comput. Math. Appl.*, submitted (2004).
- [11] A. I. Tolstykh, M. V. Lipavskii, and D. A. Shirobokov. High-accuracy discretization methods for solid mechanics. *Arch. Mech.*, 55:531–553, 2003.
- [12] T. Cecil, J. Qian, and S. Osher. Numerical methods for high dimensional Hamilton-Jacobi equations using radial basis functions. *J. Comput. Phys.*, 196:327–347, 2004.
- [13] G. Wright. *Radial Basis Function Interpolation: Numerical and Analytical Developments*. PhD thesis, University of Colorado, Boulder, 2003.
- [14] B. Fornberg. *A Practical Guide to Pseudospectral Methods*. Cambridge University Press, Cambridge, 1996.
- [15] B. Fornberg. Calculation of weights in finite difference formulas. *SIAM Rev.*, 40(3):685–691, 1998.
- [16] R. Abgrall. On essentially non-oscillatory schemes on unstructured meshes: analysis and implementation. *J. Comput. Phys.*, 114:45–58, 1994.
- [17] W. Schönauer and T. Adolph. How we solve PDEs. *J. Comput. Appl. Math.*, 131:473–492, 2001.
- [18] R. J. Y. McLeod and M. L. Baart. *Geometry and Interpolation of Curves and Surfaces*. Cambridge University Press, Cambridge, 1998.
- [19] W. R. Madych. Miscellaneous error bounds for multiquadric and related interpolants. *Comput. Math. Appl.*, 24:121–138, 1992.
- [20] B. Fornberg and N. Flyer. Accuracy of radial basis function derivative approximations in 1-d. *Adv. Comput. Math.*, To appear (2004).
- [21] T. A. Driscoll and B. Fornberg. Interpolation in the limit of increasingly flat radial basis functions. *Comput. Math. Appl.*, 43:413–422, 2002.
- [22] B. Fornberg, G. Wright, and E. Larsson. Some observations regarding interpolants in the limit of flat radial basis functions. *Comput. Math. Appl.*, 47:37–55, 2002.
- [23] E. Larsson and B. Fornberg. Theoretical and computational aspects of multivariate interpolation with increasingly flat radial basis functions. *Comput. Math. Appl.*, 2004. to appear.
- [24] R. Schaback. Multivariate interpolation by polynomials and radial basis functions. *Constr. Approx.*, submitted (2004).
- [25] L. Collatz. *The Numerical Treatment of Differential Equations*. Springer Verlag, Berlin, 1960.
- [26] S. K. Lele. Compact finite difference schemes with spectral-like resolution. *J. Comput. Phys.*, 103:16–42, 1992.
- [27] B. Fornberg and G. Wright. Stable computation of multiquadric interpolants for all values of the shape parameter. *Comput. Math. Appl.*, 48:853–867, 2004.
- [28] C. A. Micchelli. Interpolation of scattered data: distance matrices and conditionally positive definite functions. *Constr. Approx.*, 2:11–22, 1986.
- [29] R. Schaback. Error estimates and condition numbers for radial basis function interpolants. *Adv. Comput. Math.*, 3:251–264, 1995.
- [30] J. Yoon. Spectral approximation orders of radial basis function interpolation on the Sobolev space. *SIAM J. Math. Anal.*, 23(4):946–958, 2001.
- [31] R. L. Hardy. Multiquadric equations of topography and other irregular surfaces. *J. Geophys. Res.*, 76:1905–1915, 1971.
- [32] R. L. Hardy. Theory and applications of the multiquadric-biharmonic method: 20 years of discovery. *Comput. Math. Appl.*, 19:163–208, 1990.
- [33] G. E. Fasshauer. Hermite interpolation with radial basis functions on spheres. *Adv. Comput.*

- Math.*, 10:81–96, 1999.
- [34] X. Sun. Scattered Hermite interpolation using radial basis functions. *Linear Algebra Appl.*, 207:135–146, 1994.
 - [35] Z. Wu. Hermite-Birkhoff interpolation of scattered data by radial basis functions. *Approx. Theory Appl.*, 8(2):1–10, 1992.
 - [36] B. Fornberg, E. Larsson, and G. Wright. A new class of oscillatory radial basis functions. *Comput. Math. Appl.*, submitted (2004).
 - [37] G. R. Liu. *Mesh Free Methods: moving beyond the finite element method*. CRC Press, New York, 2002.
 - [38] J. W. Demmel. *Numerical Linear Algebra*. Society for Industrial and Applied Mathematicians, Philadelphia, 1997.
 - [39] R. C. Mittal and S. Gahlaut. High-order finite-difference schemes to solve Poisson’s equation in polar coordinates. *IMA J. Num. Anal.*, 11:261–270, 1991.
 - [40] M.-C. Lai and W.-C. Wang. Fast direct solvers for Poisson equation on 2D polar and spherical geometries. *Numer. Methods for Partial Differential Equations*, 18:56–68, 2002.