Radial Basis Function methods - reduced computational expense by exploiting symmetry

Scott A. Sarra
Marshall University

March 28, 2018

Abstract

Radial basis function (RBF) methods are popular methods for scattered data interpolation and for solving PDEs in complexly shaped domains. RBF methods are simple to implement as they only require elementary linear algebra operations. In this work center locations that result in matrices with a centrosymmetric structure are examined. The resulting matrix structure can be exploited to reduce computational expense and improve the accuracy of the methods while avoiding more complicated techniques such as domain decomposition.

1 Introduction

Radial basis function (RBF) methods are popular methods for scattered data interpolation and for solving PDEs in complexly shaped domains. Contributing factors to the popularity of RBF methods are their simplicity, ease of use, and flexibility. The methods are conceptually simple as the core operations in their implementation are solving linear systems of equations and matrix-vector multiplication. The extreme flexibility of RBF methods is due to the fact that there is complete freedom as to where centers may be located.

In many complexly shaped domains, this freedom can be taken advantage of to reduce the flop count of common linear algebra operations by a factor of two to four, to reduce storage requirements by a factor of two, and to increase the accuracy of the methods. Placing scattered centers symmetrically about a line that divides a domain in half results in RBF matrices that have a centrosymmetric structure. All the algorithms and several examples from this manuscript are implemented in the freely available Matlab RBF toolbox (MRBFT) [19, 22].

The centrosymmetric structure that is sought in RBF methods in this manuscript is present in differentiation matrices of the polynomial based Chebyshev pseudospectral method [3, 25]. The structure has been exploited in the Chebyshev pseudospectral method in order to halve the storage requirements and flop counts for matrix-vector multiplication. Pseudospectral methods are tied to a fixed structured grid in one dimension and to tensor products of that grid in

higher dimensions. RBFs methods can realize matrices with a centrosymmetric structure with far less restriction on center locations.

Previously in reference [14] the authors located centers on concentric circles which resulted in RBF matrices with a block circulant structure which allowed the Fast Fourier Transform to be efficiently used in the execution of the method. The method is limited in that it does not apply to scattered data or to complexly shaped domains. Subsequently [15], conformal mapping was used to map several complexly shaped domains to circles in order to apply the block circulant algorithm. Seeking a centrosymmetric rather than circulant structure allows the RBF method to work with scattered centers in complexly domains that have symmetry.

2 Radial Basis Function methods

RBF interpolation uses a set of N distinct points $X = \{x_1^c, \ldots, x_N^c\}$ in $\mathcal{R}^d$ called centers. No restrictions are placed on the shape of problem domains or on the location of the centers. In this work global, infinitely differentiable, RBFs ϕ that contain a free parameter $\varepsilon > 0$ called the shape parameter are employed. The inverse quadratic (IQ)

$$\phi(r) = \frac{1}{1 + \varepsilon^2 r^2} \tag{1}$$

and the Gaussian (GA)

$$\phi(r) = e^{-\varepsilon^2 r^2} \tag{2}$$

are representative of this type of RBF. The variable r in the definition of the IQ and GA is $r = \|x - x^c\|_2$ where $x, x^c \in \mathcal{R}^d$. A formal, broader definition of RBFs can be found in reference [28, p. 78] that includes functions with compact support, less smoothness, and which do not contain a shape parameter. Infinitely differentiable RBFs that contain a shape parameter are attractive because they potentially [28, p. 183] interpolate with exponential accuracy.

The RBF interpolant assumes the form

$$\mathcal{I}_N f(x) = \sum_{k=1}^{N} a_k \phi\left(\|x - x_k^c\|_2\right) \tag{3}$$

where $a = (a_1, a_2, \ldots, a_N)^T$ is a vector of expansion coefficients. The expansion coefficients are determined by enforcing the interpolation conditions

$$\mathcal{I}_N f(x_j^c) = f(x_j^c), \qquad j = 1, 2, \ldots, N \tag{4}$$

which result in a $N \times N$ linear system

$$Ba = f. \tag{5}$$

The matrix B with entries

$$b_{jk} = \phi(\left\|x_j^c - x_k^c\right\|_2), \qquad j, k = 1, \ldots, N \tag{6}$$

is called a system matrix. The evaluation of the interpolant (3) at M points x_j is accomplished by multiplying the expansion coefficients by the $M \times N$ evaluation matrix H that has entries entries

$$h_{jk} = \phi(\|x_j - x_k^c\|_2), \qquad j = 1, \ldots, M \text{ and } k = 1, \ldots, N. \tag{7}$$

For later reference, the signed distance matrices are defined, for example in two space dimensions, to have elements

$$(r_x)_{jk} = x_j^c - x_k^c \text{ and } (r_y)_{jk} = y_j^c - y_k^c, \qquad j, k = 1, \ldots, N \tag{8}$$

and the distance matrix has elements

$$r_{jk} = \sqrt{(r_x)_{jk}^2 + (r_y)_{jk}^2}, \qquad j, k = 1, \ldots, N.$$

By linearity, the RBF interpolant can be differentiated as

$$\mathcal{D}\left(\mathcal{I}_N f(x)\right) = \sum_{k=1}^{N} a_k \mathcal{D}\phi\left(\|x - x_k^c\|_2\right) \tag{9}$$

where $\mathcal{D}$ is a linear differential operator. The operator $\mathcal{D}$ may be a single derivative or a linear differential operator such as the Laplacian. Evaluating (9) at the centers X can be accomplished by multiplying the expansion coefficients by the evaluation matrix $H_{\mathcal{D}}$ with entries

$$h_{jk} = \mathcal{D}\phi(\left\|x_j^c - x_k^c\right\|_2), \qquad j, k = 1, \ldots, N. \tag{10}$$

That is, $\mathcal{D}f \approx H_{\mathcal{D}} a$. Alternatively, derivatives can be approximated by multiplying the grid function values $\{f(x_k^c)\}_{k=1}^N$ by the differentiation matrix $D = H_{\mathcal{D}} B^{-1}$ since

$$\mathcal{D}f \approx H_{\mathcal{D}} a = H_{\mathcal{D}}(B^{-1} f) = (H_{\mathcal{D}} B^{-1}) f. \tag{11}$$

Both equations (5) for the expansion coefficients and (11) for the differentiation matrix assume that the system matrix is invertible. The system matrix must be invertible in order to guarantee the existence of a unique solution. The IQ and GA system matrices are symmetric positive definite (SPD) and thus invertible [8]. While invertible, the system matrix is typically very poorly conditioned. The eigenvalues of B satisfy $0 < \lambda_{min} = \lambda_1 \leq \lambda_2 \leq \cdots \leq \lambda_N = \lambda_{max}$ and the matrix condition number in the 2-norm is $\kappa_2(B) = \lambda_{max}/\lambda_{min}$. Additional information on the eigenvalues of RBF system matrices can be found in reference [26]. For a fixed set of centers, the shape parameter affects both the accuracy of the method and the conditioning of the system matrix. The RBF method is most accurate for smaller values of the shape parameter where the system matrix is ill-conditioned [23]. The attainable error and the condition number of the system matrix cannot both be kept small. This relationship has been dubbed the uncertainty principle [24]. We stress that the content of this manuscript applies to the RBF method with standard basis functions and not to alternative basis RBF methods as described in [7] and the references within.

Recent monographs [4, 8, 23, 28] on RBF methods can be consulted for more information.

3 Structured Matrices

This section summarizes pertinent definitions and theorems concerning structured matrices. The definitions and theorems can be found in various forms in multiple references including [1, 2, 5, 6, 10, 16, 27].

The ***contra-identity matrix*** J is a square matrix whose elements are all zero except those on its southwest-northeast diagonal which are all $1's$. J is a permutation matrix. Multiplying a matrix B by J from the left results in reversing the rows of B and multiplying B by J from the right results in reversing the columns of B.

Definition 1 *Let x be a $N \times 1$ vector.*

1. *A vector x is* **symmetric** *if $Jx = x$. The elements of a symmetric vector satisfy $x_i = x_{N-i+1}$ for $i = 1, \ldots, N$. Such a vector is also said to be* **even**.

2. *A vector is* **skew-symmetric** *if $Jx = -x$ and its elements satisfy $x_i = -x_{N-i+1}$. Such a vector is also said to be* **odd**.

Definition 2 *Let B be a $N \times N$ matrix.*

1. *B is* **centrosymmetric** *(**centro** for short) if $B = JBJ$. The elements of a centrosymmetric matrix satisfy*

$$b_{i,j} = b_{N-i+1,N-j+1}, \qquad 1 \leq i, j \leq N. \tag{12}$$

2. *B is* **skew-centrosymmetric** *if $B = -JBJ$. The elements of a skew-centrosymmetric satisfy*

$$b_{i,j} = -b_{N-i+1,N-j+1}. \tag{13}$$

Throughout N is even and $P = N/2$. All results hold for odd N but the notation is more cumbersome and only even N are considered for clarity and space considerations. Throughout this section, B is an arbitrary centrosymmetric matrix as defined in Definition 2. In later sections, B is taken to be the RBF system matrix which is by definition symmetric and can also be centrosymmetric if the underlying center distribution that the matrix in based on is distributed in a certain way.

Theorem 1 *[5] A $N \times N$ centrosymmetric matrix B can be written as*

$$B = \begin{bmatrix} B_{11} & JB_{21}J \\ B_{21} & JB_{11}J \end{bmatrix} \tag{14}$$

where B_{11}, B_{21}, and J are $P \times P$. Additionally if B is symmetric, then $B_{11} = B_{11}^T$ (B_{11} is symmetric) and $JB_{21}J = B_{21}$.

Theorem 2 *[5] The matrix (14) and*

$$D = \begin{bmatrix} L = B_{11} - JB_{21} & 0 \\ 0 & M = B_{11} + JB_{21} \end{bmatrix} \tag{15}$$

are orthogonally similar.

Proof. *The matrix*

$$Q = \frac{1}{\sqrt{2}} \begin{bmatrix} I & -J \\ I & J \end{bmatrix} \tag{16}$$

is easily verified to be orthogonal and multiplication gives that $QBQ^T = D$ □

In Theorem 2, L and M could have been defined as $L = B_{11} - JB_{21} = B_{11} - J(-JB_{12}J) = B_{11} + B_{12}J$ and $M = B_{11} + JB_{21} = B_{11} + J(-JB_{12}J) = B_{11} - B_{12}J$. The two definitions allow the option of using either the upper half or the left half of the matrix in the algorithms that follow. The implementation of the algorithms in the MRBFT [22, 19] uses the left half of the matrices.

Theorem 3 *[5] A* $N \times N$ *skew-centrosymmetric matrix* B *can be written as*

$$B = \begin{bmatrix} B_{11} & -JB_{21}J \\ B_{21} & -JB_{11}J \end{bmatrix} \tag{17}$$

where B_{11}, B_{21}, *and* J *are* $P \times P$. *Additionally if* B *is skew-symmetric, then* $B_{11} = -B_{11}^T$ *(*B_{11} *is skew-symmetric) and* $JB_{21}J = B_{21}^T$.

4 Center locations for centrosymmetry

RBF system matrices and RBF differentiation matrices are (skew) centrosymmetric if the signed distance matrix (8) is skew-centrosymmetric for odd order differential operators and either skew-centrosymmetric or centrosymmetric for even order differential operators. The condition does not depend on the location of centers but rather on the distance between centers. Any center distribution in one dimension that is symmetrical about its mid-point causes the signed distance matrix to be skew-centrosymmetric. Such center distributions include uniformly spaced centers and the Chebyshev-Gauss-Lobatto (CGL) points $x_k = \cos(k\pi/(N-1)), \quad k = 0, 1, \ldots, N-1$ that cluster centers densely around the endpoints.

This section examines center distributions that result in RBF system and differentiation matrices that have either a centrosymmetric or skew-centrosymmetric structure. For brevity, both types of distributions are called centro center distributions throughout. Interpolation is considered an order zero (even order) derivative. For example, consider the following three center distributions within symmetric domains that are possible in two space dimensions:

origin The domain is discretized with centers (x, y) on one side of the line $y = x$ and the remaining portion of the domain is covered with centers $(-x, -y)$. For example $X_a = \{(x_1, y_1), (x_2, y_2), (-x_2, -y_2), (-x_1, -y_1)\}$. On such a center distribution, all system matrices (interpolation) will be centrosymmetric, all even order derivative matrices will be centrosymmetric, and all odd order differentiation matrices will be skew-centrosymmetric.

x-axis The domain is discretized with centers (x, y) on one side of the line $y = 0$ and the remaining portion of the domain is covered with centers $(x, -y)$. For example $X_b = \{(x_1, y_1), (x_2, y_2), (x_2, -y_2), (x_1, -y_1)\}$. On such a center distribution, system matrices will be centrosymmetric, even order differentiation matrices will be centrosymmetric, odd order differentiation matrices with respect to x will be centrosymmetric, and odd order differentiation matrices with respect to y will be skew-centrosymmetric. The discretization of the divergence operator $\mathcal{G} = \partial/\partial x + \partial/\partial y$ will not have either type of symmetry as the sum of a centrosymmetric and skew-centrosymmetric matrix is in general not centrosymmetric or skew-centrosymmetric.

y-axis The domain is discretized with centers (x, y) on one side of the line $x = 0$ and the remaining portion of the domain is covered with centers $(-x, y)$. For example $X_c = \{(x_1, y_1), (x_2, y_2), (-x_2, y_2), (-x_1, y_1)\}$. On such a center distribution, system matrices will be centrosymmetric, even order differentiation matrices will be centrosymmetric, odd order differentiation matrices with respect to x will be skew-centrosymmetric, and odd order differentiation matrices with respect to y will be centrosymmetric. The divergence will not have either type of symmetry.

The script ***isCentroTest.m*** tests a collection of differential operators on various center distributions for centrosymmetry and ***centroExtensions.m*** approximates various differential operators on each type of centrosymmetrically extended center distribution.

Figure 1 gives four examples of centrosymmetric center distributions. The centers in the circular domain in the upper left image were produced by placing quasi-random Hammersley [17] points (x, y) on the circle which are clustered near the boundary and then extended about the line $y = x$ via $(-x, -y)$. The centers in the dumbell shaped domain in the upper right image were located by covering the right half of the domain with (x, y) and then the left half with $(-x, y)$. Centrosymmetric center distributions could also be obtained in this domain by extending about the x-axis or the origin. For the region with a hole in the lower left image which is symmetric with respect to the origin, the half of the domain above the line $y = x$ is covered with centers and then the remaining part with centers located at $(-x, -y)$. In order to exploit symmetry, it is not necessary that the original domain have one of the three types of symmetry. Symmetry can also be exploited if the domains can be transformed to have symmetry via a linear transformation or a rotation as is the case in the next domain. The domain in the lower right image of Figure 1 is symmetric

with respect to the x-axis after it is rotated 0.25 radians. The domain uses two different center densities that feature Hammersley points which are located more densely near the boundary than in the interior. A different rotation of the domain would make it possible to exploit symmetry with respect to the origin. The script ***complexCentroCenters.m*** produces the center distribution. The script ***interp3dCentro.m*** produces a centro center distribution in 3d and evaluates an interpolation problem on the surface of a sphere.

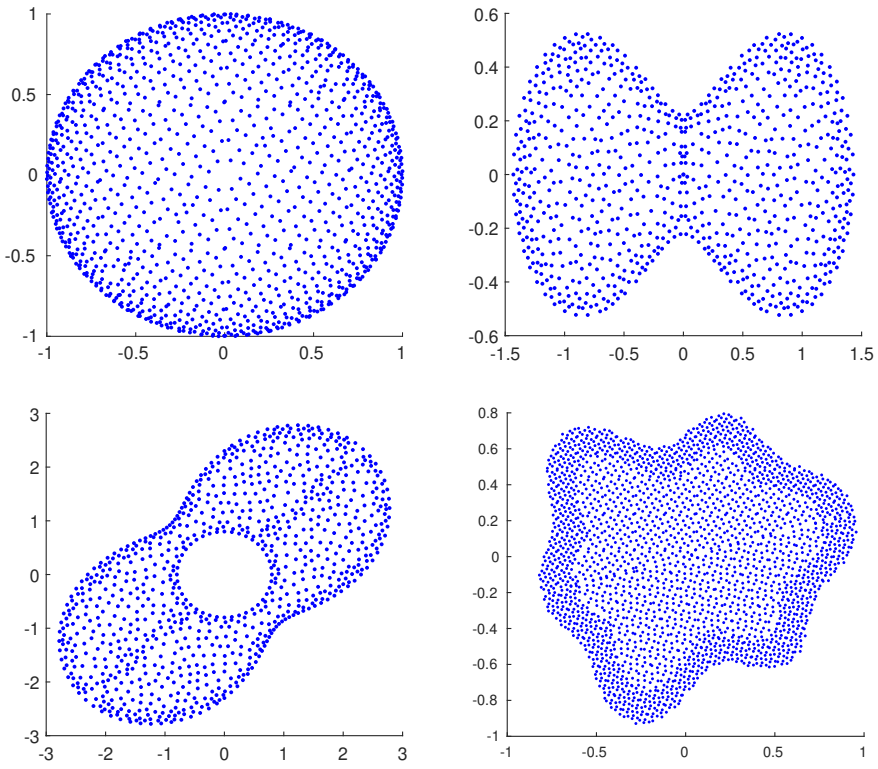

Figure 1: Center distributions that result in centrosymmetric system matrices. Upper left: 1575 clustered Hammersly point on the unit circle. Upper right: 852 scattered centers on a dumbell shaped domain. Lower left: 920 scattered centers. Lower right: 2542 scattered centers.

5 Centrosymmetric algorithms

The next several subsections detail linear algebra algorithms for centrosymmetric matrices. Flop count savings and accuracy gains are realizable compared

to standard algorithms. The algorithms include: the solution of centrosymmetric linear systems which are applicable to RBF interpolation and derivative approximation, differentiation matrix formulation, RBF system matrix condition number calculation, and centrosymmetric matrix-vector multiplication. All errors are measured in the infinity norm.

5.1 Linear systems

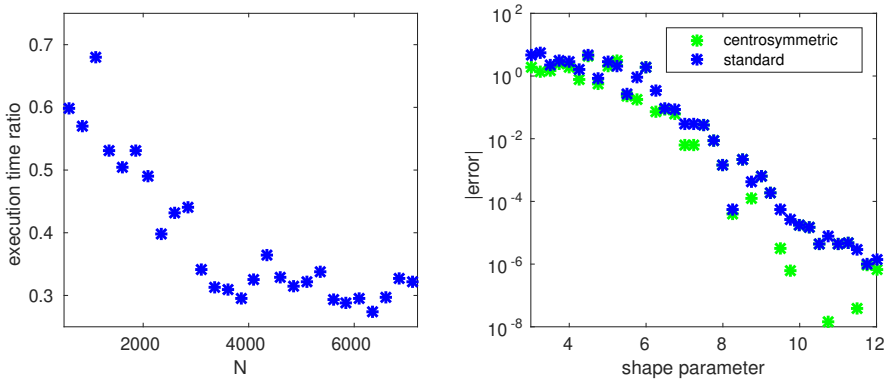

Figure 2: Left: Ratio of the centrosymmetric linear system solving algorithm execution time to the standard algorithm execution time. For $N > 350$ the centrosymmetric algorithm is faster. For larger N, the centrosymmetric algorithm is approaching the theoretical limit of four times faster. Right: accuracy of the centrosymmetric and standard linear system solver for a linear system involving the RBF system matrix over a range of the shape parameter.

This section summarizes an algorithm for solving a centrosymmetric linear system that is described in [1]. Let B be centrosymmetric, $x = [x_1\ x_2]^T$, and $f = [f_1\ f_2]^T$ where x_1, x_2, f_1, f_2 are $P \times 1$ vectors. As a result of Theorem 2, solving the linear system $Bx = f$ is equivalent to solving $QDQ^T x = f$ or $DQx = Qf$. Writing out the last equation gives

$$\begin{bmatrix} L & 0 \\ 0 & M \end{bmatrix} \begin{bmatrix} x_1 - Jx_2 \\ x_1 + Jx_2 \end{bmatrix} = \begin{bmatrix} f_1 - Jf_2 \\ f_1 + Jf_2 \end{bmatrix} \tag{18}$$

Equation (18) represents two uncoupled half-sized $P \times P$ systems with solutions $t_1 = x_1 - Jx_2$ and $t_2 = x_1 + Jx_2$. The solution of the original system is recovered as $x_1 = \frac{1}{2}(t_1 + t_2)$ and $x_2 = \frac{1}{2}J(t_2 - t_1)$.

Since B and D are similar, the eigenvalues of L and M must also be eigenvalues of B. In addition to being symmetric, both L and M are positive definite and have a Cholesky factorization. It must be that $\kappa(L) \leq \kappa(B)$ and $\kappa(M) \leq \kappa(B)$. Typically some improvement in the conditioning of the smaller matrices is realized, but the smaller matrices are still typically poorly conditioned. If the

matrices are too poorly ill-conditioned it may be that the matrices cease to be numerically SPD and a Cholesky factorization will fail as it encounters the square root of a negative number. Reference [21] describes how the method of diagonal increments (MDI) can be effectively and efficiently used to regularize a SPD system. MDI modifies the diagonal of a SPD matrix as $B = B + \mu I$ where μ is a regularization parameter with a typical value being $\mu = 5e\text{-}15$. MDI can be used to regularize the two smaller systems and insure that the matrices L and M are numerically SPD.

The dominant term in flop count

$$2 \cdot \frac{1}{3}\left(\frac{N}{2}\right)^3 = \frac{1}{12}N^3,$$

of the algorithm for solving a linear system is from the two half-sized Cholesky factorizations. When compared to the solution of the larger $N \times N$ system by a Cholesky factorization, the centrosymmetric algorithm is asymptotically more efficient by a factor of 4 concerning the flop count and by a factor of two in storage requirements. The left image of Figure 2 displays the efficiency factor as a function of N. The efficiency factor measures the ratio of the execution time of the centrosymmetric algorithm to the standard algorithm. For small $N < 100$ the set up costs of the centrosymmetric algorithm cause the standard algorithm to be more efficient, but the efficiency factor of the centrosymmetric algorithm increases with N. The script ***systemSolveBench.m*** from the MRBFT produces the image. The right image of Figure 2 compares the accuracy of the two algorithms. In this example, the centrosymmetric algorithm is more accurate than the standard algorithm as soon as the full sized system matrix becomes moderately ill-conditioned. This is illustrated in the right image of Figure 2. With $N = 44$ centers in a fixed location the condition number of the system matrix increases as the shape parameter is reduced. The script ***centroSolveAccuracy.m*** produces the image.

5.2 Derivative matrix formation

The standard way to form a RBF differentiation matrix (DM) is to use a $\frac{1}{3}N^3$ Cholesky factorization followed by a forward and a back substitution for each column ($N \times 2N^2$ flops) for a total of $\frac{7}{3}N^3 = \frac{28}{12}N^3$ in the dominant term of the flop count.

The algorithm from section 5.1 can be used to form RBF differentiation matrices using two half-sized Cholesky factorizations, $\frac{1}{12}N^3$, and 2 forward and back substitutions for each half-sized column, $2\frac{N}{2} \times 2\left(\frac{N}{2}\right)^2 = \frac{6}{12}N^3$, for a total $\frac{7}{12}N^3$ in the dominant term of the flop count. A savings of a factor of four in the flop count and a factor of two savings in storage can be realized. Figure 3 plots the efficiency factor of the centrosymmetric algorithms over the standard algorithm as a function of N. The script ***dmFormBench.m*** produces the plot. Note the outlier at $N \approx 5500$ in the plot. If the script were to be run again, outliers may appear for other N but the overall asymptotic trend will always

be apparent. Additionally, the centro algorithm preserves the theoretical centro structure that the matrix should have whereas the standard algorithm does not. The consequences of this aspect of the algorithm is commented on in section 7.

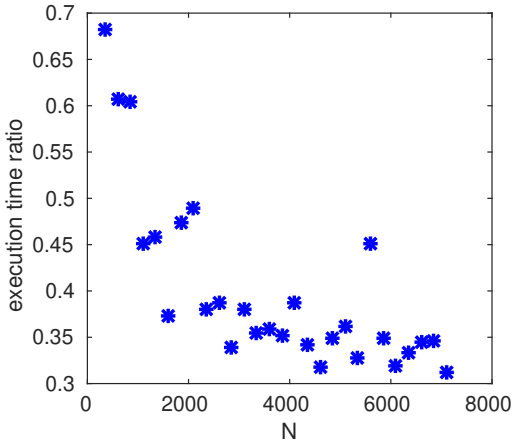

Figure 3: Ratio of centrosymmetric matrix DM formation time to standard matrix DM formation execution time. For $N > 150$ the centrosymmetric algorithm is faster. For larger N, the centrosymmetric algorithm is over three times faster.

5.3 System matrix condition number

Theoretically, the 2-norm condition number of a SPD matrix is the ratio of the largest eigenvalue of the matrix to the smallest eigenvalue. Ill-conditioning can cause the eigenvalues to be calculated as complex numbers and a more stable approach is to use the singular value decomposition (SVD) to calculate the 2-norm condition number as $\kappa_2(B) = \sigma_{max}/\sigma_{min}$ where σ are the singular values of the matrix. The SVD is necessarily an iterative process, therefore an exact flop formula is not possible. The flop count also depends on whether the singular vectors are calculated. An approximate flop count for calculating only the singular values is given in [9, p. 254] as $\frac{8}{3}N^3$. Due to Theorem 2, the 2-norm condition number of a centrosymmetric system matrix B, as well as the condition numbers of the matrices L and M, can be calculated with a factor of four reduction in the dominant term in the flop count when compared to the standard algorithm for the 2-norm condition number of a matrix. The condition numbers of L and M are of interest as they are the two $P \times P$ matrices that are inverted in order to solve a centrosymmetric linear system or to form a derivative matrix.

The left image of Figure 4 shows the execution time of the centrosymmetric algorithm versus the standard algorithm over a range of N. For larger N the centrosymmetric algorithm is approximately five times more efficient. The factor

of four savings from the SVD and additional savings from only forming half-sized distance and system matrices contribute to the overall efficiency factor. The right image of Figure 4 shows the condition number of the system matrix over a range of shape parameter as computed by the two algorithms. As expected the outputs agree up to condition numbers of size $\mathcal{O}\left(10^{16}\right)$ and then due the ill-conditioning of the problem there is a slight variation for larger condition numbers.

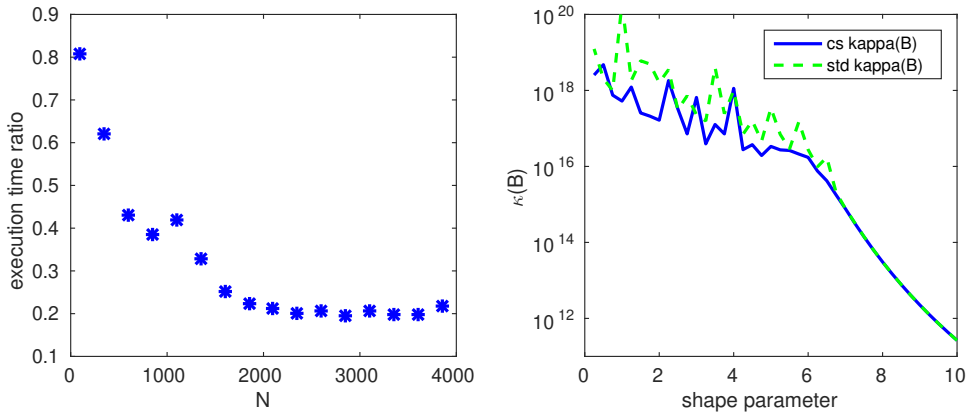

Figure 4: Left: Ratio of centrosymmetric 2-norm condition number execution time to the standard algorithm. For larger N the centrosymmetric algorithm is approximately five times faster. The script ***condBench.m*** of the MRBFT [19, 22] produces the image. Right: condition number versus the shape parameter from the two algorithms. The script ***centroCondTest.m*** of the MRBFT produces the image.

5.4 Matrix-vector multiplication

Centrosymmetric matrix multiplication is an old idea that has been reinvented several times. According to reference [3, p. 190], its origin traces back to Runge who used the idea as a building block in an early variant of the FFT algorithm. In the context of pseudospectral methods, it is discussed in [3, 25]. The algorithm has previously been referred to as parity matrix multiplication [3] as well as even-odd decomposition [25].

Given an appropriate center distribution (section 4), RBF derivative evaluation matrices H have a centro structure. The inverse of a centrosymmetric matrix is centrosymmetric ($B^{-1} = (JBJ)^{-1} = J^{-1}B^{-1}J^{-1} = JB^{-1}J$), a centrosymmetric matrix times a centrosymmetric matrix is a centrosymmetric matrix ($JBJJAJ = J(BA)J$), and a skew-centrosymmetric matrix times a centrosymmetric matrix is a skew-centrosymmetric matrix $-JBJJAJ =$

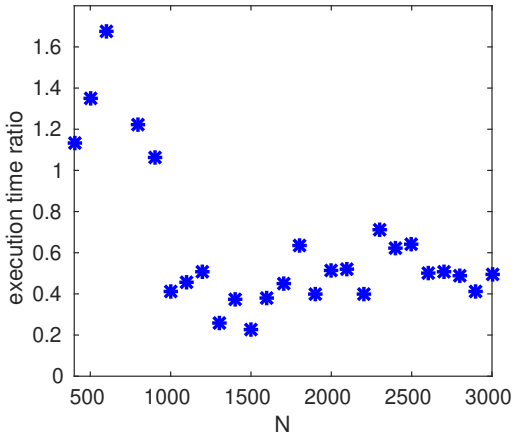

Figure 5: Ratio of centrosymmetric matrix multiplication execution time to standard matrix multiplication execution time. For $N \geq 900$ the centrosymmetric algorithm is faster and the limiting efficiency factor of two is being approached. The script ***multiplicationBench.m*** produces the image.

$-J(BA)J$. As a result RBF differentiation matrices $D = B^{-1}H$ have a centro structure if B and H have a centro structure.

Matrix vector multiplication can be structured into blocks as

$$\left[\begin{array}{c} f_1 \\ \hline f_2 \end{array}\right] = \left[\begin{array}{c|c} B_{11} & B_{12} \\ \hline B_{21} & B_{22} \end{array}\right] \left[\begin{array}{c} x_1 \\ \hline x_2 \end{array}\right].$$

The blocks can be executed as subproblems to obtain the larger result

$$\begin{aligned} f_1 &= B_{11}\, x_1 + B_{12}\, x_2 \\ f_2 &= B_{21}\, x_1 + B_{22}\, x_2 \end{aligned}$$

where the matrix blocks are $P \times P$ and the vector blocks are $P \times 1$.

The (skew-)centrosymmetric structure of a matrix can be used to accomplish a matrix-vector multiplication using half as many flops as the general case. For the skew-centrosymmetric case, N even, and $P = N/2$ matrix-vector multiplication can be accomplished as follows. Let J_N and J_P be contra-identity matrices with dimensions $N \times N$ and $P \times P$ respectively. Break the vector x which is to be multiplied up into its even (symmetric) and odd (skew-symmetric) parts as

$$x = \overbrace{\frac{1}{2}\left(x + J_N x\right)}^{x^e} + \overbrace{\frac{1}{2}\left(x - J_N x\right)}^{x^o}. \tag{19}$$

Note that

$$x^e = \frac{1}{2}\left(x + J_N x\right) = \left[\begin{array}{c} x_1^e \\ J_P x_1^e \end{array}\right]$$

and

$$x^o = \frac{1}{2}\left(x - J_N x\right) = \begin{bmatrix} x_1^o \\ -J_P x_1^o \end{bmatrix}.$$

Due to the linearity of matrix multiplication

$$f = Bx = B(x^e + x^o) = f^e + f^o.$$

That is, the matrix can be applied to x^e and x^o separately and then the results can be combined to get f. The result of multiplying the upper half of the even part is

$$f_1^e = B_{11}\, x_1^e - J_P B_{21} J_P \left(J_P x_1^e\right) = \overbrace{\left(B_{11} - J_P B_{21}\right)}^{L} x_1^e \tag{20}$$

and

$$-J_P f_2^e = -J_P \left(B_{21}\, x_1^e - J_P B_{11} J_P \left(J_P x_1^e\right)\right) = \left(B_{11} - J_P B_{21}\right) x_1^e = f_1^e. \tag{21}$$

Equations (20) and (21) imply that

$$f^e = \begin{bmatrix} f_1^e \\ -J_P f_1^e \end{bmatrix}$$

and that f^e is odd. As a result, only half of the vector needs to be computed as $Lx_1^e = f_1^e$ via multiplication by a $P \times P$ matrix. The result of multiplying the upper half of the odd part is

$$f_1^o = B_{11}\, x_1^o - J_P B_{21} J_P \left(-J_P x_1^o\right) = \overbrace{\left(B_{11} + J_P B_{21}\right)}^{M} x_1^o \tag{22}$$

and also note that

$$J_P f_2^o = J_P \left(B_{21}\, x_1^o - J_P B_{11} J_P \left(-J_P x_1^o\right)\right) = \left(B_{11} + J_P B_{21}\right) x_1^o = f_1^o. \tag{23}$$

Equations (22) and (23) imply that

$$f^o = \begin{bmatrix} f_1^o \\ J_P f_1^o \end{bmatrix}$$

and that is f^o even. Only half of the odd product needs to be computed as $Mx_1^o = f_1^o$. The full vector f can be computed as

$$\begin{bmatrix} f_1 \\ f_2 \end{bmatrix} = \begin{bmatrix} f_1^o + f_1^e \\ J_P f_1^o - J_P f_1^e \end{bmatrix} = \begin{bmatrix} f_1^o + f_1^e \\ J_P (f_1^o - f_1^e) \end{bmatrix}. \tag{24}$$

The algorithm is nearly the same if B is centrosymmetric rather than skew-centrosymmetric. The only differences are that L and M are replaced with $\hat{L} = B_{11} + J_P B_{21}$ and $\hat{M} = B_{11} - J_P B_{21}$ respectively and that the result of

multiplying the even vector is even and the result of multiplying the odd vector is odd which results in the vector f being reconstructed as

$$\begin{bmatrix} f_1 \\ f_2 \end{bmatrix} = \begin{bmatrix} f_1^o + f_1^e \\ -J_P f_1^o + J_P f_1^e \end{bmatrix} = \begin{bmatrix} f_1^o + f_1^e \\ J_P(f_1^e - f_1^o) \end{bmatrix}. \tag{25}$$

Since $L = B_{11} - J_P B_{21} = B_{11} - J_P(-J_P B_{12} J) = B_{11} + B_{12} J_P$ and $M = B_{11} + J_P B_{21} = B_{11} + J_P(-J_P B_{12} J_P) = B_{11} - B_{12} J_P$ it is only necessary to construct and use half of the matrix B. There is the option to construct either the upper half or the left half. The software [19, 22] uses the left half of the matrix in all algorithms.

The leading term in the flop count of the centrosymmetric matrix-vector multiplication algorithm is

$$2 \cdot 2 \left(\frac{N}{2}\right)^2 = N^2.$$

Asymptotically, a factor of two is saved in the flop count compared to the standard algorithm. An additional factor of two is saved in storage. The performance of the centrosymmetric algorithm versus the standard algorithm over a range of N is shown in Figure 5.

6 Numerical PDE examples

In this section a two dimensional time-dependent PDE problem and a boundary value problem, both with Dirichlet boundary conditions, are solved while taking advantage of a centro center distribution on a unit circle domain. The RBF method for PDEs is referred to as the asymmetric collocation method or Kansa's method [13].

The boundary value problem consists of the Poisson equation

$$u_{xx} + u_{yy} = -\pi^2 \sin(\pi x) \sin(\pi y) \tag{26}$$

with Dirichlet boundary conditions prescribed according to the exact solution

$$u(x, y) = 1 - x + xy + \frac{1}{2} \sin(\pi x) \sin(\pi y).$$

The time-dependent problem is the diffusion-reaction equation

$$u_t = \nu \left(u_{xx} + u_{yy}\right) + \gamma u^2 (1 - u). \tag{27}$$

with the initial condition and Dirichlet boundary conditions specified from the exact solution

$$u(x, y, t) = \left[1 + \exp(a \left[x + y - bt\right] + c)\right]^{-1} \tag{28}$$

where $a = \sqrt{\gamma/4\nu}$, $b = 2\sqrt{\gamma\nu}$, and $c = a(b - 1)$.

The centro center distribution was obtained via an origin extension, but an x-axis or y-axis extension would work as well. In order for the resulting matrices

to be centrosymmetric after boundary conditions are applied, the centers on the boundary need to be ordered into two groups. Let N be the number of total centers, n_b an even number of boundary centers, and $n_h = n_b/2$. Let X be a centro center distribution with centers ordered so that half the boundary centers come first, followed by all interior centers, and then the remaining half of the boundary centers. Then both the steady and time-dependent problem have the evaluation matrix (10) with the following block structure

$$H = \left[\begin{array}{l} b_{jk}\,(6), \quad j = 1, \ldots, n_h \text{ and } k = 1, \ldots, N \\ \hline h_{jk}\,(10), \quad j = n_h + 1, \ldots, N - n_h \text{ and } k = 1, \ldots, N \\ \hline b_{jk}\,(6), \quad j = N - n_h + 1, \ldots, N \text{ and } k = 1, \ldots, N \end{array}\right] \tag{29}$$

where the operator $\mathcal{D}$ in (10) is the Laplacian. Both problems have the same system matrix B (6).

The solution of the steady problem is found by first finding the expansion coefficients $a = H^{-1}f$ by solving the linear system $Ha = f$ by the algorithm of section 5.1 except that LU factorization is used instead of a Cholesky factorization as H is not SPD. After a shape parameter is selected, the condition number of the matrix H is calculated by the centrosymmetric algorithm of section 5.3. If the condition number is too small or too large, a different shape parameter is selected until the condition number of the matrix is approximately $\mathcal{O}\left(10^{16}\right)$. In this case there is no guarantee that H is invertible, as counter examples exist to show that it is possible for H to be singular [11]. Despite not being able to show that H is invertible, extensive application of the method has shown that in practice H is invertible and the method has been widely applied since its introduction over 25 years ago. After the expansion coefficients have been found, the approximate solution is evaluated as $u = Ba$ by the centrosymmetric multiplication algorithm of section 5.4. With $N = 5000$, standard algorithms take 30.8s to execute while the same problem takes 6.6s using centrosymmetric algorithms.

The solution of the time-dependent problem is found by first selecting a shape parameter and then the condition number of the matrix B is calculated by the centrosymmetric algorithm of section 5.3. If the condition number is too small or too large, a different shape parameter is selected until the condition number of the matrix is approximately $\mathcal{O}\left(10^{16}\right)$. Then the derivative matrix $D = HB^{-1}$ which discretizes the space derivatives is formed by the centrosymmetric algorithm of section 5.2. After the PDE is discretized in space with the RBF method, the resulting semi-discrete system $u_t = F(u)$ is advanced in time with an explicit Runge-Kutta method. At each stage of the Runge-Kutta method the matrix-vector multiplication Du is performed using the centrosymmetric algorithm of section 5.4. With $N = 5000$, standard algorithms take 592s to execute while the same problem takes 81s using centrosymmetric algorithms when advancing the solution to time $t = 5$ with a time step size of $\Delta t = 0.001$.

The script ***poissonCentro.m*** carries out the steady PDE example and the scripts ***diffusionReactionCentro.m*** and ***diffusionReactionCentroDriver.m*** carry out the time-dependent PDE example.

7 Consequences of not preserving symmetry

Forming or operating on theoretically centrosymmetric RBF matrices with algorithms that do not preserve centrosymmetry can have negative consequences.

For example, consider the 1d advection equation $u_t + u_x = 0$ on the interval $[-1, 1]$ with periodic boundary conditions. In this case the derivative matrix theoretically has purely imaginary eigenvalues [18]. The problem is discretized in space with the IQ RBF, $\varepsilon = 4.5$, and $N = 60$ centers distributed via $x_k = \cos(k\pi/(N-1))$, $k = 0, \ldots, N-1$. The system matrix has a condition number that is $\mathcal{O}\left(10^{17}\right)$. The derivative matrix D is formed two ways: 1) via a Cholesky factorization for which $\|D + JDJ\|_2 = 4.2\text{e}5$, and 2) via the centrosymmetric algorithm from section 5.2 for which $\|D + JDJ\|_2 = 0$. The skew-centrosymmetric DM differentiates a symmetric function with a smaller error than does the DM formed with the standard algorithm and the error is symmetric (left image of Figure 6). The eigenvalues of the skew-centrosymmetric DM are purely imaginary whereas the non skew-centrosymmetric DM has eigenvalues with both significant positive and negative real parts which prevents the PDE from being advanced in time with a numerical ODE method.

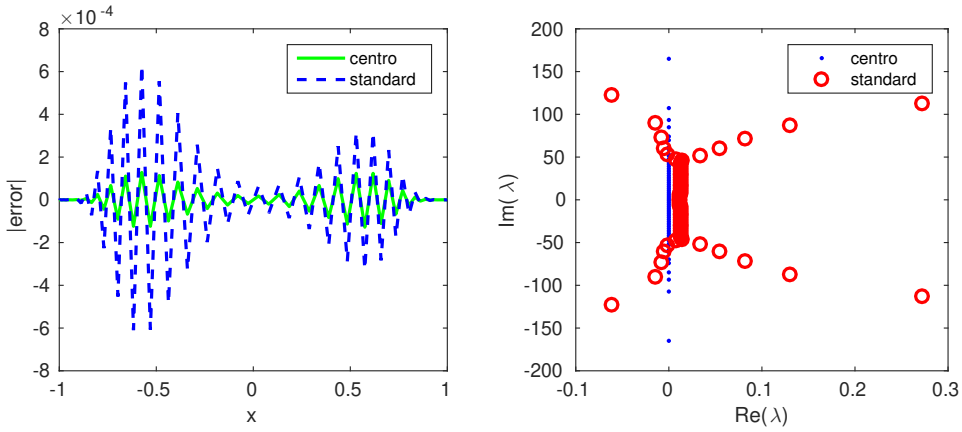

Figure 6: Differentiation matrix constructed and applied with and without preservation of centrosymmetry. Left: differentiation of a symmetric function. Right: differentiation matrix eigenvalues.

8 Conclusions

The RBF method's complete freedom of choice of center locations can be exploited to reduce computational expense and storage requirements of the methods as well as to improve accuracy and properties such as eigenvalue stability in

time-dependent PDE problems. Center distributions that result in centrosymmetric structure in RBF system and differentiation matrices do not place restrictions on the locations of the centers but rather on the distances between them. The desired distances result in signed distance matrices (8) with centrosymmetry. The framework allows scattered centers with boundary clustering that is desirable with the RBF method. In two dimensions, the center distributions are possible in domains that are symmetric with respect to either the x-axis, y-axis, the origin, or can be made so by a linear transformation or rotation.

For large N, algorithms that exploit centrosymmetry can be used to reduce the flop counts in common linear algebra operations that are associated with the RBF method. The dominant term in the flop count for matrix factorization can be reduced from $\frac{1}{3}N^3$ to $\frac{1}{12}N^3$, for forming a derivative matrix from $\frac{28}{12}N^3$ to $\frac{7}{12}N^3$, and for matrix-vector multiplication from $2N^2$ to N^2. A factor of two is saved in storage as a centrosymmetric matrix is fully described by only half its entries. Due to the ill-conditioned nature of the standard RBF basis, extended precision floating point arithmetic is often used with RBF methods [12, 20, 23]. Centrosymmetry can be used to mitigate the longer execution times of extended precision algorithms that are implemented in software.

For both larger N and for smaller N used with local methods, the use of centrosymmetric algorithms to preserve symmetry can be beneficial for maintaining accuracy and properties such as the correct eigenstructure of differentiation matrices.

All algorithms described in this manuscript are implemented in freely available software [19, 22].

References

[1] I. T. Abu-Jeib. Algorithms for centrosymmic and skew-centrosymmetric matrices. *Missouri Journal of Mathematical Sciences*, 18(1):46–53, 2006. 3, 5.1

[2] A. L. Andrew. Solution of equations involving centrosymmetric matrices. *Technometrics*, 15:405–407, 1973. 3

[3] John P. Boyd. *Chebyshev and Fourier Spectral Methods*. Dover, second edition, 2001. 1, 5.4

[4] M. D. Buhmann. *Radial Basis Functions*. Cambridge University Press, 2003. 2

[5] A. Cantoni and P. Butler. Eigenvalues and eigenvectors of symmetric centrosymmetric matrices. *Linear Algebra and its Applications*, 13:275–288, 1976. 3, 1, 2, 3

[6] A. R. Collor. On centrosymmetric and centroskew matrices. *Quarterly Journal of Mechanics and Applied Mathematics*, 15:265–281, 1962. 3

[7] G. Fasshauer and M. McCourt. Stable evaluation of Gaussian RBF interpolants. *SIAM Journal on Scientific Computing*, 34:737–762, 2012. 2

[8] G. E. Fasshauer. *Meshfree Approximation Methods with Matlab*. World Scientific, 2007. 2

[9] G. Golub and C. Van Loan. *Matrix Computations*. Johns Hopkins Press, third edition, 1996. 5.3

[10] I. J. Good. The inverse of a centrosymmetric matrix. *Technometrics*, 12:925–928, 1970. 3

[11] Y. C. Hon and R. Schaback. On unsymmetric collocation by radial basis function. *Applied Mathematics and Computations*, 119:177–186, 2001. 6

[12] C.-S. Huang, C.-F. Leeb, and A.H.-D. Cheng. Error estimate, optimal shape factor, and high precision computation of multiquadric collocation method. *Engineering Analysis with Boundary Elements*, 31:614–623, 2007. 8

[13] E. Kansa. Multiquadrics - a scattered data approximation scheme with applications to computational fluid dynamics II: Solutions to parabolic, hyperbolic, and elliptic partial differential equations. *Computers and Mathematics with Applications*, 19(8/9):147–161, 1990. 6

[14] A. Karageorghis, C.S. Chen, and Y.-S. Smyrlis. A matrix decomposition RBF algorithm: Approximation of functions and their derivatives. *Applied Numerical Mathematics*, 57(3):304–319, 2007. 1

[15] X.-Y. Liu, C. S. Chen, and A. Karageorghis. Conformal mapping for the efficient solution of poisson problems with the kansa-RBF method. *Journal of Scientific Computing*, 71(3):1035–1061, 2016. 1

[16] A. Melman. Symmetric centrosymmetric matrix-vector multiplication. *Linear Algebra and its Applications*, 320:193–198, 2000. 3

[17] H. Niederreiter. *Random Number Generation and Quasi-Monte Carlo Methods*. CBMS-NSF, SIAM, Philadelphia, 1992. 4

[18] R. Platte and T. Driscoll. Eigenvalue stability of radial basis functions discretizations for time-dependent problems. *Computers and Mathematics with Applications*, 51:1251–1268, 2006. 7

[19] S. A. Sarra. A Matlab radial basis function toolkit with symmetry exploitation, regularization, and extended precision. `http://www.scottsarra.org/rbf/rbf.html`. 1, 3, 4, 5.4, 8

[20] S. A. Sarra. Radial basis function approximation methods with extended precision floating point arithmetic. *Engineering Analysis with Boundary Elements*, 35:68–76, 2011. 8

[21] S. A. Sarra. Regularized symmetric positive definite matrix factorizations for linear systems arising from rbf interpolation and differentiation. *Engineering Analysis with Boundary Elements*, 44:76–86, 2014. 5.1

[22] S. A. Sarra. The Matlab radial basis function toolbox. *Journal of Open Research Software*, 5, March 2017. 1, 3, 4, 5.4, 8

[23] S. A. Sarra and E. J. Kansa. Multiquadric radial basis function approximation methods for the numerical solution of partial differential equations. *Advances in Computational Mechanics*, 2, 2009. 2, 8

[24] R. Schaback. Error estimates and condition numbers for radial basis function interpolation. *Advances in Computational Mathematics*, 3:251–264, 1995. 2

[25] A. Solomonoff. A fast algorithm for spectral differentiation. *Journal of Computational Physics*, 98:174–177, 1992. 1, 5.4

[26] A. J. Wathen and S. Zhu. On spectral distribution of kernel matrices related to radial basis functions. *Numerical Algorithms*, 70(4):709–726, 2015. 2

[27] J. R. Weaver. Centrosymmetric (cross-symmetric) matrices, their basic properties, eigenvalues, and eigenvectors. *American Mathematical Monthly*, 92:711–717, 1985. 3

[28] H. Wendland. *Scattered Data Approximation*. Cambridge University Press, 2005. 2, 2