Radial basis functions: achievements and challenges

E. J. Kansa
Convergent Solutions, USA

Abstract

Traditionally, the various forms of compactly supported polynomial based schemes such as finite difference, element, and volume methods were used in the numerical solution of ordinary and partial differential equations (ODEs and PDEs) as well as integral equations (IEs). The primitive computers had limited memory and processing speed. As time progressed, spectral and pseudo-spectral methods that possess exponential or spectral convergence that use tensor products of global expansions of one-dimensional orthonormal functions were used to solve ODEs, PDEs, and IEs. In the past 25 years, radial basis functions (RBFs) that can be either compactly supported or global gained in importance. However, those global RBFs that are C^∞ with shape parameters enjoy exponential convergence. The solution accuracy can be increased by either the spatial resolution or increasing the shape parameters. The performance of C^∞ RBFs has been demonstrated in a wide variety of linear and nonlinear elliptic, hyperbolic and parabolic PDEs and IE applications. Both well posed and ill-posed problems can be solved as well as with defined arithmetic and fuzzy arithmetic, problems with fractional derivatives, etc. Computers with the standard 32 bit chips limit the inherent power of the global RBFs by giving rise to potentially severely ill-conditioned systems. However, procedures using preconditioning and domain decomposition can be used to overcome this limitation. Recently, fast extended precision software enables the use of large shape parameters outperforming compactly supported finite elements and other methods. Also, the Galperin–Zheng weak formulation allows treatment of both well posed and ill-posed problems because it looks for solutions with global, rather than local minima. Presently, meshfree C^∞ RBFs appear to have the potential of solving many important scientific problems in higher dimensions.

WIT Transactions on Modelling and Simulation, Vol 61, © 2015 WIT Press
www.witpress.com, ISSN 1743-355X (on-line)
doi:10.2495/BEM380011

Keywords: partial differential equations, integral equations, radial basis functions, strong and weak form, extended precision, global minimization.

1 Introduction

The objective of numerical solutions to integral equations (IEs) and partial differential equations (PDEs) is to obtain realistic approximations to the set of dependent variables, $U(\overrightarrow{x},t)$, over a finite domain, $\Omega \subset \Re^n$. Let $\mathcal{L}$ be either a linear or nonlinear interior operator and f be the corresponding forcing function on $\Omega \backslash \partial\Omega$. Let ß be either a linear or nonlinear boundary operator and g be the corresponding forcing function on $\partial\Omega$. For convenience, the combined interior and boundary operator on Ω be $\Im$, and let $\wp$ be the combined forcing function on the interior and boundary. Thus over Ω, the respective combined operators and forcing functions on Ω are:

$$\Im = [\mathcal{L}, \text{ß}]^\top, \tag{1}$$

$$\wp = [\text{f}, \text{g}]^\top. \tag{2}$$

This general form applies to either PDEs or IEs.

2 Finite difference methods

The numerical solution of IEs and PDEs historically is linked to the currently available electronic computers. Early computers had small memory storage and slow processors. Of necessity, compactly supported methods such as finite difference and finite element methods were used. The earliest attempts of approximations were solutions of simultaneous equations using polynomial basis functions. Polynomials are simple functions capable of representing nonlinear functions, and are the starting points for complicated functions and numerical methods. The primary objective is to minimize the maximum error between the polynomial and actual solution.

Over a local stencil or grid, finite difference methods (FDMs) are based upon the assumption that a piece-wise continuous function can be approximated by a low order polynomial, p(x), such as a Lagrange polynomial and the derivatives of that function are approximated by the derivatives of p(x). The approximating function, p(x), is only continuous at the stencil interface, but its derivatives from the left and right of the stencil boundary are discontinuous. A kth degree polynomial can be recast in terms of a Taylor series expansion about the point x_i:

$$U(x_i+\Delta x) = U(x_i)+\Lambda x U_x+(\Delta x^2/2!)U_{xx}+\cdots \tag{3}$$

$$U_x(x_i+\Delta x) = U_x(x_i)+\Delta x U_{xx}+(\Delta x^2/2!)U_{xxx}+\cdots \tag{4}$$

For slowly varying functions, higher order derivatives are negligibly small and the use of local polynomial interpolants based on a small number of interpolating grid points is reasonable. In higher dimensions, tensor product grids are used.

In one dimension, the FDM yields a sparse system of equations; but with increasing dimensionality, the systems become increasingly denser, especially when cross-derivatives are included. The truncation errors can generate numerical instabilities unless the physics is changed by adding nonphysical numerical viscosity. Theoretically, the truncation errors go to zero as Δx and Δt go to zero, but in practice, these truncation errors *always* remain. These artificial derivative discontinuities alter wave speeds. However, using low-degree local polynomials to approximate solutions containing very significant spatial or temporal variation requires a very fine grid in order to resolve the solution accurately. FDM were one of the first methods used on primitive electronic computers to solve ODEs and PDEs.

3 Splines

In practice, ordinary high order polynomials that pass through k, k $\geq$ 4, points exhibit the polynomial “snaking” phenomenon, yielding very unreliable derivative estimates. Because of the drawbacks of ordinary polynomials, a different type of polynomial, namely, cubic and quintic splines that enforce varying orders of derivative continuity at stencil interfaces are sometimes used (see Ferguson [1]). In higher dimensions, tensor product approximations are used.

4 Finite volumes

Finite volume methods (FVMs) are similar to FDMs, except local definite integrals are constructed on a stencil, see Eymard *et al.* [2]. Extensive quantities such as mass, species, and total energy are cell-centered quantities, whereas the momentum components are staggered at the midpoints of the cell faces. In FVMs, the application of the Gauss divergence theorem converts spatial partial derivatives to differences in surface fluxes. If the strictly conservative dependent variables are used, then the strict conservation of mass, specie, momentum components, and total energy is guaranteed. Another advantage of the finite volume method is that it is easily formulated to allow for unstructured meshes. An important disadvantage is the false numerical diffusion arising from the linear interpolation of face and cell centered quantities; the other is the discontinuous nature of flux differences at cell interfaces.

5 Galerkin finite element methods

In the Galerkin approach, an approximation is found that makes residual orthogonal to the space from which $U(\vec{x},t)$ comes. This is accomplished by ensuring that the weighted residual error is required to vanish when integrated over the element is orthogonal to each of the basis polynomial functions. The finite element method (FEM), see Reddy [3], is a special case of Galerkin method in which integrals over local elements (meshes) are constructed as the inner product of the residual acting upon a trial function that is multiplied by a weight function. The integration converts the spatial derivatives into surface integral differences. The Galerkin method constructs an integral of the inner product of the residual and the weight functions and sets the integral to zero. The residual is the error caused by the trial functions, and the weight functions that are polynomial approximation functions that projects onto the residual. The integral over the weight function averages the residual that is minimized in a least-squares sense. However, this minimum is, in general, a local minimum because the depth of this minimum is strongly dependent upon the starting conditions, and there is no guarantee that this minimum is a deep minimum.

Such problems can be cast in terms of seeking the minimum of a function of n variables, $\mathbf{U}(\vec{\mathrm{x}})$, see Harris and Stocker [4]. Define, **J**, as the Jacobian to be a vector of the first order partial derivatives of **U**, and, **H**, the Hessian matrix to be a square matrix of second-order partial derivatives of **U**. A Taylor series expansion of **U** is given by:

$$\mathbf{U}(\vec{x}+\Delta\vec{\mathrm{x}}) = \mathbf{U}(\vec{\mathrm{x}}) + \Delta\vec{\mathrm{x}}\mathbf{J} + 1/2\Delta\vec{\mathrm{x}}^{\mathrm{T}}\mathbf{H}\Delta\vec{\mathrm{x}} + \cdots \tag{5}$$

where $\vec{\mathrm{x}}_k$ is a critical point at which the Jacobian, $\mathbf{J}(\vec{\mathrm{x}}_k)$ =0, then $\mathbf{U}(\vec{\mathrm{x}}_k)$ may be at a local maximum, minimum, or inflection point. To determine what type of point $\vec{\mathrm{x}}_k$ is, one must examine the eigenvalues of the Hessian matrix at a critical point, $\vec{\mathrm{x}}_k$. The following test can be applied at any critical point for which the Hessian matrix is invertible:

1. If the Hessian possesses all positive eigenvalues at $\vec{\mathrm{x}}_k$, then $\mathbf{U}(\vec{\mathrm{x}}_k)$ is a local minimum.
2. If the Hessian possesses all negative eigenvalues at $\vec{\mathrm{x}}_k$, then $\mathbf{U}(\vec{\mathrm{x}}_k)$ is a local maximum.
3. If the Hessian has both positive and negative eigenvalues at $\vec{\mathrm{x}}_k$, then $\mathbf{U}(\vec{\mathrm{x}}_k)$ is a saddle point.

Assume the dependent variable, $\mathrm{U}(\vec{\mathrm{x}})$ is a linear combination of polynomial functions over each kth element of the discretized domain, and the weighting function, $\mathrm{w}(\vec{\mathrm{x}})$, is likewise is a linear combination of polynomials with different expansion coefficients:

$$\mathrm{U}(\vec{\mathrm{x}}) = \sum_{k=1}^{N} \phi(\vec{\mathrm{x}})\alpha_k, \tag{6}$$

$$\mathrm{w}(\vec{x}) = \sum_{k=1}^{N} \phi(\vec{\mathrm{x}})\mu_k. \tag{7}$$

Then, the Galerkin procedure yields:

$$\sum_{k=1}^{N}\left\{\int_k w^T \Im U - \int_k w^T \wp\right\} d\vec{\boldsymbol{x}} = 0, \tag{8}$$

For arbitrary variations of μ_k, in the expansion coefficients of the weight function over all elements, one obtains

$$\sum_{k=1}^{N}\left\{\int_k w^T \Im\phi(\overrightarrow{x})\right\}d\overrightarrow{\boldsymbol{x}}\alpha_k - \left\{\int_k w^T \wp\right\}d\overrightarrow{\boldsymbol{x}} = 0. \tag{9}$$

For time dependent problems, it is assumed that the spatial and temporal variables are separable, and the time dependent expansion coefficients can be approximated by the method of lines for ODEs. Clearly, the use of fine grids requires significant computational resources in simulations of interest to science and engineering.

5.1 Wavelets

Wavelets are orthonormal polynomial basis functions that are rotationally, translationally, and dilationally invariant (see Chui [5]). Especially for time series and image transmission, high signal compression efficiencies can be achieved with relatively few basis functions. The compression can be either lossless or lossy. The popular wavelets have compact support. In higher dimensions, tensor product approximations are used.

6 Boundary element methods

The integral equation is an exact solution of certain classes of PDEs. The boundary element method (BEM) (see Cheng and Cheng [6]) uses the given boundary conditions to fit boundary values into the IEs. BEM is applicable to problems for which Green's functions can be calculated involving fields in linear homogeneous media. BEM has high accuracy because of its semi-analytic nature and use of integrals. BEM matrices are dense and non-symmetrical. BEM is good for PDEs and IEs with infinite domains such as wave propagation, stress concentration. BEMs reduces a 3D body problem to a 2D surface problem. Once this is done, in the post-processing stage, the integral equation can then be used again to calculate numerically the solution directly at any desired point in the interior of the solution domain.

7 Spectral methods

Unlike FDM, FEM, FVM, spectral methods are global methods, in which the computation at any given point depends not only on information at neighboring points, but over the entire domain. Spectral methods (see Gottlieb and Orzag [7]), yield very low errors and are exponentially convergent. Relatively few points can yield spectral convergence. Spectral methods can be constructed with other orthogonal polynomials rather than only the with Fourier basis functions. Chebyshev or Legendre polynomials are used for problems on $[-1, 1]$, Laguerre polynomials are used over $[0, \infty)$, and Hermite polynomials are used over $(-\infty, \infty)$. Global methods are preferable to local methods when the solution varies considerably in time or space, when very high spatial resolution is required, and also when long time integration is needed. Extensions to higher dimensions require tensor-product grids since the popular spectral polynomials are one-dimensional in nature.

8 Radial basis functions

A radial basis function (RBFs) is a univariate (one-dimensional) function that depends only upon the radial distance, $\mathrm{r}_{i,j} = \|\overrightarrow{\mathbf{x}_\mathrm{i}} - \overrightarrow{\mathbf{y}}_\mathrm{j}\|$. where $\mathrm{r}_{i,j}$ is usually the Euclidean metric from a data center, $\overrightarrow{\mathbf{y}_j} \in \Re^n$ to the evaluation point, $\overrightarrow{\mathbf{x}_\mathrm{i}} \in \Re^n$; an n-dimensional problem reduces to one dimension. Note, other possible metrics have not been thoroughly examined. In addition, no mesh generation is required becoming a huge advantage when the spatial dimension increases, especially beyond three dimensions.

There are three classes of RBFs:

1. Compactly supported RBFs were developed by Wendland [8], and Wu [9]:

$$\phi(r) = (1-r)_+^4(4r+1), \text{ (2D RBF, Wendland)}, \tag{10}$$

$$\phi(r) = (1-r)_+^5(5r4 + 25r3 + 48r2 + 40r + 8), \text{ (2D RBF, Wu)} \tag{11}$$

2. Poly-harmonic spline:

$$\phi(r) \ = r^k, \ k = 1, 3, 5, .. \text{ or} \tag{12}$$

$$\phi(r) = r^k \ln(r), \ k = 2, 4, 6, .. \tag{13}$$

3. C^∞ RBF splines can be any transcendental univariate function with a local scale factor, $\epsilon_j = 1/\mathrm{c}_j$, such that the product, $\mathrm{r}_{i,j}\epsilon_j$ is dimensionless. Some examples are:

$$\phi(r_{i,j}\epsilon_j) = [1 + (r_{i,j}\epsilon_j)^2]^\beta, \beta \geq -1/2 \text{ (multiquadric (MQ))}, \tag{14}$$

$$\phi(r_{i,j}\epsilon_j) = \exp(-(r_{i,j}\epsilon_j)^2), \text{Gaussians}, \tag{15}$$

$$\phi(r_{i,j}\epsilon_j) = \sin(r_{i,j}\epsilon_j)/(r_{i,j}\epsilon_j) \text{ sinc, etc.} \tag{16}$$

The MQ RBF was invented by R. L. Hardy [10, 11] working with gravitational and magnetic anomalies. Although the MQ and Gaussian C^∞ RBFs are the most commonly used, a comprehensive study of the best usage of each type of C^∞ RBF spline has yet to be undertaken. It is important to note that Hardy [11] recommended scaling all data centers to the unit volume in $\Re^n$. Currently, there is no theory to guide in the optimal choice of the shape parameters of the C^∞ RBFs, but the investigations of Luh [12] are quite promising.

RBFs have had extensive application in the areas of interpolation, least-squares approximations, neural network approximations, machine learning, image reconstruction, edge detection, and the numerical solutions of integral and partial differential equations,ill-posed problems, fractional derivatives, fuzzy mathematics, etc. RBFs have the ability to treat noisy data by regularization, providing the optimal solution. In addition, RBFs are flexible and robust and are simple to implement.

9 C^{∞} RBFs applied to PDE and IEs

Kansa [13, 14] first applied MQ RBFs to the solution of hyperbolic, parabolic, and elliptic PDEs using the strong form collocation formulation. Similarly to FEM, assume that time and space are separable, and the only time dependence occurs in the expansion coefficients. Consider a time dependent problem in which the initial conditions are specified over the domain. To find the initial of expansion coefficient, $\{\boldsymbol{\alpha}(0)\}$, split the sets of RBFs and expansion coefficients into the interior (int) and boundary (bnd) giving rise to

$$\mathbf{A}\boldsymbol{\alpha}(0)=\begin{pmatrix} \boldsymbol{\Phi}_{int} & \boldsymbol{\Phi}_{bnd} \\ \boldsymbol{\Phi}_{int} & \boldsymbol{\Phi}_{bnd} \end{pmatrix} \cdot \begin{pmatrix} \boldsymbol{\alpha}_{int}(\mathrm{t}) \\ \boldsymbol{\alpha}_{bnd}(\mathrm{t}) \end{pmatrix} = \begin{pmatrix} \mathbf{U}_{int}(\overrightarrow{\mathbf{x}},0) \\ \mathbf{U}_{bnd}(\overrightarrow{\mathbf{x}},0) \end{pmatrix} \tag{17}$$

If A is invertible, then $\boldsymbol{\alpha}(0) = \mathbf{A}^{-1}\mathrm{U}(\overrightarrow{\mathbf{x}},0)$. Then construct a system of equation with interior operator, $\mathcal{L}$ acting only upon $\Omega\backslash\partial\Omega$, and ß acting upon $\partial\Omega$ yielding:

$$\mathbf{E}\boldsymbol{\alpha}(\mathrm{t}) = \begin{pmatrix} \mathcal{L}\boldsymbol{\Phi}_{int} & \mathcal{L}\boldsymbol{\Phi}_{bnd} \\ ß\boldsymbol{\Phi}_{int} & ß\boldsymbol{\Phi}_{bnd} \end{pmatrix} \cdot \begin{pmatrix} \boldsymbol{\alpha}_{int}(\mathrm{t}) \\ \boldsymbol{\alpha}_{bnd}(\mathrm{t}) \end{pmatrix} = \begin{pmatrix} \mathbf{f}_{int}(\overrightarrow{\mathbf{x}},\mathbf{t}) \\ \mathbf{g}_{bnd}(\overrightarrow{\mathbf{x}},\mathbf{t} \end{pmatrix} = \mathbf{h}(\overrightarrow{\mathbf{x}},\mathbf{t})\,. \tag{18}$$

If **E** is invertible, then $\boldsymbol{\alpha}(\mathrm{t})= \mathbf{E}^{-1}\mathbf{h}(\overrightarrow{\mathbf{x}},\mathbf{t})$**.** For any time t, **U(x,t) = A**$\boldsymbol{\alpha}$(t). There are many hundreds of journal articles involving novel problems with themes and variations of RBFs that are too many to cite. The books by Fasshauer [15], Sarra and Kansa [16], and Chen *et al.* [17] provide an overview of the many applications that have been published such as ODES, PDEs, and IEs have been solved with compactly supported, poly-harmonic splines, and C^{∞} RBFs.

Because *both* spatial and temporal truncation errors propagate along characteristics in hyperbolic PDEs, it is important to minimize both types of errors. The traditional approach is to modify the physics to suite the traditional numerical schemes rather than modify the numerics to suite the physics. C^{∞} RBFs when combined with high order time marching schemes in a suitable moving frame, do provide both accurate time and space solutions. Consider the following PDE:

$$\partial\mathrm{U}^{\mathrm{k}}/\partial\mathrm{t} +\nabla \cdot \mathbf{F}^{\mathbf{k}} = \mathbf{J}^{\mathrm{k}}, \quad \text{rewriting} \tag{19}$$

$$\partial\mathrm{U}^{\mathrm{k}}/\partial\mathrm{t} +\nabla \cdot (\mathbf{F}^{\mathrm{k}}\text{-}\overrightarrow{\boldsymbol{\lambda}}_{\mathrm{i}}^{\mathrm{k}}\mathrm{U}^{\mathrm{k}})+\overrightarrow{\boldsymbol{\lambda}}_{\mathrm{i}}^{\mathrm{k}} \cdot \nabla\mathrm{U}^{\mathrm{k}} = \mathbf{J}^{\mathbf{k}}. \tag{20}$$

One can choose the local $\overrightarrow{\boldsymbol{\lambda}}_{\mathrm{i}}^{\mathrm{k}}$ such that $\nabla \cdot (\mathbf{F}\text{-}\overrightarrow{\boldsymbol{\lambda}}_{i}\mathrm{U}) = 0$ or appropriate linear combinations so that the total time derivative equals **J**. If *each* evaluation center in

$\Omega\backslash\partial\Omega$, has each velocity, $\overrightarrow{\boldsymbol{\lambda}}_i$=d$\overrightarrow{x}_i$/dt, then

$$\mathrm{dUk/dt} = \partial\mathrm{U^k}/\partial\mathrm{t} + \overrightarrow{\boldsymbol{\lambda}}_{\mathrm{i}}^{\mathrm{k}} \cdot \nabla\mathrm{U^k} = \mathbf{J}^{\mathrm{k}}, \text{ along each } \overrightarrow{\boldsymbol{\lambda}}_{\mathrm{i}}^{\mathrm{k}} = \mathrm{d}\overrightarrow{x}_i/\mathrm{dt}. \tag{21}$$

In this local moving frame where the space dependency is separable from the time dependency of the expansion coefficients:

$$\Phi\mathrm{d}\boldsymbol{\alpha}^k/\mathrm{dt} + [\overrightarrow{\boldsymbol{\lambda}}_{\mathrm{i}}^{\mathrm{k}} \cdot \nabla\Phi]\boldsymbol{\alpha}^k = \mathbf{J}^{\mathbf{k}}, \text{ along each } \overrightarrow{\boldsymbol{\lambda}}_{\mathrm{i}}^{\mathrm{k}} = \mathrm{d}\overrightarrow{x}_i/\mathrm{dt}. \tag{22}$$

At each $\overrightarrow{x}_i$, the expansion coefficient has an exact exponential matrix solution, see Kansa [18].

$$\alpha^{\mathrm{k}}(\mathrm{t}+\Delta t) = \mathrm{expm}(-\Delta\mathrm{t}\mathbf{H})\boldsymbol{\alpha}^k(\mathrm{t}) + \mathrm{expm}(\Delta\mathrm{t}\mathbf{H})\int_t^{t+\Delta t} \mathrm{J^k}(\overrightarrow{x}_i,\tau)\mathrm{expm}(-\tau\mathbf{H})\mathrm{d}\tau, \tag{23}$$

where

$$\mathbf{H} = \Phi^{-1}[\overrightarrow{\boldsymbol{\lambda}}_{\mathrm{i}}^{\mathrm{k}} \cdot \nabla\Phi]. \tag{24}$$

In an unpublished work, Kansa examined the self-similar solution of ideal gas rarefaction fans of finite width arising from ideal gas Riemann solvers. In the absence of any physical viscosity and interaction with walls, shocks, contact surfaces or other rarefaction waves, These rarefaction fans, either right or left facing, propagate indefinitely as self-similar fans. These fans remained self-similar because the maximum eigenvalue, max$\{$eig(expm(-ΔtH)$\} = 1$; this was observed for 10000 time steps. These results occurred at very large values of c_j^2 using extended precision; extended precision will be discussed later.

Shocks, contact surfaces and material interfaces are surfaces in $\Re^{n-1}$ that can be represented as a product of a Heaviside function in the local normal direction and a RBF in $\Re^{n-1}$ in the tangential directions (see [19]). A flame was modelled as a discontinuous surface in an infinitely periodic domain within an inviscid flow field with vortices on either side of the discontinuous flame front. The local burning speed is a prescribed quantity as well as the expansion ratio that is the ratio of the burnt gas density to the unburned gas density. The burned and unburned gases communicate with each other by the Rankine-Hugoniot jump conditions. At each instant of time, t, each interior point moves in a local Galilean frame such that each PDE is an exact differential. Note at $\mathrm{t} = 0$, the front is a vertical line. Later, due to the vortical flow, the flame front becomes significantly curved, and still later, the flame front becomes vertical. This pattern of the flame front curving and flattening continues indefinitely (see [19]).

Detecting discontinuities in a time dependent problem is relatively simple if data centers move. Riemann solvers give the speed and wave types after collisions with other objects. Consider an ideal gas in which a piston is accelerating. For subsonic flow, the pressure wave resembles a finite width sigmoid function. As the flow transitions to supersonic, the width tends to zero at which time and location a propagating Heaviside function can be inserted. An algorithm detecting discontinuities was developed by Jung and Durante [20] for scattered data RBF applications.

10 Ill-conditioning

Ill-conditioning occurs on finite precision machines, the numerical act of solving a system of equations, $\mathbf{A}\alpha = \mathbf{b}$, may lose up to m digits of accuracy. While there may be many instances in which the solution process has a small relative condition number, K_{rel}, the absolute condition number, K_{abs}, may be orders of magnitude larger. K_{abs} is a general estimate when the system will be unstable due to round-off errors. $\mathrm{K}_{abs} \risingdotseq$ 1/machine epsilon, and absolute numerical stability cannot be guaranteed mathematically. It is known that increasing c_j increases the convergence rate at the expense of increasing K_{abs}, Double precision has a machine epsilon of about 1e-16 and quadruple precision has a machine epsilon of 2e-34 (see Higham [21]).

The problem requiring solutions of ill-conditioned systems of equations is not unique to applications dealing with C^{∞} RBFs. In fact, if the number of equations becomes significantly large with traditional compactly supported methods such as FDM, FEM, and FVM, ill-conditioning will result. There is a vast network of literature describing methods to treat ill-conditioning for compactly supported FEM before ever being applied to d to C^{∞} RBFs. There are a number of proven approaches to ameliorate ill-conditioning, especially using combinations of approaches. Ill-conditioned systems, most often occurs when the rows or columns of the coefficient matrix are very similar, so after an arbitrary number of elementary row operations, the round-off errors render two or more matrix rows very similar. Carlson and Foley [22] found that if data centers were aligned along a preferred direction, such as data that is finely spaced in the x_1-direction and coarsely spaced in the x_2-direction, (tract data), the system of equations is very ill-conditioned. Surprisingly, allowing such data to have a randomized appearance improves the conditioning. Other techniques used are:

1. Scaling each row of the matrix by the maximum absolute value of the element, $max_j|\mathrm{A}_{i,j}|$.

2. Iterative refinement of the equation, $\mathbf{A}\boldsymbol{\alpha} = \mathbf{b}$, helps to attain more reliable expansion coefficients. Soleymani [23] noted that Krylov subspace and accelerated over-relaxation methods require a significant amount of time to obtain a reliable solution. He developed a 7th order correct approximation to $\mathbf{A}^{-1}$ and used 256 digit floating point arithmetic. This scheme was tested on the notoriously ill-conditioned Hilbert matrix. For an N×N Hilbert matrix of rank 15, K_{abs}= 1.5e21; for a rank 20 Hilbert matrix, $\mathrm{K}_{abs} = 6.2\mathrm{e}28$. After 50 iterations for the rank 15 Hilbert matrix, the residual norm is 5.7e-250, and for the rank 20 Hilbert matrix after 66 iterations, the residual norm is 3.6e-79.

3. In the effort to be able to treat more realistic and complex problems, fast multipole methods have been developed for MQ-PDEs (see Brown *et al.* [24]).

4. Ling *et al.* [25, 26] developed a preconditioner for elliptic PDEs in which MQ were used as the RBFs.

5. Ling and Schaback [27] developed a greedy algorithm (GA) linear system equation solver to find fast RBF-PDE solutions. A GA repeatedly executes a procedure that tries to maximize the return based upon examining local conditions,

with the expectation that the result will produce a desired outcome for the global problem. The GA may not completely solve the problem, or, if it produces a solution, it may not be the optimal one. Sometimes the GA yields very good (or even the best possible) results. In some cases such a strategy is guaranteed to offer optimal solutions, and other cases, the GA may provide a compromise that produces acceptable approximations.

6. Not all applications involving C^{∞} RBFs require full matrices; a rigorous interpretation of C^{∞} RBFs is that the set of linear equations *ALWAYS* must be full. This strict interpretation is unrealistic. Domain decomposition methods (DDM) may be non-overlapping or overlapping and are a type of preconditioner for large poorly conditioned systems. DDM generates a continuous set of data centers over the entire domain, then subdivides those data centers and uses a distributed-memory parallel technique to distribute the solution for each subdomain to a network of processors. This substantially increases simulation capacity. Domain decomposition is highly scalable to large numbers of processors and takes advantage of multi-threading within the subdomains to reduce solution times for individual subdomains. Various blending procedures such as the alternating Schwarz method acting upon the Dirichlet and Neumann conditions at the subdomain interfaces can be employed, see [28].

Rolland L. Hardy [11] who first discovered MQ used overlapping DDM in an RBF-MQ application; he reconstructed from a crude coarse picture of Abraham Lincoln into a easily recognizable one, using overlapping DDM and MQ. Quoting from his article, “Again, the so-called ‘global’ MQ method was reduced to a very local interpolation scheme of four picture elements involving a simple 4×4 matrix which was then used many times to cover the whole picture without inverting a 266×266 matrix all at once, see Fig. 11.”

Other authors have used overlapping and nonoverlapping in a variety of applications, even on parallel computers [29, 30]. With regard to C^{∞} RBFs, points rather distant from one another should have only minimal influence as compared to neighboring points. To determine this, Kansa and Hon [31] solved a 2D elliptic PDE on a unit square with non-overlapping DDM. After optimizing the shape parameters for the global problem, experiments considered deleting matrix elements if the normalized distance, r_{max}, between a data center and evaluation point were greater than a prescribed value of r_{max}. The system of equations for elliptic PDEs was sparse with a rather significant bandwidth. As the r_{max} shrank, there was no noticeable change in the error between the exact solution and the MQ solution until this distance became less than 1.4, after which the errors became increasingly worse. Unfortunately, further research into truncating the expansion for different applications was not performed.

7. One of the most common causes of ill-conditioned systems of equations arising from the application of C^{∞} RBFs to PDEs and IEs is *wasteful* spatial discretization. Purists wish to have fine uniform discretization for easier analysis, but a fine discretization (h-refinement) greatly increases K_{abs}. A preferred alternative is a local refinement of multi-layers to achieve h-refinement *only where it is needed.*

Multilevel sampling methods are efficient because they can be nested and implemented in parallel. Libre *et al.* [32, 33] constructed a multi-resolution prewavelet based adaptive refinement scheme for RBF approximations for nearly singular problems. If one examines the local length scale, ℓ, of a function, $\mathrm{U}(\overrightarrow{\mathbf{x}})$, a workable definition is:

$$\ell = |\mathrm{U}|/|\nabla \mathrm{U}|. \tag{25}$$

If $\ell \longrightarrow \infty$, U is well represented by a very coarse discretization; if $\ell \longrightarrow 0$, then a very fine discretization is required. As the length scale approaches zero, this indicates there is a transition to a Heaviside function in the normal propagation direction.

8. In the strong collocation form of PDEs, the interior and boundary problems have different operators acting upon them. Usually, as c increases, the L_2 errors decrease until it reaches a minimum, then begins to exhibit unstable behavior. It is in this case where extended precision is beneficial. Hon and Schaback [34] observed a rare case in which a combination of data centers and prescribed c value produced a very ill-conditioned, or a possible singular system, However, a slight increase or decrease in c drastically dropped K_{abs}; at the value of c that appears to be singular, no increase in precision would help.

11 The Galperin and Zheng global minima method

A general unifying approach is presented by Galperin and Zheng [35, 36] to solve PDEs, IEs, and IDEs as a global minimization problem in which the interior operator, acting upon the C^∞ RBFs is volume integrated over $\Omega\backslash\partial\Omega$ and the boundary operator, $\wp$, acting upon the C^∞ RBFs is integrated over each $\partial\Omega_k$. The functional, $\mathcal{F}$, to be minimized, over the parameter set Q is:

$$\min_{\mathrm{q}\in\mathrm{Q}} \mathcal{F} = \varpi\left|\int_{\Omega\backslash\partial\Omega}(\mathcal{L}\mathbf{U}\text{-}\mathbf{f})d\overrightarrow{\mathbf{x}}\right| + (1-\varpi)\left|\int_{\partial\Omega}(\wp\mathbf{U}-\mathbf{g})d\overrightarrow{\mathbf{x}}\right| \leq \eta \tag{26}$$

where Q is the set of free parameters, $\mathbf{Q} = \{\overrightarrow{\mathbf{x}}, \overrightarrow{\mathbf{y}}, \mathrm{c}_j^2, \alpha\}$, $\varpi < 1$ and η is a prescribed error criterion. Note that unlike the strong collocation method, there is a prescribed amount of "fuzziness" in which the solution trajectory is allowed to possess due to the integration process. The integrations can be performed globally without the need for tessellation as required by finite elements or the finite-element like meshless methods. The same advantages touted by FEM applies here. Integration increases the rate of convergence and the integration of a divergence or gradient is transformed by Gauss's theorem into difference in surface fluxes. The integration of the basis function, $\int \phi(\overrightarrow{\mathbf{x}} - \overrightarrow{\mathbf{y}}_{\mathbf{j}})d\overrightarrow{\mathbf{x}}$ produces a vector that is stored. Likewise, the integration of the operators, $\mathcal{L}$ and $\wp$, act upon the

RBFs first, then the results are integrated and stored:

$$\boldsymbol{\Psi} = \{\psi_1, \psi_2, \cdots, \psi_N\}, \text{ where each } \psi_{\mathrm{j}} = \int_{\Omega\backslash\partial\Omega} \mathcal{L}\phi(\overrightarrow{\mathbf{x}}\text{-}\overrightarrow{\mathbf{y}}_{\mathrm{j}})\mathrm{d}\overrightarrow{\mathbf{x}}, \tag{27}$$

$$\boldsymbol{\Theta} = \{\theta_1, \theta_2, \cdots, \theta_N\}, \text{ where each } \theta_{\mathrm{j}} = \int_{\partial\Omega} \wp\phi(\overrightarrow{\mathbf{x}}\text{-}\overrightarrow{\mathbf{y}}_{\mathrm{j}})\mathrm{d}\overrightarrow{\mathbf{x}}, \tag{28}$$

$$\mathrm{F} = \int_{\Omega\backslash\partial\Omega} f(\overrightarrow{\mathbf{x}})\mathrm{d}\overrightarrow{\mathbf{x}}, \text{ and } \mathrm{G} = \int_{\partial\Omega} g(\overrightarrow{\mathbf{x}})\mathrm{d}\overrightarrow{\mathbf{x}}. \tag{29}$$

Then, the vectors are used to form a specific value of $\mathcal{F}(\mathbf{q})$:

$$\min_{\mathrm{q}\in\mathrm{Q}} \mathcal{F}(\mathbf{q}) = \varpi|\boldsymbol{\Psi}\cdot\overrightarrow{\boldsymbol{\alpha}}_{PDE}\text{-F}|\text{+}(1-\varpi)|\ \boldsymbol{\Theta}\cdot\overrightarrow{\boldsymbol{\alpha}}_{PDE}\text{-G}|\ \leq \eta \tag{30}$$

With ill-posed problems, the amount of fuzziness should be increased. If one wishes to have a finite element analog, then the total integration is a sum over all local integrations that yields N equations in N unknowns. Nonlinear optimization problems may have multiple local and global maxima, minima, and saddle regions. Gradient based methods such as steepest descent, Newton–Raphson or combinations thereof may only lead to the nearest local or global minimum. Searching the complicated multi-dimensional landscape for global minima can be exceedingly time-consuming since such minima may be clustered closely of widely dispersed in parameter space. Although theoretically, the Galperin and Zheng procedure eliminates the need to solve sets of linear or nonlinear equations, global minimization requires a huge number of evaluations of the functional, $\mathcal{F}$. Fast reliable global minimization algorithms capable of dealing with at least 4N parameters are not yet perfected. One way to minimize the number of functional evaluations is reduce the total number of free parameters.

12 Implementation of the Galperin–Zheng method

Experience has shown that the set of data and evaluation centers, optimal shape parameters and expansion coefficients are inter-related. Changing either the location of the data centers and evaluation centers changes the magnitude of the parameter, $\gamma = \mathrm{c}/\mathrm{h}$, that, in turn, changes K_{abs} of the system of equations involved in determining the expansion coefficients. The PDE forcing functions over $\Omega\backslash\partial\Omega$ and boundary conditions on $\partial\Omega$ usually correlate with the loci of maxima, minima, and inflection points. By having a minimal number of data centers in such critical regions, a uniform dense covering is not necessary. One simple way to use this formulation is to choose $\{\overrightarrow{\mathbf{x}}\}$, $\{\overrightarrow{\mathbf{y}}\}$, and $\{\mathrm{c}_j^2\}$, solve for $\{\overrightarrow{\boldsymbol{\alpha}}_{PDE}\}$ in the strong formulation, and use $\{\overrightarrow{\boldsymbol{\alpha}}_{PDE}\}$ in the expression for $\mathcal{F}(\mathbf{q})$ repeatedly until a very or global minimum is calculated. This is simpler than solving a 4N global optimizer repeatedly.

Aside from the general guideline of Madych and Nelson [38] and Madych [39] showing that the ratio of $\gamma = \mathrm{c}/\mathrm{h}$ should be as large as possible, practical recipes for choosing either a uniform or variable shape distribution still require more development. Kansa [13, 14] found using the MQ RBF that, if the solutions are either monotonic increasing or decreasing, then the following power law works well:

$$\mathrm{c}_{\mathrm{j}}^2 = \mathrm{c}_{\mathrm{min}}^2 * (\mathrm{c}_{\mathrm{max}}^2/\mathrm{c}_{\mathrm{min}}^2)^{\mathrm{(j\text{-}1)/(N\text{-}1)}},\ \mathrm{j}=1,2,..,\mathrm{N}, \tag{31}$$

where $\mathrm{c}_{\mathrm{max}}^2$and $\mathrm{c}_{\mathrm{min}}^2$ are input estimates of the squares of the largest and smallest length scales. Wertz *et al.* [40] demonstrated that the $\{\mathrm{c}_{\mathrm{j}}^2\}_{\partial\Omega}$ associated with boundaries should be larger than those associated with the interior. Fedoseyev *et al.* [41] found convergence improvements by extending the interior discretization, $\Omega\backslash\partial\Omega$, slightly beyond the boundary, $\partial\Omega$.

The usual procedure is to choose a set of N $\{\vec{x}\}$ and $\{\vec{y}\}$ and either a single shape parameter or a distribution of shape parameters, then solve an N×N set of equations to find the expansion coefficients. To overcome the limitations of double precision, the MATLAB compatible multi-precision package was obtained from *www.advanpix.com.* To improve the convergence rate, γ should become very large. This can be accomplished in two ways: (1) The h-scheme (spatial refinement) increases the number, N_h, of data centers, but a relatively small $\langle \mathrm{c}_j^2 \rangle$. (2) The c-scheme increases c, but requires a significantly smaller of data centers, N_c, where $\mathrm{N}_c \ll \mathrm{N}_h$; the c-scheme requires $\mathrm{O}(\mathrm{N}_c^3) \ll \mathrm{O}(\mathrm{N}_h^3)$ operations. The preferred approach here is the c-scheme.

Huang *et al.* [42] compared double and quadruple precision arithmetic for the c-scheme and h-scheme as well as with FEM. For a fixed c and h on their computer,

$$t_{CPU}^{quad} = 40 t_{CPU}^{double}. \tag{32}$$

But the

$$t_{CPU}^{quad}(\text{c-scheme}) = (1/565) t_{CPU}^{double}(\text{h-scheme}). \tag{33}$$

They demonstrated that they achieved high accuracy and efficiency with the c-scheme.

Assume that in an initial mesh, FEM can solve the PDE to an accuracy of 1%. Using a quadratic element or central difference, the error estimate is h^2. To reach an accuracy of 10^{-16}, h needs to be refined 10^7 fold. per dimension. In a 3D problem, this means 10^{21} fold more degrees of freedom. The full matrix is of the size 10^{42}. The effort of solution could be 10^{63} fold. If the original CPU is 0.01 sec, Huang *et al.* [42] concluded that to achieve an accuracy of 10^{-16}, the required time is 10^{54} years. For many people who are totally accustomed to FDM, FEM, or FVM, the h-scheme is totally natural and unlearning this habit is very difficult.

For purposes of efficiency, the c-scheme is superior and preferable over the h-scheme, see Huang *et al.* [42] and Cheng [43]. The disadvantage of either the h-scheme or the c-scheme is that as $\xi \to \infty$, the system of linear equations becomes very ill-conditioned and subject to extreme round-off errors on computers with limited arithmetic precision. Since then, new multi-precision software has been developed with improved functionality and performance allowing efficient handling of such issues even for large matrices, see Advanpix Multiprecision

Computing Toolbox (MCT) [44]. The MCT provides wide range of numerical analysis routines implemented with arbitrary precision support ranging from elementary arithmetic operations [45, 46] to advanced solvers for ill-conditioned linear systems and eigenvalue problems [47]. The MCT is heavily optimized for modern multi-core parallel architectures and orders of magnitude faster than Maple, Mathematica, and MATLABs Variable-Precision Arithmetic packages [44]. In contrast to legacy libraries in [48] used in [42], the MCT provides a quadruple precision mode that is fully compatible with IEEE 754-2008 standard for floating-point computations [49] with extended exponent range and accurate approximation of small values.

13 Numerical results of the Galperin–Zheng formulation

There are several steps involved in calculating the Galperin–Zheng global minimization. Firstly, the data centers are tensor product meshes; in each coordinate direction, the well-known geometric progression of increasing finer data centers were generated as x_1 or x_2 approaches unity. The boundary itself is defined by the lines: $(0, 0)$, $(1, 0)$, $(1, 1)$, and $(0, 1)$. Following Fedoseyev *et al.* [41], extra data centers are placed slightly beyond the boundary locus by adding 1-D randomly generated numbers multiplied by 2.03e-3 in the x_1-direction and 2.51e-3 in the x_2-direction. There are N_{x1} points along the x_1-axis and N_{x2} points along the x_2-axis.

The test problem is the Poisson equation over a unit 2D square:

$$\nabla^2 U(x_1, x_2)= (a^2+b^2)\exp(a \cdot x_1+b \cdot x_2) \text{ over } \Omega, \tag{34}$$

(a and b are parameters that will be varied) with the following Dirichlet conditions on

$$U(x_1)= \exp(a \cdot x_1) \text{ on } \partial\Omega(x_2=0) \text{ and,} U(x_2)= \exp(b \cdot x_2) \text{ on } \partial\Omega(x_1=0), \tag{35}$$

and Neumann conditions on the opposite boundaries:

$$\partial U/\partial x_1= a \cdot \exp(a \cdot x_1+b) \text{ on } \partial\Omega(x_2=1) \text{ and,}$$
$$\partial U/\partial x_2= b \cdot \exp(a+b \cdot x_2) \text{ on } \partial\Omega(x_1=1). \tag{36}$$

The exact solution is:

$$U(x_1,x_2)= \exp(a \cdot x_1+b \cdot x_2). \tag{37}$$

The test domain was chosen over a unit square, and various exponential parameters for the forcing function, f, were chosen: The parameters, a and b, were specified as $a = 2$ and $b = 3$. The maximum values of U at $x_1 = x_2 = 1$, is, $U = \exp(5) = 148.4132$.

Since most scientists and engineers are familiar with finite difference, element, or volume methods, the customary procedure is to use extremely fine meshing.

This approach is workable for 2 and 3-dimensional problems, but impractical for 6D Boltzmann equations due to the curse of dimensionality. To minimize the number of data centers, a geometric progression of increasingly finer discretization was used, and the origin was appended in the construction of the tensor product mesh. In the paper of Fedoseyev *et al.* [41], the computational domain, Ω was extended slightly beyond the boundaries, $\partial\Omega_k$. The domain was extended to 1.022 in the x_1 direction and 1.02412 in the x_2 direction.

The highlights of the parameter tests are summarized below:

- The shape parameters multipliers, χ_{Dir} and χ_{Neu}, respectively, corresponding to either the two Dirichlet and two Neumann boundary data centers were stored as adjustable input factors, see Wertz[40].
- While the power law recipe seems to work for monotonic functions, and yields the desired wavelet relations, see [13, 14]; this power-law recipe is too simplistic and is used only as a temporary measure.
- Since the maximum value of U occurs at $\mathrm{x}_1 = \mathrm{x}_2 = 1$, the geometric progression algorithm produced successively finer discretization in the x_1 and x_2 directions. The first point was fixed at 0.16 for x_1 and 0.18 for x_2.
- Afterwards, the minimum values of x_1 and x_2 was -0.02 and the maximum values of x_1 and x_2 was 1.025 were appended to the tensor product 2D mesh.
- The set of expansion coefficients, $\{\overrightarrow{\alpha}_{PDE}\}$, was obtained by solving the point collocation for the PDE and boundary conditions.
- Using the set of expansion coefficients, $\{\overrightarrow{\alpha}_{PDE}\}$, and the integrated basis functions over $\Omega\backslash\partial\Omega$ and $\partial\Omega$, the global $\mathrm{L}_\infty^{strong}$error was calculated for various choices of evaluation centers, data centers, $\{\mathrm{c}_j^2\}$, other tunable parameters, and number of digits of precision.
- The data centers are tensor product meshes; in each coordinate direction, the well-known geometric progression of increasing finer data centers were generated as x_1 or x_2 approaches unity. Extra data centers are placed slightly beyond the boundary locus by adding 1-D randomly generated numbers multiplied by 2.03e-3 in the x_1-direction and 2.51e-3 in the x_2-direction.

14 Summary of results

The L_∞ errors obtained from solving the set PDE expansion coefficients, $\{\overrightarrow{\alpha}_{\mathbf{PDE}}\}$, calculating $\mathrm{U}_{\mathrm{PDE}}$ with these expansion coefficients relative to the exact solution ranged from 0.469 to 2e-28 with the total number of points ranging from 12 to 110.

To analyze numerically the question whether the weak formulation is viable, several steps are required.

1. Choose the distribution of $\{\overrightarrow{\mathbf{x}}\}$ and $\{\overrightarrow{\mathbf{y}}\}$ and the extent that the interior points extend beyond the boundary.
2. Choose a distribution of shape parameters and the multiplicative factors for the Neumann and Dirichlet shape parameters.

Table 1: RMS errors of the Galperin–Zheng formulation: first term of geometric progression: $x_1 = 0.27$, $x_2 = 0.277$ and exponential factors: $a = 2$, $b = 3$ held constant.

	n_{digits}	c^2_{max}	c^2_{min}	χ_{Dir}	χ_{Neu}	K^{strong}_{abs}	L^{strong}_{∞}	L^{weak}_{∞}
N	32	1.021	0.9149	0.5	0.85	3.2210^{13}	1.8910^{-25}	0.0033
56	128	1.021	0.9149	0.5	0.85	3.6410^{13}	1.8910^{-25}	0.0022
56	512	1.0e5	0.0019	0.0007	79.31	3.6410^{13}	2.4410^{-25}	0.0022
56	1024	1.0e5	0.0019	0.0007	90.0	3.2210^{13}	2.9110^{-25}	0.0011
63	512	1.021	0.9149	1	1	3.6410^{8}	3.4910^{-30}	0.0003
90	32	1.021	0.9149	0.5	0.85	1.4610^{18}	1.2710^{-22}	0.0060
90	128	1.021	0.9149	0.5	0.85	2.4810^{18}	1.3610^{-22}	0.0002

3. Construct the MQ basis functions and the integrated 2D RBFs for the interior and the 1D integrated Neumann and Dirichlet RBFs, and choose the weighting factor, ϖ, for the interior and boundary problem.
4. Construct the strong form system of equation and find $\{\vec{\boldsymbol{\alpha}}_{\mathbf{PDE}}\}$.
5. Substitute $\{\vec{\boldsymbol{\alpha}}_{\mathbf{PDE}}\}$ into the strong form and reject if the L^{strong}_{∞} error exceeds 1e-6.
6. Continue varying the input parameters until the deepest L^{weak}_{∞} error, $\mathcal{F}(q)$, is achieved.

$$\min_{q \in Q} \mathcal{F}(\mathbf{q}) = \varpi|\,\boldsymbol{\Psi}\cdot\,\vec{\boldsymbol{\alpha}}_{\text{PDE}} - F| + (1-\varpi)|\boldsymbol{\Theta}\cdot\vec{\boldsymbol{\alpha}}_{\text{PDE}} - G| \le \eta. \qquad (38)$$

In the following table, the constants, a and b are $a = 2$ and $b = 3$, were used for the forcing term, and x_1 and x_2 that are the first terms, respectively for the geometric progression of the x_1 and x_2 coordinates. Table 1 presents the number of digits used, c^2_{max} and c^2_{min} are the maximum and minimum parameters for the power law shape parameter distribution, χ_{Dir} and $\chi_{Neu}$are the multiplicative parameters to modify the shape parameters associated with the Dirichlet and Neumann boundary coordinates, respectively, K^{strong}_{abs} is the condition number associated with the strong form of the PDE matrix, L^{strong}_{∞} are the strong form L_{∞} errors and L^{weak}_{∞} are the weak form L_{∞} errors.

Table 1 shows the results of increasing the number data centers for the Galperin–Zheng weak formulation. During previous testing, and using an increasing number of digits of precision, the values of c^2_{min} and c^2_{max} were pushed., fixing the set, $\{c^2_j\}$, using the set $\{\alpha\}$ from the solution of the point collocation scheme, and fixing the starting points of the geometric progression for the tensor product meshes.

Note that increasing the number of data centers, h-refinement, while fixing the set, $\{c^2_j\}$ does show successively smaller RMS errors, but it is questionable whether the extra effort is warranted. The point of this exercise is to provide a good starting point for the global optimization process in which all free parameters are varied to find very deep global minima instead of a local minimum. In summary, solving the PDE expansion coefficients as a series of strong form collocation problems and inserting the expansion coefficients into the expression for $\mathcal{F}(\mathrm{q})$ is a reasonable choice to find the set of deep minima and perhaps the global minimum.

15 Challenges

Two of the important unresolved issues are:

1. Implementation on massively parallel super computers

Implementation of multi-dimensional static and time dependent PDEs and IEs on truly massively parallel computers using domain decomposition techniques needs to be a very high priority to demonstrate the viability of C^∞ RBFs. Such problems can be the multi-dimensional PDEs and IEs such as the 6-D generalized Boltzmann equations, molecular quantum mechanics, biological systems, American and European option markets, special and general relativity, string theory, organized and chaotic behavior, and likewise the search for designer drugs and treatments for human diseases. Of the many multi-dimensional algorithms in use, C^∞ RBFs are the most viable. The question that must be firmly addressed is whether full sets of equations are absolutely essential, or as R.L. Hardy demonstrated strict globality is not needed.

The modified Gram–Schmidt algorithm also known as the QR method can be used to construct orthonormal C^∞ RBFs. The advantage of orthogonal basis functions is that the coefficients are uncorrelated, thereby minimizing approximation error and the sensitivity to round-off errors.

Constructing orthonormal was used in Galperin *et al.* [50] but, because of the available computers and software round-off errors eventually contaminated the set of RBFs, making them only approximately orthonormal. However, with the use of multi-precision arithmetic, orthonormality can be extended and whether orthonormality can be used advantageously. Recent experiments show a large number of C^∞ RBFs are indeed orthonormal. Orthonormality should help in the optimization of the set of shape parameters for various PDE and IE applications.

The first, albeit primitive application was the solution of a 4D inviscid set of Euler equations using MQ-RBFs and overlapping DDM. Because of the curse of dimensionality, any regular meshing of a N-D problem encounters the dreaded curse of dimensionality. Kansa and Geiser [37] found a scattered data distribution can cover the 4D domain fairly well, but additional placement of data centers where most needed in an adaptive manner will be required in multi-dimensional regions where fine resolution is required. Note that with Franke's [51] 2D test functions, having only a few points in rapidly changing regions performs well; dense coverings are overkill. In addition, serious multi-dimensional problems will need to be solved on exascale parallel computers. As of November 2014, China's Tianhe-2 supercomputer is the fastest in the world at 33.86 quadrillion floating point operations per second. Domain decomposition implemented in multi-dimensions will need to be studied well to achieve load balancing. Because of the curse of dimensionality, it is vital to keep the total number of data centers to a minimum because the quantum mechanical calculation of the electronic structure of a rotating, vibrating, translating water molecule is a 9D problem. If there are 10 data centers per dimension, 10^9 data centers may still be a sparsely populated problem. The unexplored question is for multi-dimensional problems is whether sacrificing exponential spectral convergence of C^∞ RBFs for sparsity worthwhile? How can fast mutlipole expansions best be implemented? How can exponential spectral convergence be maintained with domain decomposition? What trade-offs are necessary to implement even a simple 6D Boltzmann equation. Is the only alternative for multi-dimensional problems compactly supported RBFs or finite differences with C^∞ RBFs?

2. Optimal specification of the C^∞ RBF shape parameter distributions

Meshless RBFs appear currently to be the best method of solving multi-dimensional problems and the RBF community needs to pursue research and development in this important area.

References

[1] Ferguson, J.C., *J. ACM*, v. 11(2), pp. 221–228, 1964.

[2] Eymard, R., Gallouët, T.R., Herbin, R. *The finite volume method Handbook of Numerical Analysis*, v. VII, pp. 713–1020. Editors: P.G. Ciarlet and J.L. Lions, 2000.

[3] Reddy, J.N., *An introduction to Finite Elements*, 3rd ed., McGraw-Hill, ISBN 9780071267618, 2005.

[4] Harris, J.W., Stocker, H., *Handbook of Mathematics and Computational Science*, Springer Verlag, New York, 1998.

[5] Chui, C.K., *An Introduction to Wavelets*. San Diego: Academic Press. ISBN 0-12-174584-8, 1992.

[6] Cheng, A.H.-D., Cheng, D.T., *Eng. Anal. Bound. Elem.*, v. 29, pp. 268–302, 2005.

[7] Gottlieb, D., Orzag, S., *Numerical Analysis of Spectral Methods: Theory andApplications*, SIAM, Philadelphia, PA, 1977.

[8] Wendland, H. *Adv. Comput. Math.*, v. 4, pp. 389–396, 1995.

[9] Wu, Z. *Adv. Comput. Math.*, v. 4, pp. 283–292, 1995.

[10] Hardy, R.L., *J. Geophys.Res*. v. 76, pp. 1905–1915, 1971.

[11] Hardy, R.L., *Comput. Math. Appl.* v. 19(8-9), pp. 163–208, 1990.

[12] Luh, L.-T., *Inter. J. Numer. Methods. Appl.* v. 1(2), pp.155–74, 2009.

[13] Kansa, E.J., *Comput. Math. Appl.*, v. 19(6-8), pp. 127–145, 1990.

[14] Kansa, E.J., *Comput. Math. Appl.*, v. 19(6-8), pp. 147–161, 1990.

[15] Fasshauer, G.E., *Meshfree Aproximation Methods with Matlab,* Interdisciplinary Sciences, v. 6, World Scientific, Hackensack, NJ, 2007.

[16] Sarra, S.A, Kansa, E.J., *Multiquadric Radial Basis Function Approximation Methods for the Numerical Solution of Partial Differential Equations*, Adv. Comput. Mech, v. 2, ISSN: 1940-5820, Techscience, Atlanta, 2009.

[17] Chen, W., Fu, Z.-J., Chen, *Recent Advances in Radial Basis Function Collocation*, Springer Verlag, Heidelberg, 2014.

[18] Kansa, E.J., *Comput Math. Applic.*, v. 43(3-5), pp. 501–512, 2002.

[19] Kansa, E.J., Aldredge, R.C., Ling, L., *Eng. Anal. Bound. Elem.*, v. 33, pp. 940–950, 2009.

[20] Jung, J.-H., Durante, V.R., *Appl. Num. Math.*, v. 59, pp. 1449–1466, 2009.

[21] Higham, N., *Accuracy and Stability of Numerical Algorithms* (2nd ed), SIAM, pp. 37, 2002.

[22] Carlson, R.E., Foley, T.A., *Comput.Math. Appl.*, v. 24, pp. 27–34, 1992.

[23] Soleymani, F., *Opuscula Math.* v. 33, pp. 337–344, 2013.

[24] Brown, D., Ling, L., Kansa, E.J., Levesley,J., *Eng. Anal. Bound. Elem.*, v. 29, pp. 343–353, 2005.

[25] Ling, L., Kansa, E.J. *Adv. Comput. Math.* v. 23, pp. 31–54, 2005.

[26] Ling, L., Kansa, E.J., *Math.& Comput. Model.*, v. 40, pp. 1413–1427, 2004.

[27] Ling, L., Schaback, R., *Int. J. Num. Meth. Eng.* v. 80, pp. 1623–1639, 2009.

[28] Smith, B., Bjorstad, P., Gropp, W., *Domain Decomposition: Parallel, Multilevel Methods for Elliptic Partial Differential Equations*, Cambridge University Press, 1996.

[29] Zhou, X., Hon, Y.-C., Li, J. *Appl. Num. Math.*, v. 44, pp. 241–255, 2003.

[30] Ingber, M.A., Chen, C.S., Tanski, J.A., *Int. J. Num. Meth. Eng.*, v. 60, pp. 2183–2201, 2004.

[31] Kansa, E.J., Hon, Y.-C., *Comput. Math. Applic.*, v. 39(7/8), pp. 123–137, 2000.

[32] Libre, N.A. Emdadia, A., Kansa, E.J., Shekarchi, M., Rahimian, M., *Eng. Anal. Bound. Elem.*, v. 33, pp. 901–914, 2009.

[33] Libre, N.A., Emdadia, A., Kansa, E.J., Shekarchi, M., Rahimian, M., *CMES: Comput. Model Eng & Sci.*, v. 38, pp. 263–284, 2009.

[34] Hon, Y.C., Schaback, R., *Appl. Math. Comput.*, v. 119, pp. 177–186, 2001.

[35] Galperin, E.A., Zheng, Q., *Comput. Math. Applic.*, v. 25(10/11), pp. 103–118, 1993.

[36] Galperin, E.A., Zheng, Q., *Comput. Math. Applic.*, v. 25(10/11), pp. 119–124, 1993.

[37] Kansa, E.J., Geiser. J., *Eng. Anal. Bound. Elem.*, v. 3, pp. 637–645, 2013.

[38] Madych, W.R, Nelson S.A., *Math. Comp.*, v. 54, pp. 211–30, 1990.

[39] Madych, W.R, *Comput. Math. Applic.*, v. 24(12), pp. 121–138, 1992.

[40] Wertz, J., Kansa, E.J., Ling, L., *Comput. Math. Applic.*, v. 51(8), pp. 1335–1348, 2006.

[41] Fedoseyev, A.I., Friedman, M.J., Kansa, E.J., *Comput. Math. Appl.*, v. 43(3-5), pp. 491–500, 2002.

[42] Huang, C.-S., Lee, C.-F., Cheng, A.H.D., *Eng. Anal. Bound. Elem.*, v. 31, pp. 615–623, 2007.

[43] Cheng, A.H.D., *Eng. Anal. Bound. Elem.*, v. 36, pp. 220–39, 2012.

[44] Multiprecision Computing Toolbox for MATLAB, Advanpix LLC., Yokohama, Japan, *http://www.advanpix.com*

[45] Brent, R., Zimmermann, P., *Modern Computer Arithmetic*, Cambridge, *Monographs on Computational and Applied Mathematics*, 2010.

[46] Fousse, L., Hanrot, G., Lefavre, V., Paulissier, P., Zimmermann, P., *ACM Trans. Math. Softw.*, v. 33, 2, Article 13, 2007.

[47] Golub, G.H., Van Loan, C.F., *Matrix Computations*, Johns Hopkins University Press; 4th ed., 2012.

[48] Bailey, D.H., Hida, Y, Jeyabalan, K., Li, X.S., Thompson, B. "High precision software directory", *http://crd.lbl.gov/dhbailey/mpdist/*

[49] IEEE Computer Society, "*IEEE Standard for Floating-Point Arithmetic*", August 29, 2008.

[50] Galperin, E.A., Kansa, E.J., Makroglou, A, Nelson, S.A., *J. Comput. Appl. Math.*, v. 115, pp. 193–211, 2000.

[51] Franke, R., *Math. Comput.* v. 38(157), pp. 181–200 (1982).