Copyright © 2010 ICCES ICCES, vol.14, no.2, pp.51-56

A simple and effective preconditioner for integrated-RBF-based Cartesian-grid schemes

N. Mai-Duy[1] **and T. Tran-Cong**[1]

Summary

This paper presents a preconditioning scheme to improve the condition number of integrated radial-basis-function (RBF) matrices in solving large-scale 2D elliptic problems. The problem domain is discretised using a Cartesian grid, over which integrated RBF networks are employed to represent the field variable. The present preconditioner is constructed from 1D integrated RBF networks along grid lines. Test problems defined on rectangular and non-rectangular domains are employed to study the performance of the scheme.

Keywords: preconditioner, integrated radial basis functions, Cartesian grids.

Introduction

Integrated RBF networks (IRBFNs), which have the ability to avoid the reduction in convergence rate caused by differentiation, were successfully developed for the solution of partial differential equations (PDEs) [[Mai-Duy and Tran-Cong(2005)]; [Mai-Duy and Tanner(2005)]; [Mai-Duy and Tran-Cong(2006)]]. IRBFN-based methods were shown to work well with both scattered and gridded data points. This paper is concerned with a preconditioning scheme designed for the latter in solving large-scale problems such as porous rocks under high hydraulic pressure.

Integrated radial-basis-function networks incorporating Cartesian grids

RBFNs allow a conversion of a function f from a low- to a high-dimensional space in which the function can be expressed as a linear combination of RBFs

$$f(\mathbf{x}) = \sum_{i=1}^{N} w^{(i)} G^{(i)}(\mathbf{x}), \tag{1}$$

where the superscript (i) is the summation index, $\mathbf{x}$ the input vector, N the number of RBFs, $\{w^{(i)}\}_{i=1}^{N}$ the set of weights to be found, and $\{G^{(i)}(\mathbf{x})\}_{i=1}^{N}$ the set of RBFs.

This paper is concerned with second-order differential problems in two dimensions. The integral approach uses RBFNs (1) to represent the second-order

[1] Computational Engineering and Science Research Centre, Faculty of Engineering and Surveying, University of Southern Queensland, Toowoomba, QLD 4350, Australia

derivatives of the field variable u in a given PDE. Approximate expressions for the first-order derivatives and the variable itself are then obtained through integration as

$$\frac{\partial^2 u(\mathbf{x})}{\partial x_j^2} = \sum_{i=1}^{N} w^{(i)} G^{(i)}(\mathbf{x}), \tag{2}$$

$$\frac{\partial u(\mathbf{x})}{\partial x_j} = \sum_{i=1}^{N} w^{(i)} H^{(i)}(\mathbf{x}) + C_1(x_k), \tag{3}$$

$$u(\mathbf{x}) = \sum_{i=1}^{N} w^{(i)} \overline{H}^{(i)}(\mathbf{x}) + x_j C_1(x_k) + C_2(x_k), \tag{4}$$

where $C_1(x_k)$ and $C_2(x_k)$ are the constants of integration which are univariate functions of the variable other than x_j (i.e. x_k with $k \neq j$); $H^{(i)}(\mathbf{x}) = \int G^{(i)}(\mathbf{x}) dx_j$ and $\overline{H}^{(i)}(\mathbf{x}) = \int H^{(i)}(\mathbf{x}) dx_j$.

Using IRBFNs to represent the variations of the constants of integration and then expressing them in terms of the nodal values of C_1 and C_2, one has

$$\frac{\partial u(\mathbf{x})}{\partial x_j} = \sum_{i=1}^{N} w^{(i)} H^{(i)}(\mathbf{x}) + \sum_{i=1}^{N_k} P^{(i)}(x_k) C_1^{(i)}, \tag{5}$$

$$u(\mathbf{x}) = \sum_{i=1}^{N} w^{(i)} \overline{H}^{(i)}(\mathbf{x}) + \sum_{i=1}^{N_k} x_j P^{(i)}(x_k) C_1^{(i)} + \sum_{i=1}^{N_k} P^{(i)}(x_k) C_2^{(i)}. \tag{6}$$

For convenience of presentation, expressions (2), (5) and (6) can be rewritten as

$$\frac{\partial^2 u(\mathbf{x})}{\partial x_j^2} = \sum_{i=1}^{N+2N_k} w^{(i)} G^{(i)}(\mathbf{x}), \tag{7}$$

$$\frac{\partial u(\mathbf{x})}{\partial x_j} = \sum_{i=1}^{N+2N_k} w^{(i)} H^{(i)}(\mathbf{x}), \tag{8}$$

$$u(\mathbf{x}) = \sum_{i=1}^{N+2N_k} w^{(i)} \overline{H}^{(i)}(\mathbf{x}), \tag{9}$$

where

$$\{G^{(i)}(\mathbf{x})\}_{i=N+1}^{N+2N_k} \equiv \{0\}_{i=1}^{2N_k},$$

$$\{H^{(i)}(\mathbf{x})\}_{i=N+1}^{N+N_k} \equiv \{P^{(i)}(x_k)\}_{i=1}^{N_k}, \qquad \{H^{(i)}(\mathbf{x})\}_{i=N+N_k+1}^{N+2N_k} \equiv \{0\}_{i=1}^{N_k},$$

$$\{\overline{H}^{(i)}(\mathbf{x})\}_{i=N+1}^{N+N_k} \equiv \{x_j P^{(i)}(x_k)\}_{i=1}^{N_k}, \qquad \{\overline{H}^{(i)}(\mathbf{x})\}_{i=N+N_k+1}^{N+2N_k} \equiv \{P^{(i)}(x_k)\}_{i=1}^{N_k},$$

$$\{w^{(i)}\}_{i=N+1}^{N+N_k} \equiv \{C_1^{(i)}\}_{i=1}^{N_k}, \text{ and } \{w^{(i)}\}_{i=N+N_k+1}^{N+2N_k} \equiv \{C_2^{(i)}\}_{i=1}^{N_k}.$$

We seek an approximate solution in terms of nodal values of the field variable. To do so, multiple spaces of the network weights will be transformed into the physical space. Collocating (9) at the nodal points associated with the x_j grid lines, $\left\{\mathbf{x}^{(i)}\right\}_{i=1}^{N}$, leads to

$$\widetilde{\mathscr{T}}\begin{pmatrix}\widetilde{w}\\ \widehat{C}_1\\ \widehat{C}_2\end{pmatrix}=\widetilde{u}, \tag{10}$$

where $\widetilde{\mathscr{T}}$ is a $N\times(N+2N_k)$ matrix, $\widetilde{w}=\left(w^{(1)},\cdots,w^{(N)}\right)^T$, $\widehat{C}_1=\left(C_1^{(1)},\cdots,C_1^{(N_k)}\right)^T$, $\widehat{C}_2=\left(C_2^{(1)},\cdots,C_2^{(N_k)}\right)^T$, and $\widetilde{u}=\left(u(\mathbf{x}^{(1)}),\cdots,u(\mathbf{x}^{(N)})\right)^T$. The transformation matrix $\widetilde{\mathscr{T}}$ has the entries $\widetilde{\mathscr{T}_{li}}=\overline{H}^{(i)}(\mathbf{x}^{(l)})$ for $1\leq l\leq N$ and $1\leq i\leq(N+2N_k)$. It is noted that at a grid node $P^{(i)}(x_k^{(j)})$ is equal to 0 if $i\neq j$ and 1 if $i=j$.

Solving (10) for the coefficient vector yields

$$\begin{pmatrix}\widetilde{w}\\ \widehat{C}_1\\ \widehat{C}_2\end{pmatrix}=\widetilde{\mathscr{T}}^{+}\widetilde{u}, \tag{11}$$

where $\widetilde{\mathscr{T}}^{+}$ is the generalised inverse of $\widetilde{\mathscr{T}}$.

The values of first- and second-order derivatives of u at the nodal points associated with the x_j grid lines can then be computed in terms of nodal variable values as

$$\widetilde{\frac{\partial u}{\partial x_j}}=\widetilde{\mathscr{H}}\widetilde{\mathscr{T}}^{+}\widetilde{u}, \tag{12}$$

$$\widetilde{\frac{\partial^2 u}{\partial x_j^2}}=\widetilde{\mathscr{G}}\widetilde{\mathscr{T}}^{+}\widetilde{u}, \tag{13}$$

where $\widetilde{\mathscr{H}}$ and $\widetilde{\mathscr{G}}$ are $N\times(N+2N_k)$ matrices, derived from (8) and (7), respectively. Their corresponding entries are $\widetilde{\mathscr{H}_{li}}=H^{(i)}(\mathbf{x}^{(l)})$ and $\widetilde{\mathscr{G}_{li}}=G^{(i)}(\mathbf{x}^{(l)})$ for $1\leq l\leq N$ and $1\leq i\leq(N+2N_k)$.

Expressions (12) and (13) can be rewritten in compact form

$$\widetilde{\frac{\partial u}{\partial x_j}}=\widetilde{\mathscr{D}}_j'\widetilde{u}, \tag{14}$$

$$\widetilde{\frac{\partial^2 u}{\partial x_j^2}}=\widetilde{\mathscr{D}}_j''\widetilde{u}, \tag{15}$$

where $\widetilde{\mathscr{D}}'_j = \widetilde{\mathscr{H}}\widetilde{\mathscr{T}}^+$ and $\widetilde{\mathscr{D}}''_j = \widetilde{\mathscr{G}}\widetilde{\mathscr{T}}^+$ are the first and second-order differentiation matrices in the physical space.

Consider a Poisson equation $\nabla^2 u = b$ with Dirichlet boundary conditions. Using point collocation, it can be transformed into

$$\widetilde{\mathscr{A}}\widetilde{u}_{(\theta)} = \left(\widetilde{\mathscr{D}}''_{1(\eta,\theta)} + \widetilde{\mathscr{D}}''_{2(\eta,\theta)}\right)\widetilde{u}_{(\theta)} = \widetilde{b}_{(\eta)}, \qquad (16)$$

where $\widetilde{\mathscr{A}}$ is the system matrix, and η and θ the two sets of indices representing the interior points. The integral solution procedure involves computing the transformation matrix $\widetilde{\mathscr{T}}$ and the system matrix $\widetilde{\mathscr{A}}$. From a computational point of view, it is desirable to have $\widetilde{\mathscr{T}}$ and $\widetilde{\mathscr{A}}$ with low condition numbers.

Present preconditioning scheme

Consider the transformation system (10). The numerical stability of this system is dependent on the condition number of $\widetilde{\mathscr{T}}$. In the case that $\widetilde{\mathscr{T}}$ is ill-conditioned, special treatments are required. Here, we adopt a preconditioning approach. Both sides of (10) are multiplied by a matrix, denoted by $\widetilde{\widetilde{\mathscr{B}}}$, that is close to the inverse of $\widetilde{\mathscr{T}}$.

We propose the use of 1D-IRBFNs to construct the preconditioner $\widetilde{\widetilde{\mathscr{B}}}$. For 1D-IRBFNs, the approximations are constructed "locally" on each grid line. On a grid line that is parallel to the x_j axis, the field variable u is sought in the form

$$u(x_j) = \sum_{i=1}^{M} w^{(i)}\overline{h}^{(i)}(x_j) + x_j c_1 + c_2, \qquad (17)$$

where M is the number of RBF centres (interior and boundary points) on the grid line ($M = N_j$ for a rectangular domain). It can be seen that the number of RBFs used in (17) is much less than that in (2) (i.e. $M \ll N$). One can describe the transformation system for the 1D case as

$$\widehat{\mathscr{T}}\begin{pmatrix}\widehat{w}\\ c_1\\ c_2\end{pmatrix} = \widehat{u}, \qquad (18)$$

or

$$\begin{pmatrix}\widehat{w}\\ c_1\\ c_2\end{pmatrix} = \widehat{\mathscr{T}}^+\widehat{u}, \qquad (19)$$

where $\widehat{\mathscr{T}}^+$ is the generalised inverse of dimensions $(M+2)\times M$, and $\widehat{w}$ and $\widehat{u}$ the vectors of length M. The first M rows of $\widehat{\mathscr{T}}^+$ are associated with the values of w

at the grid points and we use this sub-matrix to construct the preconditioner $\widetilde{\mathscr{B}}$. In the case of rectangular domains, the assembly process can be simply carried out by means of Kronecker tensor products. Assume that the grid node is numbered from bottom to top and from left to right. The preconditioner will take the form

$$\widetilde{\mathscr{B}} = \widehat{\mathscr{T}}(1:N_j,:) \otimes \mathbf{1}, \tag{20}$$

for $x_j \equiv x_1$, and

$$\widetilde{\mathscr{B}} = \mathbf{1} \otimes \widehat{\mathscr{T}}(1:N_j,:), \tag{21}$$

for $x_j \equiv x_2$. In (20) and (21), $\mathbf{1}$ represents a unit matrix of dimensions $N_2 \times N_2$ and $N_1 \times N_1$, respectively. For the case of non-rectangular domains, the assembly process is similar to that used in the finite-element method.

The transformation system (10) can be preconditioned as

$$\widetilde{\mathscr{B}}\widetilde{\mathscr{T}} \begin{pmatrix} \widetilde{w} \\ \widehat{C}_1 \\ \widehat{C}_2 \end{pmatrix} = \widetilde{\mathscr{B}}\widetilde{u}. \tag{22}$$

It leads to

$$\begin{pmatrix} \widetilde{w} \\ \widehat{C}_1 \\ \widehat{C}_2 \end{pmatrix} = \left(\widetilde{\mathscr{B}}\widetilde{\mathscr{T}}\right)^+ \widetilde{\mathscr{B}}\widetilde{u}. \tag{23}$$

Numerical results

The proposed preconditioning scheme is examined numerically for both rectangular and non-rectangular domains.

Rectangular domain

Consider a square domain $[0,1]^2$. Condition numbers of the transformation matrix are computed for uniform grids, $[3\times 3,\ 5\times 5,\ \cdots,\ 95\times 95]$. The growth in the condition number is reduced from $O(N^{2.71})$ (unpreconditioning) to $O(N^{1.74})$ (preconditioning). At $N = 9025$, the proposed preconditioning scheme produces the condition number lower by about 4 orders of magnitude than the original system.

To study the numerical stability of the system matrix $\widetilde{\mathscr{A}}$, we also employ conventional RBFN techniques to provide a basis for the assessment. Conventional techniques seek the solution in the RBF space so that their solution procedures involve computing the system matrix only. The field variable u is decomposed into RBFs, which are then differentiated to obtain expressions for its derivatives (differentiated RBFNs (DRBFNs)). We employ a set of RBFs for DRBFNs which is exactly the same as that for IRBFNs (i.e. both approaches have the same number of RBFs, centres and widths (grid spacing)). Grid employed are $[7\times 7, 11\times 11, \cdots, 71\times 71]$. The present system matrix is much better conditioned. The condition number grows at the rate of $O(N^{1.10})$ and $O(N^{1.62})$ for IRBFNs and DRBFNs, respectively. At $N = 5041$, the gap is about 4 orders of magnitude between the two RBF techniques (i.e. 4.89×10^3 for IRBFNs and 2.58×10^7 for DRBFNs).

Non-rectangular domain

The domain of interest is a circular domain of radius 1/2. The problem domain is embedded in a uniform Cartesian grid and the exterior grid nodes are removed. We generate boundary nodes through the intersection of the grid lines and the boundary. The preconditioned system has a much lower condition number. Its rate is reduced from $O(N^{2.52})$ (unpreconditioning) to $O(N^{1.86})$ (preconditioning). The condition number of the system matrix is in the range of 4.72×10^1 to 4.38×10^3 for grids, $[7\times 7, 13\times 13, \cdots, 61\times 61]$.

Concluding remarks

This paper presents a simple and effective preconditioning scheme for IRBFN-based Cartesian-grid methods. Numerical results obtained show that the IRBFN matrix condition number is significantly improved.

Acknowledgement

This work is supported by the Australian Research Council.

*

References

[Mai-Duy and Tanner(2005)] **Mai-Duy, N.; Tanner, R.** (2005): Computing non-Newtonian fluid flow with radial basis function networks. *International Journal for Numerical Methods in Fluids*, vol. 48, pp. 1309–1336.

[Mai-Duy and Tran-Cong(2005)] **Mai-Duy, N.; Tran-Cong, T.** (2005): An efficient indirect RBFN-based method for numerical solution of PDEs. *Numerical Methods for Partial Differential Equations*, vol. 21, pp. 770–790.

[Mai-Duy and Tran-Cong(2006)] **Mai-Duy, N.; Tran-Cong, T.** (2006): Solving biharmonic problems with scattered-point discretisation using indirect radial-basis-function networks. *Engineering Analysis with Boundary Elements*, vol. 30, pp. 77–87.