Available online at www.sciencedirect.com

SCIENCE DIRECT

Applied Mathematical Modelling 30 (2006) 1118–1133

APPLIED MATHEMATICAL MODELLING

www.elsevier.com/locate/apm

A finite volume method based on radial basis functions for two-dimensional nonlinear diffusion equations

T.J. Moroney, I.W. Turner [*]

School of Mathematical Sciences, Queensland University of Technology, GPO Box 2434, Brisbane, QLD 4001, Australia

Received 14 January 2005; received in revised form 5 July 2005; accepted 11 July 2005
Available online 25 October 2005

Abstract

The finite volume method is the favoured numerical technique for solving (possibly coupled, nonlinear, anisotropic) diffusion equations. The method transforms the original problem into a system of nonlinear, algebraic equations through the process of discretisation. The accuracy of this discretisation determines to a large extent the accuracy of the final solution.

A new method of discretisation is presented, designed to achieve high accuracy without imposing excessive computational requirements. In particular, the method employs radial basis functions as a means of local gradient interpolation. When combined with high order Gaussian quadrature integration methods, the interpolation based on radial basis functions produces an efficient and accurate discretisation.

The resulting nonlinear, algebraic system is solved efficiently using a Jacobian-free Newton–Krylov method. Information obtained from the Newton–Krylov iterations is used to construct an effective preconditioner in order to reduce the number of nonlinear iterations required to achieve an accurate solution.

Results to date have been promising, with the method giving accuracy several orders of magnitude better than simpler methods based on shape functions for both linear and nonlinear diffusion problems.
© 2005 Elsevier Inc. All rights reserved.

Keywords: Control volume finite element; Jacobian free; Newton–Krylov; GMRES-DR; Deflation

1. Introduction

The finite volume method is a numerical method for solving partial differential equations. It is particularly well suited for solving problems involving diffusion. When using the finite volume method in this way an integral involving the diffusive flux component must be evaluated numerically, which requires both a means of numerical integration, and a means of gradient evaluation at the integration points. The accuracy of these two components has a large bearing on the overall accuracy of the finite volume method.

[*] Corresponding author. Tel.: +61 7 3864 2259; fax: +61 7 3864 2310.
E-mail addresses: t.moroney@qut.edu.au (T.J. Moroney), i.turner@qut.edu.au (I.W. Turner).

0307-904X/$ - see front matter © 2005 Elsevier Inc. All rights reserved.
doi:10.1016/j.apm.2005.07.007

The method of radial basis functions (RBFs) [1] is a method of scattered data interpolation that has applications in many different fields. Computer graphics [2], neural networks [3] and the solution of partial differential equations using collocation methods [4] are just three examples of the application of RBFs. In this work, we investigate the application of the finite volume method for solving two-dimensional steady-state diffusion equations, whereby radial basis functions are used to approximate fluxes at the integration points, and Gaussian quadrature is employed to evaluate the underlying line integrals. This approach can result in highly accurate finite volume discretisations, meaning that accurate solutions can be obtained using a much coarser mesh than is possible when using simpler discretisation methods, such as shape function-based methods.

However, by using the method of radial basis functions as an extension of shape function-based methods, an even more efficient method can be developed, generating an accurate solution in fewer iterations. Furthermore, when cast in this framework, the method of shape functions can provide useful information that can be used to precondition the underlying linear system based on radial basis functions, thus accelerating convergence of the iterative solver.

Some numerical experiments are presented that demonstrate the high accuracy of the new method, compared to that of shape functions. These involve both linear and nonlinear diffusion problems over unstructured, triangular meshes.

2. Finite volume method

The finite volume method utilises a mesh-based discretisation of a partial differential equation. Around each node in the mesh, control volumes are constructed, and solving the discretised PDE over these volumes yields a set of approximate values of the function at the mesh nodes.

Fig. 1 shows two methods of forming control volumes which we will call method (a) and method (b). In method (a), the element centroids have been connected to the midpoints of the element faces to form a control volume V_P that encloses node P. In method (b) the element centroids have been connected directly to neighbouring element centroids.

The finite volume discretisation proceeds as follows. Consider the steady-state diffusion equation

$$\nabla \cdot (\mathbf{D}\nabla\varphi) + S = 0. \tag{1}$$

This partial differential equation is transformed into control-volume form by integrating over each control volume V_P:

$$\int_{V_P} \nabla \cdot (\mathbf{D}\nabla\varphi)\,\mathrm{d}V + \int_{V_P} S\,\mathrm{d}V = 0. \tag{2}$$

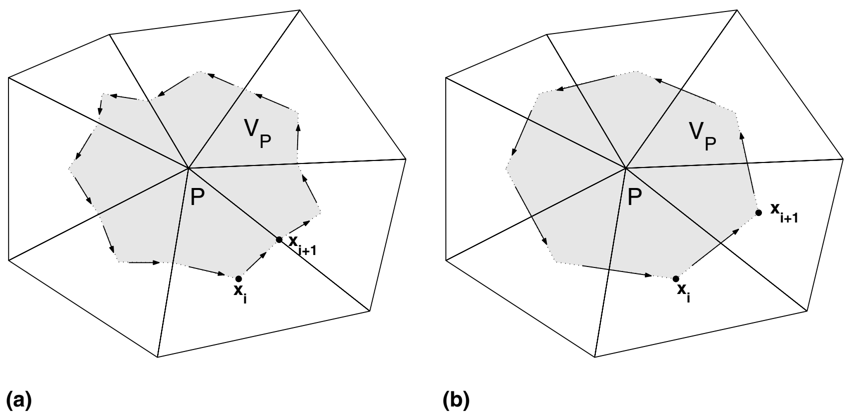


Fig. 1. Control volumes and line integral paths: (a) joining element centroids to face midpoints and (b) joining element centroids to neighbouring element centroids.

Applying the divergence theorem to (2), and defining

$$\bar{S} = \frac{1}{\Delta V_P}\int_{V_P} S\,\mathrm{d}V \tag{3}$$

as the volume-averaged value of the source term, the equation may be written

$$\int\!\!\int_{\sigma_P} \mathbf{D}\nabla\varphi\cdot\hat{\mathbf{n}}\,\mathrm{d}\sigma + \Delta V_P\bar{S} = 0$$

or, in two dimensions, more correctly as

$$\int_{C_P} \mathbf{D}\nabla\varphi\cdot\hat{\mathbf{n}}\,\mathrm{d}s + \Delta A_P\bar{S} = 0 \tag{4}$$

which, since no approximation has been made at this stage, is an exact reformulation of (1).

In solving (4), the integral $\int_C \mathbf{D}\nabla\varphi\cdot\hat{\mathbf{n}}\,\mathrm{d}s$ is computed through a process known as discretisation. The accuracy of this discretisation has a large bearing on the overall accuracy of the finite volume method.

3. Gaussian quadrature

In Section 2, two methods (a) and (b) for forming control volumes were presented. In both methods, the integral around the control volume boundary can be expressed as the sum of integrals over each control volume face (illustrated in Fig. 1):

$$\int_C \mathbf{D}\nabla\varphi\cdot\hat{\mathbf{n}}\,\mathrm{d}s = \sum_{i=0}^{n-1}\int_{\mathbf{x}_i}^{\mathbf{x}_{i+1}} \mathbf{D}\nabla\varphi\cdot\hat{\mathbf{n}}\,\mathrm{d}s. \tag{5}$$

Since analytic representations of the integrands are unavailable, each flux integral over the line segment from $\mathbf{x}_i$ to $\mathbf{x}_{i+1}$ must be computed numerically. Gaussian quadrature is a method of numerical integration that has a higher order than the commonly used midpoint rule. To apply this quadrature method to (5) the parameterisation

$$\mathbf{x}(t) = \frac{1}{2}(1-t)\mathbf{x}_i + \frac{1}{2}(1+t)\mathbf{x}_{i+1}, \quad -1 \leqslant t \leqslant 1$$

is introduced.

In the case of control volume method (a), a two-point scheme can be applied over each line segment. This requires four points in total to compute the flux between two adjoining control volumes (Fig. 2(a)). In the case of control volume method (b), a three-point scheme can be applied over the entire line segment, giving three points in total (Fig. 2(b)).

In terms of accuracy, it can be shown [5] that n-point Gaussian quadrature is $\mathcal{O}(h^{2n})$, where h is the length of (in this case) the control volume face. This means that two-point Gaussian quadrature is an $\mathcal{O}(h^4)$ method, while three-point Gaussian quadrature is an $\mathcal{O}(h^6)$ method. Importantly though, in this application, h for

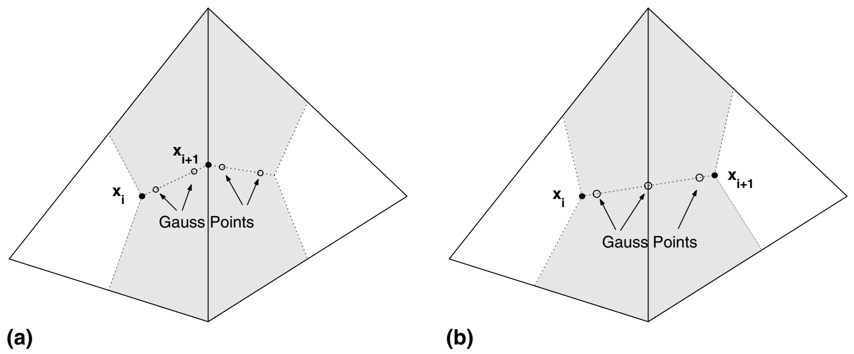


Fig. 2. (a) Two-point Gaussian quadrature and (b) three-point Gaussian quadrature.

the three-point scheme is (on average) twice the h for the two-point scheme (refer again to Fig. 2). Despite this, for small h we would expect the higher order of the three-point scheme to dominate, making it the more accurate method. This, combined with the fact that it requires one fewer point per flux evaluation, makes control volume method (b) using three-point Gaussian quadrature the preferred choice in this work. The discretisation of (5) is then given by

$$\int_C \mathbf{D}\nabla\varphi \cdot \hat{\mathbf{n}}\,\mathrm{d}s \approx \frac{1}{2}\sum_{i=0}^{n-1}\|\mathbf{x}_{i+1}-\mathbf{x}_i\|\sum_{j=1}^{3} w_j[\mathbf{D}\nabla\varphi\cdot\hat{\mathbf{n}}]_{\mathbf{x}_{ij}}, \tag{6}$$

where $\mathbf{x}_{ij}$ is the jth Gauss point between $\mathbf{x}_i$ and $\mathbf{x}_{i+1}$ (see Fig. 2) and w_j is the corresponding weight. We use the standard Gauss–Legendre points and weights given in [5].

Fig. 3 shows a comparison of the accuracy of two and three-point Gaussian quadrature schemes, along with the midpoint rule, for meshes of various refinements. These plots were produced by using the exact gradients taken from the analytical solution of test problem 1 in Section 6.1 and using these values in the various

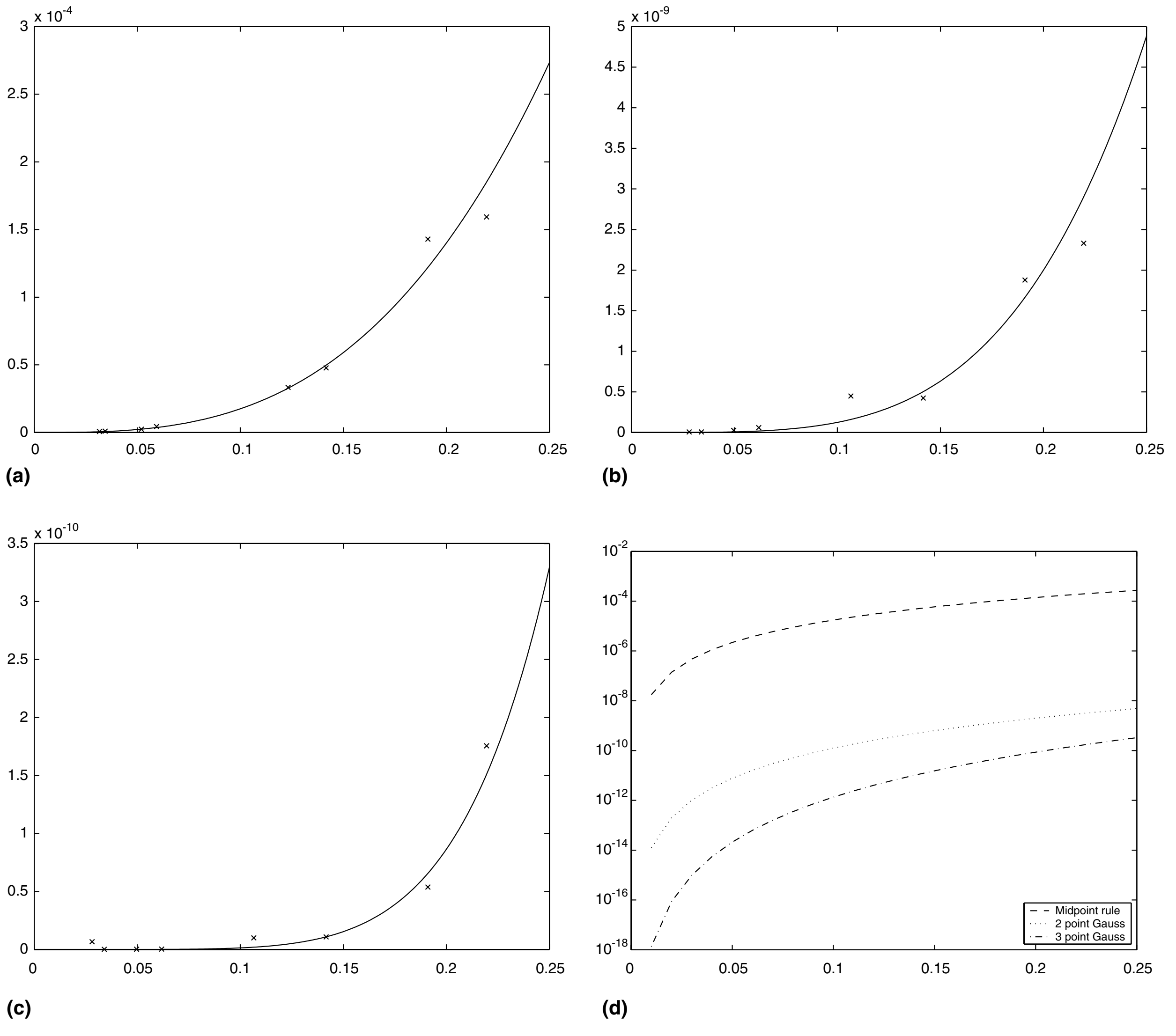


Fig. 3. Accuracy of line integrals versus face length. (a) Midpoint rule, (b) two-point Gaussian quadrature, (c) three-point Gaussian quadrature and (d) comparison of all three methods on a vertical log-scale.

quadrature methods to compute the fluxes across each control volume face. The graphs of $\mathcal{O}(h^2)$, $\mathcal{O}(h^4)$ and $\mathcal{O}(h^6)$ are superimposed over the midpoint (a), two-point (b) and three-point (c) rules, respectively. The fourth plot (d) compares the three methods on a vertical-log scale, demonstrating that for this problem the three-point method is the most accurate.

One final point to take into account is the method used to approximate (3). Although this is not the main focus of our investigation, it is nevertheless important to consider how to compute this integral accurately, for otherwise the error arising from approximating this source term could swamp any improvements made in computing the fluxes. In this work we use three-point Gaussian quadrature in two dimensions, ensuring that any errors in computing (4) are due to inaccuracies in the flux term, and not in the source term.

4. Interpolation

To compute the line integral (5) using a method such as Gaussian quadrature, the values of the integrand $\mathbf{D}\nabla\varphi \cdot \hat{\mathbf{n}}$ are needed at points on the control volume face. As was discussed in Section 2, the finite volume method involves building control volumes around the nodes, and solving the approximate Eq. (4) over these volumes to yield a value for φ at each node. As such, only the values of φ at these nodes can be used directly to approximate the diffusive flux. Values for φ or its gradient $\nabla\varphi$ at any other point must be computed using interpolation.

4.1. Shape functions

A popular method of interpolation used in the finite volume method is borrowed from finite element theory. The method of *shape functions* uses the interpolant

$$s(\mathbf{x}) = \sum_{i=1}^{3} \varphi_i N_i(\mathbf{x})$$

involving the value of φ at the vertices of a triangular element to compute a linear approximation of φ within the element [6].

Finite volume methods that employ these shape functions are often called control volume-finite element, or CV-FE, methods [7–9]. For problems where the gradient does not vary greatly over a given element this approach can be satisfactory. However, in many problems, there are regions where the gradient can vary significantly and using shape functions may not adequately capture this behaviour.

Some work has been done (see for example [10,11]) on extending CV-FE methods of flux approximation to second and higher orders of accuracy. Typically, a least-squares approach is used to formulate a higher-degree interpolating polynomial. Rather than extend or improve upon this work, it was decided to investigate alternative methods of interpolation that might also yield highly accurate gradient approximations.

4.2. Radial basis functions

The method of radial basis functions, or RBFs, is a scattered data interpolation method in $\mathbb{R}^n$ [1]. Given a set of nodes $\{\mathbf{x}_i\}_{i=1}^{N}$ and corresponding function values $\{\varphi_i\}_{i=1}^{N}$, the RBF interpolating function s is given by

$$s(\mathbf{x}) = \sum_{j=1}^{N} \lambda_j \phi(\|\mathbf{x} - \mathbf{x}_j\|) + \sum_{k=1}^{K} c_k q_k(\mathbf{x}), \tag{7}$$

where the λ_j and c_k are determined by the conditions

$$s(\mathbf{x}_j) = \varphi(\mathbf{x}_j), \quad j = 1, 2, \ldots, N \tag{8}$$

and

$$\sum_{j=1}^{N} \lambda_j q_k(\mathbf{x}_j) = 0, \quad k = 1, \ldots, K \tag{9}$$

with q_k the kth standard basis polynomial for the space π_p^n of all n-variate real polynomials of degree up to p (in two dimensions for example, these are $x^\alpha y^\beta$ for α, β nonnegative integers where $0 \leqslant \alpha + \beta \leqslant p$), and $K = \dim(\pi_p^n)$. In this work we use the multiquadric basis function [1]

$$\phi(r) = \sqrt{c^2 + r^2}. \tag{10}$$

This function includes the free parameter c, which we choose based on numerical experimentation.

The conditions (8) and (9) can be written in matrix form as follows [1]:

$$\mathbf{\Lambda w} = \mathbf{d}, \tag{11}$$

where

$$\mathbf{\Lambda} = \begin{pmatrix} \mathbf{\Phi} & \mathbf{P} \\ \mathbf{P}^{\mathrm{T}} & \mathbf{0} \end{pmatrix}, \quad \mathbf{w} = \begin{pmatrix} \boldsymbol{\lambda} \\ \mathbf{c} \end{pmatrix}, \quad \mathbf{d} = \begin{pmatrix} \mathbf{f} \\ \mathbf{0} \end{pmatrix}$$

and

$$\begin{aligned}
&(\mathbf{\Phi})_{ij} = \phi(\|\mathbf{x}_i - \mathbf{x}_j\|), \quad i = 1,\ldots,N, \; j = 1,\ldots,N, \\
&(\mathbf{P})_{i,k} = q_k(\mathbf{x}_i), \quad i = 1,\ldots,N, \; k = 1,\ldots,K, \\
&(\boldsymbol{\lambda})_i = \lambda_i, \quad i = 1,\ldots,N, \\
&(\mathbf{c})_k = c_k, \quad k = 1,\ldots,K, \\
&(\mathbf{f})_i = \varphi_i, \quad i = 1,\ldots,N.
\end{aligned}$$

Note that the system is square, as there are as many conditions as unknowns. Furthermore, for $\phi(r)$ given by (10), the interpolation matrix $\mathbf{\Lambda}$ is nonsingular [1]. In practice however, this matrix does tend to be very poorly conditioned [1]. We overcome this by using the truncated singular value decomposition [12] to compute $\mathbf{w}$ as

$$\mathbf{w} = \sum_{i=1}^{\tau} \frac{1}{\sigma_i} (\mathbf{u}_i^{\mathrm{T}} \mathbf{d}) \mathbf{v}_i,$$

where $\mathbf{u}_i$ and $\mathbf{v}_i$ are the left and right singular vectors of $\mathbf{\Lambda}$, respectively, and σ_i are the corresponding singular values ordered from largest to smallest, with τ the largest integer such that $\sigma_i > \dim(\mathbf{\Lambda})\epsilon$ for $i = 1,2,\ldots,\tau$, where ϵ is the machine epsilon.

4.2.1. Gradient evaluation

It is shown in [1] that the function s of (7) is differentiable when ϕ is the multiquadric (10), on any region of $\mathbb{R}^n$ that excludes the interpolation points. Thus from (7), ∇s is given by

$$\nabla s(\mathbf{x}) = \sum_{j=1}^{N} \lambda_j \left(\frac{\mathbf{x} - \mathbf{x}_j}{\|\mathbf{x} - \mathbf{x}_j\|} \right) \phi'(\|\mathbf{x} - \mathbf{x}_j\|) + \sum_{k=1}^{K} c_k \nabla q_k(\mathbf{x}). \tag{12}$$

4.2.2. Improving accuracy near boundaries

It is well-known that the accuracy of RBF-based interpolations is poorest near the boundaries [13]. We attempt to overcome this problem by using "local" interpolations, as with shape function-based methods, rather than a single "global" interpolant. Fig. 4 shows, for two different elements, how a local subset of nodes may be selected. The procedure is to include the nodes of the element itself (darkest), followed by the nodes of the element's neighbours (that have not already been included), followed by the nodes of their neighbours, and so on. Each step outwards in this way includes a new set of nodes and continues until a specified number of nodes have been included. In this work we use a constant proportion of the total nodes in the mesh as this limit, but more sophisticated selection techniques are possible.

From Fig. 4 it is clear that when evaluating the interpolating function s within interior elements, the points of evaluation will lie in the centre of this "cloud of points". For boundary elements however, the figure illustrates how the cloud becomes asymmetric. In these cases the points of evaluation within the element will be

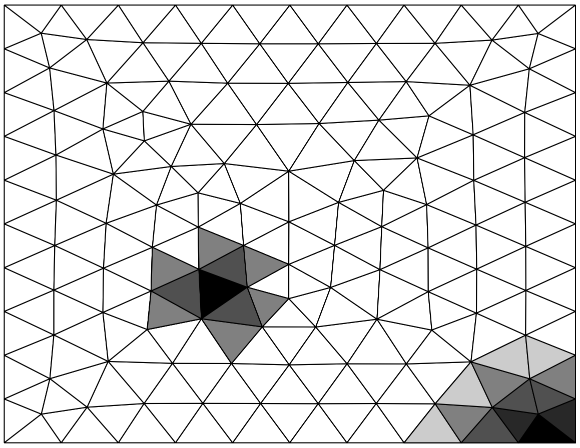


Fig. 4. Selecting a local subset of nodes.

near the edges of the cloud. Some methods that could be used to deal with these cases are presented in [13], but in this work when cast in the finite volume framework they were found to be ineffective.

There is another reason why this local fitting of radial basis functions is important in the finite volume setting. In the next section, a Newton–Krylov method will be presented for solving the system that arises when using the finite volume method to solve a nonlinear PDE. It is well known that Krylov-based methods are efficient when the underlying matrix is sparse. If local fitting of RBFs is used then the Jacobian matrix will be sparse, whereas a single global interpolant would give rise to a dense matrix.

5. Solution of nonlinear system

Eq. (4) is the two-dimensional control-volume analogue of the partial differential equation (1). In Sections 3 and 4, methods were presented that allow the flux component of (4) to be accurately computed. The value so computed depends on neighbouring values of φ—those values corresponding to the nodes used in constructing the interpolating function s. Denoting by $\boldsymbol{\varphi}$ the vector

$$\boldsymbol{\varphi} = (\varphi_1, \varphi_2, \ldots, \varphi_N)^{\mathrm{T}},$$

where N is the number of nodes in the mesh, the left hand side of (4), evaluated at node P using (6), is a function f_P of $\boldsymbol{\varphi}$:

$$f_P(\boldsymbol{\varphi}) = \frac{1}{2}\sum_{i=0}^{n-1} \|\mathbf{x}_{i+1} - \mathbf{x}_i\| \sum_{j=1}^{3} w_j [\mathbf{D}\nabla\varphi \cdot \hat{\mathbf{n}}]_{\mathbf{x}_{ij}} + \Delta A_P \bar{S} = 0. \tag{13}$$

The finite volume solution is found by simultaneously solving each co-ordinate function f_P to be zero. In general, the underlying partial differential equation is assumed to be nonlinear, and so each f_P is itself a nonlinear function of $\boldsymbol{\varphi}$. Since there are N such functions (one per node), a nonlinear system of equations is generated:

$$\mathbf{F}(\boldsymbol{\varphi}) = \mathbf{0}, \tag{14}$$

where $\mathbf{F} = (f_1, f_2, \ldots, f_N)^{\mathrm{T}}$. The solution of this system yields the approximate values of φ at the node points. This section discusses the technique used to compute this solution.

5.1. Jacobian-free Newton–Krylov

We use a Jacobian-free Newton–Krylov method based on the strategy given in [14]. In this approach, Eq. (14) is solved using the sequence of iterations

$$\boldsymbol{\varphi}^{(n+1)} = \boldsymbol{\varphi}^{(n)} + \lambda^{(n)}\boldsymbol{\delta\varphi}^{(n)}, \tag{15}$$

where $\lambda^{(n)}$ is determined according to a line search to ensure a sufficient decrease in $\|\mathbf{F}\|$ [14]. Each Newton step $\boldsymbol{\delta\varphi}^{(n)}$ is solved approximately according to

$$\|\mathbf{F} + \mathbf{J}\boldsymbol{\delta\phi}\| \leqslant \eta_n \|\mathbf{F}\| \tag{16}$$

with η_n known as the forcing term [14]. We use the NSOLI implementation given in [15] which utilises a three-point parabolic line search. The Jacobian matrix $\mathbf{J}$ is not computed explicitly, but rather its action on a vector $\mathbf{v}$ is approximated using the difference quotient

$$\mathbf{J}\mathbf{v} = \|\mathbf{v}\| \frac{\mathbf{F}(\boldsymbol{\varphi} + \varepsilon\hat{\mathbf{v}}) - \mathbf{F}(\boldsymbol{\varphi})}{\varepsilon}.$$

We follow the suggestion given in [16] in choosing ε as

$$\varepsilon = \frac{\sqrt{\epsilon}}{\|\mathbf{v}\|} \max(|\boldsymbol{\varphi}^{\mathrm{T}}\mathbf{v}|, \operatorname{typ}\boldsymbol{\varphi}|\mathbf{v}|)\operatorname{sign}(\boldsymbol{\varphi}^{\mathrm{T}}\mathbf{v}),$$

where ϵ is the machine epsilon, and typ $\boldsymbol{\varphi}$ is the "typical size" of $\boldsymbol{\varphi}$.

The method of GMRES with deflated restarting, or GMRES-DR [17] is used as the underlying linear Krylov method for computing the Newton step. This is a variant of restarted GMRES that augments the usual Krylov subspace with approximate eigenvectors of $\mathbf{J}$. Not only does this have the effect of improving convergence [17], the eigenvectors can also be used in preconditioning, as discussed in Section 5.2.

Each nonlinear iteration requires the solution of (16), which in turn requires the formation of the Krylov subspace $\mathscr{K}_m(\mathbf{J}, \mathbf{F})$ through repeated evaluation of $\mathbf{F}(\boldsymbol{\varphi} + \varepsilon\hat{\mathbf{v}})$ for different $\hat{\mathbf{v}}$. There are two costs associated with this evaluation when using radial basis functions. The first is the solution of the matrix system (11). This system must be solved prior to each evaluation in order to determine the RBF coefficients in (7), which are then used in the evaluation of the flux integrals in each of the co-ordinate functions (13).

Fortunately though, only the right hand side of (11) depends on $\hat{\mathbf{v}}$. The matrix itself depends only on geometrical quantities, so it may be decomposed at the beginning of the method, and its decomposition used to repeatedly generate new sets of RBF coefficients efficiently.

The second cost in the evaluation $\mathbf{F}(\boldsymbol{\varphi} + \varepsilon\hat{\mathbf{v}})$ is in computing the gradients themselves using the RBF interpolant. In practice this was found to be where the majority of computational time was spent. Naturally there is a trade-off between the number of nodes used in each interpolation (and the potential for increased accuracy) and the time spent in computing gradient approximations based on these nodes.

One way to reduce the impact of these RBF evaluations is simply to reduce the number of evaluations of $\mathbf{F}(\boldsymbol{\varphi} + \varepsilon\hat{\mathbf{v}})$ required to solve (14). To achieve this, it is useful to employ the following two-stage process in solving the nonlinear system:

Stage 1: Use shape functions as the method of interpolation (CV-FE).
Stage 2: Use radial basis functions as the method of interpolation.

The process of switching between stages is discussed in Section 6. Essentially, stage 1 updates the initial solution to one of reasonable accuracy using shape functions and their low-cost method of interpolation. Having obtained a reasonable solution, radial basis functions are introduced as the means of "correcting" this solution to produce a solution of high accuracy.

5.2. Preconditioning

It is widely recognised that a good preconditioning strategy is essential within a Newton–Krylov method. In this work we use two preconditioners, combining scaling and deflation to effectively remove both the largest and the smallest eigenvalues from the matrix spectrum.

The first preconditioner is based on Frobenius norm minimisation [18]. We build a matrix $\mathbf{M}_1$, with the same sparsity pattern as $\mathbf{J}$, that minimises

$$\|\mathbf{I} - \mathbf{J}\mathbf{M}_1\|_{\mathrm{F}}. \tag{17}$$

Using the Frobenius norm is the key to this method, for it allows problem (17) to be decomposed into N independent least squares sub-problems [18]

$$\|\mathbf{I}-\mathbf{J}\mathbf{M}_1\|_{\mathrm{F}}^2=\sum_{j=1}^{N}\|\mathbf{e}_j-\mathbf{J}\mathbf{m}_j\|_2^2, \tag{18}$$

where $\mathbf{e}_j$ is the jth unit vector, and $\mathbf{m}_j$ is the jth column of $\mathbf{M}_1$.

The second preconditioner uses the k approximate eigenvectors generated within the GMRES-DR iterations. These eigenvectors are used within GMRES-DR itself to improve the convergence by deflating the corresponding k smallest eigenvalues from the spectrum [17]. However, once these approximate eigenvectors are of sufficient accuracy, they can be used to form a preconditioner that does the deflation explicitly [19], leaving GMRES-DR to work on deflating the next k smallest. This matrix $\mathbf{M}_2$ takes the form

$$\mathbf{M}_2=\mathbf{I}_N+\mathbf{U}(\mathbf{T}^{-1}-\mathbf{I}_k)\mathbf{U}^{\mathrm{T}}, \tag{19}$$

where

$$\mathbf{T}=\mathbf{U}^{\mathrm{T}}\mathbf{J}\mathbf{U} \tag{20}$$

and the columns of $\mathbf{U}$ are the k approximate eigenvectors of $\mathbf{J}$. It can be shown that the matrix $\mathbf{J}\mathbf{M}_2$ has the same eigenvalues as $\mathbf{J}$, except that the k smallest eigenvalues have all been mapped to one [19].

When applying these preconditioners to the two-stage nonlinear process discussed in Section 5.1, it is feasible that although the Jacobian matrix will be different between stages, the same preconditioners might be effective throughout. The justification for this is that each Jacobian matrix is representative of the same physical process, and so could be expected to have similar spectral properties. As such, the shape function-based Jacobian could be used to compute $\mathbf{M}_1$ in (18) and $\mathbf{U}$ in (19) and (20) with a subsequent reduction in the cost of each matrix-vector product compared to using radial basis functions. This is the approach we take in this work.

6. Results and discussion

Two test problems are presented, the first a linear problem and the second a nonlinear problem. Both are based on the classical steady-state heat diffusion equation (1) on the square domain $0\leqslant x\leqslant 1$, $0\leqslant y\leqslant 1$. For testing purposes, both problems were solved using the full nonlinear framework discussed in Section 5. The parameter values used in solving these two problems are shown in Table 1. In the case of the Newton–Krylov method, these values follow recommendations made in [15] (this includes the dimension of the Krylov subspace m), while the number of eigenvectors k was chosen based on the values used in [17,20]. The radial basis function parameters were selected based on numerical experimentation. The mesh used was the unstructured, triangular mesh with 139 nodes and 236 elements shown in Fig. 4.

6.1. Test problem 1

The first test problem uses a constant diffusion matrix $\mathbf{D}=\mathrm{diag}(D_{xx},D_{yy})$ in (1), along with a constant source S. The boundaries $x=0$ and $y=0$ are taken to be insulated, and the boundaries $x=1$ and $y=1$ to be subject to Newtonian cooling with external temperature φ_∞. This problem is solvable analytically, with its solution given by [21]

$$\varphi(x,y)=\sum_{n=1}^{\infty}\frac{2(\lambda_n^2+H_x^2)F_n\cos(\lambda_n x)\cosh(\lambda_n\eta y)}{(\lambda_n^2+H_x+H_x^2)(\lambda_n\eta\sinh(\lambda_n\eta)+H_y\cosh(\lambda_n\eta))}+S\left[\frac{1-x^2}{2D_{xx}}+\frac{1}{h}\right]+\varphi_\infty, \tag{21}$$

where

$$\eta^2=\frac{D_{xx}}{D_{yy}},\quad H_x=\frac{h}{D_{xx}},\quad F_n=\int_0^1\frac{S}{D_{yy}}\left[\frac{h(x^2-1)}{2D_{xx}}-1\right]\cos(\lambda_n x)\,\mathrm{d}x$$

and the λ_n are the solutions to $\frac{H_x}{\lambda}=\tan(\lambda)$.

Table 1
Parameter values for numerical experiments

Parameter	Description	Value
NSOLI [15]		
η_0	Initial forcing term	0.9
$\eta_{\max}$	Safeguard value for forcing term	0.9
γ	Coefficient in forcing term	0.9
α	Exponent in forcing term	2
$\theta_{\min}$	Safeguard lower bound for line search	0.1
$\theta_{\max}$	Safeguard upper bound for line search	0.5
t	Small parameter in line search	10^{-4}
maxarm	Maximum step size reductions	10
Linear		
m	Dimension of Krylov subspace	40
k	Number of eigenvectors used in deflation	5
RBF		
ϕ	Radial basis function	MQ
c^2	Multiquadric parameter	3.0
prop	Proportion of nodes used for interpolation	0.2
p	Degree of polynomial term	3

Table 2
Physical parameter values for test problem 1

Parameter	Description	Value
D_{xx}	Thermal diffusivity in x direction	$5.0\ \mathrm{m^2\,s^{-1}}$
D_{yy}	Thermal diffusivity in y direction	$5.0 \times 10^i\ \mathrm{m^2\,s^{-1}}$
S	Source	$10.0\ \mathrm{K\,m^{-2}\,s^{-1}}$
h	Heat transfer coefficient	$2.0\ \mathrm{W\,m^{-2}\,K^{-1}}$
φ_∞	External temperature	20.0 K

The physical parameter values used for this problem are given in Table 2. The parameter D_{yy} was increased in powers of 10 from 5 to 5000, to provide an increasing challenge to the numerical methods. The exact solution of the problem with these parameters is shown in Fig. 5.

Fig. 6 shows the residual norm $\|\mathbf{F}\|$ plotted against the number of function evaluations for stages 1 and 2 of Section 5.1 in solving (14). In all cases, the problem was initially "over-solved" by employing two cycles of GMRES-DR using shape functions. Although this many iterations were not necessary to reduce the residual norm to the desired tolerance, they were employed to ensure that the eigenvector approximations computed as part of GMRES-DR were sufficiently accurate for use in the preconditioner $\mathbf{M}_2$ of Section 5.2. In practice then, the point at which the transition is made from stage 1 to stage 2 is determined by the accuracy of these eigenvector approximations. In this work, two GMRES-DR cycles were always used, but a more adaptive approach could be built which monitored the eigenvector residuals directly.

In stage 2, radial basis functions are introduced. Initially the residual norm increases, revealing that the current solution is not highly accurate with respect to this more sophisticated interpolation scheme. The subsequent stage 2 iterations improve upon this solution, finally generating a solution of high accuracy. In this way, the radial basis functions act as a corrector to the initial shape function-based solution.

To test the effectiveness of the preconditioning methods of Section 5.2, the system was solved, using $D_{yy}/D_{xx} = 1000$, using four different preconditioning strategies. The first strategy was to not precondition at all, the second strategy was to use just $\mathbf{M}_1$ (17), the third strategy was to use just $\mathbf{M}_2$ (19), and the fourth strategy was to use both $\mathbf{M}_1$ and $\mathbf{M}_2$, as was done in producing Fig. 6. Fig. 7 illustrates the stage 2 residual norm for these four strategies, showing that for this problem, both $\mathbf{M}_1$ and $\mathbf{M}_2$ are needed to achieve a satisfactory rate of convergence.

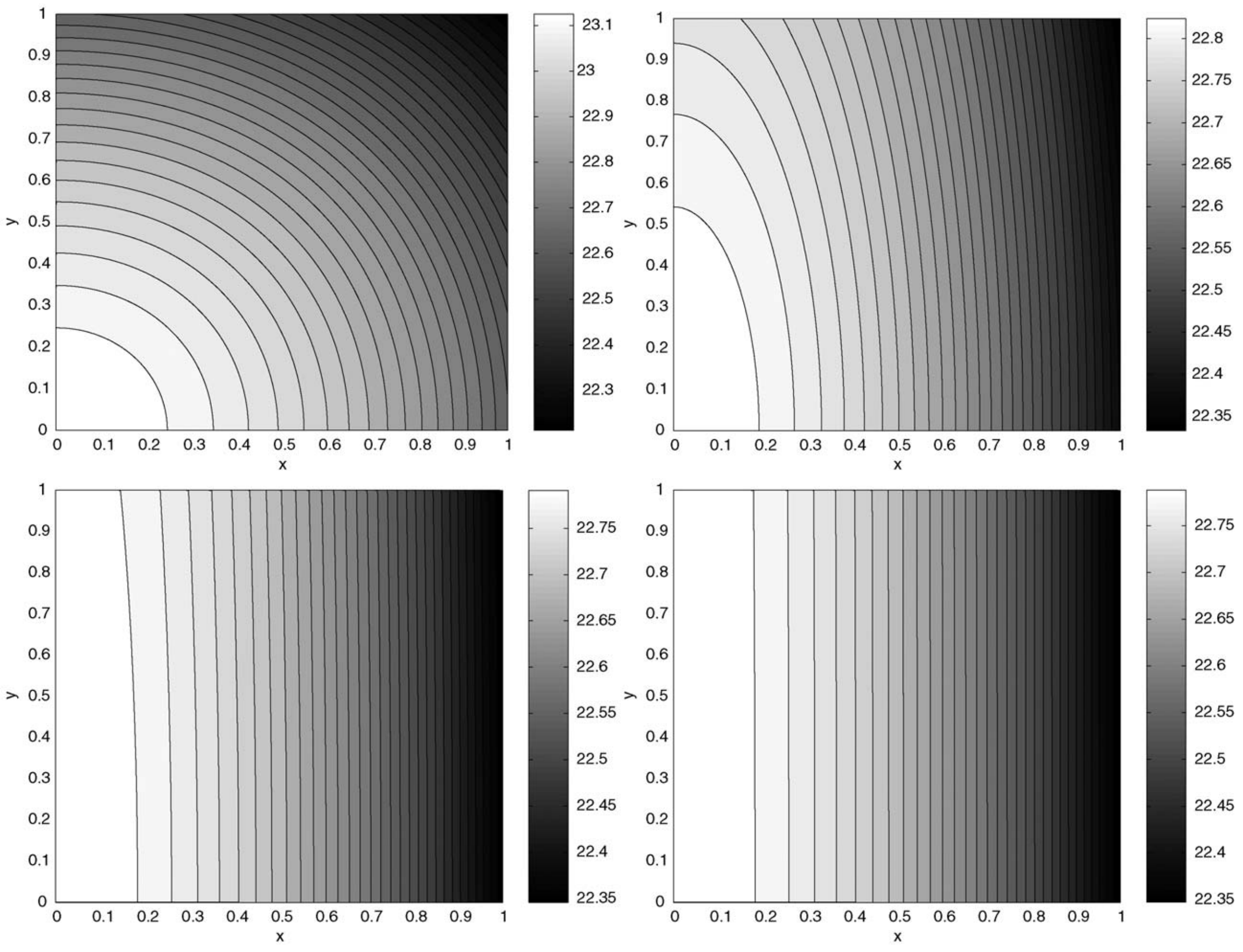


Fig. 5. Exact solution (21) where $D_{yy}/D_{xx} = 1$ (top left), $D_{yy}/D_{xx} = 10$ (top right), $D_{yy}/D_{xx} = 100$ (bottom left) and $D_{yy}/D_{xx} = 1000$ (bottom right).

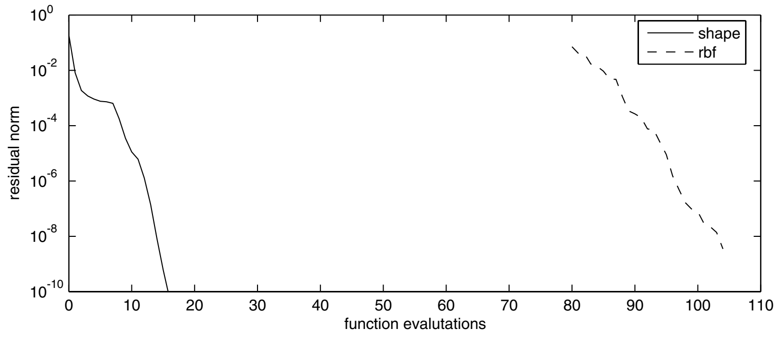
Table 3 shows the accuracy of the solution obtained using both shape functions and radial basis functions, as well as the time taken to converge. The error is measured by the formula

$$\text{error} = \sqrt{\frac{\sum_{i=1}^{N}(\varphi_i^{\mathrm{e}} - \varphi_i^{\mathrm{a}})^2}{\sum_{i=1}^{N}(\varphi_i^{\mathrm{e}})^2}}, \tag{22}$$

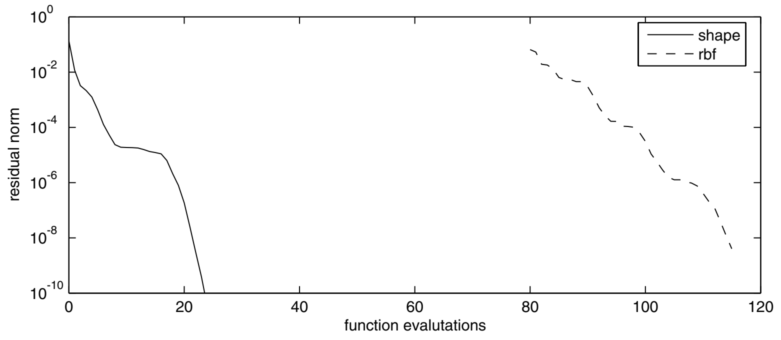
where φ_i^{e} is the exact solution at node i, obtained from (21), and φ_i^{a} is the approximate solution at node i. It is apparent that the accuracy of the approximate solution diminishes as the ratio D_{yy}/D_{xx} is increased, particularly when shape functions are used. In all cases however, the accuracy offered by radial basis functions is much better than that offered by shape functions.

The increased accuracy offered by radial basis functions is not without cost. It is clear from Table 3 that employing radial basis functions results in an increase in execution time compared with using shape functions. The time taken to solve the problem using shape functions is small, and consistent over the four problems. Using radial basis functions leads to an increase in execution time—for this problem, as much as five times longer.

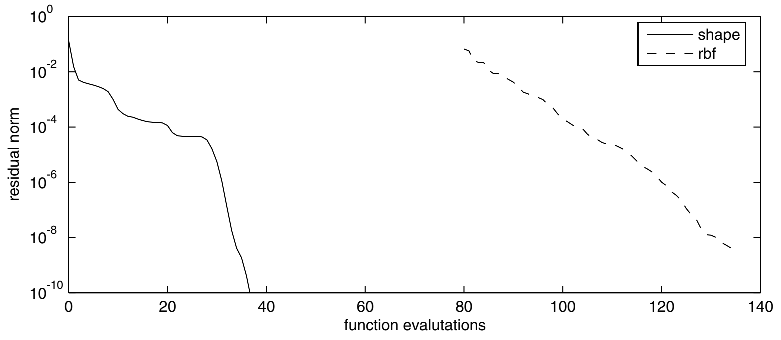
For a fair comparison, we must however take into consideration the increased accuracy offered by radial basis functions, despite the longer time needed to compute the solution. Table 4 shows, for the case where $D_{yy}/D_{xx} = 1000$, the mesh and time requirements for shape functions alone to generate a solution of



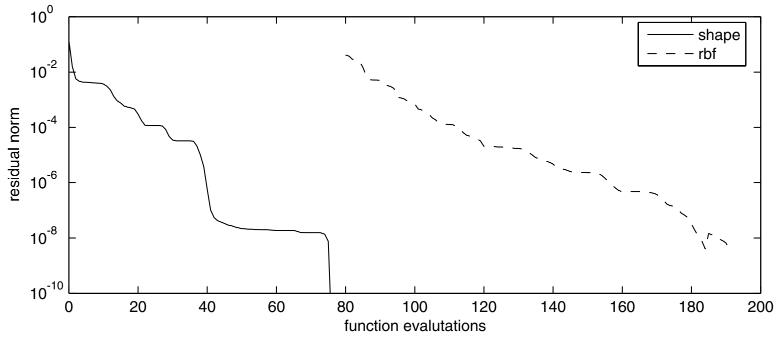
(a)



(b)



(c)



(d)

Fig. 6. Residual norm versus number of function evaluations for the two stages of the Newton–Krylov method for (a) $D_{yy}/D_{xx} = 1$, (b) $D_{yy}/D_{xx} = 10$, (c) $D_{yy}/D_{xx} = 100$ and (d) $D_{yy}/D_{xx} = 1000$.

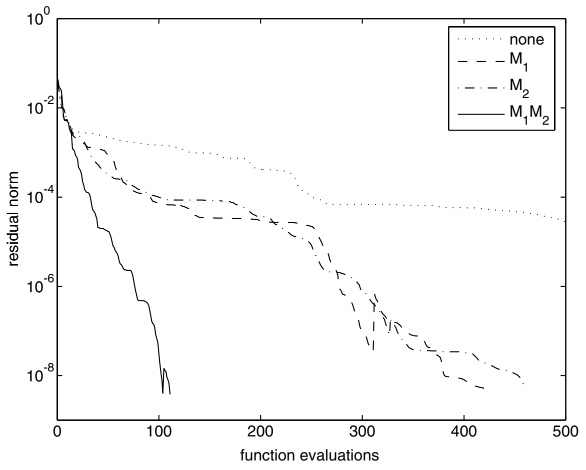


Fig. 7. Residual norm versus number of function evaluations for stage 2 of the Newton–Krylov method for $D_{yy}/D_{xx} = 1000$ using various preconditioning strategies.

Table 3
Results for solving test problem 1

D_{yy}/D_{xx}	Error		Time (s)	
	Shape	RBF	Shape	RBF
1	3.0E−5	6.9E−9	0.8	1.9
10	1.7E−5	1.9E−9	0.8	2.2
100	1.4E−4	4.4E−9	0.8	2.8
1000	8.7E−4	4.2E−8	0.8	4.6

Table 4
Number of elements needed and time taken for comparable accuracy between shape functions and RBFs for test problem 1

	Elements	Time (s)
Shape functions	5790	167
Radial basis functions	236	5

comparable accuracy to radial basis functions. We see that shape function-based interpolation requires a very fine mesh to achieve a solution of comparable accuracy to the RBF-based method, with a corresponding increase in the time to taken compute it.

6.2. Test problem 2

For the second test problem, the nonlinear diffusion matrix $\mathbf{D} = \mathrm{diag}(\varphi^{1.3}, \varphi^{1.3})$ was used in problem (1). The source term S was constructed by substituting the imposed solution

$$\varphi(x,y) = xy(1-x)(1-y)\mathrm{e}^{-(x^2+y^2)} \tag{23}$$

into (1). This source term was integrated, as per Eq. (3), using three-point Gaussian quadrature in two dimensions.

Fig. 8 shows both the linear (GMRES-DR) and nonlinear residuals, for stages 1 and 2 of the Newton–Krylov method for solving this problem. As for problem 1, the method begins by over-solving using shape functions, again generating accurate eigenvector approximations. However, because this is a nonlinear problem, one stage 1 iteration does not solve (14) completely—rather it computes only the first of several Newton steps

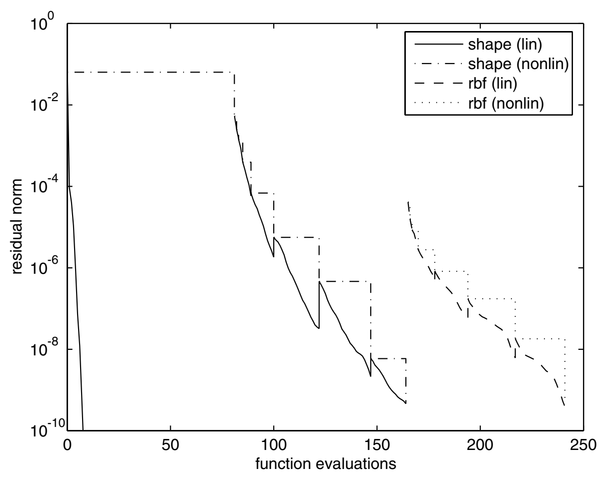


Fig. 8. Linear and nonlinear residuals for test problem 2.

(15). Further stage 1 iterations are needed to produce a solution of reasonable accuracy before switching to stage 2 of the method.

Table 5 shows why these iterations are important. It shows the total number of stage 2 iterations needed for convergence, based on when the switch from stage 1 to stage 2 is made (as measured by the stage 1 residual norm). The symbol × indicates that the method failed to converge. The table shows how, for this problem, further stage 1 iterations to reduce the nonlinear residual are beneficial in reducing the number of stage 2

Table 5
Number of stage 2 iterations, for given tolerance in switching between stages in test problem 2

Tolerance	10^{-1}	10^{-2}	10^{-3}	10^{-4}	10^{-5}	10^{-6}	10^{-7}	10^{-8}	10^{-9}
Iterations	×	112	98	96	79	70	82	77	78

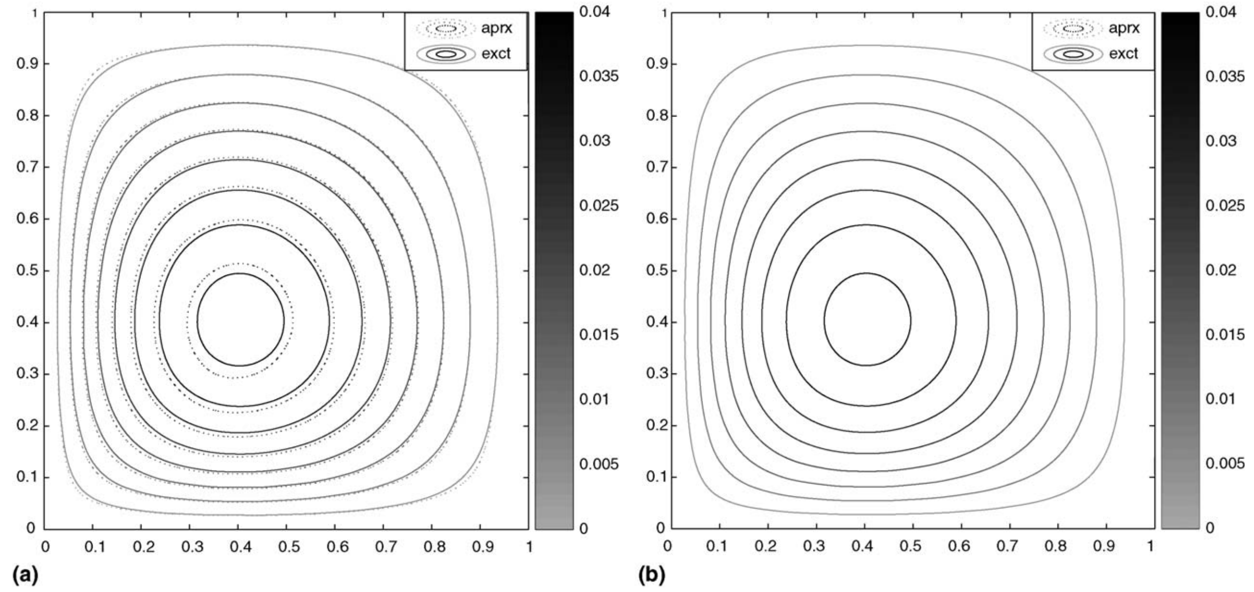


Fig. 9. Exact solution (23) compared to the approximate solution: (a) shape functions and (b) radial basis functions.

Table 6
Results for solving test problem 2

	Error	Time (s)
Shape functions	2.48E−2	1.3
Radial basis functions	5.98E−6	6.1

iterations required to achieve final convergence. Given that these shape function-based iterations are quite cheap, we recommend reducing the residual norm in stage 1 to the desired final tolerance before switching to stage 2.

Fig. 9 shows the computed solution compared to the true solution, using both shape functions and radial basis functions. The accuracy of these solutions, along with the time taken to compute them are shown in Table 6. Again, radial basis functions offer significant improvements in accuracy over shape functions.

7. Conclusions

The use of radial basis functions along with three-point Gaussian quadrature in the discretisation of the finite volume method can yield accurate solutions on a much coarser mesh than is possible when using shape functions. Inaccuracies in radial basis function-based interpolations near boundaries can be overcome by fitting local interpolants rather than a single global interpolant.

This approach is well-suited for steady-state diffusion problems, both linear and nonlinear, and can be incorporated into a Jacobian-free Newton–Krylov method. The use of this method as a corrector to that of shape functions is advantageous in that it allows for accurate solutions to be obtained with fewer iterations. Efficient preconditioning is also made possible by re-using information gathered from the shape function-based iterations.

In the test cases studied, the method produced solutions several orders of magnitude more accurate than shape functions alone. Although this increased accuracy was at the cost of increased computational time, to obtain the equivalent accuracy using shape functions would require finer meshes and even more computational time.

The proposed two-stage solution process appears to work well for both the linear and nonlinear test problems, however the decision of when to switch in an adaptive manner from stage 1 to stage 2 requires further investigation. For the linear problem, continuing stage 1 for two GMRES-DR cycles is beneficial. Adapting the nonlinear solution process is more complex and is the subject of further research.

References

[1] M. Powell, The theory of radial basis function approximations in 1990, in: Advances in Numerical Analysis, Wavelets, Subdecision Algorithms and Radial Basis Functions, vol. II, Oxford University Press, 1990, pp. 105–210.

[2] J.C. Carr, R.K. Beatson, J.B. Cherrie, T.J. Mitchell, W.R. Fright, B.C. McCallum, T.R. Evans, Reconstruction and representation of 3D objects with radial basis functions, Computer Graphics (SIGGRAPH 2001 Proceedings) 103 (2001) 67–76.

[3] D. Lowe, Radial basis function networks, The Handbook of Brain Theory and Neural Networks, MIT Press, 1998, pp. 779–782.

[4] G.E. Fasshauer, Solving partial differential equations by collocation with radial basis functions, Surface fitting and multiresolution methods, in: Proceedings of the 3rd International Conference on Curves and Surfaces, vol. 2, 1997, pp. 131–138.

[5] J. Epperson, An Introduction to Numerical Methods and Analysis, Wiley, 2002.

[6] R. Cook, Concepts and Applications of Finite Element Analysis, Wiley, 2001.

[7] S.V. Patankar, Numerical Heat Transfer and Fluid Flow, Taylor & Francis, 1980.

[8] W.J. Ferguson, I.W. Turner, A control volume finite element numerical simulation of the drying of spruce, J. Comput. Phys. 125 (1996) 59–70.

[9] C. Bailey, G. Taylor, M. Cross, P. Chow, Discretisation procedures for multi-physics phenomena, J. Comput. Appl. Math. 103 (1999) 3–17.

[10] P. Jayantha, I. Turner, A second order control-volume finite-element least-squares strategy for simulating diffusion in strongly anisotropic media, J. Comput. Math. 13 (1) (2005) 31–56.

[11] S.L. Truscott, I.W. Turner, An investigation of the accuracy of the control volume finite element method based on triangular prismatic elements for simulating diffusion in anisotropic media, Numer. Heat Trans, Part B: Fund. 46 (3) (2004) 243–268.

[12] P.C. Hansen, The truncated SVD as a method for regularization, BIT 27 (4) (1987) 534–553.

[13] B. Fornberg, T. Driscoll, G. Wright, R. Charles, Observations on the behavior of radial basis function approximations near boundaries, Comput. Math. Appl. 43 (3–5) (2002) 473–490.
[14] S. Eisenstat, H. Walker, Choosing the forcing terms in an inexact Newton method, SIAM J. Sci. Comput. 17 (1) (1996) 16–32.
[15] C. Kelly, Solving Nonlinear Equations with Newton's Method, SIAM, 2003.
[16] P. Brown, Y. Saad, Hybrid Krylov methods for nonlinear systems of equations, Siam J. Sci. Statist. Comput. 11 (3) (1990) 450–481.
[17] R. Morgan, GMRES with deflated restarting, SIAM J. Sci. Comput. 24 (1) (2002) 20–37.
[18] B. Carpentieri, I.S. Duff, L. Giraud, Sparse pattern selection strategies for robust Frobenius-norm minimization preconditioners in electromagnetism, Numer. Linear Algebr. Appl. 7 (7–8) (2000).
[19] K. Burrage, J. Erhel, On the performance of various adaptive preconditioned GMRES strategies, Numer. Linear Algebr. Appl. 5 (2) (1998) 101–121.
[20] A. Chapman, Y. Saad, Deflated and augmented Krylov subspace techniques, Numer. Linear Algebr. Appl. 4 (1) (1997) 43–66.
[21] J. Hill, J. Dewynne, Heat Conduction, Oxford, 1987.