Numer Algor (2017) 76:441–471
DOI 10.1007/s11075-017-0265-5

ORIGINAL PAPER

Adaptive radial basis function interpolation using an error indicator

Qi Zhang[1] · Yangzhang Zhao[1] · Jeremy Levesley[1]

Received: 23 November 2015 / Accepted: 5 January 2017 / Published online: 26 January 2017

Abstract In some approximation problems, sampling from the target function can be both expensive and time-consuming. It would be convenient to have a method for indicating where approximation quality is poor, so that generation of new data provides the user with greater accuracy where needed. In this paper, we propose a new adaptive algorithm for radial basis function (RBF) interpolation which aims to assess the local approximation quality, and add or remove points as required to improve the error in the specified region. For Gaussian and multiquadric approximation, we have the flexibility of a shape parameter which we can use to keep the condition number of interpolation matrix at a moderate size. Numerical results for test functions which appear in the literature are given for dimensions 1 and 2, to show that our method performs well. We also give a three-dimensional example from the finance world, since we would like to advertise RBF techniques as useful tools for approximation in the high-dimensional settings one often meets in finance.

Keywords Radial basis function · Error indicator · Adaptive

✉ Jeremy Levesley
jl1@le.ac.uk

Qi Zhang
qz49@le.ac.uk

Yangzhang Zhao
yz177@le.ac.uk

[1] Department of Mathematics, University of Leicester, University Road, Leicester, UK

1 Introduction

In most applications, data is generated with no knowledge of a function from which it was derived, so that an approximation model is needed. When sampling from the target function is expensive and time-consuming, a model that can indicate the location for generating the next samples and can provide enough accuracy with as few as possible samples is very desirable. Such examples include industrial processes, such as engine performance, where one experiment for a different set of (potentially many) parameters might take hours or days. Adaptive radial basis function (RBF) interpolation is suitable for such problems, mainly due to its ease of implementation in the multivariate scattered data setting.

There are several feasible adaptive RBF interpolation methods. For example, Driscoll and Heryudono [8] have developed the residual sub-sampling method of interpolation, used in boundary-value and initial-value problem with rapidly changing local features. The residual sub-sampling method is based on RBF interpolation. Their method approximates the unknown target function via RBF interpolation on uniformly distributed centres. Then, the error is evaluated at intermediate points; this stage could be called the indication stage. When the error exceeds a pre-set refinement threshold, corresponding points are added to the centre set, and when the error is below a pre-set coarsening threshold, corresponding centres are removed from the centre set. In this method, knowledge of the target function is assumed.

In [6], the authors developed an adaptive method by applying the scaled multiquadrics

$$\phi_j(x) := \sqrt{1 + c_j^2 (x - x_j)^2} \tag{1.1}$$

to replace piecewise linear spline interpolant in classical B-spline techniques as:

$$\begin{aligned} B_j(x) := & \frac{1}{2h_{j-1}} \sqrt{1 + c_{j-1}^2 (x - x_{j-1})^2} - \frac{h_{j-1} + h_j}{2h_{j-1}h_j} \sqrt{1 + c_j^2 (x - x_j)^2} \\ & + \frac{1}{2h_j} \sqrt{1 + c_{j+1}^2 (x - x_{j+1})^2}; \end{aligned} \tag{1.2}$$

here, x_j are data nodes, $h_j := x_{j+1} - x_j$ and $c_j \in [0, \infty)$ are some design parameters. This method provides smoother interpolants and also superior shape-preserving properties. Schaback et al. [16] and Hon et al. [10] have proposed an adaptive greedy algorithm which gives linear convergence. Behrens and Iske et al. [4] have combined an adaptive semi-Lagrangian method with local thin-plate spline interpolation. The local interpolation gives out the fundamental rule of adaptation, and it is crucial for approximation accuracy and computational efficiency.

In this paper, we present a new method for adaptive RBF interpolation which could be a suitable solution for the kind of problems mentioned in the first paragraph. As the numerical examples show, the method can indicate the best location to generate the next sample and can provide sufficient accuracy with fewer samples than the competitor methods.

Our goal is achieved by the use of an error indicator, a function which indicates the approximation quality at nodes inspected by the algorithm. The error indicator compares a global RBF interpolant and a local RBF interpolant. The advantage of this

error indicator is that it requires no extra function evaluation and indicates regions where the approximation error is high, so that we generate sets of points which are good candidates for optimally reducing the global error. This is the key differences between our method and the sub-sampling method in [8], which needs to sample the target function at each indication stage.

With the current state of the art in error estimates for RBF approximation, it is not possible to theoretically justify convergence of the algorithm we present. In particular, we interpolate with a variable shape parameter in the multiquadric. Clearly, if we allow small perturbations from a uniform choice of shape parameter, a continuity argument will show that the interpolation matrix is still non-singular. However, quantification of "small" in this case is not possible. A theoretical justification for invertibility of systems with a variable shape parameter is given in [5], but there is no convergence analysis. Since, in this case, the approximation sits on a submanifold of a higher-dimensional ambient space, a modification of the analysis in [13] might be used to prove convergence. This assumes that the data points are becoming dense in the region under consideration, so for a full proof, one would need to show that the refinement routine produced data sets of increasing density.

Our method is easy to implement in high-dimensional cases due to the nature of RBF. In Section 2, we describe RBF approximation, in Section 3, we describe our adaptive algorithm and in Section 4, we present numerical examples in one, two and three dimensions, comparing these to other available algorithms. We close the section of numerical examples by demonstrating that the algorithm is robust to choices of parameters.

2 Radial basis function interpolation

In this section, the basic features of the grid-free radial basis function interpolation are explained. Consider a function $f : \mathcal{R}^d \to \mathcal{R}$, a real valued function of d variables, that is to be approximated by $S_{\mathbf{X}} : \mathcal{R}^d \to \mathcal{R}$, given values $\{f(\mathbf{x}_i) : i = 1, 2, 3, \ldots, n\}$, where $\{\mathbf{x}_i : i = 1, 2, 3 \ldots, n\}$ is a set of distinct points in $\mathcal{R}^d$, called the centre set $\mathbf{X}$.

The approximation to the function f is of the form

$$S_{\mathbf{X}}(\mathbf{x}) = \sum_{i=1}^{n} \alpha_i \phi_i(\|\mathbf{x} - \mathbf{x}_i\|) + \sum_{j=1}^{q} \beta_j p_j(\mathbf{x}), \tag{2.1}$$

where $\phi_i : \mathcal{R}_+ \to \mathcal{R}$ is a univariate basis function, p_j, $j = 1, \cdots, q = \binom{m-1+d}{d}$ are a basis for Π_m^d, the the linear space of all d-variate polynomials of degree less than or equal to m. The coefficients α_i, $i = 1, \cdots, n$, and β_j, $j = 1, \cdots, q$, are to be determined by interpolation. Here, $\|\cdot\|$ denotes the Euclidean norm on $\mathcal{R}^d$.

The above form of the approximation is different to the standard form one might find (see e.g. [14]), in which the basis function ϕ_i is the same for all i. We are leaving ourselves the flexibility of changing the basis function, via a different choice of shape parameter (e.g. width of the Gaussian), depending on the density of data points in a particular region. We will comment later on how we do this. The standard theory for

RBF approximation breaks down if we do this, but we follow the standard approach, and see that the results are promising. We will comment as we go along on where the standard theory does not apply, but we use it to guide our choices.

The interpolation condition is $S_{\mathbf{X}}(\mathbf{x}_k) = f(\mathbf{x}_k)$, $k = 1, \cdots, n$. If we write this out in full, we get

$$\sum_{i=1}^{n} \alpha_i \phi_i(\|\mathbf{x}_k - \mathbf{x}_i\|) + \sum_{j=1}^{q} \beta_j p_j(\mathbf{x}_k) = f(\mathbf{x}_k), \quad k = 1, 2, 3 \ldots, n. \tag{2.2}$$

However, this leaves us with q-free parameters to find, so we need some extra conditions. Mimicking the natural conditions for cubic splines, we enforce the following:

$$\sum_{i=1}^{n} \alpha_i p_j(\mathbf{x}_i) = 0, \quad j = 1, 2, 3, \ldots, q. \tag{2.3}$$

One rationale (the one which the authors favour) is that these conditions ensure that, in the standard case with $\phi_i = \phi$, the interpolant decays at ∞ at an appropriate rate. It also turns out that, again in the standard case, these conditions mean that, for data in general positions (we call this Π_m^d-*nondegeneracy*), the interpolation problem has a unique solution if the basis function has certain properties (which we discuss below). As commented in [1], the addition of polynomial terms does not improve greatly the accuracy of approximation for non-polynomial functions.

Combining the interpolation condition and side condition together, the system can be written as

$$\begin{pmatrix} A & P^T \\ P & 0 \end{pmatrix} \begin{pmatrix} \alpha \\ \beta \end{pmatrix} = \begin{pmatrix} f \\ 0 \end{pmatrix}, \tag{2.4}$$

where $A_{ij} = (\phi(\|\mathbf{x}_i - \mathbf{x}_j)\|))$, $1 \le i, j \le n$ and $P_{ij} = (p_j(\mathbf{x}_i))$, $1 \le i \le n$, $1 \le j \le q$. Schaback [15] discusses the solvability of the above system, which is guaranteed by the requirement that $\text{rank}(P) = q \le n$ and

$$\lambda \|\alpha\|^2 \le \alpha^T A \alpha \tag{2.5}$$

for all $\alpha \in \mathcal{R}^n$ with $P\alpha = 0$, where λ is a positive constant. The last condition is a condition on the function ϕ, and functions which satisfy this condition, irrespective of the choice of the points in $\mathbf{X}$, are called *conditionally positive definite of order m*. The condition $\text{rank}(P) = q \le n$ is called Π_m^d-nondegeneracy of X, because such sets of polynomials are uniquely determined by their values on the set X.

Commonly used RBFs are

$$\phi(r) = \begin{cases} \sqrt{1 + (cr)^2}, & \text{multiquadric,} \\ \exp(-cr)^2, & \text{Gaussian,} \\ r^2 \log(r), & \text{thin-plate spline,} \\ r, & \text{linear spline.} \end{cases}$$

For the multiquadric and the Gaussian, we have a free parameter c which is named the *shape parameter*, which can be decided by the user. In this paper, our interpolant in (2.1) will have

$$\phi_i(r) = \sqrt{1 + (c_i r)^2},$$

so that a different choice of shape parameter will be used at each point $\mathbf{x}_i$, $i = 1, \cdots, n$. Thus, we call our interpolant $S_{\mathbf{X}}^{\text{multi}}$. The Gaussian is positive definite (conditionally positive definite of order 0) and the multiquadric is of order 1. The thin-plate spline and linear spline are examples of polyharmonic splines, and analysis of interpolation with these functions was initiated by Atteia [2], and generalised by Duchon [9]. The thin-plate spline is conditionally positive definite of order 2, and the linear spline of order 1.

The polyharmonic splines have the following form:

$$\phi_{d,k}(r) = \begin{cases} r^{2k-d}\log(r), & \text{if } d \text{ is even,} \\ r^{2k-d}, & \text{if } d \text{ is odd,} \end{cases} \tag{2.6}$$

where k is required to satisfy $2k > d$.

When solving a linear system, we often find that the solution is very sensitive to changes in the data. Such sensitivity of a matrix B is measured by the condition number:

$$\kappa(B) = \|B\|\|B^{-1}\|, \tag{2.7}$$

where

$$\|B\| = \sup_{\|\mathbf{x}\|=1} \|B\mathbf{x}\|$$

(recall that $\|\cdot\|$ is the Euclidean norm (2-norm) of a vector). If B is symmetric, then

$$\kappa(B) = \frac{\sigma_{\max}}{\sigma_{\min}},$$

where $\sigma_{\max}$ and $\sigma_{\min}$ are the largest and smallest eigenvalue (in absolute size) of B. A well-conditioned matrix will have a small condition number $\kappa(B) \geq 1$, while an ill-conditioned matrix will have a much larger condition number. The reason we want to keep the condition number in a moderate scale is that theoretically one less digit of accuracy will be obtained in a computed solution as the condition number increases by a factor of 10. To do this, we need (at least) to try to keep the $\sigma_{\min}$ getting close to 0.

The multiquadric interpolation matrix A above is, in the standard case, symmetric. However, if we change the shape parameter at each point, A is no longer symmetric. In the symmetric case, Ball et al. [3] show that the smallest eigenvalue of the interpolation matrix has the lower bound $\sigma_{\min} \geq he^{-\mu cd/h}$, for some constant μ, where h is the minimum separation between points in the data set. Thus, even though this theory does not cover our algorithm in which we change the shape parameter depending on the local density of data, we choose $c = \mu/h$ for some positive constant ν (which we will specify later), in order to keep the above lower bound from decreasing (at least at an exponential rate).

As we said previously, if we change the shape parameter at each point, then we have no guarantee of the invertibility of the interpolation matrix A. In [6], Lenarduzzi et al. have proposed a method of interpolation with a variably scaled kernel method. The idea is to define a scale function $c(\cdot)$ on the domain $\Omega \in \mathcal{R}^d$ to transform an interpolation problem from data locations $\mathbf{x}_j$ in $\mathcal{R}^d$ to data locations $(\mathbf{x}_j, c(\mathbf{x}_j))$ and to use a fixed shape parameter basis function on $\mathcal{R}^{d+1}$ for interpolation. By this method, the invertibility of interpolation matrix A is guaranteed, and the scale function $c(\cdot)$ serves as adaptive shape parameter to keep the condition number $\kappa(A)$ small.

3 Adaptive point sets and the error indicator

In our method, we generate a sequence of sets $\mathbf{X}_0, \mathbf{X}_1, \cdots$, where we generate $\mathbf{X}_{k+1}$ from $\mathbf{X}_k$ via a refinement and coarsening strategy which we describe below. In contrast with e.g. Iske and Levesley [12], we do not use a nested sequence of points. Our strategy for including or removing points depends on an error indicator. We follow the idea of Behrens et al. [4] who wished to decide on the positioning of points in a semi-Lagrangian fluid flow simulation. They compared a local interpolant with some known function and refined where the error was large, and coarsened where small.

Our error indicator is based on the principle that in a region which is challenging for approximation, two different approximation methods will give significantly different results, when compared with regions where approximation is more straightforward. Our first approximation method is our current interpolant $S^{\text{multi}}_{\mathbf{X}_k}$ at level k. Our second approximation method will be via a polyharmonic spline interpolant based on values of our approximation on a local set of points. Then, a function $\eta(x)$ with domain in the current centre set assigns a positive value to each centre and each indication point ξ. This value indicates the local approximation quality at each indication nodes and serves to determine where the approximate solution $S^{\text{multi}}_{\mathbf{X_k}}$ requires more accuracy at these specified indication nodes and requires no extra function evaluation. Below, we give the definition of the error indicator which is proposed in this paper.

Definition 3.1 For $k \geq 0$, let the indication set Ξ_k, corresponding to $\mathbf{X}_k$, be a set of scattered points, at which we want to know the approximation quality. The error indicator function $\eta(\xi)$ is defined by

$$\eta(\xi) = |S^{\text{multi}}_{\mathbf{X}}(\xi) - S^{\text{ps}}_{N_\xi}(\xi)| \qquad \xi \in \Xi. \tag{3.1}$$

The function $S^{\text{multi}}_{\mathbf{X}}(\xi)$ is the multiquadric radial basis function approximation of the target function at ξ by the centre set $\mathbf{X}$. The function $S^{\text{ps}}_{N_\xi}(\xi)$ is the polyharmonic spline radial basis function reconstruction which matches the target function value at ξ by a scattered point set N_ξ in a neighbourhood around ξ. N_ξ is a subset of the centres set $\mathbf{X}$. We call N_ξ the neighbourhood set of ξ, the elements in N_ξ are the M nearest neighbour points to ξ from the centre set $\mathbf{X}$. Hence,

$$S^{\text{ps}}_{N_\xi}(v) = f(v) \qquad \text{for } v \in N_\xi. \tag{3.2}$$

For $k = 0$, the indication set Ξ_0 is determined by $\mathbf{X}_0$. For $k > 0$, the indication set Ξ_k is determined by $\mathbf{X}_k$ and $\mathbf{X}_{k-1}$. The details of the relationship between Ξ_k and $\mathbf{X}_k$ is explained in the algorithm flow steps.

For $d = 1$, the neighbourhood set of ξ is $N_\xi = \{x_1, x_2, ..., x_M\}$ and the local approximation

$$S^{\text{ps}}_{N_\xi}(x) = \sum_{i=1}^{M} \alpha_i (\|x - x_i\|)^5 + \beta_0 + \beta_1 x + \beta_2 x^2, \tag{3.3}$$

where we will specify M in the numerical examples.

For $d = 2$, the neighbourhood set of ξ is $N_\xi = \{\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_M\}$ with $\mathbf{x} = (x_1, x_2) \subseteq \mathcal{R}^2$, and

$$S^{\mathrm{ps}}_{N_\xi}(\mathbf{x}) = \sum_{i=1}^{M} \alpha_i (\|\mathbf{x} - \mathbf{x}_i\|)^2 \log(\|\mathbf{x} - \mathbf{x}_i\|)) + \beta_0 + \beta_1 x_1 + \beta_2 x_2. \tag{3.4}$$

The error indicator defined above measures the deviation between a global approximation and a local approximation at the point ξ. The intuition inside this method is simple; when ξ lies in a smooth region of the function, two different approximation should give out similar results, then the error indicator $\eta(\xi)$ is expected to be small, whereas in the region of less regularity for f, or around discontinuities, the error indicator $\eta(\xi)$ is expected to be large. In [11], the authors use standard RBF error estimates for polyharmonic spline approximation to show that as points get close together, the local error of approximation converges at order $h^{k-d/2}$ (see (2.6)), where h measures the local spacing of points. Thus, assuming that the global approximation process converges rapidly for smooth functions; the error indicator will get small at the rate of the local approximation process.

So, the error indicator $\eta(\xi), \xi \in \Xi$ is used to flag points $\xi \in \Xi$ as "to be refined" or its corresponding centre x "to be coarsened" according to the following definition.

Definition 3.2 Let $\theta_{\mathrm{coarse}}, \theta_{\mathrm{refine}}$ be two tolerance values satisfying $0 < \theta_{\mathrm{coarse}} < \theta_{\mathrm{refine}}$. We refine a point $\xi \in \Xi$, and place it in N_{refine}, if and only if $\eta(\xi) > \theta_{\mathrm{refine}}$, and we move a point from the active centre set $\mathbf{X}$ into the coarse set $\mathbf{X}_{\mathrm{coarse}}$, if and only if corresponding $\eta(\xi) < \theta_{\mathrm{coarse}}$.

These two parameters $\theta_{\mathrm{coarse}}, \theta_{\mathrm{refine}}$ should be specified according to the user's need. Thus, we have two processes: coarsening where a coarse set $\mathbf{X}_{\mathrm{coarse}}$ is removed from the current centre set $\mathbf{X}$, that is the new centre set $\mathbf{X}$ is modified by replacing $\mathbf{X}$ with $\mathbf{X} \setminus \mathbf{X}_{\mathrm{coarse}}$; and refinement where a set of nodes $\mathbf{X}_{\mathrm{refine}}$ is added to the current centre set where the error is large; in other words, $\mathbf{X}$ is modified by replacing $\mathbf{X}$ with $\mathbf{X} \cup \mathbf{X}_{\mathrm{refine}}$.

When applying this error indicator, we require no extra evaluation of the target function so that no extra cost is paid in finding where approximation is likely to be poor. When function evaluation is very costly, this is a very positive feature of the method.

In mind of the above definitions, adaptive RBF interpolation is achieved by the following procedure:

(1) Centre set $\mathbf{X}_k$ and its corresponding indication set Ξ_k are specified.
(2) Global RBF approximation $S^{\mathrm{multi}}_{\mathbf{X}}$ is generated on the centre set $\mathbf{X}_k$, and the neighbourhood sets N_ξ for every ξ in Ξ are decided.
(3) The local RBF approximation $S^{\mathrm{ps}}_{N_\xi}$ is generated for each ξ, and the error indicator $\eta(\xi)$ is computed.
(4) The centre set $\mathbf{X}_k$ is updated by adding the refinement set $\mathbf{X}_{\mathrm{refine}}$ and deleting the coarse set $\mathbf{X}_{\mathrm{coarse}}$, that is $\{\mathbf{X}_{k+1} = \{\mathbf{X}_k \cup \mathbf{X}_{\mathrm{refine}}\} \setminus \mathbf{X}_{\mathrm{coarse}}$.

(5) When $\mathbf{X}_{\text{refine}} \cup \mathbf{X}_{\text{coarse}} = \emptyset$, the algorithm terminates. Otherwise, return to the first step.

Now, we describe the relationship between the centre set $\mathbf{X}_k$ and the corresponding indication set Ξ_k. In one-dimensional cases, the initial centre set $\mathbf{X}_0 = \{x_1, x_2, \cdots, x_{n_1}\}$ is a set of uniformly distributed nodes in the domain. For $k \geq 0$, the indication nodes in Ξ_k are the middle points of the current centres, that is $\Xi_k = \{\xi_i = 0.5(x_i + x_{i+1}), i = 1, 2, \cdots, n_k - 1\}$.

In two-dimensional cases, we follow the scheme described in [8] implemented in $[-1, 1]^2$, since any other rectangle domains can be transformed linearly into $[-1, 1]^2$.

In Fig. 1, we show the indicator set (red nodes) corresponding to the equally spaced points in the square (black nodes). The initial centres that consist of two types: (1) the interior nodes and (2) the boundary including four vertices. The red nodes are the indication set Ξ_0. Algorithm 1 describes the generation of the indicator set from the centre set more generally.

In three-dimensional cases, we extend the two-dimensional node scheme, the relationship between centre set $\mathbf{X}_k$ and indication set $\Xi_k, k = 0, 1, 2, \cdots$ following the same principal as in Algorithm 1.

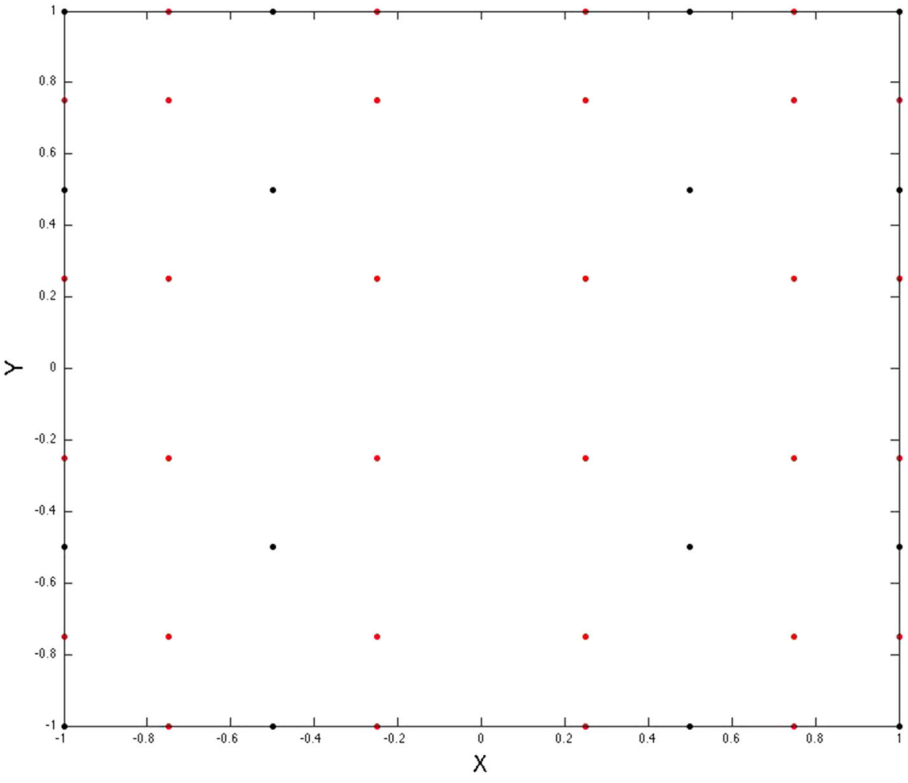

Fig. 1 $n = 2^j$, $j = 1$ in two dimensions with initial centre set $\mathbf{X}_0$ and Ξ_0

Algorithm 1 Calculate $\mathbf{X}_k, \Xi_k, k = 0, 1, 2, \cdots$ in $[-1, 1]^2$

Initialization: $n = 2^j$, $j \in \mathbf{N}$, $h = 2/n$, $k = 0$.
For interior nodes
$Inodes = \{(-1 + h/2 + rh, -1 + h/2 + sh),\ r, s = 0, 1, \cdots, n-1\}$.
For boundary nodes
$Bnodes = \{(-1 + h/2 + rh, \pm 1), (\pm 1, -1 + h/2 + rh),\ r = 0, 1, \cdots, n-1\}$.
$Pnodes = \{(\pm 1, \pm 1)\}$.
Initial center set $\mathbf{X}_0 \leftarrow Inodes \cup Pnodes \cup Bnodes$.
endflag = true
while endflag **do**
$k \leftarrow k + 1$
$h \leftarrow h/2$.
$Inodes_{\text{new}} \leftarrow$ Form squares centered at each node in $Inodes$ with boundary length h_k. Put the vertices of each square into $Inodes_{\text{new}}$.
$Bnodes_{\text{new}} \leftarrow$ Choose points on the boundary which are $0.5h$ away from the current nodes in $Bnodes$. Add these to $Bnodes_{\text{new}}$.
$\Xi_k \leftarrow Inodes_{\text{new}} \cup Bnodes_{\text{new}}$.
Use error indicator to decide the points at which to refine $\mathbf{X}_{\text{refine}} \subset \Xi_k$.
$Inodes \leftarrow Inodes \cup (\mathbf{X}_{\text{refine}} \cap Inodes_{\text{new}})$
$Bnodes \leftarrow Bnodes \cup (\mathbf{X}_{\text{refine}} \cap Bnodes_{\text{new}})$.
Use error indicator to locate the points need to be coarsen $\mathbf{X}_{\text{coarse}}$ in $\mathbf{X}_{k-1}$.
$\mathbf{X}^*_{k-1} \leftarrow \mathbf{X}_{k-1} \setminus \mathbf{X}_{\text{coarse}}$.
$\mathbf{X}_k \leftarrow \mathbf{X}^*_{k-1} \cup Inodes \cup Bnodes$.
if $\mathbf{X}^k_{\text{refine}} \cup \mathbf{X}^k_{\text{coarse}} = \emptyset$ **then**
endflag = false
end if
end while

4 Numerical results

In this section, we demonstrate the effectiveness of our error indicator in locating the worst errors, and in the corresponding refinement and coarsening strategy to improve the error. We do this to one-dimensional and two-dimensional test functions and compare to results in the papers [5, 8]. We also consider a three-dimensional example from finance, the option surface from the Black-Scholes model as we are interested in our methodology being applied in the finance world. In our examples, the multiquadric radial basis function $\phi(r) = \sqrt{(1 + c^2 r^2)}$ is applied to generate the global approximation $S^{\text{multi}}_{\mathbf{X}}$. The multiquadric RBF contains a free parameter c, the shape parameter. As we increase c, the basis function behaves more and more like the function cr, so that we get a sharp corner near to $r = 0$. We know from [15] that the conditioning of the interpolation problem increases with the smoothness of the basis function and with the proximity of points. Thus, in order to maintain control of the condition number of interpolation matrix, an adaptive shape parameter is applied, increasing the shape parameter as the distance between the centres decreases.

4.1 One-dimensional function adaptive interpolation

For one-dimensional test functions, we set the initial centre set $\mathbf{X}$ to be the uniformly distributed points in the interval $[-1, 1]$, and the indication set Ξ is the set of midpoints $\mathbf{X}$, as explained above. The neighbourhood set N_ξ contains $M = 4$ points. The shape parameter c of each centre is set to be a constant divided by the distance to the nearest neighbour, that is $c = 0.75/\text{distance}$. A test set T containing 5001 equally spaced nodes is used to test the approximation quality: $e_{\mathbf{X}} = \max_{t\in T} |f(t) - S_{\mathbf{X}}^{\text{multi}}(t)|$ and the root mean square value in $e_{\mathbf{X}}$, that is $\text{RMS}(e_{\mathbf{X}})$.

4.1.1 The Runge function

We first consider a standard approximation problem, the Runge function $f(x) = (1 + 25x^2)^{-1}$ on $[-1, 1]$. In Fig. 2, we see the final result obtained by adaptive interpolation, with $|\mathbf{X}| = 13$ initially, refinement threshold $\theta_{\text{refine}} = 2(-5) = 2 \times 10^{-5}$ and $\theta_{\text{coarse}} = \theta_{\text{refine}}/200$. We observe that centres cluster near the boundaries where approximation is more challenging due to the one-sided nature of the information, and at the origin, where the target function changes more rapidly. Note that the final maximum error is $1.4(-5)$ which is very close to θ_{refine} suggesting that our error indicator is working well. The largest condition number of this case is $3.1(+6)$.

In Table 1, we present the results of the adaptive process, which stops after eight iterations. The final interpolant $S_{\mathbf{X}}^{\text{multi}}$ has 83 centres, and the whole process computed a total of 85 evaluations of the target function. At each stage, N_{refine}, N_{coarse} are

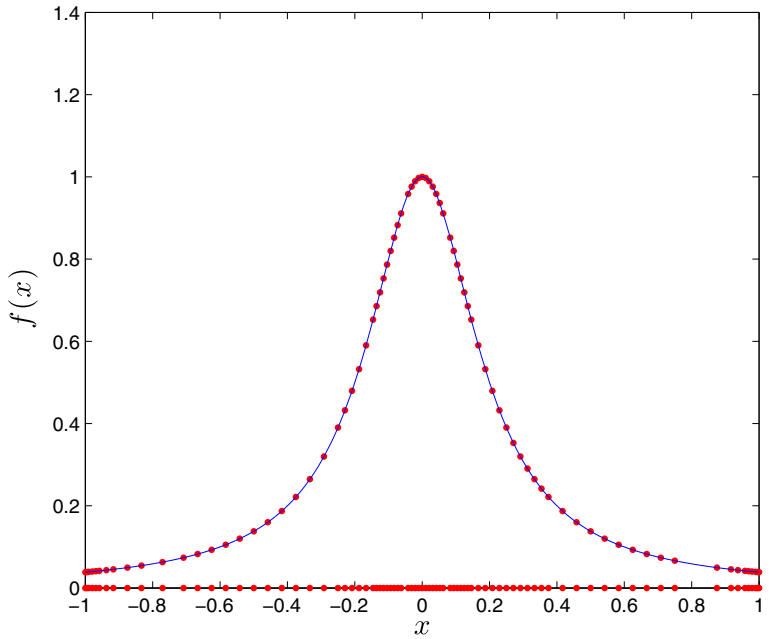

Fig. 2 Runge function with final RBF centre distribution, initial $|\mathbf{X}| = 13$, $\theta_{\text{refine}} = 2(-5)$

Table 1 Iterative process for the adaptive interpolation of Runge function with $\theta_{\text{refine}} = 2(-5)$

It	N_{total}	$\lvert\mathbf{X}\rvert$	N_{coarse}	N_{refine}	$e_{\mathbf{X}}(f)$	RMS$(e_{\mathbf{X}}(f))$	$\kappa(A)$
1	13	13	0	12	1.2(−2)	5.1(−3)	3.2(+3)
2	25	25	0	22	4.9(−4)	1.3(−4)	1.2(+4)
3	47	47	0	16	1.1(−4)	1.7(−5)	1.0(+5)
4	63	63	0	13	5.6(−5)	6.3(−6)	2.6(+5)
5	76	76	0	5	2.8(−5)	3.3(−6)	5.1(+5)
6	81	81	2	3	2.1(−5)	3.4(−6)	1.0(+6)
7	84	82	0	1	1.3(−5)	2.5(−6)	3.0(+6)
8	85	83	0	0	1.4(−5)	2.6(−6)	3.1(+6)

respectively the numbers of modes to be added/removed from the centre set. If we use the full centre set with 85 points to construct an interpolant, we get infinity and root mean square errors1.4(−5) and 1.4(−6), respectively, a small improvement on the error with 83 centres.

Figure 3 shows how the error decreases with the number of points in the set $\mathbf{X}$, starting at 13, and finishing with 646 centres; N_{total} is the total centres that sampled from target function, staring at 13 and finishing with 710. The final interpolant $S_{\mathbf{X}}^{\text{multi}}$ used 646 centres, with $e_{\mathbf{X}}(f) = 1.8(-8)$ and RMS$(e_{\mathbf{X}})(f) = 3.5(-9)$.

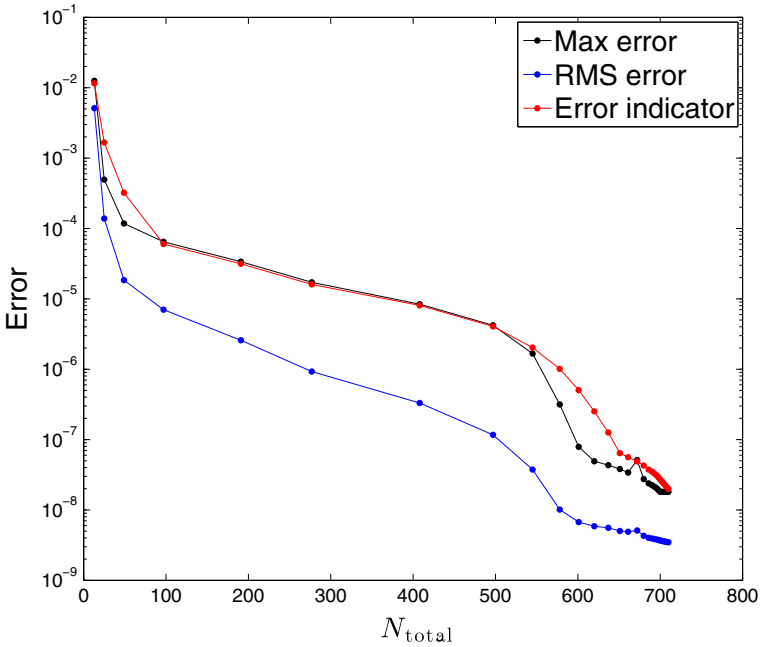

Fig. 3 Interpolation error at each iteration for the Runge function; N_{total} is the total number of samples of the target function, $\theta_{\text{refine}} = 2(-8)$

Using all the 710 centres, the interpolant $S^{\text{multi}}_{N_{\text{total}}}$ provide $e_{N_{\text{total}}}(f) = 1.8(-8)$ and $\text{RMS}(e_{N_{\text{total}}})(f) = 3.3(-9)$. The red nodes in Fig. 3 are the maximum values of the error indicator function at each iteration in absolute value, so we can see that the error indicator is a good measure of approximation error because the measured error (red line) tracks the approximation error (black line). We see, in this and all examples, that the rate of improvement of the error slows as the error approaches the tolerance θ_{refine}.

The condition numbers at each iteration is below $4(+11)$ due to the application of the adaptive shape parameter strategy. If we use the adaptive interpolation algorithm with a constant shape parameter in this example, the condition number of the interpolation matrix increases to $5(+20)$ after one or two iterations. While it has been observed that good approximation can be maintained with very large condition numbers, any user would, quite reasonably, doubt that such poor conditioning could lead to reliable results. Our goal is to provide answers that users can trust.

In [8], Driscoll and Heryudono use the residual sub-sampling method on the same example. They record the number of centres $|\mathbf{X}|$ used in the final interpolant, but the total numbers of function samples computed from the target function is not reported. In Table 2, we compare the results and the function evaluations needed for the residual sub-sampling method as implemented by the authors. We can see that residual sub-sampling achieves a better result marginally, but with a much larger number of function evaluations. We emphasise that our applications include examples where function evaluation is expensive.

4.1.2 The hyperbolic tan function

In this example, we consider $f(x) = \tanh(60x - 0.1)$. Table 3 shows the adaptive process of interpolation with threshold $\theta_{\text{refine}} = 2(-5)$. Our adaptive approximation converges in 9 iterations with final 82 nodes selected from 141 centres at which we compute the target function.

The final interpolant $S^{\text{multi}}_{\mathbf{X}}$ has error $e_{\mathbf{X}}(f) = 1.1(-5)$ and and $\text{RMS}(e_{\mathbf{X}})(f) = 1.8(-6)$. Using all the 141 centres, the interpolant $S^{\text{multi}}_{N_{\text{total}}}$ provide $e_{N_{\text{total}}}(f) = 3.2(-6)$ and $\text{RMS}(e_{N_{\text{total}}})(f) = 1.3(-7)$. Thus, depending on the user, one can have a more compact representation of the target function, guided by θ_{refine} and θ_{coarse}, or for a more accurate approximation using all points at which the target has been evaluated. The condition number grows quickly during the first few iterations, but never grows too large.

In Fig. 4, we see how the error indicator distributes centres around the steepest part of f. Figure 5 shows the adaptive process with $\theta_{\text{refine}} = 2(-8)$, starting with 13 centres. The algorithm stops with $|\mathbf{X}| = 595$ and $N_{\text{total}} = 726$ in 38 iterations.

Table 2 Error indicator versus residual sub-sampling for Runge function

Method	$e_{\mathbf{X}}(f)$	$\lvert\mathbf{X}\rvert$	N_{total}
Residual sub-sampling	$1.3(-5)$	53	285
Error indicator	$1.4(-5)$	83	85

Table 3 Iterative process of adaptive algorithm interpolation of $\tanh(60x - 0.1)$, with $\theta_{\text{refine}} = 2(-5)$

It	N_{total}	$\lvert\mathbf{X}\rvert$	N_{coarse}	N_{refine}	$e_{\mathbf{X}}(f)$	$\text{RMS}(e_{\mathbf{X}}(f))$	$\kappa(A)$
1	13	13	0	12	7.2(−1)	1.7(−1)	3.2(+3)
2	25	25	0	22	5.3(−1)	9.7(−2)	1.2(+4)
3	47	47	6	22	2.6(−1)	3.7(−2)	4.7(+4)
4	69	63	25	20	5.3(−2)	6.1(−3)	4.5(+5)
5	89	58	26	17	2.1(−3)	1.7(−4)	2.3(+6)
6	106	49	0	24	1.5(−4)	6.1(−5)	2.2(+7)
7	130	73	1	10	1.2(−5)	2.1(−6)	7.1(+6)
8	140	82	1	1	1.1(−5)	1.8(−6)	9.3(+6)
9	141	82	0	0	1.1(−5)	1.8(−6)	9.7(+6)

Notice that in this example, there is oscillation in the error related to the deletion of points and insertion of points in the refinement process. The more difficult the problem, the greater this oscillation can be (we refer you to the next example). The final interpolant $S_{\mathbf{X}}^{\text{multi}}$ has $e_{\mathbf{X}}(f) = 1.7(-8)$ and and $\text{RMS}(e_{\mathbf{X}})(f) = 2.1(-9)$, while the interpolant using all the available centres $S_{N_{\text{total}}}^{\text{multi}}$ gives uniform and root mean square errors $3.4(-8)$ and $9.8(-10)$, respectively. The condition number at each iteration is below $5(+8)$.

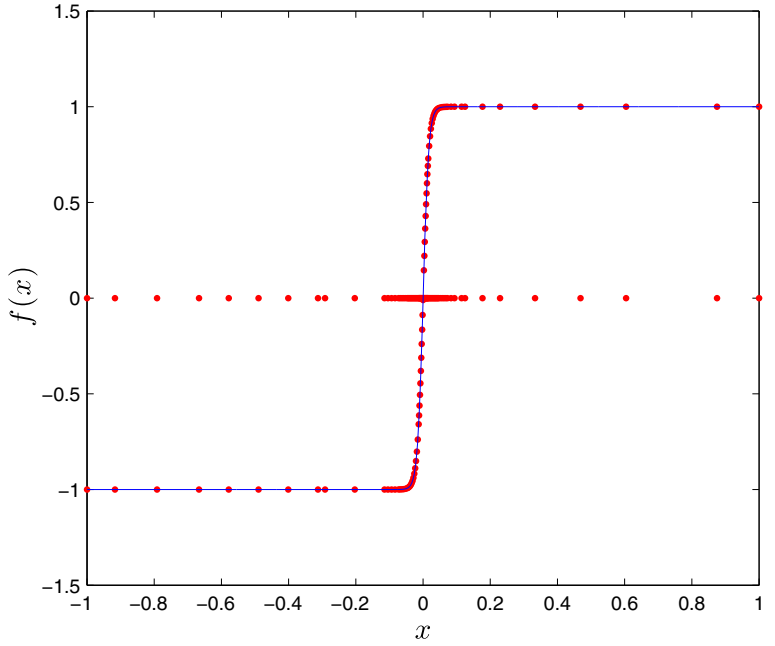

Fig. 4 Graph of $f(x) = \tanh(60x - 0.1)$ with the final distribution of centres produced by the algorithm, with $\theta_{\text{refine}} = 2(-5)$. The final number of centres used is 82

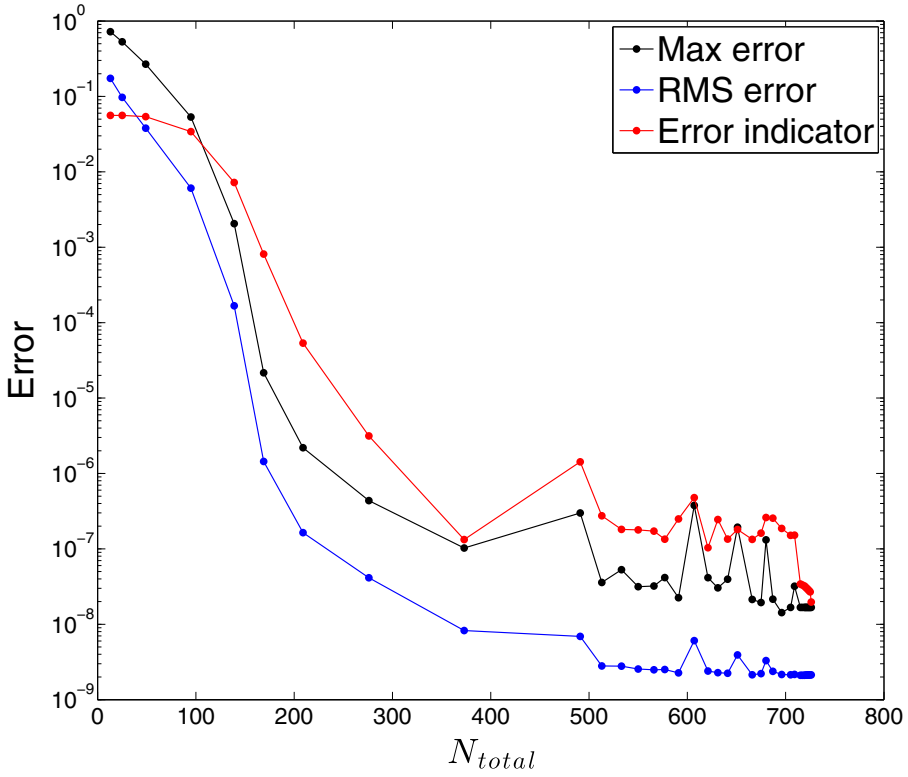

Fig. 5 Interpolation error at each iteration for $f(x) = \tanh(60x - 0.1)$ with $\theta_{\text{refine}} = 2(-8)$

Table 4 compares the results and the total number of function evaluations needed for the error indicator algorithm and residual sub-sampling algorithm. In this example, our algorithm achieves a better result with significantly less function evaluation.

4.1.3 The shifted absolute value function

Our final univariate example is $f(x) = |x - 0.04|$.

In Fig. 6, we see the centre distributed around the derivative discontinuity of $|x - 0.04|$. The final RBF representation uses 44 centres. Table 5 shows the adaptive process starting with 13 uniformly distributed centres, and ending with 44 centres. The total number of function evaluations was 121. The final interpolant $S_{\mathbf{X}}^{\text{multi}}$ has

Table 4 Error indicator versus residual sub-sampling for $f(x) = \tanh(60x - 0.1)$

Method	$e_{\mathbf{X}}(f)$	$\lvert\mathbf{X}\rvert$	N_{total}
Residual sub-sampling	2.5(-5)	129	698
Error indicator	1.1(-5)	82	141

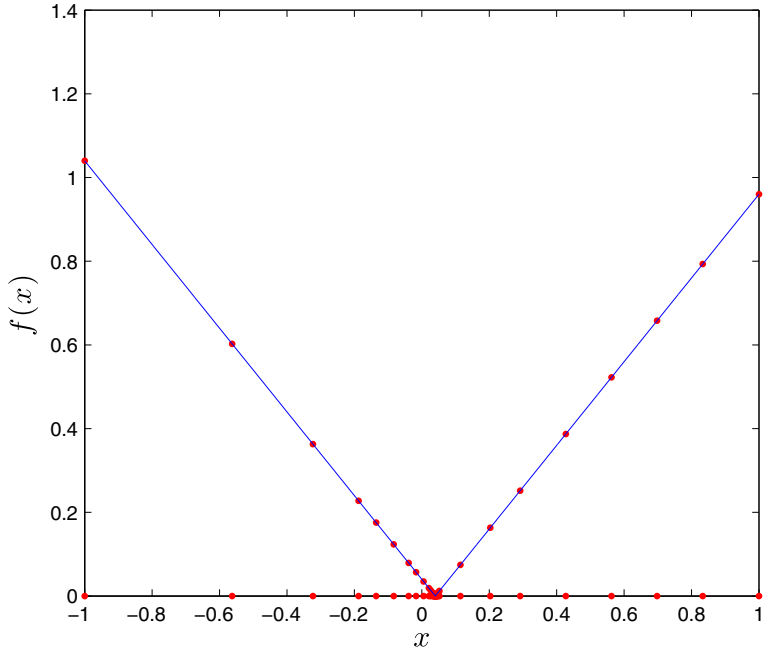

Fig. 6 Final centre distribution (44 points) for approximating $f(x) = |x - 0.04|$ with $\theta_{\text{refine}} = 2(-8)$

infinity and root mean square errors of 3.9(−5) and 2.7(−6), respectively, while using all the 121 centres, we obtain uniform and root mean square errors of 3.9(−5) and 6.0(−7), respectively.

Table 5 Iterative process of adaptive algorithm interpolation of $f(x) = |x - 0.04|$, with $\theta_{\text{refine}} = 2(-5)$

It	N_{total}	$\lvert\mathbf{X}\rvert$	N_{coarse}	N_{refine}	$e_{\mathbf{X}}(f)$	RMS$(e_{\mathbf{X}}(f))$	$\kappa(A)$
1	13	13	0	12	3.7(−2)	6.3(−3)	3.2(+3)
2	25	25	0	16	2.7(−2)	3.0(−3)	1.2(+4)
3	41	41	3	14	7.1(−3)	9.8(−4)	5.3(+4)
4	55	52	18	12	3.3(−3)	2.9(−4)	3.4(+5)
5	67	46	9	15	1.6(−3)	2.1(−4)	1.1(+8)
6	82	52	23	6	1.4(−3)	4.1(−5)	9.2(+6)
7	88	35	6	14	7.7(−4)	9.2(−5)	1.3(+7)
8	102	43	6	5	3.3(−4)	6.3(−6)	2.0(+7)
9	107	42	5	10	1.1(−3)	4.2(−4)	1.0(+9)
10	117	47	3	4	5.2(−5)	2.1(−6)	6.2(+7)
11	121	48	4	0	3.8(−5)	1.8(−6)	8.6(+7)
12	121	44	0	0	3.8(−5)	2.7(−6)	7.5(+7)

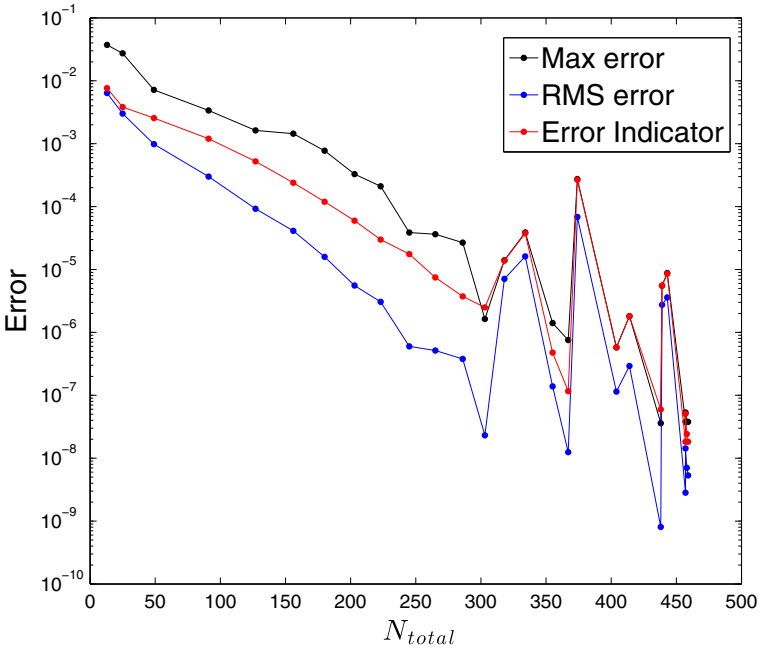

Fig. 7 Error convergence of the adaptive algorithm for $f(x) = |x - 0.04|$ and $\theta_{\text{refine}} = 2(-8)$

In Fig. 7, we show the progress of the adaptive algorithm starting with 13 centres, and $\theta_{\text{refine}} = 2(-8)$. We can see more extreme oscillations of the error in the process than in the previous two examples, indicating that this is a more difficult problem. The algorithm terminates after 27 iterations with $|\mathbf{X}| = 81$. The total number of points used N_{total} starts at 13 and stops at 459. The final interpolant $S^{\text{multi}}_{\mathbf{X}}$ with 81 centres has maximum error $e_{\mathbf{X}}(f) = 3.8(-8)$ and $\text{RMS}(e_{\mathbf{X}})(f) = 5.3(-9)$. The condition number at each iteration is below $2(+11)$. The interpolant using all the available centres $S^{\text{multi}}_{N_{\text{total}}}$ gives infinity error $3.8(-8)$ and root mean square error $5.3(-10)$. The condition numbers for this interpolation is $1.8(+13)$.

Table 6 compares the results and function evaluations required for the error indicator algorithm and the residual sub-sampling algorithm. We have needed to choose a tolerance which generates a similar number of points in the final representation.

Table 6 Error indicator versus residual sub-sampling for $f(x) = |x - 0.04|$

Method	$e_{\mathbf{X}}(f)$	$\lvert\mathbf{X}\rvert$	N_{total}
Residual sub-sampling	1.5(-5)	53	878
Error indicator	1.9(-6)	55	196

4.2 Two-dimensional function adaptive interpolation

We now consider five two-dimensional examples, where the node refinement scheme explained above is applied. We set $j = 3$ in the initialisation step of Algorithm 1 to achieve the initial centre set $\mathbf{X}_1$, with $|\mathbf{X}_1| = 100$, and its indication set Ξ_1. The neighbourhood set N_ξ has $M = 24$ neighbours. A test grid T of 101×101 uniformly spaced nodes on $[-1, 1]^2$ is used to test the approximation quality: $e_{\mathbf{X}} = \max_{t\in T} |f(t) - S^{\text{multi}}_{\mathbf{X}}(t)|$ with root mean square error $\text{RMS}(e_{\mathbf{X}})$. The shape parameter c for each centre is set to be a constant divided by its distance to the nearest neighbour, as in the univariate case: $c = 0.5/\text{distance}$, and $\theta_{\text{coarse}} = \theta_{\text{refine}}/100$.

4.2.1 The Franke function

The Franke function (the first panel of Fig. 8)

$$f(x, y) = \exp^{-0.1(x^2+y^2)} + \exp^{-5((x-0.5)^2+(y-0.5)^2)} + \exp^{-15((x+0.2)^2+(y+0.4)^2)} + \exp^{-9((x+0.8)^2+(y-0.8)^2)} \quad (4.1)$$

is a standard test function for RBF approximation. With $\theta_{\text{refine}} = 5.0(-4)$, only 14 iterations are needed to reach the stopping criteria. The second panel in Fig. 8 shows the final node distribution and demonstrates that the error indicator locates points in regions of rapid variation. In this case, we have $|\mathbf{X}| = 1318$ centres with max error $7.2(-4)$, and in the process, all the condition numbers are below $2.1(+7)$.

In Table 7, we show results corresponding to different values of θ_{refine}. We use $\kappa(A)_{\max}$ to represent the largest value of $\kappa(A)$ observed during the adaptive process.

In Fig. 9, we see how the maximum and root mean square error decrease with the pre-set threshold, and the number of points required to achieve the given threshold. We see that the error decays approximately like $|\mathbf{X}|^{-1}$.

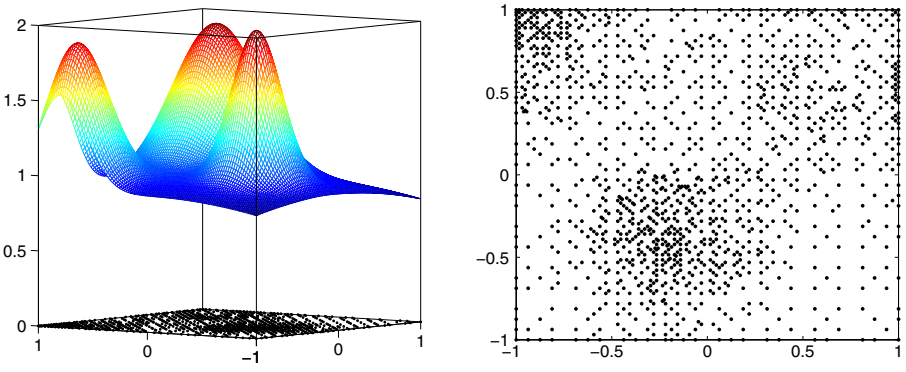

Fig. 8 Centre distribution for adaptive interpolation for the Franke function with $\theta_{\text{refine}} = 5.0(-4)$. The number of points in this centre set is 1318

Table 7 Adaptive algorithm interpolation results of Franke function with different θ_{refine}

θ_{refine}	N_{total}	$\lvert\mathbf{X}\rvert$	$e_{\mathbf{X}}(f)$	RMS$(e_{\mathbf{X}}(f))$	$\kappa(A)_{\max}$
1.0(−3)	697	697	2.2(−3)	1.7(−4)	6.1(+8)
7.5(−4)	907	907	1.4(−3)	9.9(−5)	1.1(+7)
5.0(−4)	1319	1318	7.2(−4)	6.3(−5)	2.1(+7)
2.5(−4)	2703	2702	6.3(−4)	3.8(−5)	7.6(+7)
1.0(−4)	6693	6692	2.1(−4)	1.3(−5)	3.4(+8)
7.5(−5)	8823	8820	1.1(−4)	8.4(−6)	5.8(+8)

4.2.2 The two-dimensional hyperbolic tan function

The second test function is $f(x, y) = -0.4\tanh(20xy) + 0.6$ on $[-1, 1]^2$. With $\theta_{\text{refine}} = 5.0(-4)$, the algorithm took eight iterations to reach the stopping criteria. A total of $|\mathbf{X}| = 2106$ centres were used to give an error $e_{\mathbf{X}}(f) = 5.4(-5)$. All the condition numbers were below 5.5(+7). In Table 8, we see how the number of points need by the algorithm varies with the choice of θ_{refine} (Figs. 10 and 11).

The observant reader will notice that the final error may not decrease with the choice of the error indicator, since we have a decrease in θ_{refine} in the last two rows of Table 8, but an increase in maximum error. This may happen as the indicator we use is only that—an indicator. However, we observe that the trend is decreasing, so

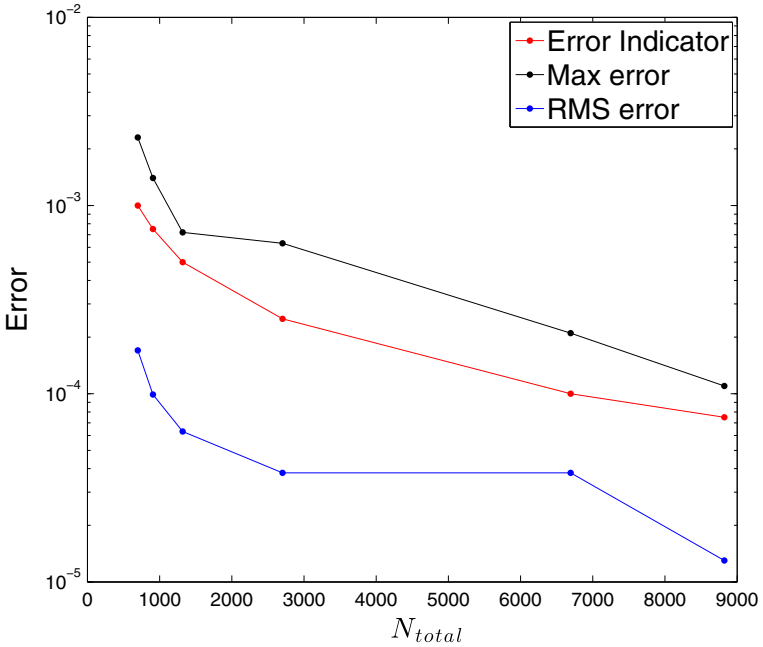

Fig. 9 Convergence of adaptive interpolation for the Franke function

Table 8 Adaptive algorithm interpolation results of $f(x, y) = -0.4\tanh(20xy) + 0.6$ with different θ_{refine}

θ_{refine}	N_{total}	$\lvert\mathbf{X}\rvert$	$e_{\mathbf{X}}(f)$	RMS$(e_{\mathbf{X}}(f))$	$\kappa(A)_{\max}$
1.0(-3)	1605	1561	4.2(-3)	1.6(-4)	2.9(+7)
7.5(-4)	1810	1776	2.8(-3)	1.1(-4)	3.7(+7)
5.0(-4)	2176	2106	5.4(-4)	5.6(-5)	5.5(+7)
2.5(-4)	3911	3840	2.3(-4)	2.8(-5)	1.4(+8)
1.0(-4)	9168	9080	1.2(-4)	1.1(-5)	7.5(+8)
7.5(-5)	12144	12078	1.4(-4)	9.4(-6)	1.4(+9)

that in a global sense, a decrease of the threshold results in a decrease in errors. The decrease is again of the order of $|\mathbf{X}|^{-1}$.

4.2.3 The two-dimensional exponential function

In this example, $f(x, y) = \exp(-60((x-0.35)^2+(y-0.25)^2))+0.2$ in $[-1, 1]^2$. In Table 9, we see how the error of adaptive interpolation depends on the error indicator. Figure 12 shows how the error indicator puts more centres in the region where the function changes rapidly.

In the two previous examples, there is no big difference in $|\mathbf{X}|$ and N_{total}. In this example, there is notable difference between $|\mathbf{X}|$ and N_{total}. With $\theta_{\text{refine}} = 7.5(-5)$, when using all the available centres to construct $S^{\text{multi}}_{N_{\text{total}}}$, we get better approximation quality with $8.4(-5)$ and $9.8(-6)$, respectively, for uniform and root mean square error (Fig. 13).

The results presented here using the the error indicator are comparable and sometimes improve upon the results generated by residual sub-sampling method in [8]. In particular, where we wish to limit the number of function evaluations, we demonstrate a significant saving.

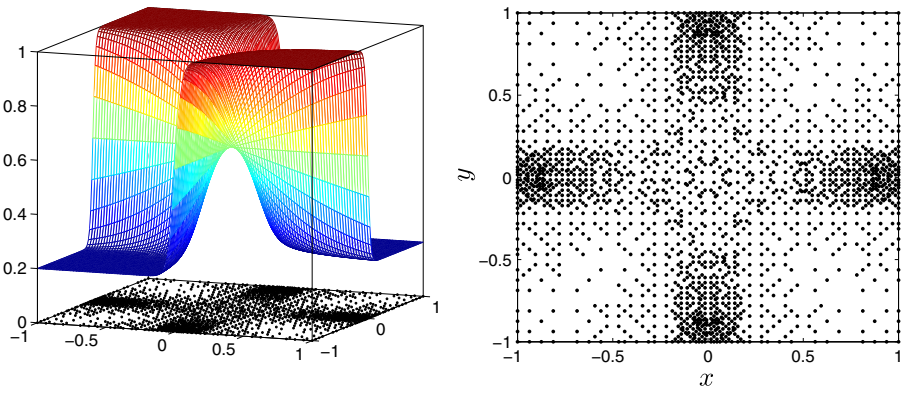

Fig. 10 Final node distribution for approximation of $f(x, y) = -0.4\tanh(20xy) + 0.6$ with $\theta_{\text{refine}} = 1(-4)$

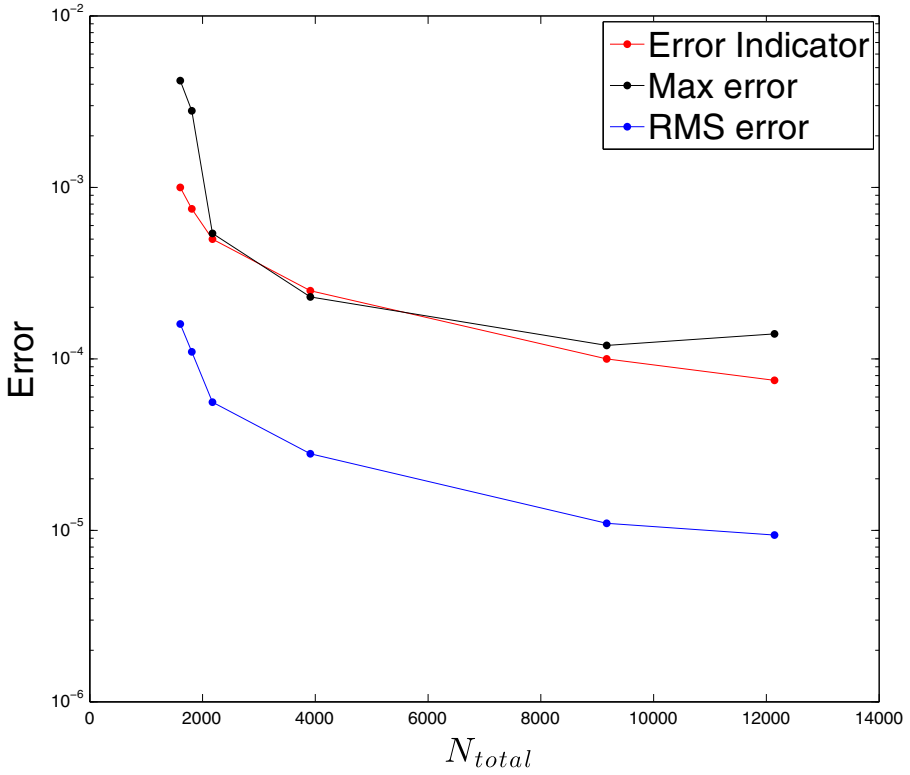

Fig. 11 Error versus number of points for approximation of $f(x, y) = -0.4 \tanh(20xy) + 0.6$

4.2.4 The cone shape function

In one-dimensional cases, we see the shifted absolute function $f(x) = |x - 0.04|$ is hard to approximate due to the derivative singularity at $x = 0.04$. In this example, we explore the same singularity in two dimensions $f(x, y) = \sqrt{x^2 + y^2} + 0.2$ (the first panel of Fig. 14).

Table 9 Error in adaptive interpolation of $f(x, y) = \exp(-60((x - 0.35)^2 + (y - 0.25)^2)) + 0.2$ with different θ_{refine}

θ_{refine}	N_{total}	$\lvert\mathbf{X}\rvert$	$e_{\mathbf{X}}(f)$	RMS$(e_{\mathbf{X}}(f))$	$\kappa(A)_{\max}$
1(−3)	594	476	2.6(−4)	4.5(−5)	8.4(+6)
7.5(−4)	776	650	2.6(−4)	3.8(−5)	1.3(+7)
5(−4)	1078	933	2.6(−4)	3.3(−5)	1.4(+7)
2.5(−4)	1660	1511	2.3(−4)	2.6(−5)	1.4(+7)
1(−4)	3483	3324	1.6(−4)	1.5(−5)	1.4(+8)
7.5(−5)	4516	4335	9.5(−5)	1.1(−5)	2.2(+8)

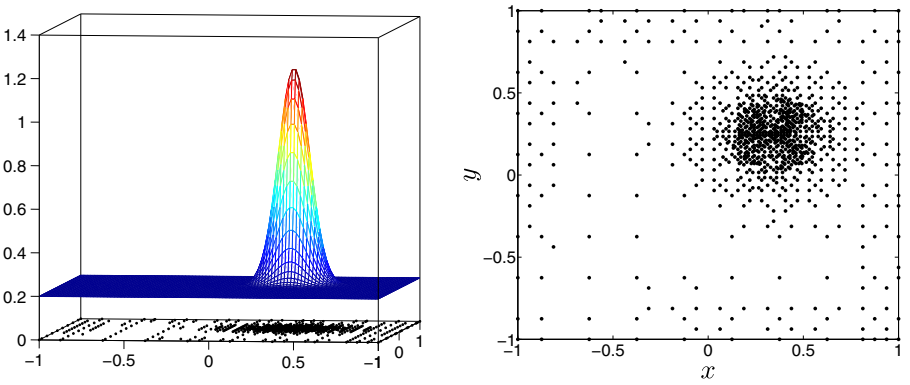

Fig. 12 Final node distribution for approximation of $f(x, y) = \exp(-60((x-0.35)^2+(y-0.25)^2))+0.2$ with $\theta_{\text{refine}} = 7.5(-5)$

With $\theta_{\text{refine}} = 7.5(-5)$, only 11 iterations are needed to reach the stopping criteria. The second panel of Fig. 14 shows the final node distribution and demonstrates that the error indicator places points near to the singularity. We have $|\mathbf{X}| = 2050$ centres with max error $7.8(-4)$. The condition numbers are below $5.6(+7)$. In Table 10, we see results corresponding to different values of θ_{refine}.

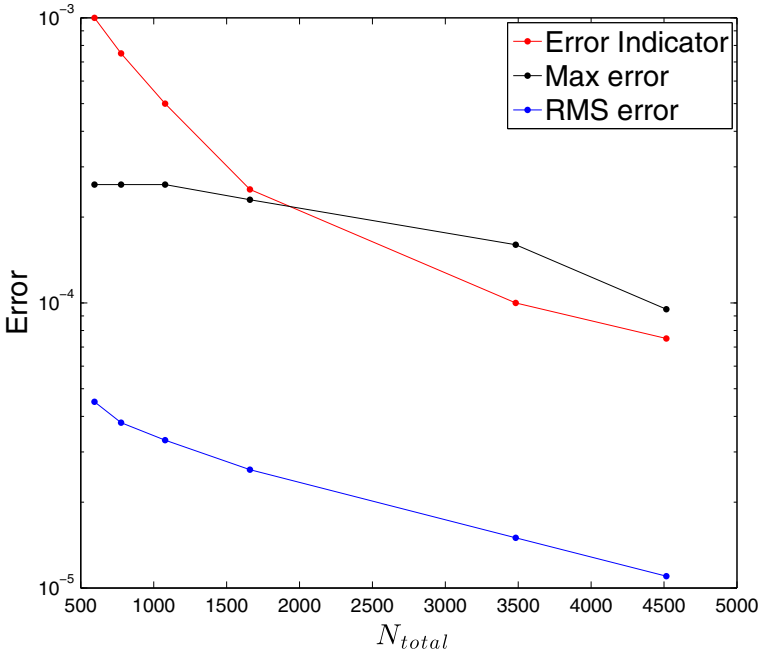

Fig. 13 Error versus number of points for approximation of $f(x, y) = \exp(-60((x - 0.35)^2 + (y - 0.25)^2)) + 0.2$

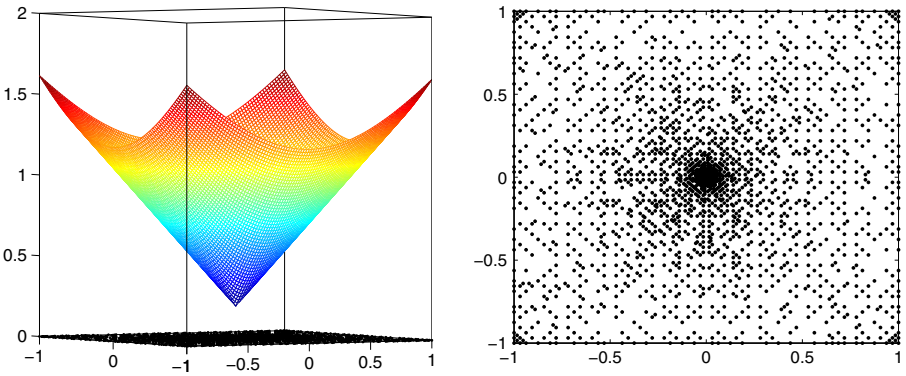

Fig. 14 Final node distribution for approximation of $f(x, y) = \sqrt{x^2 + y^2} + 0.2$ with $\theta_{\text{refine}} = 7.5(-5)$

In the left panel of Fig. 15, we show the process of approximation with $\theta_{\text{refine}} = 2.5(-5)$; the black line is the maximum error, the red line is the maximum error indicator value and the blue line is the root mean square error. We see that the in the latter part of the approximation process, the error decays much faster than in the beginning, demonstrating an acceleration of the accuracy as the singularity becomes better resolved. In the right panel of Fig. 15, we see how the maximum and root mean square error decrease with the pre-set threshold and the number of points required to achieve the given threshold. We see, as in the two-dimensional examples, that the error decays approximately like $|\mathbf{X}|^{-1}$.

4.2.5 The Lena image case

In our previous examples, we have approximated functions with rapid variation or derivative singularities. These are conventional examples in which we have seen that

Table 10 Error in adaptive interpolation of $f(x, y) = \sqrt{x^2 + y^2} + 0.2$ with different θ_{refine}

θ_{refine}	N_{total}	$\lvert\mathbf{X}\rvert$	$e_{\mathbf{X}}(f)$	$\text{RMS}(e_{\mathbf{X}}(f))$	$\kappa(A)_{\max}$
1.0(−3)	269	269	6.3(−3)	9.6(−5)	1.9(+6)
7.5(−4)	350	350	6.3(−3)	9.0(−5)	2.2(+6)
5.0(−4)	406	406	3.1(−3)	4.3(−5)	4.4(+6)
2.5(−4)	712	712	1.6(−3)	2.1(−5)	1.0(+7)
1.0(−4)	1752	1752	7.8(−4)	9.6(−6)	4.4(+7)
7.5(−5)	2050	2050	7.8(−4)	1.2(−5)	5.6(+7)
5.0(−5)	3267	3267	3.9(−4)	5.4(−6)	1.5(+8)
2.5(−5)	6346	6346	1.9(−4)	2.6(−6)	4.2(+8)

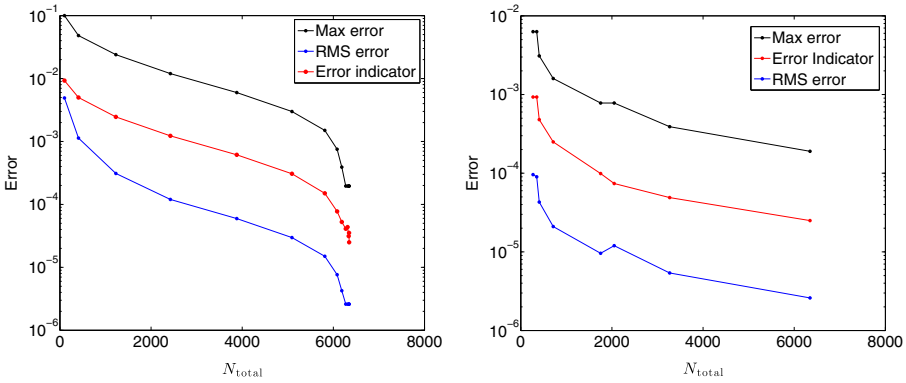

Fig. 15 Error versus number of points for approximation of $f(x, y) = \sqrt{x^2 + y^2} + 0.2$

the adaptive error indicator RBF approximation method delivers good accuracy with the aim of minimising the number of function evaluation used. In this case, the Lena picture (see Fig. 16 left panel) is used as the target. Here, the function is a 128×128 pixel image, so is discrete, i.e. it has discontinuities everywhere. For our method, we use local RBF approximation to compute an approximation to the image between the centres of pixels (we term this an emulator); (Fig. 16 right panel). This emulator is our target function.

In Table 11, we show three approximation results, and Fig. 17 shows the corresponding reconstructed images. In this case, we could see that adaptive error indicator method can deal with this more complicated target function.

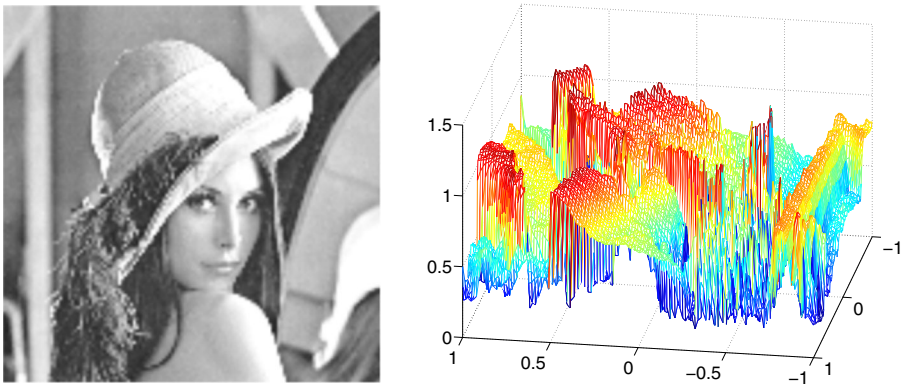

Fig. 16 The original Lena picture and its emulator function S_P^{multi}

Table 11 Error in adaptive interpolation of S_P^{multi} with different θ_{refine}

θ_{refine}	N_{total}	$\lvert\mathbf{X}\rvert$	$e_{\mathbf{X}}(f)$	$\text{RMS}(e_{\mathbf{X}}(f))$	$\kappa(A)_{\max}$
2.0(−2)	3604	3603	6.4(−1)	7.5(−2)	6.6(+6)
1.5(−2)	6703	6702	5.3(−1)	5.3(−2)	1.6(+7)
1.2(−2)	9711	9710	3.1(−1)	3.9(−2)	2.6(+7)

4.3 Three-dimensional adaptive interpolation

We can extend the two-dimensional node adaptation scheme to three dimensions. We begin with the uniformly distributed centres X in $[-1, 1]^3$. The corresponding indication set Ξ for the centres set. Set $j = 2$ in the initialisation step of Algorithm 1 to achieve the initial centre set $\mathbf{X}_1$, with $|\mathbf{X}_1| = 208$. The corresponding indication set is Ξ_1 and the neighbourhood set N_ξ parameter M is set to 60.

A test set T containing 25,000 Halton nodes is used to test the infinity and root means square errors. The shape parameter c of each centre is set to be a constant divided by its distance to the nearest neighbour, that is $c = 1/\text{distance}$, and $\theta_{\text{coarse}} = \theta_{\text{refine}}/1000$.

4.3.1 The 3D exponential function

In this example,

$$f(x, y, z) = \exp(-81/16((x - 0.5)^2 + (y - 0.5).^2 + (z - 0.5)^2))/3.$$

In Table 12, we show the results of adaptive interpolation. In Fig. 18, we see how the maximum error behaves with regard to θ_{refine}. The maximum error (black) tracks the error indicator (red) well. The condition numbers observed in the algorithm remain relatively small.

If we compare the results in Table 12 to those in [7], where a variety of different RBFs (thin-plate spline, cubic RBF, Wendland function) are used on a grid, we provide much better accuracy for the same number of centres. In Table 13, we see

Fig. 17 Three approximations of Lena picture

Table 12 $\exp(-81/16((x-0.5)^2+(y-0.5).^2+(z-0.5)^2))/3$ interpolation results by error indicator

θ_{refine}	N_{total}	$\lvert\mathbf{X}\rvert$	$e_{\mathbf{X}}(f)$	$\text{RMS}(e_{\mathbf{X}}(f))$	$\kappa(A)_{\max}$
5(−3)	333	333	3.6(−3)	6.1(−4)	7.4(+1)
1(−3)	665	665	1.3(−3)	1.4(−4)	1.5(+4)
5(−4)	1419	1419	5.6(−4)	5.8(−5)	5.6(+4)
1(−4)	6643	6643	1.7(−4)	1.4(−5)	4.1(+6)

the results achieved with uniform centres with the multiquadric basis function. If we compare Tables 12 and 13, we see that the error indicator-adaptive algorithm put more centres in the region where it is difficult to approximate, thus providing better approximation quality for the same number of centres.

In Fig. 18, we see the rate of decay of the error with the number of points. This suggest a convergence rate of order $|\mathbf{X}|^{-1}$.

4.3.2 The European call option

The Black Scholes equation is used to describe the the price of a call option over time. The equation is as follows:

$$\frac{\partial V}{\partial t}+\frac{1}{2}\sigma^2 S^2\frac{\partial^2 V}{\partial S^2}+rS\frac{\partial V}{\partial S}-rV=0.$$

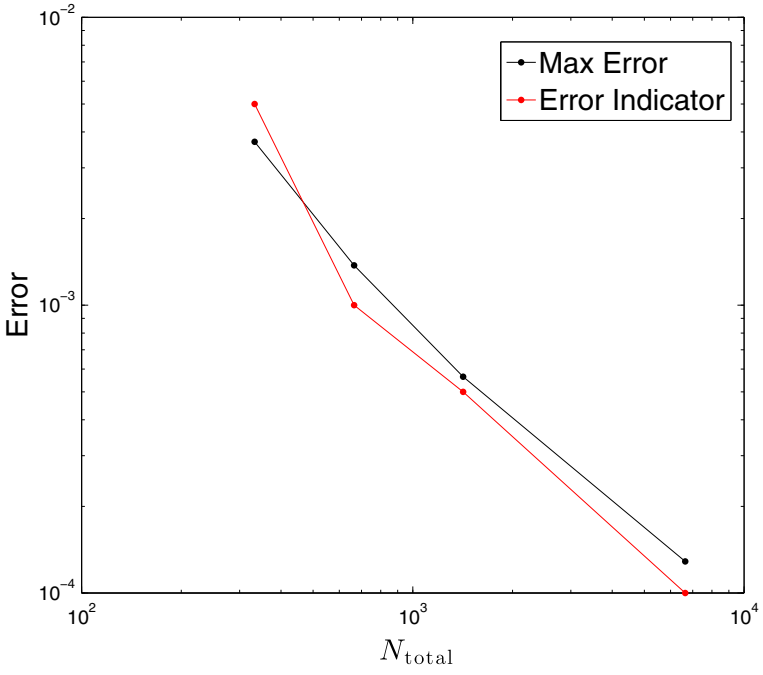

Fig. 18 Error convergence of interpolation results for $\exp(-81/16((x-0.5)^2+(y-0.5).^2+(z-0.5)^2))/3$

Table 13 $\exp(-81/16((x-0.5)^2+(y-0.5).^2+(z-0.5)^2))/3$ interpolation results by uniform centres

N_{uniform}	$e_{\mathbf{X}}(f)$
343	2.0(−2)
729	2.8(−3)
1728	1.4(−3)
6859	6.7(−4)

The value of a European call option C for a non-dividend-paying underlying stock in terms of the Black Scholes parameters is as follows:

$$C = N(d_1)S - N(d_2)K\exp(-r(T-t))$$

$$d_1 = \frac{1}{\sigma\sqrt{T-t}}\left(\ln\left(\frac{S}{K}\right)+\left(r+\frac{\sigma^2}{2}\right)(T-t)\right)$$

$$d_1 = \frac{1}{\sigma\sqrt{T-t}}\left(\ln\left(\frac{S}{K}\right)+\left(r-\frac{\sigma^2}{2}\right)(T-t)\right),$$

where

- $N(.)$ is s the cumulative distribution function of the standard normal distribution,
- $T-t$ is the time to maturity,
- S is the spot price of the underlying asset,
- K is the strike price,
- r is the risk-free rate (annual rate, expressed in terms of continuous compounding),
- σ is is the volatility of returns of the underlying asset.

The European call option price C can be thought of as a function of six variables $C(S, r, \sigma, K, T$ and $t)$. If we specify three variables $K = 100, T = 1$ and $t = 0$, we have a function $C(S, r, \sigma)$ of three variables. In Table 14, we show the results

Table 14 Adaptive interpolation of $C(S, r, \sigma)$

θ_{refine}	N_{total}	$\lvert\mathbf{X}\rvert$	$e_{\mathbf{X}}(f)$	$\text{RMS}(e_{\mathbf{X}}(f))$	$\kappa(A)_{\max}$
5(−3)	1910	1910	7.7(−3)	1.3(−3)	1.5(+6)
2.5(−3)	3595	3595	3.4(−3)	6.2(−4)	4.1(+6)
1(−3)	8914	8914	1.2(−3)	2.3(−4)	1.5(+7)
7.5(−4)	11985	11985	1.1(−3)	1.6(−4)	2.3(+7)
5(−4)	18283	18283	6.0(−4)	1.4(−5)	3.6(+7)

Table 15 Approximation errors interpolating $C(S, r, \sigma)$ using uniform centres

N_{uniform}	$e_{\mathbf{X}}(f)$
2197	4.9(−2)
4096	3.7(−2)
9261	2.2(−2)
12,167	1.8(−2)
19,683	1.1(−2)

of adaptive interpolation to this function in $r \in [0.01, 0.05], \sigma \in [0.1, 0.3]$ and $S \in [90, 110]$. For a comparison, in Table 15, we show the approximation quality using uniform centres in the parameter space.

In Fig. 19, we see how the error behave with respect to the number of points. We see the error indicator (red line) tracks the maximum error (black line) well. The algorithm appears to give convergence of order $|\mathbf{X}|^{-1}$ as in the previous example. The condition number remains of moderate size throughout the algorithm, and the adaptive algorithm is many times better than the results of using gridded data.

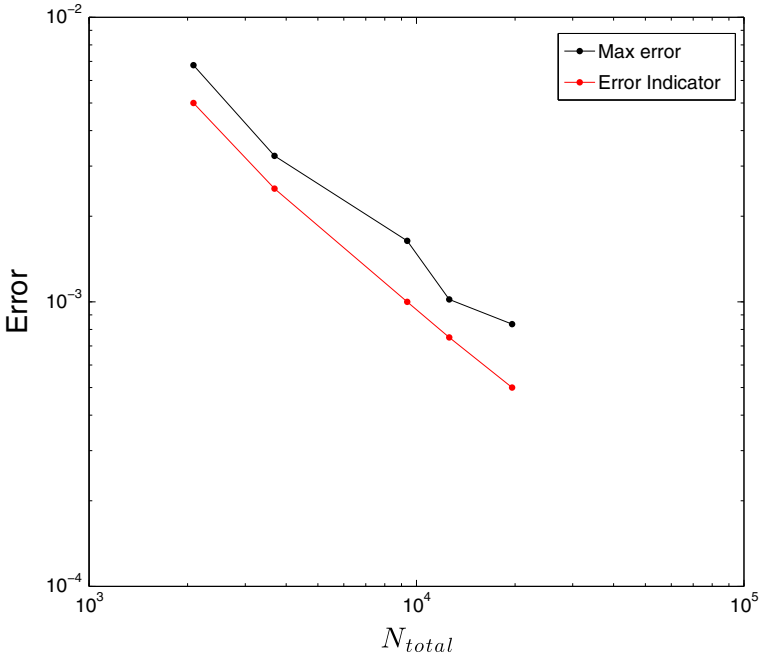

Fig. 19 Error versus points for adaptive interpolation of $C(S, r, \sigma)$

Table 16 Adaptive error indicator interpolation results of Runge function with different parameter settings, with $\theta_{\text{refine}} = 2.0(-5)$

Parameter settings	N_{total}	$e_{\mathbf{X}}(f)$	RMS$(e_{\mathbf{X}}(f))$	$\kappa(A)_{\max}$
$a = 0.5, M = 3$	140	1.6(−5)	3.4(−6)	2.0(+7)
$a = 0.5, M = 5$	88	1.9(−5)	3.7(−6)	7.4(+6)
$a = 1, M = 6$	79	2.1(−5)	3.3(−6)	4.3(+5)
$a = 1, M = 8$	74	4.0(−5)	5.8(−6)	8.7(+5)

4.4 Stability discussion

In this section, we explore the sensitivity of the algorithm to the parameters which need to be chosen in the algorithm. The user should be reassured that the final output is not strongly dependent on perfect choice of parameters.

The different components of the adaptive error indicator method are the global interpolant $S^{\text{multi}}_{\mathbf{X}_k}$, the local interpolant $S^{\text{ps}}_{N_\xi}$ and the Algorithm 1 which generates the sample set $\mathbf{X}_k$ and the corresponding indication set Ξ_k. In the previous section, the adaptive error indicator method has been applied to different target functions, and the approximation results show that, with the parameter choices made, the algorithm delivers good approximations. The parameters chosen for our experiments we call Parameter Set 1. The values for Parameter Set 1 are summarised below:

1. The shape parameter at the ith point, c_i in the global interpolant $S^{\text{multi}}_{\mathbf{X}_k}$, it is set by $c_i = a/d_i$, where d_i is the distance to the nearest neighbour. Then, $a = 0.75$ for one-dimensional cases, $a = 0.5$ for two-dimensional cases and $a = 1$ for three-dimensional cases.
2. The number of points M in set N_ξ for the local interpolant $S^{\text{ps}}_{N_\xi}$. In the one - dimensional cases, $M = 4$, in the two-dimensional cases, $M = 24$ and in three-dimensional cases, $M = 60$. There is no shape parameter for $S^{\text{ps}}_{N_\xi}$.

Parameter Set 1 is not the optimal choice for parameters, since for different functions f, the optimal parameter choices might change. We will show the robustness of

Table 17 Adaptive error indicator interpolation results of Runge function with different parameter settings, with $\theta_{\text{refine}} = 2.0(-8)$

Parameter settings	N_{total}	$e_{\mathbf{X}}(f)$	RMS$(e_{\mathbf{X}}(f))$	$\kappa(A)_{\max}$
$a = 0.5, M = 3$	1346	2.3(−8)	8.3(−10)	1.8(+11)
$a = 0.5, M = 5$	814	1.6(−8)	9.1(−10)	1.5(+11)
$a = 1, M = 6$	1162	2.5(−8)	5.9(−9)	3.8(+10)
$a = 1, M = 8$	1211	1.9(−8)	5.3(−9)	4.0(+10)

Table 18 Adaptive error indicator interpolation results of Franke function with different parameter settings, with $\theta_{\text{refine}} = 1.0(-3)$

Parameter settings	N_{total}	$e_{\mathbf{X}}(f)$	RMS$(e_{\mathbf{X}}(f))$	$\kappa(A)_{\max}$
$a = 2, M = 20$	1128	1.7(-3)	2.7(−4)	3.0(+5)
$a = 2, M = 30$	851	2.2(-3)	3.4(−4)	2.1(+5)
$a = 5, M = 35$	1060	2.4(-3)	4.0(−4)	1.3(+5)
$a = 5, M = 40$	1019	2.4(-3)	4.4(−4)	1.3(+5)

adaptive error indicator method by varying the parameters from Parameter Set 1, and observing that results do not change too much.

Table 16 shows the results of approximating the Runge function $f(x) = (1 + 25x^2)^{-1}$ (see Section 4.1.1) with different parameter settings for $\theta_{\text{refine}} = 2.0(-5)$. Table 17 shows the approximation results with different parameter settings for $\theta_{\text{refine}} = 2.0(-8)$. We observe that the approximation degrades as a result of increasing the shape parameter, at the same time as the condition number decreases. This is what standard theory suggests. We also see that a change in the indication set size beyond 5 makes little difference. The number of points required is a reflection of the approximation accuracy, and this is governed by the choice of shape parameter. The balance here is between the amount of ill-conditioning that one is prepared to accept, and the accuracy.

Table 18 shows the approximation results for the two-dimensional Franke function (see Section 4.2.1) approximation results with different parameter settings with $\theta_{\text{refine}} = 1.0(-3)$. Table 19 shows the approximation results with different parameter settings with $\theta_{\text{refine}} = 5.0(-4)$. In these tables, there is a strong correlation between number of points and error once we have enough points in the indicator set. An increase in the shape parameter does lead to an increase in error, and a decrease in condition number, though less extreme than in one dimension.

These results are similar to the results generated by adaptive error indicator method with Parameter Set 1, and the effect of changes in parameters decreases with increasing dimension. Since our main applications are in higher dimensions, the above experiments make us confident that Parameter Set 1 gives robust results.

Table 19 Adaptive error indicator interpolation results of Franke function with different parameter settings, with $\theta_{\text{refine}} = 5.0(-4)$

Parameter settings	N_{total}	$e_{\mathbf{X}}(f)$	RMS$(e_{\mathbf{X}}(f))$	$\kappa(A)_{\max}$
$a = 2, M = 20$	2180	8.3(−4)	1.1(−4)	8.1(+5)
$a = 2, M = 30$	1550	8.9(−4)	1.4(−4)	5.6(+5)
$a = 5, M = 35$	1713	7.3(−4)	1.8(−4)	3.1(+5)
$a = 5, M = 40$	1617	8.1(−4)	1.9(−4)	2.9(+5)

5 Conclusion

In this paper, we have proposed the use of an error indicator based on the idea that different approximation methods should give different results in regions where it is difficult to approximate. We compare the global RBF interpolatant with a local RBF interpolant to provide a quantitative measure of approximation error. The error indicator assigns a value to the current global interpolation, and this value describes the approximation quality. According to this error indicator value, an area with poor approximation quality and an area with good quality are determined. This detection process requires no unnecessary sampling form the target function, so it provides a considerable saving in time especially where the target function is costly to evaluate.

Applying this error indicator gives an adaptive interpolation algorithm, with the following steps: 'approximation - detect (achieved by error indicator) - refine/coarse - approximation'. On the examples, we have presented in one, two and three dimensions, this adaptive algorithm provides an accuracy level which may be pre-set by the user. We observe in our examples how effective the indicator is in automatically clustering centres in regions of high variation of the target function.

In order to delivery reliable approximation, the condition number of interpolation matrix $\kappa(A)$ should be kept to a moderate scale. In this paper, an adaptive shape parameter for the multiquadric RBF has been used, and we observe that as theoretically predicted, this is effective in keeping keep $\kappa(A)$ to a reasonable size. However, there is no guarantee that the interpolation matrices which arise in the method are invertible. In future work, we hope to implement the variably scaled kernels in [6] to guarantee the invertibility of the interpolation matrices.

In the numerical experiments, we see that adaptive error indicator approximation provides similar global accuracy to comparator methods with fewer evaluations of the target function. This is a desirable property especially when the target function is expensive (many hours for instance) to evaluate once.

Finally, we have demonstrated that the choice of parameters in the approximation model does not have a significant effect on the quality of the results, suggesting that the method described is robust.

References

1. Agnantiaris, J.P., Polyzos, D., Beskos, D.E.: Some study on dual reciprocity BEM for elastodynamic analysis. Comput. Meach. **17**, 270–277 (1996)
2. Atteia, M.: Fonctions spline et noyaux reproduisants d'Aronszajn-Bergman. Rev. Fran caise Informat. Recherche Op erationnelle **4**, 31–43 (1970)
3. Ball, K., Sivakumar, N., Ward, J.D.: On the sensitivity of radial basis interpolation to minimum point separation. J. App. Theory **8**, 401–426 (1992)
4. Behrens, J., Iske, A.: Grid-free adaptive semi-Lagrangian advection using radial basis functions. Comput. Math. Appl. **43**(3–5), 319—327 (2002)

5. Bozzini, M., Lenarduzzi, L., Schaback, R.: Adaptive interpolation by scaled multiquadrics. Adv. Comput. Math. **16**, 375–387 (2002)
6. Bozzini, M., Lenarduzzi, L., Rossini, M., Schaback, R.: Interpolation with variably scaled kernels. IMA J. Numer. Anal. **35**, 199–219 (2015)
7. Bozzini, M., Rossini, M.: Testing methods for 3D scattered data interpolation. Monografia de la Academia de Ciencias de Zaragoza **20**, 111–135 (2002)
8. Driscoll, T.A., Heryudono, A.R.H.: Adaptive residual subsampling methods for radial basis function interpolation and collocation problems. Comput. Math. Appl. **53**(6), 927–939 (2007)
9. Duchon, J.: Splines minimizing rotation-invariant semi-norms in Sobolev spaces. In: Constructive theory of functions of several variables (Proc. Conf., Math. Res. Inst., Oberwolfach, 1976). Lecture Notes in Math., vol. 571, pp. 85–100. Springer, Berlin (1977)
10. Hon, Y.C., Schaback, R., Zhou, X.: An adaptive greedy algorithm for solving large RBF collocation problems. Numer. Algor. **32**(1), 13–25 (2003)
11. Gutzmer, T., Iske, A.: Detection of discontinuities in scattered data approximation. Numer. Algor. **16**, 155–170 (1997)
12. Iske, A., Levesley, J.: Multilevel scattered data approximation by adaptive domain decomposition. Numer. Algor. **39**, 187–198 (2005)
13. Levesley, J., Ragozin, D.L.: Local approximation on manifolds using radial functions and polynomials. In: International conference on curves and surfaces [4th], Saint-Malo, Proceedings, vol. 2. Curve and Surface Fitting, pp. 291–300 (1999)
14. Powell, M.J.D.: The theory of radial basis function approximation in 1990, advance in numerical analysis. In: Light, W. (ed.), vol. II, pp. 105–210. Oxford University Press (1992)
15. Schaback, R.: Error estimates and condition numbers for radial basis function interpolation. Adv. Comput. Math. **17**, 270–277 (1995)
16. Schaback, R., Wendland, H.: Adaptive greedy techniques for approximate solution of large RBF systems. Numer. Algor. **24**(3), 239–254 (2000)