2018

Fully adaptive kernel-based methods

Leevan Ling
Hong Kong Baptist University, lling@hkbu.edu.hk

Sung Nok Chiu
Hong Kong Baptist University, snchiu@hkbu.edu.hk

FULLY ADAPTIVE KERNEL-BASED METHODS

LEEVAN LING* AND SUNG NOK CHIU*

Abstract. By exploiting the meshless property of kernel-based collocation methods, we propose a fully automatic numerical recipe for solving interpolation/regression and boundary value problems adaptively. The proposed algorithm is built upon a least-squares collocation formulation on some quasi-random point sets with low discrepancy. A novel strategy is proposed to ensure that the fill distances of data points in the domain and on the boundary are in the same order of magnitude. To circumvent the potential problem of ill-conditioning due to extremely small separation distance in the point sets, we add an extra dimension to the data points for generating shape parameters such that nearby kernels are of distinctive shape. This effectively eliminates the needs of shape parameter identification. Resulting linear systems were then solved by a greedy trial space algorithm to improve the robustness of the algorithm. Numerical examples are provided to demonstrate the efficiency and accuracy of the proposed methods.

Key words. Radial basis function, Kansa method, overdetermined collocation, adaptive trial space selection.

1. Introduction. We are interested in kernel-based methods for solving interpolation, regression or curve fitting, and boundary values problems equations in some bounded domain $\Omega \subset \mathbb{R}^d$. For the sake of discussion, we focus on elliptic differential equations subject to some boundary conditions on $\partial\Omega$:

$$\mathcal{L}u = f \text{ in } \Omega \qquad \text{and} \qquad \mathcal{B}u = g \text{ on } \partial\Omega, \tag{1.1}$$

for some linear differential operators $\mathcal{L}$ and $\mathcal{B}$. We assume the domain Ω is Lipschitz continuous and satisfies an interior cone condition. Operators $\mathcal{L}$ and $\mathcal{B}$ define a well-posed PDE. Moreover, we assume the coefficients of $\mathcal{L}, \mathcal{B}$ and functions f, g are sufficiently smooth to allow a classical solution $u^* \in H^m(\Omega)$.

Strong-form asymmetric kernel-based collocation methods, also known as Kansa methods [1], are easy to implement and meshfree in nature. A recent review of the method can be found in monograph [2]. They are widely used for solving engineering problems and partial differential equations [3–6]. Given any smooth scalar radial basis function (RBF) $\phi : \mathbb{R}_+ \to \mathbb{R}$ and shape parameter ϵ, one can define a translation-invariant scaled kernel function $\Phi_\epsilon : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$ by

$$\Phi_\epsilon(x, z) := \phi(\epsilon\|x - z\|_2) \text{ for any } \epsilon > 0.$$

To solve (1.1) by Kansa methods, users have to provide the following point sets:

- Interior collocation points $X = \{x_1, \ldots, x_{n_X}\} \subset \Omega$,
- Boundary collocation points $Y = \{y_1, \ldots, y_{n_Y}\} \subset \partial\Omega$, and
- Trial centers $Z = \{z_1, \ldots, z_{n_Z}\} \subset \Omega$.

If ones adopt the variable-shape formulation, a shape parameter ϵ_j has to be specified at each trial center z_j and we need another set of

- Shape parameters $\mathcal{E} = \{\epsilon_1, \ldots, \epsilon_{n_Z}\} \subset \mathbb{R}_+$.

The denseness of the domain-filling point sets X and Z can be measured by the mesh norm and the separation distance, which are respectively defined as

$$h_\chi := \sup_{\varsigma \in \Omega} \min_{x \in \chi} \|x - \varsigma\|_{\ell_2(\mathbb{R}^d)} \quad \text{and} \quad q_\chi := \frac{1}{2} \min_{\substack{x_i,\, x_j \,\in\, \chi \\ x_i \,\neq\, x_j}} \|x_i - x_j\|_{\ell_2(\mathbb{R}^d)},$$

*Department of Mathematics, Hong Kong Baptist University, Kowloon Tong, Hong Kong.

and the quantity $h_\chi/q_\chi =: \rho_\chi$ is commonly referred as the *mesh ratio* of $\chi \in \{X, Z\}$. For any discrete set of collocation points $X \subset \Omega$, we define a discrete norm by

$$\|w\|^2_{\ell^2(X)} = \sum_{x_i \in X} \big|w(x_i)\big|^2 \quad \text{for any } w \in \mathbb{C}(\Omega).$$

Analogously, with any discrete set $Y \subset \partial\Omega$, we can also define denseness measures h_Y, q_Y, ρ_Y and a discrete $\ell^2(Y)$ norm on the boundary for functions in $\mathbb{C}(\partial\Omega)$.

To ensure stability of Kansa formulations, we usually require some linear ratio of oversampling, i.e.,

$$\gamma h_Z \le \min(h_X, h_Y) \le \max(h_X, h_Y) \le h_Z \quad \text{for some } \gamma > 0. \tag{1.2}$$

Kansa approximations can then be sought from the trial space

$$\mathcal{U}_{\Phi,Z,\mathcal{E}} := \operatorname{span}\{\Phi_{\epsilon_j}(\,\cdot\,, z_j) \;:\; z_j \in Z, \epsilon_j \in \mathcal{E}\} \tag{1.3}$$

by minimizing the strong-form residuals at collocation points X and Y. In particular, weighted *least-squares* (LS) Kansa solutions are given by

$$U_{LS} := \operatorname*{arg\,inf}_{u \in \mathcal{U}_{\Phi,Z,\mathcal{E}}} \Big\{ \|\mathcal{L}u - f\|^2_{\ell^2(X)} + W\|u - g\|^2_{\ell^2(Y)} \Big\} \tag{1.4}$$

for some weight $W > 0$; see [7, 8] for works on finding appropriate scaling factors for W. Theoretically, (1.4) consists a class of $H^2(\mathbb{R}^d)$-optimal formulations [9] for elliptic PDEs (1.1) subject to Dirichlet boundary conditions. Suppose that $d \le 3$ and the Sobolev regularity of the true solution satisfies $m > 3 + d/2$. If the kernel Φ_ϵ, with constant shape parameters $\mathcal{E} = \{\epsilon, \ldots, \epsilon\}$ for some constant $\epsilon > 0$, reproduces the Sobolev space $H^m(\Omega)$, then the LS-Kansa solution with an appropriate weight W satisfies an error estimate

$$\|U_{LS} - u^*\|_{H^2(\Omega)} \le C h_Z^{m-2}$$

for some constant C depending on Ω, Φ, $\mathcal{L}$, m, mesh ratios ρ_Z, ρ_X and ρ_Y but independent of the classical solution u^*.

In this paper, we are interesting in solving PDE with popular C^∞ RBFs [10] such as the Gaussian $\phi(r) = \exp(-r^2)$, multiquadric $\phi(r) = \sqrt{1 + r^2}$, and high order kernels like the Whittle–Matérn–Sobolev kernels $\phi(r) = r^{m-d/2}\mathcal{K}_{m-d/2}(r)$ that reproduces $H^m(\mathbb{R}^d)$. To design a fully adaptive numerical recipe for kernel-based methods, we need ways to generate the four point sets X, Y, Z, and $\mathcal{E}$ in order to set up LS-Kansa matrix systems and deal with the potential problem of ill-conditioning. In Section 2, we propose an algorithm to generate interior and boundary data points $X \subset \Omega$ and $Y \subset \partial\Omega$ automatically so that they have similar density. If these data points are used as trial centers, the algorithm also outputs a set of quasi-random numbers $\Theta = \{\theta_1, \ldots, \theta_{n_Z}\} \subset [0, 1]$ such that θ_j will be used to generate a shape parameter ϵ_j for the jth trial center. By construction, the quasi-uniformity of points in $Z \times \Theta$ ensures $\theta_j \not\approx \theta_k$ whenever $z_j \approx z_k$. The role of the quasi-random point set Θ, when used properly, is to ensure $\epsilon_j \not\approx \epsilon_k$ if $z_j, z_k \in Z$ are nearby. We then consider an extended set of trial centers Z with $n_Z \ge n_X + n_Y$ and associated with shape parameter $\mathcal{E} = \mathcal{E}(\Theta)$. From all the associated trial basis function

$$\Phi_{\epsilon_i}(\,\cdot\,) = \phi(\epsilon_j \| \,\cdot\, - z_j\|_2) \;\text{ for } (z_j, \epsilon_j) \in Z \times \mathcal{E},$$

we employ an adaptive-greedy algorithm [11] to select a subset of n_{sel} trial centers, denoted by Z_{sel}, with the associated shape parameters $(Z_{sel}, \mathcal{E}_{sel}) \subset (Z, \mathcal{E})$ that defines the trial space (1.3), within which we seek for numerical approximation (1.4). In Section 3, the adaptive LS-Kansa algorithm will be presented in detail. If U_{LS} uses n_{sel} trial basis functions, the overall complexity of the proposed algorithm is $\mathcal{O}(n_{sel}^3)$. In Section 4, numerical examples are provided to show the robustness of the proposed fully-adaptive algorithm.

2. Adaptive point sets generator. In literature, we can find different ways to generate point sets adaptively to run Kansa methods, for example, by quad-tree [12], wavelet [13, 14], and Voronoi diagram [11, 15]. In common, all these methods are designed to keep the separation distance q_Z bounded away from zero. The main drawback is that the number of additional points is no longer arbitrary but has to obey the respective data structure. In this paper, we focus on low-discrepancy sequences by quasi-Monte Carlo methods [16], which generate deterministic sequences of points such that any finite subsequences fill the unit hypercube uniformly, where the deviation from uniformity is assessed by different so-called discrepancies. Examples include the Faure, Halton, Sobol, and van der Corput sequences. Points from such a low-discrepancy sequence are also known as quasi-random points, which can be thought of as points distributed according to the uniform distribution. However, unlike pseudo-random numbers, quasi-random points should be interpreted not as independent but as correlated realizations of the uniform distribution, so that in such a sample of realizations (i.e. a finite subsequence) we would not be 'unlucky' to observe a large deviation from uniformity or a cluster of arbitrarily close points. For example, in the Sobol sequence in $\mathbb{R}^d$, the distance between the $2i$th and the $(2i+1)$st points is $\sqrt{d}/2$, and the minimum distance between the first n points is bounded below by $\sqrt{d}/(2n)$ and empirically found to be proportional to $n^{-1/d}$ for large n [17]. Nevertheless, in such a sample there may still be points that should better not be used as trial centers, and this will be explained and dealt with in Section 3.

The goal here is to expand any given $X \in \Omega$ and $Y \in \partial\Omega$ with $h_X \approx h_Y$ to $X' \supset X$ and $Y' \supset Y$ in such a way that $n_{X'}$ can be pre-specified and $h_{X'} \approx h_{Y'}$ with some appropriate $n_{Y'}$. Trial centers in Z can be generated by the same approach.

2.1. Interior points. Consider smooth bounded domains $\Omega \subset B \subset \mathbb{R}^d$ in some bouncing box B with a parametric boundary $\partial\Omega$. We assume there exists a sign function

$$\mathrm{Sign}_\Omega(x) = \begin{cases} +1 & x \notin \overline{\Omega} \\ 0 & x \in \partial\Omega \\ -1 & x \in \Omega. \end{cases}$$

Denote by $\{p_1, p_2, \ldots\} \subset B$ a low-discrepancy point set in $\mathbb{R}^d$; see [18] for the related computational issues. Constructing the set X for Ω is equivalent to identifying the strictly increasing index functions $\varphi_\Omega : \mathbb{N} \to \mathbb{N}$ such that $\mathrm{Sign}_\Omega(p_{\varphi_\Omega(j)}) = -1$ for all $j \in \mathbb{N}$ and $\mathrm{Sign}_\Omega(p_i) \neq -1$ for all $i \in \mathbb{N} \setminus \{\varphi_\Omega(j) : j \in \mathbb{N}\}$. Then, for any given n_X, we take

$$X := \{p_{\varphi_\Omega(j)} : 1 \le j \le n_X\}.$$

If the points are used as trial centers, we will begin with a low-discrepancy point set in $\{(p_1, \theta_1), (p_2, \theta_2), \ldots\} \subset B \times [0, 1] \subset \mathbb{R}^{d+1}$, and then obtain the set of quasi-random

numbers Θ by

$$\Theta := \{\theta_{\varphi_\Omega(j)} : 1 \le j \le n_X\}.$$

The computational cost here is dominated by the *rejection rate*. To seek for n_X points, it requires approximately $n_X \mathrm{Vol}(B)/\mathrm{Vol}(\Omega)$ evaluations of Sign_Ω. Expanding X to X' simply means evaluating more $\mathrm{Sign}_\Omega(p_{\varphi_\Omega(j)})$ until the pre-specified number of collocation points $n_{X'}$ are included in the point set, i.e.

$$X' := \{p_{\varphi_\Omega(j)} : 1 \le j \le n_{X'}\} \text{ and } \Theta' := \{\theta_{\varphi_\Omega(j)} : 1 \le j \le n_{X'}\}.$$

2.2. Estimating the required number of boundary points. The construction of Y and expanding it to Y' can be done analogously. However, we have to determine an appropriate value n_Y of boundary points so that if we place n_Y quasi-random points on $\partial\Omega$, its denseness is similar to the denseness of X in Ω. Our goal is to control their minimum separating distances $q_Y \approx q_X$. Since

$$q_Y \sim \mathrm{Vol}(\partial\Omega) n_Y^{-1/(d-1)} \text{ and } q_X \sim \mathrm{Vol}(\Omega) n_X^{-1/d},$$

we have

$$n_Y \approx \left(\frac{\mathrm{Vol}(\partial\Omega)}{\mathrm{Vol}(\Omega)} n_X^{1/d}\right)^{(d-1)}. \tag{2.1}$$

As $\mathrm{Vol}(\Omega)$ and $\mathrm{Vol}(\partial\Omega)$ are not always handy for use, we use (2.1) with some computational efficient estimates. From Section 2.1, we can easily estimate

$$\mathrm{Vol}(\Omega) \approx \frac{n}{\varphi_\Omega(n)}\mathrm{Vol}(B) \quad \text{for any sufficiently large } n \in \mathbb{N}.$$

To estimate $\mathrm{Vol}(\partial\Omega)$, let a narrow-domain be defined by $\Gamma_\delta = \{x \in \Omega \colon \mathrm{dist}(x, \Gamma) < \delta\}$ for some sufficiently small $\delta > 0$ such that $\mathrm{Vol}(\Gamma_\delta) \approx \delta \mathrm{Vol}(\partial\Omega)$. However, we need not estimate $\mathrm{Vol}(\Omega)$ and $\mathrm{Vol}(\partial\Omega)$ separately because $\mathrm{Vol}(\Gamma_\delta)/\mathrm{Vol}(\Omega)$ can be estimated directly by the ratio of the number of points in $X \cap \Gamma_\delta$ to n_X.

To determine the value of δ for practical applications in our context, we expediently treat the quasi-random points in X as independent random variables uniformly distributed in Ω. Consider the indicator function $\mathbf{1}_{\Gamma_\delta}(\cdot)$ and define

$$N_{\Gamma_\delta} := \sum_{j=1}^{n_X} \mathbf{1}_{\Gamma_\delta}(p_{\varphi_\Omega(j)}),$$

which is the number of points in $X \cap \Gamma_\delta$ and follows the binomial distribution $\mathtt{B}(n_X, p)$, where

$$p := \Pr\{\mathbf{1}_{\Gamma_\delta}(p_{\varphi_\Omega(j)}) = 1\} = \frac{\mathrm{Vol}(\Gamma_\delta)}{\mathrm{Vol}(\Omega)}.$$

By Chebyshev's inequality, for any $\varepsilon > 0$,

$$\Pr\left(\left|\frac{N_{\Gamma_\delta}}{n_X} - \frac{\mathrm{Vol}(\Gamma_\delta)}{\mathrm{Vol}(\Omega)}\right| \ge \varepsilon\right) \le \frac{\left(1 - \frac{\mathrm{Vol}(\Gamma_\delta)}{\mathrm{Vol}(\Omega)}\right)\frac{\mathrm{Vol}(\Gamma_\delta)}{\mathrm{Vol}(\Omega)}}{\varepsilon^2 n_X}.$$

Experience suggests that choosing ε as 10% of the ratio $\mathrm{Vol}(\Gamma_\delta)/\mathrm{Vol}(\Omega)$ and using 0.1 as the upper bound of the above probability will lead to satisfactory results. That is to say, we want to have

$$\frac{\left(1-\frac{\mathrm{Vol}(\Gamma_\delta)}{\mathrm{Vol}(\Omega)}\right)\frac{\mathrm{Vol}(\Gamma_\delta)}{\mathrm{Vol}(\Omega)}}{n_X\left(\frac{1}{10}\frac{\mathrm{Vol}(\Gamma_\delta)}{\mathrm{Vol}(\Omega)}\right)^2}=0.1.$$

Approximating $\mathrm{Vol}(\Gamma_\delta)$ by $\delta\mathrm{Vol}(\partial\Omega)$ yields

$$\delta=\frac{\mathrm{Vol}(\Omega)}{\mathrm{Vol}(\partial\Omega)}\left(\frac{1000}{1000+n_X}\right). \tag{2.2}$$

Ironically, the expression contains $\mathrm{Vol}(\Omega)/\mathrm{Vol}(\partial\Omega)$, which is (the reciprocal of) the ratio we are estimating in order to use (2.1). However, this probabilistic argument is not the derivation of the exact value of δ but a determination of the order of magnitude for δ to be used in the estimation procedure. Thus, we can use the volume-to-surface area ratio of a d-dimensional sphere of radius $r_\circ$ to replace $\mathrm{Vol}(\Omega)/\mathrm{Vol}(\partial\Omega)$ in (2.2) and get

$$\delta=\frac{r_\circ}{d}\left(\frac{1000}{1000+n_X}\right),$$

where $r_\circ$ is half of the longest side of the bouncing box B. The main computational cost here is the identification of N_{Γ_δ}. In our implementation, a kd-tree is built for estimating the distance from each point in X to $\partial\Omega$.

2.3. Generating boundary points. Given any existing sets of collocation points X and Y. Based on a user/algorithm instructed number of extended interior collocation points $n_{X'}$, we can generate $X'\supset X$ as in Section 2.1. Then, we estimate the desired number of extended boundary collocation points, denoted as $n_{Y'}$, based on $n_{X'}$ and the approach in Section 2.2. This section discusses how to add/put a specific number of low-discrepancy points on the boundary $\partial\Omega$.

To construct n_Y quasi-uniform points on $\partial\Omega$, we follow the ideas in [19]. Let the boundary $\partial\Omega\subset\mathbb{R}^d$ be defined parametrically by $\rho: D\to\mathbb{R}^d$ for some rectangular domain $D\subset\mathbb{R}^{d-1}$ of parameters as

$$\rho(q)=\begin{pmatrix}x_1(q)\\x_2(q)\\\vdots\\x_d(q)\end{pmatrix}\quad\text{for } q\in D.$$

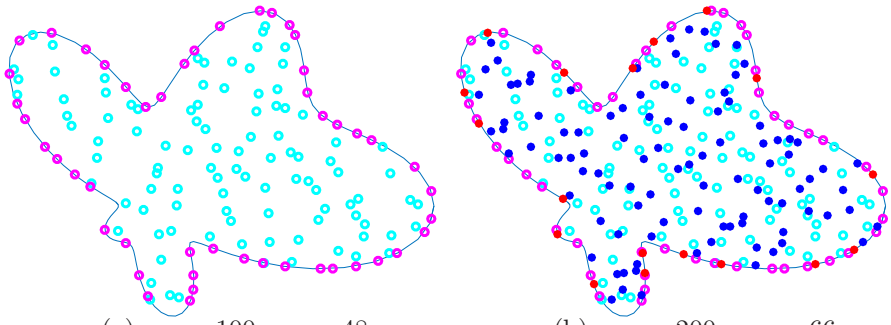
Then we can define a non-negative function $\mathcal{G}: D\to\mathbb{R}$ by

$$\mathcal{G}:=\det([\mathcal{G}_{ij}])$$

using the metric tensor of $\partial\Omega$ with element

$$\mathcal{G}_{ij}=\left(\frac{\partial\rho}{\partial u_i}\cdot\frac{\partial\rho}{\partial u_j}\right),\qquad\text{for } 1\le i,j\le d-1.$$

Note that the surface area of $\partial\Omega$ is given as $\mathrm{Vol}(\partial\Omega)=\int_D\sqrt{\mathcal{G}}\,dq$ and $\sqrt{\mathcal{G}}/\mathrm{Vol}(\partial\Omega)$ is a probability density function that allows us to control the density of points in D in order to yield quasi-random points on $\partial\Omega$.



(a) $n_X = 100$, $n_Y = 48$ (b) $n_{X'} = 200$, $n_{Y'} = 66$

FIG. 2.1. *A schematic demonstration of data points refinement. (a) Request $n_X = 100$ interior point and estimate that $n_Y = 48$ is required to maintain a similar fill distance. (b) Request an extended set of $n_{X'} = n_X + 100$ interior points that requires $n_{Y'} = 48 + 18$ boundary points to maintain $h_{X'} \sim h_{Y'}$.*

To avoid computing $\mathrm{Vol}(\partial\Omega)$, we generate another low-discrepancy point set $\{(q_1, s_1), (q_2, s_2), \ldots\} \subset D \times [0, \max(\sqrt{\mathcal{G}})]$. For any $j \in \mathbb{N}$, a point $q_j \in D$ will be *rejected* if

$$\sqrt{\mathcal{G}(q_j)} < s_j.$$

The *rejection rate* here is dominated by the range of $\sqrt{\mathcal{G}}$, which is related to the variation of the curvature of the boundary. When the point $q_j \in Q$ is accepted, the corresponding $\rho(q_j) \in \partial\Omega$ will be stored. Similar to the generation of interior trial centers, we can begin with a low-discrepancy point set in $D \times [0, \max(\sqrt{\mathcal{G}})] \times [0, 1]$ in order to obtain a quasi-random number θ_j associated with each accepted point $\rho(q_j)$.

Figure 2.1 shows a schematic demonstration of our intended outcome. From the figure, we can already see that some data points generated by the proposed quasi-random approach could be placed close together and we will handle this in the coming section.

3. System-free algorithm for solving PDEs. Using the procedures in Section 2, we can obtain nested sequences of sets of collocation points and trial centers:

- Interior collocation: $X_1 \subset X_2 \subset \cdots \subset X_k \subset \cdots$
- Boundary collocation: $Y_1 \subset Y_2 \subset \cdots \subset Y_k \subset \cdots$
- Trial centers: $Z_1 \subset Z_2 \subset \cdots \subset Z_k \subset \cdots$

along with sets of quasi-random numbers $\Theta_k \subset [0, 1]$ to accompany each set Z_k, $k = 1, 2, \ldots$, respectively.

Firstly, we present a *system-free algorithm* that enlarges trial spaces $Z_{sel} \subset Z_k$ on-the-fly. We suppose that a kernel Φ is chosen and sets of shape parameters $\mathcal{E}_k$ are available in Section 3.1 so that the finite dimensional trial space as in (1.3) is well defined for each Z_k. Implementations details will be discussed in Section 3.2. In Section 3.3, we complete the description of the proposed algorithm by specifying a strategy for generating shape parameters $\mathcal{E}_k$ based on the pre-selected trial centers Z_k and the quasi-random numbers in Θ_k.

3.1. System-free algorithm. Suppose that we work on discrete sets of data points $X_k \subset \Omega$, $Y_k \subset \partial\Omega$ and a set of trial centers candidates $Z_k \subset \overline{\Omega}$ for some $k \in \mathbb{N}$. Out of these candidates, further suppose that we have a subset of *selected trial centers*

$Z_{sel} \subset Z_k$ and subsets of *selected collocation points* $X_{sel} \subset X_k$ and $Y_{sel} \subset Y_k$. Denote $N = |X_{sel}| + |Y_{sel}|$, $M = |Z_{sel}|$ and we assume $N \geq M$. Then, these preselected point sets specify an $N \times M$ Kansa system

$$A([X_{sel}; Y_{sel}], Z_{sel})\lambda = b([X_{sel}; Y_{sel}]) \tag{3.1}$$

with matrix entries

$$[A]_{ij} = \begin{cases} \mathcal{L}\Phi_{\epsilon_j}(x_i, z_j), & \text{for } x_i \in X_k, \quad z_j \in Z_k, \\ \mathcal{B}\Phi_{\epsilon_j}(y_i, z_j), & \text{for } y_i \in Y_k, \quad z_j \in Z_k, \end{cases}$$

for any variable-shape kernel $\Phi_\epsilon(x, z) = \phi(\epsilon\|x - z\|_2)$ of our choice. The right-hand side vector is given as

$$b := [f(X_k), g(Y_k)]^T.$$

We solve (3.1) in the least-squares sense to obtain λ_{LS}. We propose a system-free algorithm that addresses the followings automatically:

- Is $\lambda_{LS} \in \mathbb{R}^{|Z_{sel}|}$ a satisfactory approximation to the solution of the full system

$$A([X_k; Y_k], Z_{sel})\lambda_{LS} \approx b([X_k; Y_k])?$$

  If not, expand the sets of selected collocation points X_{sel} and Y_{sel} to further reduce error.
- Are there enough trial center candidates in $Z_k \setminus Z_{sel}$ to identify new appropriated trial centers? If not, expand the set of selected trial centers Z_{sel} to enlarge the trial space.

To answer the first questions, we need to measure errors by primal and dual residuals whose definitions are given in the next section. The primal residual simply measures the error $A([X_k, Y_k], Z_{sel})\lambda_{LS} - b([X_k, Y_k])$ at *all collocation points*. Provided that we have some stability estimate [9] for the least-squares Kansa formulation, small primal discrete residual ensures the numerical error $\|U_{LS} - u^*\|$ of our least-squares solution is also small in some appropriate norm. If the residual is unsatisfactorily large, we can identify collocation points with large error by examining the magnitudes of the entries in the primal residual. This allows us to expand X_{sel} and/or Y_{sel} to include new collocation points with large error, and hopefully the next approximate solution can reduce their errors.

The dual residual, on the other hand, measures a scaled distance between the approximate solution and the hyperplanes of the affine space containing the exact solution (and the corresponding Lagrange multipliers), see [20]. In a similar manner, it can be used to identify which unused trial centers are causing large dual error. Including these new trial centers expands Z_{sel}. By design, trial centers with large dual errors are *far away* from the selected ones and hence keeping the condition number of the selected submatrix $A([X_{sel}; Y_{sel}], Z_{sel})$ small.

These are the rationales of greedy trial subspace selection algorithms [20, 21], which are designed to run on matrix system with a fixed size. The key features making the greedy algorithm attractive are that:

- it is matrix-free and only the entries of A in the selected rows and columns are required, and
- it has linear complexity in M and N if run in block-form,

each of which allows us to use large sets of collocation points and trial centers. In our context, the underneath full matrix $A([X_k; Y_k], Z_k)$ can increase in size by using more data points, i.e.,

$$X_k \leftarrow X_{k+1}, \quad Y_k \leftarrow Y_{k+1}, \quad Z_k \leftarrow Z_{k+1}.$$

As the selected sets expand, the numbers of unselected candidates drop, and it is easy to make undesired selections simply because there is *no choice*. In the proposed system-free algorithm, we do not allow selecting more than half of the available candidates, i.e., keeping $|X_{sel}| + |Y_{sel}| < \frac{1}{2}(|X_k| + |Y_k|)$ and $|Z_{sel}| < \frac{1}{2}|Z_k|$ at all time. When more data points are needed, we expand all sets and continue. After the selection process terminates, which is usually due to ill-conditioned selected submatrix instead of tiny residuals, we can obtain a least-squares Kansa solution as in (1.4) using the selected trial centers Z_{sel} and the final sets of collocation points X_k and Y_k.

3.2. Implementation in block from. To implement the system-free algorithm, it is sufficient to consider a potentially huge in size and ill-conditioned $M \times N$ with $M \gtrsim N$ underneath linear system

$$A\lambda = b,$$

which *need not be fully computed/stored or even be determined.* If the system-free algorithm terminates with point sets X_k and Y_k, then $M = |X_k| + |Y_k|$ and $N = |Z_k|$; the exact value of M, N and k are *a posteriori* and somewhat irrelevant.

In this section, we will use the Matlab matrix notations to describe the necessary matrix operations in the proposed system-free algorithm. The expansion of point sets can be viewed as marking some rows and columns as candidates, indexed by I_{can} and J_{can} respectively. The algorithm always terminates with $I_{can} = \{1, 2, \ldots, M\}$ and $J_{can} = \{1, 2, \ldots, N\}$ from this point of view. The aim is to select a set of columns of A, indexed by $J_{sel} \subset J_{can}$, such that the submatrix $A(1 : M, J_{sel})$ is well-conditioned, and the least-squares solution $\lambda_{LS} = A(1 : M, J_{sel})^{+} b$ gives a good approximation to the true solution in the sense that its zero-upsampled vector (by patching zero to entries associated with unselected trial centers) solves $A\widehat{\lambda}_{LS} \approx b$.

We begin with relative small sets of I_{can} and J_{can}, i.e., we generate some initial sets of quasi-random collocation points X_1, Y_1 and trial centers Z_1. To initialize, find index $i \in I_{can}$ such that the ith entry $[b(I_{can})]_i = \|b(I_{can})\|_\infty$ and set $I_{sel}^{(1)} = \{i\}$. Then, find index $j \in J_{sel}$ such that the column $A(I_{can}, j)$ has the largest norm and set $J_{sel}^{(1)} = \{j\}$.

For $\ell \in \mathbb{N}$, we use the reduced-QR factorization of $A\left(I_{sel}^{(\ell)}, J_{sel}^{(\ell)}\right)$, which will be available in memory from the previous iteration, to solve the following linear systems (in the least-squares sense if it is overdetermined):

$$\begin{cases} A\left(I_{sel}^{(\ell)}, J_{sel}^{(\ell)}\right) \eta^{(\ell)} &= b(I_{sel}^{(\ell)}), \\ A\left(I_{sel}^{(\ell)}, J_{sel}^{(\ell)}\right)^T \zeta^{(\ell)} &= -\eta^{(\ell)}, \end{cases} \tag{3.2}$$

and get the primal and dual residuals by

$$\mu^{(\ell+1)} := A\left(I_{can},\, J_{sel}^{(\ell)}\right) \eta^{(\ell)} - b(I_{can}) \in \mathbb{R}^{|I_{can}|}, \tag{3.3}$$

$$\nu^{(\ell+1)} := \widehat{\eta}^{(\ell)} + A\left(I_{sel}^{(\ell)},\, J_{can}\right)^T \zeta^{(\ell)} \in \mathbb{R}^{|J_{can}|}, \tag{3.4}$$

using the zero-upsampled vector $\widehat{\eta}^{(\ell)}$ of $\eta^{(\ell)}$ from (3.2). From (3.3)–(3.4), we can see that the main storage requirement is $M|J_{sel}| + N|I_{sel}| - |I_{sel}| \cdot |J_{sel}|$ instead of MN.

To run the proposed algorithm in *block form*, we *roughly double* the numbers of selected rows and columns based on the residuals of each iteration. Expand data sets whenever necessary, we can assume that $|I_{sel}| < \frac{1}{2}|I_{can}|$ and $|J_{sel}| < \frac{1}{2}|J_{can}|$ at all times. Then, the expansion of the *selected rows indices* I_{sel} from the ℓth to $(\ell+1)$st iteration is relatively straightforward: we sort the primal residual $\mu^{(\ell+1)}$ according to the magnitude of its entries, partition the sorted vector into $\gtrsim |I_{sel}^{(\ell)}|$ parts, and select a new row index from each partition to expand $I_{sel}^{(\ell)}$ to $I_{sel}^{(\ell+1)}$. Now, we compute the reduced-QR factorization $A\left(I_{sel}^{(\ell+1)},\, J_{sel}\right) = Q_\ell R_\ell$.

Using a similar procedure, we sort and partition the dual residual $\nu^{(\ell+1)}$ to select $n_{sc} \geq 2|J_{(\ell)}|$ *shortlisted candidate columns*, indexed by J_{sc}, and only part of them will be added to $J_{sel}^{(\ell)}$ to form $J_{sel}^{(\ell+1)}$. It is mathematically equivalent to say that we select new column indices from J_{sc} by the first $|J_{sel}^{(\ell)}|$ ones in the permuted-QR factorization of $A\left(I_{sel}^{(\ell+1)}, J_{sel}^{(\ell)} \cup J_{sc}\right)$. In the actual implementation, this can be done by updating the QR-factorization of the previous iteration stored in memory, see [11] for details. Moreover, the reduced QR-factorization of the expanded matrix $A\left(I_{sel}^{(\ell+1)}, J_{sel}^{(\ell+1)}\right)$ can be obtained as a byproduct without extra computation.

Repeat the above process until the condition number of the expanded matrix exceeds certain tolerance τ, say $\tau = 1/\epsilon_{machine}$, and obtain indices I_{sel} and J_{sel} of the selected rows and columns. Since half of the column indices in J_{sel} are newly added, we may have to remove some selected columns to keep the condition number of submatrix $A\,(I_{sel}, J_{sel})$ below tolerance τ; this is done by a bisection search with a condition number estimator. Readers are referred to the original article [11] for the details. Finally, we can obtain the least-squares solution by computing $A(1:M, J_{sel})^{+}b$. By using the specific value of n_{sc} derived in [11], the cost of running the proposed system-free algorithm is lower than running the greedy algorithm in [11] once on a fixed $M \times N$ linear system, which is $\mathcal{O}(N\, n_{sel}^2 + M\, n_{sel})$ with $n_{sel} := |J_{sel}|$ is the number of trial centers used in defining the solution in (1.4). If we expand data points in such a way that $M \approx N$, then we must have the value of N somewhere between $2n_{sel}$ to $4n_{sel}$ in order to keep a sufficient number of candidate columns. Thus, the overall cost of the proposed system-free algorithm would be of $\mathcal{O}(n_{sel}^3)$ making the method compatible with blindfolded direct approaches using the same number of trial basis functions.

3.3. Quasi-random shape parameters. At this point, we almost have an executable system-free algorithm except that we need a way to generate shape parameters. Finding an optimal constant shape parameter for a given function recovery problem is a long-standing open question. Different strategies have been proposed, see [22–27]. Back in 1992, Kansa et al. [28] observed that variable shape parameters can improve the accuracy of meshfree collocation methods. Using variable shape parameters also helps reduce the Runge phenomenon in RBF interpolation [29]. The search for good strategies for variable shape parameters is ongoing [30]. This problem is analogous to the problem of selecting smoothing parameters in nonparametric curve estimation. A constant smoothing parameter, which typically is the minimizer of the mean integrated squared error estimated empirically by cross-validation or obtained theoretically by considering asymptotic expressions of bias and variance, is often good enough for the estimation of smoothed curves, see e.g. [31–36], but variable smoothing parameters will do a better job when the curves to be estimated have complicated

structures, see [37–41]. These approaches rely on spatial information of the data points, and some also require information about the unknown solution or the curve to be estimated, to determine values of shape or smoothing parameters.

We propose a *quasi-random shape parameters* strategy for the system-free algorithm that is

- data driven as the algorithm iterates,
- independent of spatial information, and
- robust for different PDE solutions,

based on experience and empirical experimental results. For sake of simple discussion and applications of well-established rule-of-thumb, we assume the bounding box $B \supset \Omega$ has edges of length $\mathcal{O}(1)$. To be efficient, the system-free algorithm should be able to use between $10^d \leq n_{sel} \leq 100^d$ trial basis functions to solve any given problem. Taking the *stationary approach* [42, Ch.2], one would use shape parameters $\epsilon_j = \mathcal{O}(n_{sel}^{1/d})$; in this length scale, we can be sure that $\epsilon_j \lesssim n_{sel}^{1/d} := \epsilon_{\max}$ yields very peaky basis functions. The system-free algorithm will probably need a lot of basis to obtain any good approximation. The value of $\epsilon_{\max}$ is not clearly determined because of the unknown value n_{sel}. In the coming section, we shall demonstrate its role in the proposed algorithm. On the other end, we simply require $\epsilon_j > 0$ for all j to avoid having an extra parameter for the lower bound and allow really flat basis functions getting into the trial space.

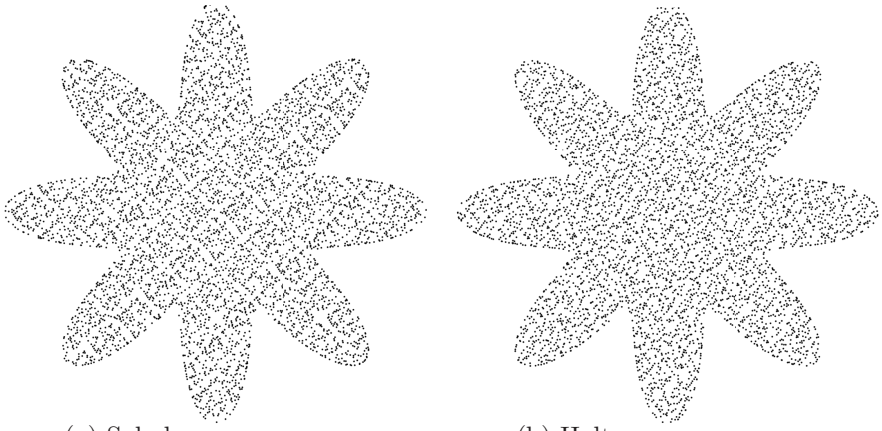
Now that we set a range for all the shape parameters, i.e., $0 < \epsilon_j \leq \epsilon_{\max}$. To generate shape parameters from the quasi-random numbers Θ_k, we need to assign some suitable probability densities to $\mathcal{E}_k$ based on empirical knowledge. On one hand, the system-free algorithm terminates sooner with small n_{sel} if more flat basis functions are in the system. On the other hand, it tends to select peaky trial basis functions and occasionally select some very flat basis functions near the boundary. Knowing that we need *more flat than peaky basis functions* as candidates, we propose using the following probability density

$$f_{\mathcal{E}_k}(\theta) = \begin{cases} \dfrac{2}{\epsilon_{\max}^{(k)}} - \dfrac{2\theta}{(\epsilon_{\max}^{(k)})^2} & \text{for } \theta = [0, \epsilon_{\max}^{(k)}], \\ 0 & \text{otherwise,} \end{cases}$$

for any $z \in Z_k \cap \Omega$ without adding new parameters. We only need to set the first $\epsilon_{\max}^{(1)}$, say equals to 10 or 100, as in the above discussion. In each iteration we reduce the upper bound $\epsilon_{\max}^{(k+1)}$ in a data-driven way to the *mean of all selected shape parameters*.

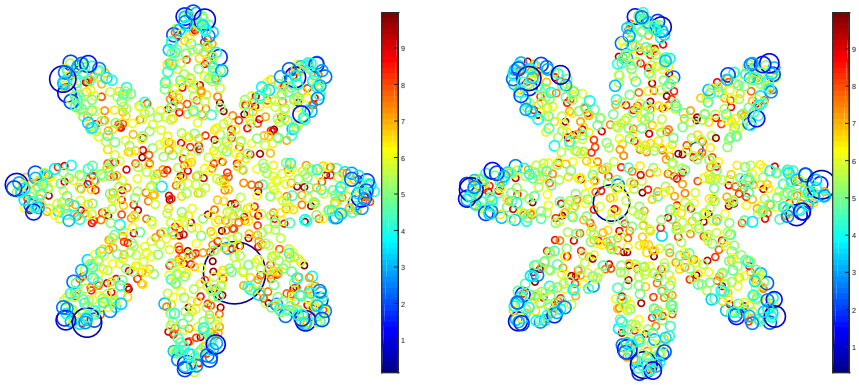
The set of quasi-random (uniform) numbers Θ_k we generated alongside with Z allows us to generate shape parameters that are quasi-random having the distribution $f_{\mathcal{E}_k}$. Denote by $\mathcal{F}_k$ the cumulative distribution function of $f_{\mathcal{E}_k}$. The shape parameter of $z_j \in Z_k \cap \Omega$ is generated by the transformation $\epsilon_j = \mathcal{F}_k^{-1}(\theta_j)$ for $\theta_j \in \Theta_k$. By design, if $z_i, z_j \in \Omega$ and $|z_i - z_j|$ for any $i \neq j$ is small, then $|\theta_i - \theta_j|$ will be large without any computational overhead in checking the densities of Z. The same holds true for any pair of trial centers on the boundary if we use the same probability density function to generate shape parameters for trial centers $z \in Z_k \cap \partial\Omega$. To extend this property to any pair of points in $(Z_k \cap \Omega) \times (Z_k \cap \partial\Omega)$, we set $\epsilon_j = \frac{1}{2}\mathcal{F}_k^{-1}(\theta_j)$ to be the shape parameter of $z_j \in Z_k \cap \partial\Omega$.

4. Numerical examples. We are now ready to run some numerical experiments on the system-free adaptive meshfree collocation methods. Throughout this section,



(a) Sobol sequence (b) Halton sequence

FIG. 4.1. *Example 4.1: Quasi-random points $X \cup Y$ by different generators.*



(a) Sobol sequence (b) Halton sequence

FIG. 4.2. *Example 4.1: System-free algorithm selected flat basis functions with shape parameters ≤ 10; peaky ones were omitted from this illustration.*

initial collocation point set X_1 contains 128 interior points. The set of trial centers Z_k is taken to be the union of the collocation sets X_k and Y_k. That is, the underneath matrix system is the square interpolation matrix or traditional Kansa matrix. Quasi-random points are generated by built-in Matlab fuctions `sobolset` and `haltonset`. All reported errors were evaluated on sets of quasi-uniform points, generated as in Section 2, differ from and finer than the sets of collocation points used in the algorithm.

4.1. Example: Quasi-random number generators. We set the system-free algorithm to run in an asterisk-shape domain to solve an interpolation/regression problem with the Gaussian basis. Shape parameters are randomly generated as in Section 3.3 using $\epsilon_{\max} = 100$. Figure 4.1 shows the resulting quasi-random data points generated by the Sobol and Halton sequences within the algorithm. To the eyes, different quasi-random number generators give point sets with different hidden patterns. Figure 4.2 shows some shape parameters of the selected trial basis. In terms of the magnitudes of n_{sel} and maximum error, the hidden patterns in X_k, Y_k and Z_k make no obvious difference.

GA	Bad	n_{sel}		$L^2(\Omega)$		$L^\infty(\Omega)$	
$\epsilon_{\max}$	Run	Min	Max	Mean	S.D.	Mean	S.D.
25	0	635	739	2.6347e-07	5.1091e-07	8.5665e-06	1.7673e-05
50	0	887	1175	1.1854e-09	1.6752e-09	3.7677e-08	5.3122e-08
100	2	1021	1749	4.7907e-09	2.5103e-08	4.9567e-08	2.5110e-07
MQ	Bad	n_{sel}		$L^2(\Omega)$		$L^\infty(\Omega)$	
$\epsilon_{\max}$	Run	Min	Max	Mean	S.D.	Mean	S.D.
25	0	1023	2013	1.8662e-07	9.4187e-07	1.8582e-06	9.2901e-06
50	2	2015	3909	1.7300e-08	5.6953e-08	1.7804e-07	4.5865e-07
100	1	2007	7343	5.6439e-06	1.8148e-05	6.9103e-05	2.1448e-04
MS	Bad	n_{sel}		$L^2(\Omega)$		$L^\infty(\Omega)$	
$\epsilon_{\max}$	Run	Min	Max	Mean	S.D.	Mean	S.D.
25	0	401	429	7.2527e-05	7.4555e-05	1.7176e-03	2.4994e-03
50	0	673	745	6.5219e-07	1.0320e-06	1.6340e-05	3.1292e-05
100	0	923	1003	2.7574e-08	3.3238e-08	7.3817e-07	1.0683e-06

TABLE 4.1
Example 4.2: `peaks` *out of 30 runs*

4.2. Example: Choosing kernels and $\epsilon_{\max}$. We verify the accuracy and efficiency of the system-free algorithm using Gaussian (GA), multiquadric (MQ) and Whittle-Matérn-Sobolev (MS) kernels that reproduce the Sobolev space $H^{10}(\mathbb{R}^2)$. As benchmark, we consider interpolation/regression problems in rectangular domain for the `peaks` function in $\Omega = [-3, 3]^2$. Since the core mechanism of the system-free algorithm is random, we run each interpolation/regression problem 30 times and report the statistics. What we are looking for is the appropriate value of $\epsilon_{\max}$ for these kernels. With the suitable $\epsilon_{\max}$, we want the system-free algorithm to be robust in the sense that it yields small errors in mean with small standard deviation among different functions and, ideally, no *bad run*. A bad run is easy to identify by its early termination with very few selected basis and large residual errors. Also, we want the algorithm to be efficient (with small n_{sel}) with predictable run-time (range of n_{sel}). Table 4.1 shows that our proposed strategy does not work efficiently with the MQ kernel.

Next, we further verify the GA and MS kernels by interpolating the `franke` function and a function

$$f(x, y) = \max(0, x)^3 - \max(0, y)^3 \tag{4.1}$$

with nonsmooth Laplacian in $\Omega = [-1, 1]^2$. Resulting statistics are listed in Tables 4.2–4.3. For MS, it is clear that setting $\epsilon_{\max} = 100$ causes no trouble to the algorithm. For GA, we see that $\epsilon_{\max} = 100$ may result in a few bad runs. When $\epsilon_{\max} = 50$, it performs similar to MS in terms of both accuracy and efficiency, i.e., n_{sel}. To this end, the MS kernel wins in terms of robustness.

4.3. Example: Solving PDEs. From the results in Example 4.2, readers can already see some sort of *refinement* the system-free algorithm offers. We can ask for higher accuracy by allowing more basis functions, i.e., more computational time. This, however, should not be done by increasing the value of $\epsilon_{\max}$. If the system-free algorithm terminates due to ill-conditioned submatrix system (as in all reported data in this example), providing more candidate trial functions will not help reducing the

GA	Bad	n_{sel}		$L^2(\Omega)$		$L^\infty(\Omega)$	
$\epsilon_{\max}$	Run	Min	Max	Mean	S.D.	Mean	S.D.
25	0	655	733	9.6738e-04	1.3783e-03	2.8117e-02	4.0347e-02
50	0	823	1217	2.4058e-04	4.8471e-04	8.3423e-03	1.7349e-02
100	2	1015	1777	5.7304e-06	7.5535e-06	2.5603e-04	4.0785e-04
MS	Bad	n_{sel}		$L^2(\Omega)$		$L^\infty(\Omega)$	
$\epsilon_{\max}$	Run	Min	Max	Mean	S.D.	Mean	S.D.
25	0	397	423	2.5519e-03	2.0556e-03	5.6153e-02	6.8083e-02
50	0	677	749	4.3585e-04	3.8856e-04	1.0987e-02	1.3409e-02
100	0	911	1227	1.5049e-04	4.6402e-04	4.8690e-03	1.6973e-02

TABLE 4.2
Example 4.2: `franke` *out of 30 runs*

GA	Bad	n_{sel}		$L^2(\Omega)$		$L^\infty(\Omega)$	
$\epsilon_{\max}$	Run	Min	Max	Mean	S.D.	Mean	S.D.
25	0	623	757	1.4682e-05	2.5899e-05	3.7926e-04	7.6777e-04
50	0	869	1007	1.2687e-05	1.9572e-05	4.3372e-04	6.9755e-04
100	2	1023	1785	1.4470e-06	7.7554e-07	3.9665e-05	5.5451e-05
MS	Bad	n_{sel}		$L^2(\Omega)$		$L^\infty(\Omega)$	
$\epsilon_{\max}$	Run	Min	Max	Mean	S.D.	Mean	S.D.
25	0	403	427	2.0743e-05	1.0798e-05	3.5904e-04	3.9292e-04
50	0	671	733	7.9690e-06	8.9910e-06	1.6165e-04	2.9417e-04
100	0	927	1009	4.5236e-06	4.4479e-06	1.0258e-04	1.5014e-04

TABLE 4.3
Example 4.2: max-cube out of 30 runs

condition number and will not keep the algorithm going. Instead, the *refinement* feature can be done by putting a cap on n_{sel} so that, when desired, the system-free algorithm can continue to run. In this example, we fix $\epsilon_{\max} = 100$ and solve PDEs.

Now, we solve a Poisson equation $\Delta u = f$ in $[-3, 3]^2$ subject to Dirichlet boundary conditions with the `peaks` as exact solution. Table 4.4 shows the resulting accuracy of the proposed algorithm at different stages. If we allow the system-free algorithm to run till termination, accuracy for solving this PDE is indeed similar to those in interpolation/regression reported in Table 4.1. In Figure 4.3, we show selected Gaussian basis as the system-free algorithm iterates and the condition numbers of the $n_X \times n_{sel}$ overdetermined systems. By design, the proposed algorithm always keeps submatrix systems well-conditioned. Automatically, the system-free algorithm uses more trial functions to solve the PDE than its interpolation/regression counterpart. In terms of convergence with respect to n_{sel}, GA is desirable if high accuracy if needed. Error reduction in the case of MS is more gradual and is suitable if one seeks for fast numerical approximation.

Next, we are interested in the effect of solution regularity. We consider a modified Helmholtz equation $-\Delta u + u = f$ in the unit circle subject to Neumann boundary conditions. We take the function with nonsmooth Laplacian in (4.1) as the exact solution. The convergence pattern shown in Table 4.5 matches with Table 4.1. However, due to low regularity of the exact solution, the rapid reduction of error of the GA solution from $n_{sel} = 1024$ to termination disappears. The errors in both kernels stagnate around 10^{-4}. To affirm the advantage of using the proposed methods, we

$\max n_{sel}$	256	512	1024	Terminate
GA	2.1239e+00	7.2920e-01	1.1103e-03	3.8550e-09 ($n_{sel} = 1893$)
MS	2.6413e+00	6.9293e-03	2.3652e-07	6.5507e-08 ($n_{sel} = 1611$)

TABLE 4.4

Example 3: Refining the solution of a Poisson equation with Dirichlet boundary conditions generated by the `peaks` *function and the corresponding* $L^2(\Omega)$ *error.*

$\max n_{sel}$	256	512	1024	Terminate
GA	3.5357e-01	4.4497e-01	2.9196e-04	2.7960e-04 ($n_{sel} = 1599$)
MS	5.1411e-01	7.9600e-04	2.4273e-04	2.1213e-04 ($n_{sel} = 1475$)

TABLE 4.5

Example 4.3: Refining the solution of a modified Helmholtz equation with Neumann boundary conditions generated by the function with nonsmooth Laplacian in (4.1) *and the corresponding* $L^2(\Omega)$ *error.*

show some results of the traditional Kansa methods in Figure 4.4. We use 2500 quasi-random data points and some shape parameter in the range of $0.5 \leq \epsilon \leq 10$ to set up Kansa systems. None of them can match the accuracy we see in Table 4.5.

5. Conclusion. We propose a black-box algorithm for running kernel based collocation methods. Numerical examples of problems with both high and low regularities show that using the Whittle-Matérn-Sobolev (MS) kernel in the proposed algorithm yields an efficient solver that can produce cheap and reasonable numerical approximations. The coupling is also robust in the sense that the algorithm performs reliably without any bad run despite the random nature of the system-free algorithm. To this end, users only need to specify the interpolation/regression or PDE problem and the domain of computation in order to run the proposed algorithm.

The low discrepancy point sets used in this proposed method are generated in a non-adaptive way. We may consider an extension of the method by transforming such quasi-uniform random points using the approximated solutions in intermediate steps, resulting in more general non-uniform random point sets that are generated adaptively. However, the (unweighted) least squares approach may cause problems because when collocation points are very unevenly distributed, then regions with denser points will actually get heavier weight. Therefore, we have to use weighted least squares, where the weights should also be adaptive. This extension seems appealing and will be left for future endeavor.

Acknowledgements. This work was partially supported by the Hong Kong Research Grant Council GRF Grants, the Hong Kong Baptist University FRG Grants, and the National Natural Science Foundation of China (11528205).

REFERENCES

[1] E. J. Kansa, Multiquadrics—a scattered data approximation scheme with applications to computational fluid-dynamics. II. Solutions to parabolic, hyperbolic and elliptic partial differential equations, Comput. Math. Appl. 19 (8-9) (1990) 147–161.
[2] W. Chen, Z.-J. Fu, C. Chen, Recent Advances in Radial Basis Function Collocation Methods, Springer-Verlag, Berlin Heidelberg, 2014.
[3] C. S. Chen, C. M. Fan, P. H. Wen, The method of approximate particular solutions for solving elliptic problems with variable coefficients, Int. J. Comput. Methods 8 (3) (2011) 545–559.
[4] E. J. Kansa, J. Geiser, Numerical solution to time-dependent 4D inviscid Burgers' equations, Eng. Anal. Bound. Elem. 37 (3) (2013) 637–645.

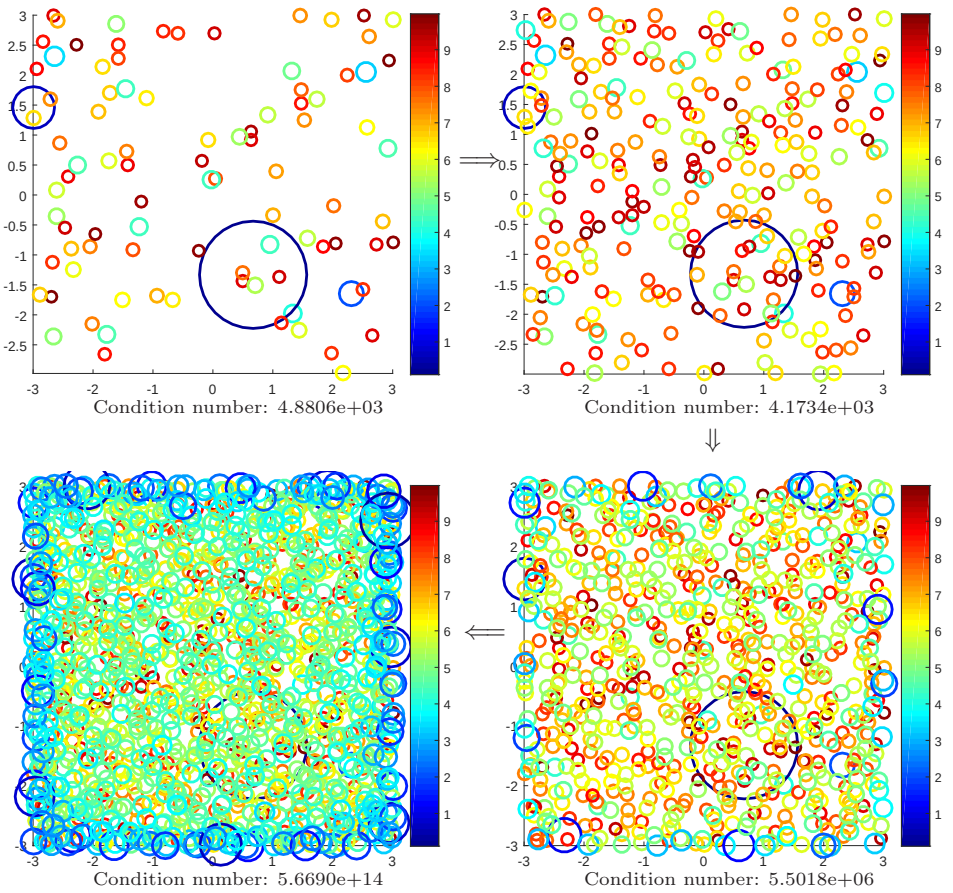


FIG. 4.3. *Example 4.3: Locations of the selected flat Gaussian basis with* $\epsilon \leq 10$ *and the condition numbers of the resulting overdetermined matrix corresponding to Table 4.4.*

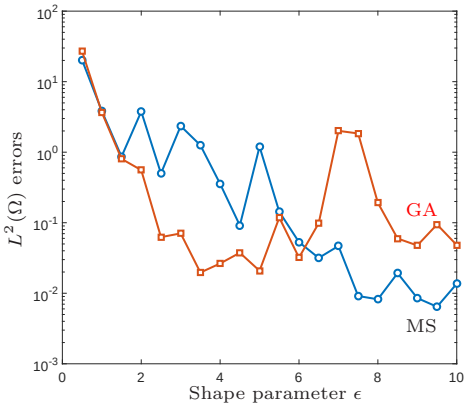


FIG. 4.4. *Example 4.3:* $L^2(\Omega)$ *errors when solving the modified Helmholtz equation with Neumann boundary conditions in Table 4.5 by traditional Kansa methods with 2500 quasi-random data points and various constant shape parameters.*

[5] W. Li, M. Li, C. S. Chen, X. Liu, Compactly supported radial basis functions for solving certain high order partial differential equations in 3D, Eng. Anal. Bound. Elem. 55 (SI) (2015) 2–9.
[6] G. Pang, W. Chen, Z. Fu, Space-fractional advection-dispersion equations by the Kansa method, J. Comput. Phys. 293 (SI) (2015) 280–296.
[7] H. Y. Hu, J. S. Chen, W. Hu, Weighted radial basis collocation method for boundary value problems., Int. J. Numer. Methods Eng. 69 (13) (2007) 2736–2757.
[8] C.-S. Liu, W. Chen, Z.-J. Fu, A multiple-scale MQ-RBF for solving the inverse Cauchy problems in arbitrary plane domain, Eng. Anal. Bound. Elem. 68 (Supplement C) (2016) 11 – 16.
[9] K. C. Cheung, L. Ling, R. Schaback, H^2–convergence of least-squares kernel collocation methods (SIAM J. Numer. Anal., Accepted, 2017).
[10] A. H.-D. Cheng, M. A. Golberg, E. J. Kansa, G. Zammito, Exponential convergence and h-c multiquadric collocation method for partial differential equations, Numer. Methods Partial Differential Equations 19 (5) (2003) 571–594.
[11] L. Ling, A fast block-greedy algorithm for quasi-optimal meshless trial subspace selection, SIAM J. Sci. Compt. 38 (2) (2016) A1224–A1250.
[12] L. Ling, An adaptive–hybrid meshfree approximation method, Int. J. Numer. Methods Eng. 89 (5) (2011) 637–657.
[13] N. A. Libre, A. Emdadi, E. J. Kansa, M. Shekarchi, M. Rahimian, A fast adaptive wavelet scheme in RBF collocation for nearly singular potential PDEs, Comput. Model. Eng. Sci. 38 (3) (2008) 263–284.
[14] N. A. Libre, A. Emdadi, E. J. Kansa, M. Shekarchi, M. Rahimian, A multiresolution prewavelet-based adaptive refinement scheme for RBF approximations of nearly singular problems, Eng. Anal. Bound. Elem. 33 (7) (2009) 901–914.
[15] A. Okabe, B. Boots, K. Sugihara, S. N. Chiu, Spatial Tessellations – Concepts and Applications of Voronoi Diagrams, 2nd Edition, Wiley, Chichester, 2000.
[16] H. Niederreiter, Random Number Generation and Quasi-Monte Carlo Methods, Society for Industrial and Applied Mathematics, Philadelphia, 1992.
[17] I. Sobol, B. Shukhman, Quasi-random points keep their distance, Math. Comput. Simulation 75 (34) (2007) 80–86.
[18] L. Kocis, W. J. Whiten, Computational investigations of low-discrepancy sequences, ACM Trans. Math. Softw. 23 (2) (1997) 266–294.
[19] N. Kopytov, E. Mityushov, The method for uniform distribution of points on surfaces in multi-dimensional euclidean space, Intellectual Archive (2013) Item id:1170.
[20] L. Ling, R. Schaback, An improved subspace selection algorithm for meshless collocation methods, Int. J. Numer. Methods Eng. 80 (13) (2009) 1623–1639.
[21] L. Ling, R. Opfer, R. Schaback, Results on meshless collocation techniques, Eng. Anal. Bound. Elem. 30 (4) (2006) 247–253.
[22] E. Acar, Simultaneous optimization of shape parameters and weight factors in ensemble of radial basis functions, Structural Multidisciplinary Optimization 49 (6) (2014) 969–978.
[23] V. Bayona, M. Moscoso, M. Kindelan, Optimal constant shape parameter for multiquadric based rbf-fd method, J. Comput. Phys. 230 (19) (2011) 7384–7399.
[24] S. Simonenko, V. Bayona, M. Kindelan, Optimal shape parameter for the solution of elastostatic problems with the RBF method, J. Eng. Math. 85 (1) (2014) 115–129.
[25] C. H. Tsai, J. Kolibal, M. Li, The golden section search algorithm for finding a good shape parameter for meshless collocation methods, Eng. Anal. Bound. Elem. 34 (8) (2010) 738–746.
[26] C. M. C. Roque, A. J. M. Ferreira, R. M. N. Jorge, An optimized shape parameter radial basis function formulation for composite and sandwich plates using higher order formulations, J. Sandwich Structures Materials 12 (3, SI) (2010) 279–306.
[27] C. M. C. Roque, A. J. M. Ferreira, Numerical experiments on optimal shape parameters for radial basis functions, Numer. Methods for PDEs 26 (3) (2010) 675–689.
[28] E. J. Kansa, R. E. Carlson, Improved accuracy of multiquadric interpolation using variable shape parameters, comput.. Math. Appl. 24 (12) (1992) 99–120.
[29] B. Fornberg, J. Zuev, The Runge phenomenon and spatially variable shape parameters in RBF interpolation, Comput. Math. Appl. 54 (3) (2007) 379–398.
[30] A. Golbabai, E. Mohebianfar, H. Rabiei, On the new variable shape parameter strategies for radial basis functions, comput.. Appl. Math. 34 (2) (2015) 691–704.
[31] S.-T. Chiu, Bandwidth selection for kernel density estimation, Annals Stat. 19 (4) (1991) 1883–1905.
[32] P. Hall, S. J. Sheather, M. C. Jones, J. S. Marron, On optimal data-based bandwidth selection in kernel density estimation, Biometrika 78 (2) (1991) 263–269.
[33] N.-B. Heidenreich, A. Schindler, S. Sperlich, Bandwidth selection for kernel density estimation:

a review of fully automatic selectors, Adv. Stat. Anal. 97 (4) (2013) 403–433.
[34] M. C. Jones, J. S. Marron, S. J. Sheather, A brief survey of bandwidth selection for density estimation, J. American Stat. Association 91 (433) (1996) 401–407.
[35] S. J. Sheather, M. C. Jones, A reliable data-based bandwidth selection method for kernel density estimation, J. Royal Stat. Society. Series B (Methodological) 53 (3) (1991) 683–690.
[36] M. P. Wand, M. C. Jones, Kernel Smoothing, Chapman and Hall, London, 1995.
[37] J. Fan, I. Gijbels, Variable bandwidth and local linear regression smoothers, Annals Stat. 20 (4) (1992) 2008–2036.
[38] J. Fan, I. Gijbels, Data-driven bandwidth selection in local polynomial fitting: Variable bandwidth and spatial adaptation, J. Royal Stat. Society. Series B (Methodological) 57 (2) (1995) 371–394.
[39] J. Fan, I. Gijbels, Local Polynomial Modelling and its Applications, Chapman and Hall, London, 1996.
[40] O. V. Lepski, E. Mammen, V. G. Spokoiny, Optimal spatial adaptation to inhomogeneous smoothness: an approach based on kernel estimates with variable bandwidth selectors, Annals Stat. 25 (3) (1997) 929–947.
[41] H.-G. Müller, U. Stadtmüller, Variable bandwidth kernel estimators of regression curves, Annals Stat. 15 (1) (1987) 182–201.
[42] G. E. Fasshauer, Meshfree approximation methods with Matlab, Interdisciplinary Mathematical Sciences 6, World Scientific, Hackensack, NJ:, 2007.