Open access • Journal Article • DOI:10.1007/S10915-014-9935-9

A Radial Basis Function Partition of Unity Collocation Method for Convection---Diffusion Equations Arising in Financial Applications — Source link

Ali Safdari-Vaighani, Alfa Heryudono, Elisabeth Larsson

Institutions: Allameh Tabataba'i University, University of Massachusetts Dartmouth, Uppsala University

Published on: 01 Aug 2015 - Journal of Scientific Computing (Springer US)

Topics: Collocation method, Meshfree methods, Numerical partial differential equations, Partition of unity and Finite difference

Journal of Scientific Computing manuscript No.
(will be inserted by the editor)

A radial basis function partition of unity collocation method for convection-diffusion equations arising in financial applications

Ali Safdari-Vaighani · Alfa Heryudono · Elisabeth Larsson

Received: date / Accepted: date

Abstract Meshfree methods based on radial basis function (RBF) approximation are of interest for numerical solution of partial differential equations (PDEs) because they are flexible with respect to geometry, they can provide high order convergence, they allow for local refinement, and they are easy to implement in higher dimensions. For global RBF methods, one of the major disadvantages is the computational cost associated with the dense linear systems that arise. Therefore, research is currently directed towards localized RBF approximations such as the RBF partition of unity collocation method (RBF–PUM) proposed here. The objective of this paper is to establish that RBF–PUM is viable for parabolic PDEs of convection-diffusion type. The stability and accuracy of RBF-PUM is investigated partly theoretically and partly numerically. Numerical experiments show that high-order algebraic convergence can be achieved for convection-diffusion problems. Numerical comparisons with finite difference and pseudospectral methods have been performed, showing that RBF–PUM is competitive with respect to accuracy, and in some cases also with respect to computational time. As an application, RBF–PUM is employed for a two-dimensional American option pricing problem. It is shown that using a node layout that captures the solution features improves the accuracy significantly compared with a uniform node distribution.

Keywords collocation method · meshfree · radial basis function · partition of unity · RBF–PUM · convection-diffusion equation · American option

Mathematics Subject Classification (2000) MSC 65M70 · MSC 35K15

The work of the second author was supported in part by Marie Curie Grant FP7 235730, AFOSR grant FA-9550-12-1-0224, and NSF grant DMS 1318427.

A. Safdari-Vaighani
Department of Mathematics and Statistic, Allameh Tabataba'i University, Tehran, Iran
E-mail: a_safdari@iust.ac.ir

A. Heryudono
Department of Mathematics, University of Massachusetts Dartmouth, Dartmouth, Massachusetts, USA
E-mail: aheryudono@umassd.edu

E. Larsson
Department of Information Technology, Uppsala University, Uppsala, Sweden
E-mail: elisabeth.larsson@it.uu.se

1 Introduction

Convection-diffusion equations are ubiquitous in physics and chemistry as models for flow problems or heat transfer, but they also arise in other non-physical application fields. The solution of a convection-diffusion problem can be interpreted as the probability distribution of one or more underlying stochastic processes. This is the view taken in financial applications where convection-diffusion problems therefore are abundant.

There are two main classes of financial problems in this category. The first problem class is valuation of financial derivatives such as options. Assuming that the underlying asset prices are modeled by Brownian motion together with a (positive) drift under a no arbitrage assumption leads to the original Black–Scholes equation [6]. In the one-dimensional case, with one underlying asset, this problem has a closed form solution. However, for several underlying assets the corresponding partial differential equation (PDE) is a high-dimensional generalization [8,23] of the Black–Scholes equation, which needs to be solved by numerical methods. This is the test case that we will consider in this paper. However, more advanced valuation models involve jump diffusion in the asset price processes [29,22,7] or jumps in the (stochastic) volatility of the assets [5]. This leads to partial integro-differential equations or fractional PDEs instead of PDEs, which require special numerical treatment.

The second problem class is calibration or parameter inference, where appropriate problem parameters describing drift and diffusion are sought from observed market data. Given one market observation, the forward Kolmogorov equation (of convection-diffusion type) describes the transition probability density for the next observation (in time) under a given model. The forward Kolmogorov equation needs to be solved many times for each observation with different model parameters. These solutions form the basis for, e.g., a maximum likelihood estimate of the model parameters [9].

Meshfree methods based on radial basis functions (RBFs) are of general interest for solving PDEs because they can provide high-order or spectral convergence for smooth solutions in complex geometries. In finance, geometries are mostly of hypercube type, meaning that ordinary spectral methods would easily apply. However, it has been shown in [33] that for some types of options, solving the pricing problems on a simplex domain instead of a hypercube leads to significant savings in computational time. If a (quasi) uniform node distribution is used, the number of unknowns is reduced by a factor of $d!$ in d dimensions. Furthermore, another important advantage of meshfree methods is that adaptive refinement can be applied locally without the necessity of preserving the integrity of an underlying grid. Typically, in valuation problems, the features of the solution are located in the vicinity of a lower dimensional manifold determined by the contract function of the financial derivative. Similarly for the Kolmogorov problems, the probability density is concentrated to certain regions. Finally, RBF-based methods are easy to implement in any number of dimensions as the only geometrical information they use is pairwise distances between node points.

In [10,33,4], meshfree methods based on RBF approximation have been shown to perform better than finite difference methods for option pricing problems in one and two spatial dimensions. Similar problems have also been solved in [44,17]. Forward Kolmogorov problems have been solved in [2,3] with promising results. However, all of these papers employ global RBF collocation methods, leading to dense linear systems, and computational costs that become prohibitive as the number of dimensions increase [25]. This problem is partly addressed in [4] where a tensor product formulation is exploited. However, a tensor product approach also limits the opportunity for local adaptivity.

In a partition of unity (PU) scheme, local approximations on overlapping patches that form a cover of the computational domain are weighted together by compactly supported partition of unity weight functions to form the global approximation. The convergence properties of the local approximations can be leveraged, while local couplings between approximations on different patches are enforced through the PU framework. When RBFs are used locally instead of globally, the computational cost is reduced because the previously dense linear systems then become sparse at the patch level.

PU schemes have been used for interpolation since around 1960 [38,28,15], and more recently, they have also been combined with RBFs in [42] and [11]. PU methods for solving PDEs were introduced and analyzed by Babuška and Melenk [1] in the late 1990's. In the forthcoming paper [26] by Larsson and Heryudono, an RBF-based PU collocation method (RBF–PUM) is introduced for elliptic (time-independent) PDEs. High order algebraic or spectral convergence rates, depending on the type of refinement, are predicted theoretically and confirmed by numerical experiments.

In this paper, we investigate the capability of RBF–PUM for numerical solution of parabolic (time-dependent) PDEs. We will show that the method is viable through analysis and numerical experiments, and compare the results with those of other methods. However, strategies for automatic adaptive node refinement are not pursued here, but left for future work. As a general test problem, we use the two-dimensional convection-diffusion equation, and as a specific test problem in finance, we consider a multi-asset American put option pricing problem.

2 Radial basis function collocation schemes

RBF methods are meshfree and work with data given at scattered node points. Given N distinct points $x_1,\dots,x_N \in \mathbb{R}^d$ and corresponding scalar function values $u(x_1),\dots,u(x_N)$, the standard RBF interpolation problem is to find an interpolant of the form

$$s(x) = \sum_{j=1}^{N} \lambda_j \phi(\|x - x_j\|), \tag{2.1}$$

where $\|\cdot\|$ is the Euclidean norm, $\lambda_j \in \mathbb{R}$ for $j = 1,\dots,N$, and ϕ is a real-valued function such as the inverse multiquadric $\phi(r) = \frac{1}{\sqrt{\varepsilon^2 r^2+1}}$ or the Gaussian $\phi(r) = e^{-\varepsilon^2 r^2}$. The parameter ε is called a shape parameter and governs the flatness of the RBFs. It has a significant effect on the accuracy of the RBF approximation. The coefficients $\lambda_1,\dots,\lambda_N$ are determined by enforcing the conditions $s(x_i) = u(x_i)$, $i = 1,\dots,N$. Imposing these conditions leads to a symmetric linear system of equations

$$A\underline{\lambda} = \underline{u}, \tag{2.2}$$

where $A_{ij} = \phi(\|x_i - x_j\|)$, $i,j = 1,\dots,N$, $\underline{u} = [u(x_1)\dots u(x_N)]^T$, and $\underline{\lambda} = [\lambda_1 \dots \lambda_N]^T$. When $\underline{\lambda}$ is known, we can with this notation evaluate the RBF interpolant at a point x as

$$s(x) = \bar{\phi}(x)\underline{\lambda}, \tag{2.3}$$

where $\bar{\phi}(x) = [\phi(\|x - x_1\|),\dots,\phi(\|x - x_N\|)]$.

In the following derivations, we have chosen to express the interpolant in Lagrange form, using cardinal basis functions. The cardinal basis functions, $\psi_j(x)$, $j = 1,\dots,N$, have the property

$$\psi_j(x_i) = \begin{cases} 1 & \text{if } i = j, \\ 0 & \text{if } i \neq j, \end{cases} \qquad j = 1,\dots,N, \tag{2.4}$$

leading to the alternative formulation for the interpolant

$$s(x) = \bar{\psi}(x)\underline{u}, \tag{2.5}$$

where $\bar{\psi}(x) = [\psi_1(x), \ldots, \psi_N(x)]$. Combining (2.3), (2.5), and (2.2) leads to the following relation between the cardinal basis and the original radial basis:

$$s(x) = \bar{\psi}(x)\underline{u} = \bar{\phi}(x)\underline{\lambda} = \bar{\phi}(x)A^{-1}\underline{u} \quad \Rightarrow \quad \bar{\psi}(x) = \bar{\phi}A^{-1}. \tag{2.6}$$

This transformation is valid whenever A is non-singular. This holds for distinct node points $x_1, \ldots, x_N$ and commonly used RBFs such as Gaussians, inverse multiquadrics and multiquadrics.

For a linear operator $\mathscr{L}$, we have

$$\mathscr{L}s(x) = \sum_{j=1}^{N} \mathscr{L}\psi_j(x)u(x_j). \tag{2.7}$$

To evaluate $\mathscr{L}s(x)$ at the node points, i.e., to evaluate $\underline{s}_{\mathscr{L}} = [\mathscr{L}s(x_1), \ldots, \mathscr{L}s(x_N)]^T$, we need the differentiation matrix $\Psi_{\mathscr{L}} = [\mathscr{L}\psi_j(x_i)]_{i,j=1,\ldots,N}$. Using relation (2.6), this leads to

$$\underline{s}_{\mathscr{L}} = \Psi_{\mathscr{L}}\underline{u} = \Phi_{\mathscr{L}}A^{-1}\underline{u}, \tag{2.8}$$

where $\Phi_{\mathscr{L}} = [\mathscr{L}\phi(\|x-x_j\|)|_{x=x_i}]_{i,j=1,\ldots,N}$.

When the Lagrangian form of the RBF interpolation method is used in the context of solving a time-dependent PDE problem, the solution $u(x,t)$ is approximated by

$$s(x,t) = \sum_{j=1}^{N} \psi_j(x)u_j(t), \tag{2.9}$$

where $u_j(t) \approx u(x_j,t)$ are the unknown functions to determine.

3 The radial basis function based PUM

This section defines the RBF–PUM collocation method for time-dependent PDEs in terms of its weight functions and local RBF approximations.

3.1 The partition of unity weight functions

Let $\Omega \subset \mathbb{R}^d$ be an open set, and let $\{\Omega_i\}_{i=1}^{M}$ be an open cover of Ω satisfying a pointwise overlap condition and that

$$\forall x \in \Omega \quad I(x) = \{j | x \in \Omega_j\}, \quad \operatorname{card}(I(x)) \le K, \tag{3.1}$$

where the constant K is independent of the number of patches M. In the RBF–PUM, the global approximation function $s(x)$ in Ω to the solution function $u(x)$ is constructed as

$$s(x) = \sum_{j=1}^{M} w_j(x)s_j(x), \tag{3.2}$$

where s_j is an RBF approximation of $u(x)$ on patch Ω_j and $w_j : \Omega_j \to \mathbb{R}$ are compactly supported, non-negative weight functions subordinate to the cover. The partition of unity weight functions w_j, which also occur under the name *shape functions*, are constructed using Shepard's method

$$w_j(x) = \frac{\varphi_j(x)}{\sum_{k\in I(x)} \varphi_k(x)}, \quad j = 1,\dots,M, \tag{3.3}$$

where $\varphi_j(x)$ are compactly supported functions with support on Ω_j. Here, we select compactly supported Wendland functions [41] such as

$$\varphi(r) = \begin{cases} (1-r)^4(4r+1) & \text{if } 0 \le r \le 1, \\ 0 & \text{if } r > 1, \end{cases} \tag{3.4}$$

for the construction of the weight functions. Let $\{X_j\}_{j=1}^M$ be the center points, and $\{R_j\}_{j=1}^M$ be the radii of the circular, spherical, or hyper-spherical patches Ω_j, $j = 1,\dots,M$. Non-negativity and compact support are guaranteed if the weight functions are generated using

$$\varphi_j(x) = \varphi\left(\frac{\|x - X_j\|}{R_j}\right), \quad j = 1,\dots,M. \tag{3.5}$$

It follows from (3.3) that the weight functions $w_j(x)$ satisfy the partition of unity property

$$\sum_{j\in I(x)} w_j(x) = 1. \tag{3.6}$$

Moreover, the equations (3.4)-(3.5) show that $w_j(x) = 0$, $\forall j \notin I(x)$. Therefore, equation (3.2) can be rewritten as

$$s(x) = \sum_{j\in I(x)} w_j(x) s_j(x). \tag{3.7}$$

If the functions $s_j(x)$, $j = 1,\dots,M$ from equation (3.7) are local interpolants with $s_j(x_i) = u(x_i)$ for each node point $x_i \in \Omega_j$, then the global PU approximant inherits the interpolation property of the local interpolants, i.e.

$$s(x_i) = \sum_{j\in I(x_i)} w_j(x_i) s_j(x_i) = u(x_i) \sum_{j\in I(x_i)} w_j(x_i) = u(x_i). \tag{3.8}$$

The patches can be of any (regular enough) geometrical shape such as squares, cubes, circles, and spheres. The common requirement for all shapes of patches is that they cover the domain and the boundary. In this paper, circular and elliptic patches will be employed. In the case of elliptic patches, the functions used for generating the weight functions are modified to have support on an ellipse instead of a circle. Exactly how this is done is described in Section 7.1.

When we use these types of patches, the overlap between patches can be regulated, and covering ensured, by adjusting the radius of the patches. Flexibility in selection of the radius of the patches is another advantage of the local properties of the PUM. Figures 4 and 16 demonstrate the discretization of a square domain with circular and elliptic patches.

3.2 RBF–PUM for time-dependent problems

In RBF–PUM, the solution $u(x,t)$ for a time-dependent problem is approximated by

$$s(x,t) = \sum_{j\in I(x)} w_j(x) s_j(x,t), \tag{3.9}$$

where $s_j(x,t)$ is an RBF approximant of the type (2.9) on Ω_j, i.e.

$$s_j(x,t) = \sum_{k\in J(\Omega_j)} \psi_k(x) u_k(t), \tag{3.10}$$

where $J(\Omega_j) = \{k | x_k \in \Omega_j\}$ is the set of node points in Ω_j. Combining (3.9) and (3.10), we can express the global approximant as

$$s(x,t) = \sum_{j\in I(x)} w_j(x) \sum_{k\in J(\Omega_j)} \psi_k(x) u_k(t) = \sum_{j\in I(x)} \sum_{k\in J(\Omega_j)} \Big(w_j(x)\psi_k(x) \Big) u_k(t), \tag{3.11}$$

Note that by interpolating the initial condition we get $s(x_k,0) = u(x_k,0)$ for all k, but $s(x_k,t) \approx u(x_k,t)$ for $t > 0$.

3.3 Differentiating the RBF–PUM approximant

In order to use the RBF–PU approximation (3.11) for a PDE problem, we need to compute the effect of applying a spatial differential operator $\mathscr{L}$ at the interior node points. Let α and β be multi-indices and adopt common rules for multi-index notation. Then, using Leibniz' rule, a derivative term of order α in the differential operator can be applied to the global approximation (3.11) as

$$\begin{aligned} \frac{\partial^{|\alpha|}}{\partial x^\alpha} s(x,t) &= \sum_{j\in I(x)} \sum_{k\in J(\Omega_j)} \frac{\partial^{|\alpha|}}{\partial x^\alpha} \left(w_j(x)\psi_k(x)\right) u_k(t) \\ &= \sum_{j\in I(x)} \sum_{k\in J(\Omega_j)} \left(\sum_{\beta\le\alpha} \binom{\alpha}{\beta} \frac{\partial^{|\alpha-\beta|} w_j}{\partial x^{\alpha-\beta}}(x) \frac{\partial^{|\beta|}\psi_k}{\partial x^\beta}(x) \right) u_k(t), \end{aligned} \tag{3.12}$$

Fixing $x = x_i$ and k in equation (3.12) gives us the ik-element of the global differentiation matrix corresponding to the α-derivative. For composite linear operators, we sum up the contributions from each term. We denote the global differentiation matrix under operator $\mathscr{L}$ by $W_{\mathscr{L}}$.

3.4 Computational cost for RBF–PUM

In all linear time-dependent PDE test cases we provide here, the two main parts of the computational cost for RBF–PUM are the cost to form and assemble RBF-PUM differentiation matrices and the cost for matrix–vector multiplications to advance solutions in time. Depending on the type of solver, this may be a matrix–vector multiplication for an explicit time step, as part of an iterative solver, or solving the factorized linear system in an implicit method, all with the same order of cost.

Given M patches, each with n_{loc} local nodes, the cost to form and assemble differentiation matrices is $\mathscr{O}(Mn_{\text{loc}}^3)$, where the n_{loc}^3 factor comes from the factorization of the local interpolation matrices, see equations (2.8) and (3.12). This process is embarrassingly parallel in terms of the patches. For the time-stepping process, the sparsity of the resulting differentiation matrix operators results in a cost $\mathscr{O}(Mn_{\text{loc}}^2)$ for the matrix–vector multiplication. This operation is also embarrassingly parallel.

Moreover, if we are given a global unstructured set of N node points initially, we need to determine which nodes fall into which patch. A direct computation of the distance between each node points and the center points of the patches comes with a cost $\mathscr{O}(MN)$. This may become expensive for large node sets. If the N node points are instead organized with a suitable data structure (e.g. a k-d tree), the cost of associating nodes with patches becomes $\mathscr{O}(Mn_{\text{loc}} \log N)$.

3.5 Characterizing the RBF–PUM approximation

When we later discuss the approximation errors of RBF-PUM, we will do it terms of two levels of discretization parameters. Let $\tilde{\Omega}_j = \Omega_j \cap \Omega$. At the node level, we define the local fill distance

$$h_j = \sup_{x \in \tilde{\Omega}_j} \min_{k \in J(\Omega_j)} \|x - x_k\|, \tag{3.13}$$

which can be explained as measuring the radius of the largest ball empty of nodes in the part of patch j that falls within Ω. We also define the global fill distance

$$h = \max_{1 \le j \le M} h_j. \tag{3.14}$$

At the patch level, we define the patch diameter H_j and the patch fill distance

$$H = \sup_{x \in \Omega} \min_{1 \le j \le M} \|x - X_j\|, \tag{3.15}$$

which similarly measures how densely the patch centers X_j cover the domain. For uniform discretizations, h is proportional to the node distance and H to the patch size.

Furthermore, to discuss results, the chosen type of RBF and its shape parameter ε (see equation (2.1)) needs to be stated. The shape parameter can influence both the approximation accuracy and the conditioning of the linear systems that arise. If not otherwise declared, ε is assumed to be the same for all basis functions, but it can also be varied according to location.

4 The unsteady convection-diffusion equation

Consider an unsteady convection-diffusion equation of the form

$$\frac{\partial u(x,t)}{\partial t} = \kappa \Delta u(x,t) + v \cdot \nabla u(x,t) \equiv \mathscr{L}u(x,t), \quad x \in \Omega \subset \mathbb{R}^d, \quad t > 0, \tag{4.1}$$

where κ is the diffusion coefficient, v is a constant velocity vector, and $u(x,t)$ may represent concentration or temperature for mass or heat transfer, respectively. This equation also serves as a simplified model problem for the Black–Scholes equation and other equations in financial mathematics. The equation (4.1) must be supplemented with an initial condition of the form

$$u(x,0) = f_0(x), \tag{4.2}$$

and boundary conditions

$$\mathscr{B}u(x,t) = f(x,t), \quad x \in \partial\Omega, \quad t > 0, \tag{4.3}$$

where $\mathscr{B}$ can be a Dirichlet, a Neumann or a mixed boundary operator. In the case of Dirichlet boundary conditions, if we use equation (3.12) for the differentiation matrices, and collocate the PDE (4.1) at the interior node points, we get the system of ODEs

$$S'(t) = \left(\kappa W_{\Delta,I} + v\cdot W_{\nabla,I}\right)S(t) + \left(\kappa W_{\Delta,b} + v\cdot W_{\nabla,b}\right)F(t), \tag{4.4}$$

where $W_{\cdot,I}$ contains the columns of the differentiation matrix corresponding to interior nodes and $W_{\cdot,b}$ contains the columns of the differentiation matrix corresponding to the boundary nodes. The vector $S(t) = [u_1(t),\ldots,u_{N_I}(t)]^T$ contains the unknown functions at the interior node points and the vector $F(t) = [f(x_{N_I+1},t),\ldots,f(x_N,t)]^T$ contains the known boundary values. The matrices $W_{\nabla,\cdot}$ are vector valued and the dot product with the velocity v should be taken for each node point.

The system of ODEs in equation (4.4) can be solved in MATLAB for example with the ODE solver command `ode15s`, which is suitable for stiff ODEs, or with any other common time stepping method.

4.1 Error estimate

In the calculation of an upper bound for the semidiscrete error, we need the following three functions: the exact solution $u(x,t)$, the RBF approximation $s(x,t)$ from (3.11), and the auxiliary function $z(x,t)$, which interpolates the exact solution at each time

$$z(x,t) = \sum_{j\in I(x)} \sum_{k\in J(\Omega_j)} (w_j(x)\psi_k(x))u(x_k,t). \tag{4.5}$$

The initial conditions for all three functions coincide at the collocation points. That is,

$$s(x_i,0) = z(x_i,0) = u(x_i,0), \quad 1 \le i \le N_I. \tag{4.6}$$

We define the error function $e(x,t) = u(x,t) - s(x,t)$ and the interpolation error $\mathscr{E}(x,t) = u(x,t) - z(x,t)$. We will derive an estimate for the semidiscrete error. Therefore, we define the vectors $E(t) = [e(x_1,t),\ldots,e(x_{N_I},t)]^T$ and $\mathscr{E}_{\mathscr{L}}(t) = [\mathscr{L}\mathscr{E}(x_1,t),\ldots,\mathscr{L}\mathscr{E}(x_{N_I},t)]^T$, as well as $U(t) = [u(x_1,t),\ldots,u(x_{N_I},t)]^T$ and $Z(t) = [z(x_1,t),\ldots,z(x_{N_I},t)]^T$. We also define the discrete spatial operator $Q = \kappa W_{\Delta,I} + v\cdot W_{\nabla,I}$ and the associated matrix multiplying the boundary points, $B = \kappa W_{\Delta,b} + v\cdot W_{\nabla,b}$. For the error function evaluated at the collocation points we have

$$\begin{aligned}
E'(t) &= U'(t) - S'(t)\\
&= \mathscr{L}U(t) - (QS(t) + BF(t))\\
&= \mathscr{L}Z(t) - (QS(t) + BF(t)) + \mathscr{L}U(t) - \mathscr{L}Z(t)\\
&= Q(Z(t) - S(t)) + \mathscr{L}(U(t) - Z(t))\\
&= QE(t) + \mathscr{E}_{\mathscr{L}}(t),
\end{aligned} \tag{4.7}$$

where we have used the fact that $Z(t) = U(t)$ because of the evaluation at the node points, and that both $Z(t)$ and $S(t)$ are PU approximations of the forms (4.5) and (3.11). The system of ODEs (4.7) can be formally integrated to yield

$$E(t) = \int_0^t e^{Q(t-\tau)} \mathscr{E}_{\mathscr{L}}(\tau) d\tau. \tag{4.8}$$

A simple worst case estimate for the semidiscrete error becomes

$$\|E(t)\|_\infty = \left\| \int_0^t e^{Q(t-\tau)} \mathscr{E}_{\mathscr{L}}(\tau) d\tau \right\|_\infty \leq \left\| \int_0^t e^{Q(t-\tau)} d\tau \right\|_\infty \max_{0 \leq \tau \leq t} \|\mathscr{E}_{\mathscr{L}}(\tau)\|_\infty. \tag{4.9}$$

If we then assume that Q can be diagonalized with eigenvector matrix V and (diagonal) eigenvalue matrix Λ, and use $e^{Qt} = V e^{\Lambda t} V^{-1}$ together with $\int e^{\Lambda t}\, dt = \Lambda^{-1} e^{\Lambda t}$, we can evaluate the norm of the integral to

$$E_Q \equiv \left\| \int_0^t e^{Q(t-\tau)} d\tau \right\|_\infty = \left\| -V \Lambda^{-1} (I - e^{\Lambda t}) V^{-1} \right\|_\infty = \left\| Q^{-1} (e^{Qt} - I) \right\|_\infty. \tag{4.10}$$

Combining (4.9) and (4.10) we get the estimate

$$\|E(t)\|_\infty = E_Q \max_{0 \leq \tau \leq t} \|\mathscr{E}_{\mathscr{L}}(\tau)\|_\infty. \tag{4.11}$$

In order to understand the behavior of E_Q over time, we will investigate its asymptotic behavior. For small enough t, we can Taylor expand the matrix exponential as $e^{Qt} = I + tQ + \frac{t^2}{2} Q^2 + \mathscr{O}(t^3 Q^3)$, which leads to

$$Q^{-1}(e^{Qt} - I) = t + \frac{t^2}{2} Q + \mathscr{O}(t^3 Q^2).$$

For large enough values of t, we instead use the form

$$Q^{-1}(e^{Qt} - I) = V \Lambda^{-1} (e^{\Lambda t} - I) V^{-1}. \tag{4.12}$$

Numerical experiments indicate that all eigenvalues have a negative real part. In this case, the exponential in (4.12) approaches zero as time increases, and the limit value of E_Q becomes $\|Q^{-1}\|_\infty$. In Figures 1 and 2, we investigate numerically how E_Q varies with time, with the problem parameters, and with the RBF-PUM parameters. In all cases, inverse multiquadric RBFs have been used. We can see that $E_Q < 1$ in all the performed experiments. However, the value becomes larger for convection dominated problems and will grow further as $\kappa \to 0$. A smaller shape parameter value (see equation (2.1)) leads to a larger value of E_Q, although the differences are not large in the range of ε-values we can explore without running into ill-conditioning. There is some variation with the discretization parameters, but no strong trend.

In light of the numerically observed decay of e^{Qt}, we can incorporate the damping effect of the parabolic operator on the error over time. Let t_s represent the the time scale over which the matrix exponential becomes negligible. Then we can split the integral in (4.9) into two parts as

$$\|E(t)\|_\infty \leq \left\| \int_0^{t-t_s} e^{Q(t-\tau)} d\tau \right\|_\infty \max_{0 \leq \tau \leq t-t_s} \|\mathscr{E}_{\mathscr{L}}(\tau)\|_\infty + \left\| \int_{t-t_s}^{t} e^{Q(t-\tau)} d\tau \right\|_\infty \max_{t-t_s \leq \tau \leq t} \|\mathscr{E}_{\mathscr{L}}(\tau)\|_\infty.$$

The first integral can be made arbitrarily small by increasing t_s, and to emphasize this, we write the estimate as

$$\|E(t)\|_\infty \le E_Q\left(\delta \max_{0\le\tau\le t-t_s} \|\mathscr{E}_{\mathscr{L}}(\tau)\|_\infty + (1-\delta) \max_{t-t_s\le\tau\le t} \|\mathscr{E}_{\mathscr{L}}(\tau)\|_\infty\right), \tag{4.13}$$

indicating that a large initial error loses importance over time. The relevant time scale t_s can be observed in Figure 1 as the time it takes for $E_Q(t)$ to approach its asymptotic value. In the following, when working with the spatial error, we will start from (4.11) for simplicity, but we will keep in mind that we can also combine the spatial estimates with (4.13).

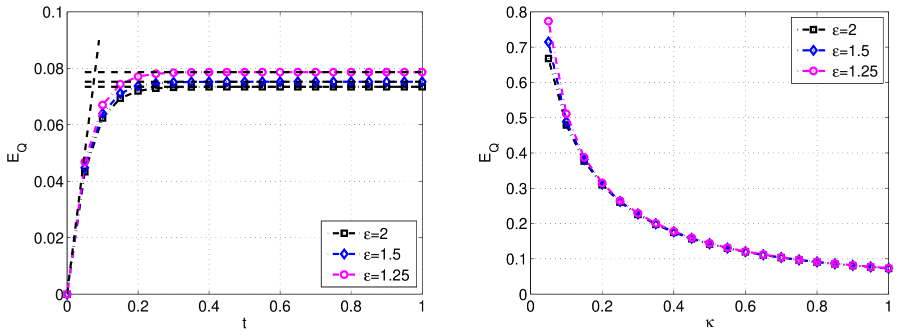

Fig. 1 Left: The variation of E_Q with time for a convection diffusion problem with $\kappa = 1$ and $v = (1,1)$. The discretization parameters are $h = 0.05$ and $H = 0.2$. Initially the value is close to t and then approaches the asymptotic value of $\|Q^{-1}\|$. The asymptotic results are indicated by the dashed trend lines. Right: The variation of the maximum value of E_Q with the diffusion coefficient κ for a fixed $v = (1,1)$ and the same discretization.

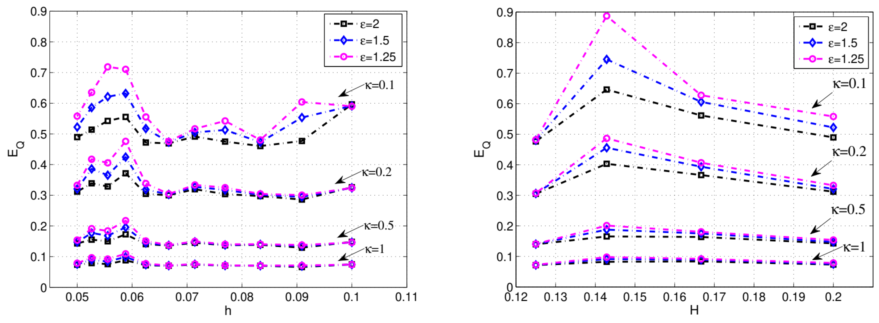

Fig. 2 The dependence of the maximum value of E_Q on the fill distance h when $H = 0.2$ (left) and the dependence on the patch fill distance H when $h = 0.05$ (right) for different values of the shape parameter ε and the diffusion coefficient κ. In both cases, the convection speed is $v = (1,1)$.

We change the focus to the second part of the error estimate (4.11) and expand the interpolation error to get

$$\|E(t)\|_\infty \leq E_Q\left(\kappa \max_{0\leq\tau\leq t}\|\mathscr{E}_\Delta(\tau)\|_\infty + d\|v\|_\infty \max_{0\leq\tau\leq t}\|\mathscr{E}_\nabla(\tau)\|_\infty\right). \tag{4.14}$$

The approximation errors of RBF–PUM are discussed extensively in the forthcoming paper [26]. Here, we briefly recapitulate the parts that relate to the interpolation error to make the present paper self contained. A similar derivation, but with a focus on finitely smooth RBFs can be found in [42]. Assuming that we have constructed a k-stable partition of unity according to the definition in [42]. Then the derivatives of the weight functions satisfy

$$\|D^\alpha w_j\|_{L_\infty(\Omega_j)} \leq \frac{C_\alpha}{H_j^{|\alpha|}}, \quad |\alpha| \leq k, \tag{4.15}$$

where H_j is the diameter of Ω_j. For the local interpolation errors, we rely on the sampling inequalities derived in [36]. Assume that the domains $\tilde{\Omega}_j = \Omega_j \cap \Omega$ are bounded by a Lipschitz boundary and satisfy an interior cone condition. Then we can use the following estimates from [36] for the local interpolation errors and their derivatives when using inverse multiquadrics:

$$\|D^\alpha(z_j - u_j)\|_{L_\infty(\tilde{\Omega}_j)} \leq c_{\alpha,j} h_j^{m_j - \frac{d}{2} - |\alpha|}\|u_j\|_{\mathscr{N}(\tilde{\Omega}_j)}, \tag{4.16}$$

$$\|D^\alpha(z_j - u_j)\|_{L_\infty(\tilde{\Omega}_j)} \leq e^{-\gamma_{\alpha,j}/\sqrt{h_j}}\|u_j\|_{\mathscr{N}(\tilde{\Omega}_j)}, \tag{4.17}$$

where u_j is the global solution restricted to the local domain $\tilde{\Omega}_j$, and z_j is the local RBF interpolant. The norm in the right hand side denoted by $\|\cdot\|_{\mathscr{N}(\cdot)}$ is the native space norm (cf. [11,36]) associated with the type of RBFs employed in the approximation.

Using Leibniz' rule we can express a derivative of the global interpolation error as

$$\mathscr{E}_\alpha = D^\alpha(z - u) = \sum_{j=1}^{M}\sum_{|\beta|\leq|\alpha|}\binom{\alpha}{\beta} D^\beta w_j D^{\alpha-\beta}(z_j - u_j). \tag{4.18}$$

Together with the overlap condition (3.1) this yields the estimate

$$\|\mathscr{E}_\alpha\|_{L_\infty(\Omega)} \leq K \max_{1\leq j\leq M}\sum_{|\beta|\leq|\alpha|}\binom{\alpha}{\beta}\|D^\beta w_j\|_{L_\infty(\Omega_j)}\|D^{\alpha-\beta}(z_j - u_j)\|_{L_\infty(\tilde{\Omega}_j)}. \tag{4.19}$$

We choose to consider two different modes of refinement in order to separate the dependence on h and H in the error estimates. For the first refinement mode, we require the number of nodes per patch to remain constant while we refine the patches, meaning that $H_j/h_j = c$. Then, applying (4.16), we get

$$\|\mathscr{E}_\alpha\|_{L_\infty(\Omega)} \leq K \max_{1\leq j\leq M} C_{H/h,j} H_j^{m_j - \frac{d}{2} - |\alpha|}\|u\|_{\mathscr{N}(\tilde{\Omega}_j)}. \tag{4.20}$$

For the other refinement mode, we fix the patches and then change the number of node points locally or globally. We can then apply (4.17) to get

$$\|\mathscr{E}_\alpha\|_{L_\infty(\Omega)} \leq K \max_{1\leq j\leq M} C_{H,j} e^{-\gamma_j/\sqrt{h_j}}\|u\|_{\mathscr{N}(\tilde{\Omega}_j)}. \tag{4.21}$$

When the estimates are expressed on this form, we clearly see the potential for adaptive refinement in relation to the local behavior of the solution. However, when performing numerical convergence studies in the following sections, we work with (quasi) uniform discretizations, in which case H_j can be replaced by H and h_j by h.

Inserting (4.20) or (4.21) into (4.11) and assuming a uniform discretization we get the global error estimates for the convection diffusion problem as

$$\|E(t)\|_\infty \leq C E_Q H^{m-\frac{d}{2}-2} \max_{0\leq\tau\leq t} \max_j \|u(\tau)\|_{\mathscr{N}(\tilde{\Omega}_j)}, \tag{4.22}$$

$$\|E(t)\|_\infty \leq C E_Q e^{-\gamma/\sqrt{h}} \max_{0\leq\tau\leq t} \max_j \|u(\tau)\|_{\mathscr{N}(\tilde{\Omega}_j)}, \tag{4.23}$$

where the constants m and γ that determine the order of or rate of convergence, are taken as the minimum values over all patches. We conclude that we expect to observe algebraic convergence in H, when the number of nodes per patch is fixed, and spectral convergence in h when the patches are fixed.

Remark: When the shape parameter ε is small, the local RBF approximation is close to polynomial [24], and assuming that the node set is polynomially unisolvent, the rate constant m approximately relates to the multi-variate polynomial degree J supported by the number of node points within the patch as $m = J + 1$. As an example, ten degrees of freedom/nodes in two dimensions corresponds to a polynomial of degree 3, leading to $m = 4$ and an overall convergence rate of H^1.

Remark: The spectral estimate involves $\sqrt{h}$ instead of h. This has to do with boundary effects and can be mitigated if the nodes are distributed more densely near the boundary of the approximation domain [37]. This is not practical in the PU case, since it would mean refining nodes near all patch boundaries. However, the errors at the interior boundaries are in the PU case suppressed by two effects. The weight functions and their derivatives are small near the patch boundaries, and hence the errors at patch boundaries are weighted with small numbers. Furthermore, the problems at boundaries in general are related with lack of information, but in the PU formulation, the boundary values of one patch are connected with the interior values in another patch and actually 'receive' information also from outside the patch.

Remark: For most realistic problems, there are parts of the solution for which $u \notin \mathscr{N}(\tilde{\Omega}_j)$. For smooth solutions, the experience is that approximation works well anyway. See for example [34], where convergence of RBF interpolants to analytic functions is investigated. However, for solutions of limited smoothness, the convergence rates will also be limited accordingly.

5 Numerical results for the convection-diffusion equation

With appropriate initial condition and Dirichlet boundary conditions, the following function is a solution to the unsteady convection-diffusion equation (4.1) in $d = 2$ space dimensions

$$u(x,y,t) = a \exp^{bt}(\exp^{-cx} + \exp^{-cy}), \tag{5.1}$$

where a and b can be chosen freely, and $c = \frac{v\pm\sqrt{v^2+4b\kappa}}{2\kappa} > 0$. The experiments below have been performed with $a = 1$ and $b = 0.1$. The convection velocity is chosen to be $v = (1,1)$ in most experiments, and $\kappa = 1$ is used as default diffusion strength, but other values are used as well.

We discretize the domain $\Omega = [0,1] \times [0,1]$ both uniformly by $N = n^2$ nodes and non-uniformly with a similar number of Halton node points [16]. The discretizations of the square domain Ω are shown in Figure 3. The PU cover consists of $M = m^2$ circular patches. We let the overlap of the patches be 20% of the distance between the centers. An example of patches for the square domain Ω is shown in Figure 4.

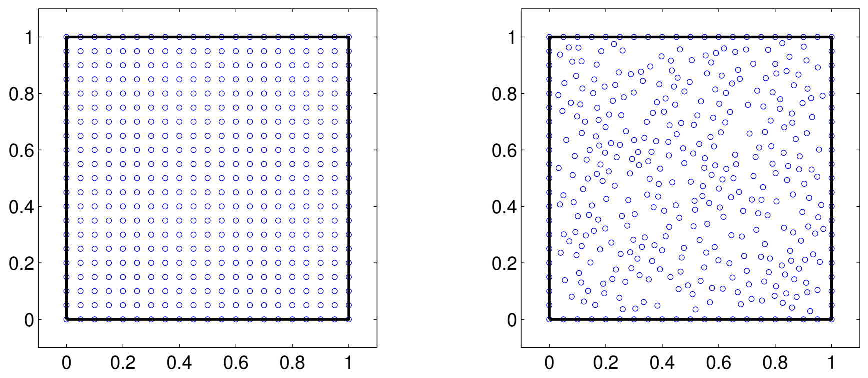

Fig. 3 Uniform and quasi random node distributions.

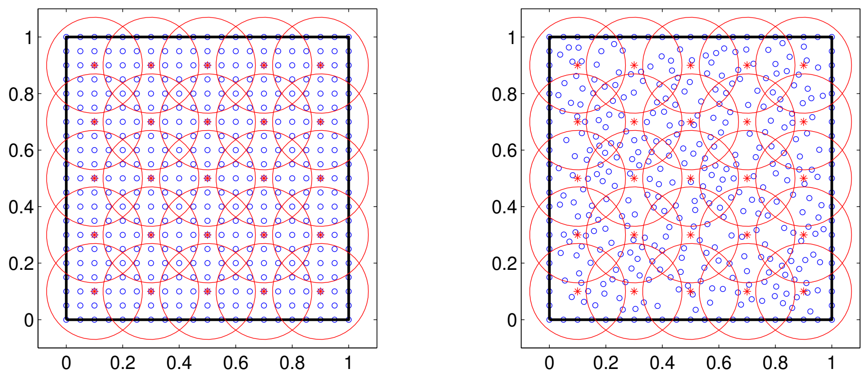

Fig. 4 Partitioning of the square domain with circular patches.

Unless otherwise stated, inverse multiquadric RBFs have been used. The shape parameter ε has not been optimized for accuracy. Instead the range of ε-values has been chosen such that ill-conditioning is avoided. In some cases this has a negative effect on the results. The conditioning problem can be avoided by using a stable method for evaluation of RBF approximations such as the RBF-QR method [13,27], which allows computations for any small value of ε. This will be further discussed and implemented in the forthcoming paper [26].

For the time-stepping, the MATLAB function `ode15s` is used. The MATLAB codes for the two- and three-dimensional convection-diffusion problems can be downloaded from the authors' web sites.

5.1 Properties of the RBF–PUM discretization matrices

Using a partition based approach instead of a global RBF approximation introduces sparsity in the discretization matrices. In Figure 5, the sparsity patterns of the convection-diffusion RBF–PUM matrix Q for two different numbers of patches are shown. More patches lead to more sparsity, but with the same number of nodes, the global convergence is also lower. Even if only the diffusion term is present, the matrices are non-symmetric due to the collocation involving the partition of unity weight functions.

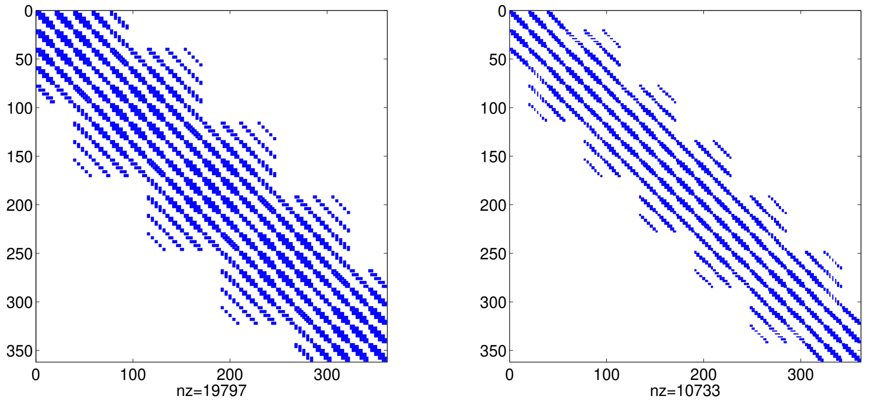

Fig. 5 Sparsity structure of $Q = \kappa W_{\Delta,I} + \nu W_{\nabla,I}$ with 21×21 uniform nodes and 5×5 patches (left) and 7×7 patches (right).

A numerical study of the stability of the semidiscrete problem has been performed, by investigating the spectra and pseudospectra [40] of the RBF–PUM discretization matrices. We define the μ-pseudospectrum of the matrix Q as

$$\Lambda_\mu = \left\{ z \in \mathbb{C} \,|\, \|(zI - Q)^{-1}\| \geq \mu^{-1} \right\}, \quad \mu \geq 0.$$

We identify the eigenvalues (spectrum) Λ of Q with the 0-pseudospectrum Λ_0. We denote the μ-pseudospectral abscissa, the largest real part of the μ-pseudospectrum, by

$$\lambda_\mu^* = \sup_{z \in \Lambda_\mu} (\mathrm{Re}(z)).$$

As the stability for a linear problem is typically not affected by forcing terms(here due to the boundary conditions), we consider the homogeneous semidiscrete problem

$$U'(t) = QU(t),$$

where n and m are the discretization parameters, which has the solution

$$U(t) = e^{Qt} U(0).$$

Following Reddy and Trefethen [35], we first define stability of the semidiscrete problem as

$$\|e^{Qt}\| \leq g(t), \quad \forall t \geq 0,$$

where the function $g(t)$ does not depend on the discretization parameters m and n. Then, again from [35], there is a theorem stating that $\log(\|e^{Qt}\|)$ grows linearly in t if and only if the μ-pseudospectral abscissas grow linearly with μ. Specifically, if

$$\lambda_{\mu}^{*} \leq \omega + C\mu, \quad \forall \mu \geq 0, \tag{5.2}$$

where ω and C are constants, then

$$\|e^{Qt}\| \leq eC(n-1)^2 e^{\omega t}, \tag{5.3}$$

where $(n-1)^2$ is the size of the matrix Q. Figure 6 shows examples of spectra and pseudospectra of Q for a convection dominated problem with $\kappa = 0.001$ and for a problem with strong diffusion $\kappa = 1$. The plots have been generated using EigTool [43].

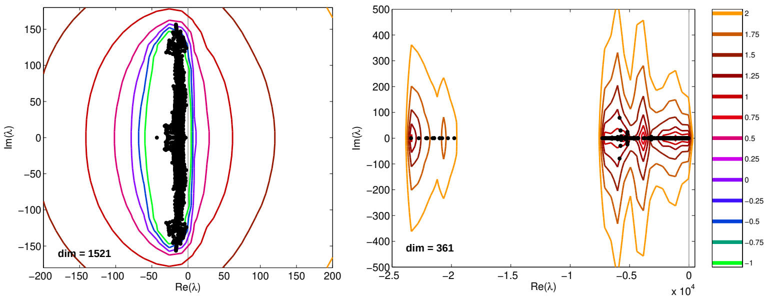

Fig. 6 Pseudospectra (contour lines) and eigenvalues (dots) for the RBF–PUM coefficient matrix Q for $\kappa = 0.001$, $n = 41$, $m = 8$, $\varepsilon = 3$ (left), and $\kappa = 1$, $n = 21$, $m = 5$, $\varepsilon = 1.5$ (right).

The eigenvalue with the largest real part λ_0^* only changes marginally with the discretization, but does depend on κ. For a well resolved problem, $\lambda_0^* < 0$, but approaches zero as $\kappa \to 0$. However, it should be noted that for convection dominated problems, we have observed eigenvalues in the right half plane if the resolution in terms of nodes and/or patches is too low.

If we approximate ω with λ_0^*, then the requirement for stability becomes

$$\lambda_{\mu}^{*} - \lambda_0^{*} \leq C\mu.$$

In the left part of Figure 7, we plot the growth of $\lambda_{\mu}^{*} - \lambda_0^{*}$. If the slope of these curves asymptotically is less than or equal to one, we can find a constant C such that (5.2) and (5.3) are satisfied. Numerically, this holds for the range of μ and the different problem parameters that have been tested. The variation with the shape parameter, the number of nodes, and the number of patches is small, but the amount of diffusion κ has a clearly visible effect.

In the right part of Figure 7, we investigate if the numerically estimated value of C for a fixed problem, but with different discretizations stays bounded. There does not seem to be an increasing trend with node refinement or with patch refinement.

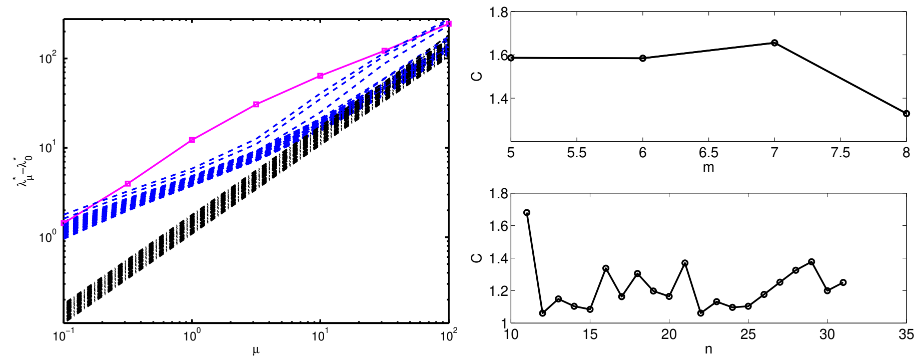

Fig. 7 Left: The growth of the pseudospectral abscissa of Q as a function of the pseudospectrum level μ for different parameter combinations. The three groups correspond to $\kappa = 0.001$ (solid line), $\kappa = 0.1$ (dashed lines), and $\kappa = 1$ (dash-dot lines). In the first case, $M = m^2 = 64$ patches and $n^2 = 41^2$ nodes were used, and the shape parameter was $\varepsilon = 3$. For the other cases, discretizations combining $n = 11,\dots,21$ with $m = 5$, and combining $m = 5,\dots,8$ with $n = 21$ have been tested for $\varepsilon = 1.25, 1.5, 2$. The three dashed lines that deviate somewhat from the pattern correspond to $m = 7$ and $n = 21$, which is an unlucky combination in the sense that some overlap regions between 'diagonal' neighbor patches are empty of nodes. Right: The estimated constant C in the bound on the growth of the pseudospectral abscissa for $\kappa = 1$ for different discretizations. In the top subplot $\varepsilon = 1.5$ and $n = 21$, and in the bottom subplot $\varepsilon = 3$ and $m = 5$.

5.2 Errors and convergence

First, we test how RBF–PUM responds to the type of node distribution and the geometry of the computational domain. Figure 8 displays the absolute error for the uniform and quasi random node distributions shown in Figure 4 with $\varepsilon = 1.25$ at $t = 1$. The errors are of a similar magnitude in both cases, and there are no obvious artifacts due to the geometry of the patches and their overlaps.

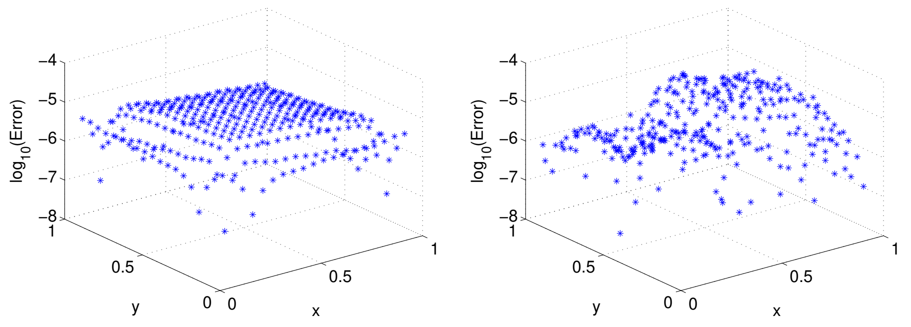

Fig. 8 Absolute error in the solution of the unsteady convection-diffusion equation with 21×21 uniform node and 400 non-uniform node points.

To illustrate the capability of the proposed method for irregularly shaped domains, the same convection-diffusion problem (4.1) as in the previous experiment is solved over the non-convex domain in the left part of Figure 9. The absolute error of the approximation is plotted in Figure 9 at time $t = 1$ with $\varepsilon = 0.75$. The error is again of a similar magnitude,

Table 1 L_∞ error of RBF–PUM for 21×21 uniformly distributed nodes and 400 non-uniformly distributed nodes with $\varepsilon = 1.25$ in the square domain and 325 nodes with $\varepsilon = 0.75$ in the non-convex domain at different times.

t	non-uniform points	uniform points	non-convex domain
0.1	$3.0139e-005$	$9.3965e-006$	$3.4108e-005$
0.5	$3.2595e-005$	$1.0554e-005$	$3.7100e-005$
1.0	$3.4275e-005$	$1.1100e-005$	$3.9008e-005$
3.0	$4.1867e-005$	$1.3554e-005$	$4.7640e-005$
10.0	$8.4308e-005$	$2.7307e-005$	$9.5952e-005$

and apart from generating the nodes and patches for the domain, there is no added difficulty in applying RBF–PUM for this problem.

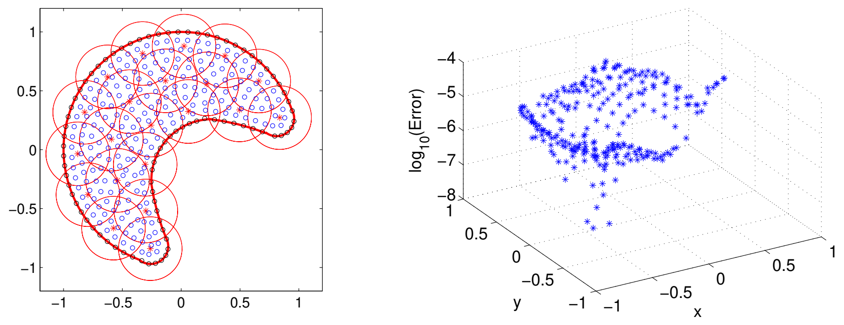

Fig. 9 Left: the partitioning of the non-convex domain with circle patches for 325 points. Right: absolute error in the solution of the unsteady convection-diffusion equation for the node distribution shown in the left figure, at time $t = 1$.

In Table 1, the errors over time for the two discretizations of the square, and for the discretization of the non-convex domain are listed. It can be seen that the error growth with time is slow, and that the accuracy of the three different solutions is similar, with a slight advantage for the uniform distribution.

The convergence of RBF–PUM has been investigated numerically for the two refinement modes described and analyzed in section 4.1 In the first scenario, for a fixed number of patches, uniformly distributed nodes with varying fill distance are employed. In the second scenario, different numbers of patches, ranging from 2×2 to 6×6, with a close to fixed number of local nodes per patch are considered. For the experiment, a fixed shape parameter $\varepsilon = 1.25$ was used.

As shown in Figure 10, increasing the number of local points for a fixed number of patches results in spectral convergence. The rate constants γ from (4.23) are estimated from the experimental results. For the second scenario we get algebraic convergence with respect to H for a fixed number of local nodes. The slopes of the lines in the right figure represent the approximate convergence rates, and show that we can attain high order convergence. The results are even a little better than expected from (4.22).

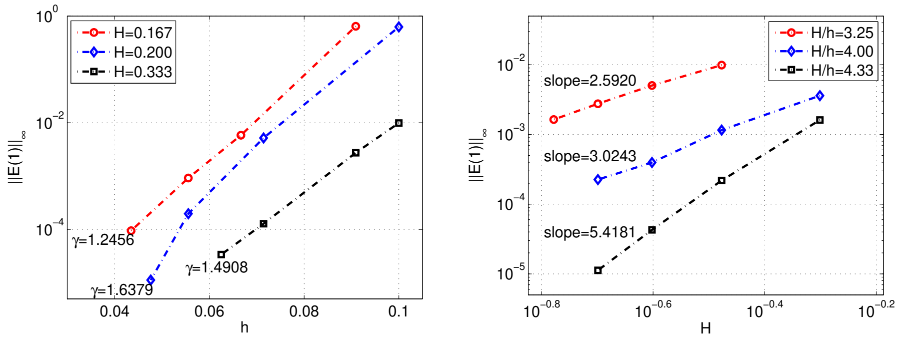

Fig. 10 The convergence as a function of h for three different patch sizes H (left). Convergence as a function of H when the number of nodes per patch is kept (almost) fixed. The three curves correspond to approximately 21, 28, and 36 nodes per interior patch (right). The theoretically expected convergence rates are H^2, H^3, and H^4.

5.3 Comparison with other methods

For a square domain, both finite difference methods (FD) and pseudospectral methods (PS) are easy to implement. Therefore, we will compare the accuracy of RBF–PUM with FD and PS for such a domain, while keeping in mind that RBF–PUM is directly applicable also for other domains. We do not compare timings here, since the implementations are not optimized and the times are quite short. However, generally for the problem sizes and implementations considered here, FD and PS take approximately the same time for a given resolution, while RBF–PUM is about 3 times slower.

Figure 11, shows the error as function of h for the case $\nu = (1,1)$ and $\kappa = 0.1$. The

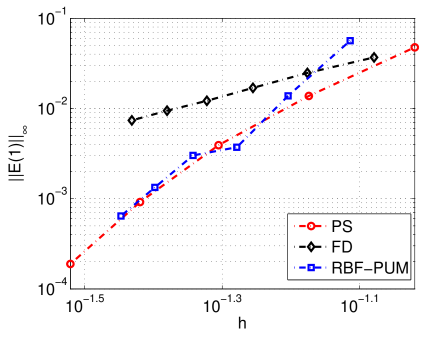

Fig. 11 Error as a function of h for FD, PS, and RBF–PUM with problem parameters $\nu = (1,1)$ and $\kappa = 0.1$ at time $t = 1$. For the RBF-PU method $\varepsilon = \frac{0.1}{h}$ and $H = 0.2$ were used.

accuracy for PS and RBF–PUM is almost the same, while FD is less accurate.

Figure 12, shows the error as function of κ for $\varepsilon = 1.25$, $H = 0.2$, and $h = 0.05$ for two different values of ν. In the left subfigure, the error of the RBF–PUM method tapers off at around 10^{-5}. This is a typical behavior of RBF approximations in the presence of ill-conditioning, and may not be the true behavior of the method. For convection dominated problems, RBF–PUM is more accurate than PS (and FD).

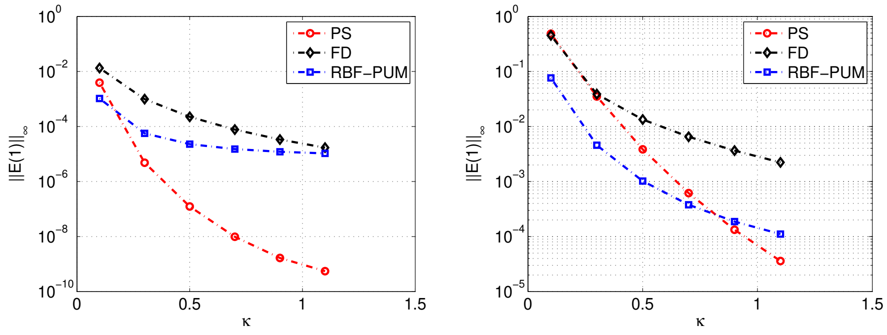

Fig. 12 Error as a function of κ for $v = (1,1)$ (left) and $v = (5,5)$ (right) at $t = 1$. For RBF–PUM $h = 0.05$, $H = 0.2$, and $\varepsilon = \frac{0.1}{h}$ were used.

We conclude that RBF–PUM shows as good as or better approximation properties than PS, at least in convection dominated cases. It is a bit more computationally expensive, but can be applied to arbitrary geometries, and allows for local adaptivity.

5.4 Experiments in three dimensions

We have also solved the convection-diffusion problem (4.1) in three space dimensions in a solid domain bounded by the surface

$$x^2 + y^2 + z^2 - \sin(2x)^2 \sin(2y)^2 \sin(2z)^2 = 1, \tag{5.4}$$

as shown in Figure 13. Boundary conditions at the surface are chosen based on the exact solution

$$u(x,y,z,t) = e^{bt-c(x+y+z)} \tag{5.5}$$

where $b = \frac{1}{10}$ and $c = \sqrt{b/6}$. With that particular choice of exact solution, the vector v in equation (4.1) can be exactly determined as $v = (-c,-c,-c)$.

The solid domain is discretized with a total of $N = 2046$ node points and covered by $M = 512$ patches. Initially, all node points are distributed uniformly. Interior nodes are then slightly perturbed in a random way in the direction towards the boundary. Non-overlapping boxes that cover the domain, which form the basis for constructing the ball cover, are shown in Figure 13.

As in the two-dimensional case, `ode15s` is used for the time-stepping. Figure 14 shows the distribution of the eigenvalues of the three-dimensional convection-diffusion operator with the number of nodes per patch $n_{\text{loc}} = 26$. Gaussian RBFs with shape parameter $\varepsilon = 0.75\bar{R}$, where $\bar{R}$ is the average radius of the ball patches, are used. All eigenvalues lie in the left half of the complex plane. The right subfigure in Figure 14 shows how the error evolves in time for three different values of n_{loc}. The growth in time is very limited and as expected the error decreases with increasing numbers of local nodes.

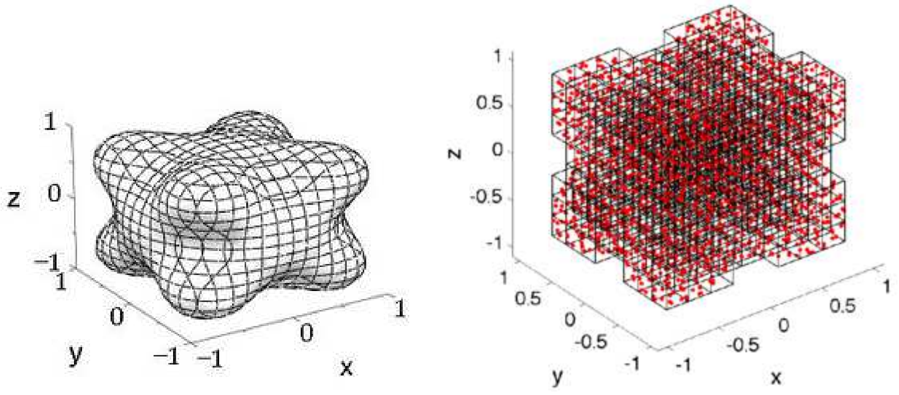

Fig. 13 The solid domain bounded by the surface equation (5.4) (left). The layout of the non-overlapping boxes which form the skeleton for the ball cover. The dots illustrate the node points (right).

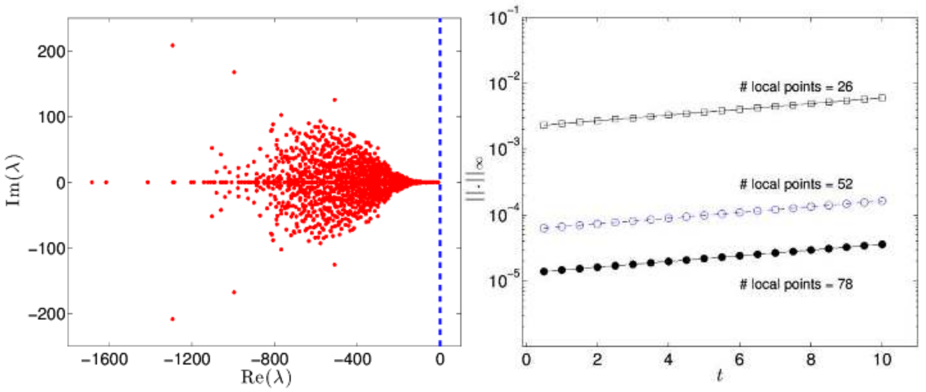

Fig. 14 Eigenvalues of the three-dimensional convection-diffusion operator discretized with Gaussian RBFs in the partition of unity setting with 26 nodes per patch (left). Error of the numerical solutions compared with the exact solution as a function of time (right).

6 Multi-asset American option pricing

The multi-dimensional version [8,23] of the Black–Scholes equation [6] takes the form

$$\frac{\partial P}{\partial t}+\frac{1}{2}\sum_{i=1}^{d}\sum_{j=1}^{d}\rho_{ij}\sigma_i\sigma_j S_i S_j\frac{\partial^2 P}{\partial S_i\partial S_j}+\sum_{i=1}^{d}(r-d_i)S_i\frac{\partial P}{\partial S_i}-rP=0,\quad 0\le t\le T,\tag{6.1}$$

where P is the value of the contract, S_i is the value of the ith underlying asset, T is the time to expiry, d is the number of underlying assets, ρ_{ij} is the correlation between asset i and asset j, σ_i is the volatility of asset i, r is the risk-free interest rate and d_i is the (continuous) dividend yield paid by the ith asset. Equation (6.1) is a final value problem, i.e., the solution is known at time T and the PDE is integrated backwards in time.

The payoff function of the American basket put option is given by

$$F_T(S) = \max(E - \sum_{i=1}^{d} \alpha_i S_i, 0), \tag{6.2}$$

where E is the exercise price of the option and α_i, $i = 1, \dots, d$ are given constants. The final condition is given by

$$P(S,T) = F_T(S), \quad S \in \Omega = \mathbb{R}_+^d. \tag{6.3}$$

The boundary of the computational domain can be divided into two parts. The near-field boundary, where one or more asset prices are zero, and the far-field boundary, where one or more asset-prices tend to infinity.

For the near-field boundary, it can be noted that if one of the asset prices is zero at time t^*, then the asset will be worthless for any $t \geq t^*$, i.e., the solution remains at the boundary. We denote the d near-field boundaries by $\Gamma_i = \{S \in \Omega | S \neq 0, S_i = 0\}$, $i = 1, \dots, d$. Then the boundary values at Γ_i can be propagated by solving a $(d-1)$-dimensional Black–Scholes problem. We denote the solutions of the reduced problems by h_i and use the boundary conditions

$$P(S,t) = h_i(S,t), \quad S \in \Gamma_i, \quad i = 1, \dots, d. \tag{6.4}$$

However, already in [12] it was shown that the problem is well posed without boundary conditions at the near-field boundaries, assuming the Fichera condition on the relative strength of the drift and diffusion term holds. For a more recent discussion of the well-posedness of the problem, see also [20]. In the numerical experiments here, we will use (6.4) as in [10] even if it is not needed. For an example where near-field conditions are not used, see [33].

For put options, the contract becomes worthless as the price of any of the underlying assets tends to infinity. Therefore, we employ the following far-field boundary conditions [21]:

$$\lim_{S_i \to \infty} P(S,t) = 0, \quad S \in \Omega, \ i = 1, \dots, d. \tag{6.5}$$

The American option allows early exercise, which means that at some values of S, where it is more profitable to use the option than to keep it until the expiry date, this will be done. Mathematically, this corresponds to a free boundary problem. This issue can be treated in different ways. Ito and Toivanen [19] as well as Persson and von Sydow [32] use an operator splitting approach. The approach we will use here employs a penalty term as described in [45] and later refined in [30]. The penalty term takes the form

$$\frac{\delta C}{P + \delta - q}, \tag{6.6}$$

and ensures that the solution stays above the payoff function as the solution approaches expiry. Here $0 < \delta \ll 1$ is a small regularization parameter, $C \geq rE$ is a positive constant. The so called *barrier function* $q(S)$ is defined as

$$q(S) = E - \sum_{i=1}^{d} \alpha_i S_i, \tag{6.7}$$

see [45] for a motivation of this choice. Adding the penalty term to the Black–Scholes equation (6.1) for the American option converts it to a fixed domain problem. The penalty term is small enough so that the PDE still resembles the Black–Scholes equation closely. The error introduced by the penalty term is expected to be of the order of δ. The penalty term (6.6) together with equation (6.1) lead to

$$\begin{aligned}\frac{\partial P}{\partial t} + \frac{1}{2}\sum_{i=1}^{d}\sum_{j=1}^{d}\rho_{ij}\sigma_i\sigma_j S_i S_j \frac{\partial^2 P}{\partial S_i \partial S_j} + \sum_{i=1}^{d}(r-d_i)S_i\frac{\partial P}{\partial S_i} - rP \\ + \frac{\delta C}{P+\delta-q} = 0, \quad S \in \Omega,\ 0 \le t \le T. \end{aligned} \tag{6.8}$$

The terminal and boundary conditions on the fixed domain are just like before

$$P(S,T) = F_T(S), \quad S \in \Omega, \tag{6.9}$$
$$P(S,t) = h_i(S,t), \quad S \in \Gamma_i, \quad i = 1,\ldots,d, \tag{6.10}$$
$$\lim_{S_i\to\infty} P(S,t) = 0, \quad S \in \Omega,\ i = 1,\ldots,d. \tag{6.11}$$

6.1 RBF–PUM for American option pricing

Using the RBF–PUM approximation (3.12) and collocating the PDE (6.8) at the node points we get the system of ODEs

$$\begin{aligned} P_I'(t) = -\frac{1}{2}\sum_{i=1}^{d}\sum_{j=1}^{d}\rho_{ij}\sigma_i\sigma_j S_{iI}S_{jI}W_{ij,I}P_I(t) - \sum_{i=1}^{d}(r-d_i)S_{iI}W_{i,I}P_I(t) + rP_I(t) \\ - \frac{\delta C}{P_I(t)+\delta-q} + F(t), \end{aligned} \tag{6.12}$$

where $W_{\cdot,I}$ contains the columns of the differentiation matrix corresponding to interior points, S_{iI} are diagonal matrices containing the respective coordinates of the interior node points, $P_I(t) = [P_1(t),\ldots,P_{N_I}(t)]^T$, and

$$F(t) = -\frac{1}{2}\sum_{i=1}^{d}\sum_{j=1}^{d}\rho_{ij}\sigma_i\sigma_j S_{iI}S_{jI}W_{ij,b}F_b(t) - \sum_{i=1}^{d}(r-d_i)S_{iI}W_{i,b}F_b(t), \tag{6.13}$$

where $W_{\cdot,b}$ contains the columns of the differentiation matrix corresponding to boundary points and $F_b(t) = [P(x_{N_I+1},t),\ldots,P(x_N,t)]^T$ contains the known boundary values.

Going back to the theoretical convergence estimates in section 4.1, there are some features of the American option pricing problem that can degrade the convergence. The jump in the derivative of the initial condition limits the accuracy with which it can be approximated. However, if we perform a split in time of the error terms as suggested in (4.13), we see that the initial error looses importance for the error at later times. A worse problem is the free boundary, where the solution itself is only C^1. With the penalty formulation of the problem, the modified solution is smooth, and we can expect high order convergence with RBF–PUM. However, for the error in relation to the true solution, the convergence order is limited in the patch(es) where the free boundary is located.

7 Numerical experiments for two-asset American option pricing

The option prices are approximated using uniform and non-uniform discretizations of the domain $\Omega = [0, S_{1,\infty}] \times [0, S_{2,\infty}]$. For the numerical illustrations throughout this section, we use the parameter values from [10,31] given by $r = 0.1$, $\sigma_1 = 0.2$, $\sigma_2 = 0.3$, $\rho = 0.5$, $\alpha_1 = 0.6$, $\alpha_2 = 0.4$, $d_1 = 0.05$, $d_2 = 0.01$. The time interval, the computational domain, and the penalty term are defined by $T = 1$, $E = 1$, $S_{i,\infty} = 4E$, $\delta = 0.00001$ and $C = 0.1$. In all experiments, inverse multiquadric RBFs are used, and the arising system of ODEs (6.12) is solved in MATLAB using the ode solver command `ode15s`. The MATLAB code for the American option pricing problem can be downloaded from the authors' web sites.

7.1 A non-uniform space discretization

We note that the payoff function (6.2) possesses a discontinuity in its first derivative at the exercise price, see Figure 15. In practice, the region near the exercise price in the (S_1, S_2) domain is the financially most interesting. Along the S_i-directions we want to have a distribution of node points which is clustered in a neighborhood of the exercise price. By using a tailored node distribution we aim to increase the accuracy of the approximation in the region of interest as well as to capture the initial discontinuity in the solution better. We would like to apply the non-uniform discretization that has recently be employed, e.g., in [39,18].

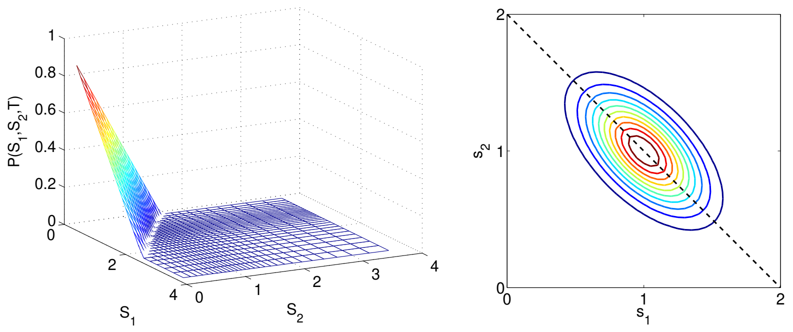

Fig. 15 The payoff function of the two-dimensional American put option (left). Contour lines of the function $\omega(S)$ used for the evaluation of the weighted error norm (right).

In order to cluster nodes around the exercise price E, we define the node coordinates in each direction i through

$$S_{i,j} = E + l \sinh(\xi_j), \quad 0 \le j \le m, \tag{7.1}$$

where $\xi_j \in [\xi_0, \xi_m]$ are equidistant values and l is a parameter that determines the amount of clustering. By the requirement that the nodes should fall in the interval $[0, S_{i,\infty}]$ we can compute the range of ξ to

$$\xi_0 = \sinh^{-1}(-E/l)$$
$$\xi_m = \sinh^{-1}((S_{i,\infty} - E)/l).$$

Note that the centers of the patches are defined with a similar pattern as for the node distribution. In our numerical experiments we have used $l = E/2$ for both nodes and patch centers. When the patch centers are non-uniformly distributed, circular patches do not have a suitable shape. Instead, we use elliptic patches, as illustrated in Figure 16. The Wendland

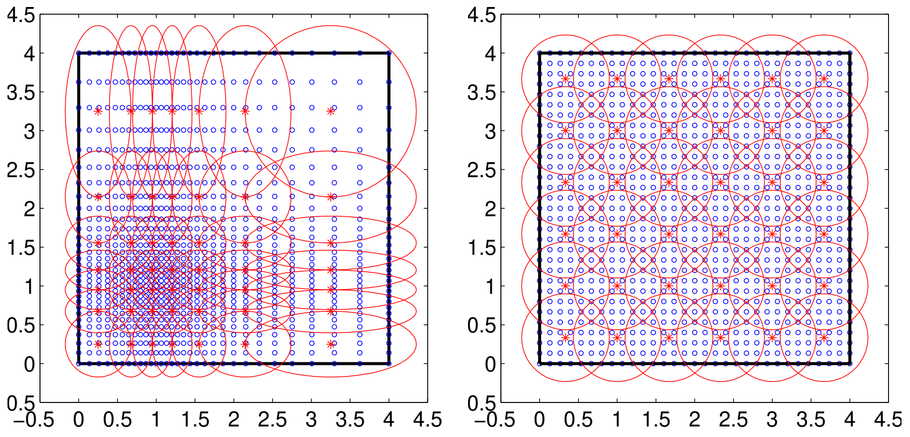

Fig. 16 Discretization of the computational domain for the two-dimensional American option pricing problem with elliptic and circular patches.

function (3.4) is used for constructing the partition of unity weights, but here the argument is scaled anisotropically such that for a point $x = (x_1, x_2)$ in patch Ω_j with center point $X_j = (X_{j,1}, X_{j,2})$

$$\varphi_j(x) = \varphi\left(\sqrt{\frac{(x_1 - X_{j,1})^2}{R_{j,1}^2} + \frac{(x_2 - X_{j,2})^2}{R_{j,2}^2}}\right), \quad j = 1, \ldots, M, \tag{7.2}$$

where $R_{j,\cdot}$ are the radii of the elliptic patches in the different coordinate directions. The resulting function is compactly supported on the elliptic patch. The shape parameters for the radial basis functions used for the local approximations are scaled with respect to the node density in the patch such that

$$\varepsilon_j = \varepsilon \frac{h}{\delta_j}, \tag{7.3}$$

where h is the uniform node distance corresponding to the number of nodes used, and δ_j is the actual minimum node distance within the patch. This makes sense because the adjustment of the node distribution is based on the local smoothness of the solution. Where there are higher derivatives, the shape parameter is larger and the nodes more dense, see also [14].

7.2 Errors and convergence

As mentioned earlier, the region around the strike price is where the solution is of financial interest. Our computational objective is to make the error small in that region. Therefore, we use a weighted error norm of the form

$$E_\omega = \int_\Omega \omega(S)|E(S)|\,dS,$$

where the function $\omega(S)$ is normalized such that $\int_\Omega \omega(S)\,dS = 1$. A similar approach was taken in [33]. Here, we have used a function $\omega(S) \propto e^{-9(S_1+S_2-2)^2} e^{-9((S_1-S_2)/2)^2}$. In Figure 15, the contour lines of the function are shown.

We have compared the accuracy and convergence of RBF–PUM for the American option pricing problem with two different FD implementations. In the left part of Figure 17, RBF–PUM with uniform nodes and non-uniform nodes is compared with an FD implementation that uses a penalty term. This means that the same (fixed domain) PDE is solved by all methods. In the right part of Figure 17 the same comparison is performed while instead using an FD implementation with an operator splitting approach [19,32]. In this case, the reference solution is closer to the actual solution, as the error introduced by the penalty term is eliminated.

From the figures, it can be seen that the convergence rates for the uniform RBF–PUM and uniform FD seem to be quite similar in all cases, but with a smaller error for RBF–PUM. Using the non-uniform RBF–PUM discretization, we achieve significantly better results for the same number of node points. The differences between the two sets of plots are small, but with the penalty reference the error can falsely continue to decrease because the penalty error is not measured.

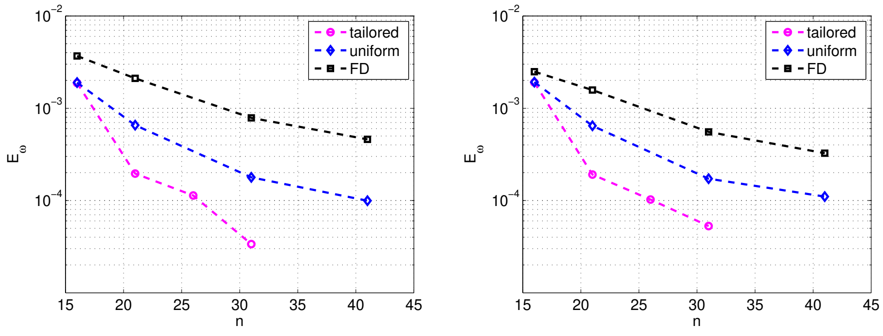

Fig. 17 Convergence with respect to n in the weighted error norm for RBF–PUM with a uniform discretization, RBF–PUM with a tailored non-uniform discretization, compared with a uniform FD implementation using a penalty approach (left) and a uniform FD implementation using an operator splitting approach (right). The shape parameter ε in equation (7.3) is chosen as $\varepsilon = 2$ in the tailored case and $\varepsilon = 1.5$ in the uniform case. In both cases, errors are computed against an FD reference solution of the corresponding type with $n = 101$ discretization points per dimension.

7.3 Time comparison

To really compare the performance of different methods, we need to consider the execution times for a given problem and required accuracy. Figure 18 shows runtime comparisons for the different RBF–PUM discretizations and the two FD implementations. For the FD–penalty implementation, the MATLAB routine `ode45` is used for time-stepping, while the operator splitting approach uses the Crank–Nicholson scheme. The non-uniform RBF–PUM discretization is fastest for the range of tolerances considered here. It should be noted that the implementations used in this comparison have not been optimized for performance.

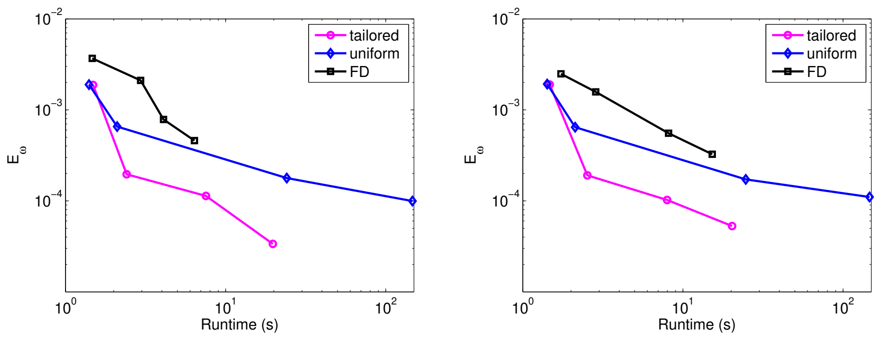

Fig. 18 Error as a function of runtime for the same experimental setup as in Figure 17 with FD–penalty as reference (left) and FD–operator splitting as reference (right).

8 Conclusions

We have implemented and tested RBF–PUM for convection-diffusion problems such as those typically arising in valuation and calibration problems in computational finance. A combination of theoretical and experimental analysis indicates that the method is stable for a wide range of problem parameters, and that we can achieve both spectral and algebraic convergence rates depending on the mode of refinement.

Different comparisons with FD and PS methods show that with sufficient smoothness of the solution to the problem, RBF–PUM is as accurate or more accurate than the PS method, but about three times slower for the problem sizes considered here. However, RBF–PUM provides a different level of flexibility, where local approximations can easily be varied both with respect to resolution and type, in arbitrary geometries.

A main advantage of RBF–PUM is that it allows for local adaptivity. Patches can be locally refined and have shapes adapted to the local solution behavior as in our option pricing example. Furthermore, the node density in each partition can be locally adjusted. To develop support for automatic adaptivity will be part of our future work, and we will also consider larger and higher-dimensional computational problems.

Acknowledgments

The authors would like to thank Victor Shcherbakov, Uppsala University who provided the FD–operator splitting implementation for the American option pricing problem.

References

1. Babuška, I., Melenk, J.M.: The partition of unity method. Internat. J. Numer. Methods Engrg. **40**(4), 727–758 (1997)

2. Ballestra, L.V., Pacelli, G.: Computing the survival probability density function in jump-diffusion models: a new approach based on radial basis functions. Eng. Anal. Bound. Elem. **35**(9), 1075–1084 (2011). DOI 10.1016/j.enganabound.2011.02.008. URL http://dx.doi.org/10.1016/j.enganabound.2011.02.008
3. Ballestra, L.V., Pacelli, G.: A radial basis function approach to compute the first-passage probability density function in two-dimensional jump-diffusion models for financial and other applications. Eng. Anal. Bound. Elem. **36**(11), 1546–1554 (2012). DOI 10.1016/j.enganabound.2012.04.011. URL http://dx.doi.org/10.1016/j.enganabound.2012.04.011
4. Ballestra, L.V., Pacelli, G.: Pricing European and American options with two stochastic factors: A highly efficient radial basis function approach. J. Econ. Dynam. Control **37**(6), 1142–1167 (2013)
5. Bates, D.: Jumps and stochastic volatility: exchange rate processes implicit in deutsche mark options. Rev. Financ. Stud. **9**(1), 69–107 (1996). DOI 10.1093/rfs/9.1.69
6. Black, F., Scholes, M.: The pricing of options and corporate liabilities. J. Polit. Econ. **81**(3), 637–654 (1973)
7. Carr, P., Geman, H., Madan, D.B., Yor, M.: Stochastic volatility for lévy processes. Math. Financ. **13**(3), 345–382 (2003)
8. Duffie, D.: Dynamic Asset Pricing Theory. Princeton University Press (1996)
9. Durham, G.B., Gallant, A.R.: Numerical techniques for maximum likelihood estimation of continuous-time diffusion processes. J. Bus. Econ. Stat. **20**(3), 297–338 (2002)
10. Fasshauer, G., Khaliq, A.Q.M., Voss, D.A.: Using meshfree approximation for multi asset American options. in: C.S. Chen (Ed.), Meshfree methods, Journal of Chinese Institute of Engineers **27**, 563–571 (2004). Special issue
11. Fasshauer, G.E.: Meshfree approximation methods with MATLAB, *Interdisciplinary Mathematical Sciences*, vol. 6. World Scientific Publishing Co. Pte. Ltd., Hackensack, NJ (2007). With 1 CD-ROM (Windows, Macintosh and UNIX)
12. Fichera, G.: Sulle equazioni differenziali lineari ellittico-paraboliche del secondo ordine. Atti Accad. Naz. Lincei. Mem. Cl. Sci. Fis. Mat. Nat. Sez. I. VIII, Ser. 5 pp. 3–30 (1956)
13. Fornberg, B., Larsson, E., Flyer, N.: Stable computations with Gaussian radial basis functions. SIAM J. Sci. Comput. **33**(2), 869–892 (2011). DOI 10.1137/09076756X. URL http://dx.doi.org/10.1137/09076756X
14. Fornberg, B., Zuev, J.: The Runge phenomenon and spatially variable shape parameters in RBF interpolation. Comput. Math. Appl. **54**(3), 379–398 (2007)
15. Franke, R., Nielson, G.: Smooth interpolation of large sets of scattered data. Internat. J. Numer. Methods Engrg. **15**(11), 1691–1704 (1980)
16. Halton, J.H.: On the efficiency of certain quasi-random sequences of points in evaluating multi-dimensional integrals. Numer. Math. **2**, 84–90 (1960)
17. Hon, Y.C., Mao, X.Z.: A radial basis function method for solving options pricing models. J. Financial Engineering **8**, 31–49 (1999)
18. In 't Hout, K.J., Foulon, S.: ADI finite difference schemes for option pricing in the Heston model with correlation. Int. J. Numer. Anal. Model. **7**(2), 303–320 (2010)
19. Ito, K., Toivanen, J.: Lagrange multiplier approach with optimized finite difference stencils for pricing American options under stochastic volatility. SIAM J. Sci. Comput. **31**(4), 2646–2664 (2009)
20. Janson, S., Tysk, J.: Feynman-Kac formulas for Black-Scholes-type operators. Bull. London Math. Soc. **38**(2), 269–282 (2006)
21. Kangro, R., Nicolaides, R.: Far field boundary conditions for Black-Scholes equations. SIAM J. Numer. Anal. **38**(4), 1357–1368 (2000). Electronic
22. Kou, S.G.: A jump-diffusion model for option pricing. Manag. Sci. **48**(8), 1086–1101 (2002)
23. Kwok, Y.K.: Mathematical models of financial derivatives, second edn. Springer Finance. Springer, Berlin (2008)
24. Larsson, E., Fornberg, B.: Theoretical and computational aspects of multivariate interpolation with increasingly flat radial basis functions. Comput. Math. Appl. **49**(1), 103–130 (2005)
25. Larsson, E., Gomes, S., Heryudono, A., Safdari-Vaighani, A.: Radial basis function methods in computational finance. In: Proc. CMMSE 2013. Almería, Spain, 12 pp. (2013)
26. Larsson, E., Heryudono, A.: A partition of unity radial basis function collocation method for partial differential equations (2013). Manuscript in preparation
27. Larsson, E., Lehto, E., Heryudono, A., Fornberg, B.: Stable computation of differentiation matrices and scattered node stencils based on Gaussian radial basis functions. SIAM J. Sci. Comput. **35**(4), A2096–A2119 (2013). DOI 10.1137/120899108. URL http://dx.doi.org/10.1137/120899108
28. McLain, D.H.: Two dimensional interpolation from random data. Comput. J. **19**(2), 178–181 (1976)
29. Merton, R.C.: Option pricing when underlying stock returns are discontinuous. J. Financ. Econ. **3**(1-2), 125–144 (1976)

30. Nielsen, B.F., Skavhaug, O., Tveito, A.: Penalty and front-fixing methods for the numerical solution of American option problems. J. Comput. Finance **5**, 69–97 (2002)
31. Nielsen, B.F., Skavhaug, O., Tveito, A.: Penalty methods for the numerical solution of American multi-asset option problems. J. Comput. Appl. Math. **222**(1), 3–16 (2008)
32. Persson, J., Sydow, L.v.: Pricing european multi-asset options using a space-time adaptive fd-method. Tech. Rep. 2003-059, Dept. of Information Technology, Uppsala Univ., Uppsala, Sweden (2003). An optional note
33. Pettersson, U., Larsson, E., Marcusson, G., Persson, J.: Improved radial basis function methods for multi-dimensional option pricing. J. Comput. Appl. Math. **222**(1), 82–93 (2008)
34. Platte, R.B.: How fast do radial basis function interpolants of analytic functions converge? IMA J. Numer. Anal. **31**, 1578–1597 (2011)
35. Reddy, S.C., Trefethen, L.N.: Stability of the method of lines. Numer. Math. **62**(2), 235–267 (1992). DOI 10.1007/BF01396228. URL http://dx.doi.org/10.1007/BF01396228
36. Rieger, C., Zwicknagl, B.: Sampling inequalities for infinitely smooth functions, with applications to interpolation and machine learning. Adv. Comput. Math. **32**(1), 103–129 (2010)
37. Rieger, C., Zwicknagl, B.: Improved exponential convergence rates by oversampling near the boundary. Constr. Approx. (2013). In press
38. Shepard, D.: A two dimensional interpolation function for irregularly space data. J. Comput. Appl. Math. pp. 517–524 (1968)
39. Tavella, D., Randall, C.: Pricing Financial Instruments. Wiley, New York (2000)
40. Trefethen, L.N., Embree, M.: Spectra and pseudospectra. Princeton University Press, Princeton, NJ (2005). The behavior of nonnormal matrices and operators
41. Wendland, H.: Piecewise polynomial, positive definite and compactly supported radial functions of minimal degree. Adv. Comput. Math. **4**(4), 389–396 (1995)
42. Wendland, H.: Fast evaluation of radial basis functions: methods based on partition of unity. In: Approximation theory, X (St. Louis, MO, 2001), Innov. Appl. Math., pp. 473–483. Vanderbilt Univ. Press, Nashville, TN (2002)
43. Wright, T.G.: EigTool. http://www.comlab.ox.ac.uk/pseudospectra/eigtool/ (2002)
44. Wu, Z., Hon, Y.C.: Convergence error estimate in solving free boundary diffusion problem by radial basis functions method. Engrg. Anal. Bound. Elem. **27**, 73–79 (2003)
45. Zvan, R., Forsyth, P.A., Vetzal, K.R.: Penalty methods for American options with stochastic volatility. J. Comput. Appl. Math. **91**(2), 199–218 (1998)