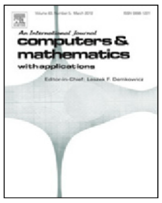
Computers and Mathematics with Applications xxx (xxxx) xxx

Contents lists available at ScienceDirect

Computers and Mathematics with Applications

journal homepage: www.elsevier.com/locate/camwa



RBF-FD solution for a financial partial-integro differential equation utilizing the generalized multiquadric function

Fazlollah Soleymani [a,b,*], Shengfeng Zhu [a]

[a] Department of Data Mathematics and Shanghai Key Laboratory of Pure Mathematics and Mathematical Practice, School of Mathematical Sciences, East China Normal University, Shanghai 200241, China
[b] Department of Mathematics, Institute for Advanced Studies in Basic Sciences (IASBS), Zanjan 45137–66731, Iran

ARTICLE INFO

Article history:
Received 19 April 2020
Received in revised form 17 October 2020
Accepted 13 November 2020
Available online xxxx

Keywords:
Generalized multiquadric
Unstructured grids
RBF-FD method
Jump–diffusion model
Stochastic volatility

ABSTRACT

This work concerns the weights of the radial basis function generated finite difference (RBF-FD) formulas for estimation of the first and second derivatives of an unknown function applying the generalized multiquadric function (GMQ). Several discussions about their error equations on structured and unstructured grids of points are worked out. Next, the formulas are applied on a new non-uniform mesh of points based on modified Legendre polynomial zeros in order to computationally solve a (1+2) dimensional partial integro-differential equation (PIDE) arising in the model of stochastic volatility with contemporaneous jumps (SVCJ). Numerical results show a fast convergence for solving this problem.

1. Introductory notes

The aim of this paper is to construct the weighting coefficients of the meshfree radial basis function (RBF) generated finite difference (FD) scheme, which are of importance in computational simulations of partial integro-differential equations (PIDEs) and partial differential equations (PDEs) [1]. Localized meshfree methods offer several advantages over traditional FD, global meshfree, finite element (FE), and spectral schemes such as being less sensitive to the structure of the domain, easy implementations for high dimensional problems, etc., see [2,3].

RBF methods are a significant part of the computational mathematics. Such schemes were first introduced in [4,5], for interpolation of scattered data by illustrating that the interpolation based on RBFs could overcome some cases at which the polynomial interpolation failed.

The meshfree RBF methods are categorized as globalized and localized meshless schemes. Another localized improved version of such schemes is the RBF-FD estimate [6], which is an attractive choice in contrast to the globalized meshfree RBF schemes by providing better conditioned and sparse discretization matrices. Indeed, as the matrix size increases, their condition number typically increases as well. The RBF-FD reduces the computational cost of applying RBF methods for practical problems in higher dimensions, see for more [7]. Unlike the traditional procedures, the RBF-FD method can deal with scattered node layouts and irregular domains [8].

It is straightforward to consider the RBF-FD method as an extension over the classical methods, see e.g., [9]. It is recalled that the global RBF scheme for surface PDEs can be considered as an extension of conventional-based (or Fourier-based) pseudo-spectral schemes to surfaces, while the RBF-FD approach can be taken into account as an extension of standard FD schemes based on polynomial estimates to surfaces, see for more, [10,11] and the references therein.

* Corresponding author.
E-mail addresses: soleymani@iasbs.ac.ir (F. Soleymani), sfzhu@math.ecnu.edu.cn (S. Zhu).

https://doi.org/10.1016/j.camwa.2020.11.010
0898-1221/© 2020 Elsevier Ltd. All rights reserved.

It is well-known that the generalized multiquadric RBF (GMQ RBF) is defined by [12]:

$$\phi(r_i) = (c^2 + r_i^2)^{\mu}, \qquad i = 1, 2, \ldots, \mathfrak{m}, \tag{1}$$

where $r_i = \|\mathbf{x} - \mathbf{x}_i\|_2$ is the distance in the Euclidean sense and the parameter of shape is c. Actually choosing μ as $1/2$ and $-1/2$, yield the multiquadric (MQ) and the inverse multiquadric (IMQ) RBFs, respectively, [13].

Noting that the nonzero parameter c is an important key in the precision of estimated solutions, for more refer to [14, Chapter 15.5]. In addition, we have $\mu \neq 3/2, 7/2, 5.89507$ in the GMQ RBF (1), because they make the denominators of weights to become zero as carried forward in (7), (20), (29) and (32). Analyzing multiquadric RBF theoretically to determine its parameter has already been investigated separately in several works such as [15,16]. Our proposed procedure in this work differs from the aforementioned papers in several respects. Firstly, we propose a method, which enjoys better convergence rate with some relatively cheaper MQ parameter tuning than the standard RBF-FD method based on the same discretization. Secondly, we focus on GMQ version. Finally, we benefit from the point that we can now avoid solving linear systems in the construction of RBF-FD weights.

The contribution and motivation of this work are briefly given as follows. In this paper, the RBF-FD weights using the GMQ RBF applying several structured and unstructured points are constructed for the first and second derivatives of a function. Here we use the GMQ basis due to its generality, i.e., for various choices of μ we can derive the existing well-known RBFs and thus a family of functions is investigated. This is quite a novel branch of research, since the application of GMQ RBF for many problems has not yet been discussed in literature. To the best of the authors' knowledge, the weighting coefficients of the GMQ RBF have not yet been treated in closed forms for the most practical cases of three- and five-node stencils (the formulas for the MQ RBF was treated [13]). The motivation of this paper also lies in solving a practical PIDE in finance, similar as the ones arising in stochastic volatility (SV) jump model. Duffie et al. [17, section 4] introduced a model with SV and jumps in volatility as well as in returns. It is in fact considered that the jumps occurred independently (denoted by SVIJ) as well as the SV with jumps contemporaneously (denoted by SVCJ) which is easier to be handled, see [18,19] for background.

Further investigation of the SVCJ model in financial engineering is an active topic of research, see [20,21] for some background on this topic. Option pricing under the SV with contemporaneous jumps for the function $p = p(s, v, \tau)$ can be modeled in the form of a 1 + 2 PIDE as follows [22]:

$$\begin{aligned} \frac{\partial p}{\partial \tau} &= \frac{1}{2} v s^2 \frac{\partial^2 p}{\partial s^2} + \frac{1}{2}\sigma^2 v \frac{\partial^2 p}{\partial v^2} + \rho\sigma v s \frac{\partial^2 p}{\partial s \partial v} + (r - q - \lambda\xi) s \frac{\partial p}{\partial s} + \kappa(\theta - v)\frac{\partial p}{\partial v} - (r + \lambda)p \\ &\quad + \lambda \int_0^{\infty}\int_0^{\infty} p(sz^s, v + z^v, \tau) g(z^s, z^v)\, dz^v\, dz^s, \quad \text{on} \quad \Omega \subset \mathbb{R}^2, \quad s, v > 0, \end{aligned} \tag{2}$$

where $\tau = T - t$, r is the risk-free rate of interest, q is a dividend yield, ξ is the mean jump, κ is the reversion rate of the variance, while θ and σ are the mean level and volatility constants, respectively. Here ξ is defined by: $\xi = \frac{e^{\left(\gamma + \frac{1}{2}\delta^2\right)}}{1 - \nu\rho_J} - 1$, where ρ_J is the correlation between jumps in returns and volatility. In this model, the jump size distribution in variance is considered to be exponentially distributed with mean ν. The 2D probability density function (PDF) g with log-normal distribution is defined as follows [23]:

$$g(z^s, z^v) = \frac{1}{\sqrt{2\pi} z^s \delta \nu} e^{\left(-\frac{z^v}{\nu} - \frac{(\ln(z^s) - \gamma - \rho_J z^v)^2}{2\delta^2}\right)}. \tag{3}$$

The other notations are ρ as the correlation between the Brownian motions, λ as the rate of jump arrival, δ^2 as the jump size log-variance and γ as the jump size log-mean.

Here, the initial criterion for the call and put options [11] is defined by the following expressions $p(s, v, 0) = \phi(s) = (s - E)^+$, and $p(s, v, 0) = \phi(s) = (E - s)^+$, respectively, where E is the strike price.

After this introduction regarding the motivation and the importance of computing and presenting the weights of the GMQ RBF-FD scheme, the remaining sections are structured as follows. Section 2 derives the weights of the GMQ RBF-FD method in the presence of a stencil having three nodes. This is the most useful case once the RBF-FD method is applied for solving high dimensional time-dependent P(I)DE problems using method of lines (see for a background [24, chapter 3.8]), since the application of three nodes would result in matrices having three bands (in the 1D case). Both structured and unstructured nodes are discussed. Section 3 tackles the weights for a stencil including five structured points. Section 4 provides an application of these new RBF-FD formulas for solving the challenging PIDE (2) for vanilla option pricing. Noting that providing a general error analysis of the discretization error for the whole procedure or for the general non-tensor grid is almost impossible here since it relies on many aspects and parameters, thus we focus on a parameter study for solving (2) by also contributing some points in this respect. We also contribute by proposing a new non-uniform mesh based on Legendre zeros with an emphasis on the hot zone, i.e., the area at which (financially speaking) the initial condition is non-smooth and the spontaneous variance is zero. It is necessary to compare the results of the new scheme using the RBF-FD formulas alongside the new non-uniform mesh of points with the existing solvers for tackling (2). Hence, Section 5 gives such a comparison and shows stable and smooth numerical solution for solving this (2). A brief conclusion follows in Section 6.

2. Weights of the GMQ RBF-FD formulas

Assume that there is an unstructured stencil containing 3 grid points. Here we first obtain the weighting coefficients for such a case and then the weights and error equations for its corresponding structured case can be derived (in a simplified manner).

To contribute and find the weighting coefficients Ξ_i of the RBF-FD formulas, we consider three unstructured points as follows:

$$\{x_i - h, x_i, x_i + \omega h\}, \quad \omega > 0, h > 0, \tag{4}$$

and write the approximation formula by

$$f'(x_i) \simeq \sum_{j=i-1}^{i+1} \Xi_j f(x_j) = \hat{f}'(x_i). \tag{5}$$

Now we provide the following lemma to summarize the calculation of and accuracy order of the weighting coefficients.

Lemma 2.1. *The equation of error for approximating the 1st derivative of a given smooth function considering the GMQ RBF-FD formulation (5) is expressed by:*

$$\varepsilon(x_i) = \frac{1}{6}\omega\left(f^{(3)}(x_i) - \frac{6(\mu-1)f'(x_i)}{c^2}\right)h^2 + \mathcal{O}\left(h^3\right), \tag{6}$$

where $\varepsilon(x_i) = \hat{f}'(x_i) - f'(x_i)$ *and*

$$\begin{aligned}
\Xi_{i-1} &= \frac{\omega\left(c^2(9-6\mu) - h^2(\mu-1)(4(\mu-5)\omega - 10\mu + 29)\right)}{3c^2h(2\mu-3)(\omega+1)},\\
\Xi_i &= \frac{(\omega-1)\left(c^2(6\mu-9) + 4h^2(\mu-5)(\mu-1)\omega\right)}{3c^2h(2\mu-3)\omega},\\
\Xi_{i+1} &= \frac{c^2(6\mu-9) - h^2(\mu-1)\omega(2\mu(5\omega-2) - 29\omega + 20)}{3c^2h(2\mu-3)\omega(\omega+1)}.
\end{aligned} \tag{7}$$

Proof. Substituting the function f in (5) by the GMQ RBFs (1), centered at $r = -h$, in the derivative of (1), i.e., $\phi'(r) = 2\mu r\left(c^2 + r^2\right)^{\mu-1}$, whereas r in the right hand side of (5) sets to the three values of $r = -2h$, $r = -h$, $r = -h + \omega h$, yields the following equation:

$$-2h\mu\left(c^2 + h^2\right)^{\mu-1} = \Xi_{i+1}\left(c^2 + h^2(\omega-1)^2\right)^{\mu} + \Xi_i\left(c^2 + h^2\right)^{\mu} + \Xi_{i-1}\left(c^2 + 4h^2\right)^{\mu}. \tag{8}$$

Noticing that based on the methodology of RBF-FD method, the values for r in the left hand side of (5) are chosen based on the distance (increment) of the three non-equidistant meshes in (4). Now by centering at $r = 0$, in the derivative of (1), whereas this time, r in the right hand side of (5) sets to the three values of $r = -h$, $r = 0$, $r = +\omega h$, leads to the following equation:

$$\Xi_i c^{2\mu} + \Xi_{i+1}\left(c^2 + h^2\omega^2\right)^{\mu} + \Xi_{i-1}\left(c^2 + h^2\right)^{\mu} = 0. \tag{9}$$

And finally at $r = \omega h$, in the derivative of (1), by considering $r = \omega h - h$, $r = \omega h$, $r = 2\omega h$ in the right hand side of (5), leads to the following equation:

$$2h\mu\omega\left(c^2 + h^2\omega^2\right)^{\mu-1} = \Xi_i\left(c^2 + h^2\omega^2\right)^{\mu} + \Xi_{i+1}\left(c^2 + 4h^2\omega^2\right)^{\mu} + \Xi_{i-1}\left(c^2 + h^2(\omega-1)^2\right)^{\mu}. \tag{10}$$

The obtained equations can be written as the following matrix notation:

$$\begin{pmatrix} -\left(c^2+4h^2\right)^{\mu} & -\left(c^2+h^2\right)^{\mu} & -\left(c^2+h^2(\omega-1)^2\right)^{\mu} \\ \left(c^2+h^2\right)^{\mu} & c^{2\mu} & \left(c^2+h^2\omega^2\right)^{\mu} \\ -\left(c^2+h^2(\omega-1)^2\right)^{\mu} & -\left(c^2+h^2\omega^2\right)^{\mu} & -\left(c^2+4h^2\omega^2\right)^{\mu} \end{pmatrix} \begin{pmatrix} \Xi_{i-1} \\ \Xi_i \\ \Xi_{i+1} \end{pmatrix} = -b, \tag{11}$$

where

$$b = \left(-2h\left(c^2+h^2\right)^{\mu-1}\mu, 0, 2h\mu\omega\left(c^2+h^2\omega^2\right)^{\mu-1}\right)^{*}. \tag{12}$$

Throughout the work $*$ denotes the transpose symbol. It is remarked that the diagonal entries of the coefficient matrix are not fixed since the mesh is non-uniform. In addition, to guarantee the existence of the weights, no vanishing eigenvalues for the coefficient matrix must exist which equals to a non-zero determinant as follows:

$$\mathrm{Det}(A) = o_1 - (c^2+h^2)^{2\mu}(c^2+4h^2\omega^2)^{\mu} - (c(c^2+h^2(\omega-1)^2))^{2\mu} + c^{2\mu}((c^2+4h^2)(c^2+4h^2\omega^2))^{\mu} \neq 0, \tag{13}$$

where $o_1 = (c^2+4h^2)^{\mu}(-(c^2+h^2\omega^2)^{2\mu}) + 2((c^2+h^2)(c^2+h^2(\omega-1)^2)(c^2+h^2\omega^2))^{\mu}$ and is true when $c > h$, $h, \omega, \mu > 0$.

Considering the following parameters $a_1 = 2h\mu(-c^2(\omega+1)(c^2+h^2)^{\mu}(c^2+h^2\omega^2)^{2\mu} - h^2\omega(\omega+1)(c^2+h^2)^{\mu}(c^2+h^2\omega^2)^{2\mu} + c^{2\mu+2}(\omega((c^2+h^2(\omega-1)^2)(c^2+h^2\omega^2))^{\mu} + ((c^2+h^2)(c^2+4h^2\omega^2))^{\mu}) + h^2\omega c^{2\mu}(((c^2+h^2(\omega-1)^2)(c^2+h^2\omega^2))^{\mu} + \omega((c^2+h^2)(c^2+4h^2\omega^2))^{\mu}))$, $a_2 = (c^2+h^2)(c^2+h^2\omega^2)((c^2+4h^2)^{\mu}(c^2+h^2\omega^2)^{2\mu} - 2((c^2+h^2)(c^2+h^2(\omega-1)^2)(c^2+h^2\omega^2))^{\mu} + (c^2+h^2)^{2\mu}(c^2+4h^2\omega^2)^{\mu} + (c(c^2+h^2(\omega-1)^2))^{2\mu} - c^{2\mu}((c^2+4h^2)(c^2+4h^2\omega^2))^{\mu})$, $b_1 = 2h\mu(c^2\omega(c^2+4h^2)^{\mu}(c^2+h^2\omega^2)^{2\mu} + h^2\omega(c^2+4h^2)^{\mu}(c^2+h^2\omega^2)^{2\mu} + c^2((c^2+h^2)(c^2+h^2(\omega-1)^2)(c^2+h^2\omega^2))^{\mu} + h^2\omega^2((c^2+h^2)(c^2+h^2(\omega-1)^2)(c^2+h^2\omega^2))^{\mu} - c^2\omega((c^2+h^2)(c^2+h^2(\omega-1)^2)(c^2+h^2\omega^2))^{\mu} - h^2\omega((c^2+h^2)(c^2+h^2(\omega-1)^2)(c^2+h^2\omega^2))^{\mu} - (c^2+h^2)^{2\mu}(c^2+h^2\omega^2)(c^2+4h^2\omega^2)^{\mu})$, $b_2 = (c^2+h^2)(c^2+h^2\omega^2)((c^2+4h^2)^{\mu}(c^2+h^2\omega^2)^{2\mu} - 2((c^2+h^2)(c^2+h^2(\omega-1)^2)(c^2+h^2\omega^2))^{\mu} + (c^2+h^2)^{2\mu}(c^2+4h^2\omega^2)^{\mu} + (c(c^2+h^2(\omega-1)^2))^{2\mu} - c^{2\mu}((c^2+4h^2)(c^2+4h^2\omega^2))^{\mu})$, $c_1 = 2h\mu(c^2(\omega+1)(c^2+h^2)^{2\mu}(c^2+h^2\omega^2)^{\mu} + h^2\omega(\omega+1)(c^2+h^2)^{2\mu}(c^2+h^2\omega^2)^{\mu} - h^2\omega c^{2\mu}(((c^2+4h^2)(c^2+h^2\omega^2))^{\mu} + \omega((c^2+h^2)(c^2+h^2(\omega-1)^2))^{\mu}) - c^{2\mu+2}(\omega((c^2+4h^2)(c^2+h^2\omega^2))^{\mu} + ((c^2+h^2)(c^2+h^2(\omega-1)^2))^{\mu}))$, $c_2 = (c^2+h^2)(c^2+h^2\omega^2)((c^2+4h^2)^{\mu}(c^2+h^2\omega^2)^{2\mu} - 2((c^2+h^2)(c^2+h^2(\omega-1)^2)(c^2+h^2\omega^2))^{\mu} + (c^2+h^2)^{2\mu}(c^2+4h^2\omega^2)^{\mu} + (c(c^2+h^2(\omega-1)^2))^{2\mu} - c^{2\mu}((c^2+4h^2)(c^2+4h^2\omega^2))^{\mu})$, the solution to the linear system of Eqs. (11) can be written as follows: $(\Xi_{i-1}, \Xi_i, \Xi_{i+1}) = (a_1/a_2, b_1/b_2, c_1/c_2)$. However, the obtained solution is bulky and may not be useful in practice! Because of this, we impose a first order Taylor expansion under the condition that $c > h$, to obtain more useful closed-form formulas for the weights in (7).

Now the obtained weights (7) are put back to (5) to have

$$\begin{aligned} f'(x_i) =& \frac{(\omega-1)f(x_i)\left(c^2(6\mu-9)+4h^2(\mu-5)(\mu-1)\omega\right)}{3c^2h(2\mu-3)\omega}, \\ &+ \frac{\omega f(x_i-h)\left(c^2(9-6\mu)-h^2(\mu-1)(4(\mu-5)\omega-10\mu+29)\right)}{3c^2h(2\mu-3)(\omega+1)}, \\ &+ \frac{\left(c^2(6\mu-9)-h^2(\mu-1)\omega(2\mu(5\omega-2)-29\omega+20)\right)f(x_i+\omega h)}{3c^2h(2\mu-3)\omega(\omega+1)}, \end{aligned} \tag{14}$$

and then to obtain the final error equation of the RBF-FD approximation. Here the Taylor expansion around $h = 0$ up to the second order is written on (14) to obtain the following error equation

$$\varepsilon(x_i) = \frac{h^2\omega\left(c^2f^{(3)}(x_i)-6\mu f'(x_i)+6f'(x_i)\right)}{6c^2} + \mathcal{O}\left(h^3\right). \tag{15}$$

After some algebraic simplification (15) yields the final error Eq. (6), which proves a quadratical convergence order for our RBF-FD approximation. The proof is complete now. $\square$

Selecting $\mu = 1$ in (6) could yield formulas which their asymptotic error coefficient is independent of $f'(x_i)$. The weights of the GMQ RBF-FD method in (7) were derived in a stable way regardless the ill-conditioning of the coefficient matrix over a stencil having three unstructured nodes.

Now one can find the weights for structured nodes, i.e., the following equidistant grid of nodes:

$$\{x_i-h, x_i, x_i+h\}, \quad h > 0, \tag{16}$$

at which the value of ω is one. Taking this into account, the weights can be given for three equidistant points as follows:

$$\Xi_{i-1} = -\frac{c^2-h^2\mu+h^2}{2c^2h}, \quad \Xi_i = 0, \quad \Xi_{i+1} = -\Xi_{i-1}, \tag{17}$$

while the error equation can be simplified to: $\varepsilon(x_i) = \frac{1}{6}\left(f^{(3)}(x_i) - \frac{6(\mu-1)f'(x_i)}{c^2}\right)h^2 + \mathcal{O}\left(h^3\right)$.

To compute the weights for the 2nd function's derivative, we can write the RBF-FD approximation in a similar way as follows:

$$f''(x_i) \simeq \sum_{j=i-1}^{i+1} \Theta_j f(x_j) = \hat{f}''(x_i). \tag{18}$$

The order of accuracy for this case is given in the following lemma.

Lemma 2.2. *The equation of error for the 2nd derivative of a smooth function considering the GMQ RBF-FD formula (18) is expressed by:*

$$\hat{\varepsilon}(x_i) = \frac{(\omega-1)\left(c^2f^{(3)}(x_i)-6(\mu-1)f'(x_i)\right)}{3c^2}h + \mathcal{O}\left(h^2\right), \tag{19}$$

where $\hat{\varepsilon}(x_i) = \hat{f}''(x_i) - f''(x_i)$ *and*

$$\begin{aligned} \Theta_{i-1} &= \frac{2\left(c^2(6\mu-9)-h^2(\mu-1)\left(4(\mu-5)\omega^2+(34-8\mu)\omega+10\mu-29\right)\right)}{3c^2h^2(2\mu-3)(\omega+1)}, \\ \Theta_i &= \frac{2\left(c^2(9-6\mu)+h^2(\mu-1)\left(4(\mu-5)\omega^2+(25-2\mu)\omega+4(\mu-5)\right)\right)}{3c^2h^2(2\mu-3)\omega}, \end{aligned} \tag{20}$$

$$\Theta_{i+1} = \frac{2\left(c^2(6\mu-9)-h^2(\mu-1)(2\mu(\omega(5\omega-4)+2)+\omega(34-29\omega)-20)\right)}{3c^2h^2(2\mu-3)\omega(\omega+1)}.$$

Proof. The proof is similar to the steps taken for Lemma 2.1. We briefly state this as follows. To obtain the weighting coefficients and the accuracy order, we first obtain a set of three equations as comes next:

$$\begin{aligned}
&2\mu\left(c^2+h^2\right)^{\mu-2}\left(c^2+h^2(2\mu-1)\right) = \Theta_{i+1}\left(c^2+h^2(\omega-1)^2\right)^{\mu}+\Theta_i\left(c^2+h^2\right)^{\mu}+\Theta_{i-1}\left(c^2+4h^2\right)^{\mu},\\
&2\mu c^{2\mu-2} = \Theta_i c^{2\mu}+\Theta_{i+1}\left(c^2+h^2\omega^2\right)^{\mu}+\Theta_{i-1}\left(c^2+h^2\right)^{\mu},\\
&2\mu\left(c^2+h^2\omega^2\right)^{\mu-2}\left(c^2+h^2(2\mu-1)\omega^2\right) = \Theta_i\left(c^2+h^2\omega^2\right)^{\mu}+\Theta_{i+1}\left(c^2+4h^2\omega^2\right)^{\mu}+\Theta_{i-1}\left(c^2+h^2(\omega-1)^2\right)^{\mu},
\end{aligned} \tag{21}$$

which can be written as follows:

$$\begin{pmatrix} -\left(c^2+4h^2\right)^{\mu} & -\left(c^2+h^2\right)^{\mu} & -\left(c^2+h^2(\omega-1)^2\right)^{\mu} \\ -\left(c^2+h^2\right)^{\mu} & -c^{2\mu} & -\left(c^2+h^2\omega^2\right)^{\mu} \\ -\left(c^2+h^2(\omega-1)^2\right)^{\mu} & -\left(c^2+h^2\omega^2\right)^{\mu} & -\left(c^2+4h^2\omega^2\right)^{\mu} \end{pmatrix}\begin{pmatrix}\Theta_{i-1}\\ \Theta_i\\ \Theta_{i+1}\end{pmatrix} = -\mathfrak{b}, \tag{22}$$

where

$$\mathfrak{b} = \left(2\left(c^2+h^2\right)^{\mu-2}\mu\left(c^2+h^2(2\mu-1)\right), 2c^{2\mu-2}\mu, 2\mu\left(c^2+h^2\omega^2\right)^{\mu-2}\left(c^2+h^2(2\mu-1)\omega^2\right)\right)^{*}. \tag{23}$$

By solving (22) symbolically when $c > h$, we obtain (24). Taking into account (18)–(20), we obtain the error Eq. (19). □

The accuracy of the estimates for the 2nd derivative is of first-order unlike the approximations for the 1st derivatives, which are of quadratic convergence for both equidistant and non-equidistant grid of three nodes.

If we consider the equidistant grid of three nodes (16), viz, with $\omega = 1$, then the simplified weights are expressed by

$$\Theta_{i-1} = \frac{-\mu+\frac{1}{2\mu-3}+2}{c^2}+\frac{1}{h^2},\quad \Theta_i = \frac{c^2(6-4\mu)+2h^2(\mu-1)(2\mu-5)}{c^2h^2(2\mu-3)},\quad \Theta_{i+1}=\Theta_{i-1}, \tag{24}$$

with the associated error equation:

$$\hat{\varepsilon}(x_i) = \left(\frac{((7-2\mu)\mu-5)f''(x_i)}{c^2(2\mu-3)}+\frac{1}{12}f^{(4)}(x_i)\right)h^2+\mathcal{O}\left(h^3\right). \tag{25}$$

The important observation in (25) is that using three equidistant nodes would yield approximations which are of quadratic convergence for estimating the 2nd derivative.

The weights computed and presented in this section are the most applicable weights in practice. This means that they lead to tri-diagonal differentiation matrices when be applied for solving 1D (ordinary, partial and stochastic) differential equations [11].

For higher dimensional problems, weights on tensor grids can be calculated based on the 1D case, and then be extended for higher dimensional case via Kronecker products. This will be discussed later in this paper. But there is no straightforward extension for non-tensor arbitrarily distributed meshes, if we wish to derive and code the weighting coefficients. For such a general case, linear systems must be tackled for each set of points considered in the working domain.

3. Weights for 5-node uniform stencil

The purpose of this section is to derive the weighting coefficients of the GMQ RBF-FD method for a stencil including 5 nodes. We consider a stencil having five equidistant nodes as follows:

$$\{x_i-2h, x_i-h, x_i, x_i+h, x_i+2h\},\quad h>0. \tag{26}$$

To compute the weights of the 1st derivative, we proceed by considering the following RBF-FD formula

$$f'(x_i) \simeq \sum_{j=i-2}^{i+2}\Upsilon_j f(x_j) = \hat{f}'(x_i). \tag{27}$$

Here Υ_j are the weights in this case. The procedure is now summarized in the following lemma.

Lemma 3.1. *The equation of error for estimating the 1st derivative of a given smooth function applying the GMQ RBF-FD formulation (27) is given by:*

$$\varepsilon(x_i) = \left(\frac{2(\mu-2)(2\mu-9)f^{(3)}(x_i)}{3c^2(2\mu-7)}-\frac{1}{30}f^{(5)}(x_i)\right)h^4+\mathcal{O}\left(h^5\right), \tag{28}$$

while the weights are

$$\Upsilon_{i-2} = \frac{c^2(2\mu-7) - 4h^2(\mu-2)(2\mu-9)}{12c^2h(2\mu-7)},\quad \Upsilon_{i-1} = \frac{c^2(14-4\mu) + 2h^2(\mu-2)(2\mu-9)}{3c^2h(2\mu-7)},\quad \Upsilon_i = 0, \tag{29}$$

with $\Upsilon_{i+1} = -\Upsilon_{i-1}$, $\Upsilon_{i+2} = -\Upsilon_{i-2}$.

The proof is furnished in Appendix A.

To tackle the 2nd derivative using the stencil (26), we may write

$$f''(x_i) \simeq \sum_{j=i-2}^{i+2} \Phi_j f(x_j) = \hat{f}''(x_i). \tag{30}$$

We now give the following lemma.

Lemma 3.2. *The error equation for approximating the 2nd derivative of a given smooth function applying the GMQ RBF-FD formulation (30) is given by:*

$$\begin{aligned}\hat{\varepsilon}(x_i) = \Big(&\big(30\mu - 2\left(-6(\mu-1)\left(2\mu\left(4\mu^2 - 54\mu + 211\right) - 525\right)f''(x_i)\right.\\ &+c^2\left(8\mu^3 - 92\mu^2 + 294\mu - 273\right)f^{(4)}(x_i)\big)\\ &+c^4\left(165 - 2\mu\left(4\mu^2 - 38\mu + 99\right)\right)f^{(6)}(x_i)\big)\Big)h^4/\mathrm{d} + \mathcal{O}\left(h^5\right),\end{aligned} \tag{31}$$

where $\mathrm{d} = \left(90c^4\left(2\mu\left(4\mu^2 - 38\mu + 99\right) - 165\right)\right)$ *and*

$$\begin{aligned}\Phi_{i-2} &= \frac{1}{36}\left(-\frac{3}{h^2} + \frac{12(\mu-2)\left(8\mu^3 - 92\mu^2 + 294\mu - 273\right)}{c^2\left(2\mu\left(4\mu^2 - 38\mu + 99\right) - 165\right)} - \eta_1\right)\\ \Phi_{i-1} &= \frac{2}{9}\left(\frac{6}{h^2} - \frac{6(\mu-2)\left(8\mu^3 - 92\mu^2 + 294\mu - 273\right)}{c^2\left(2\mu\left(4\mu^2 - 38\mu + 99\right) - 165\right)} - \eta_2\right),\\ \Phi_i &= \frac{1}{6c^4h^2\left(165 - 2\mu\left(4\mu^2 - 38\mu + 99\right)\right)^2}(\zeta + \vartheta),\quad \Phi_{i+1} = \Phi_{i-1},\quad \Phi_{i+2} = \Phi_{i-2},\end{aligned} \tag{32}$$

with $\eta_1 = \dfrac{a_1}{c^4\left(165 - 2\mu\left(4\mu^2 - 38\mu + 99\right)\right)^2}$, $\eta_2 = \dfrac{a_2}{c^4\left(165 - 2\mu\left(4\mu^2 - 38\mu + 99\right)\right)^2}$, *and*

$$\begin{aligned}\zeta &= 12c^2h^2(\mu-2)(8\mu^3 - 92\mu^2 + 294\mu - 273)(2\mu(4\mu^2 - 38\mu + 99) - 165) - 15c^4(165 - 2\mu(4\mu^2 - 38\mu + 99))^2,\\ \vartheta &= 4h^4(\mu-2)(2\mu(2\mu(2\mu(2\mu(2\mu(2\mu(2\mu-61)+1427)-17237)+116451)-443571)+897345)-748035),\\ a_1 &= 4h^2(\mu-2)(4\mu(\mu(4\mu(\mu(4\mu(\mu(4\mu-83)+659)-9817)+16869)-28713)-58365)+228285),\\ a_2 &= h^2(\mu-2)(\mu(4\mu(\mu(4\mu(\mu(4(\mu-50)\mu+2963)-20947)+315615)-651000)+2808765)-1236195).\end{aligned} \tag{33}$$

The proof is given in Appendix A.

The proposed weights for the 1st and 2nd derivatives of a sufficiently smooth function yield RBF-FD approximations with fourth order of convergence. Clearly, once these weights are applied to fill the differentiation matrices for the first and second derivatives of a function in solving PDE problem, the matrices would be penta-diagonal (in the 1D case).

It is requisite to recall that authors in [25] found that $\mu = 1.99$ is a good choice for a very *specific* problem and not in general. Kansa in [26] recommended that the GMQ has good numerical outputs by choosing $\mu = 5/2$ for interpolation problems. However in this paper, we choose different values for this parameter since we are dealing with a PIDE problem in high dimension rather than the specific problems or the interpolation problems and see how the approximations end in good convergence behavior.

4. A shifted Legendre mesh generation and solving the financial SVCJ PIDE

For solving the option pricing problem (2), let us first propose a new mesh of points. We know that the spatial domain should be written as: $\Omega = [0, s_{\max}] \times [0, v_{\max}]$, where $s_{\max}$, and $v_{\max}$ are positive fixed values and large enough for minimizing the error of the truncation of the domain, [23].

A denser grid in the hotzone, viz, the area at which the initial condition is non-differentiable and the variance rate (spontaneous variance v) is zero, could circumvent the issues occurring in pricing (2). Noting that increasing the stencil size in general can be preferable for some practical problems but the main problem that happens for higher dimensional PIDE based problems is the curse of dimensionality that occurs and yields in requirement of large memory space and time for time-marching. Hence, the presented mesh must be coarse enough to decrease the elapsed computing time, but as fine as possible to reach a good accuracy at the hotzone. In what follows, such a new grid is constructed by a modification on the Legendre zeros coming out of the Legendre orthogonal polynomials.

Although there are several zooming functions mainly based on the hyperbolic functions to produce non-uniform meshes for such PIDE problems, see e.g., [27], here the motivation behind constructing a non-uniform mesh is not only limited to its novelty but also to choose Legendre polynomial because of its least (minimum) error property and independency to any weight functions, which make the construction process easier with better error properties in terms of interpolation.

The polynomial of Legendre $L_{\mathtt{n}}(x)$ of the $\mathtt{n}$th-degree can be expressed using the Rodrigues' formula [28, chapter 8]:

$$L_{\mathtt{n}}(x) = \frac{1}{2^{\mathtt{n}}\mathtt{n}!}\frac{\mathrm{d}^{\mathtt{n}}}{\mathrm{d}x^{\mathtt{n}}}\left(x^2-1\right)^{\mathtt{n}}. \tag{34}$$

Here to find the zeros we employ the following built-in function of Mathematica (the procedure in other programming packages is similar):

```
digits = 30;
P[n_] := LegendreP[n, x];
setroot[n_] := N[x /. NSolve[P[n] == 0, x, WorkingPrecision -> digits]]
```

In order to produce the computational nodes along s by concentrating the points around the strike price, the procedure of producing a non-uniform modified Legendre nodes is presented in Algorithm 1. One point of this adaptation is that the strike price E is always a point of the grid while in standard uniform gridding, it relies on the number of points and the domain boundaries.

In order to produce the computational nodes along v by an emphasis when $v = 0$, the non-uniform modified Legendre nodes are proposed in Algorithm 2 (also see Appendix B).

```
while not reaching the required mesh points do
    1. Choose an even $m \gg 4$ for the number of nodes, and obtain the $m-3$ Legendre zeros.
    2. Since the zeros $l_i$, $1 \le i \le m$ of orthogonal Legendre polynomials have concentration around the boundaries
       to minimize the error of interpolation and approximations as much as possible, take all the positive zeros.
       Put these roots into a new variable $s_1$. The concentration of the points is on the right boundary for $s_1$.
    3. Compute $1-s_1$ and sort the nodes. Put these points into a new variable $s_2$. $1-s_1$ means that the array of
       roots $s_1$ is subtracted from a vector of ones. The concentration of the points is on the left boundary for $s_2$.
    4. To stretch the nodes into our main domain $[0, s_{\max}]$, we first join 0 and 1 as the first and last element to $s_1$,
       while multiplying the whole set of nodes into the strike price in order to shift the nodes to the area $[0, E]$
       with a focus around $E$ from the left side and put these points into a new variable $s_3$.
    5. Second, this time join 0 and 1 as the first and last element to $s_2$ and shift these nodes to the interval
       $[E+\alpha, s_{\max}]$, where $\alpha$ is a small free positive constant. Gather these points into a new variable $s_4$. The
       concentration of the points is around $E+\alpha$ from the right side.
    6. The final set of non-uniform nodes for $s$ is given by joining $s_3$ and $s_4$.
end
```

Algorithm 1: How to produce a non-uniform modified Legendre nodes along s for the working domain.

```
while not reaching the required mesh points do
    1. Choose $n \gg 4$ for the number of points, and obtain the $2(n-2)$ Legendre zeros $l_j$, $1 \le j \le n$ employing the
       above piece of code.
    2. With a same reason as above, take all the positive zeros and put these points into a new variable $v_1$.
    3. Compute $1-v_1$ and sort the nodes. Put these points into a new variable $v_2$.
    4. To stretch the nodes into our main domain $[v_{\min}, v_{\max}]$, join 0 and 1 as the first and last element to $v_2$ and
       then extend to the domain $[v_{\min}, v_{\max}]$.
end
```

Algorithm 2: How to produce a non-uniform modified Legendre nodes along v for the working domain.

Remark 4.1. Even though this work touches on many aspects and parameters of the RBF-FD and mesh generation issues, its main focus and conclusions stay on providing an efficient procedure to solve the financial PIDE problem (2). For this focus, after deriving the generalized weights in Sections 2–3, we have designed shifted Legendre meshes in Algorithms 1–2 to have more concentration of discretization nodes on the hot area. Then, the solving procedure is built up.

The proposed procedure of mesh generation gives a set of non-uniform mesh of nodes along v with an emphasis around 0. Further presentation of the grids produced here for (2) is given in Appendix B.

Let us now, build the derivative matrix corresponding to the state variable s as follows:

$$D_s = \left(\Xi_{i,j}\right)_{m\times m} = \begin{cases} \Xi_{i,j} \text{ from (7)} & i=j \text{ or } |i-j|=1, \\ 0 & \text{otherwise,} \end{cases} \tag{35}$$

and

$$D_{ss} = \left(\Theta_{i,j}\right)_{m\times m} = \begin{cases} \Theta_{i,j} \text{ from (20)} & i=j \text{ or } |i-j|=1, \\ 0 & \text{otherwise.} \end{cases} \tag{36}$$

Note that for the points on the boundaries, we only apply two points in order to keep the differentiation matrices tri-diagonal. See Appendix C for more details. The derivative matrices (35)–(36) contain the GMQ RBF-FD weights of the 1st and 2nd derivatives along s. For the other spatial variable, i.e., the variance rate v, it can be obtained in a similar way.

The easiest way to solve a problem on a tensor product (spectral) grid is to use tensor products in linear algebra, also known as Kronecker products, [29, chapter 3.5].

By considering the Kronecker product (denoted by $\otimes$, see [30, Chapter 16] for more details) of the matrices of differentiation into suitable matrices, one can find a matrix associated with the final spatially discretization problem. In fact, our problem in two spatial dimensions could be formulated in terms of the Kronecker products, [31].

Again we recall that for the 2D case it is not necessary to apply the following two-dimensional GMQ RBF formula: $\phi(x_i, y_j) = \left(c^2 + (x-x_i)^2 + (y-y_i)^2\right)^{\mu}$, and calculate the appropriate weighting coefficients, because we work on tensor product meshes. We consider the one-dimensional weights in each dimension and finally build the matrices associated to the terms in (2). Hence, we will have two different c, viz, one along the first state variable called $c_{\text{along } s}$ and the other as $c_{\text{along } v}$. However, one may be interested in the most general case of unstructured non-tensor product grids. For such a case, the weights of the RBF-FD method could be computed per nodes. In fact, finding very elegant formulations for non-tensor product grids is an intensive action and in practice linear systems must be solved for each set of interior points of the unstructured non-tensor grid in higher dimensions.

The principle for the Kronecker product of the differentiation matrix associated to the mixed derivative term is described as follows:

$$\frac{\partial^2 f}{\partial x\,\partial y} = \frac{\partial}{\partial x}\left(\frac{\partial f}{\partial y}\right) = (f_y)_x = f_{yx} = \partial_{xy} f = \partial_x \partial_y f, \tag{37}$$

where $f = f(x, y)$.

On the other hand, using [32, Corollary 2] and (35)–(36), we have

$$D_{sv} = (D_s \otimes I_{n\times n})\,(I_{m\times m} \otimes D_v)\,. \tag{38}$$

To be more precise, the matrix of weighting coefficients corresponding to the cross derivative term $\frac{\partial^2}{\partial s \partial v}$ in (2) can be deduced using (38) as follows:

$$(D_{sv})_{N\times N} = (D_s)_{m\times m} \otimes (D_v)_{n\times n}\,, \tag{39}$$

where $N = m \times n$. Recalling that the eigenvalues of $D_s \otimes D_v$ are the products of all eigenvalues of D_s with all eigenvalues of D_v [33]. A deep discussion showing how to construct the differentiation matrices for higher dimensional problems in solving PDEs using tensor product grid was introduced in [34].

Finally, taking the weights arising from the reaction, diffusion and convection terms into account, a system matrix A can be obtained.

Now the discretization of the operator of the integral presenting in (2) must be done as follows:

$$\int_0^{\infty}\int_0^{\infty} p(sz^s, v+z^v, \tau) g(z^s, z^v)\, dz^v\, dz^s. \tag{40}$$

To do this challenging work, it is necessary to first apply the transformation below: $z_1 = sz^s$, $z_2 = v + z^v$. Thus, it is possible to write at the computational node (s_i, v_j):

$$\int_0^{s_{\max}}\int_{v_j}^{v_{\max}} \frac{1}{s_i} p\left(z_1, z_2, \tau\right) g\left(\frac{z_1}{s_i}, z_2 - v_j\right)\, dz_2\, dz_1 = \sum_{\iota=1}^{\iota=m-1}\sum_{\varpi=j}^{\varpi=n-1} \mathcal{M}_{\iota,\varpi}, \tag{41}$$

where

$$\mathcal{M}_{\iota,\varpi} = \int_{s_\iota}^{s_{\iota+1}}\int_{v_\varpi}^{v_{\varpi+1}} \frac{1}{s_i} p\left(z_1, z_2, \tau\right) g\left(\frac{z_1}{s_i}, z_2 - v_j\right) dz_2 dz_1. \tag{42}$$

Here it is remarked that for a call-type option after the truncation, one should also consider the integral from $s_{\max}$ to ∞, which is finite and must be approximated by taking into account the behavior of the solution as s tends to infinity.

However, this value is negligible if $s_{\max}$ is quite large. Hence, by choosing a large value for $s_{\max}$ we can ignore this part in our calculations.

Now, we consider p as a fixed function on each rectangular/cell as follows:

$$p(z_1, z_2, \tau) \simeq \frac{1}{4}\left(p(s_\iota, v_\varpi, \tau) + p(s_\iota, v_{\varpi+1}, \tau) + p(s_{\iota+1}, v_\varpi, \tau) + p(s_{\iota+1}, v_{\varpi+1}, \tau)\right), \tag{43}$$

where $(z_1, z_2) \in [s_\iota, s_{\iota+1}] \times [v_\varpi, v_{\varpi+1}]$. Thus, we have

$$\begin{aligned}\mathcal{M}_{\iota,\varpi} =& \frac{1}{4}\left(p(s_\iota, v_\varpi, \tau) + p(s_\iota, v_{\varpi+1}, \tau) + p(s_{\iota+1}, v_\varpi, \tau) + p(s_{\iota+1}, v_{\varpi+1}, \tau)\right)\\ &\times H\left(\frac{s_\iota}{s_i}, \frac{s_{\iota+1}}{s_i}, v_\varpi - v_j, v_{\varpi+1} - v_j\right),\end{aligned} \tag{44}$$

where

$$H(a_1, a_2, a_3, a_4) = \int_{a_1}^{a_2}\int_{a_3}^{a_4} g(z_1, z_2) dz_2 dz_1. \tag{45}$$

Remarking that, the computation of the integral term in (2) via different numerical methods can be done in an efficient way, e.g., by using FFT, see [22] for more details. In this section, we do not combine our proposed explicit procedure with any other methods such as FFT.

Financially speaking to avoid arbitrage pricing and complete the procedure of solving (2) numerically, this is now the time to impose of the boundary conditions for the PIDE, [23,35]. For two sides of the boundaries, we first obtain the zeroth order Taylor approximation of the boundary condition and then go ahead by considering that the option price is time stationary. For $v = 0$, and $v = v_{\max}$, the sided discretized equation for the points located on such positions is considered to tend to the boundary condition while number of discretization points increases.

By imposing the boundaries, we attain the following set of semi-discretized (linear) ODEs with constant coefficients subject to a non-smooth initial condition:

$$\begin{cases} P'(\tau) = \mathcal{A}P(\tau), & 0 < \tau \le T,\\ P(0) = \phi(s). \end{cases} \tag{46}$$

It is stated that throughout the paper A and $\mathcal{A}$ stand for the system matrix before imposing the boundaries and after incorporation, respectively.

Now the main issue is to solve the set of ODEs (46). The exact resolution of the system of stiff ODEs (46) can be represented as [36]:

$$P(\tau) = e^{\tau\mathcal{A}}P(0). \tag{47}$$

Noting that when $\tau \neq 1$, by using the following substitution

$$\mathcal{A} \leftarrow \tau\mathcal{A}, \tag{48}$$

we use (47) similarly.

As long as one faces with a large time horizon, many time steps may be required in order to obtain a given accuracy. In lieu of applying time-stepping methods for system of ODEs, the authors in [27] investigated the exponential time integration (ETI) method. The merit of the ETI approach is that it is a one step method and hence we need not to discuss its stability and temporal discretized accuracy. For more one may refer to [37].

Here, we use a Krylov method at which no matrix products or divisions but matrix-vector multiplications are needed in the whole solving procedure. Recently, a Krylov method has been applied for a 3D advection–diffusion–reaction (ADR) PDE in [38]. Note that a rational approach based on Carathéodory–Féjer is an efficient method [39] for computing the matrix exponential and then finding its action on a vector. Here we compute the action of this matrix function on the payoff vector directly.

In this approach, the main concept is to roughly project the exponential of a matrix into a small Krylov subspace by applying the well-known Arnoldi process, [40, chapter 13.2.1].

We can simply and efficiently employ this approach (see [41] for several related discussions and algorithms) by calling the built-in programs have already been coded in the common programming packages (such as Matlab or Mathematica). As such, the action of this matrix exponential on the payoff vector can be computed in Mathematica environment as follows:

```
MatrixExp[coefficient matrix, payoff, Method -> "Krylov"];
```

5. Computational aspects

The results of various methods are compared for several practical tests using $v_{\min} = s_{\min} = 0$, $s_{\max} = 3E$, $v_{\max} = 1$ and $\alpha = 5.5$. Although the choice $s_{\max} = 4E$ is a common selection for the truncation of the underlying asset domain, $s_{\max} = 3E$ has also been used widely in literature [23] and it is here considered for all the compared methods. These values are mainly chosen as same as the compared papers in this category. To solve (2), we employ the following methods.

- The uniform FD scheme with second-order spatial discretizations (denoted by FD), a discretization for the double integral as in Section 4 and the explicit Euler's method for time-stepping.
- The uniform FD method with second-order spatial discretizations denoted by FD-Krylov and the discretization of the double integral as in Section 4 and the Krylov subspace one-step method already coded inside Mathematica.
- The scalable algebraic multigrid method (AMG–STS) with $\tilde{N}$ being the number of temporal discretizations in [23].
- The exact simulations based on the results in [42].
- The proposed approach in Sections 2–4 denoted by RBF-FD using different values of μ and the Krylov subspace one-step method already coded inside Mathematica.
- And the second order FD method (FD-Krylov) but with the non-uniform meshes chosen for the RBF-FD method. This scheme is shown by FD-Krylov 2.

Since the error of GMQ RBF-FD scheme for five nodes is smaller than the one for three nodes, one may state that the numerical results for five nodes should also be included. Although this can be done similarly, here the main point is that only the GMQ RBF-FD method with three unstructured nodes can yield quick convergence. The reason is in three things, first the nature of RBF-FD schemes. Second by focusing on the non-differentiable part of the domain. And third by imposing a non-uniform (adaptive) grid of discretization points, which increases the accuracy at the hot area while it allocates fewer points away from this area.

Hence, because of these issues we do not include the numerical results based on five uniform nodes. Recalling that higher-order differences produce more off-diagonals and make the coefficient matrices to have larger bandwidth.

Also here we have written the codes in Mathematica 11.0 [43, chapter 1.11], and use an interpolation (with a built-in function in this programming package) to find the required option value at any position of the domain.

The tests are done on a laptop having Windows 7 Ultimate having Intel(R) Core(TM) i5-2430M CPU 2.40 GHz processor and 16.00 GB of RAM on a 64-bit operating system. Time is reported in seconds. Recalling that option prices should be accurate to the nearest cent. In addition in this section, $\lambda_{\max}$ indicates the largest eigenvalue of $\mathcal{A}$ in absolute value sense. Besides, whenever the one-step Krylov method is applied for finding the solution of (46), the transformation (48) is imposed and then $\lambda_{\max}$ is reported in tables.

A dropping technique is imposed on the matrix $\mathcal{A}$ with the threshold 10^{-10} (it is really small and relatively close to machine precision) to replace all the very small entries with zero, see a basic notion about dropping elements for matrices in [44]. The choice of this threshold is based on the command `Chop[]` in our programming language to remove the unnecessary calculations and make the final system matrix to be more sparse.

Throughout the section, due to independency of our Krylov subspace method for solving the resulting set of linear ODEs (46), we use "–" to show that it does not require a step size $\Delta\tau$. Additionally, throughout the tables "$a(-b)$" denotes "$a \times 10^{-b}$". Furthermore, the reference solution is extracted from the fundamental papers published in this category as will be pointed out later.

As a measure for checking the convergence, the root mean square relative error (RMSRE) of $\mathcal{N}$ computed solutions $\bar{p}$ whose expected values are $\mathbf{p}$, respectively, is considered as follows:

$$\epsilon = \left(\sum_{i=1}^{\mathcal{N}} \frac{1}{\mathcal{N}} \left| \frac{\bar{p}(s_i, v, \tau) - \mathbf{p}(s_i, v, \tau)}{\mathbf{p}(s_i, v, \tau)} \right|^2 \right)^{\frac{1}{2}}. \tag{49}$$

This is only computed and reported in the reference points which are important in financial point of view.

Using the discussions in Section 2, we know that $c > h$ should be chosen for the shape parameter. Accordingly, a way to choose this parameter adaptively is applied throughout this section (unless stated clearly) as follows [45,46]:

$$c_{\text{ along } s} = 1.1 \max\{\Delta s_i\}, \tag{50}$$

and

$$c_{\text{ along } v} = 1.7 \max\{\Delta v_j\}, \tag{51}$$

where Δs_i and Δv_j are the increment along s and v, respectively. The coefficients in (50) and (51) can somehow be optimized, i.e. they should be chosen to guarantee optimal accuracy. However, such an investigation can be studied in future works.

Example 5.1 ([23]). This problem evaluates the prices of a vanilla put case for the SVCJ PIDE having the parameters: $T = 0.5$, $\lambda = 0.2$, $E = 100$, $\rho_J = -0.5$, $v = 0.2$, $\rho = -0.5$, $r = 0.03$, $q = 0$, $\sigma = 0.25$, $\kappa = 2$, $\theta = 0.04$, $\delta^2 = 0.16$, $\gamma = -0.5$. The referenced prices at $v = 0.04$ and $s = \{90, 100, 110\}$ are 11.475480 6.928637, and 4.641829, respectively.

The numerical results for this test are furnished in Tables 1–2. We have chosen four different values for μ in Table 1 for our RBF-FD method to show how it behaves once we increase the number of discretization points using the modified Legendre zeros discussed in Section 4. Results reveal that FD and FD-Krylov require more number of discretization points to provide the desired accuracy, while the proposed scheme applying the RBF-FD method and non-uniform nodes converges. FD-Krylov 2 sounds to be the worst method not because of applying the Krylov method, but because the

Table 1
Error decay history in Example 5.1.

Method	m	n	$\Delta\tau$	Re($\lambda_{\max}$)	p at 90	p at 100	p at 110	ϵ	Time
FD									
	8	4	0.05	−55.05	13.4336	9.1727	6.1160	2.798(−1)	0.16
	16	8	0.01	−313.25	9.5236	1.6523	1.2730	6.152(−1)	0.88
	16	16	0.005	−321.24	11.1125	5.1646	3.4573	2.089(−1)	2.64
	32	16	0.001	−1526.89	11.4755	6.6748	4.5948	2.194(−2)	12.98
	32	20	0.0005	−1528.86	11.4842	6.7247	4.5759	1.886(−2)	23.65
	32	24	0.0005	−1529.00	11.4880	6.7486	4.5644	1.782(−2)	35.08
FD-Krylov									
	8	4	–	−27.52	13.4332	9.1705	6.1115	2.793(−1)	0.11
	16	8	–	−156.62	11.0157	4.9236	3.4220	2.268(−1)	0.71
	16	16	–	−160.62	11.1116	5.1575	3.4515	2.098(−1)	2.86
	32	16	–	−764.43	11.4843	6.7240	4.5752	4.949(−1)	10.70
	32	20	–	−764.43	13.7018	10.045	7.9019	1.895(−2)	14.77
	32	24	–	−764.50	11.4881	6.7479	4.5637	1.792(−2)	25.09
FD–Krylov 2									
	8	4	–	−100.47	11.8576	6.6281	6.9438	2.880(−1)	0.14
	16	8	–	−821.25	11.1761	7.1081	4.7323	2.403(−2)	0.74
	16	16	–	−827.09	11.4796	6.9439	4.8650	2.778(−2)	2.81
	32	16	–	−8401.94	11.2639	6.6636	4.5043	2.988(−2)	10.27
	32	20	–	−8399.19	11.2494	6.6820	4.4961	2.966(−2)	15.81
	32	24	–	−8395.22	11.2522	6.6863	4.4970	2.929(−2)	23.85
RBF-FD									
$\mu = 1/2$	8	4	–	−137.82	11.5707	6.1563	6.3747	2.249(−1)	0.14
	16	8	–	−967.17	11.0481	7.0239	4.3486	4.307(−2)	0.79
	16	16	–	−974.92	11.4381	6.7194	4.4716	2.749(−2)	2.67
	32	16	–	−13 152.60	11.4329	6.8106	4.6263	1.024(−2)	10.15
	32	24	–	−13 161.00	11.4218	6.8642	4.6043	7.602(−3)	23.31
RBF-FD									
$\mu = -1/2$	8	4	–	−139.43	11.7510	6.2651	6.4323	2.298(−1)	0.15
	16	8	–	−980.29	11.1916	7.1177	4.3987	3.696(−2)	0.78
	16	16	–	−987.76	11.5912	6.8296	4.5305	1.713(−2)	2.96
	32	16	–	−13 344.40	11.5771	6.9236	4.6951	8.385(−3)	10.25
	32	24	–	−13 351.60	11.5461	6.9650	4.6667	5.604(−3)	24.86
RBF-FD									
$\mu = 0.001$	8	4	–	−138.73	11.6724	6.2181	6.4096	2.279(−1)	0.15
	16	8	–	−974.62	11.1258	7.0749	4.3789	3.907(−2)	0.79
	16	16	–	−982.26	11.5193	6.7784	4.5058	2.115(−2)	2.96
	32	16	–	−13 262.70	11.5108	6.8736	4.6658	5.751(−3)	11.64
	32	24	–	−13 270.60	11.4897	6.9209	4.6406	9.743(−4)	27.11
RBF-FD									
$\mu = -0.02$	8	4	–	−138.76	11.6761	6.2203	6.4108	2.280(−1)	0.16
	16	8	–	−974.88	11.1287	7.0769	4.3799	3.896(−2)	0.78
	16	16	–	−982.51	11.5224	6.7807	4.5070	2.094(−2)	2.78
	32	16	–	−13 266.50	11.5138	6.8759	4.6672	5.742(−3)	11.42
	32	24	–	−13 274.30	11.4922	6.9229	4.6418	9.676(−4)	27.52

Table 2
Computational results of the AMG–STS method in Example 5.1 based on [23].

m	n	N	$\tilde{N}$	ϵ
17	17	289	2	1.248(−1)
33	33	1089	4	4.590(−2)
65	65	4225	8	1.021(−2)
129	129	16 641	16	2.474(−3)
257	257	66 049	32	7.314(−4)

generated mesh does not sound to be that useful if the FD method gets employed on it. This will be re-observed in the next test. Apart from a comparison with the standard FD method, results of the AMG–STS method are extracted from [23] and reported in Table 2. Numerical results manifest that our new procedure converges to the solution and is competitive.

A question may arise now that Are the theoretical estimates confirmed? To respond this, it is recalled that the convergence rate of an algorithm can be assessed in terms of the ratio of successive norms of numerical errors based on the analytical solutions. As a matter of fact, the approximated computational order of convergence (ACOC) can be

Table 3
Rate of ACOCs for various schemes in Example 5.1.

Method	m	n	ϵ	ACOC
RBF-FD	8	8	6.844(−2)	
$\mu = 1/2$	16	16	2.749(−2)	1.66
RBF-FD	8	8	6.699(−2)	
$\mu = -1/2$	16	16	1.713(−2)	1.67
RBF-FD	8	8	6.702(−2)	
$\mu = 0.001$	16	16	2.115(−2)	1.31
RBF-FD	8	8	6.699(−2)	
$\mu = -0.02$	16	16	2.094(−2)	1.96
Mean of ACOCs				1.65

roughly written as [47]:

$$\text{ACOC} = \left|\log_2 \frac{\epsilon_{2m,2n}}{\epsilon_{m,n}}\right|, \tag{52}$$

where $\epsilon_{m,n}$, $\epsilon_{2m,2n}$ are computed using (49) by doubling the number spatial discretization points. So, we compute the ACOCs now for various members of proposed procedure for solving the financial PIDE problem. Results based on (52) for different values of μ are given in Table 3. They confirmed the theoretical error equations given in Section 2 since the proposed RBF-FD approximations for the first and second derivatives are quadratic and linear due to non-uniform meshes. Hence, the means of ACOCs, 1.65 confirms the discussions. Although this rate of convergence mainly due to the non-smooth payoff shows not a fast rate of convergence, in essence the superiority of the RBF-FD scheme helps to get much more accurate results than second order or higher order FD-like methods.

Besides though the numerical convergence is linear, due to applying non-uniform nodes with an emphasis around the hotzone we can overcome on the non-smoothness difficulty of the initial condition and also arrive at the convergence phase quickly. This is unlike the FD and FD-Krylov methods, which requires many discretization nodes to arrive at the convergence phase. Furthermore, the FD and FD-Krylov methods suffer from the effect of the non-smoothness of the initial condition because of applying uniform discretization points.

Based on the results in Tables 1–2, we comment that an optimal choice of μ should be investigated more deeply which is clearly depends on the number of discretization points in each spatial variables. This could be focused for future research in this category. But the numerical results in this section reveal that for different choice of this parameter, one may get desirable results in option pricing in a reasonable piece of time in contrast to the existing solvers. We also point out that parameter tuning to attain results which are accurate up 5 or 6 digits is unnecessary for our mathematical model since in market, option prices are required to be correct up to at most three digits. This is because prices are reported at most in base of cents and numerical results up to 5 or 6 digits is unnecessary in practice.

Example 5.2. This problem tests the valuation of a SVCJ PIDE for a vanilla call case with the option parameters below [42]: $\rho_J = -0.38$, $\upsilon = 0.05$, $\rho = -0.82$, $E = 100$, $r = 0.0319$, $q = 0$, $\sigma = 0.14$, $\kappa = 3.46$, $T = 1$, $\theta = 0.008$, $\lambda = 0.47$, $\delta = 0.0001$, and $\xi = -0.1$. The reference value at $(100, 0.007567, 1)$ is 6.8619.

This test is important since the 2D PDF (3) is almost unchanged in terms of its graph and makes the resulting system of ODEs to have a sparse matrix unlike the previous experiments at which the system matrix was rather dense. In general the presence of the integral term in (2) results in dense matrices due to this non-local integral operator ($\lambda \neq 0$). Despite this, for some parameter settings as the ones in Example 5.2, the resulting discretization matrices would not be fully dense and they are somewhat sparse.

Results of the new scheme are given in Table 4, while simulation based results are extracted from [42] and reported in Table 5. Our scheme is again competitive in terms of the error accuracy and the computational elapsed times.

A significant issue to discuss is that the choice of the free parameter μ is *rather* problem-dependent and it relies on the choice of the number of discretization nodes. An experimental investigation for the choice of c is done in Fig. 1 by choosing $c_{\text{along } s} = \varkappa \max \Delta s_i$ and $c_{\text{along } v} = \varkappa \max \Delta v_j$ for $m = 32$ and $n = 16$ and the same conditions as in Example 5.2. It is observed that an optimal value could occur for the choice of the shape parameter around $\varkappa = 1.12$ with this number of discretization nodes along each dimension.

Noting that based on the comparison tables here, whatever the real part of the eigenvalues increase, the more stiff system should be solved but more accurate results can be obtained. Another observation is that although the RBF-FD formulas on non-uniform grids for estimating the 2nd derivative is only of first order, this order is enough even for the challenging model (2). This is mainly because of using non-uniform grids as well as the nature of RBF-FD approach. Hence, we even do not need to apply RBF-FD scheme with stencils having five uniform nodes.

Before ending up this section, we give the numerical results by varying various parameters of the proposed procedure to check the convergence as well as the positivity of the numerical solution and obtain the Greeks for Example 5.2. To

Table 4
Convergence history of a call SVCJ option pricing in Example 5.2 for different methods.

Method	m	n	$\Delta\tau$	$\mathrm{Re}(\lambda_{\max})$	p at 100	ϵ	Time
FD							
	8	4	0.05	−61.00	9.7507	4.209(−1)	0.18
	16	8	0.001	−326.94	5.7325	1.645(−1)	1.07
	16	16	0.001	−341.272	5.7152	1.671(−1)	3.73
	32	16	0.001	−1560.56	6.7550	1.556(−2)	14.75
	32	20	0.0005	−1569.93	6.7593	1.494(−2)	23.03
FD-Krylov							
	8	4	–	−42.21	9.7823	4.256(−1)	0.10
	16	8	–	−280.96	5.7088	1.680(−1)	0.90
	16	16	–	−294.53	5.7123	1.675(−1)	3.60
	32	16	–	−1457.74	6.7411	1.760(−2)	14.69
	32	20	–	−1467.40	6.7548	1.559(−2)	23.29
FD–Krylov 2							
	8	4	–	−207.14	6.8949	4.813(−3)	0.14
	16	8	–	−1658.95	6.4575	5.892(−2)	0.91
	16	16	–	−1685.88	6.6153	3.592(−2)	3.42
	32	16	–	−16 851.00	6.5008	5.261(−2)	14.54
	32	20	–	−16 861.30	6.5091	5.140(−2)	22.63
RBF-FD							
$\mu = 1/2$	8	4	–	−279.82	6.6343	3.316(−2)	0.21
	16	8	–	−1945.29	6.5063	5.181(−2)	0.96
	16	16	–	−1968.19	6.7368	1.822(−2)	3.27
	32	16	–	−26 331.70	6.8791	2.519(−3)	12.29
	32	20	–	−26 347.10	6.8941	4.695(−3)	20.11

Table 5
Simulation with the exact method for Example 5.2 via [42].

Number of simulation trials	ϵ	Time
10,000	7.20(−2)	1.5
40,000	3.69(−2)	5.8
160,000	1.84(−2)	23.3
640,000	9.20(−3)	93.5
2,560,000	4.60(−3)	373.0
10,240,000	2.30(−3)	1491.3
40,960,000	1.10(−3)	5967.5

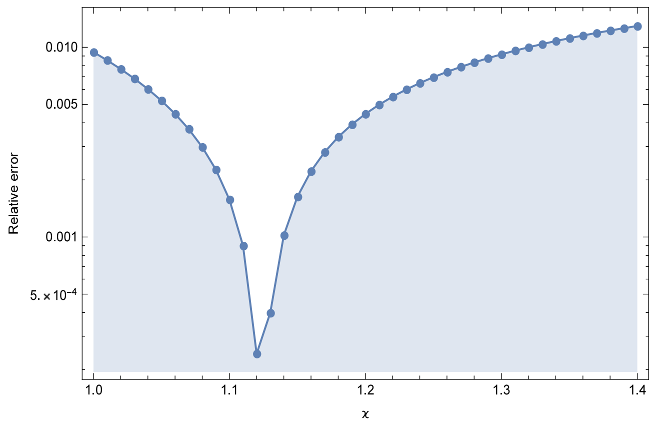


Fig. 1. Error decay for Example 5.2 by employing RBF-FD with $\mu = 0.5$ and changing the shape parameter.

this goal, let us illustrate the results of the GMQ RBF-FD method (using a three-node non-uniform stencil) with $\mu = 0.5$ for $m = 80$ and $n = 30$, and the following new values for the involved parameters, $\alpha = 0.1$, $c_{\text{along } s} = 10\max\{\Delta s_i\}$, and $c_{\text{along } v} = 10\max\{\Delta v_j\}$. This yields $p = 6.85936$ and subsequently the error $\epsilon = 3.707(-4)$ at $(100,007567,1)$. The numerical results are given in Fig. 2 showing the stable nature of the proposed GMQ RBF-FD method for tackling this PIDE problem even with the new parameters.

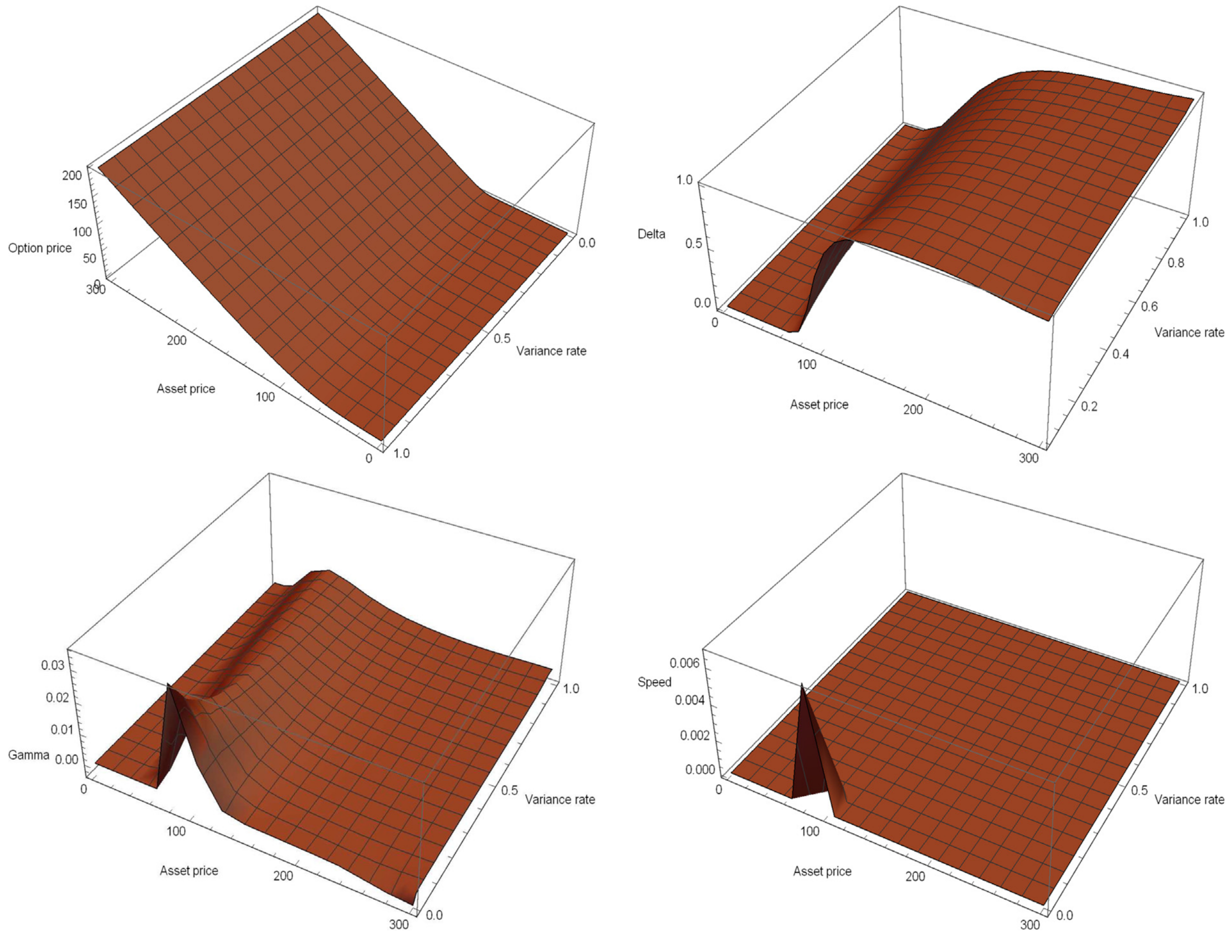


Fig. 2. The numerical solution for option pricing Example 5.2 in top-left, its Delta, Gamma and Speed in top-right, bottom-left and bottom-right, respectively.

Recalling that (see e.g. [48]), Delta, $\frac{\partial u}{\partial s}$, is one of four important risk measures applied by option traders. In fact, it measures the degree to which an option is exposed to shifts in the price of the underlying asset. Furthermore, Gamma, $\frac{\partial^2 u}{\partial s^2}$, is the rate of change in an option's delta per one-point move in the underlying asset's price. Gamma is a significant measure of the convexity of a derivative's value, in relation to the underlying.

Speed, $\frac{\partial^3 u}{\partial s^3}$, measures the rate of change in Gamma with respect to changes in the underlying price. This is also sometimes referred to as the Gamma of the Gamma. A high Speed value indicates that Gamma is more sensitive to moves in price of the underlying asset. A non-smooth behavior around the strike price for the Speed can simply be observed and confirmed from the numerical results.

Remark 5.1. One may ask that how to choose n, m, c, and μ for any other model problem. To respond this, it is proposed that c must be greater than the spatial step sizes (on nonuniform grids) and accordingly one simple choice would be as follows:

$$c = \frac{3}{2}\max\{h_i\}. \tag{53}$$

Clearly by increasing m and n the size of the system of ODEs increases but it is expected to observe good accuracy for non-uniform meshes by 20 to 30 nodes only. And lastly, $\mu = 1/2$ or $\mu = -1/2$ are recommended for general PDE/PIDE problems since they are the multiquadric and inverse multiquadric forms which have experienced the most stability in terms of numerical computations (in general setting).

We end this section be a general conclusion that a reader can take away. Beside Remark 5.1, one can depend on RBF-FD approximations as efficient solvers for different PDE-based models as long as an efficient mesh has been generated. Such a mesh for tensor grids can be generated based on our proposed strategies in Section 4 or its variants as needed.

6. Conclusions

Many financial assets, such as currencies, commodities, and equity stocks, exhibit both jumps and stochastic volatility, which are specially prominent in the market after the financial crisis. Numerically speaking, in RBF generated FD formulas, all estimations are local, but points could be distributed freely with the several features of being easy to implement for high dimensional problems (such as the ones in financial mathematics), and being excellent for distributed memory computers.

We have presented the weights of the RBF-FD scheme in the presence of the GMQ RBF, which is an extension of the well-known MQ and IMQ RBFs. The weights have been constructed for stencils having three equidistant and non-equidistant nodes, as well as for a stencil having five equidistant nodes. The error equations have been proposed showing the convergence behavior of these approximations. Some discussions about how to choose the free parameter μ have also been furnished.

Application of these novel formulas along with a new procedure based on orthogonal Legendre polynomials for generating a non-uniform (adaptive) distribution of the nodes (for a financial problem), have been discussed and we have observed a fast and stable convergence in solving our challenging PIDE problem (2) in practice.

Implementation of the GMQ RBF-FD procedure based on "with memorization" of the nonzero parameter μ is under investigation. This matter or an application of such formulas for other asset pricing problems could be considered as forthcoming works in this research field.

Acknowledgments

This work is supported by the Fundamental Research Funds for the Central Universities, National Science Foundation of China (Grant No. 12071149), Natural Science Foundation of Shanghai (Grant No. 19ZR1414100), and Science and Technology Commission of Shanghai Municipality (No. 18dz2271000). The authors are grateful to all three referees (particularly referee 2) for many comments and corrections, which helped improve the readability and reliability of this work.

Appendix A

A.1. The proof of Lemma 3.1

Proof. Substituting the function f by the GMQ RBF (1), centered at $x_{i-2} = x_i - 2h$, $x_{i-1} = x_i - h$, x_i, $x_{i+1} = x_i + h$, and $x_{i+2} = x_i + 2h$ results in the following set of equations:

$$\begin{aligned}
&4h\mu\left(c^2+4h^2\right)^{\mu-1} = \Upsilon_{i-2}c^{2\mu} + \Upsilon_{i-1}\left(c^2+h^2\right)^{\mu} + \Upsilon_i\left(c^2+4h^2\right)^{\mu}\\
&\qquad + \Upsilon_{i+1}\left(c^2+9h^2\right)^{\mu} + \Upsilon_{i+2}\left(c^2+16h^2\right)^{\mu},\\
&2h\mu\left(c^2+h^2\right)^{\mu-1} = \Upsilon_{i-1}c^{2\mu} + \Upsilon_{i-2}\left(c^2+h^2\right)^{\mu} + \Upsilon_i\left(c^2+h^2\right)^{\mu}\\
&\qquad + \Upsilon_{i+1}\left(c^2+4h^2\right)^{\mu} + \Upsilon_{i+2}\left(c^2+9h^2\right)^{\mu},\\
&\Upsilon_i c^{2\mu} + \Upsilon_{i-1}\left(c^2+h^2\right)^{\mu} + \Upsilon_{i+1}\left(c^2+h^2\right)^{\mu}\\
&\qquad + \Upsilon_{i-2}\left(c^2+4h^2\right)^{\mu} + \Upsilon_{i+2}\left(c^2+4h^2\right)^{\mu} = 0,\\
&-2h\mu\left(c^2+h^2\right)^{\mu-1} = \Upsilon_{i+1}c^{2\mu} + \Upsilon_i\left(c^2+h^2\right)^{\mu} + \Upsilon_{i+2}\left(c^2+h^2\right)^{\mu}\\
&\qquad + \Upsilon_{i-2}\left(c^2+9h^2\right)^{\mu} + \Upsilon_{i-1}\left(c^2+4h^2\right)^{\mu},\\
&-4h\mu\left(c^2+4h^2\right)^{\mu-1} = \Upsilon_{i+2}c^{2\mu} + \Upsilon_{i+1}\left(c^2+h^2\right)^{\mu}\\
&\qquad + \Upsilon_{i-2}\left(c^2+16h^2\right)^{\mu} + \Upsilon_{i-1}\left(c^2+9h^2\right)^{\mu} + \Upsilon_i\left(c^2+4h^2\right)^{\mu}.
\end{aligned} \tag{54}$$

The system (54) can be written as:

$$\mathrm{A}\begin{pmatrix}\Upsilon_{i-2}\\ \Upsilon_{i-1}\\ \Upsilon_i\\ \Upsilon_{i+1}\\ \Upsilon_{i+2}\end{pmatrix} = -\mathbf{b}, \tag{55}$$

where

$$\mathrm{A} = \begin{pmatrix}
-c^{2\mu} & -\left(c^2+h^2\right)^{\mu} & -\left(c^2+4h^2\right)^{\mu} & -\left(c^2+9h^2\right)^{\mu} & -\left(c^2+16h^2\right)^{\mu}\\
-\left(c^2+h^2\right)^{\mu} & -c^{2\mu} & -\left(c^2+h^2\right)^{\mu} & -\left(c^2+4h^2\right)^{\mu} & -\left(c^2+9h^2\right)^{\mu}\\
\left(c^2+4h^2\right)^{\mu} & \left(c^2+h^2\right)^{\mu} & c^{2\mu} & \left(c^2+h^2\right)^{\mu} & \left(c^2+4h^2\right)^{\mu}\\
-\left(c^2+9h^2\right)^{\mu} & -\left(c^2+4h^2\right)^{\mu} & -\left(c^2+h^2\right)^{\mu} & -c^{2\mu} & -\left(c^2+h^2\right)^{\mu}\\
-\left(c^2+16h^2\right)^{\mu} & -\left(c^2+9h^2\right)^{\mu} & -\left(c^2+4h^2\right)^{\mu} & -\left(c^2+h^2\right)^{\mu} & -c^{2\mu}
\end{pmatrix}, \tag{56}$$

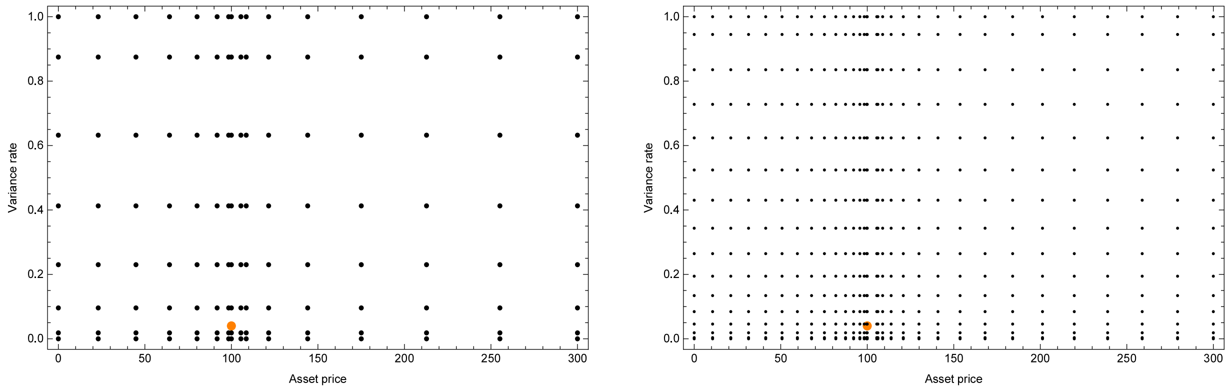


Fig. 3. Nonuniform mesh of nodes based on Legendre polynomials when $m = 16$ and $n = 8$ in left, when $m = 32$ and $n = 16$ in right. The yellow dot shows the hotzone area. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

and $\mathbf{b} = (4h\left(c^2+4h^2\right)^{\mu-1}\mu\ , 2h\left(c^2+h^2\right)^{\mu-1}\mu\ , 0, -2h\left(c^2+h^2\right)^{\mu-1}\mu, -4h(c^2+4h^2)^{\mu-1}\mu)^*$. Hence, the weights of the GMQ RBF-FD technique for approximating f' when $c > h$ are given as (29). Now by doing Taylor expansions, one may get (28). □

A.2. The proof of Lemma 3.2

Proof. The steps of the proof are similar to the previous cases. Applying the GMQ RBF (1), would yield a system of five linear equations as in (55) but with a different right hand side vector as follows:

$$\mathbf{b} = \begin{pmatrix} 2\left(c^2+4h^2\right)^{\mu-2}\mu\left(c^2+4h^2(2\mu-1)\right) \\ 2\left(c^2+h^2\right)^{\mu-2}\mu\left(c^2+h^2(2\mu-1)\right) \\ 2c^{2\mu-2}\mu \\ 2\left(c^2+h^2\right)^{\mu-2}\mu\left(c^2+h^2(2\mu-1)\right) \\ 2\left(c^2+4h^2\right)^{\mu-2}\mu\left(c^2+4h^2(2\mu-1)\right) \end{pmatrix}. \tag{57}$$

This leads to the closed-form expressions for the weights (in the limit case $c > h$) as in (32). Similar symbolic computations would result in the final error equation. This completes the proof. □

Appendix B

Fig. 3 illustrate such a meshing based on Legendre polynomials for two choices of m and n (with $\alpha = 5.5$).

To ease up the use of this procedure, we present a simple yet effective Mathematica code for producing these sets of nodes in what follows.

A Mathematica code to produce a non-uniform grid of nodes based on modified orthogonal Legendre polynomials:

```
ClearAll["Global‘*"];
e = 100.; alpha = 5.5; digits = 30;
smin = 0.; smax = 3 e; vmin = 0.; vmax = 1.;

P[n_] := LegendreP[n, x];
setroot[n_] :=
 N[x /. NSolve[P[n] == 0, x, WorkingPrecision -> digits]];
legendre[n_] := setroot[n];

m = 32;
roots = legendre[(m - 3)];
s1 = Select[roots, # > 0 &];
s2 = Sort[1 - s1];
s3 = e*Join[{0}, s1, {1}];
s4 = (e + alpha) + (smax - (e + alpha))*Join[{0.}, s2, {1.}];
sGrid = Join[s3, s4]
```

```
n = 16;
roots = legendre[2 (n - 2)];
v1 = Select[roots, # > 0 &];
v2 = Sort[1 - v1];
vGrid = vmin + (vmax - vmin) Join[{0.}, v2, {1.}]

origrid = Flatten[Outer[List, sGrid, vGrid], 1];
p0 = {e, 0.04};
hotzone1 =
  Graphics[{PointSize[Large], Orange, Point[p0]},
  ImageSize -> 450,
   AspectRatio -> 1/1.6];
ListPlot[origrid, PlotRange -> All, Frame -> True,
    PlotStyle -> Black,
   Axes -> False, ImageSize -> 450];
Show[hotzone1, %, Frame -> True,
 FrameLabel -> {"Asset price", "Variance rate"}]
```

Appendix C

The formulations (7) and (20) would be useful for the rows 2 to $m-1$, while for the first and last rows of the differentiation matrices (35) and (36), the weights using a stencil having only two nodes should be constructed based on the RBF-FD methodology. In such a manner, we have $\Xi_{1,1} = \Xi_{m,m-1} = \frac{h-2h\mu}{2c^2} - \frac{1}{h}$, $\Xi_{1,2} = \Xi_{m,m} = \frac{1}{h} - \frac{h}{2c^2}$, and $\Theta_{1,1} = \Theta_{m,m-1} = \frac{4\mu-3}{c^2}$, $\Theta_{1,2} = \Theta_{m,m} = \frac{3-2\mu}{c^2}$.

References

[1] S.A. Sarra, A local radial basis function method for advection–diffusion–reaction equations on complexly shaped domains, Appl. Math. Comput. 218 (2012) 9853–9865.
[2] J. Li, B. Nan, Simulating backward wave propagation in metamaterial with radial basis functions, Resu. Appl. Math. 2 (2019) 100009.
[3] W.E. Schiesser, Spline Collocation Methods for Partial Differential Equations with Applications in *R*, Wiley, USA, 2017.
[4] R.L. Hardy, Theory and applications of the multiquadric–biharmonic method: 20 years of discovery, Comput. Math. Appl. 19 (1990) 163–208.
[5] Y.C. Hon, A quasi-radial basis functions method for American options pricing, Comput. Math. Appl. 43 (2002) 513–524.
[6] I. Tolstykh, On using RBF-based differencing formulas for unstructured and mixed structured–unstructured grid calculations, in: Proc. 16th IMACS World Congress, Vol. 228, 2000, pp. 4606–4624.
[7] A. Heryudono, E. Larsson, A. Ramage, L. von Sydow, Preconditioning for radial basis function partition of unity methods, J. Sci. Comput. 67 (2016) 1089–1109.
[8] E.F. Bollig, N. Flyer, G. Erlebacher, Solution to PDEs using radial basis function finite-differences (RBF-FD) on multiple GPUs, J. Comput. Phys. 231 (2012) 7133–7151.
[9] V. Bayona, N. Flyer, G.M. Lucas, A.J.G. Baumgaertner, A 3-D RBF-FD solver for modeling the atmospheric global electric circuit with topography, Geosci. Model Dev. 8 (2015) 3007–3020.
[10] B. Fornberg, N. Flyer, Accuracy of radial basis function interpolation and derivative approximations on 1-D infinite grids, Adv. Comput. Math. 23 (2005) 5–20.
[11] S. Milovanović, L. von Sydow, Radial basis function generated finite differences for option pricing problems, Comput. Math. Appl. 75 (2018) 1462–1481.
[12] S.A. Sarra, M.E. Chenoweth, A numerical study of generalized multiquadric radial basis function interpolation, SIAM Undergrad. Res. Online 22 (2009) 58–70.
[13] V. Bayona, M. Moscoso, M. Carretero, M. Kindelan, RBF-FD formulas and convergence properties, J. Comput. Phys. 229 (2010) 8281–8295.
[14] G.E. Fasshauer, Meshfree Approximation Methods with Matlab, World Scientific Publishing Co., Singapore, 2007.
[15] J.-H. Jung, A note on the gibbs phenomenon with multiquadric radial basis functions, Appl. Numer. Math. 57 (2007) 213–229.
[16] J.-H. Jung, V.R. Durante, An iterative adaptive multiquadric radial basis function method for the detection of local jump discontinuities, Appl. Numer. Math. 59 (2009) 1449–1466.
[17] D. Duffie, J. Pan, K. Singleton, Transform analysis and asset pricing for affine jump diffusions, Econometrica 68 (2000) 1343–1376.
[18] H. Buehler, in: R. Cont (Ed.), Heston Model, Wiley, USA, 2010.
[19] A. Itkin, P. Carr, Using pseudo-parabolic and fractional equations for option pricing in jump diffusion models, Comput. Econ. 40 (2012) 63–104.
[20] B. Eraker, M. Johannes, N. Polson, The impact of jumps in volatility and returns, J. Finance 58 (2003) 1269–1300.
[21] K.-H. Kim, M.-G. Sin, Efficient hedging in general Black–Scholes model, Finan. Math. Appl. 3 (2014) 1–9.
[22] L. Feng, V. Linetsky, Pricing options in jump-diffusion models: An extrapolation approach, Oper. Res. 56 (2008) 304–325.
[23] S. Salmi, J. Toivanen, L. von Sydow, An IMEX-scheme for pricing options under stochastic volatility models with jumps, SIAM J. Sci. Comput. 36 (2014) 817–834.
[24] G.W. Griffiths, Numerical Analysis using *R*: Solutions to ODEs and PDEs, Cambridge University Press, UK, 2016.
[25] J.R. Xaio, M.A. McCarthy, A local heaviside weighted meshless method for two-dimensional solids using radial basis functions, Comput. Mech. 31 (2003) 301–315.
[26] E.J. Kansa, R.C. Aldredge, L. Ling, Numerical simulation of two-dimensional combustion using mesh-free methods, Eng. Anal. Bound. Elem. 33 (2009) 940–950.

[27] F. Soleymani, M.Z. Ullah, A multiquadric RBF–FD scheme for simulating the financial HHW equation utilizing exponential integrator, Calcolo 55 (2018) 1–26, 51.
[28] M. Abramowitz, I.A. Stegun, Handbook of Mathematical Functions with Formulas, Graphs, and Mathematical Tables, Applied Mathematics Series, vol. 55, USA, 1983.
[29] B. Fornberg, A Practical Guide to Pseudospectral Methods, Cambridge University Press, UK, 1996.
[30] W. Hackbusch, Tensor Spaces and Numerical Tensor Calculus, Springer–Verlag, Berlin Heidelberg, 2012.
[31] L.N. Trefethen, Spectral Methods in Matlab, SIAM, USA, 2000.
[32] H. Zhang, F. Ding, On the kronecker products and their applications, J. Appl. Math. 2013 (2013) 8, 296185.
[33] H.V. Henderson, F. Pukelsheim, S.R. Searle, On the history of the kronecker product, Linear Multilinear Algebra 14 (1983) 113–120.
[34] R.E. Lynch, J.R. Rice, D.H. Thomas, Tensor product analysis of partial difference equations, Bull. Amer. Math. Soc. 70 (1964) 378–384.
[35] M.D. Marcozzi, S. Choi, C.S. Chen, On the use of boundary conditions for variational formulations arising in financial mathematics, Appl. Math. Comput. 124 (2001) 197–214.
[36] J. Loffeld, M. Tokman, Comparative performance of exponential, implicit, and explicit integrators for stiff systems of ODEs, J. Comput. Appl. Math. 241 (2013) 45–67.
[37] H.-K. Pang, H.-W. Sun, Fast exponential time integration for pricing options in stochastic volatility jump diffusion models, East Asian J. Appl. Math. 4 (2014) 52–68.
[38] H.P. Bhatt, A.Q.M. Khaliq, B.A. Wade, Efficient Krylov–based exponential time differencing method in application to 3D advection–diffusion–reaction systems, Appl. Math. Comput. 338 (2018) 260–273.
[39] J.V. Deun, L.N. Trefethen, A robust implementation of the Carathéodory–Fejér method for rational approximation, BIT 51 (2011) 1039–1050.
[40] N.J. Higham, Functions of Matrices: Theory and Computation, Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, 2008.
[41] M. Caliari, P. Kandolf, A. Ostermann, S. Rainer, Comparison of software for computing the action of the matrix exponential, BIT 54 (2014) 113–128.
[42] M. Broadie, Ö. Kaya, Exact simulation of stochastic volatility and other affine jump diffusion processes, Oper. Res. 54 (2006) 217–231.
[43] M. Trott, The Mathematica Guidebook for Numerics, Springer, New York, NY, USA, 2006.
[44] Y. Saad, Preconditioning techniques for nonsymmetric and indefinite linear systems, J. Comput. Appl. Math. 24 (1988) 89–105.
[45] L.V. Ballestra, G. Pacelli, Computing the survival probability density function in jump–diffusion models: A new approach based on radial basis functions, Eng. Anal. Bound. Elem. 35 (2011) 1075–1084.
[46] L.V. Ballestra, G. Pacelli, 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 (2012) 1546–1554.
[47] E. Cătinaş, A survey on the high convergence orders and computational convergence orders of sequences, Appl. Math. Comput. 343 (2019) 1–20.
[48] P. Ursone, How to Calculate Options Prices and their Greeks, Wiley, UK, 2015.