Noname manuscript No.
(will be inserted by the editor)

Explicit time stepping of PDEs with local refinement in space-time

Dylan Abrahamsen · Bengt Fornberg

Received: date / Accepted: date

Abstract Traditional numerical time stepping allows variable node densities in space, but not also in time. Having the ability to utilize nodes that are placed irregularly in the space-time domain leads to many advantages when solving time dependent problems. In this paper we introduce a new method utilizing the radial basis function generated finite difference (RBF-FD) approach in order to accomplish this goal. Benefits include improved stability conditions and the option to use small time steps only in select spatial regions.

Keywords Radial Basis Functions · PDEs · space-time · meshless · MOL

Mathematics Subject Classification (2010)
65M50 · 65M70
65M06 · 65M20

1 Introduction

Efficient numerical simulation of time dependent PDEs are important for investigating physical phenomena. Usually time dependent PDEs are advanced numerically in time steps that are equally long everywhere across the spatial domain. A common such approach is method of lines (MOL), amounting to discretization in space followed by the application of a standard ODE solver in time. While this often works satisfactorily, and is well supported by accuracy and stability theory, there are also important situations when this becomes very ineffective. Such cases include when

Department of Applied Mathematics
University of Colorado
Boulder, CO 80309
E-mail: dylan.abrahamsen@colorado.edu

– geometric features locally require very small space steps; CFL conditions may then force explicit time stepping methods to use much smaller time steps than what are required for temporal accuracy and
– high resolution is needed only in small regions of the space-time domain (such as around narrow propagating pulses).

Whenever resolution requirements or CFL restrictions force time steps to be small somewhere in space, the MOL concept requires them to be equally small everywhere. A somewhat related situation arises for stiff ODE systems, when different equations evolve on different time scales (discussed in the context of space-time discretization in [7]). The two main ideas in the literature for allowing time steps of different sizes in different parts of a spatial domain are

– use block-like (or, more generally, patch-like) space-time domain decomposition, where discretizations can differ between blocks, and
– use a 'tent pitching idea', where a piecewise linear time front is advanced sufficiently short in time by raising 'tents' with 'tent poles' that the tent's side slopes are small enough so that no characteristics can enter through these sides. The numerical solution is then advanced independently within the different 'tents'.

In the first case, various idea of interpolation, extrapolation, and projections have been considered for matching solutions along the sides of the blocks. The tent pitching concept is mainly applicable to wave equations (with limited characteristic speeds). In localized refined patches, fully implicit or implicit-explicit (IMEX) time stepping may be cost-effective. Several papers contain in their introductions reviews of these (and other) concepts for achieving 'local time stepping' (often given the acronym LTS); see for example [1,2,8,9,19].

In the context of using RBFs, partly grid-based linked space-time approaches are described in [10,24]. Regarding fully scattered nodes in the space-time domain, all versions so far are based on the idea of collocation within extended regions of the space-time domain, followed by solving the resulting large coupled systems of equations (which may be of least squares type). Within this approach, the space-time domain RBF approximations can themselves be global [22,23,26] or local [20] (if so, producing sparse systems). In this latter case, moving least squares (MLS) approximations can be a viable alternative to (also local) RBF-FD approximations [24] [1].

In this paper we present a new and conceptually entirely different approach, utilizing radial basis function generated finite difference (RBF-FD) for explicit time stepping. This approach is mesh-free on all scales with nodes scattered throughout the space-time domain. Local node densities are chosen to match local resolution

[1] Although similar in many respects, RBF-FD (in its form known as PHS+poly, see Sections 5.1.5 and 5.1.7 in [16]) and MLS have also significant differences, as analyzed in [3]. The last paragraph in the Conclusions of this paper notes: "Overall, PHS+poly has performed superior than MLS. It can not only achieve at least the same accuracy than MLS, but can also overcome the harmful Runge's phenomenon for any polynomial degree. This result potentially opens new opportunities for PHS+poly in areas of application where MLS is the preferred choice."

requirements. We abbreviate it RBF-TD, standing for RBF-based time discretization. No matter how the nodes are scattered in space-time, the PDE solution is at each step advanced to the lowest node in time that has not yet received a solution value.

This paper is organized as follows: we first demonstrate the RBF-TD procedure for a 1-D wave equation in Section 2.1. We note in Section 2.4 that, even if the nodes are restricted to a Cartesian grid in space-time (i.e. not utilizing any of RBF-TD's local space-time refinement capability), it can still improve both accuracy and stability compared to standard explicit MOL discretizations. Section 2.5 demonstrates the RBF-TD approach for the 1-D heat equation. Further equations are considered in Section 3, including variable speed for the 1-D transport equation and the viscid Burger's equation. Some concluding observations are given in Section 4.

2 Description of the RBF-TD procedure

The RBF-TD procedure is most easily first described for a 1-D one-way wave equation test case. We first discuss the test problem and node setup for the transport equation in Section 2.1. In Section 2.2 the calculation of each stencil is explained. Sections 2.3 and 2.4 demonstrate the results of the one-way transport equation. Finally, in Section 2.5, the heat equation is examined.

2.1 Demonstration problem: 1-D wave equation

We consider first the problem of solving the PDE

$$\frac{\partial u}{\partial t} + \frac{\partial u}{\partial x} = 0 \tag{1}$$

together with an initial condition (IC) at $t = 0$ and a boundary condition (BC) at $x = 0$ that obey the analytic solution

$$u(x,t) = e^{-(20(x-t))^2}.$$

Figure 1 shows the node set that will be used in the (x,t)-plane in this first test. The nodes have been placed densely along the expected path of the traveling pulse. Node sets of this kind (with variable density, locally quasi-uniform, and conforming with boundaries) can be generated very rapidly [15]. This node distribution algorithm is of 'moving front type', allowing nodes either to be placed in advance according to some preexisting knowledge of how the PDE solution is evolving in time or placed adaptively (discussed later in Section 3.1.1). The present case features $N = 3,481$ nodes separated by distances between 0.01 and 0.06. The nodes at which IC or BC are provided (along the bottom and left boundaries, respectively) are marked with small squares. At all other node locations, the solution will be calculated by the present RBF-TD algorithm. In contrast to typical high order lattice based discretizations, no special treatment is called for near or at boundaries.

In preparation for advancing (1) forwards in time, we first create a stencil of size n (with $n \ll N$) for each node where the solution is sought. Each stencil includes the node itself and its $n-1$ nearest neighbors *that are located below it in time.* For each of these n stencil entries, we then calculate a set of weights as described next in Section 2.2. The weights for all these different stencils can be computed entirely independently of each other, i.e. their calculation is an 'embarrassingly parallel' task. The stencils are then applied to advance the PDE solution one node at a time. This explicitly solves the PDE with no need for a space-time mesh (i.e. 'grid' with regular node separations in space and time).

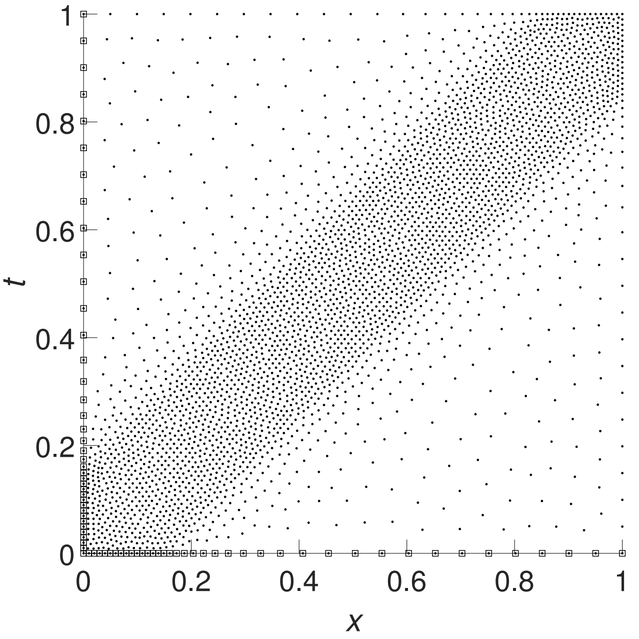


Fig. 1 Node set used over $0 \leq x \leq 1$, $0 \leq t \leq 1$ in the first test problem: one-way scalar wave equation (1). The nodes at which IC or BC are provided are marked by small squares.

2.2 Weight calculation for single time-space stencil

Figure 2 (a) illustrates a typical single stencil with $n = 20$ nodes. The top node, marked by a square, is the location of the node we want to use the stencil to compute a solution value at, once solution values already have been obtained at all the lower node locations. Figure 2 (b) illustrates the weights obtained for the nodes in (a) by the RBF-TD algorithm applied to $\frac{\partial}{\partial t} + \frac{\partial}{\partial x}$. The sign of the weight is indicated by the line type for its surrounding circle and the absolute value is indicated by the *area* inside each circle. When using these weights, the computed solution value at the top node will become greatly dominated by the values below it and to the left in the (x, t)-plane, in excellent agreement with the analytic character of this PDE. We also note that this stencil comes close to being stable even in max-norm since the area of the top circle roughly matches the sum of the other circle areas. We next describe the process to arrive at these weights given node locations and the PDE.

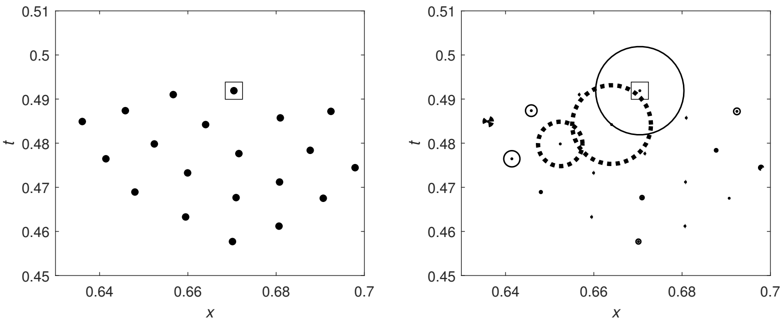


Fig. 2 (a) Example of node layout for a $n = 20$ stencil in the (x, t)-plane, with the top node marked by a square. (b) The associated weights in the case of equation (1). Here the area of each circle indicates the magnitude of the weight used at that node location. Solid circles and dashed circles represent positive and negative weight respectively.

2.2.1 Some background about RBF-FD stencils and their weights

Several monographs provide general surveys of RBFs [11,16,27]. It has furthermore been demonstrated convincingly in the last decade that RBFs can be particularly effective in a 'local' RBF-FD mode generalizing FD-like computing to mesh-free node sets [5,12–14,17,25,28]. For key RBF-FD formulas and summaries of their applications, see in particular [16], Chapter 5. In the present work, we limit ourselves to weight calculations represented by the formula

$$\left[\begin{array}{cccc|ccc} \phi(\|\underline{x}_1-\underline{x}_1\|) & \phi(\|\underline{x}_2-\underline{x}_1\|) & \cdots & \phi(\|\underline{x}_n-\underline{x}_1\|) & 1 & x_1 & t_1 \\ \phi(\|\underline{x}_1-\underline{x}_2\|) & \phi(\|\underline{x}_2-\underline{x}_2\|) & \cdots & \phi(\|\underline{x}_n-\underline{x}_2\|) & 1 & x_2 & t_2 \\ \vdots & \vdots & & \vdots & \vdots & \vdots & \vdots \\ \phi(\|\underline{x}_1-\underline{x}_n\|) & \phi(\|\underline{x}_2-\underline{x}_n\|) & \cdots & \phi(\|\underline{x}_n-\underline{x}_n\|) & 1 & x_n & t_n \\ \hline 1 & 1 & \cdots & 1 & 0 & 0 & 0 \\ x_1 & x_2 & \cdots & x_n & 0 & 0 & 0 \\ t_1 & t_2 & \cdots & t_n & 0 & 0 & 0 \end{array}\right] \left[\begin{array}{c} w_1 \\ w_2 \\ \vdots \\ w_n \\ \hline w_{n+1} \\ \vdots \\ \end{array}\right] = \left[\begin{array}{c} L\phi(\|\underline{x}-\underline{x}_1\|)|_{\underline{x}=\underline{x}_c} \\ L\phi(\|\underline{x}-\underline{x}_2\|)|_{\underline{x}=\underline{x}_c} \\ \vdots \\ L\phi(\|\underline{x}-\underline{x}_n\|)|_{\underline{x}=\underline{x}_c} \\ \hline L1|_{\underline{x}=\underline{x}_c} \\ Lx|_{\underline{x}=\underline{x}_c} \\ Lt|_{\underline{x}=\underline{x}_c} \end{array}\right]. \quad (2)$$

When approximating only spatial derivatives, $\underline{x}$ would in 2-D abbreviate $\underline{x} = (x, y)$. In the present (x, t)-plane, $\underline{x}$ will instead stand for $\underline{x} = (x, t)$, and $\underline{x}_i$ be the location of the i^{th} stencil node. When solving (1), $L = \frac{\partial}{\partial t} + \frac{\partial}{\partial x}$ and $\underline{x}_c$ denotes the location at which we impose that the stencil-wide RBF interpolant obeys the PDE. We note in (2) both column and row vectors $\{1, 1, \cdots, 1\}$, $\{x_1, x_2, \cdots, x_n\}$, $\{t_1, t_2, \cdots, t_n\}$ (more generally, also vectors with higher degree terms). The column vectors enrich the approximating function space (beyond the RBFs alone) and the row vectors ensure a matching formal order of accuracy. The full system matrix is symmetric and, in general, non-singular. The norm $|| \cdot ||$ is the Euclidean 2-norm, and $\phi(r)$ denotes the *radial function*, which we will choose as either a polyharmonic spline (PHS), $\phi(r) = r^m$ (m odd), or as a Gaussian $\phi(r) = e^{-\varepsilon^2 r^2}$. However, in the latter case, the *shape parameter* ε for the large number of varying stencils may become difficult to optimize. Furthermore, it has been shown that polyharmonic splines with polynomials maintain excellent accuracy at boundaries [4, 5], making PHS+poly the ideal choice of RBF. The order of accuracy for a PHS+poly type stencil is typically determined by the order of the polynomial terms attached in (2). Since PHS derive their accuracy through polynomials and initial nodes are placed at a single time, it transpires that no accuracy can be obtained in the time direction initially. Thus, another approach is required to start the calculation. RBF-TD using Gaussians can fill this role until far enough into the computational domain.

2.2.2 L_1 optimization of stencil weights

An analytic solution satisfies its governing PDE at all (x, t)-locations. Like for regular FD stencils applied on a grid, there is some arbitrariness in the choice of location $\underline{x}_c$ at which each stencil is enforced to be exact. We consider for each stencil several $\underline{x}_c$ choices, namely $\underline{x}_c$ is chosen as the midpoints between the latest unresolved node and its closest $p < n$ neighbors. In the style of Figure 2, Figure 3 shows the RBF-FD generated weights with $L = \frac{\partial}{\partial t} + \frac{\partial}{\partial x}$ for nine such choices of $\underline{x}_c$, each marked by a black square. While all the nine weight sets are accurate representations of the PDE, none is suitable for stepping forward in time since stability is doubtful unless the top circle's area dominates the remaining ones. The key to remedying this is to note that any linear combination of these weight sets will also be an accurate weight set. We look for the combination such that, for the resulting weights, $w_1 = 1$ while $\sum_{i=2}^{n} |\, w_i \,|$ is minimized. This creates accurate stencils while being dominated by the "top" weight.

A convenient way to arrange this minimization calculation starts by placing the p (here with $p = 9$) weight vectors side-by-side to form an $n \times p$ matrix W. We next choose a linear combination of the weight vectors such that the right-hand side of equation (3) is achieved. This process can also be expressed as a matrix multiplication with any non-singular $p \times p$ matrix B such that WB takes the form

$$WB = \left[\begin{array}{c|c} 1 & 0 \cdots 0 \\ \hline & \\ -\underline{b} & A \\ & \end{array}\right], \qquad (3)$$

(with zeros in the last $p - 1$ positions in the top row). The desired weight set $\underline{w} = \left[\begin{array}{c} 1 \\ \hline \\ \underline{w}_2 \end{array}\right]$ is now found by determining the vector $\underline{v}$ that minimizes $||A\underline{v} - \underline{b}||_1$ with $\underline{w}_2 = b - A\underline{v}$. For this, we use the MATLAB routine *l1decode_pd.m* from the package *L1-Magic* [6]. These steps produce the single stable weight set shown in Figure 2 (b) from the $p = 9$ different unstable weight sets shown in Figure 3. Algorithm 1 below shows the pseudo-code implementation of the RBF-TD method described thus far.

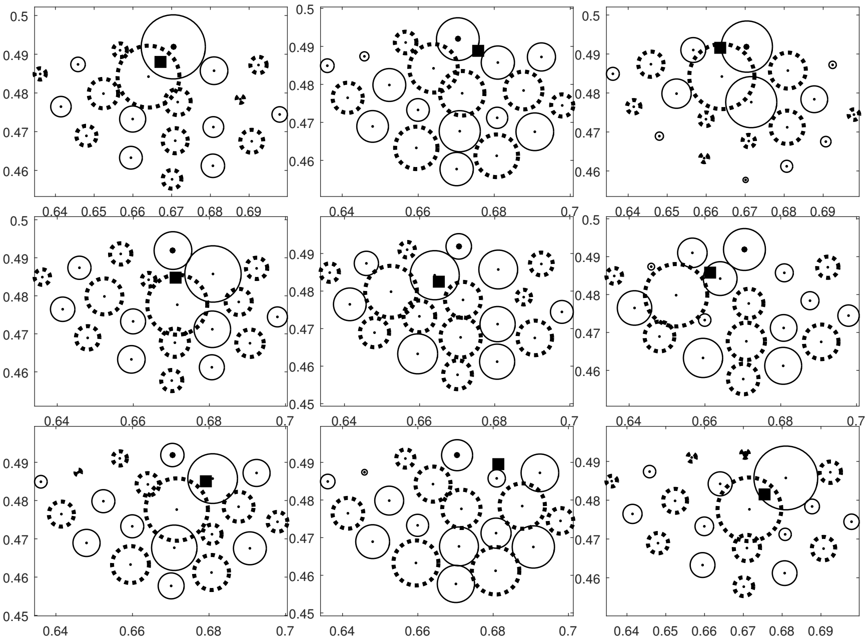


Fig. 3 Nine different sets of weights, obtained by choosing different locations (black squares) at which we require that the RBF-FD (Gaussian) approximation match the PDE. The locations of the squares are half-way between the top node and, in turn, each of its 9 nearest neighbors.

Algorithm 1: RBF-TD

```
while Unsolved nodes exist do
    Find the lowest unresolved node in time.
    if node has IC or BC then
        Apply solution at node.
    else
        Find the $n$ closest neighbors below the unresolved node.
        Choose $p < n$ center points $\underline{x}_c$.
        Compute $W$ using (2) for each center point with Gaussians or PHS and
          desired polynomials.
        Form the $WB$ matrix and collect $A$ and $\underline{b}$.
        Find the weight vector $\underline{w}$ that minimizes $||A\underline{w}_2 - \underline{b}||$.
        Apply $\underline{w}$ to find solution at unresolved node.
    end
end
```

2.3 Time stepping of the 1-D wave equation test case

Stepping forward in time over the $N = 3,481$ nodes shown in Figure 1 as described above produces the solution seen from two perspectives in Figure 4 [2]. Gaussians ($\varepsilon = 20$) were used to initialize the solution a distance of 0.05 (an arbitrarily suitable choice) into the time domain, followed by PHS with fourth degree polynomials for the remaining distance of 0.95. The error reaches a maximum magnitude of just less than 0.025. In order to put this accuracy in some perspective, we compare against standard MOL discretizations, using a space step of $h = 0.01$ (thus in space with the same node separation in x as the densest one appearing anywhere in the scattered node discretization).

If we combine centered fourth order FD (FD4) in space with third order Adams-Bashforth (AB3) in time, von Neumann analysis shows the stability condition to be $\lambda = \frac{k}{h} \leq \frac{24}{25}\sqrt{\frac{8\sqrt{6}-3}{55}} \approx 0.52734$. Obeying this by choosing $k = 0.5h = 0.005$ leads to a grid with $N = 20,301$ space-time grid points, and gives a maximum error of 0.17, i.e. about 7 times larger error in spite of using nearly 6 times as many nodes. Changing from AB3 to AB4 in time helps very little. The stability restriction then becomes $\lambda = \frac{k}{h} \leq \frac{8}{375}\sqrt{13\sqrt{6}-3} \approx 0.31335$. Using $k = 2h/7 \approx 0.29h = 0.0029$ gives a grid of $N = 35,451$ points and an error of 0.15 (i.e. a very moderate error reduction in spite of many more grid points). In contrast to the RBF-TD approach, both these schemes need special modifications at all boundaries (apart from at the top one, $t = 1$), since their stencils would otherwise extend outside the computational domain. For Runge-Kutta methods the stability conditions are comparable. Accounting for their internal stages, they are $\lambda < 0.420$ for RK3 and $\lambda < 0.515$ for RK4.

[2] A code for producing this figure can be downloaded from https://github.com/DylanAbrahamsen/RBF-TD

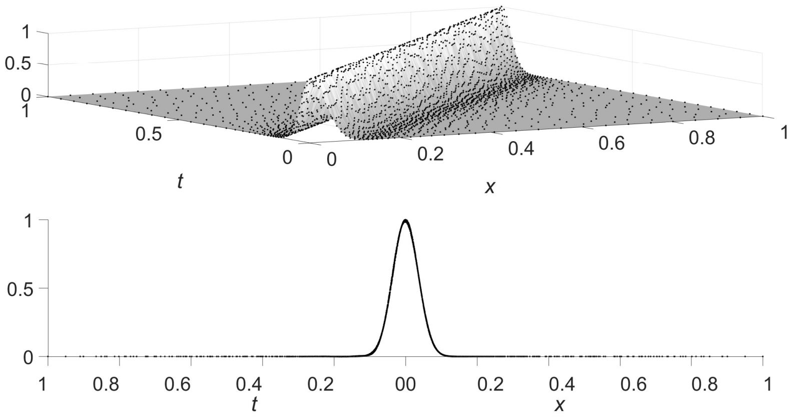


Fig. 4 Two perspectives of the RBF-TD solution to the one-way transport equation detailed in Section 2.3 with $n = 30$ and $\phi(r) = r^5$ + fourth degree polynomials.

Figure (5) demonstrates the convergence rate using different order polynomials on globally quasi-uniform node sets. No local refinement was used for this convergence rate testing to simplify the interpretation of the result. It is clear that the convergence rates match the polynomial powers included.

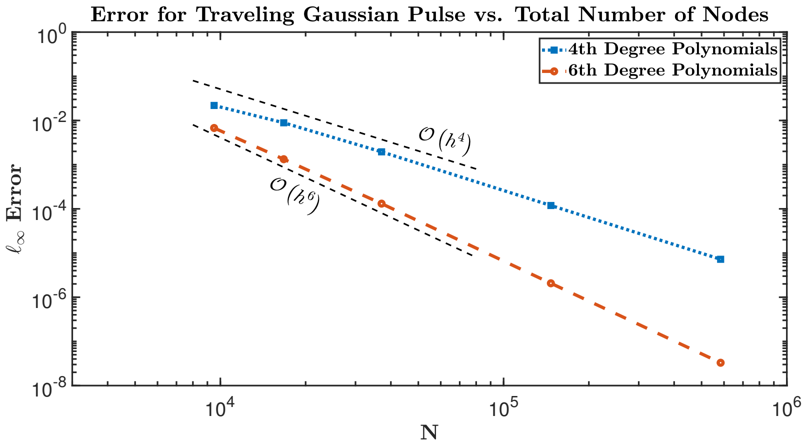


Fig. 5 RBF-TD ℓ_∞ error to the test problem in Section 2.3 for varying N using $\phi(r) = r^5$ + fourth degree polynomials with $n = 30$ and $\phi(r) = r^5$ + sixth degree polynomials with $n = 40$. The dashed lines represent fourth and sixth order convergence rates recalling that $N = \mathcal{O}\left(\frac{1}{h^2}\right)$.

2.4 Comparison between AB3-FD4 and RBF-TD on a Cartesian grid

The AB3-FD4 scheme for (1) has a stencil shape which can in the (x, t)-plane be illustrated as[3]

$$\begin{array}{ccccccccc} & & & & \square & & & & \\ & & & & | & & & & \\ \square & - & \square & - & \square & - & \square & - & \square \\ \square & - & \square & - & 0 & - & \square & - & \square \\ \square & - & \square & - & 0 & - & \square & - & \square \end{array} \tag{4}$$

With the two entries associated with the time step being assigned weights 1 and -1, respectively, the 3×5 block of entries will be given by the rank 1 matrix

$$A_{\mathrm{MOL}} = -\lambda \begin{bmatrix} \frac{23}{12} \\ -\frac{4}{3} \\ \frac{5}{12} \end{bmatrix} \left[\, \frac{1}{12} \; -\frac{4}{3} \; 0 \; \frac{4}{3} \; -\frac{1}{12} \,\right], \tag{5}$$

where $\lambda = k/h$. The top row of subplots in Figure 6 display the weights that this approximation to (1) produces for $\lambda = 0.5,\ 0.75,\ 1.0$, respectively. The restriction of the A_{MOL}-matrix to be of rank 1 and also to be left-right anti-symmetric forces a number of weights in the AB3-FD4 stencils to become quite large for increasing λ. The corresponding set of weights produced by the RBF-TD approach with Gaussians ($\varepsilon = 0.08$) on the same node set, bottom row of subplots (based on using 7 values for $\underline{x}_c$, as shown by black squares), reflect more directly the PDE's characteristic direction (in all the six subplots marked by a dashed line segment).

2.4.1 Action of the stencils on a Fourier mode

The Fourier modes $e^{i\omega x}$ that can be present on a grid of spacing h satisfy $-\frac{\pi}{h} \leq \omega \leq \frac{\pi}{h}$. When advancing the solution to (1) forward by a time distance k, each mode should (analytically) get multiplied by $e^{-i\omega k}$, i.e. by $e^{-i\xi}$ with $\xi = \omega k$. The small circles in each subplot of Figure 7 display this factor $e^{-i\xi}$ for 40 equispaced values of ξ across its range of $|\xi| \leq \pi$. Both of the numerical schemes under consideration have 4 levels in time, and thus, for each ξ-value, 3 roots in their characteristic equation. One of these three roots should ideally match each circular marker in turn, whereas the remaining two roots should be 'harmlessly' located well inside the unit circle. These roots are shown as dots for each subplot.

[3] We focus here on AB-FD4 schemes (rather than on RK-FD4 schemes, with internal stages) since these can be displayed as space-time stencils.

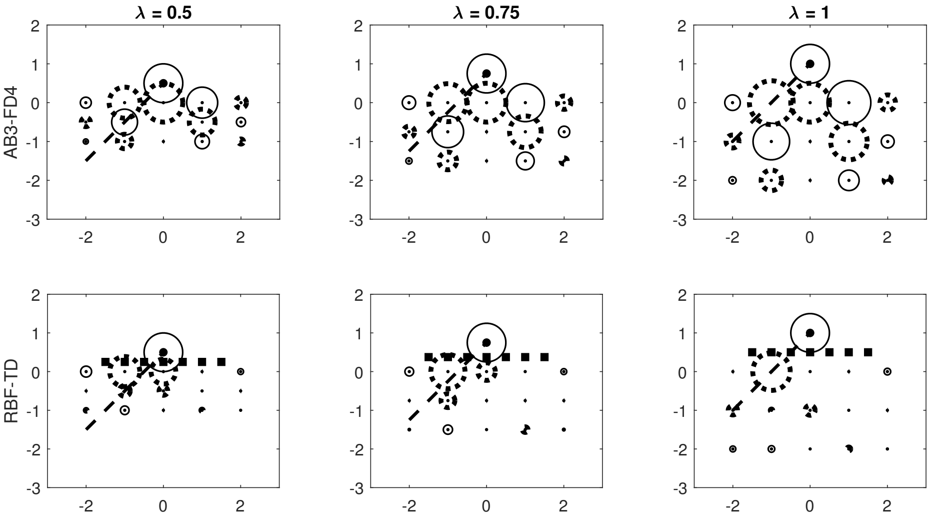


Fig. 6 Top row of subplots: Weights obtained for the the AB3-FD4 discretization of (1) for three choices of $\lambda = 0.5,\ ,0.75,\ 1.0$, respectively. The dashed line marks the characteristic direction of the analytic solution. Bottom row of subplots: The equivalent data when the weights are instead calculated by the RBF-TD approach. All stencil weights are normalized with the "top" weight being one.

In the three AB3-FD4 cases, we see at first good agreement for the lowest modes. However, as λ increases, fewer modes approximate the analytic solution well and already at $\lambda \approx 0.52734$ (as noted above, using von Neumann analysis) stability is lost as spurious roots exit the unit circle. In the RBF-TD cases, the accuracy remains high also for larger λ-values, and stability is not lost until just before λ reaches one. A very small amount of hyperviscosity [18] is well suited for 'nudging' roots to the inside the unit circle if they are marginally outside due to some inherent 'noise' in the L_1 minimization (in the first two cases, less than 10^{-4} in the positions of the roots). If the governing equation is not purely hyperbolic, but has some low level of natural dissipation, or if nodes are scattered rather than purely lattice based, this will not be an issue.

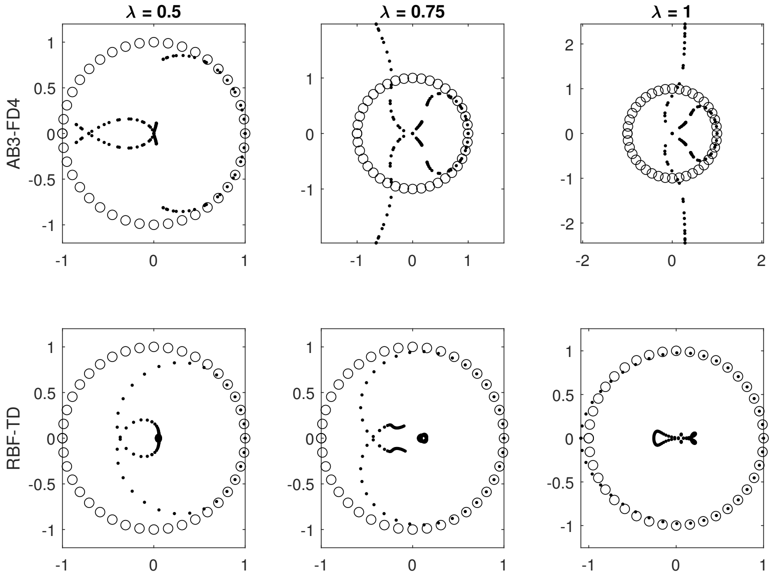


Fig. 7 The layout of the subplots is the same as for Figure 6. The ‘actions’ of the schemes are shown when applied to the Fourier modes $e^{i\omega x}$ as described in Section 2.4.1.

2.4.2 Heuristic discussion about stability restrictions

The previous Section 2.4.1 gave some indication that stability conditions can become more relaxed when using the RBF-TD approach compared to grid-based MOL. Key reasons for this include:

(i) MOL approximations come with severe constraints (such as the A_{MOL}-matrix in (5) being rank one and anti-symmetric) whereas the RBF-TD approach naturally finds weights that more directly reflect the ‘character’ of the PDE. As a consequence of this, noted above and seen in Figure 7, spurious roots grow much larger for MOL.

(ii) An explicit MOL scheme on a lattice (compare (4) with Figure 2 (a)) is an extreme case of the evaluation node being particularly far ahead of any neighboring node. When using RBF-TD stencils on scattered nodes in the space-time domain there are typically nearby nodes (with data) at an almost equally advanced time.

(iii) A lattice-based MOL scheme is unstable if just a single spatial Fourier mode has a growth factor larger than one in magnitude (since exactly the same growth will then repeat for every subsequent time step). With scattered time-space nodes, every stencil is different in this regard. If some modes grow for one particular stencil, chances are that they will decay for other stencils. Overall spurious growth becomes less likely.

2.5 Demonstration problem: 1-D heat equation

2.5.1 Test problem

We consider here the PDE

$$\frac{\partial u}{\partial t} = \alpha \frac{\partial^2 u}{\partial x^2} \tag{6}$$

over the domain $x \in [0,1]$, $t \in [0,1]$, with initial and boundary conditions that match the analytic solution

$$u(x,t) = \frac{1}{\sqrt{4\pi\alpha(t+\tau)}}\, e^{-\left(x-\frac{1}{2}\right)^2/(4\alpha(t+\tau))},$$

choosing $\alpha = \frac{1}{30}$ and $\tau = \frac{1}{50}$.

2.5.2 RBF-TD solution

Figure 8 shows the solution, advancing in time from the left far boundary to the right as calculated with the RBF-TD approach using the node set illustrated in Figure 9 (a). This node set is refined near $t = 0$, $x = \frac{1}{2}$ (motivated by the initial steepness of the solution) and also around $t = \frac{1}{2}$, $x = \frac{1}{2}$ (in order to to further test its numerical stability). This calculation arbitrarily used GA-type RBFs, with shape parameter

$$\varepsilon = 0.2 \,/\, \{\text{distance from top stencil node to the node nearest to it}\}$$

to demonstrate that both GA and PHS with polynomials are viable options. Each stencil contained $n = 15$ nodes and was based on L_1-minimization combining $p = 7$ different stencils (centered at the half-way locations between the top node and its p nearest neighbors below it).

The quasi-uniform node set has locally the same spacing in both the $x-$ and $t-$directions. To assess if its stability restriction is comparable with grid-based MOL approaches, we compare against the 'routine' explicit time stepping scheme for the heat equation

$$\frac{u(x,t+k) - u(x,t)}{k} = \alpha\, \frac{u(x-h,t) - 2u(x,t) + u(x+h,t)}{h^2}.$$

This is stable if $\lambda = \frac{k}{h^2} \leq \frac{1}{2\alpha}$. If we use $k = h$, this condition translates to $\frac{2\alpha}{h} \leq 1$. This constant value 1 forms the bottom surface in Figure 9 (b). In the RBF-TD calculation, $k \approx h$, this quantity $2\alpha/\{\text{distance to nearest node}\}$ is varying as shown on the top surface. We see that this RBF-TD implementation used steps that exceeded what would be possible with the FD scheme by factors that vary between 2 and 8. The stability situation is again greatly improved over traditional explicit FD / MOL schemes.

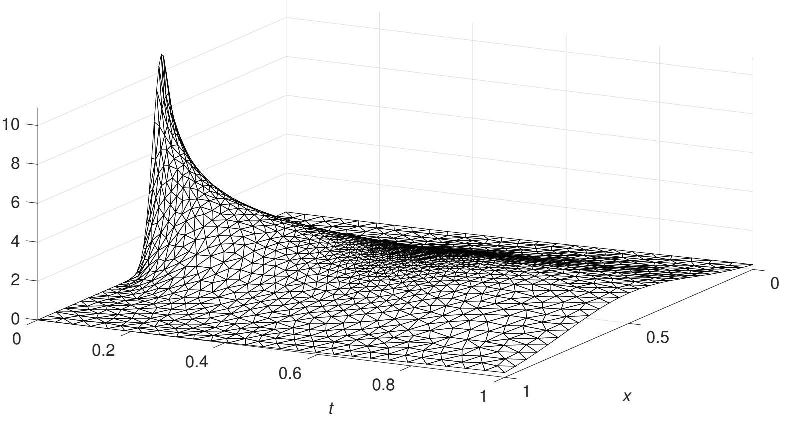
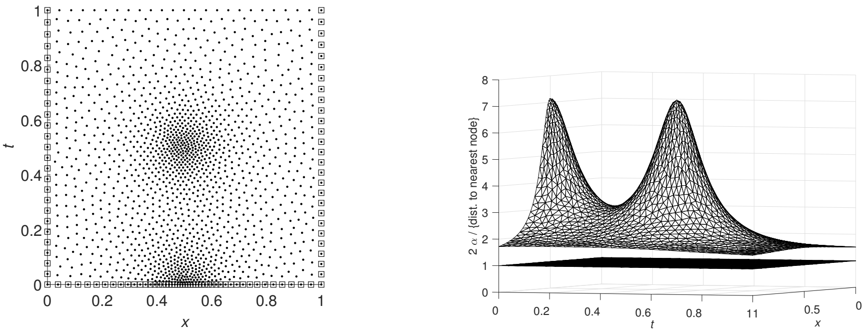


Fig. 8 Solution to the heat equation test problem described in Section 2.5.1, as obtained by the RBF-TD approach described in Section 2.5.2.



(a) Distribution of $N = 1,533$ nodes.

(b) Stability comparison.

Fig. 9 (a) Node set used in test calculation of the heat equation and (b) values for $\frac{2\alpha}{h}$, as described in Section 2.5.2, upper limit in the FD case (bottom plane) vs. actually used in the RBF-FD case (top surface)

3 Additional test problems

3.1 Transport equation with variable velocities

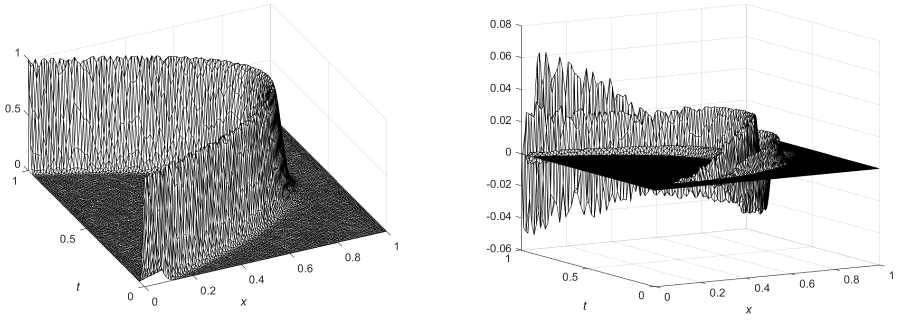
We consider next the transport equation with variable wave speed in time,

$$u_t + \alpha(t)u_x = 0 \tag{7}$$

where $\alpha(t) = 6\left(\frac{1}{2} - t\right)$. In the case that $\alpha(t)$ is large, waves will travel long distances in short time. This could potentially cause CFL conditions to be violated. To better account for this, it becomes natural to correspondingly stretch the RBFs horizontally, i.e. to replace the Euclidean distance definition $||\underline{x}_1 - \underline{x}_2|| = \sqrt{(x_1 - x_2)^2 + (t_1 - t_2)^2}$ by a Mahalanobis counterpart,

$$||\underline{x}_1 - \underline{x}_2|| = \sqrt{(x_1 - x_2)^2 + \left(\frac{t_1 - t_2}{\sigma}\right)^2}. \tag{8}$$

The RBF is accordingly stretched using $r = \sqrt{x^2 + \left(\frac{t}{\sigma}\right)^2}$. The parameter σ is in our present test case chosen as $\frac{1}{\alpha}$ when $\alpha > 1$, otherwise as $\sigma = 1$ (then resulting in the standard Euclidean distance).



(a) Numerical solution

(b) Error

Fig. 10 (a) Numerical RBF-TD solution of (7) using stencil size $n = 30$ and PHS+poly of the form $\phi(r) = r^5$ + fourth degree polynomials on $N = 9,405$ nodes. (b) Difference between numerical and analytic solution.

The same Gaussian pulse from Section 2.1 is used as an initial condition. Figure 10 shows that the error initially increases roughly linearly in time, as expected due to the trailing waves. The amplitude of the main pulse has decayed by roughly 2% at $t = 1$. Below is the augmented pseudo-code for RBF-TD with high wave speed.

Algorithm 2: RBF-TD for high wave speed

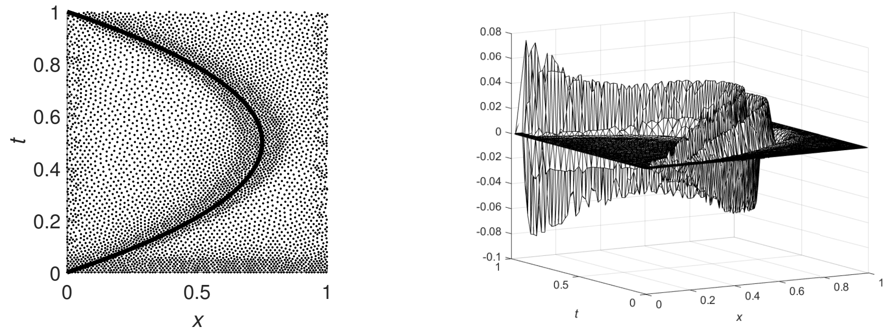
```
while Unsolved nodes exist do
    Find the lowest unresolved node in time.
    if node has IC or BC then
        Apply solution at node.
    else
        Check the wave speed $\alpha$ at node location.
        if $\alpha \leq 1$ then
            Find the $n$ closest neighbors below the unresolved node using
              standard Euclidean norm.
        else
            Find the $n$ closest neighbors below the unresolved node using (8) with
              $\sigma = \frac{1}{\alpha}$.
        end
        Choose $p < n$ center points $\underline{x}_c$.
        Compute $W$ using (2) for each center point with Gaussians or PHS and
          desired polynomials.
        Form the $WB$ matrix and collect $A$ and $\underline{b}$.
        Find the weight vector $\underline{w}$ that minimizes $||A\underline{w_2} - \underline{b}||$.
        Apply $\underline{w}$ to find solution at unresolved node.
    end
end
```

3.1.1 Adaptive node generation

Placing densely packed nodes in regions of high activity and sparely otherwise takes full advantage of the geometric flexibility of RBF-TD. However, creating node sets for different problems can be time consuming and can result in non-optimal node placement. RBF-TD can easily be combined with adaptive node generation, for example based on [15]. Rather than generating the node set prior to running RBF-TD, the process runs alongside RBF-TD. A variety of stratagies are possible for dynamic node placements. Just to test the concept, we here choose an exclusion radius at each node that is determined by a 'curvature' measure

$$\omega \left| \frac{u_x(x_0, t_0)}{\max\limits_x u_x(x, t_0)} \right| + (1 - \omega) \left| \frac{u_{xx}(x_0, t_0)}{\max\limits_x u_{xx}(x, t_0)} \right| \tag{9}$$

at the latest node in time located at (x_0, t_0). Here $\omega \in [0, 1]$ is some weighting between the first and second derivative in space. The derivative values are found by extrapolating across the spatial domain at the latest time t_0. Finally, the exclusion radius is scaled from a prescribed minimum and maximum value based on the nodes relative 'curvature'. Figure 11 illustrates the concept. We have here generated nodes with no effort to (i) use local refinement during the initialization, or to (ii) correct for edge-related issues. Nevertheless, the errors seen in Figure 11(b) are comparable to those in 10(b) in spite of only using $N = 5,084$ nodes (versus $N = 9,405$ nodes).



(a) Numerical solution

(b) Error

Fig. 11 (a) Adaptive node set generated to solve (7) with N=5,084 and $\omega = \frac{1}{2}$. The line represents the peak of the moving pulse. (b) Difference between numerical and analytic solution.

Figure 11 demonstrates the proof of concept for adaptively creating node sets alongside RBF-TD. Also, a slight exclusion radius averaging is used to smooth the node density and can create node placements that favor the trailing edge of the wave. The method currently is being developed further to help eliminate phenomea like the edge effects seen above.

3.2 Viscid Burger's equation

Our last test case is the viscid Burger's equation,

$$u_t + uu_x = \nu u_{xx}. \tag{10}$$

The nonlinear term uu_x requires special attention when generating accurate stencils. Multiple accurate stencils that depend on u at center points $\underline{x}_c$ are desired. Due to this, each stencil requires some iteration. First $u|_{\underline{x_c}}$ is approximated using RBFs at nodes with known values. Next the weights for the differential operator $L = \frac{\partial}{\partial t} + \left(u|_{\underline{x_c}}\right) \frac{\partial}{\partial x} - \nu \frac{\partial^2}{\partial x^2}$ are found using (2) utilizing the closest neighbors with known values of u. Use these weights to approximate $u|_{\underline{x_c}}$ using the previous known values of u. This process can be repeated until desired accuracy is reached. Typically two or three iterations are sufficient. Finally, the weights for L are found that now include the unresolved node. This process generates accurate stencils for the differential operator $\frac{\partial}{\partial t} + u\frac{\partial}{\partial x} - \nu\frac{\partial^2}{\partial x^2}$ at the p chosen center points. The process of Section 2.2.2 is then used on these p stencils to create a single stable stencil. Below are the corresponding pseudo-codes.

Algorithm 3: RBF-TD for viscid Burger's

```
while unsolved nodes exist do
    Find the lowest unresolved node in time.
    if node has IC or BC then
        Apply solution at node.
    else
        Find the $n$ closest neighbors below the unresolved node.
        Choose $p < n$ center points $\underline{x}_c$.
        Compute $W$ using algorithm 4.
        Form the $WB$ matrix and collect $A$ and $\underline{b}$.
        Find the weight vector $\underline{w}$ that minimizes $||A\underline{w_2} - \underline{b}||$.
        Apply $\underline{w}$ to find solution $u$ at unresolved node.
    end
end
```

Algorithm 4: Stencil Iteration

```
Use RBFs at nodes with known values of $u$ to approximate $u|_{\underline{x_c}}$ for each center
  point.
while $u|_{\underline{x_c}}$ is below desired accuracy do
    Find the weights $W_{\text{temp}}$ that approximate $L$ at each center point using (2) on
      the closest neighbors with known values of $u$ with Gaussians or PHS and
      polynomials.
    Use $W_{\text{temp}}$ to approximate $u|_{\underline{x_c}}$ for each center point.
end
Find the weights $W$ that approximate $L$ at each center point using (2) on the
  closest neighbors that also include the unresolved node with Gaussians or PHS
  and polynomials.
```

To test this approach an analytic solution is chosen as

$$u(x,t) = \frac{\left[\alpha + \beta + (\beta - \alpha)\, e^{\zeta}\right]}{1 + e^{\zeta}} \tag{11}$$

where $\zeta = \frac{\alpha}{\nu}(x - \beta t - \eta)$, $\alpha = 0.4$, $\beta = 0.6$, and $\eta = 0.125$ taken from [21]. The approach in [21] is to use RBF-FD in space and Crank-Nicolson in time to solve (10). The reported ℓ_∞ errors were 8.65E-5, 1.25E-2, and 1.25E-2 when using three different RBF choices of MQ, GA, and IQ respectively, on roughly $N = 100,000$ nodes.

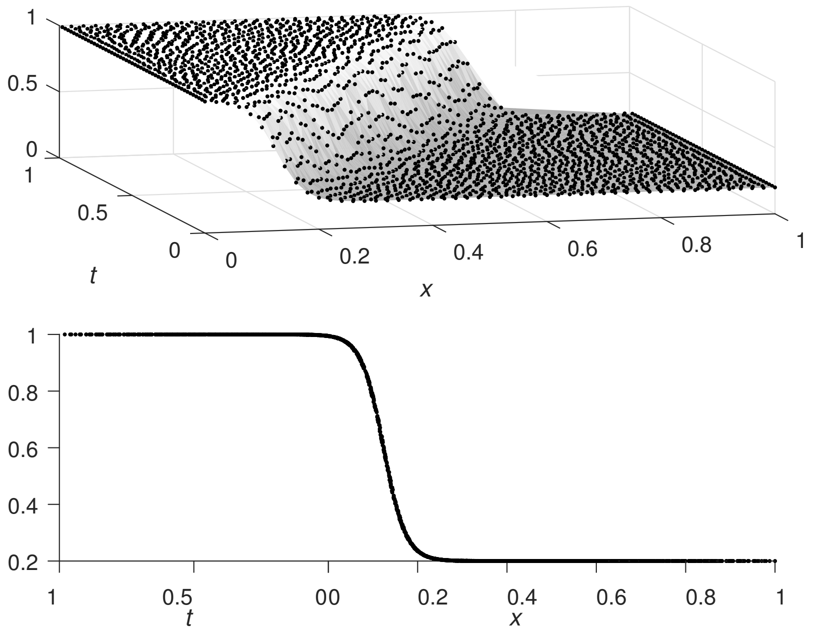


Fig. 12 Solution to the viscid Burger's test problem described above as obtained by the RBF-TD approach using r^5 with fourth degree polynomials on $N = 2,483$ globally quasi-uniform nodes. The bottom plot shows the full surface in the top plot viewed along the direction of travel in the (x,t)-plane.

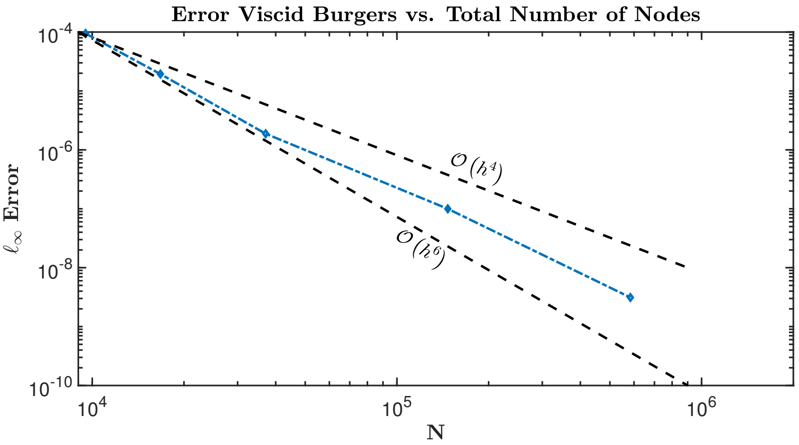


Fig. 13 Convergence plot for solving equation (10) using RBF-TD with r^5 and sixth degree polynomials with initial conditions given by equation (11) on globally quasi-uniform nodes. The dashed lines represent fourth and sixth order convergence.

Figure 13 demonstrates the convergence rate of the present iterative approach to RBF-TD for the same test case as described above for [21]. RBF-TD has reached an error of 1.94E-5 on $N = 16,740$ nodes (with no local space-time node refinement) converging between fourth and sixth order as N is increased further. More iterations of Algorithm 4 may be needed to maintain higher order. No local refinement was used for convergence rate testing, to simplify the interpretation of the result. As just noted the number of nodes, N, can be drastically lowered by using fewer nodes in regions where the solution is less steep.

4 Conclusions

This study introduced the idea of utilizing the geometric flexibility of RBF-FD to place locally quasi-uniform nodes in the space-time domain. This has been demonstrated to be very advantageous in several regards:

- Larger temporal spacing of nodes may be used (even on Cartesian lattices compared with MOL).
- Geometric freedom to place nodes densely in areas of high interest and sparsely in regions of low interest, thereby greatly reducing the total number of nodes needed for a prescribed accuracy.
- The generated stencils accurately ‘pick up’ the natural characteristics of the problem.

These highly useful features of RBF-TD also provide ideas for further improvements. Some logical next steps are

- Finding a way to speed up the $\mathcal{L}_1$ minimization process while calculating stencil weights (training a neural network is one option to consider).
- Further improve dynamic node placement.
- Application to higher space dimensions plus time.
- Tests on nonlinear wave equations (such as interacting soliton waves).

References

1. Abedi, R., Petracovici, B., Haber, R.: A space-time discontinuous Galerkin method for linearized elastodynamics with element-wise momentum balance. Computer Methods in Applied Mechanics and Engineering **195**(25-28), 3247–3273 (2006)
2. Almquist, M., Mehlin, M.: Multilevel local time-stepping methods of Runge-Kutta-type for wave equations. SIAM Journal on Scientific Computing **39**(5), A2020–A2048 (2017)
3. Bayona, V.: On the similarities between moving least squares and RBF+poly for interpolation and derivative approximation. J. Scientific Computing (to appear)
4. Bayona, V., Flyer, N., Fornberg, B.: On the role of polynomials in RBF-FD approximations: III. Behavior near domain boundaries. J. Comput. Phys. **380**, 378–399 (2019)
5. Bayona, V., Flyer, N., Fornberg, B., Barnett, G.: On the role of polynomials in RBF-FD approximations: II. Numerical solution of elliptic PDEs. J. Comput. Phys **332**, 257–273 (2017)
6. Candes, E.: L1-magic. https://statweb.stanford.edu/ candes/l1magic/

7. Demirel, A., Niegemann, J., Busch, K., Hochbruck, M.: Efficient multiple time-stepping algorithms of higher order. J. Comput. Phys **285**, 133–148 (2015)
8. Descombes, S., Lanteri, S., Moya, L.: Locally implicit time integration strategies in a discontinuous Galerkin method for Maxwell's equations. J. Comput. Phys **56**(1), 190–218 (2013)
9. Diaz, J., Grote, M.: Multi-level explicit local time-stepping methods for second-order wave equations. Computer methods in applied mechanics and engineering **291**, 240–265 (2015)
10. Driscoll, T.A., Heryudono, A.R.: Adaptive residual subsampling methods for radial basis function interpolation and collocation problems. Computers & Mathematics with Applications **53**(6), 927–939 (2007)
11. Fasshauer, G.: Meshfree Approximation Methods with MATLAB. Interdisciplinary Mathematical Sciences - Vol. 6. World Scientific Publishers, Singapore (2007)
12. Flyer, N., Barnett, G., Wicker, L.: Enhancing finite differences with radial basis functions: Experiments on the Navier-Stokes equations. J. Comput. Phys **316**, 39–62 (2016)
13. Flyer, N., Fornberg, B., Barnett, G., Bayona, V.: On the role of polynomials in RBF-FD approximations: I. Interpolation and accuracy. J. Comput. Phys. **321**, 21–38 (2016)
14. Flyer, N., Lehto, E., Blaise, S., Wright, G., St-Cyr, A.: A guide to RBF-generated finite differences for nonlinear transport: Shallow water simulations on a sphere. J. Comput. Phys **231**, 4078–4095 (2012)
15. Fornberg, B., Flyer, N.: Fast generation of 2-D node distributions for mesh-free PDE discretizations. Comput. Math. Appl. **69**, 531–544 (2015)
16. Fornberg, B., Flyer, N.: A Primer on Radial Basis Functions with Applications to the Geosciences. SIAM, Philadelphia (2015)
17. Fornberg, B., Flyer, N.: Solving PDEs with radial basis functions. Acta Numerica **24**, 215–258 (2015)
18. Fornberg, B., Lehto, E.: Stabilization of RBF-generated finite difference methods for convective PDEs. J. Comput. Phys. **230**, 2270–2285 (2011)
19. Gopalakrishnan, J., Schöberl, J., Wintersteiger, C.: Mapped tent pitching schemes for hyperbolic systems. SIAM Journal on Scientific Computing **39**(6), B1043–B1063 (2017)
20. Hamaidi, M., Naji, A., Charafi, A.: Space–time localized radial basis function collocation method for solving parabolic and hyperbolic equations. Engineering Analysis with Boundary Elements **67**, 152–163 (2016)
21. Haq, S., Siraj-Ul-Islam, Uddin, M.: A mesh-free method for the numerical solution of the KDV-Burgers equation. Applied Mathematical Modelling **33**, 3442–3449 (2008)
22. Li, Z., Mao, X.Z.: Global multiquadric collocation method for groundwater contaminant source identification. Environmental Modelling & Software **26**(12), 1611–1621 (2011)
23. Li, Z., Mao, X.Z., Li, T.S., Zhang, S.: Estimation of river pollution source using the space-time radial basis collocation method. Advances in Water Resources **88**, 68–79 (2016)
24. Netuzhylov, H., Zilian, A.: Space–time meshfree collocation method: Methodology and application to initial-boundary value problems. International Journal for Numerical Methods in Engineering **80**(3), 355–380 (2009)
25. Shan, Y., Shu, C., Lu, Z.: Application of local MQ-DQ method to solve 3D incompressible viscous flows with curved boundary. Comp. Mod. Eng. & Sci. **25**, 99–113 (2008)
26. Uddin, M., Ali, H.: The space–time kernel-based numerical method for Burgers' equations. Mathematics **6**(10), 212 (2018)
27. Wendland, H.: Scattered Data Approximation, *Cambridge Monographs on Applied and Computational Mathematics*, vol. 17. Cambridge University Press, Cambridge (2005)
28. Wright, G., Fornberg, B.: Scattered node compact finite difference-type formulas generated from radial basis functions. J. Comput. Phys. **212**, 99–123 (2006)